Universität Passau
Fakultät für Informatik und Mathematik

CLASSIFICATION OF VISUALIZATIONS IN SCIENTIFIC LITERATURE

Masterarbeit zur Erlangung des akademischen Grades
Master of Science (M.Sc.)

Lehrstuhl für Intelligent Systems und Lehrstuhl für Data Science
der Fakultät für Informatik und Mathematik
der Universität Passau

Name:	Arnold Azeem
Matrikelnummer:	79176
Fachbereich:	Informatik
Studiengang:	Master Informatik
Erstprüfer:	Prof. Dr. Christin Siefert
Zweitprüfer:	Prof. Dr. Michael Granitzer
Date:	May 17, 2018

Contents

List of Figures

List of Tables

1 ABSTRACT

Distinct visualization techniques are used in scientific research publications to summarize large amount of data and also represent a variety of data. These visualizations help to communicate complex information and support the arguments presented in the paper in a easy to understand and follow way. These figures tend to reveal trends,patterns or relations that might otherwise be difficult to grasp using only text. In this context, classifying these visualizations is really relevant since there is a variety of visualizations and each one will have a different approach to processing it, example is extracting the raw data from it.

2 INTRODUCTION

"A picture is worth a thousand words" even though a widely used phrase stands to be very true especially when complex data is visualized and presented in scientific research publications. Data is ever growing and sometimes complex, using figures and diagrams to interpret and represent this data cannot be undermined since, they provide a way to easily give insight into the research findings, which would have otherwise been more complex relying on only textual data. For this reason, there has been a growing interest in the chart analysis area and quite a number of techniques have been developed. I spite of this growing interest, there has been little groundbreaking results achieved due to different variations in appearance of plots [1]. For example, Manollis Savva, et a [2] proposed a model to classify charts using extracted low-level features and textual features. After extracting the features, a Support Vector Machines (SVMs) classifier is used for the classification step. This method was limited since most charts contain the same type of features like axes, grid lines, and legends. In V Shiv Naga Prasad's work [3] classification was based on using features based on the shape and spatial relationships of their primitives. This work was limited due to the inconstancy in which data in most charts can be depicted. The process of extracting data already visualized as figures can be done relatively easier manually but becomes more complicated if done automatically. This process can be divided into two main steps [2]. The first step which our work focuses on, classifying the chart and the second step which involves extracting the data from the classified chart. To achieve the classification step, this paper presents an approach where charts are created with real-world datasets, different plotting programs (Python, Matlab, R, and Java) and different libraries supported by these plotting programs were used together with downloaded chart images from the Internet. We then use these images as input to Convolutional Neural Network (CNN). CNN was used instead of primitive approaches because it has achieved ground breaking results in the area of image classification [4]. Our model can identify four classes of plots namely Box-plots, Line Charts, Scatter-plots and, Bar-charts. The other parts of this paper are organized in the following way. In the next Sections, we present the motivation behind this work. Other works related to this, our pro-

posed method is described, Experimental evaluation and results are reported, and finally, the conclusion and the way to approach this work in the future.

2.1 MOTIVATION

Complex data is better explained in scientific papers with the aid of visualizations. These plots present complex data in an easy to understand way compared to textual representation. The data which these visualizations contain when extracted play an important role in events where another researcher wants to verify the work of the publisher, this data can also be used to develop other visualizations in situations where the paper needs to be presented to a different audience with a different background as opposed to the audience which the visualizations were created for, Also when comparing two plots the raw data helps make a better decision than just the figures. Since each plot will be processed differently to extract the raw data, it very relevant that we can distinguish one plot from another and this is the main aim of this thesis.

2.2 OBJECTIVE

The purpose of this thesis is to answer the question:

> HOW WELL CAN WE CLASSIFY THE DIFFERENT TYPES OF PLOTS IN SCIENTIFIC LITERATURE.

In this work we focus on only four plots. These plots are scatter plots,bar charts,line charts and Box plots. The diagram below shows the vision of this work, The first part of the diagram involves extracting or obtaining the four different types of plots mentioned earlier,after which we then label our plots and train a neural network model to be able to classify with high accuracy any of the four plots if shown to our model, then finally the raw data can be extracted from the detected plot. But this work mainly focuses on the red dotted lines shown below in the diagram which which is getting the plots, labeling them, training the model and classifying the plots.

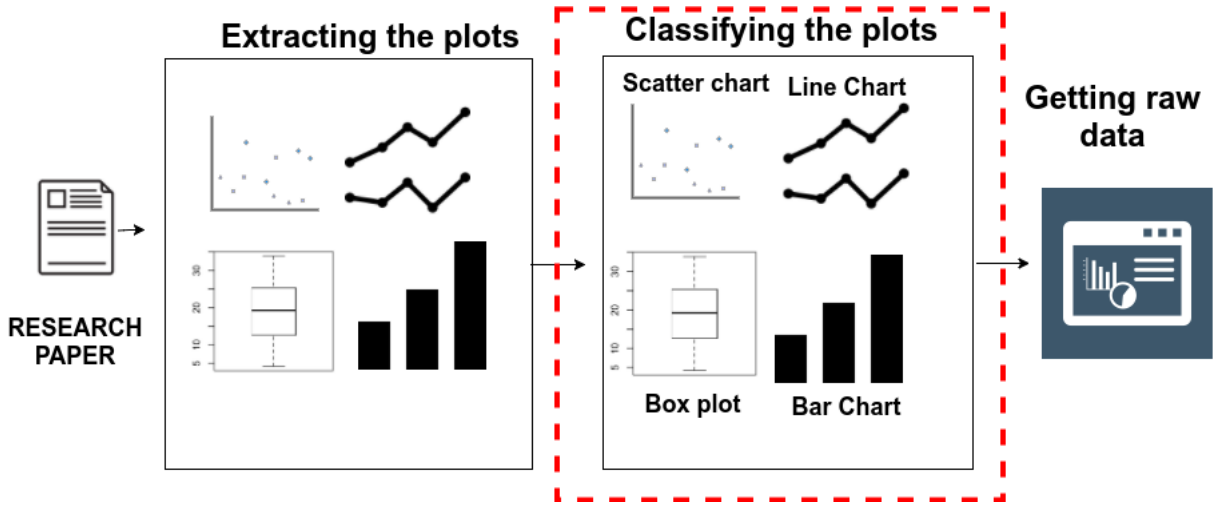


Figure 2.1: What we hope to achieve in the thesis

3 RELATED WORK

Our work is inspired by De Freitas et al. [5], proposed a way to extract the wealth of information contained in different visualization techniques. The paper talks about two main stages of accomplishing this task. Firstly, classification of the charts is done since it allows a different variety of chart to be detected automatically allowing the next step, which is the extraction of data from the classified plots. The paper, however, focuses on the first step, classification of charts. In this paper, a Convolutional Neural Network is used for the classification task. The Convolutional neural network encapsulates the characterization and classification processes during its learning process, unlike other techniques. The dataset used for this task were searched for and downloaded from Google image search. Table 3.1 shows the chart types which were collected and the number of train and test sets which the respective charts were divided into.

Chart Type	Test	Train
Area Chart	50	555
Bar Chart	50	657
Line Chart	50	489
Map	50	476
Pareto Chart	50	261
Pie Chart	50	361
Radar Chart	50	454
Scatter Chart	50	552
Table	44	236
Venn Diagram	48	304
Total	498	4345

Table 3.1: Number of Train and Test Dataset collected

For the classification, a variant of convolutional neural network called LeNet-based CNN model is used. The model was implemented using [1], LeNet-based

[1]Tensorflow

CNN has an architecture which is comprised of 3 convolutional layers, followed by a fully connected layer. The model is trained in a way that the dataset is divided into mini-batches, samples of fixed sizes(100) are selected and fed into the CNN, as a result of this process the model becomes robust since it learns to generalize from the different min-batches which are fed into the model. Also, all the images are converted to JPG and resized to 224x224x3, that is, 224 pixels of height, 224 of width and 3 layers of output. The other parameters used were 1000 epochs and a learning rate of 0.003. The accuracy at the end of the training process was 70%. In another paper by Liu et al. [1], a new approach was proposed for the process of chart classification. The process involves using convolutional neural network to extract deep hidden features of charts and then deep belief networks then use the extracted features to predict the labels of the charts. Due to a difficulty in acquiring a large number of charts as training data, natural images where first used to train the model and later the model was fined tuned with just over 5,000 collected charts. The types of charts collected were pie charts, scatter charts, line charts, bar charts, and flowcharts. The architecture of the CovNet is made up of five Convolutional layers and two fully-connected layers and then an output layer. The preprocessing steps for the images involve down-sampling them to 256 x 256 x 3, after which each is cropped to a size 227 x 227 from the center and its horizontal flip are extracted as the input of the CovNet, other parameters used for the CovNet include a learning rate that starts with 0.01 initially and is then decreased by a factor of 0.1 after every lOOk iterations, the weight decay parameter was set at 0.0005 and a dropout rate of 0.5. This results in an output of a 5-way softmax which produces the distribution over the 5 class labels and this is used as input for the deep belief network. The deep belief network architecture has three hidden layers, whose dimensions are 5000, 500 and 2000. This results in a softmax predicting the probability distribution over the 5 categories of charts as output. The training process was done with 4000 randomly selected images and the rest were used as test set. The accuracy of the model after the evaluation was 75.4%. Table 3.2 show the results after the training was done without deep belief networks but pre-trained with the natural images and finally the training done with only the chart dataset but with deep belief networks.

Chart	ConvNets	ConvNets+DBN without pre-training	ConvNets+DBN
Bar Chart	75.6%	45.6%	74.2%
flow Chart	88.3%	56.8%	91.3%
line Chart	71.2%	22.3%	67.9%
scatter Chart	69.8%	44.5%	84.2%
pie Chart	58.1%	50.1%	59.4%
ave Accuracy	72.6%	43.9%	75.4%

Table 3.2: COMPARISON RESULTS OF PROPOSED FRAMEWORK

3.1 Dataset

In this section, the various datasets used in plotting are described. For each language a different set of CSV files are used for generating the plots. This is done to generate more diverse plots.

3.1.1 Dataset for Matlab

The Data used for creating the plots in Matlab were randomly chosen from Project Dataset [6], a free CSV data repository, DatPlot [7] and Plotly CSV repository in github [8]. The datasets are multidimensional and compiled from normal day to day activities like dating, what makes people happy etc, and objects like cameras and cars. On the average the datasets used contain about 500 instance and 5 different columns. The biggest dataset is called Speed dating data. It is made up of over 8,000 observations of answers to survey questions about how people rate themselves and how they rate others on several dimensions. The smallest dataset used has 33 instances and 12 columns. It contains information about cars. The number of gears and speed, just to name a few attributes.

3.1.2 Dataset for R

For the plots in R, 13 random CSV files where downloaded from an archive of datasets distributed with R called Rdatasets [9]. Rdatasets is a collection of dataset distributed with R. On the average there are 80 instances and 5 columns in each dataset. The biggest CSV file is the Australian athletes dataset. Its made of 203 instances and 14 columns and contains attributes like sex,height,weight and sports. The smallest dataset is the Canadian Women's Labour-Force Participation. This dataset has 30 rows and 7 columns. It con-

tains information like average wages of women, percent of adult women in the workforce etc.

3.1.3 Dataset for Python

The data used for creating the plots in Python were 15 randomly seleted csv files also from Rdatasets [9]. The biggest dataset among the 15 is the Monoclonal gammapothy data, it contains natural history patients with monoclonal gammapothy of undetermined significance. The dataset is made up of 1384 observations with 10 columns, it has attributes like age, sex, time of death and last contact in months. On the average each dataset contains about 200 instances and 7 columns of multi-dimensional data. The smallest dataset however contains only 33 instances with 11 columns and is called the Nuclear Power Station Construction Data.The data relate to the construction of 32 light water reactor (LWR) plants constructed in the U.S.A in the late 1960's and early 1970's.

3.1.4 Dataset for Java

For the plots created in java, I used the dataset made available by Plotly [8], a github repository of CSV datasets used in the Plotly API examples. 14 random CSV files were downloaded, the biggest file has 1002 instances and 9 columns, and on the average each file contains about 100 instances and 9 columns. The smallest file however is made of 33 instances and 12 columns called the mtcars file. It contain information about a variety of different car models like the number of gears, speed etc. The table 3.3 contains the names of all CSV files that were used in the different languages with the different plotting programs.

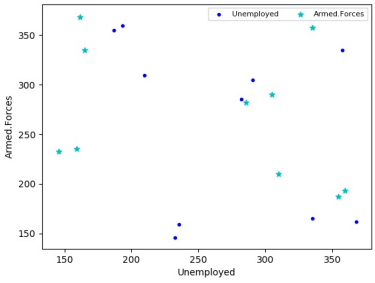
DUMMY DATA			
PYTHON	MATLAB	R LANGUAGE	JAVA
3d_line_sample_data.csv LightFordwardFlapStall.csv line_3d_dataset.csv longley.csv loti.csv lung.csv nuclear.csv timeseries.csv USJudgeRatings WVSCulturalMap.csv wind_rose.csv volcano.csv uspop2.csvm tips	Camera.csv Cars.csv speedDating.csv Cereal.csv happiness.csv TestData1.csv TestData2.csv mpg.csv okcupid-religion.csv spectral.csv stockdata.csv subplots.csv	ais.csv Angell.csv Baumann.csv Bfox.csv cane.csv carprice.csv Chirot.csv Davis.csv Ericksen.csv Florida.csv Highway1.csv Pottery.csv Prestige.csv salinity.csv urine.csv	3d-line-plot.csv 3d-scatter.csv 2011_flight_paths.csv 2011_us_exports.csv auto-mpg.csv candlestick_dataset.csv finance-charts-apple.csv globe_contours.csv hobbs-pearson-trials.csv motor_trend_tests.csv nz_weather.csv volcano.csv iris.csv mtcars.csv

Table 3.3: Names of datasets used in each plotting program

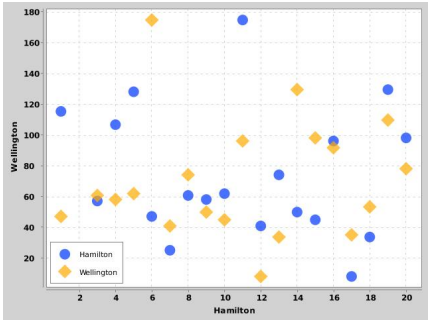
4 CREATING PLOTS

The inspiration for creating a variety of plots to capture all type of plots used in scientific papers was gotten by inspecting the dataset of Architecture proposal for data extraction of chart images using Convolutional Neural Network paper [5] and Viziometrics: Analyzing visual information in the scientific literature [10] dataset. Scripts in various languages were written to handle the plotting and labeling process automatically. All datasets for a particular plot (example scatter plot for python) are put into one folder. The scripts reads each CSV file column by column while creating the plots.

Table 4.1 describe how the plots where created in each language. The type column describes the different variety of a particular plot, for example bar charts can be of type stacked, grouped, vertical and horizontal bar charts, also scatter plots types can be a scatter plot consisting of one type of marker, one scatter plot with multiple markers and finally a scatter plot with a line showing the correlation between the plots. Figure 4.1 shows two different types of bar charts. Figure 4.1a is a stacked bar chart and Figure 4.1b a normal vertical bar chart. The Library column shows the different plotting libraries used, the parameter column describes parameters that were changed and finally the number of plots created were also added. The images below the tables are sample images that exist in our dataset of created plots for each language.



(a) Matplotlib scatter plot with star and circular markers

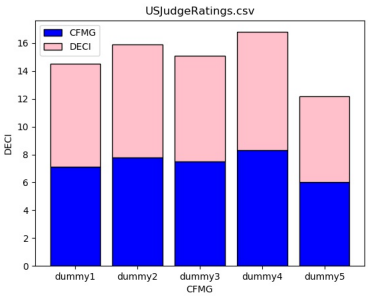


(b) Java scatter plot with circular and diamond markers

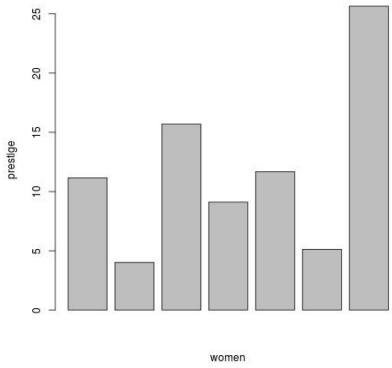
Figure 4.1: Scatter plots

Table 4.1: Overview of the varied parameters for creating plots in the different plotting programs

SCATTER PLOTS				
Language	Library	Parameters	Number of plots	Type
Python	Matplotlib v2.1.2 Plotly v2.5.1 Seaborn v0.8.1	MarkerStyle ['o', '*', '.', '+','x']	1020	Unique markers, With legends, multiple markers regplot
MATLAB	Default Plotly	MarkerStyle ['o', '*','+','x','s']	1044	
R	Plotly Lattice Ggplot2	MarkerStyle ['o', '*', '+','x','s']	1644	
JAVA	XChart 3.5.1 jfreechart 1.0.1	MarkerSize (15 - 18)	1644	
BAR CHARTS				
Language	Library	Parameters	Number of plots	Types(bar)
Python	Matplotlib v2.1.2 Plotly v2.5.1 Seaborn v0.8.1		1000	Horizontal and Vertical, Stacked, Grouped bar charts
MATLAB	Default	Width of bar(14-16)	1000	
R	Default,Plotly R Library ggplot2	space (0-3)	1144	
JAVA	XChart 3.5.1 jfreechart:1.0.192 javafx.scene	PlotOrientation (vertical or horizontal) with error bars	1144	
LINE CHARTS				
Language	Library	Parameters	Number of plots	Types(Line with)
Python	Matplotlib v2.1.2 Plotly v2.5.1 Seaborn v0.8.1	Linestyle ['-', '–', '-.', ':']	1000	Markers, Multiple Lines
MATLAB	Default Plotly	MarkerStyle ['o', '*', '.', '+','x','s'] markersize [8-10]	1000	
R	Default,Plotly R Library ggplot2		1644	
JAVA	XChart 3.5.1 javafx JFreeChart	MarkerSize (12-16)	1644	
Box Plots				
Language	Library	Parameters	Number of plots	Types(Box with)
Python	Matplotlib v2.1.2 Plotly v2.5.1 Seaborn v0.8.1		1000	Notches, Multiple Boxes
MATLAB	Default		1000	
R	Default,Plotly R Library ggplot2		1644	
JAVA	XChart 3.5.1	LegendPosition	1644	

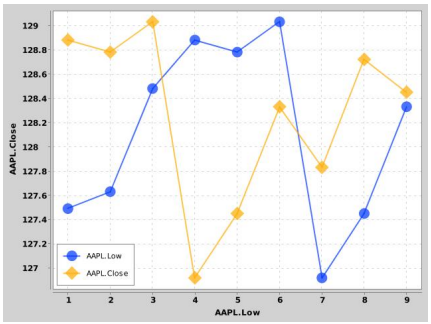


(a) Matlab stacked bar chart (bar width 16)

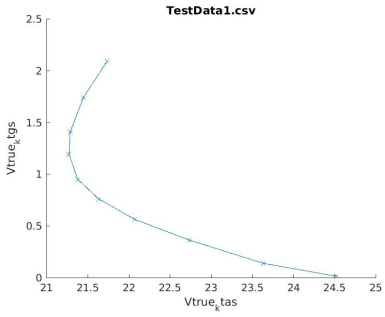


(b) R horizontal bar chart (bar width 16)

Figure 4.2: Example Bar Charts

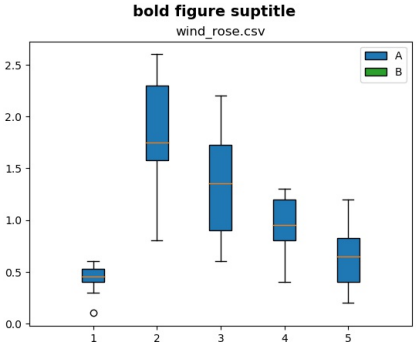


(a) Java line chart with diamond and circular markers

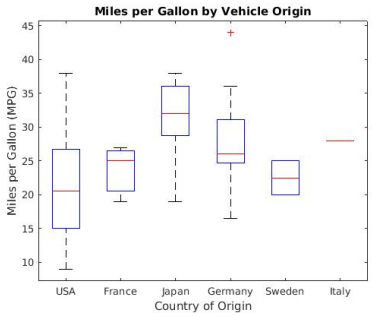


(b) simple Matlab line chart with Asterix marker

Figure 4.3: Example Line Charts



(a) Vertical multiple boxplots in python



(b) Vertical multiple boxplots in Matlab

Figure 4.4: Example Box Plots

4.0.1 Method

Convolutional Neural Network [1] was used for training and evaluation phase. In the next sections Convolutional neural networks will be discussed, the method and parameters used will also be talked about.

4.0.2 Convolutional Neural Network

The following is inspired by the following blogs by ujjwalkarn [11], Ifu Aniemeka[12] and Adit Deshpande [13]. CovNets (this abbreviation stands for Convolutional Neural Networks and will be used a lot through this work) are a variety of Artificial Neural Network which have given results near to human performance. They especially perform well in the areas of image recognition and classification. All CovNets are divided into steps called layers, the layers are Convolution, Pooling or Sub Sampling, Classification (Fully Connected Layer). When an image is presented to a computer, the image is seen as an array of pixel values. For instance if a color image of size 48x48 is presented the computer sees it as 48x48x3 where the 3 represents the RGB values.

Convolutional Layer

The convolutional layer will be explained using the following example, let consider the image as a magical cake of height and width 48x48, and there is a cake cutter of size say 5x5. The cutter is used to cut the cake from the top left. The cutter is referred to as a filter in machine learning. The cutter in this case is also an array of numbers called weights and for the maths to work the cutter must have the same depth as our cake (5x5x3). Lets first consider the cutter being in the first position ie. the top left of the cake. The values in the cutter are multiplied with the values of the cake (element wise multiplications). These multiplications are all summed up and result in a single number. This single number is only for the first part and this process is repeated through out the whole cake (Next step would be moving the cutter to the right by 1 unit, then right again by 1, and so on) keep in mind its a magical cake so you can cut a part more than once. After using the cutter on the whole of the magical cake we result in a new cake of size 28x28x1, the results is called a feature map. The filters are low level feature identifiers (straight edges, simple colors, and curves).

[1] https://deeplearning4j.org/convolutionalnetwork

ReLU Layer

The next layer after the convolution phase is the Rectified Linear Units (ReLU). This layers helps to handle situations where the relation between the input values and the CovNet output is non-linear. The ReLU has a function $f(x) = max(0, x)$ which means if you give it a value x, it will return 0 if x is negative and will return the value itself if its positive. There are also other functions which can used n place of the ReLU; that is the tanh or the sigmoid function but the ReLU mostly works well.

Pooling Layer

The pooling layers basically have two main functions, first one is that it helps the CovNet to locate features regardless of which part of the image it is located. This results in the model being robust against small changes in position of the features of the images. The second function is that it also helps to reduce the size of the feature map. Therefore computations in the futures layers are relatively less complex. One way of performing pooling is using max pooling technique, thats is sliding a window through the feature map, the windows fills a number of arrays in the feature map therefore, you pick the largest among all the numbers and disregard the rest of the numbers.

Fully Connected Layer

This is the last layer and where the decision about the image is made. i.e a probability saying which class the image belongs to. This layer takes the output of the previous layers, which are the high level features and check which class these features strongly correlate to. All the above mentioned layers can be multiply stacked on each other.

4.0.3 Image Preprocesing

Image preprocessing is a very important step in any image based application. This process involves taking the image and improving it in a way that enables easier machine understanding. This ultimately makes it easier for the machine to extract important features for other operations.

5 Bibliography

[1] Xiao Liu, Binbin Tang, Zhenyang Wang, Xianghua Xu, Shiliang Pu, Dapeng Tao, and Mingli Song. Chart classification by combining deep convolutional networks and deep belief networks. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 801–805. IEEE, 2015.

[2] Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. Revision: Automated classification, analysis and redesign of chart images. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 393–402. ACM, 2011.

[3] V Shiv Naga Prasad, Behjat Siddiquie, Jennifer Golbeck, and Larry S Davis. Classifying computer generated charts. In *Content-Based Multimedia Indexing, 2007. CBMI'07. International Workshop on*, pages 85–92. IEEE, 2007.

[4] Jihen Amara, Pawandeep Kaur, Michael Owonibi, and Bassem Bouaziz. Convolutional neural network based chart image classification. 2017.

[5] Paulo Roberto Silva Chagas Junior, Alexandre Abreu De Freitas, Rafael Daisuke Akiyama, Brunelli Pinto Miranda, Tiago Davi Oliveira De Araújo, Carlos Gustavo Resque Dos Santos, Bianchi Serique Meiguins, and Jefferson Magalhães De Morais. Architecture proposal for data extraction of chart images using convolutional neural network. In *Information Visualisation (IV), 2017 21st International Conference*, pages 318–323. IEEE, 2017.

[6] James Eagan. Project datasets. https://perso.telecom-paristech.fr/eagan/class/igr204/datasets. [Online; accessed 20-April-2018].

[7] Michael Vogt. Datplot. https://vincentarelbundock.github.io/Rdatasets/datasets.html, 2011. [Online; accessed 20-April-2018].

[8] plotly/datasets. Latex — Wikipedia, the free encyclopedia. https://github.com/plotly/datasets, 2011. [Online; accessed 20-April-2018].

[9] vincentarel bundock. Rdatasets. https://vincentarelbundock.github.io/Rdatasets/datasets.html, 2011. [Online; accessed 20-April-2018].

[10] Po-shen Lee, Jevin D West, and Bill Howe. Viziometrics: Analyzing visual information in the scientific literature. *IEEE Transactions on Big Data*, 4(1):117–129, 2018.

[11] ujjwalkarn. An intuitive explanation of convolutional neural networks – the data science blog. https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/, August 2016. (Accessed on 05/16/2018).

[12] Ifu Aniemeka. A friendly introduction to convolutional neural networks | hashrocket. https://hashrocket.com/blog/posts/a-friendly-introduction-to-convolutional-neural-networks#relu-layer, August 2017. (Accessed on 05/17/2018).

[13] Adit Deshpande. A beginner's guide to understanding convolutional neural networks – adit deshpande – cs undergrad at ucla ('19). https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/, July 2016. (Accessed on 05/17/2018).