Model-Based Recognition and Extraction of Information from Chart Images

Ales Mishchenko[1], Natalia Vassilieva[2]
[1]CEA, Grenoble
France
[2]HP Labs
St. Petersburg
Russia
ales.mishchenko@mail.ru, nvassilieva@hp.com

ABSTRACT: *Charts are widely used in technical and business documents as a graphical representation of numerical and qualitative data. We present a model-based method to automatically extract data carried by charts and convert them to XML format, thus making these data available for indexing, querying and analysis by common methods of textual data management. The proposed method includes several steps: 1) chart detection, 2) model-based classification by chart type and extraction of graphical components, 3) detection and recognition of textual components, 4) extraction of semantic relations between graphics and text. For testing purpose, a benchmark set was created with the XML/SWF Chart tool. By comparing the recovered data and the original data used for chart generation, we are able to evaluate our information extraction algorithm and confirm its validity. We also extensively tested each step of the proposed algorithm against existing approaches. Model-based classification showed high accuracy, comparable to those of the best supervised learning methods. The proposed text detection algorithm leads to significant improvement in text recognition rate (up to 20 times better).*

Keywords: Image recognition, Image classification, Image understanding, Information extraction, Text detection, Chart detection and classification

Received: 11 March 2011, Revised 8 April 2011, Accepted 12 April 2011

1. Introduction

Chart images are often used as a powerful representation tool in technical and business literature. They often contain important information, which is not duplicated in other data formats and thus missed by text-based document analysis tools and not exploited for indexing and knowledge extraction purposes. A chart usually represents a table of names, values and dependencies of different variables (see Figure 1). At the same time, this visual representation of tabular data is rich, complex and comes in many different forms. The automatic extraction of tabular data from all the variety of its chart representations still remains largely unresolved and challenging task.

We present a model-based method that automatically extracts the data from the charts and converts them to XML. Our work makes the chart data available for indexing, querying and analysis by common methods of textual data management. The method extracts the data points with values and labels by analyzing the graphical and textual components of a chart. The proposed method consists of the following steps.

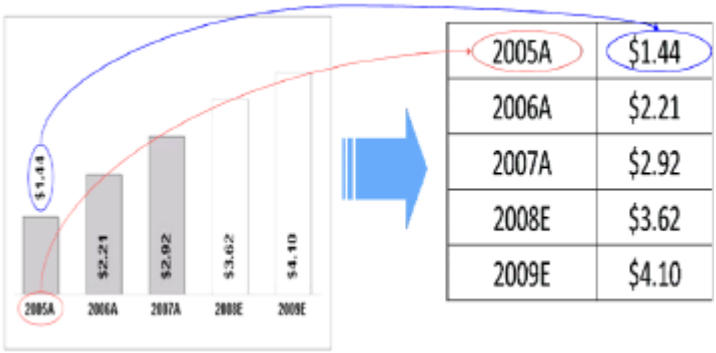


Figure 1. A chart and its data in tabular form

- Preprocessing: an input image is classified as chart/non-chart image; in case of chart image the location and size of a chart are estimated.
- Analysis of graphical chart components: graphical components such as rectangles for a bar chart or sectors for a pie chart are detected; chart type is identified using model-based classification.
- Analysis of textual chart components: text stings such as legends, axes names, etc. are detected on a chart; optical character recognition (OCR) is further applied for text recognition.
- Data extraction: graphical data components are measured (angle for a pie sector, height for a column), correlation between graphical and textual components is discovered.

To validate our approach, we have created a benchmark set with the XML/SWF Chart tool. This tool automatically creates charts from XML data. We compared the original data with extracted data to evaluate the overall solution. We also validated every step of the proposed solution separately.

The paper is organized as follows. After the review of related work in the subsequent section 2, our method is described in the section 3. Experimental results are presented in the section 4. Each of the sections 2, 3 and 4 consists from subsections, corresponding to different steps of our method. The section 5 concludes the paper.

2. Related work

Information extraction from a digital chart image includes the following steps: chart detection and classification; text detection and recognition; data extraction including measurement of chart data components and understanding chart measures. Let us discuss the previous work related to every step in subsections below.

2.1 Chart detection and classification

Chart detection might be considered as an image classification problem with two classes: chart and non-chart. Common methods developed to solve generic image classification problem can be applied here. Traditionally, texture, color and edge features, as well as their combinations are used in classification of images [1 – 3]. Many approaches have been applied to image classification task, such as support vector machines (SVM) [4], blocks statistics and hidden Markov models (HMM) [5], subwindows extraction and supervised learning [6], multiple-instance learning (MIL) [7], clustering of image features [8] etc. A particular attention to the task of detecting the charts of a specific type (line plots) among natural images was paid in [9].

Another classification task in the context of chart image understanding problem is chart classification by its type. It is an important step in chart image understanding, as it drives all the following processes, including data extraction and semantic interpretation. Classification of images by chart type belongs to the intensive research area of recognizing special types of graphics. The survey of diagrams recognition and treatment can be found in [10]. The chart is a type of diagrams, which visualizes the table of names, values and dependencies of variables. The majority of existing approaches to chart image classification and understanding were developed within the scenario of image features extraction followed by a feature-based comparison. The latter varies from comparison of a test image to training images (Learning-based approach, such as [9]) to

comparison of a test image to abstract models, representing particular classes (Model-based approach, such as [11, 12]).

Another categorization of approaches is by the type of extracted features. According to it, all methods can be divided into the following groups: low-level, middle-level and high-level. The former tends to calculate low-level features and rely on vast representative training sets. A typical example is utilizing Hough transform for feature extraction and classification [13 – 15]. This approach was proved to work well with bar and column charts [15] but can be ineffective with large amount of segments and does not provide the connection between image features and chart components, making further data interpretation step difficult. Other approaches, utilizing higher-level features, rely on optimality of their feature set. An example of high-level model-based chart classification is discussed in [11, 12]. Authors detect basic shapes in the image and check if they satisfy a certain set of constraints corresponding to a certain chart type. Examples of middle-level approaches are Multiple-Instance Learning [16] and approaches which are based on shape and spatial relationships of chart primitives [17]. Authors extract invariant to translation, rotation and scaling image features and use a multistage approach to compare feature sets (such as MIL in [16] or Pyramid Match, followed by SVM in [17]) to classify images.

2.2 Text detection and recognition

The textual data in charts includes axes names, tick mark labels, legends, captions, notes and carries important information for indexing and knowledge extraction. The traditional steps of a text recognition algorithm are binarization (local, global or adaptive) [18, 19], text detection and localization (region-based, edge-based or texture-based) [20], text line detection [18, 21], extraction and enhancement, recognition [21]. Text recognition techniques are applied mainly to document images and to a lesser degree to natural images. The majority of algorithms exploit the properties of a particular image class.

For document images the following observations are usually true: all text lines are parallel; text size is uniform; it is easy to separate the background; big text blocks of several text lines (paragraphs of text) are present [18, 21, 22].

For natural images the following observations are usually true: there are few text strings in the image; text strings are small; background is complicated; text strings have lower contrast compared to the text in document images [20, 23].

Chart images belong to none of these classes. They are somewhere between (text strings are small, but have a good contrast) and thus require an individual approach. A work by Fletcher and Kasturi is the closest one to our solution [24]. They proposed an algorithm for text string separation for engineering drawings and diagrams. This algorithm could be applied to solve text detection problem in chart images, but it implies a lot of requirements for an input document. It should be binary and of high resolution. There are constraints to acceptable font size and intercharacter distance. Our approach contains similar steps, but with the different implementation, and doesn't have these requirements on input images.

2.3 Data extraction

The measurement of chart data components can be fully-automatic or interactive. The former includes reported results for some particular cases of information extraction. Information extraction from images of linear and quadratic 2D curves was studied in [25]. The algorithm, presented in [9], is capable of detecting special data points and solid line curves together with axes, labels and data symbols, leading to 2D plot data understanding and information extraction.

The graphics constraint grammars were used for describing and analyzing diagrams in Document Understanding System, developed in [26] and used for a wide corpus of biomedical literature.

The interactive approach was exploited by many different systems for both text [27] and geometrical structure [28] extraction. During the recent years, many interactive software tools for information extraction from chart images were developed, such as VistaMetrix (skillcrest.com), Dagra (BlueLeafSoftware.com), Plot Digitizer (plotdigitizer.sourceforge.net), qplot (qplot.com), Engauge Digitizer (digitizer.sourceforge.net), Data Thief (datathief.org), and many others. However, the main focus of these tools was measuring and scaling the charts of a predefined type. It is interactive, requires human involvement and often works only as curve matching to the points, specified by the user.

Automatic information extraction from chart images of unknown type is a more difficult task. Our solution to this problem is a model-based approach, adaptable to various chart types through the addition of new models. The similar solution (addition of new rules) was suggested in [29] for map understanding.

3. The proposed method

In this section we describe our approach to the problem of chart understanding and information extraction from charts. Currently we support five commonly used chart types: column, bar, pie, line and area charts. We assume that a chart depicts a table of names, values and dependencies of different variables. Our goal is to recover these tabular data and convert them to the XML format.

We assume that a chart consists of graphical and textual components. Graphical components include chart data components (columns in case of column charts, sectors in case of pie charts, curves in case of line charts, etc) and chart metadata components (axes lines, tick marks, legend marks, gridlines). Textual components include all text strings in a chart: chart title, axis labels and units, data labels, legend text.

Our solution uses a model-based approach to chart image understanding. It constitutes from modeling the graphical components (both data and metadata) with abstract models - offline modeling stage, and matching an input image with these models - online processing stage. The online processing stage includes the following steps (see Figure 2):

- Preprocessing: chart detection and estimation of chart size and location.
- Analysis of graphical chart components: chart classification and detection of graphical components.
- Analysis of textual chart components: text detection and recognition.
- Data extraction.

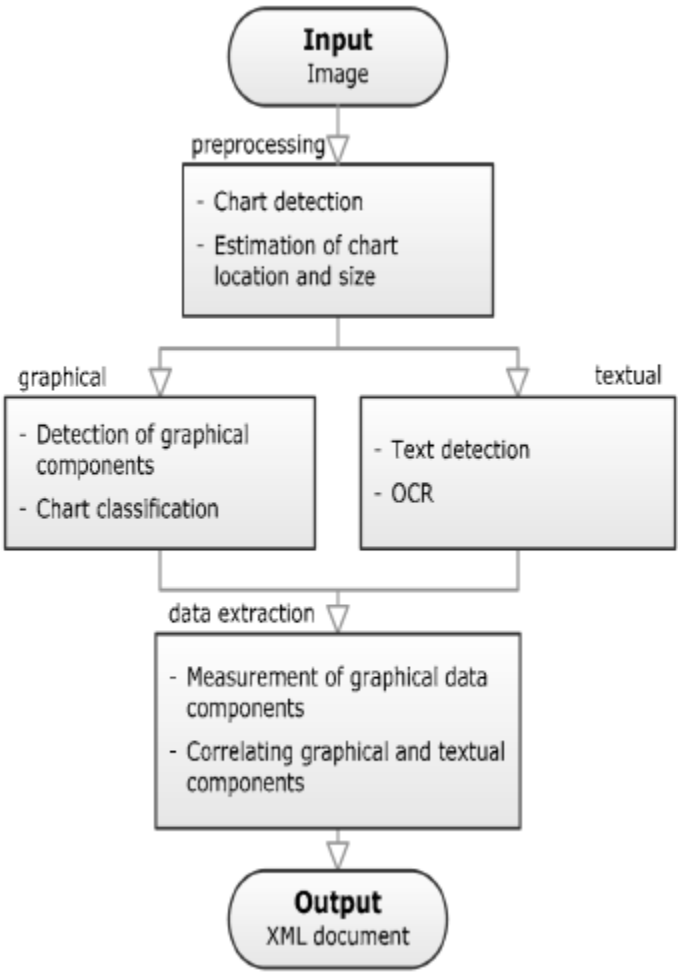


Figure 2. The flowchart of the chart image understanding and data extraction process

3.1. The offline modeling of graphical components

We exploit widely used "edge structure models" ("edge models" in short) representing geometrical and topological structure of image edges [30, 31]. The edge model is a spatial structure, consisting of line segments and circular/elliptical arcs. It reflects the typical shape and spatial layout of edges for a given graphical component.

Additionally, our models provide the rules how to estimate parameters of graphical components such as color, and how to get a value represented by a data component (how to measure a data component). The models of metadata components contain additionally the rules how to exploit the recognized text components, such as numbers for axes model, names of variables for legend model, etc. Every edge model is provided with a set of goodness-of-fit criteria to measure the discrepancy between an observed image edge set and the edges expected under the model. Let us illustrate the principles of model creation with the following examples.

Pie: The model for 2D/3D pie chart is a set of line segments - radii, and circular/elliptical arcs with the following constraints: all radii converge to the same point (the center of a pie); an opposite end of every radius is connected to the endpoints of two neighboring radii by arc segments; all radii are equal (2D pie) or the length of radii satisfies elliptical equation (3D pie); arcs are parts of the same circle (2D pie) or ellipse (3D pie); the center of this circle/ellipse coincides with the center of radii.

The goodness-of-fit criteria are: variation of lengths of radii; variation in curvature of arcs; measure of fit of arcs to a single circle/ellipse; distance between the center of circle/ellipse and the center of radii.

Graphical data components to be measured are pie sectors: the angle for each pie sector is calculated, and the percentage of each angle versus 360 degree is treated as the value for each pie sector.

Column/Bar: The model for column/bar charts is a set of line segments, which form rectangles with sides parallel to coordinate axes. Size and inter-location constraints are: every rectangle has a side (base side), lying on the same coordinate axis (the x-axis for column charts and the y-axis for bar charts). The lengths of these sides are equal.

The goodness-of-fit criteria are: variation of width of rectangles; variation of alignment: orientation of sides, base side location; quality of axes detection: lengths, perpendicularity, and alignment with rectangles.

Graphical data components to be measured are rectangles: the length of a non-base side is treated as the value for each column/bar.

Area: The model for areas is a closed polyline, with two segments, parallel to the y-axis. "Bottom" ends of these segments are connected by a single segment, parallel to the x-axis, "top" ends are connected by a chain of line segments, representing a polyline function in respect to the horizontal axis.

The goodness-of-fit criteria are: completeness of a polyline and uniqueness of polyline function values; quality of axes detection: lengths, perpendicularity, and alignment with area segments.

Graphical data components to be measured are the lengths of perpendiculars from all points of the polyline to the x-axis.

Line: The model for lines is the above mentioned polyline alone. The goodness-of-fit criteria and rules for extracting the values are the same as for area charts.

2D Cartesian axes (metadata component): The model for 2D Cartesian axes consists of the following: two line segments (or a set of small segments, forming them); "ticks" line segments (or "ticks" text areas); set of text areas corresponded to "scales" and "axes names". The constrains are: orthogonality of "axes"; size of "axes", comparable to the size of the chart; specific text alignment (uniform spacing, similar size, orientation and color) for tick marks text and "scales" text areas, specific alignment of "axes", "ticks", "scales" and "axes names" with respect to each other.

The goodness-of-fit criteria are: closeness of an angle between the axes to a right angle; variation in text alignment among tick's text areas.

Graphical metadata components to be measured are the distances from tick marks to the point of axes intersection.

The colors and values of data and metadata components are set into correspondence with each other during the data extraction step.

3.2 Preprocessing: chart detection

This step categorizes the given image as a chart or non-chart image, based on the analysis of its 3D color histogram. The presence of large uniform color regions in the chart image results in a chart-specific histogram features. Each uniform region in a chart (pie sector, column, area, bar) corresponds to a peak in a chart image histogram. The highest peak in a histogram usually corresponds to background. Histogram, consisting from a number of distinct peaks is typical for a chart image. Real-world images, such as photographs, on the contrary, have usually smoothly-varying histograms. Figure 3 below shows typical histograms for chart and natural scene images. In the bottom row of Figure 3 (a chart image and its histogram) several peaks are clearly resolved in a histogram: the background peak (rightmost), the biggest pie-sector (leftmost), and two other pie-sectors (in the middle). In this case all major peaks are easy to resolve.

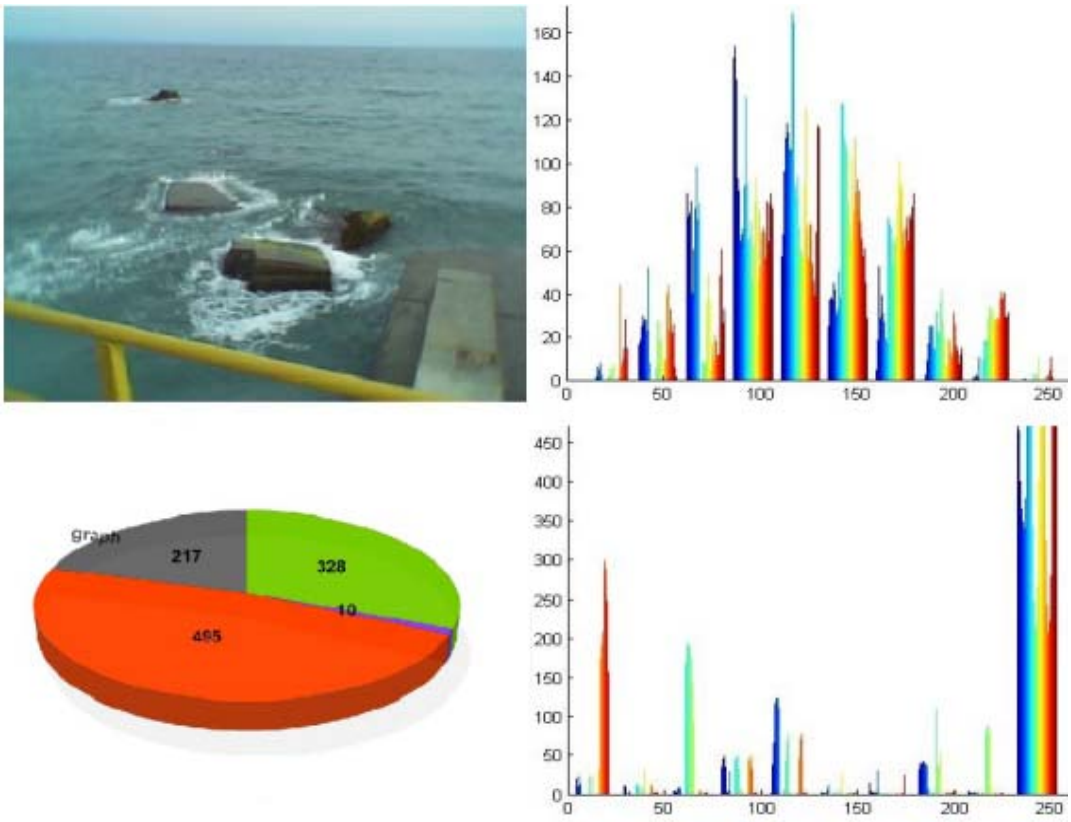


Figure 3. Histograms of a chart image and a natural scene image. R, G and B components are shown separately, grouped into 10 bins. Axes are component intensity (from 0 to 255) and number of pixels in each bin

The algorithm of chart detection and subsequent estimation of chart location and size is summarized in Figure 4.

We use a 3D color histogram with spatial constraints to identify the presence of a chart in the image and estimate size, color and location of chart elements. We define a membership function $W:(x,y)\rightarrow[0,1]$ as a function of distance from the estimated center location of a chart to a given pixel. $W:(x,y)=$ 1for pixels at the center of a chart, whereas $W:(x,y)=0$ for pixels outside of the estimated chart area. The value for a bin k of a histogram with spatial constraints for an image $I(x,y)$for bin k is calculated as follows.

$$H_k = \left\{ \begin{array}{ll} \sum_{x,y} W(x,y), I(x,y) = color_k \\ 0, \quad otherwise \end{array} \right\} \tag{1}$$

Thus image pixels outside the chart area do not influence the estimation of size and location of chart data components.

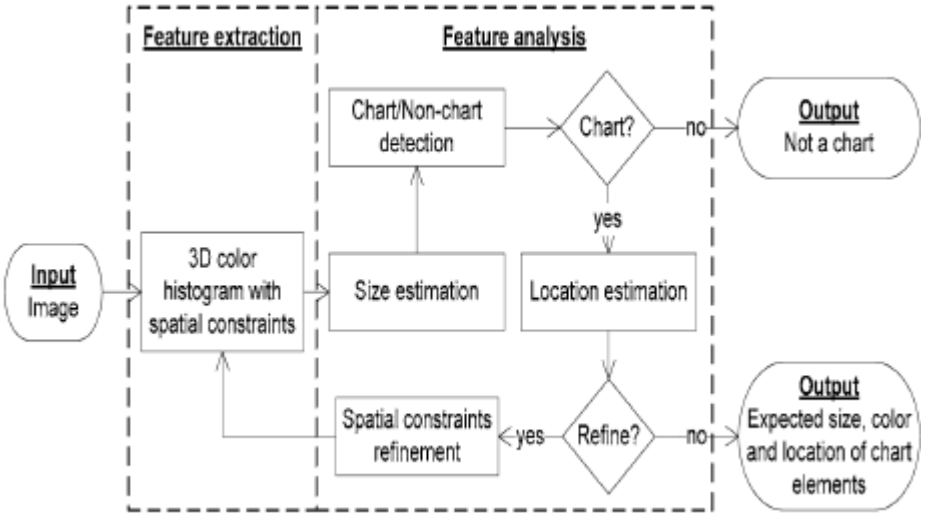


Figure 4. Chart detection and subsequent estimation of chart location and size

The process of chart location and size estimation is iterative. At the first iteration, when there is no estimation on chart size and location available, $W(x,y) = 1$ for all pixels of the image. Every iteration refines the histogram calculation domain based on the estimated chart location. The histogram structure in terms of peaks and valleys is analyzed in order to estimate size, color and location of chart components. In a chart image, major peaks in the histogram are usually clearly resolved. The peaks typically correspond to the following elements: background, chart data components, and labeling (legends, axes, tick marks, etc). The peak values provide size estimation for the corresponding chart elements. The results of this step (estimations on size and location of graphical components) are taken into account during the next step.

The details of the iterative process are as follows. The results of size estimation at every iteration allow taking a decision about further processing of an image. The image is considered to be a chart image if the following parameters exceed the experimentally optimized thresholds:

- **Background presence:** Distance between the highest histogram peak (background) and the neighboring bin values.

- **Chart elements presence:** Quantity and height of histogram peaks, exceeding 10% of highest histogram peak (large uniform color regions).

- **Degree of segmentation:** Distance in 3D-color space between average image colors, corresponding to histogram peaks.

If the histogram does not contain any clearly resolved peaks (even in the quantized color space), or the size of major peaks does not meet the set of predefined heuristic-based constraints, an image is considered as a non-chart image and the processing of this image is stopped. Otherwise, spatial distribution of colors, corresponding to different peaks, is analyzed in order to estimate the location of data components. Then, the overall chart location and size (coordinates of the center, width and height of a chart) is estimated based on locations and sizes of chart data components. The estimation of chart size and location is used to calculate the values of a membership function, which leads to the next iteration and more precise chart location as well as estimation of chart elements sizes.

Two major problems in chart detection and the following chart location and size estimation are the presence of other major regions of uniform color and non-uniformity of colors of the chart. Examples of the latter problem are chart coloring schemes which includes shadows or gradients. In this case, reducing the number of histogram bins (color space quantization) allows of determining the histogram peaks and thus performing size estimation for chart elements for the majority of chart images. An example of the former problem is a presence of other major regions of uniform color (such as thick boarders, filled text areas, etc). These regions may lead to a wrong estimation of the size of the chart data components. However, if these regions are smaller than the chart data components, their influence is corrected during the next iteration using chart location information (spatial constraints refinement step, shown in Figure 4).

3.2.3 Chart classification and detection of graphical components

This step employs a model-based approach to classify a given chart into one of the predefined types (column, bar, pie, line and area).

First, edge set is extracted from an input image by performing edge detection [32], thinning [33], linking [34] and vectorization [35]. Edges contained within the estimated chart area and having size approximately equal to the estimated size of chart data components are included into the edge set.

Second, the obtained edge set is matched with the designed models of considered chart types (pie, column, bar, line, area), which describes chart data components for every particular chart type, and to the models of metadata components (axes, legends, titles, annotations). Matching the image edge set to these models is a process of competitive classification of edges based on their geometrical features into sub-sets corresponding to the given chart data/metadata models. Then the voting procedure is performed based on the values of goodness-of-fit measures resulting in a classification decision. The voting for specific chart type and metadata components influences each other. For example, axes are obligatory for area and line charts, optional for columns and bars charts, undesirable for pie charts. Another example is that legend colors should correspond to chart colors. Chart models' matching is a single-winner competition. Metadata models' matching is not: on the contrary, all metadata components, associated with a detected chart type should be found in the image. The matching of the edge set to chart type and metadata models is summarized in a Figure 5.

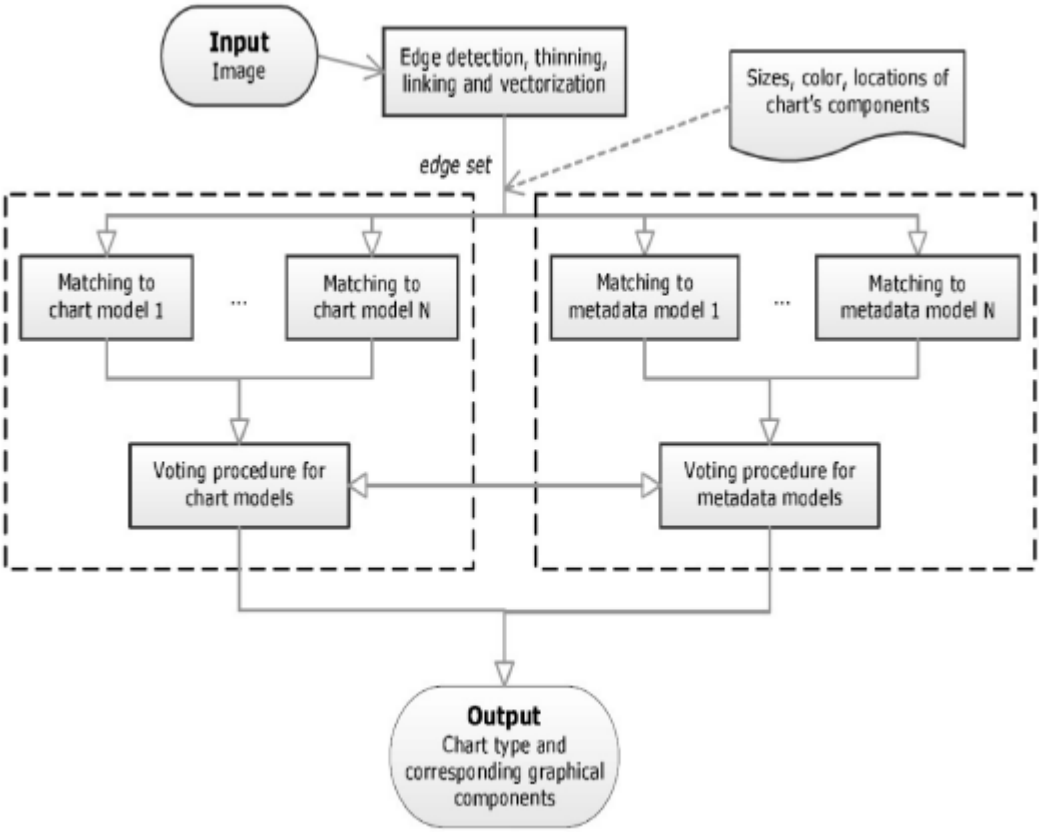


Figure 5. Model-matching for chart type and graphical components

3.3 Text detection and recognition

At the text detection and recognition we process all text areas in the image (titles, axes names, tick mark labels, legends, captions, annotations, etc), which carry important information for indexing and knowledge extraction. Common optical character recognition (OCR) engines (Tesseract, ABBYY FineReader, MODI by Microsoft) are tuned to detect the paragraphs of text and fail to detect and recognize isolated text strings in chart images due to their small size and often non-horizontal alignment. Only properly detected, cropped and deskewed text strings can be recognized by OCR engines. We propose a new algorithm for detecting text

in chart images, regardless of orientation, font size and style. Text detection is performed in a three-step approach. First, potential characters are detected by connected component labeling and strong non-character components are filtered out. Second, the Hough Transform is used to detect text lines. And third, inter-character distance and character size is analyzed for final identification of text strings. The algorithm is described in greater detail in [36].

The detected text strings after being cropped and deskewed are passed to Tesseract OCR engine, increasing significantly its recognition rate.

3.4 Data extraction

Data extraction is a process of assignment the names and numerical values to the detected chart data components. The matched chart type model provides a set of chart components and determines what exactly should be measured (such as horizontal, vertical, angular or other dimension of chart elements). The detected metadata components provide names for these measurements (through legends, titles and annotations), as well as provide scaling for these measurements (through axes scaling or numerical annotations for pie sectors). The result reveals the initial information, which was visualized by a chart. This information is recovered in tabular format showing names, values and dependencies of different variables, and stored as an XML document.

4. Experimental results

We conducted a set of experiments to evaluate our chart data extraction method and its components: chart detection, chart classification, text detection and recognition, data extraction.

For chart detection we calculated the accuracy of the proposed chart/non-chart classification solution.

For chart classification we compared the proposed model-based classifier with common supervised learning methods. To the best of our knowledge, no comparative study for model-based and learning-based approaches in the context of chart image classification task was done before. We used the publicly available WEKA Data Mining Software [37] in our experiments. This software provides implementations of many state-of-the-art classifiers, which we used for comparison with our model-based classifier.

For evaluation of the proposed text detection method we compared the text recognition rates with and without the preprocessing step of text detection.

To evaluate the data extraction method and the proposed solution overall we compared the recovered data and the original data used for chart generation.

Experiments were conducted with datasets of 980 and 300 chart images generated using XML/SWF Charts tool (http://www.maani.us/xml_charts), and a dataset of 353 natural scene images. The larger chart dataset was used for evaluation of model-based chart classifier and text detection algorithm. The smaller chart dataset was used for the evaluation of the data extraction stage and involved manual assessment. Data for charts was randomly generated. Images were collected as screenshots resulting in blur edges and small noise components due to anti-aliasing. Non-chart images were randomly selected real-world photographs and were used for evaluation of chart detection in combination with subset of 473 chart images from a larger chart dataset. The experimental results are reported below.

We used category-specific accuracy and average accuracy metrics for evaluating classification decisions. They are calculated as follows.

$$Category-specificAccuracy: A_c = \frac{TP_c}{TP_c + FN_c}$$

$$Average\ Accuracy: A_{avg} = \frac{1}{m}\sum_{c \in C} A_c$$

where TP_C - the number of True Positives, FN_C - the number of False Negatives with respect to a specific category $c \in C \equiv \{C_1,...,C_m\}$

4.1 Chart detection results

		Predicted value		Num. of images
		Chart	Non-Chart	
Actual value	Chart	423 (**89.4%**)	50	47
	Non-chart	1	352 (**99.7%**)	353

Table 1. Confusion matrix for the chart detection step

Chart detection experiments were conducted with a dataset consisted of 473 chart images and 353 natural scene images. The experimental results are shown in Table 1. The observed average accuracy is 94.6%. The category-specific accuracy for charts is 89.4%. The remaining 10.6% of charts were misclassified as natural scene images due to shadows or gradients in chart coloring schema and difficulties with resolving the histogram peaks. The category-specific accuracy for natural scene images is 99.7%. The only misclassified image has close-to-uniformly colored regions.

4.2 Chart classification results

We evaluated our chart classification step with a dataset of 980 generated chart images. The proposed model-based classifier was compared to more than 40 types of learning-based classifiers, implemented in WEKA [37]. We conducted experiments with the following methods using their WEKA implementation (categorization and names of the methods are provided according to the WEKA package).

bayes: BayesNet, NaiveBayes, NaiveBayesUpdatable.

functions: Logistic, RBFNetwork, SimpleLogistic, SMO.

meta: AttributeSelectedClissifier (J48 as classifier, CfsSubsetEval for attribute selection), Bagging (REPTree as classifier), ClassificationViaRegression (M5P as classifier), Decorate (J48 as classifier), FilteredClassifier (J48 as classifier), LogitBoost (DecisionStump as classifier), MultiClassClassifier (Logistic as classifier).

misc: HyperPipes, VFI.

rules: ConjunctiveRule, DecisionTable, JRip, NNge, OneR, PART, Ridor.

trees: DecisionStump, J48, LMT, NBTree, REPTree, RandomForest.

Default WEKA parameters were used for all methods in the experiments.

Every image from the dataset was preprocessed, its edge set was extracted and vectorized as described in section 3.2. This edge set was used by the model-based classifier to predict the type of an image. The same edge set was used to extract features for the learning-based methods. The feature set included statistics on edges, similar to those used in [16]. Line and arc segments from the edge set were grouped by their size (resulting in groups of edges of similar size) and by their connectivity (resulting in groups of connected edges). The feature set consisted of statistical measures of sizes, shapes, inter locations and connections within each group. Parameters of grouping were optimized to provide the highest accuracy for the given dataset.

The experimental results of the model-based classification method are summarized in Table 2. The average accuracy for model-based chart classification is 90% according to the experimental results. The maximum category-specific accuracy is obtained for line chart images at the level of 100%. The lowest category-specific accuracy is obtained for pie charts (74% for 2D pies and 84% for 3D pies). Other papers on chart classification report close results (although using a different chart image database): 83% average accuracy reported in [17] (with maximum of 90% for column images), 76% average accuracy reported in [16].

Learning-based classifiers were trained on a subset of the available dataset. 33% of the dataset was used to train the classifiers and the rest was used for testing. Learning-based methods demonstrated performance, which is comparable with the results of the proposed model-based solution. The best results among the learning-based methods were shown by LogitBoost, Filtered, BayesNet and NNge (names of the methods are provided according to WEKA package). These methods showed accuracy greater than 90% for pies, columns/bars, lines, and accuracy of 69% - 77% for areas.

		Predicted value				Num. of images
		line	area	column	pie	
Actual value	line	191 (**100%**)	0	0	0	191
	area	9	191 (**96%**)	0	0	200
	column	2	5	191 (**96%**)	0	198
	2D pie	34	17	0	142 (**74%**)	193
	3D pie	3	27	0	168 (**85%**)	198

Table 2. Confusion matrix for the model-based classification

Learning-based classifiers were trained on a subset of the available dataset. 33% of the dataset was used to train the classifiers and the rest was used for testing. Learning-based methods demonstrated performance, which is comparable with the results of the proposed model-based solution. The best results among the learning-based methods were shown by LogitBoost, Filtered, BayesNet and NNge (names of the methods are provided according to WEKA package). These methods showed accuracy greater than 90% for pies, columns/bars, lines, and accuracy of 69%-77% for areas.

4.3 Text detection and recognition results

The performance of the proposed text detection algorithm was evaluated with a test set of 980 images. Every test image has a corresponding xml mark-up file, which contains the information about every text string in the image such as its location, orientation, text and visual appearance properties (font family, font size, background and foreground colors, and others). The test set contains in total about 14000 text strings. The test set was processed with Tesseract OCR engine with and without prior text detection by the proposed algorithm (we will further refer to these runs as *Prep* and *NoPrep* respectively).

To evaluate the performance of the algorithm, we use location recognition rate (LRR), location precision rate (LPR), text recognition rate (TRR) and the text precision rate (TPR) that are computed on a ground truth basis as $LRR = N_{loc} / N_G$,$LPR = N_{loc} / N_F$, $TRR = N_{txt} / N_G$, $TPR = N_{txt} / N_G$, where N_{loc} is the number of correctly localized text strings, N_{txt} is the number of correctly recognized text strings, N_G is the true total number of text strings in the test set, N_F is the total number of detected text strings. The text strings are considered to be correctly localized if the intersection with one of the true text strings covers more than 90% of its area; correctly recognized - when the Levenshtein distance between the true text and the recognized one is less than 3. The Table 3 presents the experimental results.

	LRR	LPR	TRR	TPR
Prep	79.0%	88.7%	45.0%	44.6%
NoPrep	33.6%	84.0%	2.5%	2.9%

Table 3. Text detection and recognition results

The experimental results shows that using the proposed algorithm as a preprocessing step before OCR, we get up to 20 times better text recognition rate, and 15 times higher text recognition precision.

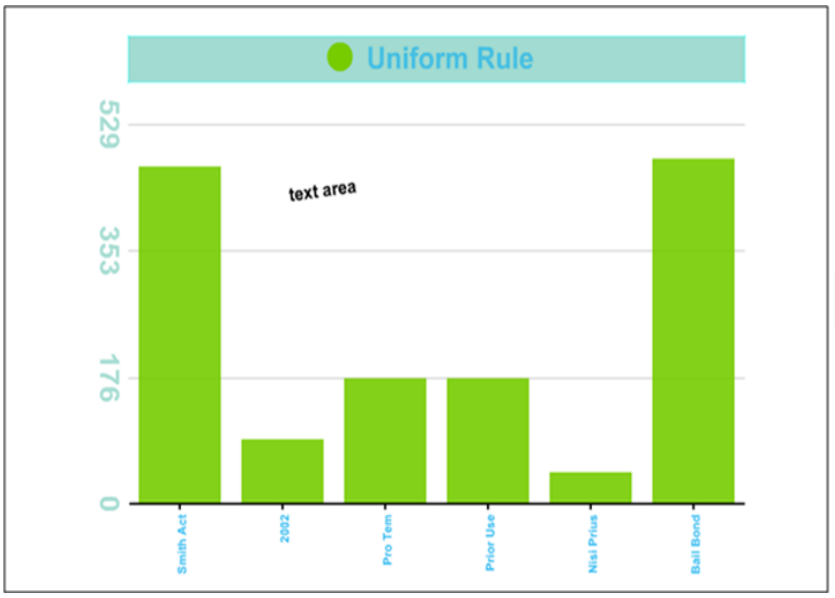
4.4 Data extraction results

Data extraction accuracy shows the overall result of the proposed solution. To evaluate the data extraction step we compared the original data which was used to generate charts in a test dataset and the data extracted by our algorithm.

Table 4 illustrates the results of the proposed information extraction pipeline. Data extraction accuracy reflects accumulation of errors from all stages of the method (chart detection, detection of graphical components and chart classification, text detection and recognition, data extraction). Chart detection, chart classification and text recognition columns shows the results obtained at the previous steps of the proposed pipeline and discussed above.

Graphical components recognition was evaluated as a number of correctly recognized variables (x-values) on a chart. For example, the set of recognized variables for a chart presented on Figure 6 is {Smith Act, 2002, ProTcm, Prior Use, Nm Prius, Bond}. The accuracy of graphical components recognition is calculated as a ratio of correctly recognized variables among all existing variables in a chart.

Data extraction accuracy was calculated as a ratio of correctly recognized data points (variables with their values, (x,y)-pairs) in a chart. Figure 6 illustrates a chart and a table of data points automatically extracted from this chart. The values (y-points) for the variables extracted at graphical components recognition step were estimated using the rules for chart data components measurement, which are described in section 3.1. The comparison of the extracted data points with the original data points was done manually. We accepted small variations of value between the original and the extracted data points.



Legends	Uniform Rule
Smith Act	422.4222
2002	41.0889
Pro Tcm	129.0889
Prior Use	129.0889
Nm Prius	20.7185
Bond	424.0519

Figure 6. An example of test image and data extraction result

To the best of our knowledge, no data extraction method, able to process all the considered types of charts was developed before. However, there exist several methods, built for data extraction from particular types of chart images. In [25], for example, a specific type plots (2D linear and quadratic curves) was used for data extraction. This narrow-class algorithm was capable to extract information with much higher, 86% accuracy. But, in difference to them, our method is able to process a wider class of input images.

5. Conclusion

In this paper we propose an algorithm for information extraction from charts images. The proposed algorithm includes a method for model-based chart recognition and classification and a method for text detection in chart images, followed by the extraction of information from digital chart images. The proposed method does not need supervised learning and rely on models of charts and their graphical components, which are matched to the image.

As it was shown by tests with 980 images database, the chart classification accuracy is comparable to the best of learning-based approaches. The experiments also proved the competitiveness of the proposed text detection algorithm compared to the common OCR engine. The overall information extraction stage was tested with 300 chart images. The method showed reasonable accuracy of reconstruction of initial information, visualized by the chart image.

The main advantage of the proposed approach is that it relies on high-level features that are useful for further high-level interpretation of charts and extraction of numerical data. It performs chart image classification and graphical components detection in one-pass. Another advantage is that it does not need supervised learning. Due to variability of design among the charts of the same type, training samples may contradict each other and the training set might never be complete.

The main drawback of the proposed method, common for all model-based approaches, is necessity of human involvement for model design. However, in case of charts, adding a new model to the classifier may require just a small amount of operator's work. In the current implementation all chart models (column, bar, pie, line and area), as well as models of metadata (axes, legends, titles, annotations) are hard-coded. In the future a simple description language can be proposed for modeling, so the operator will be able to extend the functionality of the proposed framework to the types of charts without changing the code.

The further increase of the data extraction accuracy might be achieved by tuning the connected components labeling step and by applying the Hough Transform locally for text detection. Another direction for improvement is detecting the tick marks and correlating them with the textual components on the chart. It will allow refining the measurement step of chart data components.

References

[1] Haralick, R., Dinstein, I., Shanmugam, K. (1973). Textural features for image classification. *IEEE Trans. Syst. Man Cybernet*, 3, 610–621.

[2] Szummer, M., Picard, R. (1998). Indoor/outdoor image classification. *International Workshop on Content-Based Access of Image and Video Databases*, p. 42–51.

[1] Vailaya, A., Jain, A., Zhang, H. (1998). On image classification: city images vs. landscapes. *Pattern Recognition*, 31(12) 1921–1935.

[2] Chapelle, O., Haffner, P., Vapnik, V. (1999). Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5) 1055–1064.

[3] Li, J., Najmi, A., Gray, R (2000). Image classification by a two-dimensional hidden markov model. *IEEE Transactions on Signal Processing*, 48(2) 517–533.

[4] Maree, R., Geurts, P., Piater, J., Wehenkel, L. (2005). Random subwindows for robust image classification. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, p. 34-40.

[5] Yang, C., Dong, M., Fotouhi, F. (2005). Region based image annotation through multiple-instance learning. *In: Proceedings of the ACM international conference on Multimedia*, p. 435-438.

[6] Li, J., Wang, J. (2006). Real-time computerized annotation of pictures. *Proceedings of the ACM international conference on Multimedia*, p. 911920.

[7] Lu, X., Kataria, S., Brouwer, W. J., Wang, J. Z., Mitra, P. (2009). Automated analysis of images in documents for intelligent document search. *International Journal on Document Analysis and Recognition*, 12.

[8] Blostein, D., Lank, E., Zanibbi, R. (2000). Treatment of diagrams in document image analysis. *In: Proceedings of the International Conferenceon Theory and Application of Diagrams*, p. 330-344.

[9] Huang, W., Tan, C. L., Leow, W. K. (2003). Model-based chart image recognition. *GREC*, 2003, p. 87–99.

[10] Huang, W., Tan, C. L., Leow, W. K. (2004). Elliptic arc vectorization for 3d pie chart recognition. *ICIP*, 2004, p. 2889–2892.

[11] Zhou, Y. P., Tan, C. L (2000). Hough technique for bar charts detection and recognition in document images. *International Conference on Image Processing*, 2, 605–608.

[12] Zhou, Y. P., Tan, C. L. (2001). Learning-based scientific chart recognition. *4th IAPR International Workshop on Graphics Recognition*, p. 482–492.

[13] Zhou, Y. P., Tan, C. L. (2000). Bar charts recognition using hough based syntactic segmentation. *In: Proceedings of the First International Conference on Theory and Application of Diagrams*, ser. Diagrams '00. London, UK: Springer-Verlag, p. 494–497.

[14] Huang, W., Zong, S., Tan, C. L. (2007). Chart image classification using multiple-instance learning. *WACV*, p. 27.

[15] Prasad, V. S. N., Siddiquie, B., Golbeck, J., Davis, L. S. (2007). Classifying computer generated charts. *Workshop on Content Based Multimedia Indexing*, p. 85–92.

[16] Bieniecki, W., Grabowski, S., Rosenberg, W. (2007). Image preprocessing for improving OCR accuracy. *In: Proceedings of the Int. Conf. Perspective Technologies in MEMS Design*, p. 75–80.

[17] Pilu, M. (2002). A lightweight text processing pipeline for pdas and embedded cameras. HP Lab, Tech. Rep. HPL-8.

[18] Jung, K., Kim, K., Jain, A (2004). Text information extraction in images and video: a survey. *Pattern Recognition*, 37, 977–997.

[19] Smith, R. (2007). An Overview of the Tesseract OCR Engine. *In: Proceedings of the Ninth International Conference on Document Analysis and Recognition*, 2, 629–633.

[20] Amin, A., Fischer, S (2000). A document skew detection method using the Hough transform. *Pattern Analysis and Applications*, 3, 243–253.

[21] Chen, D., Odobez, J.-M., Bourlard, H. (2004). Text detection and recognition in images and video frames. *Pattern Recognition*, 37, 595–608.

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

[23] Lu, X., Wang, J. Z., Mitra, P., Giles, C. L. (2007). Automatic extraction of data from 2-d plots in documents. *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, p. 188–192.

[24] Futrelle, R. P. et al (1992). Understanding diagrams in technical documents. *IEEE Computer*, p. 75–78.

[25] Poirier, M. D. B. (1997). An interactive system to extract structured text from a geometrical representation. *Proc. Fourth Intl. Conf. on Document Analysis and Recognition*, p. 342 – 346.

[26] Lank, E. (2003). A retargetable framework for interactive diagram recognition. *In: Proceedings of the International Conference on Document Analysis and Recognition*, p. 185-189.

[27] Satoh, S., Mo, H., Sakauchi, M. (1993). Drawing image understanding system with capability of rule learning. *Proc. Second Intl. Conf. on Document Analysis and Recognition*, Tsukuba, Japan, p. 119–124.

[28] Biederman, I. (1985). Human image understanding: Recent experiments and a theory. *Computer Vision, Graphics and Image Processing, 32*, p. 29–73.

[29] Mundy, J., Heller, A. (1990). The evolution and testing of a model-based object recognition system. *ICCV*, p. 268–282.

[30] Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8, 679–698.

[31] Kumar, P., Bhatnagar, D., Rao, P. U. (1991). Pseudo one pass thinning algorithm. *Pattern Recognition Letters*, 12, 543–555.

[32] Liu, S., Lin, N., Liang, C. (1988). An iterative edge linking algorithm with noise removal capability. *In: Proceedings of ICPR*, p. 1120–1122.

[33] Song, J., Su, F., Chen, J., Tai, C. L., Cai, S. (2000). Line net global vectorization: an algorithm and its performance analysis. *IEEE Conference on Computer Vision and Pattern Recognition*, p. 383–388.

[34] Vassilieva, N., Gladisheva, Y. (2010). Text Detection in Chart Images. *In: Proceedings of the 10th International Conference Pattern Recognition and Image Analysis: New Infromation Technologies PRIA-10,* 2, 365–368.

[35] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1).