Analysis and Modular Approach for Text Extraction from Scientific Figures on Limited Data

Falk Böschen

Dissertation
zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
(Dr.-Ing.)
der Technischen Fakultät
der Christian-Albrechts-Universität zu Kiel
eingereicht im Jahr 2021

1. Gutachter:	Prof. Dr. Ansgar Scherp Universität Ulm Ulm, Germany
2. Gutachter:	Prof. Dr. Matthias Renz Christian-Albrechts-Universität zu Kiel Kiel, Germany

Datum der mündlichen Prüfung: 09.12.2021

Zusammenfassung

Wissenschaftliche Abbildungen sind ein viel verwendetes Medium zur kompakten Darstellung von Kernaussagen. Auch über wissenschaftliche Publikationen hinaus finden diese Abbildungen, oft auch als Infografik bezeichnet, weit verbreitete Verwendung. Die Nachnutzbarkeit dieser Abbildungen ist jedoch häufig limitiert, da sie selten direkt auffindbar sind, sondern meist über die umschließenden Medien, wie beispielsweise Publikationen oder Webseiten, indexiert sind. Da diese Abbildungen in den meisten Fällen als Bilddateien gespeichert sind, lässt sich aus diesen nicht ohne weiteres deren Inhalt, sowohl weder der textuelle noch der graphisch kodierte, extrahieren. Erste Ansätze zur Extraktion des Inhalts mit Methoden der Bildverarbeitung wurden in der Vergangenheit vorgeschlagen, aber diese Ansätze sind meist auf spezifische Abbildungstypen beschränkt und nur auf einer handvoll Abbildungen getestet. In der Realität haben wissenschaftliche Abbildungen jedoch eine große Varianz in ihrer Darstellung (beispielsweise den verwendeten graphischen Elementen, beim Farbschema, der Ausrichtung und Rotation der Texte und variierenden Schriftgrößen), selbst innerhalb des selben Abbildungstyps. Eine Gemeinsamkeit, den die meisten Abbildungen besitzen, ist jedoch, dass sie in irgendeiner Form Informationen als Text enthalten.

Diese Arbeit fokussiert sich auf die Extraktion der textuellen Inhalte und versucht die bestehenden Lücken zu schließen. Zunächst erfolgt eine umfangreiche Analyse verwandter Arbeiten, um dann basierend auf dieser Analyse einen generalisierenden, modularen Ansatz für die Textextraktion aus wissenschaftlichen Abbildungen zu entwickeln. Dies bedeutet, dass keine Annahmen über die Abbildungen getroffen werden und sich nicht auf spezifische Abbildungstypen beschränkt wird. Mittels des modularen Ansatzes werden sieben unüberwachte Ansätze und ein überwachter Ansatz zusammengestellt, um diverse Methoden aus der Literatur sowie neue und bisher noch nicht genutzte Methoden zu vergleichen. Der beste der sieben unüberwachten Ansätze wird durch systematische Veränderung angepasst um 16 verschiedene Ansätze zu schaffen und zu evaluieren um den besten unüberwachten Ansatz für die Textextraktion aus wissenschaftlichen Abbildungen zu finden. Zudem wurde der überwachter Ansatz weiter entwickelt und optimiert, sodass er auch mit geringen Mengen an Trainingsdaten funktioniert. Zum Abschluss wird der beste unüberwachte Ansatz mit dem optimierten überwachten Ansatz zur Textextraktion aus wissenschaftlichen Abbildungen verglichen.

Um den Vergleich und die Evaluation durchzuführen wurden zwei Datensätze mit insgesamt 241 wissenschaftlichen Abbildungen erstellt und mit den nötigen Gold Standard Informationen annotiert. Außerdem werden drei weitere, existierende Datensätze für die Evaluation herangezogen. Für das Trainieren des überwachten Ansatzes wurden zudem zwei weitere Datensätze aus verwandten Text Extraktions Bereichen verwendet. Die Überlegenheit des unüberwachten Ansatzes über klassische Optical Character Recognition Verfahren wird mit dem ersten Experiment gezeigt. Das zweite Experiment und das dritte Experiment ermitteln aus der Menge an evaluierten Konfigurationen die beste Kombination and Methoden zur unüberwachten Extraktion von Text aus wissenschaftlichen Abbildungen. Das vierte Experiment optimiert das überwachte Verfahren. Das fünfte und letzte Experiment vergleicht die beste unüberwachte Variante mit dem optimierten überwachten Ansatz. Auch wenn der unüberwachte Ansatz vielversprechende Ergebnisse liefert, so wird er schlussendlich doch durch den überwachten Ansatz in den Schatten gestellt.

Abstract

Scientific figures are widely used as compact, comprehensible representations of important information. These kinds of information graphics are used in scientific publications, news media and other contexts. The re-usability of these figures is however limited, as one can rarely search directly for them. Most search engines index these figures via their surrounding text (e. g., publication or website) which often does not contain the full-message of the scientific figure. In addition, these figures are often stored as bitmaps which makes the content extraction difficult. Some approaches for content extraction from figures have been proposed that rely on computer vision methods. However, these approaches are mostly limited to specific figure types and are only tested on a handful of figures. In reality, figures are divers (for example with respect to graphical elements, colors, text orientation and size) even those from the same figure type. One thing almost all figures have in common is that they contain information as text.

This thesis aims to close the gaps by performing an extensive literature survey and comparison of the various approaches to build a modular pipeline for text extraction from scientific figures. This new modular pipeline does not make any assumptions about the figures and is not limited to any figure type. With the modular pipeline seven unsupervised approaches and one supervised approach are build to compare the different methods from the literature as well as methods that have not been used in the literature so far. The best of the seven unsupervised approaches is further modified to form and evaluate 16 additional approaches, which have not been analyzed in the literature before, to find the best unsupervised approach for text extraction from scientific figures. Furthermore, the supervised approach is improved and optimized, so that it can work with limited training data. A comparison between the best unsupervised approach and the best supervised approach for text extraction from scientific figures is conducted.

To conduct the comparison and evaluate the approaches, two datasets of in total 241 scientific figures were created and annotated with gold standard information. In addition three existing datasets are used for evaluation. Furthermore, additional, existing datasets for text extraction from other types of images are used for training the supervised approach. The superiority of the unsupervised pipeline over common Optical Character Recognition engines is proven with the first experiment. The second and third experiment identify the best approach for unsupervised text extraction from the set of selected unsupervised approaches. The fourth experiment optimizes the proposed supervised approach. The fifth and final experiment compares the optimized supervised approach with the best unsupervised approach. Even though the unsupervised approach produces promising results, it is clearly outperformed by the supervised approach.

Acknowledgments

First of all, I would like to thank my supervisor Prof. Ansgar Scherp for introducing me to the area of data mining and machine learning. He gave me the opportunity to learn and work independently in this interesting area, and showed me how to become a better scientist and teacher. Second, I would like to personally thank my colleagues who created a great working environment. They always had an open ear for discussing research topics and gave valuable feedback (in alphabetic order): Till Blume, Paula Bräuer, Lukas Galke, Arne Martin Klemenz, Athanasios Mazarakis, Ahmed Saleh, Falko Schönteich and Iacopo Vagliano. Last but not least, I would like to thank my student assistants Tilman Beck, Morten Jessen, Oke Nissen and Eike Lurz for their valuable support by developing software and executing experiments.

Contents

List of Figures

List of Tables

List of Abbreviations

AP	Average Precision
API	Application Programming Interface
BM25	Best Matching 25
CCL	Connected Component Labeling
CDA	Conditional Dilation Algorithm
CG	Cumulative Gain
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DCG	Discounted Cumulative Gain
ER	Element Ratio
Fast R-CNN	Fast Region Convolutional Neural Network
Faster R-CNN	Faster Region Convolutional Neural Network
FN	False Negatives
FP	False Positives
F1	F1-Measure
GPU	Graphical Processing Unit
HSL	Hue-Saturation-Luminance
HSV	Hue-Saturation-Value
IDCG	Ideal Discounted Cumulative Gain
IoU	Intersection-over-Union
LD	Levenshtein Distance

MER	Matched Element Ratio
MST	Minimum Spanning Tree
NDCG	Normalized Discounted Cumulative Gain
OCR	Optical Character Recognition
OPC	Operations Per Character
P	Precision
PDF	Portable Document Format
PSD	Perpendicular Squared Distance
R	Recall
R-CNN	Region Convolutional Neural Network
ResNet	Residual Neural Network
REST	Representational State Transfer
RGB	Red-Green-Blue
RoI	Region of Interest
RoI Align	Region of Interest Align
RoI Pool	Region of Interest Pool
RPN	Region Proposal Network
SciFiS	Scientific Figure Search
SSOD	Single String Orientation Detection
SVG	Scalable Vector Graphic
SVM	Support Vector Machine
TF-IDF	Term Frequency - Inverse Document Frequency
TN	True Negatives
TP	True Positives
VEM	Visual Extraction Module
XML	Extensible Markup Language

Chapter 1

Introduction

1.1 Motivation

Scientific Documents are more than just the text they contain. Popular platforms like Semantic Scholar have identified figures (e. g., plots, diagrams, bar charts) in scientific documents as a key component whose value has been underestimated so far [SLP+18]. Four examples of these kinds of figures are shown in Figure 1.1. Lee et al. [LWH18] identified a significant correlation between the number of figures in a document and its scientific impact, i. e., a publication with a high scientific impact tends to include more figures. Thus, these figures are an integral part of high impact publications. However, while the retrieval of documents and natural images (e. g., photos) has made progress over the years, the retrieval of figures has only gotten little attention [LSC+13]. Even though figures visualize information which should be used to retrieve these figures by their actual content [LCF+14a; YAD+14], figures are usually indexed like natural images through their surrounding text, if at all, which usually does not contain the message of the figure itself [NAA+10]. In addition it is a known, open challenge to optimize how much of the surrounding text one should use for indexing images [AM10]. Finally, creating a better understanding and retrieval of these figures also enables a new way to access scientific documents.

Scientific documents are predominantly available in the Portable Document Format (PDF). There are two image types that appear in these documents. Figures can either be bitmap images or Scalable Vector Graphics (SVGs). Scalable Vector Graphics are machine readable, but they are difficult to extract from PDFs, as their parts can be scattered throughout a PDF when parsing it. In addition, especially in older scientific documents,

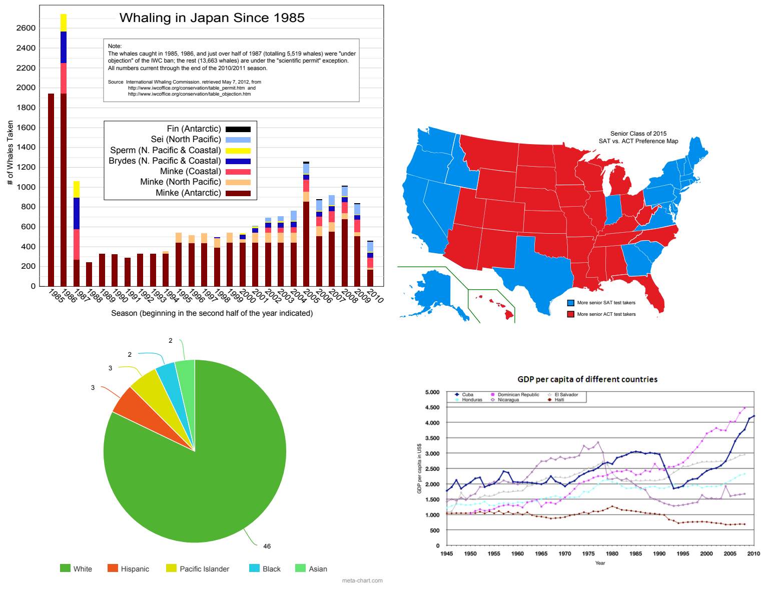


Figure 1.1. Examples of figures (Source: Wikimedia Commmons Public Domain)

only a small fraction of figures are actually vector graphics. However, every vector graphic can be converted into a bitmap figure by taking a screenshot of the rendered vector graphic at a suitable resolution. Thus, the focus in this thesis is on improving the understanding of bitmap figures from scientific documents. Scientific figures commonly consist of textual and graphical components and the information that they convey is encoded in both. While graphic components give a rough estimate of the message, text is often use to make the message more precise. Therefore the extraction of text is important since text can stand for itself while graphical elements might be imprecise or lose its meaning. This thesis introduces, describes, compares and evaluates a modular pipeline for text extraction from scientific figures which combines existing approaches as well as newly introduced approaches to find the best approach for text extraction from scientific figures.

1.2 Problem Statement and Research Questions

Text extraction from scientific figures poses several challenges since these figures are designed for human readability, but not necessarily machine comprehension.

First of all, different types of scientific figures exist - like for example bar charts, line graphs or Venn diagrams - and more and more types and variations appear over time. Second, different tools (e. g., Microsoft Excel, LibreOffice, Matlab) can be used to create scientific figures and thus a wide range of customization and variation in the visualization is possible. Third, due to the heterogeneity of the figures, a lack of (training) datasets exist. Fourth, extracting text from scientific figures poses different challenges than extracting text from natural images. For example, graphical elements of scientific figures have features more similar to text than natural objects in photos and thus can easily be mistaken as text (e. g., markers, dashed lines). While approaches for text extraction on natural images often can make assumptions about color homogeneity [GF05; LCJ+10; YT11] or the consistency of stroke width [EOW10], such assumptions usually can not be applied on scientific figures. Fifth, Optical Character Recognition (OCR) engines are designed to analyze whole pages of documents and thus work best on horizontal black and white text. However, scientific figures can contain text of various orientations and color, making the extraction even more difficult.

The following challenges are addressed in this thesis:

1. Text extraction from scientific figures with only limited amount of training data available.
2. Text extraction from scientific figures without making assumptions about the figure type.
3. Text extraction from scientific figures without making assumptions about the figure coloring.
4. Text extraction from scientific figures without making assumptions about text placement.
5. Text extraction from scientific figures without making assumptions about text orientation.

Based on the challenges presented above, four research questions were identified:

RQ1: How good are commonly used document Optical Character Recognition engines at extracting text from scientific figures?

RQ2: Which unsupervised approach is the state-of-the-art, measured by the accuracy of the text localization and recognition quality, in extracting text from scientific figures?

RQ3: Can a new, unsupervised approach combining existing and new methods improve on the text location and extraction quality?

RQ4: How does the best unsupervised approach compare with a supervised approach which is trained on a small dataset?

1.3 Contributions

Based on the research questions, in summary, this thesis makes the following contributions:

▻ A thorough analysis of the state-of-the-art is conducted to map the area of unsupervised text extraction from scientific figures and related image types. Based on this analysis, a unified pipeline for text extraction from scientific figures is abstracted which can be used to compare the different approaches and their methods.

▻ The analysis of the state-of-the-art reveals that almost no datasets for text extraction from scientific figures are publicly available and those that are available only contain horizontal text or no orientation information is given. Thus, two datasets, of 121 and 120 scientific figures respectively, are created and annotated with gold standard information to be able to evaluate the different approaches.

▻ The analysis of the state-of-the-art shows a lack in approaches that can deal with rotated text. Thus, a new approach to detect and extract rotated text from scientific figures is proposed. It is capable of detecting text at arbitrary orientation in the range $[-90°, 90°]$.

▻ A comparison of the different state-of-the-art unsupervised methods and approaches for text extraction as well as new methods is conducted to find the best unsupervised approach for text extraction from scientific figures. In further iterations, the best state-of-the-art approach is refined by creating and evaluating new pipelines. These new pipelines only differ with respect to one step to the base pipeline to identify further improvements.

▻ Supervised approaches usually rely on large training datasets which are non-existent for text extraction from scientific figures. Nevertheless, a supervised approach was developed and optimized which is capable of working with limited amount of training data which achieves more than competitive results. Finally, this optimized supervised approach is set in comparison with the best unsupervised approach for text extraction from scientific figures. The comparison shows that the supervised approach clearly outperforms the unsupervised approach.

1.4 Publication Record

This thesis is based on several publications which have been presented at/in different workshops, conferences and journals over the last years. The following publications contribute to this thesis:

- ▻ F. Böschen and A. Scherp. "Multi-Oriented Text Extraction from Information Graphics". In: *Proceedings of the 2015 ACM Symposium on Document Engineering, DocEng 2015, Lausanne, Switzerland, September 8-11, 2015, ACM, 2015, pp. 35-38.*
- ▻ F. Böschen and A. Scherp. "Formalization and Preliminary Evaluation of a Pipeline for Text Extract From Infographics". In: *Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB, Trier, Germany, October 7-9, 2015, CEUR-WS.org, 2015, 1458, pp. 20-31*
- ▻ F. Böschen and A. Scherp. "A Systematic Comparison of Different Approaches of Unsupervised Extraction of Text from Scholarly Figures [Extended Report]". *Institut für Informatik der Christian-Albrechts-Universität zu Kiel, 2016*
- ▻ F. Böschen and A. Scherp. "A Comparison of Approaches for Automated Text Extraction from Scholarly Figures". In: *Proceedings of the Multimedia Modeling - 23rd International Conference, MMM 2017, Reykjavik, Iceland, January 4-6, 2017, Part I, Springer, 2017, 10132, pp. 15-27*
- ▻ F. Böschen, T. Beck and A. Scherp. "Survey and Empirical Comparison of Different Approaches for Text Extraction from Scholarly Figures". In: *Multimedia Tools and Applications, 2018, 77, 29475-29505*
- ▻ M. Jessen, F. Böschen and A. Scherp. "Text Localization in Scientific Figures using Fully Convolutional Neural Networks on Limited Training Data". *Proceedings of the 2019 ACM Symposium on Document Engineering, DocEng 2019, Berlin, Germany, September 23-26, 2019, ACM, 2019* (Best Student Paper Award)

This thesis rephrases and reorganizes the content of these papers, articles and reports and extends them with additional aspects.

1.5 Outline

The remainder of this thesis is structured as follows: Subsequently, in Chapter 2, the related work in text extraction from scientific figures and similar image types are presented and analyzed. From that analysis, a generic pipeline for text extraction is derived and presented in Chapter 3. The pipeline is formally defined in Section 3.1. In Section 3.2, various methods for the steps of the pipeline are extracted from the related work and mapped onto the pipeline. Section 3.3 explains how to build working configurations of the pipeline from these methods. In Chapter 4 the evaluation setup is described. It starts in Section 4.1 with a description of the datasets that are used in the experiments, followed by a specification of the evaluation measurements in Section 4.2. Five experiments and their results are outlined in the Chapters 5-9 which are then discussed in Chapter 10. Each of the five experiments was executed to address one of the research questions (compare Section 1.2). The first experiment in Chapter 5 addresses **RQ1** about the usefulness of document OCR engines. The second and third experiments (see Chapter 6 and Chapter 7 respectively) answer the research question **RQ2** on the best unsupervised approach from the literature and the research question **RQ3** on the best and optimized, unsupervised approach. The fourth experiment in Chapter 8 optimizes the supervised approach in preparation for the fourth research question **RQ4**. The final, fifth experiment (see Chapter 9) addresses the last research question **RQ4** by comparing the best unsupervised approach with the best supervised approach. Finally, in Chapter 11, a summarization of the thesis and its results as well as an outlook for future research directions are given.

Chapter 2

Related Work

Different terms have been used for figures in the context of text extraction from images, e. g., information graphics [CED06], figures [CG15], charts [HTL05], diagrams [CCA15], graphs [LCF+14a] and different variations and combinations of them. In this thesis, all of these variations are summarized under the term scientific figures or figures for short.

In this chapter, an overview of existing approaches for text extraction from scientific figures is presented. It also covers a selection of related approaches for text extraction from natural images which propose interesting ideas as well as approaches that go beyond the text extraction from scientific figures and also extract the data encoded in the graphical elements for data reconstruction or figure summarization.

2.1 Unsupervised Text Extraction from Scientific Figures

Several researchers have addressed the problem of extracting text from scientific figures. In the following, a broad selection of research works are described in chronological order to show the development of text extraction from figures.

Fletcher and Kasturi [FK88] proposed an algorithm for an automated document analysis system which can extract text strings from technical document-like images with a mixture of text and graphical elements. Their algorithm separates text from graphical elements but does not perform character recognition. First, Connected Component Labeling (CCL) is used to identify coherent regions which are then grouped into character strings using the Hough transformation. They conclude that their algorithm is relatively tolerant with respect to text font style, size and orientation based on an evaluation on two test images.

Deseilligny et al. [DMS95; DML+97] focused on the extraction of text elements from scanned topographic maps. Topographic maps use text elements to show city and street names, regions and landmarks. The approach by Deseilligny et al. relies on Connected Component Labeling for region extraction. Deseilligny et al. normalize each region in order to apply a rotation invariant character recognition. Multiple character hypotheses are generated for each region and those hypotheses are selected which create coherent strings and follow specific syntactic rules. They evaluate their approach on a small set of maps from the French national geographic institute and only a small fraction of less than 5% of the strings in these maps is not correctly recognized.

Adam et al. [AOC+99; AOC+00; ARO+01] developed an approach for extracting the character layer from urban maps which are overlayed with the network of the French Telephonic Operator (France Telecom). They first use a combination of a multi-thresholding technique and morphologic criteria selection to separate the urban, the network and the characters/symbols layers. Then they use a shape extractor to find patterns which then are enriched with a feature vector based on the Mellin Fourier Transformation. Finally, the feature vectors are fed to a classifier which has to identify which of 51 classes (characters) the shape belongs to. They evaluate their approach on a clean, artificial dataset of about 400 samples for which they achieve a classification rate of 97%. Furthermore, they evaluate on a real-world dataset of 14.000 samples from France telecom where they achieve a classification rate of 87%.

Jayant et al. [JRW+07] proposed a semi-automated approach for text extraction from figures with conversion to Braille, the language for the visually impaired. First, a color reduction is conducted with Adobe Photoshop. Subsequently, the figure is manually classified into a set of predefined figure types. Connected Component Labeling is applied to the figure to extract regions. In order to separate text elements from graphical elements, the authors manually train a Support Vector Machine (SVM) per figure type as well as per book where the figures were taken from. Subsequently, a separation into text line structures is performed, using a so-called label training algorithm which uses a MST with manually created test data. The text line orientation is estimated by minimizing the Perpendicular Squared Distance (PSD). Finally, OCR is conducted and the recognized text can be translated into Braille.

Takagi [Tak09] addressed the specific problem of extracting and converting text from mathematical line graphs into Braille. The focus is hereby on the detection of broken lines, which can be wrongly classified as text. The proposed approach starts with binarization, noise reduction and Connected Component Labeling. A set of heuristic rules is applied to separate text components from graphical components. Local segment density, fuzzy inference and hierarchical clustering are used to identify broken line segments. The limited evaluation with 6 figures shows that the separation into text and graphical components is only successful for two figures.

An algorithm for text detection in biomedical images, as part of the Yale Image Finder, was proposed by Xu and Krauthammer [XK10]. The authors first detect and remove so-called layout elements, followed by a binarization, median filter and edge detection with the Sobel operator. The text region extraction, based on horizontal and vertical histogram projection analysis, is conducted on the edge image. This is performed recursively until the image cannot be split any further. During this recursive processing of the regions, heuristic filters are applied to only subdivide those regions that contain text and discard the others. They evaluate their approach on a self-created dataset of 161 biomedical images from PubMed Central. The evaluation measures are the Precision, Recall and F1-Measure on the pixel level between bounding boxes as well as what they call the modulated overlapping area which is calculated using the union of the bounding boxes of the gold standard and those created by their algorithm. Their best result is a F1-Measure of 0.6.

Chiang and Knoblock [CK11; CK15] presented a semi-automatic text extraction from maps. In contrast to most of the other works, the input image is not converted to grey-scale. Instead, a color quantization algorithm is applied to analyze the dominant colors separately. The authors separate text elements from graphical elements using a run-length smoothing algorithm based semi-automatic extraction that requires a positive and a negative example for each text-color/background-color combination. Text lines are detected by applying dilation operators on the connected components. The orientation of each line is estimated using a Single String

Orientation Detection (SSOD) algorithm, which is based on morphological operations. The algorithm evaluates all possible orientations of text line candidates in a brute-force manner. The text line is rotated to horizontal orientation and the commercial ABBYY FineReader[1] is applied for Optical Character Recognition. After the OCR phase, a recognition confidence score is computed to filter the results. The evaluation is conducted on 3 scanned maps and 12 computer-generated maps. The results show that on the scanned maps, the algorithm is much more inaccurate than on the computer-generated maps. However, overall, they can show an improvement over using a stand-alone OCR solution.

Sas and Zolnierek [SZ13] proposed a three-stage approach for text extraction from figures. Starting with a conversion of the input image to grey-scale, the authors apply filtering operations, binarization and Connected Component Labeling to generate coherent regions. Regions are filtered by empirical thresholds and classified into text and graphic elements using a decision tree. Tesseract[2] is used for Optical Character Recognition. Besides normal orientation, the input to the OCR engine is also rotated at a 90° angle to capture vertical text elements such as labels of the y-axis. Finally, the text detection is verified by assessing the number of special characters recognized in the text regions. Unfortunately, the authors did not assess the quality of their OCR results.

One of the contributions of this thesis (see Section 1.3 and Section 1.4) is an unsupervised text extraction pipeline from scientific figures [BS15b; BS15a]. The text extraction pipeline uses an adaptive binarization method based on Otsu's method [Ots79] to convert images to binary format. Subsequently, Connected Component Labeling is applied to extract coherent regions. Heuristic rules are applied to drop certain (noise) regions before the remaining regions are clustered using DBSCAN in order to separate text elements from graphical elements. A MST clustering is applied to the clusters to detect single text lines. The orientation of the text lines is computed using a discrete Hough transformation and each line is rotated into horizontal mode in order to send it to an OCR engine. The pipeline was extended into a modular framework for evaluating several methods from the literature [BS16; BS17; BBS18].

[1] `http://www.abbyy.com/ocr-sdk/`, last access: November, 2018

[2] `https://github.com/tesseract-ocr/`, last access: November, 2018

2.2 Supervised Text Extraction from Scientific Figures

Modern supervised methods, i. e., neural networks, have only been recently used for text extraction from scientific figures. The most relevant research is mentioned below.

Almakky et al. [APR19] presented a deep Convolutional Neural Network (CNN) architecture for text localization in biomedical figures. Their model maintains the image size from end-to-end and does not apply dimensionality reduction by padding the input in each layer. In addition, Almakky et al. used different transformation functions to account for the different possible variations in text appearances. These transformations included random crop, random sized crop, random rotation, color inverse and scaling. They used two datasets in their experiments. First, for pre-training their model, they used the SynthText in the Wild dataset which consists of 800,000 images with synthetic scene-text which accounts for the local scene geometry. Second, for training and evaluating their model,the DeTEXT dataset from the ICDAR2017 robust reading challenge with 500 biomedical figures was used. Their model achieves 91% Recall and 62% Precision under the assumption that a text element was correctly detected when the bounding box overlap is 50% or greater.

Sarshogh and Hines [SH19] proposed a model for text localization and text recognition for what they call complex documents. Their model consists of a shallow DenseNet, a pyramid network, a region proposal layer and three heads (bounding box regression, text classifier and text recognition). They pre-trained their model with 10,000 synthetical created images before fine-tuning it with the training set of the ICDAR 2017 DeTEXT dataset which was then used to evaluate their model. Their results show an increase in the mean average precision for both text localization and text recognition.

Morris et al. [MTE19] proposed a deep convolutional architecture for text extraction from scientific figures. They used a modified version of EAST (An Efficient and Accurate Scene Text Detector) [ZYW+17] to localize the text in the figures, because certain aspects like angled (rotated with respect to the camera perspective) text usually does not appear in scientific figures. In a second step, Tesseract is used to recognize the text. Two datasets were used to evaluate their model. A dataset with 500,000 labeled images from the arXiv platform and the Multi-Type Web Images dataset. They report that they outperform the best unsupervised approaches of Böschen et al. [BS17; BBS18].

Morten et al. [JBS19] presented a neural network based approach that uses a modified Faster Region Convolutional Neural Network (Faster R-CNN) architecture [RHG+15] based on ResNet [HZR+16] for the detection of text elements in scientific figures. The model is pretrained with the MS-COCO Text dataset [LMB+14]. For training and evaluation five different datasets are considered and artificially extending by creating augmented versions of the figures in the datasets. The augmentations include translation, rotations, scaling, flipping and addition of noise. After text location with the modified Faster R-CNN, Tesseract is used to recognize the text. Text at different orientations is addressed by brute-force rotating the elements before recognizing it with Tesseract. The experiments show that they surpass the unsupervised approaches of Böschen et al. [BS17; BBS18].

2.3 Text Extraction from Natural Images

Another large research area is the extraction of text from natural images, also known as scene text extraction. Approaches in this area usually make assumptions about the difference between the text and the rest of the image which can not be directly applied to scientific figures. However, certain methods are interesting and worth to evaluate on scientific figures.

Gllavata et al. [GF05] proposed an approach for text segmentation in images with varying background colors. They first execute a fuzzy c-Means clustering to identify the different colors in the image. Then they create multiple binary images, one per color cluster, and apply Connected Component Labeling to create regions. Features are assigned to the regions and evaluated to find the cluster that contains the color of the text. A heuristic enhancement process which also uses morphological operations concludes the text segmentation. Their experiments include two datasets, the MPEG-7 test set with 45 images and a second test set which consists of 18 video frames. From the in total 2,486 Latin characters 90.4% were correctly recognized and 78% of the 441 words were correctly recognized.

Liu and Samarabandu [LS06] made the assumption that text can be identified based on edge strength, density and orientation variance. Their three-staged approach consists of candidate detection, text detection and text extraction. For candidate detection, they use as multi-scale edge detector to measure the edge strength via the second derivative of edge intensity which is combined with density and orientation information via scale fusion. Text detection is performed using morphological operations to merge characters into text blobs. Finally, the text pixels are extracted and a binary image is created which is fed to an OCR engine for recognition. The approach was tested on 75 images which show text of different font sizes, colors, orientations and perspective projections under different lightning conditions and achieved a Precision of 0.918 and Recall of 0.966.

Liu and Sarkar [LS08] proposed an approach for text extraction from outdoor images. First, Niblacks algorithm is used to binarize the input image. Next, regions are generated and grouped based on four features (horizontal alignment, similar intensity, similar height and closeness). Afterwards, an intensity filter and a shape filter are applied to remove noise and non-text areas. The ICDAR2003 text locating dataset from the Robust Reading Competition with 249 images is used for evaluation and the proposed approach achieves 0.66 Precision, 0.46 Recall and 0.54 F1-Measure.

Epshtein et al. [EOW10] developed a novel image operator that estimates the stroke width of each image pixel which can be used for text detection in natural images. The creation of an edge map using the Canny edge detector is the first step in their pipeline. Next, their stroke width transformation is applied to the edge image. Text character candidates are identified using a modified version of the Connected Component Labeling and heuristic rules. Based on the assumption that single characters do not appear in images, characters are grouped into lines to eliminate noise. Finally, the line is broken down into words by analyzing the distances between the characters. Datasets from the ICDAR2003 and ICDAR2005 text detection competitions are used for evaluation and their algorithm achieves a Precision of 0.73, a Recall of 0.60 and F1-Measure of 0.66.

The pipeline proposed by Fraz et al. [FSE15] relies on color information for text extraction from natural images. The pipeline starts with an estimation of the color constancy using the grey-world algorithm, followed by a noise reduction step using median filters. The third step is a minimum-variance color quantization into 8 colors and a binary image is computed for each color on which Connected Component Labeling is applied. The generated regions are analyzed with respect to their height, width and aspect ratio to identify text candidate regions. Next, the HOG feature descriptor is computed for each remaining candidate region and a SVM is trained to differentiate between text and non-text regions. In the last step, words are formed from the individual characters. A second pipeline is applied for character identification before the characters and words are recognized. The text detection is evaluated on the ICDAR11 benchmark datasets and the Street View Text dataset. The character recognition is evaluated on the Chars74K-15 and ICDAR03-CH datasets. Their two pipelines achieve competitive results (e. g., an F1-Measure of 0.71 on ICDAR11) when compared with state-of-the-art approaches.

An edge-based approach using support vector regression was developed by Lu et al. [LCT+15] for text extraction from natural images. The edge image is created using Canny's edge detector and three features (contrast, shape and stroke structure) are derived from that edge image at multiple scales. A global thresholding is applied to detect text bounding box candidates and a support vector regression is used to remove non-text bounding boxes. Their approach achieves a F1-Measure of 0.78 on the ICDAR2013 Robust Reading Competition dataset.

Borisyuk et al. [BGS18] presented a large scale Optical Character Recognition system called Rosetta. They use a Faster R-CNN state-of-the-art object detection network to localize text in natural images. Their neural network replaces the common convolutional body of ResNet [HZR+16] with a ShuffleNet-based architecture [ZZL+18]. Two additional models for text recognition are evaluated, one based on character sequence encoding and one based on fully-convolutional model (ResNet-18). In a thorough evaluation, a trade-off between accuracy and efficiency is found.

2.4 Data Reconstruction and Summarization of Scientific Figures

Some researchers go beyond the extraction of text from figures and see it only as one part of a larger system for data reconstruction and/or summarization of the content of a figure. A selection of these approaches is presented in the following.

Huang et al. [HTL03; HTL05; HT07] proposed an approach for data and text extraction from scientific figures, especially bar charts, pie charts and line graphs. Their extraction pipeline starts with a Connected Component Labeling that generates components of coherent text elements and graphics elements. In a subsequent step, these elements are separated by applying a series of filters. In the next step, the text elements are grouped using a derivation of Newton's formula for "Gravity" from classical physics. The authors claim that this method is capable of separating text at different orientations into different groups of text elements. Optical Character Recognition is applied on these text groups and the recognized text is classified into strings and numbers. Finally, the recognized text is manually corrected in order to have a clean assignment to the corresponding graphical elements. The graphical elements are analyzed and mapped with the extracted text to retrieve the original data values.

A joint project of the University of Delaware and the Millersville University had the goal of making figures accessible to sight-impaired individuals. Chester and Elzer [CE05] presented the Visual Extraction Module (VEM) for conversion of figure images into an XML representation. Their focus was on simple bar charts, line charts and pie charts, but their system was designed with the possibility of extension to other figure types. The VEM first creates connected components from uniformly colored regions in the figure. Heuristics are used to distinguish between connected components that represent text and those that represent graphical elements. Followed by a simple optical character recognition algorithm to extract the characters from the connected component text candidates based on simple template matching. Using another heuristic, characters are grouped into words using morphological operations. Semantic labels like title or Y-axis descriptor are assigned to the recognized words based on certain location assumptions. Next, the graphical components are broken down into their core components and grouped such that the figure type can be identified. All information is outputted in XML format. Several projects are build upon VEM [BCE08; Dem08; DCM08; DOS+10; BCE10; WCE+10; ECZ11; CSM+12]. They all assume having a perfect text extraction given by the VEM and focus on figure summarization, message identification and user interaction. In 2012, Gao et al. [GZB12] from the University of Delaware tried a new approach and developed the Visual Information Extraction Widget to automatically extract the data from bar charts, pie charts and line graphs. First, the input image is segmented into text and graphic components based on pixel intensity using morphology that generates connected components which are sorted by heuristic rules. Then geometric features are extracted from the graphical components which are used by a Support Vector Machine to identify the figure type. Depending on the identified figure type, for bar charts, pie charts and line graphs, a unique type-specific data extraction method is applied. They collected 100 bar charts, 100 pie charts and 100 line graphs for their evaluation and showed that their classification accuracy achieves 97.3% in a 5-fold cross validation.

Kataria et al. [KBM+08] proposed a system for extracting the numerical values from 2D plots. First, the 2D plot is binarized. Then, the axes are detected and the plot is segmented by applying Connected Component Labeling. The text detection is based on fuzzy rules and includes a method for separating overlapping characters. Finally, OCR is applied to extract the text. The recognized text is removed from the figure and a data points detection algorithm is applied next which uses a k-Median filtering. Lu et al. [LKB+09] further developed it into a retrieval engine for scientific figures in chemistry.

Mishchenko and Vassilieva [MV11] developed a framework that automatically extracts data from figures and converts them to XML. The framework consists of a figure type classifier, text and graphic component extraction and a semantic relation extraction between text and graphics. Their system supports column charts, bar charts, pie charts, line graphs and area charts which present some tabular data. The figure type classifier works on a vectorized edge image and goes through all the edges and tries to estimate to which of the aforementioned classes it most likely belongs. The text extraction is performed in three steps. First, Connected

Component Labeling is applied with some heuristic filtering to identify text candidates. The Hough Transformation is used to find text lines from the set of potential characters. Finally, the distance between different characters and the size of the candidate characters is checked to verify the text line recognition. The extracted text lines are passed to an OCR engine for recognition. The textual information is combined with the graphical elements to extract the data of a figure. Two artificially created datasets with 980 and 300 figures were used to evaluate the framework. They conclude that the text recognition quality is 20 times better than with regular OCR.

Savva et al. [SKC+11] proposed ReVision a tool to extract the data from figures to enable restyling. A bitmap image is taken as input and the figure is classified into different figure types using a Support Vector Machine. The text extraction from the figure is crowd-sourced and not automatically performed. ReVision is only capable of extracting the encoded data of bar charts and pie charts. Under the assumption of a linear mapping in the graphics representation, the graphical objects of bar charts and pie charts are associated with the extracted text components to estimate the original data. Afterwards, it is possible to present the reconstructed data with different styles. A dataset of 52 bar charts and 53 pie charts was used in the evaluation of the extraction technique and ReVision was able to recover all graphical data objects for 79% of the bar charts and 62% of the pie charts. For 71% of the bar charts and 64% of the pie charts for which all graphical data objects were correctly extracted, ReVision could also fully reconstruct the underlying data.

Al-Zaidy and Giles [AG15] presented an approach for data extraction from bar charts. Connected Component Labeling on the image in Lab color space is used to identify the graphical components of the figure. Text extraction is conducted on a binarized version of the image. Isotropic dilation is applied on the binary image before Connected Component Labeling is used to extract words which are then recognized by an OCR engine. Next, graphic and text components are combined for data extraction. The system was tested on 18 bar charts and showed only mediocre success, being only capable of recovering 87% of the bars and having problems with identifying the y-axis. Al-Zaidy and Giles [ACG16] re-used this data extraction [AG15] for generating linguistic summaries of bar charts based on protoforms, a linguistic construct based on fuzzy-logic to create sentences from structured data. They further improved their rule-based approach [AG15] by replacing the rules with a machine learning based approach [AG17] and performing a larger evaluation on a set of 213 bar charts. The highest Precision is achieved with a random forest binary classifier, while the highest Recall is obtained with a random forest multi-class classifier.

Poco and Heer [PH17] proposed a pipeline that can extract data from a bitmap image of a figure. The pipeline consists of a text analysis pipeline and a Convolutional Neural Network for mark type classification. The text extraction pipeline starts with a binarization of the image using Otsu's method. Next, a network for text pixel identification from Darknet [Red16], a CNN framework, is used to identify and remove non-text pixel. Afterwards, the remaining pixel are merged into words using Connected Component Labeling and a Minimum Spanning Tree approach. Finally, the words are recognized by an OCR engine. They evaluate their approach on three datasets. First, an artificially created dataset of 4,318 figures was created using the Vega visualization tools [SRH+16]. 475 SVGs from the news website Quartz[3] were converted to bitmap figures and manually annotated to form the second dataset. Third, 332 figures from papers of the ACL Anthology repository were selected and manually annotated for the final dataset. The approach achieves a minimum F1-Measure of 80% for text detection and recognition. For the mark type classification a minimum F1-Measure of 98% is achieved.

Cliche et al. [CRM+17] proposed an automated data extraction approach for scatter plots using neural networks. They created an artificial dataset of 25,000 scatter plots for training and 600 scatter plots for testing their neural network using a single tool, the Python Matplotlib. Additionally, to show that their approach can generalize, a small real world dataset of 50 scatter plots was collected. Their neural network object detection model is ReInspect [SAN16] which is used to find the bounding boxes for graphical objects (points, tick marks) and text (tick values). They achieve an average Precision of 87.1% for points, 99.1% for tick values and 93.0% for tick marks. The tick value bounding boxes are extracted and rotated into horizontal position using a heuristic that minimizes the area of the

[3]http://qz.com

enclosing rectangle. The rotated bounding box is then processed by an OCR engine for text recognition. They report an accuracy of 91.2% for the text recognition (63.8% if the bounding boxes are not preprocessed). The tick values are assigned to the nearest tick mark and then clustered using DBSCAN to assign them to the X- or Y-axis. A mapping from pixels to chart coordinates is performed with RANSAC regression to enable the data extraction. The overall system is capable of extracting 89% of the data from the 600 test figures, under the assumption of a 2% tolerance.

Dai et al. [DWN+18] addressed the problem of data extraction from figures with a figure type classifier and a data extraction for bar charts. First, they compared different existing Convolutional Neural Network architectures for figure type classification and came to the conclusion that GoogLeNet performs best. For the text extraction from bar charts, they reuse and modify the approach of Poco and Heer [PH17]. Next, the graphical components are extracted and matched with the text extraction for data extraction. They report that their approach can find 90% of the text on a real world corpus and extract the text with a 75% accuracy, while it finds 98% of text in the artificial corpus with a recognition accuracy of 93%. The overall data extraction accuracy is about 57% for the 59 bar charts in the real world corpus and about 82% for the 500 bar charts in the artificial corpus.

Madan et al. [MBT+18] claimed that the text extraction from figures is solved and thus focus on the analysis of the visual components of a figure. Their goal is to identify standalone icons in figures and combine them with the extracted text to create a multi-modal summary of the figure by producing visual and textual hashtags. They rely on Google Cloud Vision API for text extraction. They collected a dataset of about 29,000 figures (Visually29k) from which 544 where annotated with tags by humans and are used for evaluating their approach.

2.5 Summary

The large variety of research for text extraction from scientific figures shows that the development of a general text extraction from figures is still an open challenge, even though some claim that it is solved [MBT+18]. Over the years, the approaches evolved from heuristic-based approaches [Tak09] and unsupervised approaches [HTL03; HTL05; HT07; BS15b; BS15a; AG15; BS17; BBS18] to supervised approaches [PH17; CRM+17; APR19; SH19; MTE19; JBS19]. Nonetheless, datasets are still scarce and thus supervised approaches are just starting to overcome their limitations to specific figure types. However, even though many researchers focus on specific problems like bar charts or line graphs to reduce the complexity, most of the proposed approaches have a similar structure. They can usually be separated into two steps:

- ▻ Text extraction
- ▻ Text recognition

Most of the unsupervised approaches can be further broken down into three core components:

- ▻ Identification of (connected) components in the figure image
- ▻ Separation of text components from graphical components
- ▻ Optical Character Recognition on the text components

These steps will be used as a basis in the remainder of this thesis for creating a general pipeline for text extraction from scientific figures.

Disclaimer: This thesis focuses on research which has been published until 2019, concluding with a first attempt at text localization from scientific figures with a Faster Region Convolutional Neural Network architecture [JBS19]. In the following two years, further research into text localization and recognition from scientific figures, for example with Faster R-CNN [ZZC+21] or super-resolution deep convolutional neural networks [CYL+20] has been conducted. However, this research is out of the scope of this thesis.

Chapter 3

General Text Extraction Pipeline

3.1 Definition of the Text Extraction Pipeline

In this chapter, a general text extraction pipeline is presented that was derived from the related work in the field of text extraction from scientific figures (see Chapter 2). Please note, for describing the steps of the general pipeline the following terminology is used: The bitmap representation of a scientific figure is referred to as **image** since it is the accepted term in computer vision. A **region** is a set of pixels of an image. Each region represents either one or sometimes multiple alphanumeric characters or graphical symbols. A **text element** or text line is a set of regions representing text (e. g., a label, word, phrase) without a line-break and is defined by its position, dimension and orientation of the bounding box enclosing the contained set of regions.

The expected input of the pipeline is a (color) image and the output of the pipeline is a set of text elements. For the text extraction pipeline, three main steps were identified from the state-of-the-art analysis (compare Section 2.5):

1. Extraction of Regions
2. Identification of Text Elements
3. Text Recognition from Text Elements

This three step pipeline is an improved version of previously published definition of a general text extraction pipeline [BS15a].

For neural network based supervised approaches, the first and second step are usually not separated but intertwined. Thus, the description below applies more to unsupervised approaches.

Next, a detailed description of the three different steps of the general text extraction pipeline follows.

3.1.1 Extraction of Regions

The first step of the pipeline is to identify the interesting regions inside a figure and extract those, i. e., separate the foreground elements (text and graphical elements) from the background.

Preprocessing

In the pipeline, an image is defined as an 3D array of pixels of dimensions $width \times height \times colorchannels$. Each pixel has a color information assigned which consists of one or multiple values depending on the color space of the image. The most common color spaces are:

▻ **Binary:** A binary image is a three dimensional array where the third dimension has a length of one and each pixel value is either black (0) or white (1). An example for a binary image is Figure 3.1.

▻ **Greyscale:** A greyscale image is a three dimensional array with only one color-channel as well but each pixel can have a value between 0 (black) and 255 (white). An example is Figure 3.2.

▻ **RGB:** A three dimensional array with more than one color channel is required to store the information for an Red-Green-Blue (RGB) color image. Each pixel has three values, the red component (R), the green component (G) and the blue component (B). Each of these values ranges from 0 to 255. A black pixel has the minimum values $[0, 0, 0]$ while a white pixel has the maximum $[255, 255, 255]$.

▻ **HSV:** The Hue-Saturation-Value (HSV) color space [Smi78] differs from the RGB color space by arranging the colors in a radial slice and using the other two components for the color saturation and brightness. The Hue component has a range from 0 to 360, while the Saturation and Value components range from 0.0 to 1.0.

The proposed pipeline accepts images with any of the above mentioned color spaces. However, the region extraction process usually requires a binary image as input. Thus, given a color or grey-scale image as input, it has to be converted into one or multiple binary images.

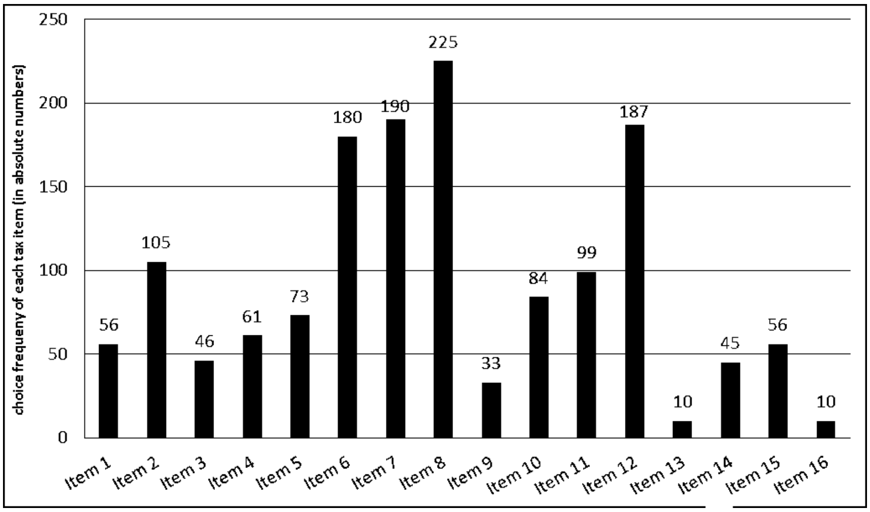


Figure 3.1. An example of a binary scientific figure

Extraction

Given a binary image, an extraction algorithm can extract interesting sets of pixels (regions) from the image which potentially depict textual characters and other graphical components. Each region is a subset of pixels and all regions are non-empty and disjoint.

Postprocessing

The generated set of regions can be filtered - if needed - to remove graphical components or background elements.

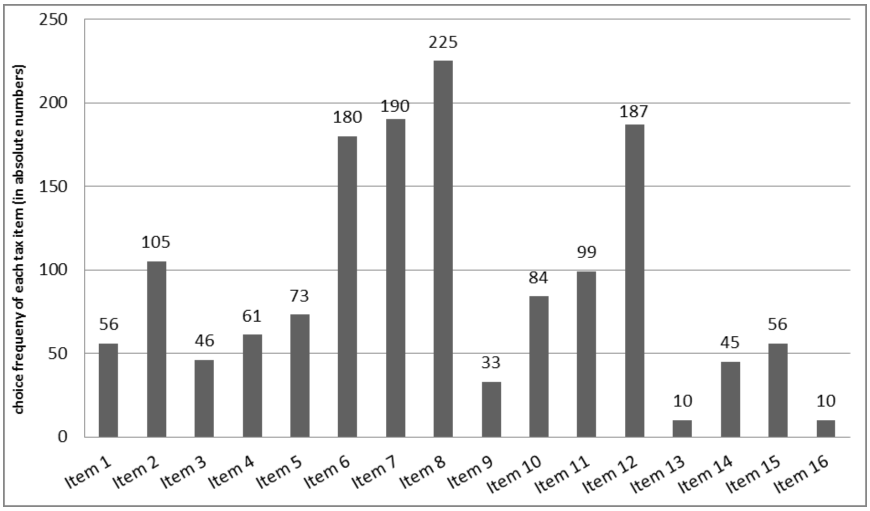


Figure 3.2. An example of a grey-scale scientific figure

3.1.2 Identification of Text Elements

The second step of the pipeline processes the extracted regions so that graphical components are separated from textual components and the latter are further processed to recognize text lines.

Feature Generation

First, for each region, features need to be identified that can be used to distinguish between text components and graphical components and should help to identify text lines. Thus, features based on the position and size of the regions are of interested so that adjacent text components can later be grouped together.

Region Grouping

Given the regions with their associated features, all the regions can be grouped to separate text components from graphical components as text components usually have unique characteristics. Each cluster is a collection of a set of regions and all cluster are disjoint. Clusters that most likely contain regions that are considered as noise/graphical elements can be dropped, using for example heuristic rules.

Text Line Identification

Many off-the-shelf OCR engines work best on horizontal text. The orientation of text can more accurately be estimated from individual text lines than from text blocks. Thus, the next step is to create a set of single line text element candidates from the text element clusters by further subdividing each cluster, if the cluster does not already comprise a text line. Each text element candidate contains a subset of the regions of a specific cluster which appear to be on a single line. All candidates for text elements are non-empty and disjoint.

3.1.3 Text Recognition from Text Elements

Given the candidates for text elements from the previous step, standard Optical Character Recognition engines can be used to recognize the text of the text element candidates. However, common OCR engines are designed to recognize text from plain document pages. Thus, most of them can only recognize horizontal text with a small tolerance margin. Therefore, different kinds of preprocessing is needed to make the text element candidates ready for recognition.

Preprocessing

Most Optical Character Recognition engines have only a small tolerance with respect to the orientation of text. Thus, for all OCR engines, it is necessary to rotate text into horizontal alignment. Every orientation angle for a text element candidate can have an integer value from -90 to 90 degree. The limitation results from the observation that text usually does not occur upside down in a scientific figure, at least not in the datasets at hand. Further, OCR-specific preprocessing might be needed like resizing the input to a certain size.

Optical Character Recognition

After rotating and preprocessing each text element candidate, the text of all text element candidates can be extracted using a standard OCR tool. The recognizable text is limited by the OCR engine, which is for this thesis the following character set: *[A-Za-z0-9!"§$%&/()=? ´°{[]}\‘+-*,.;:|’#@_-~<>€é£©®¥¢]* The output is a set of text elements, each consisting of the recognized text which is combined with its size, position and orientation.

Postprocessing

Finally, as a last step, an optional postprocessing can be applied on all text elements to mitigate recognition errors and to remove noise. This postprocessing either updates the text of each text element or removes it completely. It does not alter the position or size of a text element as no information between regions and recognized text exist. Several approaches for correcting text extraction results have been proposed in the past like for example, methods using dictionaries [SRS+03], methods using the structure of sentences [KST+96], or methods that use datasources from the web like Google [BA12a; BA12b]. However, due to the special characteristics of text in scientific figures, which are often very sparse and contain abbreviations and numbers, it is difficult to apply these methods and more simplistic methods without external input are required.

3.2 Methods of the Text Extraction Pipeline

The previous section described a general pipeline for text extraction from scientific figures. The following sections explain in detail the methods that were implemented for the different steps of the text extraction pipeline in this thesis. The structure follows the steps of the pipeline that were outlined in the previous Section 3.1.

3.2.1 Unsupervised Extraction of Regions

Three parts were identified in Section 3.1.1 for the extraction of regions from images: Preprocessing, Extraction and Postprocessing. In the following, a selection of preprocessing methods for color conversion and binarization are presented in Section 3.2.1. The presented methods are those most commonly used in the literature for generating binary images. Different methods for the extraction of regions from binary images are presented in Section 3.2.1 as well. No postprocessing method on the raw regions is used but postprocessing on the regions based on the features generated for each region (see Section 3.2.2).

Preprocessing

Given a color image, two common methods exist to reduce the number of color channels from three channels down to one channel. The first option is to convert the color image to grey-scale which can then be binarized. The second option is to split the image up into multiple images to deal with the different colors separately.

Conversion to Grey A common way to ease the processing of color images, is to convert them to grey-scale images. Different methods exist to perform such a conversion.

Color images are often represented in the RGB color space and thus have three color components (red, green and blue). Other color images that use for example the HSV color space can be converted to the RGB color space using the following equations:

$$C = V \cdot S \tag{3.2.1}$$

$$X = C \cdot (1 - |(\frac{H}{60^\circ} \bmod 2 - 1|) \tag{3.2.2}$$

$$m = V - C \tag{3.2.3}$$

$$(R', G', B') = \begin{cases} (C, X, 0) & , 0^\circ \leqslant H < 60^\circ \\ (X, C, 0) & , 60^\circ \leqslant H < 120^\circ \\ (0, C, X) & , 120^\circ \leqslant H < 180^\circ \\ (0, X, C) & , 180^\circ \leqslant H < 240^\circ \\ (X, 0, C) & , 240^\circ \leqslant H < 300^\circ \\ (C, 0, X) & , 300^\circ \leqslant H < 360^\circ \end{cases} \tag{3.2.4}$$

$$(R, G, B) = ((R' + m) \cdot 255, (G' + m) \cdot 255, (B' + m) \cdot 255) \tag{3.2.5}$$

Grey-scale images have only one color component. Thus, when transforming from color to grey scale, the three color components need to be combined into one component. One simple method is to calculate the grey value by giving each component the same weight:

$$Y = 0.\overline{33}R + 0.\overline{33}G + 0.\overline{33}B \tag{3.2.6}$$

Another popular method it to use a Luminance preserving method based on the principles of photometry, which adjusts the weights of the components to better mimic the human perception:

$$Y = 0.2126R + 0.7152G + 0.0722B \tag{3.2.7}$$

Since the pipeline is designed for processing figures, which are meant to be read by humans, the conversion to grey scale based on Equation 3.2.7 is selected.

Binarization Several methods [Ots79; JB82; KSW85; KI86; Nib86; Ber86; SSH+97; GPP06; BAO11] exist for binarizing a grey-scale image. To limit the number of options, three of those methods - namely Otsu's algorithm, given its popularity, Niblack's algorithm, because it computes a binarization threshold per pixel, and a newly developed, hierarchical, adaptive Otsu algorithm - were chosen and those methods are explained in more detail below.

Otsu's algorithm [Ots79] is a common method which uses a single threshold to binarize an image. Set to a specific grey value, all pixels with a value above the threshold are considered as foreground and all pixels with a value below the threshold are considered as background. However, finding the right threshold can be difficult. Otsu proposed a method to automatically determine the 'best' threshold for binarization [Ots79]. Otsu's algorithm finds the binarization threshold by maximizing the interclass variance. This threshold is then applied on the whole image for binarization.

The standard Otsu algorithm applies only one global threshold on the entire image. However, images often contain local inhomogeneities and varying color combinations for text and background. Thus, a global threshold may not be the best solution. Therefore, a modified version of Otsu's algorithm, which is called **Adaptive Otsu** in this thesis, was developed [BS15b] which computes several thresholds based on how detailed certain parts of an image are. The algorithm starts by calculating the threshold using Otsu's algorithm on the whole image and then subdividing the image into four equal-sized parts with a quadtree like datastructure. A quadtree is a two dimensional data structure which recursively subdivides the space into four tiles. The 'leaf'-regions contain a certain, interesting spatial information. Here, the algorithm recursively further subdivides the image when necessary to fine-tune the thresholds, i. e., that each region is homogeneous enough to be binarized by a single threshold. On each tile that was binarized using Otsu's algorithm, the popular Sobel operator [Sob14] is applied to determine the edges. The Hausdorff distance [Hau14] is computed over the edges of the current tiles and their parent tile. A tile is further subdivided if a specified empirical determined threshold for the Hausdorff Distance is not reached. The final threshold for each leaf tile is computed by averaging the thresholds estimated by Otsu's algorithm over all tiles in the hierarchy up to the root.

In contrast to Otsu's algorithm, **Niblack's algorithm** [Nib86] calculates an individual threshold per pixel. This is achieved by using a sliding window approach that takes the mean grey value m of the pixels in the sliding window and their standard deviation s into account when calculating the threshold:

$$T_{Niblack} = m + k \cdot s \tag{3.2.8}$$

There parameter k is usually set to -0.2.

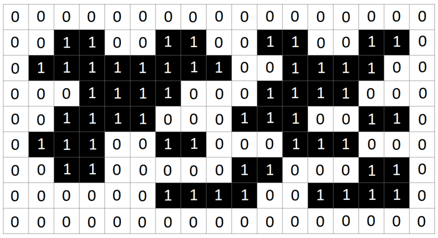
Color Quantization A different approach is to identify the different colors in an image and to create binary images for each of these colors where in each image one color is considered as the foreground and everything else is background. However, given that there is a multitude of colors possible in an image and not all of them should be considered as individual colors, a so-called color quantization can be performed to reduce the color spectrum of an image to the most prominent ones. The following approach is motivated by the work of Chiang and Knoblock [CK15], Fraz et al. [FSE15] and Jayant et al. [JRW+07] and is mainly based on the approach of Chiang and Knoblock [CK13].

The color quantization approach takes a color image as an input and first transforms it into the HSL color space. A Mean-Shift algorithm [CM02] is applied to preserve edges and remove noise, followed by a Median-Cut algorithm [Hec82] and k-Means clustering [Llo82] to identify the main colors (see Appendix B for details on Mean-Shift, Median-Cut and k-Means). For each cluster, the cluster center is chosen as the representative color and a new image is created where each pixel's color is replaced by the representative color of the cluster it was assigned to. Finally, the image is split into k binary images where k is the number of representative colors/-clusters as specified by the k-Means algorithm. The default setting for k is 8. Each of the binary images contains only one of the colors (replaced by black pixels) and the remaining pixels are set to white (background).

Extraction

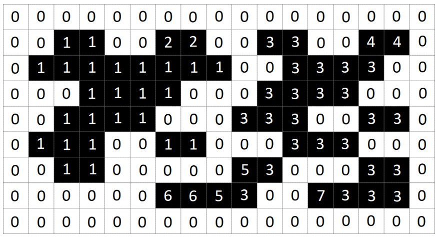
The extraction of regions is the next processing step after binarizing the image, i. e., the foreground pixels in the binary image need to be analyzed and combined into regions which may represent text elements. In the following, two methods are presented that can create regions from foreground pixels in binary images.

Connected Component Labeling The most common approach for identifying regions in a binary image is the so-called Connected Component Labeling (CCL) [ST88]. The CCL algorithm iterates over the whole image and assigns each pixel to a region by taking the assignment of the adjacent pixels into account (4-/ 8-pixel-neighborhood). In this thesis the 8-pixel-neighborhood is used. Figure 3.3 illustrates the process. The algorithm starts at the pixel in the top left corner - the origin [0,0] - and iterates through the image row-wise. If a pixel belongs to the foreground, i. e., it is black in the binary image, its neighbors that were already visited are checked to see whether they belong to a connected component/region. If the neighbor pixels belong to the same component, then the current pixel is assigned to that component as well. If the neighbor pixels are assigned to different components, then the current pixel is assigned to the component with the smallest index and mappings between the components of the neighbors are stored in the so-called mapping table.



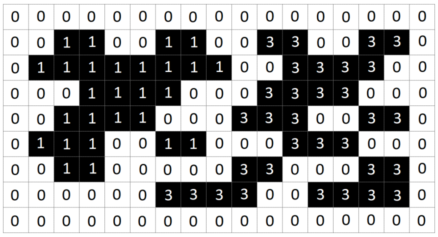
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
0	1	1	1	1	1	1	1	1	0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0
0	0	1	1	1	1	0	0	0	1	1	1	0	0	1	1	0
0	1	1	1	0	0	1	1	0	0	0	1	1	1	0	0	0
0	0	1	1	0	0	0	0	0	1	1	0	0	0	1	1	0
0	0	0	0	0	0	1	1	1	1	0	0	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(a) Binary image with black foreground pixel (1) and white background pixel (0).



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	2	2	0	0	3	3	0	0	4	4	0
0	1	1	1	1	1	1	1	1	0	0	3	3	3	3	0	0
0	0	0	1	1	1	1	0	0	0	3	3	3	3	0	0	0
0	0	1	1	1	1	0	0	0	3	3	3	0	0	3	3	0
0	1	1	1	0	0	1	1	0	0	0	3	3	3	0	0	0
0	0	1	1	0	0	0	0	0	5	3	0	0	0	3	3	0
0	0	0	0	0	0	6	6	5	3	0	0	7	3	3	3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(b) Labeled image after the first pass. The list of adjacent labels is stored separately.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	0	0	3	3	0	0	3	3	0
0	1	1	1	1	1	1	1	1	0	0	3	3	3	3	0	0
0	0	0	1	1	1	1	0	0	0	3	3	3	3	0	0	0
0	0	1	1	1	1	0	0	0	3	3	3	0	0	3	3	0
0	1	1	1	0	0	1	1	0	0	0	3	3	3	0	0	0
0	0	1	1	0	0	0	0	0	3	3	0	0	0	3	3	0
0	0	0	0	0	0	3	3	3	3	0	0	3	3	3	3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(c) Final results after merging of adjacent labels.

Figure 3.3. Connected Component Labeling with 8-pixel-neighborhood.

If none of the neighbors are part of a component, i. e., all visited neighbors are background pixel, then a new component is introduced. After the first pass through the image, a second pass is conducted to merge adjacent components based on the mapping table.

Pivoting Region Extraction Xu and Krauthammer [XK10] proposed a pivoting histogram projection approach to extract regions from images. Here, first an edge image of the binary image is computed. The edge image's pixels are alternately projected on the x and y-axes and the image is split after every projection at the minimal point(s) of the histogram into multiple sub-images. Each sub-image is further processed while alternating the direction until no further split is possible. Thus, one obtains multiple rectangular sub-images, which are converted into regions by taking all foreground pixels of each sub-image from the binary image and thus create one region per sub-image.

3.2.2 Unsupervised Identification of Text Elements

The second step in the pipeline is to identify text elements in the image as defined in Section 3.1.2. First, most methods require an aggregated view of the regions, i. e., it is necessary to compute features for the regions that were extracted from the binary image(s). The feature generation that is used in this thesis for unsupervised text extraction is described below. Second, the regions are grouped using the computed features to separate text elements from graphical elements. Different methods for this task are presented in the following. Finally, if the previous step has not already created them, individually text lines need to be extracted.

Feature Generation

Different features can be envisioned for describing the regions. They should characterize the regions with respect to their position and their size and shape as this is often a criteria for differentiating between text elements and graphical elements. Here, some of the so-called image moments [Hu62] are used as features which are a popular, simple method in computer vision.

For each region, a five-dimensional feature vector is computed. Given the pixel-based representation of the regions, a center-based representation of the location of the region is computed using the image moments:

$$m_{pq} = \sum_x \sum_y x^p y^q \Psi \quad \text{with} \;\; p,q = 0,1,2,\ldots \tag{3.2.9}$$

Please note that p, q hereby denote the p, q^{th} moment. For binary images, Ψ takes the values 0 or 1 and therefore only pixels contained in a region are considered for the computation of the moments. Using the first-order moments, one can compute each regions center of mass (m_{10},m_{01}):

$$m_{10} = \sum_x \sum_y x^1 y^0 \tag{3.2.10}$$

$$m_{01} = \sum_x \sum_y x^0 y^1 \tag{3.2.11}$$

These are the first two components of the feature vector. The third and fourth component are the width w and height h of the bounding box of the region and the final and fifth component is the area-occupation-ratio aor which is simply computed as the number of pixels of a region divided by the number of pixels of the enclosing bounding box. Thus, the full feature vector is:

$$\vec{v} = \begin{pmatrix} m_{10} \\ m_{01} \\ w \\ h \\ aor \end{pmatrix} \tag{3.2.12}$$

Having computed the feature vectors for all regions, simple heuristics [BS15b; SZ13] are used to postprocess the set of region to filter obvious graphical elements. All regions that fulfill the following constraints are discarded as they most likely compose graphical components: (a) Either width w or height h of the region's bounding box are above the average width/height plus 3 times standard deviation (e. g., axes) or (b) the bounding box is smaller than 0.001% of the figure's size (noise) as well as (c) elements occupying more than 80% of their bounding box (e. g., legend symbols).

Region Grouping

The regions with their feature vectors need to be further grouped to separate the remaining graphical components from the text components and to create an initial grouping of the text components (characters) into word-like or text-line-like structures. Four methods are considered here: Gravity-based grouping, Minimum Spanning Tree clustering, morphological grouping and Density-Based Spatial Clustering of Applications with Noise (DBSCAN).

Gravity-based Grouping Huang et al. [HTL05] proposed grouping rules based on Newtons gravity equation from classical physics:

$$G = \frac{C_1 \cdot C_2}{r^2} \tag{3.2.13}$$

C_1 and C_2 are hereby the size of the two components which are considered for grouping together and r is the distance between them. The formula computes a value which is compared with a threshold which defines whether two regions are grouped together. An empirical determined threshold of 20 was used in this thesis as no information was provided by Huang et al. [HTL05]. The connected regions form text elements, each of which - according to Huang et al. - represent a single text line.

Minimum Spanning Tree Clustering Another clustering algorithm, the Minimum Spanning Tree (MST) clustering, has also been used for the task of region grouping [JRW+07] (see also Appendix B). First, a Minimum Spanning Tree is constructed on the data points. Second, the MST is split up and the clusters are formed from the connected data points. There are multiple possibilities on how to split a tree like removing the longest edges, removing edges whose length are above a certain threshold or inconsistent edges. Here, a split at inconsistent edges is used, i. e., edges that are longer than the local average edge length are removed.

Morphological Grouping Chiang and Knoblock [CK15] proposed a method based on morphological operations to merge characters into words on a pixel level and thus do not need the features mentioned in Section 3.2.2. The Conditional Dilation Algorithm (CDA) iterates multiple times over the input image with an alternating evaluate-and-expand mechanic. Initially, all regions are marked as expandable and all background pixels that have a pixel of a region in their neighborhood are marked as expansion candidates if they connect at most two regions of similar size and similar baseline orientation. The second scan validates the expansion candidates again because during the first scan the expansion candidates themselves are not considered as belonging to a region. The remaining expansion candidates are converted to foreground pixels and are added to the regions. This process is repeated until no region can be expanded anymore. A region is not a valid expansion target if it is already connected to two other regions and/or it was extended to its maximum which was defined by Chiang and Knoblock as $^1/_5$ of the region size. After completion of the CDA, regions that got connected belong to the same cluster/are grouped together.

Density-based spatial clustering of applications with noise (DBSCAN) Different types of clustering exist and DBSCAN [EKS+96] (see Appendix B for a general description of DBSCAN) is as the name suggest a density-based clustering which is well suited for the task of separating text regions and graphical regions since text is usually more dense than the graphical components. DBSCAN has been successfully applied for this task in the past [BS15b]. Given that the feature vector includes the size and position of the regions, regions that are of similar size like characters which are located close to each other are grouped together. Graphical elements which appear in a less structured pattern and are usually of different size than text can often be seen as individual noise elements or form groups that have different characteristics than text. The standard DBSCAN algorithm is quite simple and only requires two parameters: ϵ which specifies the size of the neighborhood and the number of minimal points that are required for a point to be a cluster point. The DBSCAN algorithm starts at any data point (i. e., feature vector), evaluates whether it is a so called core point of a cluster (i. e., it has at least the minimal number of points in its ϵ-neighborhood) and if that is the case then it collects all the points in the neighborhood, adds them to the cluster and evaluates those in the same way. Thus, the clusters are build iteratively until no more neighbors are found. The algorithm then continues with another random data point that has not yet been processed or terminates otherwise. If a data point is not a core point and is not contained in the neighborhood of a core point, then it is marked as noise and the algorithm continues with the next point. The difficulty is to properly set the ϵ parameter since the size of the images and therefore the size of text vary which influences the range of values of the feature vector. Since objects of approximately the same size shall be grouped together, an adaptive ϵ-parameter is used which is calculated based on the size of the current region.

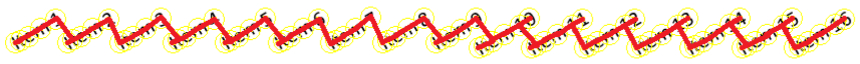
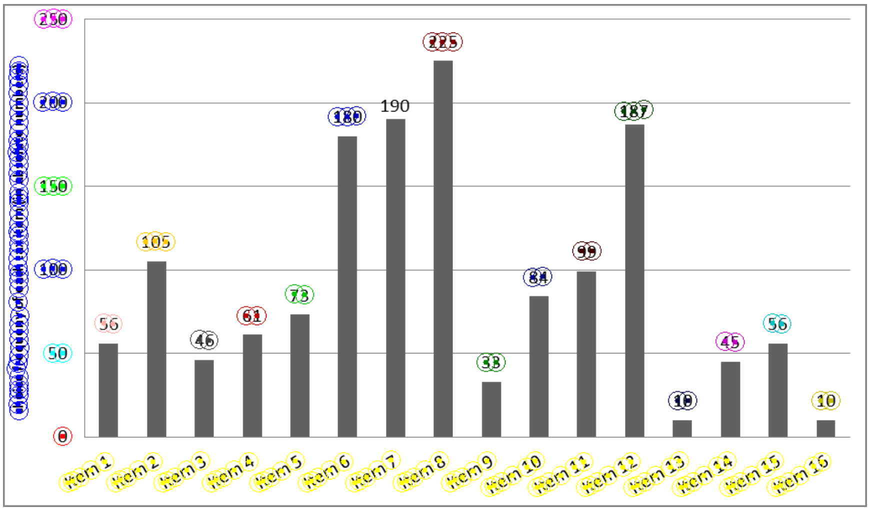


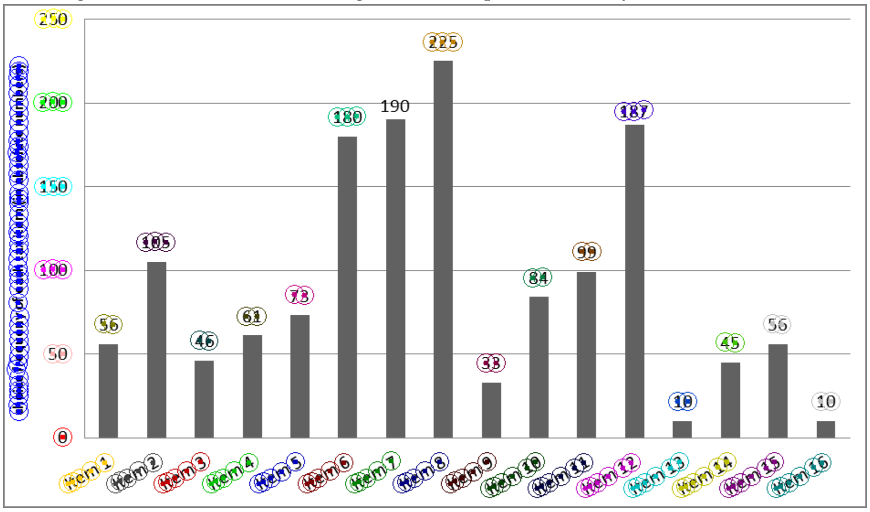
Figure 3.4. A block of rotated text with the corresponding MST on the center of mass of the characters to split the block into multiple text lines.

Text Line Extraction

Since not all grouping methods necessarily create individual text lines but rather blocks of text, a further subdivision of these clusters was proposed [BS15b] to create individual text lines. The proposed method is a Minimum Spanning Tree clustering which is computed on a reduced feature vector, i. e., only on the coordinates of the centers of mass of the regions of a cluster. The assumption is that characters of the same text line are closer to each other than characters from different text lines. Thus, most edges in the Minimum Spanning Tree will have a similar orientation along the main direction of each text line, assuming they all have the same orientation, while far lesser edges connect across the different text lines (see Figure 3.4). Edges connecting different lines will have orientations that differ strongly from the main orientation and can therefore be easily detected and removed. Figure 3.5 gives an example for the usefulness of this step.



(a) An example figure with the overlayed result from DBSCAN clustering. Several text lines are merged which should not be merged, for example the x- and y-axis labels.



(b) An example figure with the overlayed clustering results after a second iteration with the angle-based minimum spanning tree clustering. Text lines are now correctly separated.

Figure 3.5. An example of the necessity of text line extraction.

3.2.3 Supervised Extraction of Text Elements

In recent years, the most successful and promising supervised approaches for image analysis are based on neural networks. In this thesis, the focus is on one specific neural network architecture, the Faster R-CNN Object Detection Network [RHG+15] (also see Appendix B), which was lately especially successful in image processing tasks and is depicted in Figure 3.6.

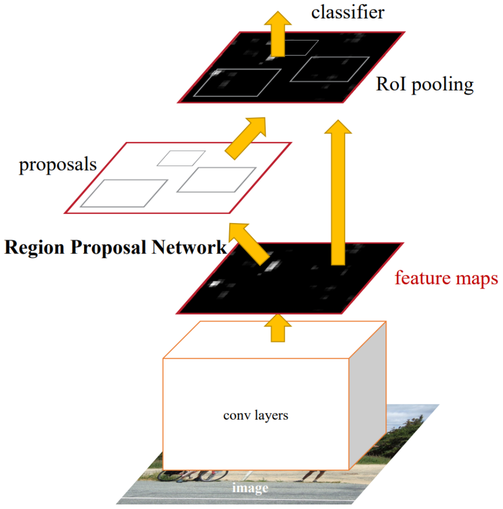


Figure 3.6. The Faster R-CNN Object Detection Network [RHG+15].

It consists of a Convolutional Neural Network (CNN) which feeds a region proposal network and both feed the final classifier. However, some changes are made to the originally proposed version. Instead of the CNN, a Residual Neural Network is used and the RoI Pool is replaced with RoI Align. More details on CNN, RoI Pool and RoI Align can be found in Appendix B.

A Residual Neural Network (ResNet) is a neural network with skip connections that jump over some layers. ResNet was developed by He et al. [HZR+16] to address the degradation problem of deep neural networks. They addressed the problem that with increasing network depth, the accuracy gets saturated and then quickly degrades. Thus, they let the network explicitly fit a residual mapping $F(x) = H(x) + x$ which can be reformulated as $H(x) = F(x) + x$ instead of $H(x)$. They hypothesize that it is easier to optimize the residual mapping than the original one. Practically, $F(x) + x$ can be realized adding shortcuts (skip connections) to the feedforward neural network. As with all neural networks, the input dimensionality (width) and the depth of the network are parameters that should be optimized.

ResNeXt is a modified version of ResNet proposed by Xie et al. [XGD+17]. It alters the ResNet architecture by adding a cardinality parameter which splits the network layers into parallel blocks. Thus, ResNeXt has three parameters to optimize: width, depth and cardinality.

In summary, the following steps are performed for supervised text extraction:

1. Pass Image to ResNet/ResNext
2. Extract feature maps from Feature Pyramid Network
3. Pass feature maps to Region Proposal Network (RPN)
4. Pass proposals from RPN by applying RoI Align to classifier to get bounding boxes that are classified as containing text

3.2.4 Text Recognition from Text Elements

Having separated graphical components from textual components and having identified individual text lines in the figure, the next step is to recognize the text from the extracted text lines to convert it into a machine readable format. Three popular OCR engines are used in this thesis: Tesseract, Ocropy and ABBYY FineReader. The methods for preprocessing of the input for the OCR engines, the OCR engines themselves and the postprocessing of their output are described below.

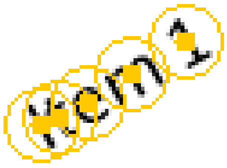
Preprocessing

Standard Optical Character Recognition engines often require that the text of the input image has a approximately horizontal orientation as they are mostly tuned for recognizing document pages. Thus, the single text lines produced in the second step of the pipeline need to be rotated into horizontal alignment if they have an orientation that differs too much from it. Different options exist for computing the orientation of a text line and three of those methods from the related work are presented here. Subsequently, text lines are rotated into the opposite direction to bring them into horizontal alignment.

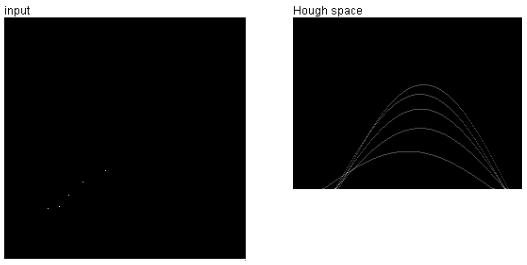
Depending on the OCR engine, further preprocessing might be necessary. For example, Tesseract works best when the text has some space to the edges of the image, i. e., the bounding box is not too cropped. Thus, each image is extended with a border of 5 pixels which has the background color. Ocropy requires the input to have a certain size and thus, the input image is modified to fulfill the required thresholds by scaling up the input images if they are too small.

Perpendicular Squared Distance Method Jayant et al. [JRW+07] proposed the PSD method to approximate the orientation of a text line by minimizing its bounding box. Their method approximates the angle by computing the perpendicular least square fit, i. e., the line that minimizes the Perpendicular Squared Distance from the pixels to the line. To achieve this, the text line is rotated with different angles until the minimum is found.

Single String Orientation Detection The Single String Orientation Detection (SSOD) method proposed by Chiang and Knoblock [CK15] is based on the RLSA method [Naj04] which is used for skew correction of documents with multiple lines of text. Their method assesses different orientation candidates from 0 to 179 degree by rotating the text elements and applying morphological operations on the regions. The morphological closing operator (see Appendix B) is used with a structure-element that has the width of the average character size to merge the character in horizontal direction into a single object. Next, the morphological erosion operator is used to prune pixels of the object that have background pixel in their neighborhood which is defined via a distance threshold that is computed from the maximal string length. The orientation with the largest pixel area remaining after applying both operators is chosen as the orientation.



(a) A text line element with its center of masses highlighted.



(b) The text line element transformed into Hough space.

Figure 3.7. The center of mass of a text line are transformed into Hough space.

Hough-based Orientation Estimation Other researchers [BS15b; DWN+18] have used a method to calculate the orientation based on the Hough transform [DH72]. The Hough transform was originally used to detect lines in images by transforming data points into a parameter space. Lines in an image are represented as points in the parameter space while points in an image are represented as sinusoidal curves in the parameter space. For line detection, the (r, θ) parameter space is used where $r = x\cos(\theta) + y\sin(\theta)$ is the distance from the origin and θ is the angle between the x-axis and a line that is orthogonal to a line in the image and connects it to the origin of the image. The maxima in the Hough parameter space refer to the most likely parameter sets for lines in the image. Here, the center of mass coordinates of the character regions of a text line are transformed into the Hough parameter space to find the maximal value which represents the main orientation (see Figure 3.7). The Hough space computation is limited to a 180 degree interval since text lines can have only an orientation between -90 and +90 degree.

Optical Character Recognition

Standard Optical Character Recognition engines can be used to extract the text from the preprocessed subimages of single, horizontal text lines. Two

Table 3.1. Page Segmentation Modes of Tesseract.

0	Orientation and script detection (OSD) only.
1	Automatic page segmentation with OSD.
2	Automatic page segmentation, but no OSD, or OCR.
3	Fully automatic page segmentation, but no OSD.
4	Assume a single column of text of variable sizes.
5	Assume a single uniform block of vertically aligned text.
6	Assume a single uniform block of text.
7	Treat the image as a single text line.
8	Treat the image as a single word.
9	Treat the image as a single word in a circle.
10	Treat the image as a single character.
11	Sparse text. Find as much text as possible in no particular order.
12	Sparse text with OSD.
13	Raw line. Treat the image as a single text line, bypassing hacks that are Tesseract-specific.

open-source OCR engines - Tesseract [Smi07] and Ocropy[1] - are used in this thesis as they are freely available and frequently updated. The commercial OCR engine ABBYY FineReader is selected for comparison.

Tesseract The OCR engine Tesseract provides pretrained models for different languages. In this thesis, the pretrained model for the English language is used without any adjustments, as the focus of this thesis is not to improve OCR of a specific engine. Tesseracts also offers some layout analysis capabilities for removing graphic elements and conducting text line detection. However, Tesseract is designed for analyzing document images which are fundamentally different from scientific figures. Thus, the layout analysis of Tesseract is deactivated in the experiments to follow. Tesseract has fourteen so called Page Segmentation Modes[2] (see Table 3.1) which influence the recognition accuracy of Tesseract. In this thesis, since

[1]`https://github.com/tmbdev/ocropy`, last access: November, 2018

[2]Source: `https://tesseract-ocr.github.io/tessdoc/ImproveQuality#page-segmentation-method`, last access: October, 2020

it provided the best results, the Tesseracts Page Segmentation Mode is set to 'Single Textblock' to cover as many input types as possible.

Ocropy Ocropy is a collection of open source tools for document analysis and Optical Character Recognition. It is designed for character recognition from full-page documents like Tesseract. However, Ocropy has several integrated constraints on parameters like minimum for image width and minimum for image height, in order to assure good results. Like Tesseract, the OCR engine Ocropy provides a pretrained model for the English language which is used in the experiments.

ABBYY The ABBYY FineReader is a widely used commercial Optical Character Recognition engine. Similar to Tesseract, ABBYY FineReader is able to recognize several languages, but only the English language model is used here. A time-limited, free-of-charge developer license for the ABBYY FineReader 11 was provided by ABBYY for the experiments.

Postprocessing

As recognition errors can happen, it can make sense to apply postprocessing on the output of the OCR engines. In the following, three postprocessing methods are presented that do not make assumptions about the text and do not need external knowledge.

Character Filtering A common indicator for recognition problems is the presence of many "dirty" symbols, i. e., special characters. One method is to simply remove all special characters, i. e., all characters that are not a white space, numbers, or characters from a-z or A-Z, from the recognized strings and evaluate only the remaining characters. This approach makes sense as recognition errors often result in special characters and removing only those should improve the results while only losing a handful of correct characters.

String Filtering A simplified version of the method that Sas and Zolnierek [SZ13] proposed is considered as well. It completely removes text lines if they contain too many special characters given a certain threshold. Thus, strings containing few special characters are kept while strings with many special characters, which are most likely noise, are removed.

Quantitative Assessment Chiang and Knoblock [CK15] proposed a method that uses the number of components NCC that went into the OCR process and the number of recognized components NRC as well as the number of suspicious components NSC to calculate a recognition confidence:

$$RC = \frac{NRC - NSC}{NCC} \tag{3.2.14}$$

Text lines with a recognition confidence below 50% are discarded. However, since not all OCR engine provide the number of suspicious characters, an adapted version is used that only relies on NCC and NRC. Chiang and Knoblock also ignored the number of suspicious characters for longer text lines.

3.3 Configurations of the Text Extraction Pipeline

Given the different methods described in the prior section, one can combine them in various ways to create so-called pipeline configurations.

3.3.1 Unsupervised Pipeline Configurations

For the unsupervised approach, some methods are restricted in how they can be combined as illustrated in Figure 3.8. Still, more than 24,000 configuration of the general pipeline (see Section 3.1) are possible given the methods defined in Section 3.2 (excluding the supervised methods).

Given color images of scientific figures as input, one could build the following, exemplary, unsupervised configuration to extract text from these figures: First, one needs to select some preprocessing methods to convert the color images into binary images. For example, one can select the RGB to Grey conversion and binarization with Otsu for this step of the pipeline. Next, one could choose the Connected Component Labeling for region extraction and then apply the morphological grouping to create text lines from those regions. As preprocessing for the text recognition, a calculation method like the Hough-based method is needed to estimate the orientation of each text line. With this information, the text lines can be rotated into horizontal orientation for text recognition. And thus, finally, one could use the Tesseract Optical Character Recognition engine to extract the text. Figure 3.9 shows this configuration.

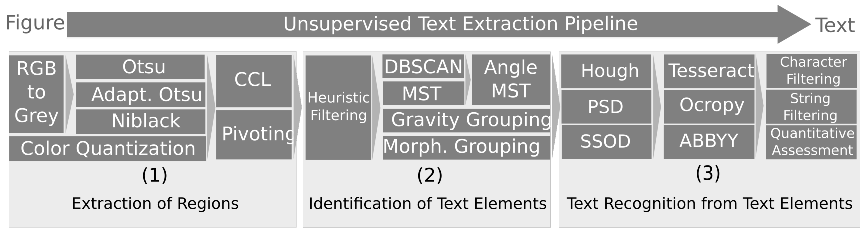


Figure 3.8. An overview of the different possibilities to combine the methods of the unsupervised text extraction pipeline into configurations.

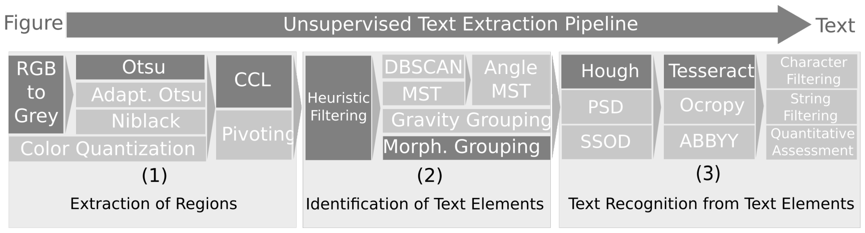


Figure 3.9. One possible configuration of the unsupervised text extraction pipeline.

The specific configurations that are used in the experiments are described in the respective experiment chapters.

3.3.2 Supervised Pipeline Configurations

For the supervised approach, the pipeline steps were reduced to two steps by aggregating the region extraction and text identification step into one text extraction step which will be performed by the complex neural network approach, which consists of multiple individual neural networks. Figure 3.10 shows this configuration which was build and evaluated by Morten Jessen [JBS19].

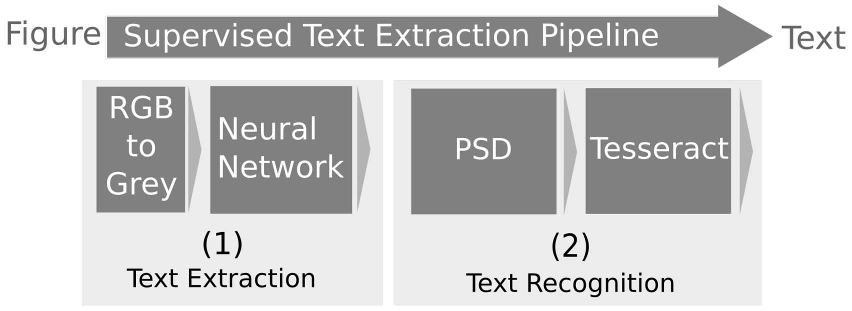


Figure 3.10. The supervised text extraction pipeline.

For the supervised approach, only the architecture of the first neural network and the hyperparameter are being optimized. The details can be found in the experiment description in Chapter 8. The only other variable is the pretraining of the model and augmentations of the training data and these details can be found in the experiment description in Chapter 8 as well.

Chapter 4

Experimental Apparatus for the Evaluation of the Modular Text Extraction Pipeline

In this chapter, the basis for all the following experiments are described. This includes the datasets that are used as well as the evaluation measurements.

4.1 Datasets

This section presents and describes the datasets that were used in the experiments for evaluating the text extraction from scientific figures. First, in Section 4.1.1, the datasets that already existed are presented. Second, in Section 4.1.2, the manually created datasets and the process of how they were created are described.

4.1.1 Existing Datasets

The following existing datasets have been used in the experiments of this thesis. The first two datasets contain scientific images, while the latter two are natural images with scene text which have been used to pretrain the supervised models.

CHIME

The Chart Image Dataset[1] was created by members of the research group of Prof. Chew Lim Tan (retired since June 2016) of the National University of Singapore and consists of two sets of figures.

CHIME-R The first dataset consists of 115 real images that were collected on the Internet or scanned from paper and is named CHIME-R in the remainder of this thesis. The majority of the figures (75) are bar charts. The remaining figures are pie charts (25) and line graphs (15). The gold standard for the figures in the CHIME-R dataset was created by Yang Li [YHT06]. Figure 4.1 shows a few example images of the CHIME-R dataset. All text elements are annotated with their position and content but not with an orientation.

CHIME-S The second dataset, which is named CHIME-S, consists of 85 synthetically generated images. This set mainly consists of line graphs (45) and pie charts (35) and has only five bar charts. The gold standard was created by Zhao Jiuzhou [Jiu06]. Figure 4.2 shows a few example images of the CHIME-S dataset. Similar to the CHIME-R dataset, all text elements are annotated with their position and content but not with an orientation.

DeTEXT

The DeTEXT dataset [YYP+15] consisted of 500 human-annotated, biomedical figures. However, the original dataset does not seem to be available anymore. But it was used as part of the ICDAR Robust Reading Challenge 2017 [YYY+17] from which a subset of 192 biomedical figures is available which are used here. The dataset consists of the usual variety of figure types, i. e., bar charts, pie charts, scatter plots. In addition it contains medical figures (real-life diagrams and abstracted diagrams) as well as figures which contain subfigures with various figure types. Figure 4.3 shows a few examples.

[1] `https://www.comp.nus.edu.sg/~tancl/ChartImageDataset.htm`, last access: September, 2017 (The website is nowadays unavailable after the retirement of Prof. Chew Lim Tan)

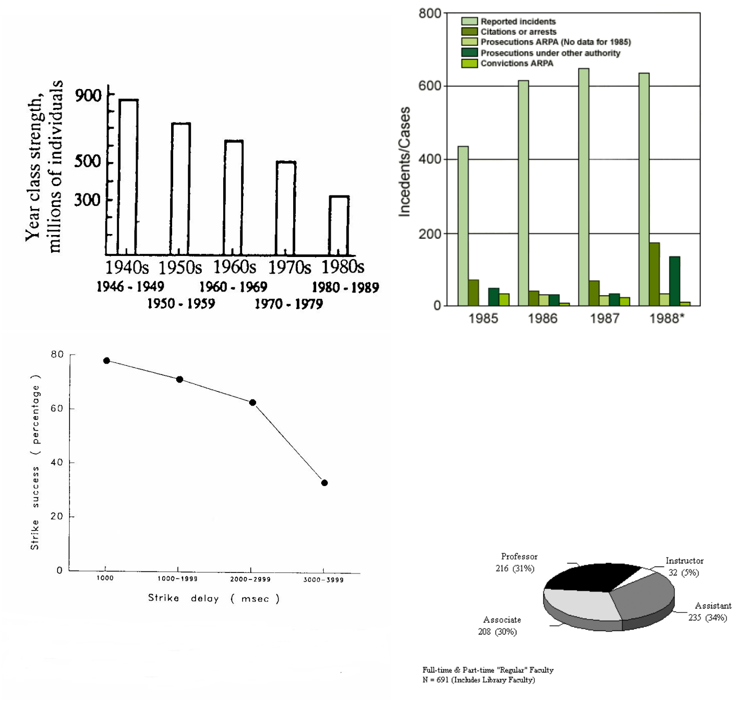


Figure 4.1. Four example figures from the CHIME-R dataset.

Only text elements which have at least one word with a length of minimum two characters are annotated, e. g., single characters are not included. For all text elements, the location, ground truth text and orientation is given.

MS-COCO-Text

One of the datasets that is used for pretraining the supervised neural network approach is the MS-COCO-Text dataset in Version 1.3 [VMN+16]. The MS-COCO-Text dataset is a subset of the MS-COCO dataset which consists of natural images. The subset of 63,686 images is annotated with 145,000 text annotations. For pretraining, a subset of 17,237 images with

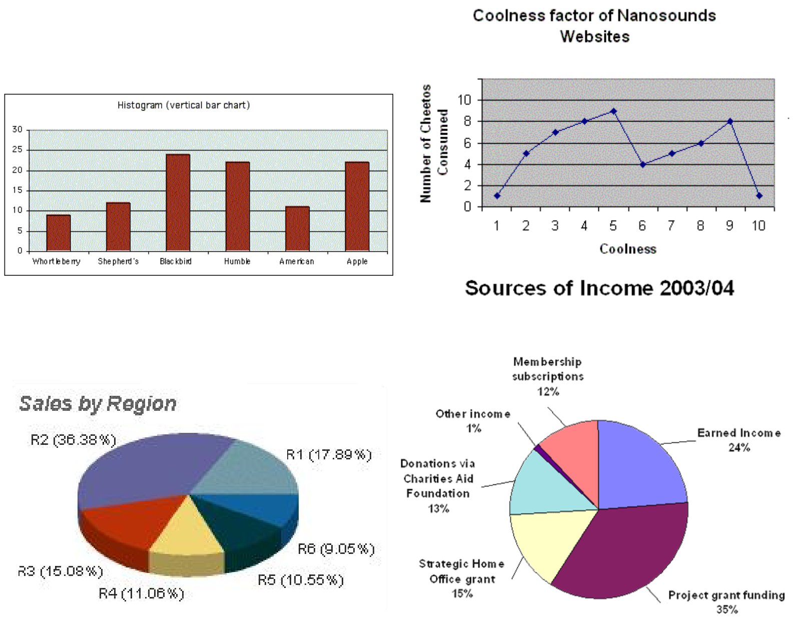


Figure 4.2. Four example figures from the CHIME-S dataset.

English, machine-written text annotations were used. Figure 4.4 shows a few examples.

Total-Text-Dataset

The Total-Text-Dataset [CC17] is the other dataset that is used for pre-training the supervised neural network approaches. It is similar to the MS-COCO-Text dataset as it contains natural images with text annotations as well. The 1,555 images have in total 11,460 text annotations which can be rotated or curved. Figure 4.5 shows a few examples.

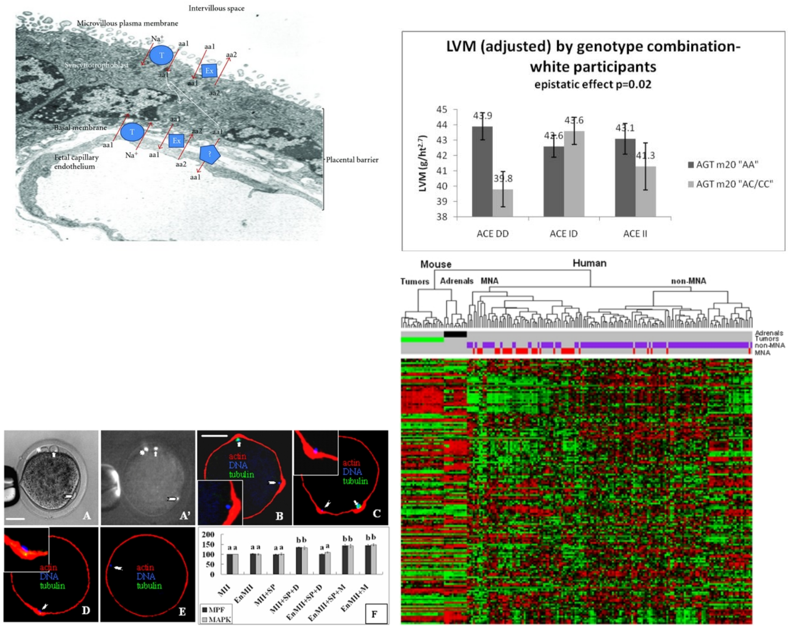


Figure 4.3. Four example figures from the DeTEXT dataset.

4.1.2 New Datasets

The Chart Image datasets do not contain any information about the orientation of text elements even though a considerable amount of text elements in scientific figures can be rotated. Thus, two datasets were manually created to be able to evaluate more accurately the text extraction from scientific figures which may contain rotated text elements. Next, the tool and process that were used to annotate the text elements in the figures of the new datasets are described. This section concludes with the description of the two new datasets - Economics and DeGruyter.

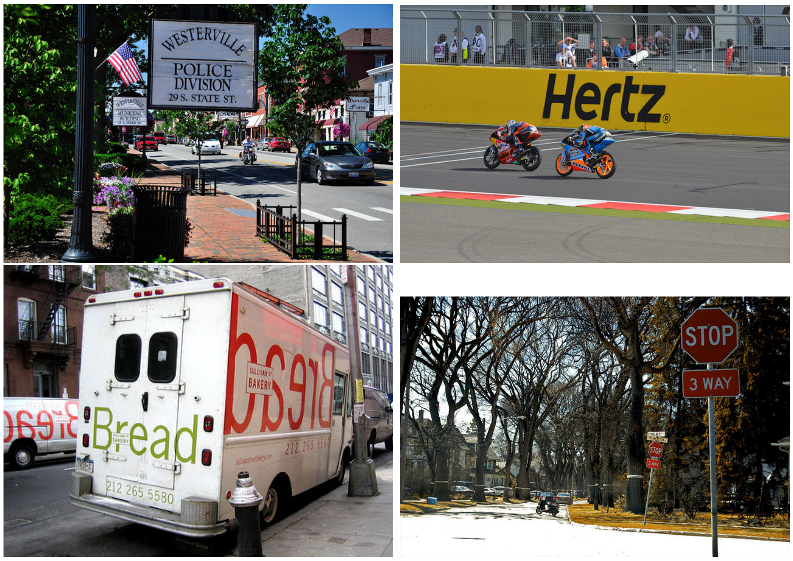


Figure 4.4. Four example figures from the MS-COCO-Text dataset.

Process of Dataset Creation

A self-created web-based tool was used to manually annotate the text elements in the scientific figures. Figure 4.6 shows the annotation tool. Every scientific figure is shown to the annotator at its original size. The annotator is then asked to annotate all text elements inside the figure. A text element is defined as sequence of alphanumerical characters (from a standard English keyboard set) which are on a line and not divided by more than approximately three blank-spaces on that line. First, the rectangular bounding box needs to be aligned with the text element, i. e., the baseline needs to be rotated such that it fits the text element. Second, the bounding box needs to be resized so that it is the minimal enclosing bounding box. Third, the text of the element needs to be typed into the system as it is, including for example any kind of typographical error that might appear in the figures. Finally, the orientation of the bounding box is calculated automatically. The tool stores for each text element the

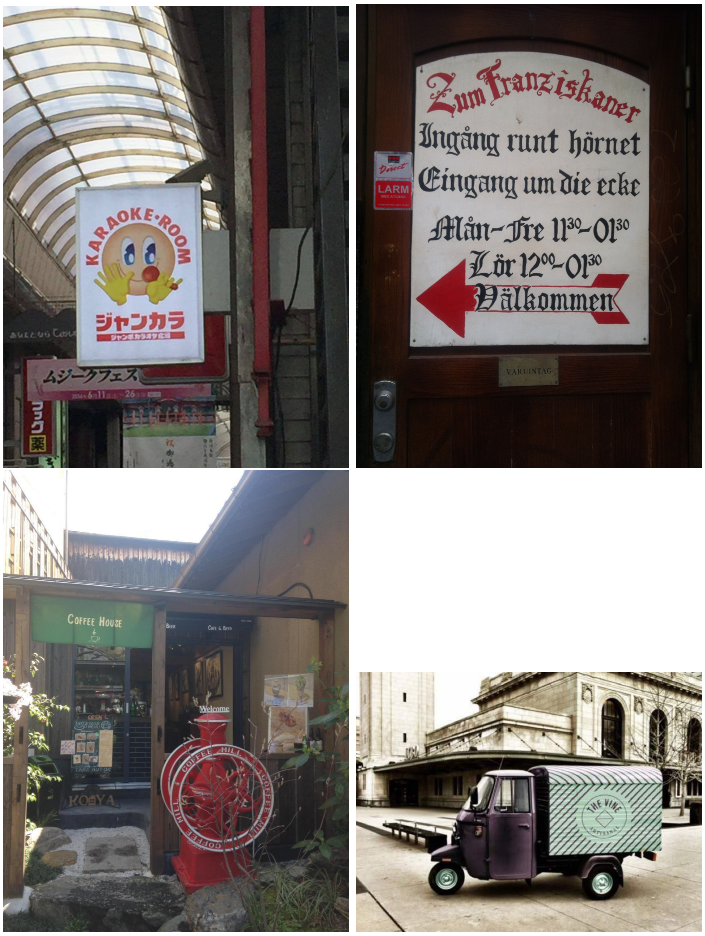


Figure 4.5. Four example figures from the Total-Text-Dataset.

information about its position, dimension, orientation and its content (text).

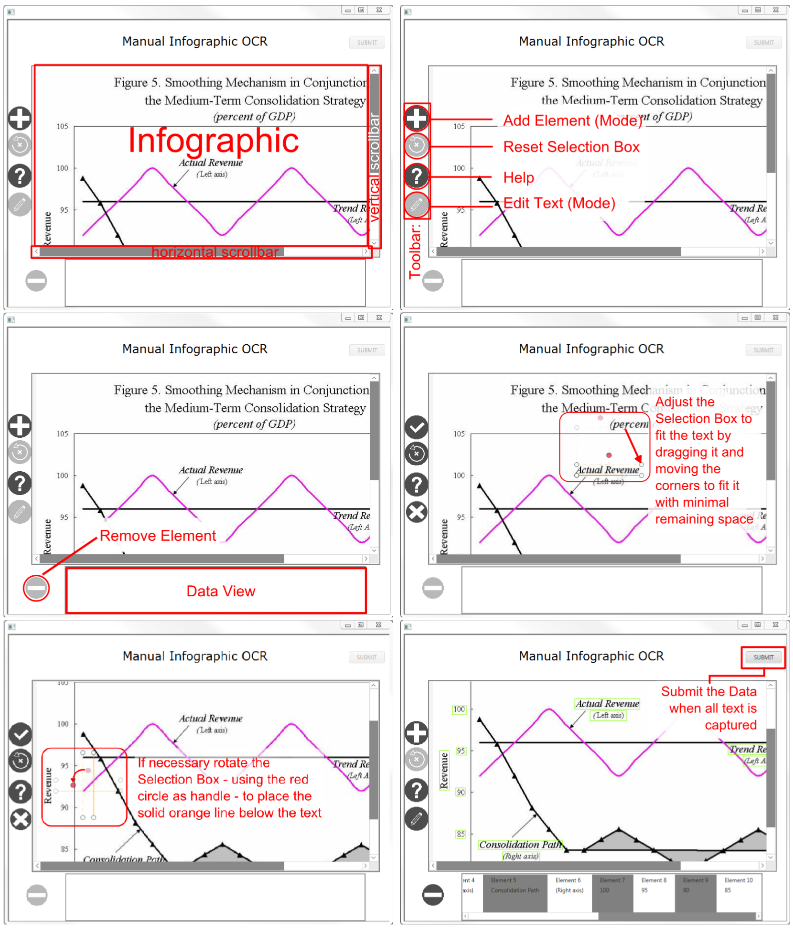

Manual Infographic OCR
SUBMIT
Infographic
vertical scrollbar
horizontal scrollbar
Toolbar:
Add Element (Mode)
Reset Selection Box
Help
Edit Text (Mode)
Remove Element
Data View
Adjust the Selection Box to fit the text by dragging it and moving the corners to fit it with minimal remaining space
If necessary rotate the Selection Box - using the red circle as handle - to place the solid orange line below the text
Submit the Data when all text is captured
Figure 5. Smoothing Mechanism in Conjunction with the Medium-Term Consolidation Strategy
(percent of GDP)
Actual Revenue
(Left axis)
Trend Re
Consolidation Path
(Right axis)
Revenue

Figure 4.6. These six images were used to explain the interface of the annotation tool to the users.

Economics

This dataset is created from a corpus of scientific open access documents that were collected from the EconBiz portal[2] for scientific economics

[2] https://www.econbiz.de, last access: December, 2018

documents. The collected corpus consists of about 147,000 open access documents (PDFs) which are presumably written in English according to an applied language detection. From these 147,000 documents 111 million uncompressed images were extracted using the pdfimages command of xpdf-utils on unix systems. From this set of 111 million images, 200,000 candidate images for scientific figures were extracted by applying aggressive thresholds: All images that have a width or height below 500 pixels were discarded as optical character recognition has problems on small(er) images. Additionally, images with a width or height above 2000 pixels were discarded as well as these are not individual graphics but rather scans of entire pages. From the candidate set, images were randomly picked and - one at a time - presented to a human assessor to confirm/reject it based on whether it is a scientific figure or not. A broad definition was used for scientific figures, i. e., any image that could be classified as some kind of numerical or categorical information visualization was considered to be a scientific figure. Scans of the front page (or any other page), photos of the authors, other natural images or purely stylistic elements were excluded as well as figure which included non-English text. The random selection resulted in a set of 121 scientific figures for which a gold standard was manually created using the tool described above. The selected subset of 121 figures resembles a wide variety of scientific figures from bar charts to maps. Figure 4.7 shows a few example images.

DeGruyter

The other dataset is composed of scientific figures from scholarly books that were separately made accessible by DeGruyter[3] under an open-access creative commons license. Figures from books in multiple languages were available from which ten books with figures in English were selected. Most of the selected books are from the chemistry domain. From this set of figures with English text, 120 figures were randomly selected and annotated with the tool described above, the same tool that has been used to annotate the Economics dataset. Figure 4.8 shows a few examples.

[3]`https://www.degruyter.com/`, last access: December, 2018

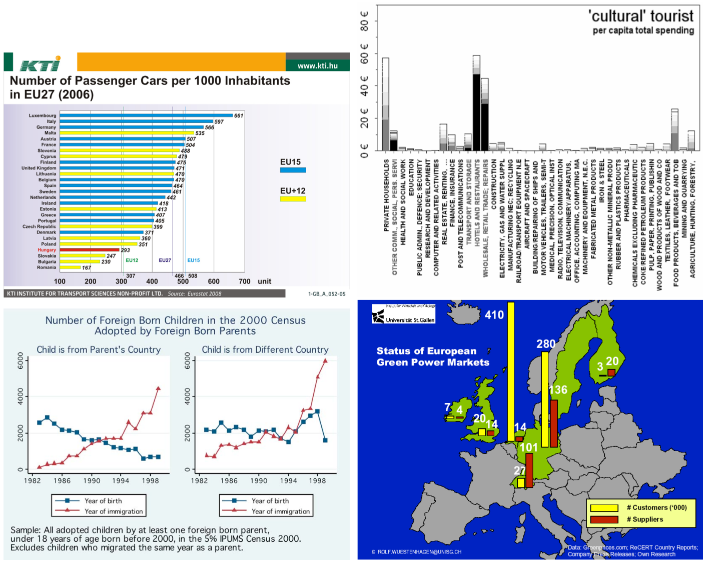


Figure 4.7. Four example figures from the Economics dataset.

4.1.3 Summary

Table 4.1 highlights some statistics about the main datasets containing scientific figures that were used in the experiments. This allows a comparison of the datasets and shows the differences between them. Both, the CHIME-R and CHIME-S datasets contain figures with an on average lower resolution than the Economics and DeGruyter datasets, which are almost equal. With respect to the average number of characters, words and text elements in a figure, the distribution of Economics and DeGruyter are similar, with about twice as many as the CHIME datasets which are similar to each other as well. In the experiments with the supervised approaches, also the DeTEXT dataset was used which has similar characteristics to the Economics and DeGruyter dataset.

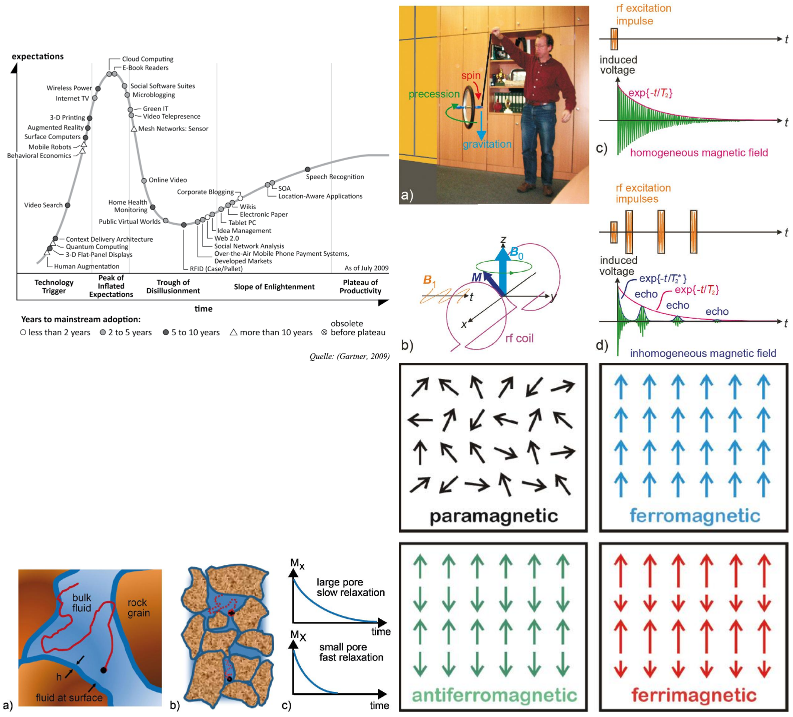


Figure 4.8. Four example figures from the DeGruyter dataset.

4.2 Evaluation Measurements

The text extraction pipeline in Section 3.1 consists of three main steps. The first step outputs a set of region, the second step outputs a set of text elements and the third and final step extends the text elements with the text recognized from the text elements. Evaluations on two different levels are conducted: An evaluation of the text element bounding boxes generation from the text detection step and an evaluation of the text recognition on word/text element level. Please note that an evaluation of the region detection or the text recognition on character level is not

Table 4.1. Number of figures, average figure width and height and average number of text elements (TE), words and characters (Chars) per figure.

Dataset	# Figures	Width	Height	# TE	# Words	# Chars
Economics	121	982	681	25	35	151
DeGruyter	120	959	619	24	34	149
CHIME-R	115	714	454	14	18	69
CHIME-S	85	440	320	12	18	76
Total	441	801	535	19	27	114

possible due to insufficient test datasets, i. e., there are no datasets with the information needed for such evaluations.

The gold standard that is used for evaluation is constructed as follows: The gold standard consists of text elements which represent single lines of text taken from a scientific figure. Each text line consists of one or multiple words which are separated by blank space. Each word may consist of any combination of characters and numbers. Every text line is defined by a specific position, size and orientation. The pipeline configurations output text elements with the same attributes. Section 4.2.1 describes the methods for evaluating the text detection and Section 4.2.2 describes the methods for evaluating the text recognition.

4.2.1 Text Detection

The evaluation of the text detection is conducted by matching elements from the gold standard and the pipeline output via their bounding boxes. For each bounding box in the gold standard, the Intersection-over-Union (IoU) - also known as Jaccard Index [Jac12] - on the pixel-level is computed with all bounding boxes that were extracted by the pipeline.

$$a_o = \frac{area(B_e \cap B_{gt})}{area(B_e \cup B_{gt})} \tag{4.2.1}$$

Two bounding boxes are matched if the IoU between the bounding boxes is larger than a pre-defined threshold. This reduces the error introduced

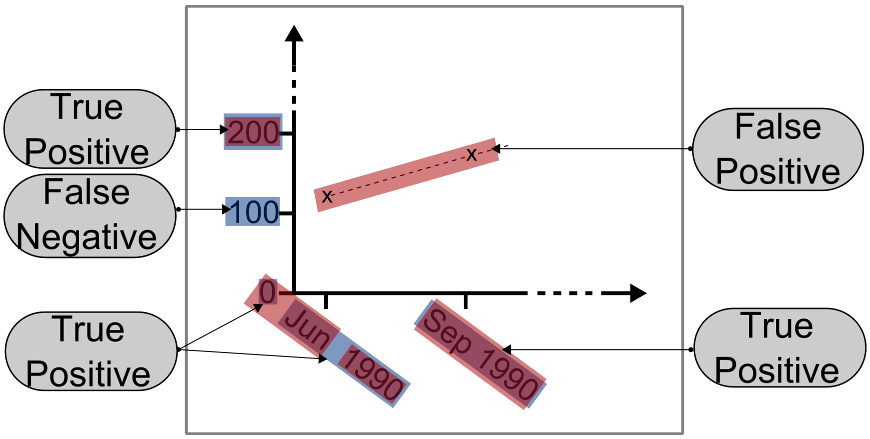


Figure 4.9. Definition of True Positives, False Positives and False Negatives after the text element matching process. Blue elements are from the gold standard. Red elements are generated by the pipeline.

through elements which are an incorrect match or only partially matches and only have a small overlap with the gold standard.

The first output to evaluate is the detection of the text element, i. e., whether all text element bounding boxes were determined correctly. The evaluation of the detection of text line elements is conducted on a macro (element) and micro (pixel) level. For this evaluation, an empirical test of different IoU thresholds ranging from 5% up to 30% showed that on average a threshold of 10% gives the best results regarding finding correct matches (True Positives) and avoiding incorrect matches (False Positives). It is noteworthy that multiple text elements from the pipeline may exceed the IoU threshold. Thus, a gold standard element can have multiple matching elements and a text element from the pipeline can be assigned to multiple elements from the gold standard if it fulfills the matching constraint for each match.

On the macro level, the evaluation focuses on whether all elements in the gold standard have a matching element in the output of the text extraction pipeline. Figure 4.9 illustrates this measure. If at least one match is found, given the constraints outlined above, for an element from the gold standard, it counts as a True Positives, regardless of size, orientation and content. If no match was found, it is considered as a False Negatives. A False Positives is an element from the pipeline output which has no match. Given the True Positives and False Positives/False Negatives, it is possible to compute Precision (P), Recall (R) and F1-Measure (see Appendix B) for assessing the text location detection on the macro level. Additionally, the ER which is the number of elements recognized by the pipeline divided by the number of elements in the gold standard and the MER which is the number of matched items from the pipeline divided by the number of elements of the gold standard are computed. These ratios give an idea about how greedy a configuration is at generating elements and whether gold standard elements get matched by multiple elements.

For the micro level evaluation, the True Positives elements are further analyzed by calculating the text element coverage. For each gold standard text element, the pixel-wise overlap is taken into account and Precision, Recall and the harmonic F1-Measure are calculated for its mappings. The True Positives in this case are the overlapping pixel and the False Positives are those pixel from the text elements from the pipeline which are outside the gold standard element. The False Negatives are the pixels of the gold standard element which were not covered by a text element from the pipeline. These values are averaged over all gold standard text elements in a figure.

In the optimization of the supervised approach, a slightly different metric is used, the Average Precision (AP), which has been used in different challenges when evaluating supervised approaches [EVW+10; YYY+17].

For each detected bounding box, the IoU is calculated using Equation 4.2.1, but with different thresholds than those used with the macro and micro level measures above. Commonly used thresholds are 50% and 75% (in comparison to the 10% used above). In addition, if multiple predicted bounding boxes match the same ground truth element then only the bounding box with the highest confidence score is considered a True Positives and all other matches for that ground truth element are marked as False Positives. This stands in contrast to the assumption made for the macro and micro measures described above which allowed multiple matches. Given the True Positives, True Negatives, False Positives and

False Negatives, one can calculate Precision and Recall as mentioned above while considering the confidence scores. And from those one can compute the Average Precision using the following equation:

$$AP = \frac{1}{11} \sum_{r \in \{0,0.1,\ldots,1\}} p_{interp}(r) \tag{4.2.2}$$

With the precision interpolated at each Recall level r by taking the maximum Precision:

$$p_{interp}(r) = max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}) \tag{4.2.3}$$

where $p(\tilde{r})$ is the measured Precision at Recall $\tilde{r}$.

4.2.2 Text Recognition

The text recognition, which is conducted in the third and final step of the pipeline, is evaluated on a macro(figure)- and micro(element)-level as well. However, an evaluation on character/pixel-level is not possible since that information is missing in the gold standard.

The Levenshtein Distance (LD) [Lev66] (see also Appendix B) is used to evaluate the text recognition on both the macro(figure)-level and the micro(element)-level. On the macro-level, for each figure, all characters from the text elements of the gold standard are combined into one string. Likewise, a string for the text elements extracted by the pipeline is created. The characters in both strings are sorted alphabetically and the Levenshtein Distance is computed for both strings. This approximates the overall number of operations needed to match these strings without considering the position of the individual characters. This measure is denoted as macro Levenshtein Distance (macro-LD/ωLD) in the remainder of this thesis. Since this macro Levenshtein Distance depends on the number of characters inside a figure, a second macro-level measure, the OPC score, is computed by dividing the Levenshtein Distance by the number of characters in the gold standard. This normalizes the macro Levenshtein Distance and makes it comparable across scientific figures with different amounts of characters. On the micro level, the Levenshtein Distance can be applied as well. The Levenshtein Distance is calculated for each gold standard element and the average over all gold standard elements is reported for the whole figure. Since multiple text elements from the pipeline can be matched to a gold standard text line, it is necessary to combine their textual content into one string. These extracted elements are combined using the position information of the bounding boxes. All elements are rotated according to their orientation and then are combined by starting at the top left element and ending with the bottom right element. The textual content of the elements is concatenated in that order and the resulting string is compared with the gold standard text using the Levenshtein Distance. If no match is found for a gold standard element or an element from the text extraction pipeline than the length of associated text is added to the calculation of the average Levenshtein Distance. This measure is called micro Levenshtein Distance (micro-LD/μLD) in the remainder of this thesis.

One drawback of the Levenshtein Distance is that is does not take the order of the characters into account. Another option for evaluating the text recognition is a n-grams based evaluation which can also be done on the micro(word) or macro(figure) level and takes the order of characters into account up to a certain degree. For the latter, all n-grams in a figure are collected into one set instead of comparing the n-gram sets between matching elements. On the n-grams, the standard evaluation metrics Precision (P), Recall (R) and F1-Measure (F1) can be defined as follows:

$$P = \frac{|Extr \cap Rel|}{|Extr|},\ R = \frac{|Extr \cap Rel|}{|Rel|},\ F = \frac{2 \cdot P \cdot R}{P + R}$$

Extr refers to the n-grams as they are computed from text elements that are extracted from a figure. *Rel* refers to the relevant n-grams from the gold standard. Both *Extr* and *Rel* are multisets, so its necessary to adjust the definitions of P and R. Multisets can appear insofar as the same n-gram can appear multiple times in both the extractions result from the pipeline as well as the gold standard. To properly account for the number of occurrences of an n-gram in *Extr* or *Rel*, a counter function $\mathbf{C}_A(x) := |\{x|x \in A\}|$ (as an extension of a set indicator function) over a multiset A is defined. For an intersection of multisets A and B, the counter function is defined as follows:

$$\mathbf{C}_{A \cap B}(x) := \min\{\mathbf{C}_A(x), \mathbf{C}_B(x)\} \tag{4.2.4}$$

Based on $\mathbf{C}_{A\cap B}(x)$, the following definitions of Precision and Recall for multisets can be used:

$$P = \frac{\sum_{x\in Extr\cup Rel} \mathbf{C}_{Extr\cap Rel}(x)}{\sum_{x\in Extr} \mathbf{C}_{Extr}(x)} \tag{4.2.5}$$

$$R = \frac{\sum_{x\in Extr\cup Rel} \mathbf{C}_{Extr\cap Rel}(x)}{\sum_{x\in Rel} \mathbf{C}_{Rel}(x)} \tag{4.2.6}$$

In addition, it can happen that one of the sets *Extr* and *Rel* is empty. This refers to the situation when no text element got extracted where it should have been, i. e., $Extr = \emptyset$ and $Rel \neq \emptyset$. In this case, the definition of $P := 0$ and $R := 0$ (false negative) is used following Groot et al. [GHT00]. In the other case where some text element got extracted but there is no match in the gold standard, i. e., $Extr \neq \emptyset$ and $Rel = \emptyset$, the definition $P := 0$ and $R := 1$ (false positive) is used. Since scientific figures often contain sparse and short text as well as short numbers, only n-grams of length 1, 2 and 3 are evaluated.

Chapter 5

Experiment I: Baseline Comparison with Optical Character Recognition Engines

In this experiment, comparisons between popular Optical Character Recognition engines and the proposed unsupervised text extraction pipeline are made to answer the first research question **RQ1 - "How good are two common, open-source document Optical Character Recognition (OCR) engines at extracting text from scientific figures?"**. In its core, text extraction from scientific figures is an OCR problem. Thus, it is interesting to see how well OCR engines can handle scientific figures even though they are designed for text extraction from document images which have different characteristics. While in document images text is usually well separated from graphical elements, they are often mixed up in scientific figures. With this experiment, the goal is to get a better understanding of the capabilities of OCR engines to identify open challenges with regard to scientific figures.

5.1 Procedure

Here, two OCR engines are analyzed and compared with the unsupervised text extraction pipeline of this thesis. First, a comparison with the open-source Optical Character Recognition engine Tesseract is made. Second, a comparison with the commercial ABBYY FineReader is conducted. Both experiment procedures are described below.

Open-Source OCR Engine: Tesseract The Tesseract OCR engine [Smi07] is a well known and widely used [SZ13; BS15b; MV11; SKC+11] open-source solution for text recognition. Tesseract was originally designed for text extraction from scanned document images, but, due to its accessibility as an open-source tool, it has been used in other contexts as well. To assess the capabilities of Tesseract for text extraction from scientific figures, a comparison with the unsupervised pipeline has been conducted [BS15b]. Tesseract is used in its default mode, i. e., including layout analysis over the entire figure. Tesseract supports a rotation margin of ±15° [Smi95]. In addition, it can detect text rotated at ±90°. It is noteworthy that Tesseract was not specifically trained for recognizing text in scientific figures and instead the provided default English language model was used.

In total, three extraction approaches are compared to each other, two based on Tesseract and one unsupervised pipeline configuration. The first Tesseract-based approach *T1* just applies Tesseract as it is. The second Tesseract-based approach *T2* executes multiple runs of Tesseract over the figure at different rotation angles of 0°, ±45° and ±90°. The results from the different orientations are merged and in case of overlaps the element with the greatest width is taken. The pipeline configuration *P1* has been developed as a contribution of this thesis and has been published prior [BS15b]. It starts with its adaptive Otsu binarization and Connected Component Labeling for region extraction, followed by heuristic filters and DBSCAN for clustering into text elements. These are further subdivided into individual lines with angle-based Minimum Spanning Tree clustering and the orientation for each text line is estimated with the Hough-based method. The text of each line is recognized with Tesseract.

All three approaches are evaluated on the Economics dataset against the gold standard using the n-gram based text recognition evaluation described in Section 4.2.2.

Commercial OCR Engine: ABBYY FineReader The ABBYY FineReader Optical Character Recognition engine is a commercial OCR engine which has been used by multiple researchers [JRW+07; CK15]. In this experiment, the ABBYY FineReader *A1* is compared with two unsupervised pipeline configurations *P2-O* and *P2-T*. Both unsupervised pipeline configurations apply adaptive Otsu binarization, Connected Component Labeling, Heuristic filters, DBSCAN, angle-based MST clustering and orientation estimation with the Single String Orientation Detection algorithm. They differ with respect to the OCR engine that is used, i. e., Tesseract or Ocropy.

Modified versions of the Economics and DeGruyter datasets are used to assess the capabilities of the ABBYY FineReader in comparison with the pipeline. Since the extraction of arbitrarily rotated text is the most challenging aspect for common OCR engines and since ABBYY FineReader has a smaller rotation tolerance than Tesseract, both datasets are duplicated and modified. One subset of each datasets contains only the horizontal text and the other version contains only rotated text. The information from the gold standard is used to either remove horizontal text which is defined as text with an orientation angle between -2 and 2 degrees or to remove rotated text which is all the text lines which have an orientation outside the range of -2 to 2 degrees. The F1-Measure on macro- and micro-level for text detection as well as the macro- and micro-level Levenshtein Distance are used to compare the approaches.

5.2 Results

In this section, the results for the comparison with the optical character recognition engines Tesseract and ABBYY FineReader are presented.

Open-Source OCR Engine: Tesseract This section presents the results of the evaluation of the Tesseract OCR engine in comparison with the unsupervised text extraction pipeline. First, descriptive statistics of the gold standard and the extraction results of the Economics datasets are presented. Subsequently, the evaluation results in terms of Precision, Recall and F1-Measure for figure and word-level evaluation of the Tesseract approaches and the *P1* pipeline configuration.

Table 5.1 presents the average numbers and standard deviation (in brackets) with regard to n-grams, words and word length for both Tesseract based approaches (*T1*/*T2*), the unsupervised text extraction pipeline configuration (*P1*) and the gold standard (*GS*). Table 5.1 shows that the *P1* detects at least 1.5 as many n-grams and words as *T1* and about 20% some more than *T2*. Compared with the gold standard, *P1* extracts about 10%

more n-grams and words. In addition *T1*, *T2* and *P1* extract words shorter than the gold standard. Overall, high standard deviations can be observed in the extraction results as well as the gold standard.

Table 5.1. Average number of n-grams, average number of words and average word length for GS, P1, T1 and T2 on the Economics dataset

	unigrams	bigrams	trigrams	words	length
GS	150.65 (122.28)	115.93 (103.09)	84.95 (85.61)	35.46 (22.24)	4.22 (1.48)
P1	177.21 (128.21)	127.34 (100.51)	89.34 (79.35)	50.07 (31.95)	3.63 (2.69)
T1	106.30 (87.71)	80.17 (69.12)	60.79 (54.54)	25.21 (22.12)	4.15 (2.25)
T2	135.08 (125.56)	100.20 (98.20)	75.08 (78.10)	35.25 (33.94)	4.08 (1.95)

The mean Precision, Recall and F1-Measure for the micro-level n-grams evaluation in Table 5.2 (standard deviation in brackets) show a relative improvement of about 30% on average for *P1* compared with *T1*. While the F1-Measure of *P1* for unigram, bigrams and trigrams is at minimum 0.47, *T1* achieves only a maximum of 0.42. The improvement was verified using significance tests, i. e., it was checked whether the two distributions obtained from *T1*/*T2* and *P1* significantly differ. Table 5.2 reports the results for the second approach based on Tesseract (*T2*). Comparing the results for *P1* and *T2* shows a similar difference, but for Recall over unigrams and F1-Measure over trigrams the improvement is smaller. Here, all differences are significant except for the Recall and F1-Measure over trigrams. Finally, one can observe high standard deviations for all measures. For details on the significance tests please refer to Böschen and Scherp [BS15b].

Table 5.3 contains the results for the evaluation on the macro-level (figure). While having on average higher values for all measures in both comparisons, the relative improvement for Precision, Recall and F1-Measure compared with the micro-level evaluation decreases. The significance of the results is only given for Recall and F1-Measure, but not for Precision.

Table 5.2. Mean micro Precision, micro Recall, micro F1-Measure for T1, T2 and P1

		μPrecision	μRecall	μF1-Measure
P1	unigram	.50 (0.41)	.68 (0.36)	.47 (0.39)
	bigram	.58 (0.39)	.54 (0.38)	.54 (0.34)
	trigram	.52 (0.39)	.48 (0.37)	.49 (0.37)
T1	unigram	.37 (0.36)	.48 (0.36)	.36 (0.35)
	bigram	.42 (0.33)	.42 (0.34)	.42 (0.33)
	trigram	.42 (0.31)	.42 (0.31)	.36 (0.33)
T2	unigram	.37 (0.37)	.51 (0.38)	.36 (0.36)
	bigram	.42 (0.34)	.42 (0.35)	.42 (0.34)
	trigram	.42 (0.32)	.42 (0.32)	.42 (0.32)

Table 5.3. Mean macro Precision, macro Recall, macro F1-Measure for T1/T2/P1

		ωPrecision	ωRecall	ωF1-Measure
P1	unigram	.67 (0.23)	.79 (0.20)	.71 (0.21)
	bigram	.60 (0.27)	.67 (0.25)	.62 (0.25)
	trigram	.57 (0.29)	.60 (0.29)	.57 (0.28)
T1	unigram	.67 (0.29)	.54 (0.31)	.58 (0.30)
	bigram	.60 (0.33)	.50 (0.33)	.53 (0.32)
	trigram	.55 (0.35)	.48 (0.34)	.49 (0.34)
T2	unigram	.65 (0.25)	.59 (0.29)	.60 (0.26)
	bigram	.57 (0.31)	.52 (0.31)	.53 (0.30)
	trigram	.51 (0.33)	.50 (0.34)	.49 (0.32)

Commercial OCR Engine: ABBYY FineReader Two separated evaluations were conducted as described in Section 5.1. Table 5.4 shows the detailed results of the experiments on horizontal text as well as non-horizontal text for both the ABBYY FineReader configuration *A1* and the unsupervised pipeline configurations *P2-T* and *P2-O*. The results show that the ABBYY FineReader OCR engine can extract horizontal text very well. The configuration *A1* achieves the best ωF1 score of 0.75 and the second best μF1 score of 0.63 which is only a little behind the best score of 0.64 for the configuration *P2-T* for text detection. With respect to the text recognition, configuration *A1* outperforms the unsupervised pipeline configurations on horizontal text elements by a large margin with a μLD score of 2.02, a ωLD score of 44.86 and an OPC score of 0.28. However, graphic elements and rotated text are problematic and are falsely recognized. Thus, for the rotated text, the modified linear pipeline configuration *P2* achieves better results. For text detection on rotated text, it outperforms the ABBYY OCR engine by roughly 50%. For text recognition, however, the results are quite similar.

Table 5.4. Macro F1-Measure (ωF1), micro F1-Measure (μF1), Average micro Levenshtein Distance (μLD), Average macro Levenshtein Distance (ωLD) and Operations Per Character (OPC) for standalone ABBYY FineReader *A1* and the configuration *P2-T/O* using Tesseract as well as Ocropy as OCR engine.

Config.	$\omega F1(SD)$	$\mu F1(SD)$	$\mu LD(SD)$	ωLD	OPC
only horizontal text					
A1	0.75 (0.20)	0.63 (0.16)	2.02 (2.61)	44.86	0.28
P2-T	0.70 (0.20)	0.64 (0.12)	4.03 (3.21)	70.26	0.51
P2-O	0.68 (0.21)	0.55 (0.18)	3.28 (3.18)	69.57	0.49
only rotated text					
A1	0.27 (0.21)	0.33 (0.15)	12.59 (9.43)	84.97	1.58
P2-T	0.49 (0.29)	0.64 (0.21)	17.46 (14.74)	82.01	1.76
P2-O	0.47 (0.28)	0.40 (0.22)	11.75 (9.84)	72.51	1.03

Chapter 6

Experiment II: Comparison of Unsupervised Approaches from the Related Work

This experiment conducts a thorough evaluation of unsupervised pipeline configurations motivated by the approaches that were identified from the related work to address the second research question **RQ2 - "Which unsupervised approach is the state-of-the-art, measured by the accuracy of the text localization and recognition quality, in extracting text from scientific figures?"**. First, the unsupervised pipeline configurations from the related work are described. Second, the experiment procedure is explained. Third, the results of the experiment are presented.

6.1 Pipeline Configurations

Seven pipeline configurations are derived from the related work. As some of the approaches rely on substeps with supervised methods or manual interaction, certain modifications had to be made to keep the configurations unsupervised and automated. The configurations are described in detail below. Each linear configuration inspired by the literature has an alphanumerical identifier (e. g., *L1*). For a better distinguishability, the identifier is extend where needed with the name of the first author whose paper was the main inspiration for that specific configuration (e. g., *L1[Huang]*).

The first configuration *L1[Huang]* is based on the work of Huang et al. [HTL03; HTL05]. First, regions are extracted with Connected Component Labeling. As no binarization was specified, binarization using Otsu's method was chosen to prepare the input for the Connected Component Labeling algorithm. Heuristic filters are applied to separate graphic and text element and then the text regions are grouped using the Gravity method. Different orientations are ignored and all text is assumed to be horizontal as the original did not have such a component. Finally, the grouped regions are processed with Tesseract, since no specific OCR engine was mentioned in their papers, and no postprocessing is used.

Based on the work of Jayant et al. [JRW+07], the second configuration *L2[Jayant]* was created. Jayant et al. trained a supervised system to recognize text. Thus, as a replacement, binarization with Otsu's method and Connected Component Labeling is selected. Subsequently, the regions are clustered with a Minimum Spanning Tree and the orientation is approximated by minimizing the Perpendicular Squared Distance. The original work mentions commercial OCR engines like Omnipage[1] for text recognition, but since the focus of this research is not primarily on comparing existing OCR engines, Tesseract was used to achieve a better comparability. Their proposed manual postprocessing and correction is omitted to keep the pipeline automated.

The third configuration *L3[Xu]* is motivated by the work of Xu and Krauthammer on text detection in biomedical images [XK10]. They did not specify which binarization algorithm they use, therefore Otsu binarization is selected for this configuration. Next, the edge image is computed from the binary image and the pivoting region extraction is applied to extract all text regions. Their proposed approach ends here. However, to complete this configuration and make it comparable to the other configurations, the following methods are used: The regions are filtered using heuristics and grouped into lines using DBSCAN and angle-based MST clustering. The orientation of each line is estimated via the Hough-based method and the text of each line candidate is recognize with the Tesseract OCR engine without postprocessing.

The fourth configuration *L4[Sas]* is inspired by the work of Sas and Zolnierek [SZ13]. The configuration starts with a binarization using Otsu's method. Next, regions are extracted with Connected Component Labeling. Subsequently, heuristic filtering similar to the original approach is

[1] `https://www.nuance.com/print-capture-and-pdf-solutions/optical-character-recognition/omnipage.html` last visited: December 2018

applied to filter graphical elements. Sas and Zolnierek's approach relies on a supervised decision tree which is replaced by the unsupervised line generation with a Minimum Spanning Tree. This configuration does not use any method for orientation estimation since the original work by Sas and Zolnierek does not have such a feature. Tesseract is used for text recognition since it was also used in the original paper. In the postprocessing step, all strings that contain too many special characters are removed which they similarly used in their work.

The fifth configuration *L5[Chiang]* is a modified version of the supervised approach for text extraction that was proposed by Chiang and Knoblock [CK15]. Different from the previous configurations, this configuration uses color quantization to generate multiple binary images. The suggested approach relies on manual training data for identifying text color and text region extraction, which is replaced by Connected Component Labeling on each binary image in this configuration. Subsequently, heuristic filtering and morphological clustering is applied on the regions to identify text line candidates. The orientation of each text line candidate is estimated with the Single String Orientation Detection method. Chiang and Knoblock used a commercial OCR engine which, in this configuration, is replaced by the Tesseract OCR engine for comparability. Finally, quantitative postprocessing is applied to the recognized text elements.

The sixth configuration *L6[Fraz]* is vaguely inspired by the approach of Fraz et al. [FSE15] for scene text detection and recognition. It starts with color quantization followed by Connected Component Labeling. The original approach uses a supervised Support Vector Machine to form words, which had to be replaced with unsupervised methods from the methods set. The extracted regions are filtered and DBSCAN is applied, followed by a MST clustering into text lines. The orientation of the text lines is calculated using the Hough method and the text is recognized using Tesseract. No postprocessing is applied.

The seventh configuration *L7[Böschen]* has been developed as a contribution of this thesis and has been published prior [BS15b]. The configuration starts with its adaptive Otsu binarization and Connected Component Labeling for region extraction. The extracted regions are filtered with heuristics and the remaining regions are clustered with DBSCAN into text elements. The text elements are further subdivided into text line candidates by the angle-based MST clustering. The orientation for each text line candidate is estimated with the Hough-based method. Finally, Tesseract is used to recognize the text. No postprocessing is applied.

6.2 Procedure

Four datasets (Economics, DeGruyter, CHIME-R, CHIME-S) are used when comparing the different approaches. Each of the seven configurations is used to process every figure from all four datasets. The configurations are compared with respect to their text detection quality with the macro- and micro-level Precision, Recall and F1-Measure as well as the Element Ratio (ER) and Matched Element Ratio (MER). Their text recognition quality is evaluated with the macro- and micro-level Levenshtein Distance and the Operations Per Character (OPC) score. For reasons of simplicity, only the average values over all datasets for all configurations as well as for each dataset separately are reported and no individual results per figure. First, the average micro Precision, micro Recall and micro F1-Measure are computed over the elements of each figure. Second, the macro and micro level results for Precision, Recall and F1-Measure in terms of mean and standard deviation are computed over all individual results per figure. The micro Levenshtein Distance is reported as the average of the mean values per figure and the average standard deviation. The macro Levenshtein Distance is defined by the mean and standard deviation over all figures and the normalized OPC score.

Table 6.1. Macro Precision (ωP), macro Recall (ωR), macro F1-Measure ($\omega F1$), Element Ratio (ER) and Matched Element Ratio (MER). Results are averaged over all datasets for the configurations.

Config.	ωP	ωR	$\omega F1$ (SD)	ER	MER
L1	0.61	0.43	0.48 (0.28)	0.77	0.57
L2	0.59	0.45	0.49 (0.28)	0.83	0.51
L3	**0.73**	0.35	0.45 (0.26)	0.43	0.39
L4	0.63	0.47	0.54 (0.23)	0.80	0.59
L5	0.52	0.50	0.53 (0.23)	1.37	0.60
L6	0.55	0.51	0.54 (0.25)	1.44	**0.72**
L7	0.66	**0.55**	**0.58 (0.25)**	**1.04**	0.69

6.3 Results

This section presents the results for the configurations of the comparison of unsupervised approaches from the related work with respect to text localization and text recognition. The macro text detection results for the configurations computed over all datasets are reported in Table 6.1. The best macro F1-Measure is achieved by configuration *L7[Böschen]* with a value of 0.58. The worst macro F1-Measure of 0.45 is achieved by configuration *L3[Xu]*. Looking at each dataset separately as documented in Table 6.2, one can see that configuration *L7[Böschen]* works best for the DeGruyter (0.70) and CHIME-R (0.63) datasets while *L4[Sas]* has the best result for the CHIME-S (0.55) and Economics (0.57) datasets. The configuration *L3[Xu]* has the worst macro F1-Measure for all four datasets. The micro text detection results in Table 6.3 shows the best micro Precision of 0.79 for *L1[Huang]*, the best micro Recall of 0.59 for *L4[Sas]* and the best micro F1-Measure of 0.57 for *L1[Huang]*. The configuration *L3[Xu]* has again the worst average micro F1-Measure. The individual results per dataset for all configurations are shown in Table 6.4. The best result for the Economics dataset is by *L1[Huang]* with 0.48. For the DeGruyter dataset *L6[Fraz]* and *L7[Böschen]* have the same micro F1-Measure of 0.62 but the latter has a slightly lower standard deviation. On CHIME-R, *L1[Huang]* achieves the best micro F1-Measure of 0.66 and on CHIME-S *L4[Sas]* comes out on top with a micro F1-Measure of 0.55.

Table 6.2. Average macro F1-Measure and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L1	0.45(0.29)	0.55(0.30)	0.50(0.25)	0.34(0.26)
L2	0.47(0.29)	0.50(0.27)	0.52(0.26)	0.33(0.22)
L3	0.38(0.24)	0.50(0.24)	0.48(0.26)	0.29(0.18)
L4	0.57(0.25)	0.51(0.24)	0.53(0.21)	0.55(0.23)
L5	0.53(0.22)	0.54(0.21)	0.56(0.24)	0.49(0.25)
L6	0.48(0.26)	0.62(0.21)	0.58(0.24)	0.41(0.21)
L7	0.55(0.25)	0.70(0.18)	0.63(0.23)	0.43(0.25)

Table 6.3. Micro Precision (μP), micro Recall (μR) and micro F1-Measure ($\mu F1$). Results are averaged over all datasets for the configurations.

Config.	μP	μR	$\mu F1$ (SD)
L1	**0.79**	0.54	**0.57 (0.20)**
L2	0.41	0.32	0.32 (0.21)
L3	0.33	0.34	0.30 (0.22)
L4	0.52	**0.59**	0.47 (0.21)
L5	0.53	0.41	0.42 (0.21)
L6	0.65	0.54	0.54 (0.23)
L7	0.60	0.49	0.50 (0.24)

Table 6.4. Average micro F1-Measure and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L1	0.48(0.17)	0.58(0.22)	0.66(0.18)	0.47(0.22)
L2	0.28(0.19)	0.33(0.20)	0.41(0.22)	0.21(0.19)
L3	0.23(0.17)	0.34(0.21)	0.37(0.26)	0.16(0.11)
L4	0.42(0.18)	0.44(0.23)	0.49(0.21)	0.55(0.20)
L5	0.37(0.19)	0.48(0.20)	0.45(0.22)	0.35(0.23)
L6	0.42(0.21)	0.62(0.15)	0.62(0.25)	0.45(0.25)
L7	0.40(0.20)	0.62(0.14)	0.60(0.26)	0.31(0.21)

The text recognition results are presented in Table 6.5 and the individual results per dataset for all configurations are presented in Table 6.6. The best results are obtained with *L7[Böschen]* with 0.67 Operations Per Character, an average micro Levenshtein Distance of 6.23 and an average macro Levenshtein Distance of 108.81. Only configuration *L5[Chiang]* has a slightly better average micro Levenshtein Distance (6.07).

Table 6.5. Average micro Levenshtein Distance (μLD) and macro Levenshtein Distance (ωLD) and Operations Per Character (OPC) over all datasets for the configurations using Tesseract.

Config.	$\mu LD(SD)$	$\omega LD(SD)$	OPC
L1	6.65 (5.41)	126.35 (138.95)	0.71
L2	7.92 (5.56)	150.25 (140.59)	1.13
L3	7.06 (5.41)	125.45 (134.88)	0.74
L4	6.67 (4.82)	122.28 (141.03)	0.70
L5	**6.07 (5.08)**	120.12 (125.87)	0.71
L6	6.72 (6.02)	135.64 (201.31)	0.85
L7	6.23 (4.93)	**108.81 (108.53)**	**0.67**

Table 6.6. Average micro Levenshtein Distance and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L1	5.69(3.33)	5.69(3.74)	6.75(6.95)	7.94(6.13)
L2	7.15(3.62)	7.97(3.85)	7.94(7.25)	8.36(6.34)
L3	6.27(3.46)	6.46(3.56)	6.99(7.14)	7.88(6.09)
L4	6.00(3.24)	6.13(3.25)	7.10(6.25)	7.29(5.64)
L5	5.27(3.11)	4.96(2.78)	6.65(7.21)	6.74(5.30)
L6	5.74(3.34)	6.53(5.87)	6.81(7.76)	7.26(5.55)
L7	5.42(3.06)	4.88(2.58)	6.51(6.85)	7.21(5.34)

Chapter 7

Experiment III: Optimization of the Unsupervised Approach

This chapter presents a more in-depth analysis of the unsupervised methods that were extracted from the related work to answer the third research question **RQ3 - "Can a new, unsupervised approach combining existing and new methods improve on the text location and extraction quality?"** to optimize the unsupervised approach. Based on the results of the previous experiment, a systematic comparison of the different unsupervised methods for the different steps of the pipeline is conducted to find the overall best unsupervised configuration. First, the new, unsupervised pipeline configurations are described. Second, the experiment procedure is explained. Third, the results of the experiment are presented.

7.1 Pipeline Configurations

The seventh configuration *L7[Böschen]* that was motivated from the related work has shown the on average best results in the previous experiment (see Chapter 6). The configuration starts with adaptive Otsu binarization and Connected Component Labeling for region extraction. The extracted regions are filtered with heuristics and the remaining regions are clustered with DBSCAN into text elements. The text elements are further subdivided into text line candidates by the angle-based Minimum Spanning Tree clustering. The orientation for each text line candidate is estimated with the Hough-based method. Finally, Tesseract is used to recognize the text. However, some of the other configurations showed competitive results. Thus, in this experiment, the configuration *L7[Böschen]* is selectively modified to determine whether it can be further improved.

The systematic modifications are organized along the steps of the general pipeline in Figure 3.8. Each of the systematic configurations has an alphanumeric identifier (e. g., *M01*). The identifier is extended with an abbreviation of the main change with respect to the base configuration *L7*, if needed. In all configurations, both open source OCR engines (**T**esseract and **O**cropy) and the commercial **A**BBYY FineReader OCR engine are used to generate the results. The last character of the identifier of a configuration clarifies which OCR engine is used. Since the base configuration *L7[Böschen]* uses only Tesseract, it is also assessed with the other two OCR engines: Ocropy and ABBYY FineReader.

The following configurations are evaluated in this experiment. They differ from the base configuration *L7[Böschen]* with respect to only one module of the text extraction pipeline:

- ▻ Configuration *M01[NIBLACK]-T/O/A* uses Niblack instead of adaptive Otsu for binarization.
- ▻ Configuration *M02[OTSU]-T/O/A* uses standard Otsu for binarization.
- ▻ Configuration *M03[PIVOT]-T/O/A* exchanges the binarization algorithm and Connected Component Labeling with color quantization and the pivoting region extraction.
- ▻ Configuration *M04[NOFILTER]-T/O/A* differs from the base configuration by not applying the optional heuristic filtering method.
- ▻ Configuration *M05[GRAVITY]-T/O/A* uses the Gravity Grouping instead of DBSCAN and the angle-based MST clustering.
- ▻ Configuration *M06[MST]-T/O/A* applies only the (normal) Minimum Spanning Tree clustering to cluster regions and create text lines.
- ▻ Configuration *M07[MORPH]-T/O/A* uses the morphological text line generation.
- ▻ Configuration *M08[PSD]-T/O/A* uses the Perpendicular Squared Distance based method for orientation estimation.

- ▻ Configuration *M09[SSOD]-T/O/A* uses the Single String Orientation Detection method to estimate the orientation.
- ▻ Configuration *M10[CHAR]-T/O/A* uses the special character filtering for post-processing.
- ▻ Configuration *M11[STR]-T/O/A* uses the string filter for post-processing.
- ▻ Configuration *M12[QUANT]-T/O/A* uses the quantitative assessment method for post-processing.
- ▻ Configuration *L7-O* uses the Ocropy OCR engine.
- ▻ Configuration *L7-A* uses the ABBYY FineReader OCR engine.

One of the main challenges is the extraction of rotated text from scientific figures. Common OCR engines are only good at recognizing horizontally aligned text. Thus, in addition to the pipeline configurations described above, two so called two-pass configurations were created. They are called two-pass configurations because they first extract horizontal text with an OCR engine and in a second pass with the proposed text extraction pipeline extract the rotated text.

The ABBYY FineReader OCR engine was chosen for the first pass as it is a commercial OCR engine and provides the best results in a direct comparison with the open source engines Tesseract and Ocropy. After extracting the horizontal text, the figure is manipulated by filling the corresponding bounding box of the text area with the color white. In the second pass, the best linear configuration *M09[SSOD]-O* is applied on the manipulated figure to extract the remaining (rotated) text. However, a decision has to be made about the output of ABBYY FineReader in the first step, i. e., which recognized text elements are acceptable since it may also falsely recognize non-horizontal text and graphic elements. The ABBYY FineReader SDK[1] provides a confidence value per word which can be used to make a decision about the quality of the recognition. According to the provided documentation, it is calculated using different bonuses (e. g., if the word was recognized from a dictionary) and penalties (e. g. bad recognition quality). No supplementary dictionary or word model was

[1] `https://www.abbyy.com/en-us/ocr-sdk/`, last access: September, 2017

used for calculating these values as scientific figures contain a large variety of abbreviations and numbers. ABBYY FineReader offers two options for calculating the confidence and the accurate calculation was chosen over the fast calculation because the performance of the pipeline is not in the focus of this thesis, but the quality of the result. Since recognized lines of text can contain multiple words, it is necessary to aggregate the word-confidence values to decide whether to accept a text line or not. Two options for aggregation were tested as separate configurations: The first configuration takes the average confidence value of all words of a line. The second configuration takes the minimum confidence value of all words of a line. If the aggregated confidence value exceeds a predefined threshold, the recognized output gets accepted and is removed from the figure. Thus, the second configuration generally extracts fewer text lines in the first pass.

For standard document pages, the confidence values are in the range from 0 to about 100 as can be seen in Figure 7.1. In contrast, the confidence values of the text elements recognized in figures can also be negative. Investigations of the produced confidence values suggest to use a threshold of zero for both text line confidence values since one can observe that short words and numbers have low confidence values close to zero assigned to them even though they were correctly recognized. Non-horizontal text or certain graphic elements (e. g., dashed lines) have larger negative confidence values whereas correctly recognized larger text elements mostly result in high positive values.

Therefore, two configurations *N1[0AVG]* and *N2[0MIN]* were created to test the two pass approach with both options for evaluating the confidence score of a text line given a confidence threshold of zero.

7.2 Procedure

In order to evaluate the impact of the individual methods, all the configurations described in Section 7.1 are evaluated on the same four datasets (Economics, DeGruyter, CHIME-R, CHIME-S) that were used in the previous experiment. The same measures as in the previous experiment (see Chapter 6) are used as well to evaluate the text detection and text recog-

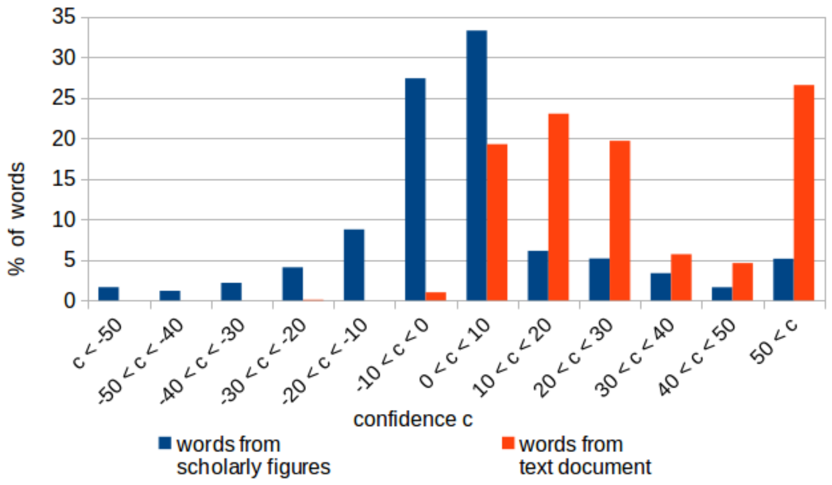


Figure 7.1. Comparison of the confidence values of words from scientific figures and words from a text document

nition, i. e., macro- and micro-level Precision, Recall and F1-Measure, the ER and MER, the macro- and micro-level Levenshtein Distance and the Operations Per Character (OPC) score (see Section 4.2 for a description of the different measures).

7.3 Results

For the systematic comparison of unsupervised methods, Table 7.1 shows the macro text detection results and Table 7.2 shows the micro text detection results. The best macro text detection F1-Measure of 0.67 is achieved by *M09[SSOD]*, which is also supported by the micro text detection results in Table 7.2 with the highest F1-Measure of 0.65. The separate evaluation of each dataset in Table 7.3 confirms as well that *M09[SSOD]* works best for Economics, CHIME-R and CHIME-S with macro F1-Measure of 0.68, 0.66 and 0.63. The best result for DeGruyter is 0.73 by *M07[MORPH]* with *M09[SSOD]* having the second best F1-Measure of 0.71. The best micro results for Economics (0.56), DeGruyter (0.67), CHIME-R (0.71) and CHIME-S (0.66) are all achieved by *M09[SSOD]*, as shown in Table 7.4. Table 7.5 shows the text recognition results using Tesseract, Table 7.6 shows the text recognition results using Ocropy and Table 7.7 shows the results using ABBYY FineReader OCR. Configuration *M09[SSOD]-O* produces the best text recognition results with an average micro Levenshtein Distance of 4.71 and an OPC of 0.53. In addition, configuration *M09[SSOD]-O* shows the best results of 95.49 for the average macro Levenshtein Distance. Comparing the different configurations shows that the only major improvement is achieved by *M09[SSOD]*. The results for ABBYY FineReader vary in a range between the results of Tesseract and Ocropy and do not show a better result than *M09[SSOD]* with Tesseract/Ocropy. Comparing the performance of the three Optical Character Recognition engines and different configurations on each dataset individually (see Table 7.8, Table 7.9 and Table 7.10), the best micro Levenshtein Distance is also achieved by configuration *M09[SSOD]-O* with values between 3.51 and 5.80.

Table 7.1. Macro Precision (ωP), macro Recall (ωR), macro F1-Measure ($\omega F1$), Element Ratio (ER) and Matched Element Ratio (MER). Results are averaged over all datasets.

Config.	ωP	ωR	$\omega F1$ (SD)	ER	MER
L7	0.66	0.55	0.58 (0.25)	1.04	0.69
M01	0.64	0.52	0.57 (0.25)	0.96	0.64
M02	0.67	0.40	0.49 (0.26)	0.74	0.53
M03	0.61	0.44	0.48 (0.25)	0.96	0.75
M04	0.60	0.46	0.51 (0.23)	0.86	0.52
M05	0.62	0.50	0.55 (0.27)	0.90	0.64
M06	0.61	0.54	0.59 (0.25)	1.19	0.74
M07	0.67	0.55	0.62 (0.23)	1.08	0.65
M08	0.62	0.53	0.57 (0.24)	**1.01**	0.66
M09	0.67	**0.63**	**0.67 (0.22)**	1.27	**0.88**
M10	**0.69**	0.54	0.59 (0.25)	0.97	0.70
M11	0.67	0.55	0.60 (0.25)	**1.01**	0.69
M12	0.66	0.38	0.48 (0.25)	0.60	0.43

Table 7.2. Micro Precision (μP), Recall (μR) and F1-Measure ($\mu F1$). Results are averaged over all datasets.

Config.	μP	μR	$\mu F1\ (SD)$
L7	0.60	0.49	0.50 (0.24)
M01	0.59	0.44	0.47 (0.24)
M02	0.46	0.40	0.38 (0.26)
M03	0.41	0.57	0.42 (0.23)
M04	0.59	0.54	0.50 (0.21)
M05	0.76	0.54	0.57 (0.20)
M06	0.57	0.47	0.47 (0.24)
M07	0.60	0.47	0.48 (0.22)
M08	0.49	0.40	0.41 (0.20)
M09	**0.77**	**0.63**	**0.65 (0.17)**
M10	0.59	0.49	0.49 (0.24)
M11	0.59	0.49	0.49 (0.24)
M12	0.39	0.29	0.31 (0.21)

Table 7.3. Average macro F1-Measure and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L7	0.55(0.25)	0.70(0.18)	0.63(0.23)	0.43(0.25)
M01	0.54(0.26)	0.65(0.20)	0.61(0.24)	0.42(0.26)
M02	0.46(0.28)	0.51(0.28)	0.52(0.24)	0.43(0.25)
M03	0.42(0.23)	0.53(0.23)	0.49(0.27)	0.36(0.22)
M04	0.46(0.22)	0.59(0.19)	0.53(0.22)	0.37(0.25)
M05	0.48(0.29)	0.64(0.22)	0.58(0.27)	0.33(0.28)
M06	0.55(0.25)	0.70(0.21)	0.62(0.22)	0.40(0.24)
M07	0.62(0.24)	**0.73(0.17)**	0.63(0.22)	0.47(0.25)
M08	0.57(0.24)	0.64(0.20)	0.59(0.24)	0.39(0.27)
M09	**0.68(0.21)**	0.71(0.18)	**0.66(0.23)**	**0.63(0.26)**
M10	0.55(0.25)	0.71(0.17)	0.62(0.23)	0.43(0.26)
M11	0.55(0.26)	0.71(0.17)	0.64(0.23)	0.43(0.25)
M12	0.38(0.24)	0.57(0.21)	0.50(0.21)	0.36(0.25)

Table 7.4. Average micro F1-Measure and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L7	0.40(0.20)	0.62(0.14)	0.60(0.26)	0.31(0.21)
M01	0.36(0.21)	0.58(0.16)	0.58(0.26)	0.27(0.17)
M02	0.32(0.24)	0.46(0.28)	0.42(0.25)	0.31(0.21)
M03	0.43(0.22)	0.45(0.23)	0.43(0.26)	0.33(0.22)
M04	0.40(0.17)	0.56(0.16)	0.60(0.22)	0.35(0.23)
M05	0.46(0.19)	0.58(0.16)	0.69(0.19)	0.45(0.23)
M06	0.36(0.20)	0.60(0.16)	0.59(0.25)	0.24(0.18)
M07	0.39(0.19)	0.62(0.14)	0.56(0.22)	0.30(0.17)
M08	0.37(0.17)	0.47(0.14)	0.48(0.21)	0.20(0.20)
M09	**0.56(0.14)**	**0.67(0.11)**	**0.71(0.21)**	**0.66(0.20)**
M10	0.39(0.20)	0.62(0.15)	0.60(0.26)	0.30(0.20)
M11	0.39(0.20)	0.61(0.15)	0.59(0.26)	0.31(0.21)
M12	0.21(0.16)	0.41(0.20)	0.36(0.22)	0.19(0.17)

Table 7.5. Average micro Levenshtein Distance (μLD) and macro Levenshtein Distance (ωLD) and Operations Per Character (OPC) over all datasets for the systematic configurations using Tesseract.

Config.	$\mu LD(SD)$	$\omega LD(SD)$	OPC
L7-T	6.23 (4.93)	108.81 (108.53)	0.67
M01-T	6.27 (4.95)	117.58 (124.23)	0.69
M02-T	6.55 (5.06)	131.58 (142.74)	0.75
M03-T	8.31 (6.14)	154.54 (168.10)	1.09
M04-T	6.55 (4.94)	111.30 (105.13)	0.75
M05-T	6.68 (5.65)	108.86 (102.93)	0.66
M06-T	6.30 (5.29)	115.43 (113.79)	0.69
M07-T	6.15 (5.12)	104.61 (105.97)	0.63
M08-T	8.30 (5.59)	147.91 (129.55)	1.04
M09-T	5.47 (4.39)	96.29 (99.44)	0.58
M10-T	5.96 (4.88)	105.50 (107.16)	0.61
M11-T	6.20 (4.90)	108.06 (109.38)	0.64
M12-T	6.07 (5.03)	120.78 (122.44)	0.67

Table 7.6. Average micro Levenshtein Distance (μLD) and macro Levenshtein Distance (ωLD) and Operations Per Character (OPC) over all datasets for the systematic configurations using Ocropy.

Config.	$\mu LD(SD)$	$\omega LD(SD)$	OPC
L7-O	5.47 (4.98)	108.55 (106.64)	0.64
M01-O	5.70 (5.09)	117.46 (128.73)	0.66
M02-O	6.16 (5.21)	131.39 (143.16)	0.73
M03-O	7.06 (5.62)	136.40 (132.05)	0.82
M04-O	6.29 (5.50)	120.71 (109.18)	0.76
M05-O	6.22 (5.75)	130.21 (127.87)	0.69
M06-O	5.85 (5.34)	110.74 (107.23)	0.67
M07-O	5.52 (5.10)	106.71 (104.05)	0.64
M08-O	7.23 (5.60)	135.21 (122.48)	0.85
M09-O	4.71 (4.66)	95.49 (94.80)	0.53
M10-O	5.46 (5.00)	109.07 (104.57)	0.63
M11-O	5.45 (4.96)	106.38 (103.29)	0.63
M12-O	5.79 (4.97)	126.92 (124.06)	0.71

Table 7.7. Average micro Levenshtein Distance (μLD) and macro Levenshtein Distance (ωLD) and Operations Per Character (OPC) over all datasets for the systematic configurations with ABBYY FineReader.

Config.	$\mu LD(SD)$	$\omega(SD)$	OPC
L7-A	6.08 (4.94)	119.47 (109.53)	0.76
M01-A	6.28 (5.12)	124.91 (125.83)	0.78
M02-A	6.54 (4.94)	137.37 (141.49)	0.85
M04-A	6.19 (4.91)	129.07 (125.91)	0.86
M05-A	6.75 (5.74)	105.06 (86.48)	0.64
M06-A	6.22 (5.25)	123.97 (111.51)	0.78
M07-A	6.06 (5.04)	111.50 (94.13)	0.75
M08-A	8.45 (5.94)	156.56 (132.74)	1.08
M09-A	5.08 (4.32)	107.18 (106.71)	0.63
M10-A	5.19 (4.69)	95.39 (94.47)	0.53
M11-A	6.05 (4.91)	119.67 (107.84)	0.73
M12-A	6.58 (5.06)	142.36 (134.57)	0.80

Table 7.8. Average micro Levenshtein Distance and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S using Tesseract.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L7-T	5.42(3.06)	4.88(2.58)	6.51(6.85)	7.21(5.34)
M01-T	5.47(3.12)	5.30(2.78)	6.42(6.85)	7.22(5.42)
M02-T	5.74(3.20)	5.76(3.05)	6.72(6.72)	7.51(5.71)
M03-T	7.13(4.13)	8.44(4.88)	8.50(7.88)	8.68(6.49)
M04-T	5.78(3.33)	5.83(3.11)	6.64(6.29)	7.63(5.72)
M05-T	5.77(3.43)	5.49(3.79)	6.63(7.46)	8.34(6.27)
M06-T	5.57(3.07)	4.95(2.75)	6.72(7.27)	7.51(6.08)
M07-T	5.17(3.16)	4.92(3.34)	6.66(6.68)	7.53(5.62)
M08-T	7.56(3.95)	8.82(3.84)	8.15(7.25)	8.49(6.47)
M09-T	**4.90(3.01)**	**4.55(2.55)**	**5.76(6.01)**	**6.27(4.97)**
M10-T	5.13(3.03)	4.77(2.33)	6.06(6.71)	7.29(5.43)
M11-T	5.39(3.02)	4.83(2.44)	6.53(6.87)	7.24(5.37)
M12-T	5.47(3.13)	4.96(2.63)	6.31(6.91)	7.43(5.70)

Table 7.9. Average micro Levenshtein Distance and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S using Ocropy.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L7-O	4.80(2.93)	4.07(2.26)	5.96(6.97)	7.20(5.66)
M01-O	4.81(2.86)	4.47(2.83)	5.94(7.00)	7.47(5.71)
M02-O	5.16(3.10)	5.21(3.23)	6.54(6.89)	7.35(5.89)
M03-O	6.18(3.33)	6.29(3.04)	7.19(7.72)	8.51(6.27)
M04-O	5.30(2.94)	5.02(2.58)	6.91(7.83)	7.78(6.22)
M05-O	5.28(3.27)	4.71(2.92)	6.31(7.83)	8.14(6.29)
M06-O	4.98(3.00)	4.45(2.42)	6.09(7.35)	7.73(6.10)
M07-O	4.56(2.89)	3.82(2.24)	6.13(7.01)	7.33(5.60)
M08-O	6.34(3.29)	6.71(3.44)	7.36(7.81)	8.22(6.44)
M09-O	**3.86(2.82)**	**3.51(2.00)**	**5.08(6.32)**	**5.80(5.50)**
M10-O	4.73(2.93)	4.28(2.23)	5.73(6.82)	7.19(5.63)
M11-O	4.75(2.89)	4.08(2.27)	5.79(6.87)	7.16(5.61)
M12-O	5.20(2.99)	4.64(2.77)	5.95(6.59)	7.57(6.05)

Table 7.10. Average micro Levenshtein Distance and Standard Deviation per configuration for the individual datasets Economics, DeGruyter, CHIME-R and CHIME-S using ABBYY FineReader.

Config.	Economics	DeGruyter	CHIME-R	CHIME-S
L7-A	5.44(3.08)	4.79(2.73)	6.19(6.71)	7.45(5.52)
M01-A	5.73(3.50)	5.09(3.03)	6.39(6.89)	7.28(5.57)
M02-A	5.89(3.48)	5.70(3.19)	6.86(6.46)	7.31(5.71)
M04-A	5.66(3.24)	5.31(3.08)	6.30(6.52)	7.36(5.69)
M05-A	5.99(3.48)	5.37(3.52)	6.78(7.76)	8.31(6.35)
M06-A	5.72(3.11)	4.77(2.44)	6.49(7.24)	7.93(6.22)
M07-A	5.10(2.96)	4.57(2.90)	6.36(6.53)	7.66(5.82)
M08-A	7.80(3.98)	8.47(3.93)	8.10(7.95)	9.34(6.73)
M09-A	4.76(3.03)	4.30(2.30)	5.38(5.82)	5.09(4.68)
M10-A	4.71(2.96)	4.00(2.34)	5.36(6.63)	6.57(5.19)
M11-A	5.43(3.07)	4.75(2.68)	6.12(6.64)	7.43(5.49)
M12-A	5.83(3.07)	5.77(2.69)	6.87(7.26)	7.85(5.96)

The results for both two-pass configurations *N1[0AVG]* and *N2[0MIN]* show an improvement over the best linear configuration *M09[SSOD]*. Regarding the macro F1-Measure both two-pass configurations improved the previous best results by 0.02. The text recognition quality clearly improved with respect to the average macro LD, the average micro LD and the Operations Per Character score. When comparing both two-pass configurations, configuration *N2[0MIN]* achieved the overall best results. Table 7.11 lists the detailed results of the two-pass configurations compared with the one-pass pipeline configuration *M09[SSOD]*.

Table 7.11. Macro F1-Measure ($\omega F1$), Micro F1-Measure ($\mu F1$), Average Micro Levenshtein Distance (μLD), Average Macro Levenshtein Distance (ωLD) and Operations Per Character (OPC) for the configurations *M09[SSOD], N1[0AVG]* and *N2[0MIN]*.

Config.	$\omega F1(SD)$	$\mu F1(SD)$	$\mu LD(SD)$	ωLD	OPC
M09[SSOD]-T	0.67 (0.22)	0.65 (0.17)	5.47 (4.39)	96.29	0.58
M09[SSOD]-O	0.64 (0.21)	0.55 (0.20)	4.71 (4.66)	95.49	0.53
M09[SSOD]-A	0.67 (0.21)	0.65 (0.17)	5.08 (4.32)	107.18	0.63
N1[0AVG]	0.69 (0.20)	0.65 (0.16)	3.89 (4.64)	83.09	0.36
N2[0MIN]	0.69 (0.20)	0.64 (0.16)	3.59 (4.26)	77.74	0.35

Chapter 8

Experiment IV: Optimization of the Supervised Approach

The content of this chapter presents the results of a comparisons of different variations of the supervised approach. The comparison was conducted by Morten Jessen in his master thesis who was supervised by the author of this thesis. Part of this work was published at DocEng'19 [JBS19]. In this experiment, to optimize the supervised approach, different sets of hyper-parameters for a neural network-based approach are evaluated as well as different datasets for pretraining the neural network and training data augmentation methods. The learning rate for the neural network was fixed to 0.0075 for all experiments and executed 250,000 training iterations per model. Section 8.1 presents the hyperparameter optimization experiments and results. Section 8.2 presents the pretraining experiments and results. Section 8.3 presents the training data augmentation experiments and results.

8.1 Hyperparameter Optimization

The first step of the supervised approach is a Residual Neural Network which can be configured in multiple ways. The following parameter sets for the Residual Neural Network were evaluated:

▷ Model (ResNet or ResNeXt)

▷ Number of input layers (50 or 101)

▷ Width of the input image (800, 750, 700, 600 pixel)

▻ Only ResNeXt: cardinality and bottleneck width (32/8 or 64/4)

Please note that the input image width was limited by the GPU memory restrictions. Table 8.1 shows the results for the differently parameterized configurations. The best result for each metric is highlighted.

Table 8.1. Comparison of the AP50 and AP75 measures for different Hyperparameters (architecture, layer (l), cardinality (c), bottleneck width (b) and image width (w)).

Architecture	l	c	b	w	AP50	AP75
ResNet	50	-	-	600px	89.81%	54.11%
ResNet	50	-	-	800px	**91.79%**	54.65%
ResNet	101	-	-	600px	90.19%	52.58%
ResNet	101	-	-	700px	91.58%	**58.51%**
ResNeXt	50	32	8	600px	88.11%	48.02%
ResNeXt	50	32	8	750px	89.44%	53.14%
ResNeXt	50	64	4	600px	87.86%	49.16%
ResNeXt	50	64	4	750px	89.38%	53.66%
ResNeXt	101	32	8	600px	88.60%	50.41%
ResNeXt	101	64	4	600px	88.84%	51.05%

The best result, when considering the AP75 measure, is achieved with ResNet with 101 layers and an input image width of 700px. This configuration is used for the following experiments.

8.2 Pretraining of the Supervised Model

Pretraining a neural network can improve the performance. However, not enough training data is available for text extraction from scientific figures. Thus, pretraining with two datasets for text extraction from natural images is evaluated. The model was separately pretrained with the MS-COCO-Text dataset and the Total-Text-Dataset (TTD). Table 8.2 shows the impact of the pretraining on the performance of the model.

The model with pretraining on MS-COCO-Text performs best by achieving an AP50 of 95.21% and an AP75 of 76.33%.

Table 8.2. Comparison of the AP50 and AP75 measures for ResNet101 with and without pretraining.

Pretraining	none	on TTD	on COCO-Text
AP50	91.35%	94.49%	95.21%
AP75	63.49%	75.51%	76.33%

8.3 Augmentation of Training Data

Besides pretraining the model, another method for improving the training of the supervised model was considered. Because real training data is scarce, methods for artificially extending the training dataset were evaluated.

The following methods were applied to each training image to artificially increase the size of the training dataset:

▻ Rotation (90°, 180°, 270°)

▻ Scaling

▻ Flipping/Mirroring

▻ Translation

▻ Adding Noise

Table 8.3 shows the result of the comparison with and without training data augmentation.

Table 8.3. Comparison of the AP50 and AP75 measures for ResNet101 with and without training data augmentation.

Augmentation	no	yes
AP50	90.34%	92.88%
AP75	53.02%	67.57%

The model with artificially augmented training data clearly outperforms the model with the regular training data.

Chapter 9

Experiment V: Best Unsupervised Approach vs Best Supervised Approach

In the four previously presented experiments, optimizations have been conducted to identify the best unsupervised approach as well as the best supervised approach, in preparation for answering the last research question **RQ4 - "How does the best unsupervised approach compare with a supervised approach which is trained on a small dataset?"**. In this experiment, the best unsupervised configuration of the text extraction pipeline is compared with the best supervised approach, a neural network based text detection. The latter was developed in the course of a master thesis by Morten Jessen who was supervised by the author of this thesis. The core results of Morten Jessen's master thesis were published at the ACM Document Engineering Symposium [JBS19]. First, the pipeline configurations are described. Second, the experiment procedure is explained. Third, the results of the experiment are presented.

9.1 Pipeline Configurations

The developed neural network approach *NeuralNet* consists of a Faster R-CNN like architecture that uses a Residual Neural Network with 101 layers for the detection of text elements in scientific figures. The approach addresses the lack of training data by pretraining the architecture with the MS-COCO Text dataset [LMB+14] and artificially extending the scientific figure training datasets by creating augmented versions of the figures in the dataset.

After locating the text, Tesseract is used to recognize the text. Text at different orientations is addressed by brute-force rotating the elements before recognizing it with Tesseract and selecting the output with the highest confidence score.

The best unsupervised configuration *N2[0MIN]-O* is chosen for the comparison. This configuration first extracts horizontal text with the ABBYY FineReader before applying the unsupervised text extraction pipeline consisting of adaptive Otsu binarization, Connected Component Labeling, Heuristic filters, DBSCAN, angle-based Minimum Spanning Tree clustering, orientation estimation with the Single String Orientation Detection algorithm and Ocropy to extract the rotated text.

9.2 Procedure

The supervised neural network approach *NeuralNet* is compared with the configuration *N2[0MIN]-O* of the unsupervised text extraction pipeline with respect to the macro-level Precision (ωP), Recall (ωR) and F1-Measure (ωF1) for text detection and the macro-level Levenshtein Distance (ωLD) and micro-level Levenshtein Distance (μLD) over four datasets (CHIME-R, CHIME-S, Economics, DeGruyter). However, since the neural network requires training data, the results for the neural network approach are the average of a 5-fold cross validation while the results for the unsupervised approach are calculated on the full datasets once.

Table 9.1. Comparison of the result of the 5-fold cross-validation of the supervised approach *NeuralNet* with the result of the unsupervised approach *N2[0MIN]-O* for the measures macro Precision, macro Recall and macro F1-Measure as well as macro Levenshtein Distance (ωLD) and micro Levenshtein Distance (μLD).

	NeuralNet	N2[0MIN]-O
ωPrecision	0.86	0.67
ωRecall	0.83	0.63
ωF1-Measure	0.87	0.69
ωLD	39.11	77.74
μLD	2.44	3.59

9.3 Results

Table 9.1 shows the results of the supervised approach *NeuralNet* in comparison with the unsupervised approach *N2[0MIN]-O*. The results show that the neural network approach surpasses the unsupervised pipeline by a large margin. While the best pipeline configuration *N2[0MIN]-O* only achieves a ωF1 score of 0.69, the neural network approach manages to achieves a ωF1 score of 0.87 which is an increase of about 30% for the text detection. Even though the text recognition of the neural network approach is based on a brute-force methodology, also the text recognition quality improves by reducing the macro Levenshtein Distance ωLD from 77.74 down to 39.1 and the micro Levenshtein Distance μLD from 3.59 down to 2.44.

Chapter 10

Discussion

This chapter concludes the evaluation of the different text extraction approaches by discussing the main results, identifying threats to the validity of the results and analyzing the limitations of the conducted research. First, the main results of the five experiments are discussed in Section 10.1. Next, possible limitations to the conducted research are analyzed in Section 10.2. Finally, in Section 10.3, possible threats to the validity of the results are identified.

10.1 Main Results

The first experiment in Chapter 5 shows that open-source as well as commercial Optical Character Recognition engines have problems with recognizing rotated text from scientific figures. They are able to extract horizontal text and occasionally text which is rotated by 90 degree as these are the kind of text which are appear in scanned document images. The experiment further shows that a simple unsupervised approach can extract rotated text better than these Optical Character Recognition engine. However, since this unsupervised is not as well optimized as the OCR engines which were tuned for many years, the overall quality of the recognition result, including horizontal text, is not as good. While the unsupervised approach finds more text appearances in the scientific figures, the result is not clean from noise.

In the second experiment in Chapter 6, a comparison of the different, unsupervised, linear configurations of the pipeline that are inspired by approaches from the literature were conducted. The results show that the pipeline configuration *L7[Böschen]*, which was proposed by the author of this thesis, achieves the on average best results on all four datasets

(CHIME-R, CHIME-S, Economics and DeGruyter), taking all measures into account. The most probable reason is that the pipeline does not make many assumptions about the figures, e. g., figure type, font or color. Thus, performing better on the heterogeneous datasets which differ from the datasets on which the approaches in the literature were tested. However, other approaches are competitive with respect to certain measures, thus, a systematic analysis of the different methods has been conducted by modifying the configuration *L7[Böschen]*.

The third experiment in Chapter 7 presents the results for the individual pipeline steps based on the systematically modified linear configurations.

The first part of the pipeline is the extraction of regions which includes binarization and region extraction. When comparing the configurations that test the different binarization methods, one can observe that the adaptive binarization works best because it better adapts to local color variations in a figure. The standard Otsu's method only estimates one global threshold which is often insufficient for scientific figures which can contain multiple colors and grey levels for text as well as graphical elements and the background. Niblack's method also seems to have problems and works best when fewer color variations are present, i. e., it is more suited for document images. The non-competitive results for the pivoting algorithm for region extraction can be explained with the larger regions and the possibility that a region can be a mixture of text and graphic elements due to the only horizontal and vertical subdivision. Thus, Connected Component Labeling is the obvious choice.

The second step of the pipeline is the identification of text elements and the construction of text lines. Only one method, the morphological clustering method in configuration *M07[MORPH]* shows slightly better results than the base configuration *L7[Böschen]* which relies on a combination of DBSCAN and Minimum Spanning Tree clustering. The gain could be explained by the data on which the algorithms work. While DBSCAN and the Minimum Spanning Tree approach work on the centers of the character regions, which is an approximative representation, the morphological clustering method processes the character regions on pixel-level, thus having all the detailed information. However, the processing on pixel-level comes with the drawback of a large increase in processing time and memory consumption which was observed during the experiments (e. g., additional experiments with new configurations that use the morphological clustering did not finish at all). The gravity grouping based approach *M05[GRAVITY]* and the pure Minimum Spanning Tree based approach *M06[MST]* could not compete with the previously mentioned configurations, probably due to the inability to distinguish between text elements and graphic elements of similar characteristics.

The third and last step of the pipeline comprises all the necessities for text recognition. One important preprocessing step is the estimation of the orientation of a text line such that it can be rotated into horizontal alignment for correct recognition. When comparing the different orientation calculation methods, again, the pixel-based Single String Orientation Detection *M09[SSOD]* works best while the orientation estimation via Hough in the base configuration *L7[Böschen]* works only on the centers of mass of the character regions. However, the differences are minimal and an in-depth analysis of each text region would be needed to identify the advantages/disadvantages of the different methods which is out of scope of this thesis.

When comparing the three different Optical Character Recognition engines, Ocropy generally produces better results than Tesseract and the commercial ABBYY FineReader is some where in the middle. Ocropy seems to be more conservative than Tesseract, having built in much more restrictions about what input to accept and when to execute the OCR process, i. e., less noise gets processed. Furthermore, each OCR engine comes with its own English language model and, since the development of a new language model or Optical Character Recognition engine was not a goal of this thesis, no evaluation of the influence of different language models was conducted. Since all three Optical Character Recognition engines have similar results, including the commercial OCR engine ABBYY FineReader, one may draw the conclusion, that the Optical Character Recognition step does not have that much influence on the overall performance of the text extraction pipeline. This means that in the region extraction, region classification, text line detection or orientation estimation steps important information is lost so that the OCR engines are not able to produce better recognition results from the given input. The methods for postprocessing do not improve the results at all. One reason might be the simplicity of the methods. The other reason might be that the later steps of the pipeline have less and less influence on the overall results.

The overall best results for the linear configurations in the third experiment were achieved by the configuration *M09[SSOD]-O*. The two-pass configurations improve on this results by using the ABBYY FineReader as a first step to detect horizontal text and then applying the linear configuration *M09[SSOD]* to recognize rotated text. However, one difficulty is the choice of an appropriate decision criterion to accept the output of ABBYY FineReader in the first step. Using the minimum confidence value as criteria produced higher quality regarding the macro and micro Levenshtein Distances. Interestingly, the results regarding the macro and micro F1-Measures for text detection were not improved compared with the configuration using the average confidence value. This means that less output was accepted and removed from the figure but not more text was correctly recognized. One explanation might be that the linear configuration was not able to perform better on text which contained words with low confidence values. This might be due to bad image quality or characters/symbols that are unknown to the OCR engines and thus cannot be recognized. Another possible reason, which was not investigated further, is the modification of the image before applying the linear pipeline, which could have introduced additional noise since only the color white is used to erase the text that was already extracted.

One of the core motivations for building an unsupervised text extraction pipeline was the lack of data for training a supervised approach. However, due to the improvement with neural networks in recent years, it was possible to build a supervised approach that can handle limited training data. The fourth experiment in Chapter 8 presented and optimized a neural-network-based supervised approach for text detection from scientific figures. Optimizing the neural network showed that by using datasets from other domains (i. e., scene text detection) and by increasing the size of the set of scientific figures by creating modified copies of the original dataset, it is possible to train a neural network sufficiently enough such that it achieves promising results. It seems that the network is capable of recognizing features which characterize text in these scene text images which help to lay the foundation for the separation of graphical elements and text in scientific figures. The variation of text introduced to the artificial creation of modified images in the training dataset seems to work as well, to generalize the features that characterize text.

The fifth experiment in Chapter 9 made the comparison between the unsupervised approach and the supervised approach. The comparison highlighted that the supervised approach is capable of surpassing the unsupervised approach even though it has only limited training data available. However, the text detection result is still not perfect and to create the best possible approach, it would probably be necessary to create a gold standard with pixel level knowledge about characters which could be used to train a neural network, similar to the DarkNet [Red16] approach for natural images, because more and more detailed training data usually improves the performance of neural networks.

10.2 Limitations

There are certain limitations to both, supervised and unsupervised, approaches.

One of the major limitations of the evaluation of the proposed unsupervised pipeline, which is at the same time one of the strengths of the pipeline, is the flexibility of the pipeline. The module-based pipeline with its various options per step makes it in theory adaptable to different settings, however, at the same time it is impossible to test all possible configurations for all the possible different settings in reasonable time. Thus, one might only find a locally optimal solution and not the overall best solution. Even though the pipeline already offers a variety of methods, there are further methods that could be integrated. However, as already mentioned, the number of combinations is already quite high, so only the most promising and diverse methods were integrated.

Another limitation is that the pipeline is programmed in Java 8. Since Java code is executing inside the Java Virtual Machine, it is in generally slower than native code and makes it harder to assess the performance of individual parts of a Java program. Furthermore, the memory use of Java is way higher than with native code generated from other languages like C++. In addition, the garbage collector makes it difficult to properly assess and predict the memory consumption of a certain part of a Java program.

Thus, a performance evaluation of the proposed pipeline was omitted as the results were inconclusive. In addition, some configurations failed due to out-of-memory errors or never finished after weeks of execution which probably would not have happened with a more efficient implementation.

For the supervised approach, a brief analysis of the generalization capabilities showed that there is a decrease in the detection quality if unknown figure types are introduced. Thus, it would be necessary to train this neural network approach on as many figure types as possible to get the best performance. However, these datasets do not exist for all domains and their creation is expensive. Therefore, the results are still influenced by the lack of training data. In addition, the performance on the DeTEXT dataset is worse with pretraining on COCO-Text than without pretraining. The most likely reason here is that DeTEXT contains figure that contain subfigures which are very similar to the natural images of the COCO-Text dataset. Thus, in the pretraining, the network probably also learns features of the natural images and not only the text which lead to false detection on the natural image like subfigures of the DeTEXT dataset.

In addition, the experiments with the supervised approach were limited by the GPU memory of the infrastructure at hand. However, the limitation to an image width of 600px to 800px differs not so much from the average width of an image in the scientific datasets.

10.3 Threats to Validity

The proposed text extraction pipeline introduces a lot of flexibility and variables. Thus, one may question the validity of the presented results.

First, regarding the evaluation in general, one might argue that the dataset is too small to make general statements about the effectiveness of the proposes unsupervised pipeline. The collection of the gold-standard for a larger dataset via crowd sourcing had been planned, but after Amazon Mechanical Turk was limited to US-only, no suitable crowd sourcing platform remained where such a data collection would have been possible. Thus, one can only make the assumption that the semi-random selection of figures from DeGruyter and EconBiz documents is enough to show the generalization capabilities of the proposed approaches. Furthermore, the gold-standard has only a text-line level granularity, i. e., the evaluation can not fully assess the text recognition quality as strings are compared to each other and not individual characters. The creation of such a character-level gold-standard from real world figures would require exponentially more time and thus is unfeasible. In addition, the matching of text elements for comparison assumes that, if multiple elements are matched, they are all correct matches and that the recognized string just got subdivided. Thus, relying on the empirically estimated Intersection-over-Union threshold of 10%. However, it is impossible to optimize this threshold for the large variety of figures and therefore the empirically determined threshold needs to be sufficient. Finally, the reporting of the measures as averages over the figures of a dataset or all datasets does not account for the heterogeneous nature of the figures (given the high standard deviations), i. e., reporting per figure type could allow for more insights. However, also due to the heterogeneity, it is difficult to classify the figures into different types. Thus, the reporting of the averages is a valid choice to present the results.

Second, one might argue that the comparison of the different methods is flawed. The different methods taken from the literature had to be reimplemented as no implementations were available and all of the methods had to be integrated into one pipeline. The reimplementation was conducted by following the instructions given in the papers as closely as possible. Thus, the presented results should be (almost) identical as if the original implementation was used. When reimplementing the Single String Orientation Detection (SSOD), a default solution had to be implemented for the case when the algorithm can't find a proper orientation. The horizontal orientation of zero degree was selected as the default case. During recent experiments, the observation was made that the SSOD implementation often defaults to the horizontal orientation which could be the reason for the small improvement of the configuration *M09[SSOD]* over the original pipeline *L7[Böschen]* as the majority of text elements is of horizontal orientation. In addition, a couple of hyperparameters are used in the pipeline for which proper values had to be selected. Most of those values were chosen based on small empirical evaluations. Therefore, deliberately, more generic values were chosen to keep the generalization capabilities of the pipeline and not to optimize too much for the given datasets. This includes also the filter thresholds for graphical elements and the assumption that only text remains after generating text line elements. Here, on purpose, the thresholds were selected such that no text is lost even if that means that some graphical elements might make it to the text recognition step. Finally, for the same reason of not losing too much generalization capabilities, no language models were trained for the Optical Character Recognition engines, as for once, the figures in the datasets are quite heterogeneous and one may interchange the OCR engine or its language model at anytime.

Please note that also Google seems to has solved the problem with their Google Cloud Vision API which was used by Madan et al. [MBT+18] for text extraction from scientific figures even though the Google Cloud Vision API was designed for analyzing natural images. However, no papers could be found that describe the methods behind the Google Cloud Vision API and thus further investigations are needed to assess whether they are actually capable of extracting text from any kind of scientific figures. (A first, brief assessment of the Google Cloud Vision API showed that it only had problems with characters that were merged with graphical components.) In addition, Google has a much larger database at hand which most likely gives them an advantage when training supervised approaches which in other context just do not exist. Furthermore, due to privacy reasons, one might not want to give ones data into the hand of Google for processing, in certain situations. Thus, the text extraction pipeline that was proposed here could be an alternative that can be deployed locally.

Chapter 11

Conclusion and Outlook

This thesis gave an extensive overview of the field of text extraction from scientific figures. Five experiments were conducted to answer the research questions about what is the best approach for text extraction from scientific figures. The first experiment addressed the first research question **RQ1** and assessed the performance of common Optical Character Recognition tools for the task of text extraction from scientific figures. The second and third experiments gave answers to the research questions **RQ2** and **RQ3** about which unsupervised approach has the best results. In the fourth experiment the supervised approach was optimized. The final, fifth experiment made a comparison between the best supervised approach and the best unsupervised approach to answer the fourth research question **RQ4** on what is the overall best approach. The rest of this chapter summarizes the contributions of this thesis and gives and outlook for future work.

11.1 Summary of the Contributions

First, an extensive and thorough analysis of the related work in text extraction from scientific figures was conducted. Different, previously independent but related research areas were identified and set into comparison. From that analysis the conclusion was drawn that the approaches proposed so far do not have sufficient support for rotated text which is about 20-25% of the content of a scientific figure on average in the datasets used in this thesis. To address the lacking capability in handling rotating text, an unsupervised approach for text extraction from scientific figures was proposed. Based on the analysis of the related work, a generic pipeline for text extraction from scientific figures was constructed. The pipeline consists of three major parts which are common for most unsupervised approaches: region extraction, region grouping/classification and Optical Character Recognition. For each of the three parts, various methods were implemented that are inspired by the related work. Given its modular design, new methods could easily be integrated.

Second, to evaluate the pipeline and the various methods, two datasets were created. This was needed, since existing datasets lacked information about the rotation of text elements or did not have any rotated text. Therefore, two datasets of scientific figures with rotated text were manually annotated to create datasets that can be used to evaluate the extraction of rotated text. One set of scientific figures was taken from publications from the economics domain. The other set of figures came from scholarly books mostly from the chemistry domain. For both set of figure manually annotations of all text elements inside these figures including their orientation were created.

Third, an extensive evaluation of the different methods of the generic pipeline has been conducted to identify the best unsupervised approach for text extraction from scientific figures. The results showed that a slightly modified version of the originally proposed unsupervised approach produces the best results. However, the results also clearly still show room for improvements. Finally, a supervised approach was optimized which can work with a limited amount of training data. The comparison of the unsupervised approach with the supervised approach showed superior results with the supervised approach.

11.2 Future Work

One major insight, gained from the experiments conducted in this thesis, is the necessity for high quality datasets with gold standard information of the application domain regardless of whether one wants to find the optimal configuration of the unsupervised text extraction pipeline or train a supervised approach. So far, text extraction approaches were only tested on a small, limited number of domains. Thus, to develop a approach that works on scientific figures of any domain, it will be necessary to create new training datasets and/or tools to build these datasets on a large scale.

Furthermore, for an application in real-world scenarios, it is needed to optimize the performance of the proposed approaches. While the supervised approaches have a long training time, the unsupervised approaches have an in comparison longer execution time per figure. Thus, it would be necessary to evaluate which kind of approach is feasible in what kind of real-world application.

A fast and easy to deploy text extraction tool, together with a tool for dataset annotation, would allow for local deployment at companies and institutions, making one independent of services like the Google Cloud Vision API. One possible future extension would be to combine the text extraction pipeline and the annotation tool into one solution and thus allowing to fine-tune the pipeline while creating additional gold standard examples.

A second direction for future work is the integration of the text extraction into discovery systems which was outlined in the beginning of this thesis when motivating the text extraction from scientific figures. Initial experiments demonstrated (see Appendix A) that using text extracted from scientific figures of publications can be beneficial as this text often does not occur in the publication itself. It enables new and interesting access to scientific content, but it heavily depends on a good text extraction quality. The prototypical implementation shown in the Appendix is a first step, but to find the best integration into an existing platform is still an open research topic. The ultimate goal would be a pipeline which automatically indexes a publication when it is submitted and which performs a figure extraction from the publication, figure analysis and indexing while doing so.

Appendix A

SciFiS - A Prototype for Scientific Figure Search

In the literature, text extraction from (scientific) figures has been used to address various problems like reconstructing the original data a figure is based on [HT07; KBM+08; MV11; SKC+11; GZB12; CRM+17], creating a summary of the content of a figure [ECZ11; CSM+12] or translating figures into Braille [JRW+07; Tak09], to name a few. One alternative application scenario is to enhance the search for scientific content. Enhancing the search for scientific content, can be achieved in various ways.

In this section, the focus is on enhancing the search by extracting text from scientific figures and including it into the retrieval process. First, a motivation for this application scenario is discussed and existing related approaches from the literature are presented. Second, the actual prototype is outlined with its architecture, the underlying data (model) and a brief evaluation. Finally, a conclusion is drawn and options for improvement are highlighted.

A.1 Motivation

Most search engines like Google, Yahoo or Bing, are text-based or meta-data-based search engines. Even though they offer image search capabilities, all their functionality relies on the surrounding text of an image. The same holds true for specialized search engines that address specific problems like scientific documents of a certain domain like Medicine (e. g., PubMed[1] or Bio-Text[HDG+07]) even though some of them at least incorpo-

[1] `https://pubmed.ncbi.nlm.nih.gov/`, last accessed: September 2020

rate some sort of visual features (e. g., Medical Image Retrieval [KSD+17]). However, indexing scientific figures which are part of scientific documents can pose challenges. Nevertheless, these figures hold important information that is often not repeated in the rest of the document [CED06].

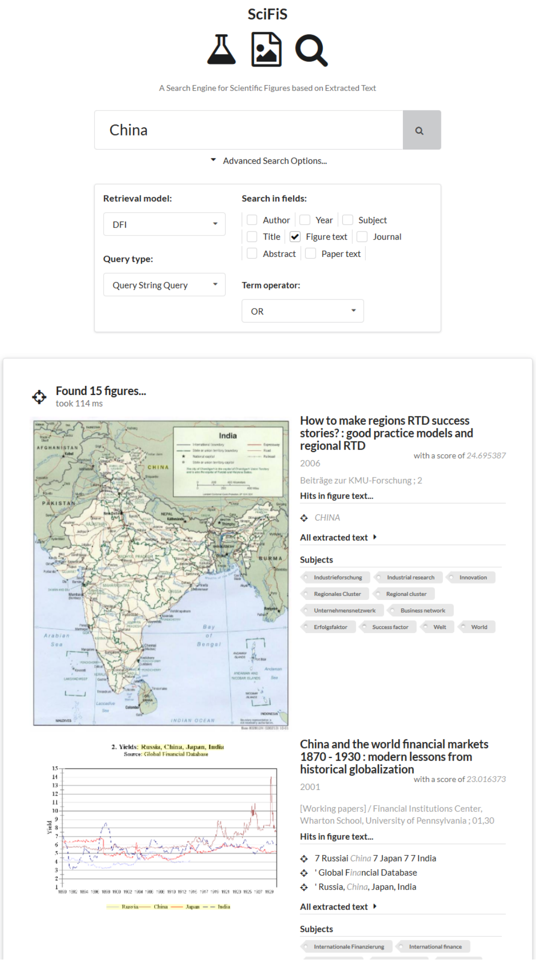
In the literature, different approaches have been proposed to utilize the extracted text from figures for information retrieval. In the biomedical domain, one can use the Yale Image Finder [XMK08; KLK12][2] while for chemistry, the research group of Lee Giles developed Chemseer [CTM+13; CG15]. Both approaches focus on the specificity of figures in their respective domain and apply restrictive filters on the extraction results to achieve a high precision in retrieval. DiagramFlyer [CCA15] is a more general approach. It extracts the text of the figure from the PDF itself, i. e., it does not analyze bitmaps but vector graphics where no text extraction from an image is needed. However, only a small fraction of figures are stored as vector graphics in documents.

In the next section, a prototype is described that uses extracted text from scientific figures to build a search engine on figures which are linked with their publication from the economics domain. Please note that the development, implementation and evaluation of the SciFiS prototype was conducted by Eike Lurz as part of his masters thesis who was supervised by the author of this thesis. Thus, due to the complexity given by the heterogeneity of figures in the economics domain, methodologies that rely on user query analysis and content mapping like the one proposed by Li et al. [LCF+14b] could not be applied.

A.2 Prototypical Implementation

With the SciFiS prototype (short for Scientific Figure Search), a search engine for scientific figures in the economics domain is proposed. Figure A.1 shows the prototype with an exemplary query result. First, a brief explanation of the data model and the data available in the prototype is presented before the software architecture of the prototype is described. Finally, an initial evaluation of the prototype is conducted.

[2] `http://sprout038.sprout.yale.edu/imagefinder/`, last accessed: December 2018



SciFiS

A Search Engine for Scientific Figures based on Extracted Text

China

Advanced Search Options...

Retrieval model: DFI

Query type: Query String Query

Search in fields:
☐ Author ☐ Year ☐ Subject
☐ Title ☑ Figure text ☐ Journal
☐ Abstract ☐ Paper text

Term operator: OR

Found 15 figures...
took 114 ms

How to make regions RTD success stories? : good practice models and regional RTD
with a score of *24.695387*
2006
Beiträge zur KMU-Forschung ; 2
Hits in figure text...
CHINA
All extracted text
Subjects
Industrieforschung · Industrial research · Innovation · Regionales Cluster · Regional cluster · Unternehmensnetzwerk · Business network · Erfolgsfaktor · Success factor · Welt · World

China and the world financial markets 1870 - 1930 : modern lessons from historical globalization
with a score of *23.016373*
2001
[Working papers] / Financial Institutions Center, Wharton School, University of Pennsylvania ; 01,30
Hits in figure text...
7 Russiai *China* 7 Japan 7 7 India
' Global F*ina*ncial Database
' Russia, *China*, Japan, India
All extracted text
Subjects
Internationale Finanzierung · International finance

Figure A.1. SciFiS search interface with two results for the search term "China" which is also highlighted in the figures.

A.2.1 Data

The dataset for the prototype was created by applying the best linear configuration *M09[SSOD]-O* to extract the text from more than 5,000 scientific figures. These figures were randomly extracted and selected from the Economics open access document corpora. The same corpora from which the Economics dataset of 121 figures was extracted that was used for evaluating the text extraction pipeline (see Section 4.1). This manually annotated set of 121 figures is later used in the evaluation as well.

Even though Li et al. [LCF+14b] came to the conclusion that the bag-of-words approach is not sufficient, there is no other option than the bag-of-words model for the SciFiS prototype due to the heterogeneity of the figures in the economics domain. Additionally, the proposed unsupervised text extraction pipeline only provides the raw text, i. e., no structural information is given.

Thus, as an alternative, not only the text extracted from the figures is indexed but the metadata of the paper from which the figures were taken as well. The following information is indexed: For each figure, the title, year, authors, journal, abstract and keywords of the enclosing paper are indexed as well as the content of the figure which is subdivided into several text elements. For each text element, the textual content is indexed and the position and orientation of the surrounding bounding box is stored. Two analyzers are implemented to index the textual content. The standard analyzer consists of the standard token filter, the lower case token filter and the stop token filter. In contrast, the custom analyzer does not filter stop tokens but builds n-grams from the tokens for a higher coverage.

A.2.2 Architecture

The SciFiS prototype consists of three components: the backend, the middleware and the frontend.

The backend of the SciFiS prototype is an Elasticsearch[3] instance. To keep it simple, no replicas and only two shards are used. The text extracted from the figures are indexed with the standard analyzer as well as the custom analyzer. The metadata is indexed as well.

[3]https://www.elastic.co/

A REST API forms the middleware that wraps the API of the backend and connects the backend with the frontend. This middleware offers the possibility of adding other services on top to query the data or to run multiple frontend instances.

The frontend is realized as a website and looks at a first glance similar to other search engines. However, as it is a research prototype, it offers a set of advanced options for querying specific data fields with different search modes as can be seen in Figure A.1. In total, seven different information retrieval models, four different query modes and seven different fields are in the selection pool. SciFiS allows to search in the text of a figure itself, as well as the metadata of the corresponding document. The result is an ordered list based on relevance where each entry corresponds to a relevant figure. Each figure is presented together with its extracted text as well as the metadata of the paper from which the figure was extracted. In addition, if the query text was found inside the figure's text, the corresponding bounding boxes inside the figure are highlighted for an easier assessment of the relevance by the user.

A.2.3 Evaluation

Building a search engine on the extracted text from scientific figures is challenging, since text in figures is often short, contains abbreviations and has lots of numbers. Furthermore, the unsupervised text extraction is not perfect as can be seen from the results presented in Chapter 7. Thus, the search engine also needs to handle flawed text. To find the best option, all seven different retrieval models, i. e., TF-IDF, BM25, Divergence from Randomness, Divergence from Independence, as well as language models are compared in combination with the standard analyzer. The evaluation showed only very small differences for NDCG@5 (see Appendix B for an explanation of NDCG) between the models. Under normal circumstances, tokenization and lemmatization would be used to optimize the retrieval performance. However, this leads to problems with numbers, short words and abbreviations. Thus, for comparison, n-grams with an empirically determined length of 3-5 were used for indexing by the custom analyzer. However, the comparison of the retrieval on the extracted text and the manually annotated text showed that recognition errors have the biggest impact on the retrieval performance and the use of n-grams could not counterbalance it.

A.3 Summary

The SciFiS prototype is a first step towards an enhanced retrieval of scientific content. Having access to the content of the figure enables new ways of accessing scientific documents. However, for a user-friendly effectiveness, the extraction quality of the text needs to improve. Using new technologies, like the proposed supervised extraction pipeline (see Chapter 9) or maybe the Google Cloud Vision API could make the approach feasible.

Appendix B

Basics

B.1 Methods and Algorithms

In this section, methods and algorithms are described in more detail which are either only briefly described in the thesis or assumed to be known. In the following, various methods and algorithms are presented in alphabetical order.

B.1.1 Conditional Dilation Algorithm

The CDA was proposed by Chiang and Knoblock [CK15]. It takes a binary image with connected components (foreground pixels) representing characters as input. The goal of the algorithm is to merge characters into text strings. The algorithm executes multiple iterations to merge the characters by expanding them. Before the first iteration, all connected components are marked as expandable. In every iteration the algorithm performs two passes over the image. In the first pass, every background pixel is tested for being an expansion candidate. A background pixel has to comply to four conditions to be an expansion candidate. The first condition is the *character connectivity condition* which means that a background pixel needs to be a neighbor of a foreground pixel of at least one and at most two different connected components. The second condition is the *character size condition* which enforces that characters that get merged through this background pixel need to be of similar size. They define the threshold such that two characters should differ in size at most by factor two. The third condition is the *character expandability condition* which checks that at least one of the connected components to which the background pixel is connected can still be expanded. A connected component is only expandable if it is connected to less than two other connected component, i. e., as soon as it is connected with two other connected components then it is not expandable anymore. The fourth condition is the *string curvature condition* which limits the curvature of three or more connected components.

In the second pass, every expansion candidate is checked to find those that should converted to foreground. This is needed since not all expansion candidates are known during the first pass. Thus, to avoid merging connected components which violate the four conditions, the expansion candidates need to be checked in the second pass.

The output is an image with the into text strings expanded characters.

B.1.2 (Convolutional) Neural Network

An artificial neural network is a supervised machine learning approach which aims at mimicking the neurons in a biological brain. A neural network consists of layers of nodes which are interconnected. The connections are called edges and have usually a weight assigned to them. A so-called activation function takes the weighted sum of the incoming edges as input and calculates the output of the node. Different activation functions, linear and nonlinear, are possible. The weights of the edges are learned for a specific task during the training phase. The nodes are often ordered in layers and edges exist primarily between layers. The first layer is called the input layer while the last layer is called the output layer. Figure B.1 shows a simple feedforward neural network.

The neural network in Figure B.1 is called a feedforward neural network because the nodes do not from a cycle. A feedforward neural network with three or more layers is also called a multilayer perceptron. The training of a multilayer perceptron, i. e., the learning of the weights, is usually done through backpropagation. Multilayer perceptron are often fully connected, i. e., a node is connected to all nodes of the next layer and so on. However, this makes them prone to overfitting.

A convolutional neural network is a special kind of multilayer perceptron. Classic multilayer perceptron have a flat input vector. However, when analyzing images, a flat input vector is not a good choice as it loses spatial information. The CNN has its name from the convolutional layers which use a kernel to reduce the dimensionality while keeping spatial

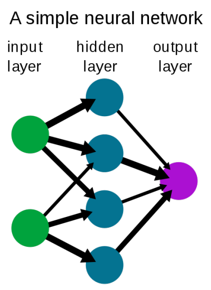


Figure B.1. An simple neural network with three layers (Source: Wikipedia Public Domain).

information. A layer of a Convolutional Neural Network usually consists of a convolutional layer and a pooling layer. Multiple of these layers can be stacked on top of each other. The output of the last pooling layer is flattened and is fed for example to a regular fully-connected feedforward neural network for classification.

B.1.3 Density-Based Spatial Clustering of Applications with Noise

The DBSCAN algorithm [EKS+96] is a special kind of clustering. As the name suggests, it a density-based clustering which also has a concept of outliers. In contrast to k-Means clustering, it does not need to know the number of cluster beforehand. DBSCAN has two parameters: The minimum number of points *minPts* a point needs to have - including itself - in its neighborhood, such that it can be a so called core point of a cluster. The other parameter is the epsilon parameter ϵ which defines the size of the neighborhood. The algorithm itself iterates over all datapoints once and checks whether a datapoint is a core point. It starts with a datapoint which has not been classified yet. If it is a core point, based on the ϵ and *minPts* parameter, then a new cluster is created an all points in the ϵ-neighborhood are added to the cluster and for each data point the check for being a core point is performed as well to grow the cluster. Once a cluster is completed that way, the next unclassified datapoint is analyzed. Datapoints that are not classified as core points and are not in the ϵ-neighborhood of a core point are classified as noise.

The advantages of DBSCAN are:

- The number of clusters does not need to be known beforehand
- The clusters can be arbitrarily shaped
- It has notion of noise

However, the algorithm has also some disadvantages:

- DBSCAN is not entirely deterministic, i. e., depending on the order, the datapoints are processed in, the result can vary. Some border points might be assigned to a different cluster, if they are in the range of two clusters. Or a border point can get classified as noise if the algorithm tries to start a cluster with it.
- In high dimensional feature spaces, a well selected distance function is needed, because DBSCAN is threatened by the curse of dimensionality.
- If the data as large differences in density, then it can be difficult to find appropriate parameter values for ϵ and *minPts*.

B.1.4 Faster R-CNN

Faster R-CNN [RHG+15] is an improved version of Fast R-CNN [Gir15], an object detection system for locating and classifying objects in an image. They first create region proposals which are classified in a second step.

Object detection systems prior to Fast R-CNN consisted of multiple components, for example a Convolutional Neural Network, a Support Vector Machine and a bounding box regressor in the R-CNN system [GDD+14], each of which could only be trained separately. Fast R-CNN solved these performance issues by building one system that could be trained end-to-end. Figure B.2 shows the architecture of Fast R-CNN. First, the image is processed with a Convolutional Neural Network to create a feature map. Second, for each region proposoal, the corresponding part of the feature map is extracted. These so called region proposal feature maps are resized with a pooling layer to a fixed size (see Section B.1.10). The fixed size feature map for each region is flattened into a vector which is the input to two fully connected neural networks for softmax classification and bounding box regression.

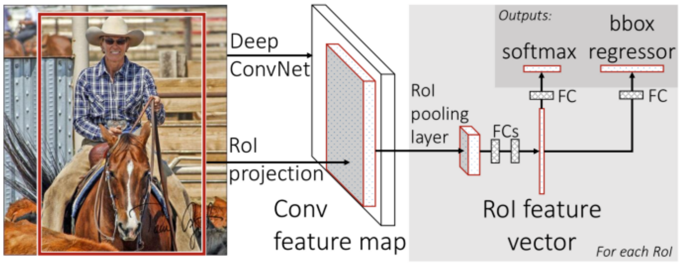


Figure B.2. The architecture of the Fast R-CNN object detection system. [Gir15]

Fast R-CNN is multiple times faster than a simple R-CNN. However, when assessing the performance of Fast R-CNN, the generation of region proposals was excluded which is in comparison an expensive step that was performed by the CPU. Faster R-CNN addresses this problem by adding a Region Proposal Network (RPN) into the Fast R-CNN architecture. The input image gets resized to a fix size and is fed to a CNN which is the backbone for both, the RPN and the classifier, and generates feature maps for both tasks. Thus, the computation is performed on the GPU as well. Please note that not all layers of the convolutional networks are used for both tasks, i. e., the number of shared layers is limited. Figure B.3 shows this combined architecture.

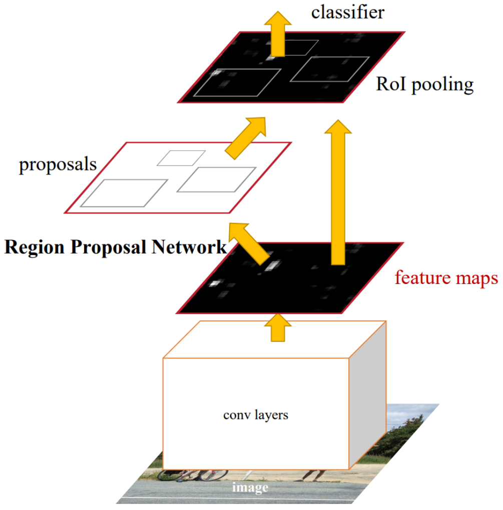
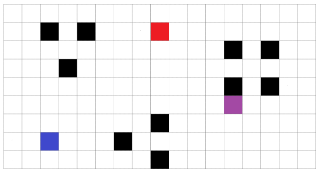


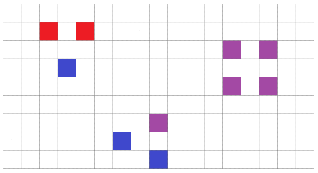
Figure B.3. The Faster R-CNN Object Detection Network [RHG+15].

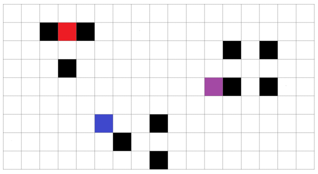
This combined architecture enables training the RPN by backpropagation and stochastic gradient descent. Alternating training is used, i. e., the network is alternately trained with both tasks.

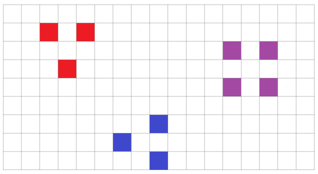
B.1.5 K-Means

A very popular clustering algorithms is the k-Means algorithm [Llo82]. It separates the data into k spherical-shaped clusters which are represented by cluster means through minimizing the within-cluster variances. Given the datapoints and an initial set of k means, the algorithm executes two alternating steps until the means converge. In the first step, the assignment step, each datapoint is assigned to the nearest mean. In the second step, the update step, each mean is recalculated using the assigned datapoints. Figure B.4 shows an exemplary execution of the k-Means algorithm.


(a) A set of datapoints (black) and three initial cluster means (red, blue and purple).


(b) The datapoints assigned to the nearest cluster means (visualized by color) in the first iteration.


(c) The datapoints with the recalculated cluster means after the first iteration.


(d) The datapoints assigned to the nearest cluster means (visualized by color) in the second iteration.

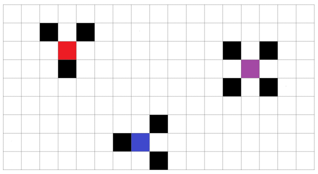

(e) The datapoints with the recalculated cluster means after the second iteration. The algorithm terminates here.

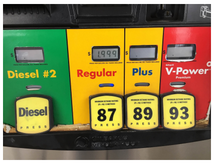
Figure B.4. Example of the k-Means algorithm on an example dataset. The algorithm terminates after two iterations (no change in the cluster means).

B.1.6 Mean-Shift

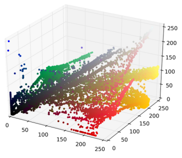
The Mean Shift algorithm [CM02] is a feature-space analysis method. The algorithm locates the maxima of a density function from a sample set of datapoints. It starts with an initial estimate of the mean x. A kernel function $K(x_i - x) = e^{-c||x_i - x||^2}$, here for example the Gaussian kernel function, is used to calculate the mean using Equation B.1.1.

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)} \tag{B.1.1}$$

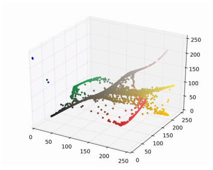
$N(x)$ is a set of points which represent the neighborhood of x and for which $K(x_i) \neq 0$. The mean shift is the difference $m(x) - x$. This calculation is repeated by setting $x \leftarrow m(x)$ until $m(x)$ converges. This is done for all datapoints which should all converge to the maxima in the dataset. Figure B.5 shows an application of mean shift for color reduction.



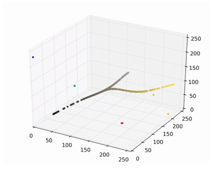
(a) The input to the Mean Shift algorithm: A color image.



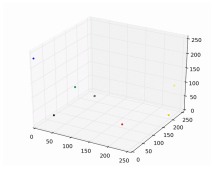
(b) The color values of all pixels in the RGB color space.



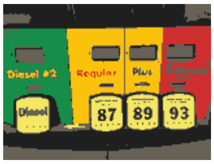
(c) The color values in the RGB color space after some iterations of the mean shift algorithm.



(d) The color values in the RGB color space after some more iterations of the mean shift algorithm.



(e) The final set of color values which converged to the means of the color values.



(f) The input image where each pixels color is replaced by the associated mean color.

Figure B.5. Color space reduction of an image using Mean Shift.
Source: `https://spin.atomicobject.com/2015/05/26/mean-shift-clustering/`

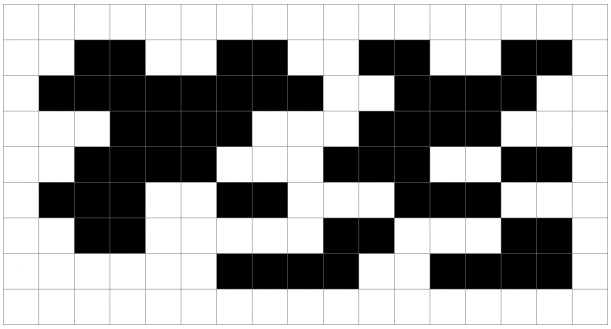
B.1.7 Median-Cut

The Media-Cut algorithm [Hec82] is a sorting algorithm for n-dimensional data and is often used for color quantization. Given a color image as input, it aims at generating a set of representative colors where each representative color represents an equal amount of pixel in the image. The algorithm first calculates the minimal enclosing bounding box in the color space which contains all colors of the image. Next, the algorithm sorts the colors by variance and splits the bounding box in two boxes at the median of the sorted colors. This process is repeated until the desired number of colors is reached, represented through a bounding box. For each bounding box, the median color is calculated which is then used to replace the original color in the image.

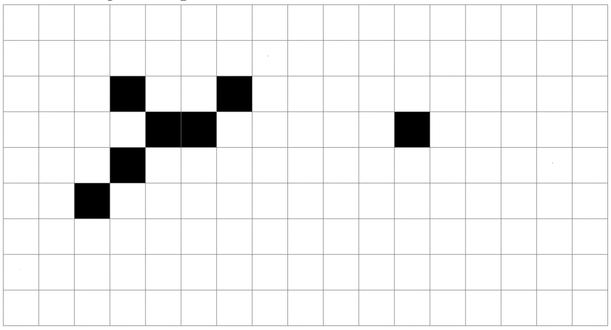
B.1.8 Morphological Operations

Different morphological operations exist that work on the pixels of an image. The two base operations are dilation and erosion. The dilation operation adds all background pixel to the foreground which have a foreground pixel as a neighbor. The erosion operation removes all foreground pixel which have a background pixel as a neighbor. The neighborhood of a pixel can be defined with an arbitrary structural element and can have a radius greater than 1. Figure B.6 shows the application of the erosion operator and dilation operator using the 4-pixel-neighborhood.

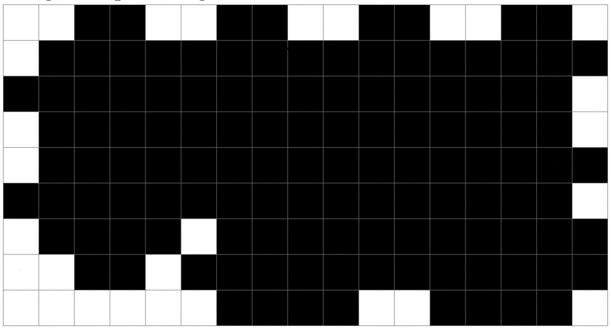
From these two operations, other operations can be derived. The closing operation first executes a dilation followed up by an erosion. This closes small holes in structures. The open operation applies an erosion operation followed by a dilation operation and thus removes small noise elements. Figure B.7 shows the application of the close operator and open operator using the 4-neighborhood.



(a) Example of a simple binary image. Foreground pixel are black while background pixel are white.

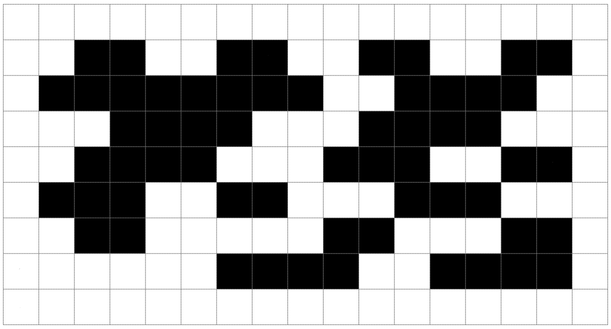


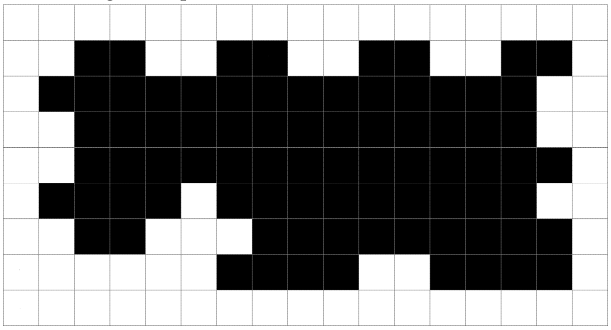
(b) The result after applying the morphological erosion operator using the 4-pixel-neighborhood.



(c) The result after applying the morphological dilation operator using the 4-pixel-neighborhood.

Figure B.6. Examples for the erosion operator and dilation operator using 4-pixel-neighborhood.


(a) Example of a simple binary image. Foreground pixel are black while background pixel are white.


(b) The result after applying the morphological close operator using the 4-pixel-neighborhood.

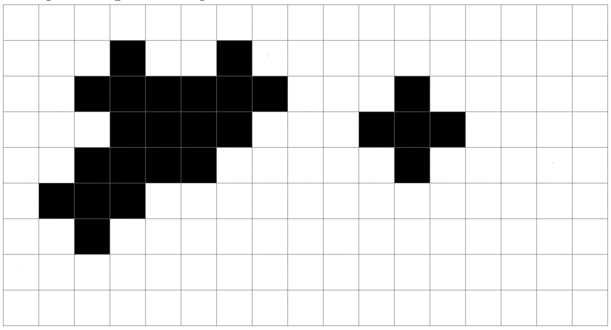

(c) The result after applying the morphological open operator using the 4-pixel-neighborhood.

Figure B.7. Examples for the close operator and open operator using 4-pixel-neighborhood.

B.1.9 Minimum Spanning Tree Clustering

The Minimum Spanning Tree (MST) clustering is a clustering algorithm with a lot of flexibility. Like other clustering algorithms, it makes the assumption that datapoints which are close together in the feature space probably belong to the same cluster. Therefore, it first computes a Minimum Spanning Tree on the datapoints. Figure B.8 illustrates such a Minimum Spanning Tree, however often more than one Minimum Spanning Tree is possible.

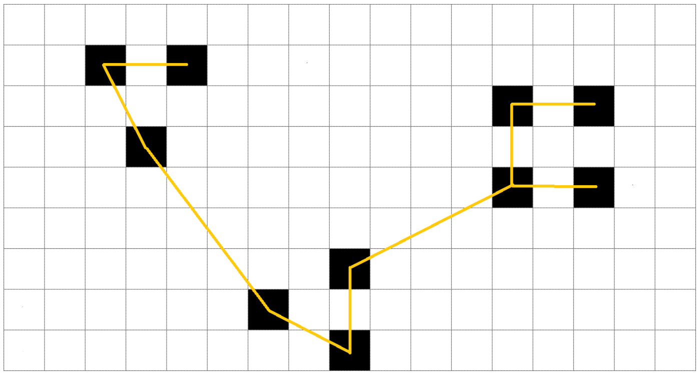


Figure B.8. An example of a Minimum Spanning Tree build on top a set of datapoints.

Once the MST is built, one needs to decide how to split it into clusters. The algorithm is quite flexible in how to determine this and different versions of the algorithm exist. Common criteria to split the minimum spanning tree are:

- ▻ Remove the n longest edges to create $n + 1$ clusters
- ▻ Remove all edges which are inconsistent, i. e., which differ a lot from the local edge length connected to a node
- ▻ Given a common feature for most edges in the minimum spanning tree, for example orientation, remove all edges which differ a lot from it

Figure B.9 shows the removal of the longest edges to create three clusters.

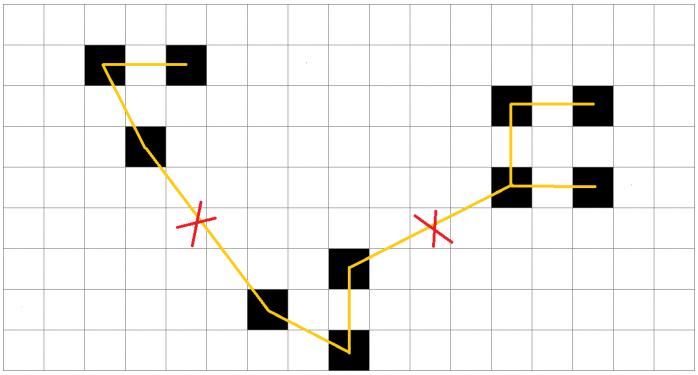


Figure B.9. An example of a MST build on top a set of datapoints with the longest edges marked for removal.

B.1.10 Region of Interest Align/Pool

When a Region Proposal Network proposes bounding boxes of interesting candidates, these bounding boxes are defined by coordinates based on the size of the original image. However, to get the Region of Interests (RoIs), it is needed to crop from the feature map which size was decreased via convolutions. Two common options to solve this problem are Region of Interest Align (RoI Align) and Region of Interest Pool (RoI Pool).

RoI Pool divides each coordinate by k, the factor the feature map was scaled down with, to create a RoI in the feature map. However, these Region of Interest can have different sizes and it is necessary to convert them to the same fixed size for further processing. This is were the pooling comes into play. Given the new coordinates for the feature map and using quantization, a max pooling is applied to create the final, fixed size region of interest. However, due to quantization, data can be lost, if the width and height can be mapped without loss. Figure B.10 illustrates the RoI Pool approach.

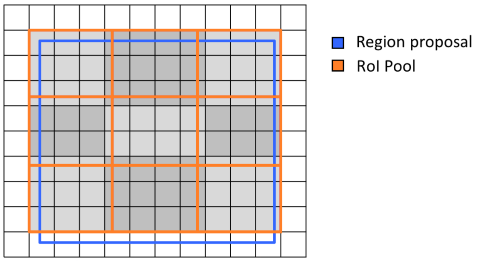


Figure B.10. A region proposal on a 10×15 feature map which is mapped to a 3×3 RoI Pooling output.

RoI Align is an alternative to RoI Pool without the need for quantization. Instead the original Region of Interest is divided into 9 equal-sized boxes and bilinear interpolation is used to calculate the value for each box. For each box, four sampling points are created by diving the width and height of the box by 3. These sampling points are then used for the bilinear interpolation. Figure B.11 illustrates this process.

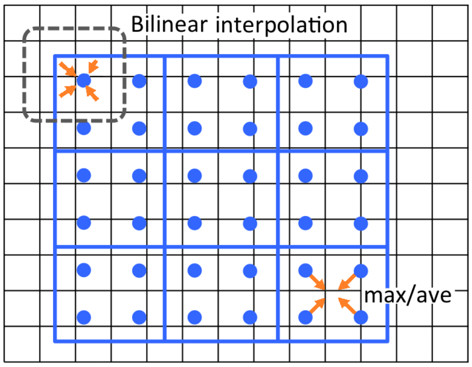


Figure B.11. A region proposal on a 10×15 feature map which is mapped to a 3×3 output by using RoI Align which uses four sample points per box for bilinear interpolation.

B.2 Evaluation Measures

In this section, evaluation measurements that are used in the thesis are described in more detail.

B.2.1 Levenshtein Distance

The Levenshtein Distance [Lev66] is a string metric which measures the minimum number of operations between two strings. The set of operations includes insertions, deletions and substitutions. Given two strings x and y is given by $lev_{x,y}(|x|,|y|)$ where $|x|$ is the length of string x and $|y|$ is the length of string y. The function $lev_{x,y}(a,b)$ is the distance between the first a characters of x and the first b characters of y and is defined as follows:

$$lev_{x,y}(a,b) = \begin{cases} max(a,b) & \text{if } min(a,b) = 0, \\ min \begin{cases} lev_{x,y}(a-1,b) + 1 \\ lev_{x,y}(a,b-1) + 1 \\ lev_{x,y}(a-1,b-1) + 1_{(x_a \neq y_b)} \end{cases} & \text{otherwise.} \end{cases} \tag{B.2.1}$$

With $1_{(x_a \neq y_b)}$ being the indicator function which equals to 0 if $x_a = y_b$ and otherwise is 1. The min-clause represents the cases, from top to bottom, deletion, insertion and match/mismatch.

For example, the Levenshtein distance between *house* and *mouse* is 1 as only one substitution is needed. In contrast, the Levenshtein distance between *mouse* and *cat* is 5 as three substitutions and two deletions are necessary to convert *mouse* to *cat*.

B.2.2 Normalized Discounted Cumulative Gain

Normalized Discounted Cumulative Gain (NDCG) is a measure for ranking quality which is often used when evaluating search engines. To understand NDCG, it is needed to first look at the Cumulative Gain (CG) and Discounted Cumulative Gain (DCG).

Given a ranked list of results with a graded relevance scores, the Cumulative Gain at a position p in the list can be computed using the Equation B.2.2.

$$CG_p = \sum_{i=1}^{p} rel_i \tag{B.2.2}$$

The Cumulative Gain is the sum of all graded relevance scores from the top of the list to position p. The problem with CG is that given a fixed p the order of the relevant items does not matter as the sum stays the same.

The Discounted Cumulative Gain addresses this problem by penalizing highly relevant items at low positions. Equation B.2.3 shows the calculation for DCG.

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)} \tag{B.2.3}$$

However, DCG has problems when different queries return lists of varying length. Normalized Discounted Cumulative Gain solves this by calculating the Ideal Discounted Cumulative Gain (IDCG) for a position p and uses this to normalize it according to Equation B.2.4.

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{B.2.4}$$

The result is always in the range of 0 to 1 (perfect score).

B.2.3 Precision, Recall and F1-Measure

Precision (P), Recall (R) and F1-Measure (F1) are common measures to evaluate results from experiments in pattern recognition, information retrieval or classification. Precision sets the correctly detected items in relation to all detected items. Recall sets the correctly detected items in relation to all relevant items. They are usually defined using the notion of True Positives (TP), False Positives (FP), False Negatives (FN) and True Negatives (TN). Equation B.2.5 and Equation B.2.6 show the definitions of Precision and Recall using these terms.

$$P = \frac{TP}{TP + FP} \tag{B.2.5}$$

$$R = \frac{TP}{TP + FN} \tag{B.2.6}$$

It is easy to optimize for one or the other, but in general one prefers to have both, a high Precision and a high Recall. Thus, the harmonic F1-Measure is often used to calculate a combined measure. Equation B.2.7 shows the definition of the F1-Measure.

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{B.2.7}$$

Bibliography

[ACG16] Rabah A. Al-Zaidy, Sagnik Ray Choudhury, and C. Lee Giles. "Automatic Summary Generation for Scientific Data Charts". In: *Scholarly Big Data: AI Perspectives, Challenges, and Ideas, Papers from the 2016 AAAI Workshop, Phoenix, Arizona, USA, February 13, 2016.* Ed. by Madian Khabsa, C. Lee Giles, and Alex D. Wade. Vol. WS-16-13. AAAI Workshops. AAAI Press, 2016. URL: http://www.aaai.org/ocs/index.php/WS/AAAIW16/paper/view/12661.

[AG15] Rabah A. Al-Zaidy and C. Lee Giles. "Automatic Extraction of Data from Bar Charts". In: *Proceedings of the 8th International Conference on Knowledge Capture, K-CAP 2015, Palisades, NY, USA, October 7-10, 2015.* Ed. by Ken Barker and José Manuél Gómez-Pérez. ACM, 2015, 30:1–30:4. DOI: 10.1145/2815833.2816956. URL: https://doi.org/10.1145/2815833.2816956.

[AG17] Rabah A. Al-Zaidy and C. Lee Giles. "A Machine Learning Approach for Semantic Structuring of Scientific Charts in Scholarly Documents". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.* Ed. by Satinder P. Singh and Shaul Markovitch. AAAI Press, 2017, pp. 4644–4649. URL: http://aaai.org/ocs/index.php/IAAI/IAAI17/paper/view/14275.

[AM10] A. Azizan and Itaza Afiani Mohtar. "The effects of surrounding text in image indexing". In: *2010 International Conference on Science and Social Research (CSSR 2010).* Dec. 2010, pp. 196–200. DOI: 10.1109/CSSR.2010.5773765.

[AOC+00] Sébastien Adam, Jean-Marc Ogier, Claude Cariou, Rémy Mullot, Jacques Labiche, and Joël Gardes. "Symbol and character recognition: application to engineering drawings". In:

IJDAR 3.2 (2000), pp. 89–101. DOI: 10.1007/s100320000033. URL: https://doi.org/10.1007/s100320000033.

[AOC+99] Sébastien Adam, Jean-Marc Ogier, Claude Cariou, Rémy Mullot, Joël Gardes, and Yves Lecourtier. "Multi-scaled and Multi-oriented Character Recognition: An Original Strategy". In: *Fifth International Conference on Document Analysis and Recognition, ICDAR 1999, 20-22 September, 1999, Bangalore, India*. IEEE Computer Society, 1999, pp. 45–48. DOI: 10.1109/ICDAR.1999.791721. URL: https://doi.org/10.1109/ICDAR.1999.791721.

[APR19] Ibrahim Almakky, Vasile Palade, and Ariel Ruiz-Garcia. "Deep Convolutional Neural Networks for Text Localisation in Figures From Biomedical Literature". In: *International Joint Conference on Neural Networks, IJCNN 2019 Budapest, Hungary, July 14-19, 2019*. IEEE, 2019, pp. 1–5. DOI: 10.1109/IJCNN.2019.8852353. URL: https://doi.org/10.1109/IJCNN.2019.8852353.

[ARO+01] Sébastien Adam, F. Rousseau, Jean-Marc Ogier, Claude Cariou, Rémy Mullot, Jacques Labiche, and Joël Gardes. "A multi-scale and multi-orientation recognition technique applied to document interpretation application to french telephone network maps". In: *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2001, 7-11 May, 2001, Salt Palace Convention Center, Salt Lake City, Utah, USA, Proceedings*. IEEE, 2001, pp. 1509–1512. DOI: 10.1109/ICASSP.2001.941218. URL: https://doi.org/10.1109/ICASSP.2001.941218.

[BA12a] Youssef Bassil and Mohammad Alwani. "OCR Context-Sensitive Error Correction Based on Google Web 1T 5-Gram Data Set". In: *CoRR* abs/1204.0188 (2012). arXiv: 1204.0188. URL: http://arxiv.org/abs/1204.0188.

[BA12b] Youssef Bassil and Mohammad Alwani. "OCR Post-Processing Error Correction Algorithm using Google Online Spelling Suggestion". In: *CoRR* abs/1204.0191 (2012). arXiv: 1204.0191. URL: http://arxiv.org/abs/1204.0191.

[BAO11] Bilal Bataineh, Siti Norul Huda Sheikh Abdullah, and Khairuddin Omar. "An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows". In: *Pattern Recognition Letters* 32.14 (2011), pp. 1805–1813. DOI: 10.1016/j.patrec.2011.08.001. URL: https://doi.org/10.1016/j.patrec.2011.08.001.

[BBS18] Falk Böschen, Tilman Beck, and Ansgar Scherp. "Survey and empirical comparison of different approaches for text extraction from scholarly figures". In: *Multimedia Tools Appl.* 77.22 (2018), pp. 29475–29505. DOI: 10.1007/s11042-018-6162-7. URL: https://doi.org/10.1007/s11042-018-6162-7.

[BCE08] Richard Burns, Sandra Carberry, and Stephanie Elzer. "Processing Information Graphics in Multimodal Documents". In: *Multimedia Information Extraction, Papers from the 2008 AAAI Fall Symposium, Mark Maybury (chair) and Sharon Walter (cochair), Arlington, Virginia, USA, November 7-9, 2008*. Vol. FS-08-05. AAAI Technical Report. AAAI, 2008, pp. 5–9. URL: http://www.aaai.org/Library/Symposia/Fall/2008/fs08-05-004.php.

[BCE10] Richard Burns, Sandra Carberry, and Stephanie Elzer. "Visual and Spatial Factors in a Bayesian Reasoning Framework for the Recognition of Intended Messages in Grouped Bar Charts". In: *Visual Representations and Reasoning, Papers from the 2010 AAAI Workshop, Atlanta, Georgia, USA, July 11, 2010*. Vol. WS-10-07. AAAI Workshops. AAAI, 2010. URL: http://aaai.org/ocs/index.php/WS/AAAIW10/paper/view/2017.

[Ber86] John Bernsen. "Dynamic thresholding of gray-level images". In: *Proceedings - International Conference on Pattern Recognition* (Jan. 1986).

[BGS18] Fedor Borisyuk, Albert Gordo, and Viswanath Sivakumar. "Rosetta: Large Scale System for Text Detection and Recognition in Images". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*. Ed. by Yike Guo and Faisal Farooq. ACM, 2018, pp. 71–79. DOI: 10.1145/3219819.3219861. URL: http://doi.acm.org/10.1145/3219819.3219861.

[BS15a] Falk Böschen and Ansgar Scherp. "Formalization and Preliminary Evaluation of a Pipeline for Text Extraction From Infographics". In: *Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB, Trier, Germany, October 7-9, 2015*. Ed. by Ralph Bergmann, Sebastian Görg, and Gilbert Müller. Vol. 1458. CEUR Workshop Proceedings. CEUR-WS.org, 2015, pp. 20–31. URL: http://ceur-ws.org/Vol-1458/D03_CRC13_Boeschen.pdf.

[BS15b] Falk Böschen and Ansgar Scherp. "Multi-oriented Text Extraction from Information Graphics". In: *Proceedings of the 2015 ACM Symposium on Document Engineering, DocEng 2015, Lausanne, Switzerland, September 8-11, 2015*. Ed. by Christine Vanoirbeek and Pierre Genevès. ACM, 2015, pp. 35–38. DOI: 10.1145/2682571.2797092. URL: http://doi.acm.org/10.1145/2682571.2797092.

[BS16] Falk Böschen and Ansgar Scherp. *A systematic comparison of different approaches of unsupervised extraction of text from scholary figures [extended report]*. Tech. rep. 1607. Institut für Informatik der Christian-Albrechts-Universität zu Kiel, Nov. 2016. URL: https://www.uni-kiel.de/journals/receive/jportal_jparticle_00000290.

[BS17] Falk Böschen and Ansgar Scherp. "A Comparison of Approaches for Automated Text Extraction from Scholarly Figures". In: *MultiMedia Modeling - 23rd International Conference, MMM 2017, Reykjavik, Iceland, January 4-6, 2017, Proceedings, Part I*. Vol. 10132. Lecture Notes in Computer Science. Springer, 2017, pp. 15–27. DOI: 10.1007/978-3-319-51811-4_2. URL: http://dx.doi.org/10.1007/978-3-319-51811-4_2.

[CC17] Chee Kheng Chng and Chee Seng Chan. "Total-Text: A Comprehensive Dataset for Scene Text Detection and Recognition". In: *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017*. IEEE, 2017, pp. 935–942. DOI: 10.1109/ICDAR.2017.157. URL: https://doi.org/10.1109/ICDAR.2017.157.

[CCA15] Zhe Chen, Michael J. Cafarella, and Eytan Adar. "Diagram-Flyer: A Search Engine for Data-Driven Diagrams". In: *Proceedings of the 24th International Conference on World Wide Web*

Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume. Ed. by Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi. ACM, 2015, pp. 183–186. DOI: 10.1145/2740908.2742831. URL: http://doi.acm.org/10.1145/2740908.2742831.

[CE05] Daniel Chester and Stephanie Elzer. "Getting Computers to See Information Graphics So Users Do Not Have to". In: *Foundations of Intelligent Systems, 15th International Symposium, ISMIS 2005, Saratoga Springs, NY, USA, May 25-28, 2005, Proceedings*. Ed. by Mohand-Said Hacid, Neil V. Murray, Zbigniew W. Ras, and Shusaku Tsumoto. Vol. 3488. Lecture Notes in Computer Science. Springer, 2005, pp. 660–668. DOI: 10.1007/11425274_68. URL: http://dx.doi.org/10.1007/11425274_68.

[CED06] Sandra Carberry, Stephanie Elzer, and Seniz Demir. "Information graphics: an untapped resource for digital libraries". In: *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*. Ed. by Efthimis N. Efthimiadis, Susan T. Dumais, David Hawking, and Kalervo Järvelin. ACM, 2006, pp. 581–588. DOI: 10.1145/1148170.1148270. URL: http://doi.acm.org/10.1145/1148170.1148270.

[CG15] Sagnik Ray Choudhury and Clyde Lee Giles. "An Architecture for Information Extraction from Figures in Digital Libraries". In: *Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume*. Ed. by Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi. ACM, 2015, pp. 667–672. DOI: 10.1145/2740908.2741712. URL: http://doi.acm.org/10.1145/2740908.2741712.

[CK11] Yao-Yi Chiang and Craig A. Knoblock. "Recognition of Multi-oriented, Multi-sized, and Curved Text". In: *2011 International Conference on Document Analysis and Recognition, ICDAR 2011, Beijing, China, September 18-21, 2011*. IEEE Computer Society, 2011, pp. 1399–1403. DOI: 10.1109/ICDAR.2011.281. URL: https://doi.org/10.1109/ICDAR.2011.281.

[CK13] Yao-Yi Chiang and Craig A. Knoblock. “A general approach for extracting road vector data from raster maps”. In: *IJDAR* 16.1 (2013), pp. 55–81. DOI: 10.1007/s10032-011-0177-1. URL: https://doi.org/10.1007/s10032-011-0177-1.

[CK15] Yao-Yi Chiang and Craig A. Knoblock. “Recognizing text in raster maps”. In: *GeoInformatica* 19.1 (2015), pp. 1–27. DOI: 10.1007/s10707-014-0203-9. URL: http://dx.doi.org/10.1007/s10707-014-0203-9.

[CM02] Dorin Comaniciu and Peter Meer. “Mean Shift: A Robust Approach Toward Feature Space Analysis”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 24.5 (2002), pp. 603–619. DOI: 10.1109/34.1000236. URL: https://doi.org/10.1109/34.1000236.

[CRM+17] Mathieu Cliche, David S. Rosenberg, Dhruv Madeka, and Connie Yee. “Scatteract: Automated Extraction of Data from Scatter Plots”. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part I.* Ed. by Michelangelo Ceci, Jaakko Hollmén, Ljupco Todorovski, Celine Vens, and Saso Dzeroski. Vol. 10534. Lecture Notes in Computer Science. Springer, 2017, pp. 135–150. DOI: 10.1007/978-3-319-71249-9_9. URL: https://doi.org/10.1007/978-3-319-71249-9_9.

[CSM+12] Sandra Carberry, Stephanie Elzer Schwartz, Kathleen F. McCoy, Seniz Demir, Peng Wu, Charles F. Greenbacker, Daniel Chester, Edward Schwartz, David Oliver, and Priscilla S. Moraes. “Access to multimodal articles for individuals with sight impairments”. In: *ACM Trans. Interact. Intell. Syst.* 2.4 (2012), p. 21. DOI: 10.1145/2395123.2395126. URL: http://doi.acm.org/10.1145/2395123.2395126.

[CTM+13] Sagnik Ray Choudhury, Suppawong Tuarob, Prasenjit Mitra, Lior Rokach, Andi Kirk, Silvia Szep, Donald A. Pellegrino, Sue Jones, and Clyde Lee Giles. “A figure search engine architecture for a chemistry digital library”. In: *13th ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL '13, Indianapolis, IN, USA, July 22 - 26, 2013.* Ed. by J. Stephen Downie, Robert H.

McDonald, Timothy W. Cole, Robert Sanderson, and Frank Shipman. ACM, 2013, pp. 369–370. DOI: 10.1145/2467696.2467757. URL: https://doi.org/10.1145/2467696.2467757.

[CYL+20] Alex Carderas, Ye Yuan, Itamar Livnat, Ryan Yanagihara, Rosita Saul, Gabrielle Montes De Oca, Kai Zheng, and Andrew W. Browne. "Automated data extraction of bar chart raster images". In: *CoRR* abs/2011.04137 (2020). arXiv: 2011.04137. URL: https://arxiv.org/abs/2011.04137.

[DCM08] Seniz Demir, Sandra Carberry, and Kathleen F. McCoy. "Generating Textual Summaries of Bar Charts". In: *INLG 2008 - Proceedings of the Fifth International Natural Language Generation Conference, June 12-14, 2008, Salt Fork, Ohio, USA*. Ed. by Michael White, Crystal Nakatsu, and David McDonald. The Association for Computer Linguistics, 2008. URL: http://www.aclweb.org/anthology/W08-1103.

[Dem08] Seniz Demir. "TAIG: textually accessible information graphics". In: *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2008, Halifax, Nova Scotia, Canada, October 13-15, 2008*. Ed. by Simon Harper and Armando Barreto. ACM, 2008, pp. 313–314. DOI: 10.1145/1414471.1414555. URL: https://doi.org/10.1145/1414471.1414555.

[DH72] Richard O. Duda and Peter E. Hart. "Use of the Hough Transformation to Detect Lines and Curves in Pictures". In: *Commun. ACM* 15.1 (1972), pp. 11–15. DOI: 10.1145/361237.361242. URL: https://doi.org/10.1145/361237.361242.

[DML+97] Marc Pierrot Deseilligny, Robert Mariani, Jacques Labiche, and Rémy Mullot. "Topographic Maps Automatic Interpretation: Some Proposed Strategies". In: *Graphics Recognition, Algorithms and Systems, Second International Workshop, GREC'97, Nancy, France, August 22-23, 1997, Selected Papers*. Ed. by Karl Tombre and Atul K. Chhabra. Vol. 1389. Lecture Notes in Computer Science. Springer, 1997, pp. 175–193. DOI: 10.1007/3-540-64381-8_48. URL: https://doi.org/10.1007/3-540-64381-8_48.

[DMS95] Marc Pierrot Deseilligny, Hervé Le Men, and Georges Stamon. "Character string recognition on maps, a rotation-invariant recognition method". In: *Pattern Recognition Letters* 16.12 (1995), pp. 1297–1310. DOI: 10.1016/0167-8655(95)00084-5. URL: http://dx.doi.org/10.1016/0167-8655(95)00084-5.

[DOS+10] Seniz Demir, David Oliver, Edward Schwartz, Stephanie Elzer, Sandra Carberry, and Kathleen F. McCoy. "Interactive SIGHT into information graphics". In: *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, W4A 2010, Raleigh, NC, USA, April 26 - 27, 2010*. Ed. by Chieko Asakawa, Hironobu Takagi, Leo Ferres, and Cynthia C. Shelly. ACM, 2010, p. 16. DOI: 10.1145/1805986.1806009. URL: https://doi.org/10.1145/1805986.1806009.

[DWN+18] Wenjing Dai, Meng Wang, Zhibin Niu, and Jiawan Zhang. "Chart decoder: Generating textual and numeric information from chart images automatically". In: *J. Vis. Lang. Comput.* 48 (2018), pp. 101–109. DOI: 10.1016/j.jvlc.2018.08.005. URL: https://doi.org/10.1016/j.jvlc.2018.08.005.

[ECZ11] Stephanie Elzer, Sandra Carberry, and Ingrid Zukerman. "The automated understanding of simple bar charts". In: *Artif. Intell.* 175.2 (2011), pp. 526–555. DOI: 10.1016/j.artint.2010.10.003. URL: https://doi.org/10.1016/j.artint.2010.10.003.

[EKS+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*. Ed. by Evangelos Simoudis, Jiawei Han, and Usama M. Fayyad. AAAI Press, 1996, pp. 226–231. URL: http://www.aaai.org/Library/KDD/1996/kdd96-037.php.

[EOW10] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. "Detecting text in natural scenes with stroke width transform". In: *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June*

2010. IEEE Computer Society, 2010, pp. 2963–2970. DOI: 10.1109/CVPR.2010.5540041. URL: https://doi.org/10.1109/CVPR.2010.5540041.

[EVW+10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. "The pascal visual object classes (voc) challenge". In: *International journal of computer vision* 88.2 (2010), pp. 303–338.

[FK88] Lloyd A. Fletcher and Rangachar Kasturi. "A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 10.6 (1988), pp. 910–918. DOI: 10.1109/34.9112. URL: https://doi.org/10.1109/34.9112.

[FSE15] Muhammad Fraz, M. Saquib Sarfraz, and Eran A. Edirisinghe. "Exploiting colour information for better scene text detection and recognition". In: *International Journal on Document Analysis and Recognition (IJDAR)* 18.2 (2015), pp. 153–167. DOI: 10.1007/s10032-015-0239-x. URL: http://dx.doi.org/10.1007/s10032-015-0239-x.

[GDD+14] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*. IEEE Computer Society, 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81. URL: https://doi.org/10.1109/CVPR.2014.81.

[GF05] Julinda Gllavata and Bernd Freisleben. "Adaptive Fuzzy Text Segmentation in Images with Complex Backgrounds Using Color and Texture". In: *Computer Analysis of Images and Patterns, 11th International Conference, CAIP 2005, Versailles, France, September 5-8, 2005, Proceedings*. Ed. by André Gagalowicz and Wilfried Philips. Vol. 3691. Lecture Notes in Computer Science. Springer, 2005, pp. 756–765. DOI: 10.1007/11556121_93. URL: http://dx.doi.org/10.1007/11556121_93.

[GHT00] P. Groot, F. van Harmelen, and A. ten Teije. “Torture Tests: A Quantitative Analysis for the Robustness of Knowledge-Based Systems”. In: *EKAW*. 2000. DOI: 10.1007/3-540-39967-4_31. URL: http://dx.doi.org/10.1007/3-540-39967-4_31.

[Gir15] Ross B. Girshick. “Fast R-CNN”. In: *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*. IEEE Computer Society, 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169. URL: https://doi.org/10.1109/ICCV.2015.169.

[GPP06] Basilios Gatos, Ioannis Pratikakis, and Stavros J. Perantonis. “Adaptive degraded document image binarization”. In: *Pattern Recognition* 39.3 (2006), pp. 317–327. DOI: 10.1016/j.patcog.2005.09.010. URL: https://doi.org/10.1016/j.patcog.2005.09.010.

[GZB12] Jinglun Gao, Yin Zhou, and Kenneth E. Barner. “View: Visual Information Extraction Widget for improving chart images accessibility”. In: *19th IEEE International Conference on Image Processing, ICIP 2012, Lake Buena Vista, Orlando, FL, USA, September 30 - October 3, 2012*. IEEE, 2012, pp. 2865–2868. DOI: 10.1109/ICIP.2012.6467497. URL: https://doi.org/10.1109/ICIP.2012.6467497.

[Hau14] F. Hausdorff. *Grundzüge der Mengenlehre*. Chelsea Publishing Co., 1914.

[HDG+07] Marti A. Hearst, Anna Divoli, Harendra Guturu, Alex Ksikes, Preslav Nakov, Michael A. Wooldridge, and Jerry Ye. “BioText Search Engine: beyond abstract search”. In: *Bioinformatics* 23.16 (2007), pp. 2196–2197. DOI: 10.1093/bioinformatics/btm301. URL: https://doi.org/10.1093/bioinformatics/btm301.

[Hec82] Paul S. Heckbert. “Color image quantization for frame buffer display”. In: *Proceedings of the 9th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1982, Boston, Massachusetts, USA, July 26-30, 1982*. Ed. by R. Daniel Bergeron. ACM, 1982, pp. 297–307. DOI: 10.1145/800064.801294. URL: https://doi.org/10.1145/800064.801294.

[HT07] Weihua Huang and Chew Lim Tan. "A system for understanding imaged infographics and its applications". In: *Proceedings of the 2007 ACM Symposium on Document Engineering, Winnipeg, Manitoba, Canada, August 28-31, 2007*. Ed. by Peter R. King and Steven J. Simske. ACM, 2007, pp. 9–18. DOI: 10.1145/1284420.1284427. URL: http://doi.acm.org/10.1145/1284420.1284427.

[HTL03] Weihua Huang, Chew Lim Tan, and Wee Kheng Leow. "Model-Based Chart Image Recognition". In: *Graphics Recognition, Recent Advances and Perspectives, 5th InternationalWorkshop, GREC 2003, Barcelona, Spain, July 30-31, 2003, Revised Selected Papers*. Ed. by Josep Lladós and Young-Bin Kwon. Vol. 3088. Lecture Notes in Computer Science. Springer, 2003, pp. 87–99. DOI: 10.1007/978-3-540-25977-0_8. URL: https://doi.org/10.1007/978-3-540-25977-0_8.

[HTL05] Weihua Huang, Chew Lim Tan, and Wee Kheng Leow. "Associating Text and Graphics for Scientific Chart Understanding". In: *Eighth International Conference on Document Analysis and Recognition (ICDAR 2005), 29 August - 1 September 2005, Seoul, Korea*. IEEE Computer Society, 2005, pp. 580–584. DOI: 10.1109/ICDAR.2005.54. URL: http://dx.doi.org/10.1109/ICDAR.2005.54.

[Hu62] M.-K. Hu. "Visual pattern recognition by moment invariants". In: *IRE Transactions on Information Theory* 8.2 (1962), pp. 179–187.

[HZR+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90. URL: https://doi.org/10.1109/CVPR.2016.90.

[Jac12] Paul Jaccard. "THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1". In: *New Phytologist* 11.2 (1912), pp. 37–50. DOI: 10.1111/j.1469-8137.1912.tb05611.x. eprint: https://nph.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1469-8137.1912.tb05611.x. URL: https://nph.onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-8137.1912.tb05611.x.

[JB82] G. Johannsen and J. Bille. "A threshold selection method using information measures". In: *6th International Conference on Pattern Recognition*. 1982, pp. 140–143.

[JBS19] Morten Jessen, Falk Böschen, and Ansgar Scherp. "Text Localization in Scientific Figures using Fully Convolutional Neural Networks on Limited Training Data". In: *Proceedings of the ACM Symposium on Document Engineering 2019, Berlin, Germany, September 23-26, 2019*. Ed. by Sonja Schimmler and Uwe M. Borghoff. ACM, 2019, 13:1–13:10. DOI: 10.1145/3342558.3345396. URL: https://doi.org/10.1145/3342558.3345396.

[Jiu06] Zhao Jiuzhou. *Creation of Synthetic Chart Image Database with Ground Truth*. Honors Year Project Report. https://www.comp.nus.edu.sg/~tancl/ChartImageDatabase/Report_Zhaojiuzhou.pdf. National University of Singapore, 2006.

[JRW+07] Chandrika Jayant, Matthew Renzelmann, Dana Wen, Satria Krisnandi, Richard E. Ladner, and Dan Comden. "Automated tactile graphics translation: in the field". In: *Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2007, Tempe, Arizona, USA, October 15-17, 2007*. Ed. by Enrico Pontelli and Shari Trewin. ACM, 2007, pp. 75–82. DOI: 10.1145/1296843.1296858. URL: http://doi.acm.org/10.1145/1296843.1296858.

[KBM+08] Saurabh Kataria, William Browuer, Prasenjit Mitra, and C. Lee Giles. "Automatic Extraction of Data Points and Text Blocks from 2-Dimensional Plots in Digital Documents". In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*. Ed. by Dieter Fox and Carla P. Gomes. AAAI Press, 2008, pp. 1169–1174. URL: http://www.aaai.org/Library/AAAI/2008/aaai08-185.php.

[KI86] Josef Kittler and John Illingworth. "Minimum error thresholding". In: *Pattern Recognition* 19.1 (1986), pp. 41–47. DOI: 10.1016/0031-3203(86)90030-0. URL: https://doi.org/10.1016/0031-3203(86)90030-0.

[KLK12] Tobias Kuhn, Thaibinh Luong, and Michael Krauthammer. "Finding and Accessing Diagrams in Biomedical Publications". In: *AMIA 2012, American Medical Informatics Association Annual Symposium, Chicago, Illinois, USA, November 3-7, 2012*. AMIA, 2012. URL: http://knowledge.amia.org/amia-55142-a2012a-1.636547/t-003-1.640625/f-001-1.640626/a-056-1.641066/a-057-1.641063.

[KSD+17] Ivan Kitanovski, Gjorgji Strezoski, Ivica Dimitrovski, Gjorgji Madjarov, and Suzana Loskovska. "Multimodal medical image retrieval system". In: *Multimedia Tools Appl.* 76.2 (2017), pp. 2955–2978. DOI: 10.1007/s11042-016-3261-1. URL: https://doi.org/10.1007/s11042-016-3261-1.

[KST+96] Koichi Kise, Tadamichi Shiraishi, Shinobu Takamatsu, and Kunio Fukunaga. "A method of post-processing for character recognition based on syntactic and semantic analysis of sentences". In: *Systems and Computers in Japan* 27.9 (1996), pp. 94–107. DOI: 10.1002/scj.4690270910. URL: https://doi.org/10.1002/scj.4690270910.

[KSW85] J. N. Kapur, Prasanna K. Sahoo, and Andrew K. C. Wong. "A new method for gray-level picture thresholding using the entropy of the histogram". In: *Computer Vision, Graphics, and Image Processing* 29.3 (1985), pp. 273–285. DOI: 10.1016/0734-189X(85)90125-2. URL: https://doi.org/10.1016/0734-189X(85)90125-2.

[LCF+14a] Zhuo Li, Sandra Carberry, Hui Fang, and Kathleen F. McCoy. "Identifying Important Features for Graph Retrieval". In: *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*. Ed. by Jan Hajic and Junichi Tsujii. ACL, 2014, pp. 600–609. URL: http://aclweb.org/anthology/C/C14/C14-1057.pdf.

[LCF+14b] Zhuo Li, Sandra Carberry, Hui Fang, Kathleen F. McCoy, and Kelly Peterson. "Infographics Retrieval: A New Methodology". In: *Natural Language Processing and Information Systems - 19th International Conference on Applications of Natural Language* *to Information Systems, NLDB 2014, Montpellier, France, June 18-20, 2014. Proceedings*. Ed. by Elisabeth Métais, Mathieu Roche, and Maguelonne Teisseire. Vol. 8455. Lecture Notes in Computer Science. Springer, 2014, pp. 101–113. DOI: 10.1007/978-3-319-07983-7_15. URL: https://doi.org/10.1007/978-3-319-07983-7_15.

[LCJ+10] Seonghun Lee, Min Su Cho, Kyomin Jung, and Jin Hyung Kim. "Scene Text Extraction with Edge Constraint and Text Collinearity". In: *20th International Conference on Pattern Recognition, ICPR 2010, Istanbul, Turkey, 23-26 August 2010*. IEEE Computer Society, 2010, pp. 3983–3986. DOI: 10.1109/ICPR.2010.969. URL: https://doi.org/10.1109/ICPR.2010.969.

[LCT+15] Shijian Lu, Tao Chen, Shangxuan Tian, Joo-Hwee Lim, and Chew Lim Tan. "Scene text extraction based on edges and support vector regression". In: *International Journal on Document Analysis and Recognition (IJDAR)* 18.2 (2015), pp. 125–135. DOI: 10.1007/s10032-015-0237-z. URL: http://dx.doi.org/10.1007/s10032-015-0237-z.

[Lev66] V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions and Reversals". In: *Soviet Physics Doklady* 10 (Feb. 1966), p. 707.

[LKB+09] Xiaonan Lu, Saurabh Kataria, William J. Brouwer, James Ze Wang, Prasenjit Mitra, and C. Lee Giles. "Automated analysis of images in documents for intelligent document search". In: *International Journal on Document Analysis and Recognition (IJDAR)* 12.2 (July 2009), pp. 65–81. DOI: 10.1007/s10032-009-0081-0. URL: http://dx.doi.org/10.1007/s10032-009-0081-0.

[Llo82] Stuart P. Lloyd. "Least squares quantization in PCM". In: *IEEE Trans. Information Theory* 28.2 (1982), pp. 129–136. DOI: 10.1109/TIT.1982.1056489.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft COCO: Common Objects in Context". In: *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*.

Ed. by David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars. Vol. 8693. Lecture Notes in Computer Science. Springer, 2014, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48. URL: https://doi.org/10.1007/978-3-319-10602-1_48.

[LS06] Xiaoqing Liu and Jagath Samarabandu. "Multiscale Edge-Based Text Extraction from Complex Images". In: *Proceedings of the 2006 IEEE International Conference on Multimedia and Expo, ICME 2006, July 9-12 2006, Toronto, Ontario, Canada*. IEEE Computer Society, 2006, pp. 1721–1724. DOI: 10.1109/ICME.2006.262882. URL: https://doi.org/10.1109/ICME.2006.262882.

[LS08] Zongyi Liu and Sudeep Sarkar. "Robust outdoor text detection using text intensity and shape features". In: *19th International Conference on Pattern Recognition (ICPR 2008), December 8-11, 2008, Tampa, Florida, USA*. IEEE Computer Society, 2008, pp. 1–4. DOI: 10.1109/ICPR.2008.4761432. URL: https://doi.org/10.1109/ICPR.2008.4761432.

[LSC+13] Zhuo Li, Matthew Stagitis, Sandra Carberry, and Kathleen F. McCoy. "Towards retrieving relevant information graphics". In: *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013*. Ed. by Gareth J. F. Jones, Paraic Sheridan, Diane Kelly, Maarten de Rijke, and Tetsuya Sakai. ACM, 2013, pp. 789–792. DOI: 10.1145/2484028.2484164. URL: https://doi.org/10.1145/2484028.2484164.

[LWH18] Po-Shen Lee, Jevin D. West, and Bill Howe. "Viziometrics: Analyzing Visual Information in the Scientific Literature". In: *IEEE Trans. Big Data* 4.1 (2018), pp. 117–129. DOI: 10.1109/TBDATA.2017.2689038. URL: https://doi.org/10.1109/TBDATA.2017.2689038.

[MBT+18] Spandan Madan, Zoya Bylinskii, Matthew Tancik, Adrià Recasens, Kimberli Zhong, Sami Alsheikh, Hanspeter Pfister, Aude Oliva, and Frédo Durand. "Synthetically Trained Icon Proposals for Parsing and Summarizing Infographics". In: *CoRR* abs/1807.10441 (2018). arXiv: 1807.10441. URL: http://arxiv.org/abs/1807.10441.

[MTE19] David Morris, Peichen Tang, and Ralph Ewerth. "A Neural Approach for Text Extraction from Scholarly Figures". In: *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*. IEEE, 2019, pp. 1438–1443. DOI: 10.1109/ICDAR.2019.00231. URL: https://doi.org/10.1109/ICDAR.2019.00231.

[MV11] Ales Mishchenko and Natalia Vassilieva. "Chart image understanding and numerical data extraction". In: *Sixth IEEE International Conference on Digital Information Management, ICDIM 2011, Melbourne, Australia, September 26-28, 2011*. IEEE, 2011, pp. 115–120. DOI: 10.1109/ICDIM.2011.6093320. URL: https://doi.org/10.1109/ICDIM.2011.6093320.

[NAA+10] Shahrul Azman Noah, Datul Aida Ali, Arifah Che Alhadi, and Junaidah Mohamad Kassim. "Going Beyond the Surrounding Text to Semantically Annotate and Search Digital Images". In: *Intelligent Information and Database Systems, Second International Conference, ACIIDS, Hue City, Vietnam, March 24-26, 2010. Proceedings, Part I*. Ed. by Ngoc Thanh Nguyen, Manh Thanh Le, and Jerzy Swiatek. Vol. 5990. Lecture Notes in Computer Science. Springer, 2010, pp. 169–179. DOI: 10.1007/978-3-642-12145-6_18. URL: https://doi.org/10.1007/978-3-642-12145-6_18.

[Naj04] Laurent Najman. "Using mathematical morphology for document skew estimation". In: *Document Recognition and Retrieval XI, San Jose, California, USA, January 18-22, 2004, Proceedings*. Ed. by Elisa H. Barney Smith, Jianying Hu, and James Allan. Vol. 5296. SPIE Proceedings. SPIE, 2004, pp. 182–191. DOI: 10.1117/12.526615. URL: https://doi.org/10.1117/12.526615.

[Nib86] W. Niblack. *An Introduction to Digital Image Processing*. Prentice-Hall, 1986. ISBN: 9780134806747.

[Ots79] N. Otsu. "A Threshold Selection Method from Gray-Level Histograms". In: *Systems, Man and Cybernetics, IEEE Transactions on* 9.1 (Jan. 1979), pp. 62–66. ISSN: 0018-9472. DOI: 10.1109/TSMC.1979.4310076.

[PH17] Jorge Poco and Jeffrey Heer. "Reverse-Engineering Visualizations: Recovering Visual Encodings from Chart Images". In: *Comput. Graph. Forum* 36.3 (2017), pp. 353–363. DOI: 10.1111/cgf.13193. URL: https://doi.org/10.1111/cgf.13193.

[Red16] Joseph Redmon. *Darknet: Open Source Neural Networks in C*. http://pjreddie.com/darknet/. 2013–2016.

[RHG+15] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett. 2015, pp. 91–99. URL: http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.

[SAN16] Russell Stewart, Mykhaylo Andriluka, and Andrew Y. Ng. "End-to-End People Detection in Crowded Scenes". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 2325–2333. DOI: 10.1109/CVPR.2016.255. URL: https://doi.org/10.1109/CVPR.2016.255.

[SH19] Mohammad Reza Sarshogh and Keegan E. Hines. "A Multi-task Network for Localization and Recognition of Text in Images". In: *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*. IEEE, 2019, pp. 494–501. DOI: 10.1109/ICDAR.2019.00085. URL: https://doi.org/10.1109/ICDAR.2019.00085.

[SKC+11] Manolis Savva, Nicholas Kong, Arti Chhajta, Fei-Fei Li, 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, Santa Barbara, CA, USA, October 16-19, 2011*. Ed. by Jeffrey S. Pierce, Maneesh Agrawala, and Scott R. Klemmer. ACM, 2011, pp. 393–402. DOI: 10.1145/2047196.2047247. URL: http://doi.acm.org/10.1145/2047196.2047247.

[SLP+18] Noah Siegel, Nicholas Lourie, Russell Power, and Waleed Ammar. "Extracting Scientific Figures with Distantly Supervised Neural Networks". In: *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries, JCDL 2018, Fort Worth, TX, USA, June 03-07, 2018*. Ed. by Jiangping Chen, Marcos André Gonçalves, Jeff M. Allen, Edward A. Fox, Min-Yen Kan, and Vivien Petras. ACM, 2018, pp. 223–232. DOI: 10.1145/3197026.3197040. URL: http://doi.acm.org/10.1145/3197026.3197040.

[Smi07] R. Smith. "An Overview of the Tesseract OCR Engine". In: *9th International Conference on Document Analysis and Recognition (ICDAR 2007), 23-26 September, Curitiba, Paraná, Brazil*. IEEE Computer Society, 2007, pp. 629–633. DOI: 10.1109/ICDAR.2007.56. URL: http://doi.ieeecomputersociety.org/10.1109/ICDAR.2007.56.

[Smi78] Alvy Ray Smith. "Color gamut transform pairs". In: *Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1978, Atlanta, GA, USA, August 23-25, 1978*. Ed. by S. H. Chasen and Richard L. Phillips. ACM, 1978, pp. 12–19. DOI: 10.1145/800248.807361. URL: https://doi.org/10.1145/800248.807361.

[Smi95] Ray Smith. "A simple and efficient skew detection algorithm via text row accumulation". In: *Third International Conference on Document Analysis and Recognition, ICDAR 1995, August 14 - 15, 1995, Montreal, Canada. Volume II*. IEEE Computer Society, 1995, pp. 1145–1148. DOI: 10.1109/ICDAR.1995.602124. URL: https://doi.org/10.1109/ICDAR.1995.602124.

[Sob14] Irwin Sobel. "An Isotropic 3x3 Image Gradient Operator". In: *Presentation at Stanford A.I. Project 1968* (Feb. 2014).

[SRH+16] Arvind Satyanarayan, Ryan Russell, Jane Hoffswell, and Jeffrey Heer. "Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization". In: *IEEE Trans. Vis. Comput. Graph.* 22.1 (2016), pp. 659–668. DOI: 10.1109/TVCG.2015.2467091. URL: https://doi.org/10.1109/TVCG.2015.2467091.

[SRS+03] Christian M. Strohmaier, Christoph Ringlstetter, Klaus U. Schulz, and Stoyan Mihov. "Lexical Postcorrection of OCR-Results: The Web as a Dynamic Secondary Dictionary?" In: *7th International Conference on Document Analysis and Recognition (ICDAR 2003), 2-Volume Set, 3-6 August 2003, Edinburgh, Scotland, UK*. IEEE Computer Society, 2003, pp. 1133–1137. DOI: 10.1109/ICDAR.2003.1227833. URL: http://dx.doi.org/10.1109/ICDAR.2003.1227833.

[SSH+97] Jaakko J. Sauvola, Tapio Seppänen, Sami Haapakoski, and Matti Pietikäinen. "Adaptive Document Binarization". In: *4th International Conference Document Analysis and Recognition (ICDAR '97), 2-Volume Set, August 18-20, 1997, Ulm, Germany, Proceedings*. IEEE Computer Society, 1997, pp. 147–152. DOI: 10.1109/ICDAR.1997.619831. URL: https://doi.org/10.1109/ICDAR.1997.619831.

[ST88] Hanan Samet and Markku Tamminen. "Efficient Component Labeling of Images of Arbitrary Dimension Represented by Linear Bintrees". In: *IEEE TPAMI* 10.4 (1988), pp. 579–586. ISSN: 0162-8828. DOI: 10.1109/34.3918. URL: http://dx.doi.org/10.1109/34.3918.

[SZ13] Jerzy Sas and Andrzej Zolnierek. "Three-Stage Method of Text Region Extraction from Diagram Raster Images". In: *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, Milkow, Poland, 27-29 May 2013*. Ed. by Robert Burduk, Konrad Jackowski, Marek Kurzynski, Michal Wozniak, and Andrzej Zolnierek. Vol. 226. Advances in Intelligent Systems and Computing. Springer, 2013, pp. 527–538. DOI: 10.1007/978-3-319-00969-8_52. URL: http://dx.doi.org/10.1007/978-3-319-00969-8_52.

[Tak09] Noboru Takagi. "Mathematical Figure Recognition for Automating Production of Tactile Graphics". In: *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11-14 October 2009*. IEEE, 2009, pp. 4651–4656. DOI: 10.1109/ICSMC.2009.5346749. URL: https://doi.org/10.1109/ICSMC.2009.5346749.

[VMN+16] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge J. Belongie. "COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images". In: *CoRR* abs/1601.07140 (2016). arXiv: 1601.07140. URL: http://arxiv.org/abs/1601.07140.

[WCE+10] Peng Wu, Sandra Carberry, Stephanie Elzer, and Daniel Chester. "Recognizing the Intended Message of Line Graphs". In: *Diagrammatic Representation and Inference, 6th International Conference, Diagrams 2010, Portland, OR, USA, August 9-11, 2010. Proceedings*. Ed. by Ashok K. Goel, Mateja Jamnik, and N. Hari Narayanan. Vol. 6170. Lecture Notes in Computer Science. Springer, 2010, pp. 220–234. DOI: 10.1007/978-3-642-14600-8_21. URL: https://doi.org/10.1007/978-3-642-14600-8_21.

[XGD+17] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. "Aggregated residual transformations for deep neural networks". In: *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE. 2017, pp. 5987–5995.

[XK10] S. Xu and M. Krauthammer. "A New Pivoting and Iterative Text Detection Algorithm for Biomedical Images". In: *Journal of Biomedical Informatics* 43 (2010), pp. 924–931.

[XMK08] Songhua Xu, James P. McCusker, and Michael Krauthammer. "Yale Image Finder (YIF): a new search engine for retrieving biomedical images". In: *Bioinformatics* 24.17 (2008), pp. 1968–1970. DOI: 10.1093/bioinformatics/btn340. URL: https://doi.org/10.1093/bioinformatics/btn340.

[YAD+14] Daekeun You, Sameer K. Antani, Dina Demner-Fushman, and George R. Thoma. "Does Figure-Text Improve Biomedical Article Retrieval? A Pilot Study". In: *2014 IEEE 27th International Symposium on Computer-Based Medical Systems, New York, NY, USA, May 27-29, 2014*. IEEE Computer Society, 2014, pp. 471–472. DOI: 10.1109/CBMS.2014.95. URL: https://doi.org/10.1109/CBMS.2014.95.

[YHT06] Li Yang, Weihua Huang, and Chew Lim Tan. "Semi-automatic Ground Truth Generation for Chart Image Recognition". In: *Document Analysis Systems VII, 7th International Workshop, DAS 2006, Nelson, New Zealand, February 13-15, 2006, Proceedings*. Ed. by Horst Bunke and A. Lawrence Spitz. Vol. 3872. Lecture Notes in Computer Science. Springer, 2006, pp. 324–335. DOI: 10.1007/11669487_29. URL: http://dx.doi.org/10.1007/11669487_29.

[YT11] Chucai Yi and Yingli Tian. "Text String Detection From Natural Scenes by Structure-Based Partition and Grouping". In: *IEEE Trans. Image Processing* 20.9 (2011), pp. 2594–2605. DOI: 10.1109/TIP.2011.2126586. URL: https://doi.org/10.1109/TIP.2011.2126586.

[YYP+15] Xu-Cheng Yin, Chun Yang, Wei-Yi Pei, Haixia Man, Jun Zhang, Erik Learned-Miller, and Hong Yu. "DeTEXT: A Database for Evaluating Text Extraction from Biomedical Literature Figures". In: *PLOS ONE* 10.5 (May 2015), pp. 1–19. DOI: 10.1371/journal.pone.0126200. URL: https://doi.org/10.1371/journal.pone.0126200.

[YYY+17] Chun Yang, Xu-Cheng Yin, Hong Yu, Dimosthenis Karatzas, and Yu Cao. "ICDAR2017 Robust Reading Challenge on Text Extraction from Biomedical Literature Figures (DeTEXT)". In: *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017*. IEEE, 2017, pp. 1444–1447. DOI: 10.1109/ICDAR.2017.235. URL: https://doi.org/10.1109/ICDAR.2017.235.

[ZYW+17] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. "EAST: An Efficient and Accurate Scene Text Detector". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 2642–2651. DOI: 10.1109/CVPR.2017.283. URL: https://doi.org/10.1109/CVPR.2017.283.

[ZZC+21] Fangfang Zhou, Yong Zhao, Wenjiang Chen, Yijing Tan, Yaqi Xu, Yi Chen, Chao Liu, and Ying Zhao. "Reverse-engineering bar charts using neural networks". In: *J. Vis.* 24.2 (2021), pp. 419–435. DOI: 10.1007/s12650-020-00702-6. URL: https://doi.org/10.1007/s12650-020-00702-6.

[ZZL+18] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018, pp. 6848–6856. URL: http://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_ShuffleNet_An_Extremely_CVPR_2018_paper.html.