Design and Implementation of a fault tolerant form processing application using machine learning

Master's Thesis in Computer Science

submitted
by

Christoph Neubauer

born 23.06.1991 in Homburg (Saar)

Written at

Lehrstuhl für Mustererkennung (Informatik 5)
Department Informatik
Friedrich-Alexander-Universität Erlangen-Nürnberg.

in Cooperation with

Universidade Federal do Parana
Curitiba

Advisor: PD Dr.-Ing. habil. Peter Wilke

Second Advisor: Prof. Luiz Eduardo S. Oliveira

Started: 12.09.2016

Finished: 13.03.2017

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien des Lehrstuhls für Studien- und Diplomarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Erlangen, den 31. Januar 2017

Übersicht

Abstract

Contents

Chapter 1

Introduction

Some general information on the context and setting.

1.1 Motivation

Specific motivation for the problem at hand.

1.2 Task

The task of this master thesis is to evaluate different electronic invoice formats. An application shall be designed and implemented that processes invoice forms and is capable of storing them in the invoice format that suits the most. The application should use machine learning in order to improve processing accuracy over time. Nevertheless, errors during the scan process should be handled by the application itself. The output of this application should be conform with the definition of the electronic invoice format that has been decided in beforehand.

1.3 Related Work

Other relevant academic work and how it differs from this work, for example “textual” citation, as shown in “parenthesis” citation

1.4 Results

What has been achieved in this work?

1.5 Outline

How is the thesis structured and why?

1.6 Acknowledgments

A big thank you for the support to ...

Chapter 2

Electronic invoice formats - A comparison

During the technologization of companies over the world, electronic invoices (also known as e-invoice) have become more and more important.

2.1 Description of leading formats

2.2 Definition of decision criteria

2.3 Comparison and decision finding

Chapter 3

OCR - State of the art, possibilities and drawbacks

During the processing of a form, the image of it is not only scanned. To retrieve additional information and work it we will use Optical Character Recognition (OCR).

The process of retrieving data, for instance in form of characters or numbers, requires several steps. In the beginning, the image to be processed is converted to a gray-scale image. Then preprocessing takes place. In this process several algorithms such as noise-reduction and the canny-edge-algorithm are applied on the image to reduce irritating and / or unnecessary information in the image and to enhance contrast.

After that, features are extracted. Those features are single characters whose vectors are defined afterwards. With the information about the feature vector it is possible to classify each character (and to detect which character of the alphabet is most likely to be represented by the feature). When all characters have been classified, post-processing takes place. Here possible failures can be corrected for example by comparing words with a predefined dictionary. These steps are also shown in figure 3.1:

The whole process of OCR is part of research since decades and several papers, dissertations and books have been published on the matter. Developing our own OCR algorithm would not only exceed the size of this master thesis, but also most likely retrieve less successful results than already developed and improved algorithms. Instead, this chapter will introduce several available OCR algorithms and compare open source solutions to find the best fit for our need.

3.1 Currently available OCR algorithms

OCR has been of interest for companies over many years. Hence we expect several possible solutions we could choose. As the amount of solutions can easily be very high we want to reduce the presented solutions to a maximum amount of 10.

Each description of a solution will contain information about the license used, the supported operating systems, programming languages used as well as if there is a software development kit. Also general information about the company will be given, the currently released version of the solution and the release date and, if existent, the number of languages supported.

3.1.1 ABBYY Fine Reader

ABBYY Fine Reader is a proprietary solution from the identically named company ABBYY founded in 1989. It is usable for all three operating systems, whereby linux-distributions are only supported as a command-line-interface. ABBYY supports a SDK for all three operating systems. Although it is written in C/C++ there exists a wrapper for java development for Linux and Windows(TODO: Link fÃ¼r Abbyy broschÃ¼re version 11).

The SDK is currently in Version 11, it supports 185 languages, the latest update was on 03.10.2016.

3.1.2 Anyline SDK

Anyline is an Austrian company founded in 2013 who aim at OCR solutions for mobile systems. They offer their SDK as a free license for non-commercial use. However, as they are focused on mobile systems, they do not explicitly support Windows-Desktop, Linux or MacOS.

It can be developed with Java and Objective-C as well as Swift, C# and Javascript. The current version of the SDK is 3.8.1 and has been released on 13.01.2017.

Currently supported are two languages: English and German.

3.1.3 Asprise OCR

Asprise has been founded in 1998 in Singapure. The company offers OCR SDKs in various programming languages (Java, C#, VB.NET, Python, C/C++ and Delphi Pascal). The SDKs are under loyalty-free license and therefore proprietary. More than 20 languages are supported, English and German are included. The SDK supports Windows, MacOS and Linux, whereby support of multiple operating systems at once increases the price.

3.1.4 GOCR

GOCR is an OCR application started by Joerg Schulenburg in 2000. The program is developed under the GNU Public License.

The latest version is 0.5 and has been released in March 2013. It is working under Windows, Linux and MacOS. The code is written in C, but is not known if there is a SDK which enables usage of the application inside another application. The number of supported languages is also unkown.

3.1.5 LEADTOOLS

Leadtools is an American company founded in 1990 and offers various products in the range of document and image processing.

The Leadtools OCR Engine can be used as an SDK and integrated in another application. Development with the SDK is possible with C# and VB as well as C/C++ and Java (and some others). The engine supports more than 40 languages, containing German and can be used on Windows, Linux and MacOS.

The current version of the SDK is 19 and has been released in December 2014.

3.1.6 MathOCR

MathOCR is a document recognition system written in Java with focus on formulas. The MathOCR project started in March 2014 and is based on the GNU General Public Liense.

The current version of the application is 0.0.3, which was released in May 2015 and is therefore still in a pre-alpha status. It can be used on Windows, Linux and MacOS. The amount of supported languages is not stated on the project page.

3.1.7 OCROpus

The OCRopus Open Source OCR System is an open source system developed and maintained by the German Research Laboratory for Artificial Intelligence under guidance by Thomas M. Breuel. It is licensed under the Apache 2.0 License.

The command-line application is written in Python and C++ and only supports Linux as Operating System. It currently uses the Tesseract as a text line recognizer but will be replace it in the future.

The current stable version is 1.0 and has been released in November 2014. The amount of supported languages is unkown, whereby it is able to work with latin-based languages.

3.1.8 OCRad

OCRad is a free OCR application under the GNU Public License and part of the GNU project. Antonio Diaz Diaz developed the application since 2003.

The current version is 0.25 and has been released in April 2015. It comes as a stand-alone console application but can also be used in the background by other applications.

3.1.9 OmniPage Capture SDK

The Nuance Communication Inc. offers an OCR tool called OmniPage Capture SDK, which enables document processing on Windows, Linux and MacOS. Depending on the underlying operating system it supports C/C++, Objective-C or C# and VB.NET.

The current version is 20 and has been released in 2016. It supports over 120 languages (German included).

3.1.10 Tesseract

The Tesseract OCR Engine historically was an early project developed by Hewlett Packard between 1984 and 1994. In 2005, it was put on an open source license. It is currently maintained by Google under the Apache 2.0 license.

Tesseract is originally written in C/C++ and can be used on Windows, MacOS and Linux. There also exists a wrapper which allows development in Java, the open source project Tess4J.

The current stable version of the Tesseract is 3.04.01 and has been released in February 2016. It supports over 100 languages (including German).

3.2 Comparison between open source algorithms

While several companies exist that offer good OCR libraries and SDKs, we have to stick to free software, since we are not able to afford a proprietary license. In a later state of the application, it could be possible to switch to a proprietary solution in order to increase our OCR efficiency. Until then, we will decide for the best fitting open source algorithm library and improve our efficiency by preprocessing the forms ourselves. This is explained in Chapter 4, Module 1.

Upon the 10 presented solutions, only 5 are free for use. These are: GOCR, Tesseract, OCROpus, MathOCR and OCRad.

The following table shows the named solutions and shows their differences regarding their version, the latest release date, the supported programming languages and operating systems as well as the license they are put on:

MathOCR is a relatively young application and therefore in a pre-alpha state. OCRad and GOCR are one step closer to the first major release. OCROpus has reached that state on November 2014. Tesseract is already on Version 3.04 and has recently released an alpha version of 4.0.

While Tesseract, OCROpus and OCRad are supporting C++, GOCR is only working with C whereas MathOCR is only Java. Tesseract is supporting C and Java as well (while using Tess4J as a wrapper). OCROpus instead is working with Python, too.

All solutions support Windows, Linux and MacOS except OCROpus, which is only working on Linux.

While GOCR, MathOCR and OCRad are licensed under the GNU Public License, Tesseract and OCROpus are licensed under the Apache 2.0 License. The difference between those two is mainly the following: Applications that are developed under usage of another program under the GNU Public License have to be licensed under the GNU Public License as well. The Apache 2.0 License allows usage of other application and enables free choice of licensing, but requires the mentioning of the underlying use of an Apache 2.0 licensed application.

3.3 Decision finding and explanation

In the beginning of this thesis, it was defined that the application should work on a Linux based operating system and be written in Java. While all presented solutions support Linux as an Operating System, not all of them work with Java as a programming language. In addition to that, OCROpus only supports Linux, which is fine for the current focus of the application, but could be of an issue later on if it should be ported to another operating system.

MathOCR is in a pre-alpha state which makes it difficult to use due to several missing functionalities and persistent bugs in the code. As it is mostly focused on mathematical equations and formulas, we will not consider MathOCR any longer, even though it supports Java as a programming language.

As explained in section 3.2, the GNU Public License requires our application to be licensed under the GNU Public License as well if we use another code which is licensed under this license. Therefore, the Apache 2.0 license is considered better, as it allows us to decide about the license

for ourselves.

In the interest of the application, we want to use solutions that are up-to-date and are still under development. Therefore, the latest release date gives us insights about the activity on a project. Since a lot of open source projects suffer from missing developers, we expect longer development cycles. But the last version of GOCR has been released around 4 years ago. Hence we consider GOCR as not up-to-date anymore.

The following table shows the solutions again, but with underlying colors regarding their ability to fit to our problem. Green is used as a best fit, whereas red significates a major problem. Yellow shows that this attribute is not as good as others but no kick-out criterion.

As shown in the table, MathOCR will not fit our needs as it is a pre-alpha version. GOCR is outdated and also supports only C as a programming language. OCROpus and OCRad only support C++ (and in the case of OCROpus Python). In addition to that, OCROpus could be of a problem when porting the application to other operating systems whereas OCRad is licensed under the GNU Public License.

Hence the Tesseract seems to be the best fit for our application. It is consistently updated and improved and by the history of it, the application itself has grown mature. The possibility to work with Tess4J enables the usage of it with Java. Multiple operating systems are supported and the Apache 2.0 license enables us to decide for ourselves under which license we will put the application.

Chapter 4

Implementation

The application is structured by different packages. Each of them providing a specific benefit to the program as a whole. Before speaking about those modules, we will talk about the actual requirements of the application. After that, each module will be explained in detail and how it works. After that, the last section will deal with problems and possible solutions.

4.1 Requirements definition

Focus of this application is the possibility to automatically process forms and retrieve the data of the forms. Hence the application should be able to deal with several files and process them without human help. But, since there is a big variety of forms and every company have different structures, retrieving all necessary information can fail. If this happens, a user has to review scanned documents that contain errors. If it doesn't fail, data should be stored without any additional help.

To improve the process of gathering data, a machine learning approach should be implement that facilitates retrieving data and to speed-up form processing over time.

Output of the application should be a storage of the processed forms, appended with electronic invoice information that is valid against the basic- or comfort-level of the ZugfErd-Invoice standard.

4.2 Definition of modules

4.3 Architectural concept

The application will make use of several design patterns. One used pattern on an architectural level is the MVC-pattern. Hence the graphical user interface will be steered by a controller which retrieves data from the database and shows it to the user using javaFX and .fxml-Files. Input and changes the user makes in the view will be transported by the controller to the model which is stored in the database again.

To access the database we will use classes for each business object. The package BO contains classes that represent a table. A data-access-object (DAO) will be used to retrieve data from the database. To do that, this application will also make use of an object-relational mapping framework (Hibernate) which facilitates the conversation between table data and java objects.

4.4 Module 1 - OCR

4.5 Module 2 - ANN

4.6 Module 3 - Converter

4.7 Problems during the implementation

Chapter 5

Conclusion

5.1 Result of the application

5.2 Findings

5.3 Outstanding issues

Chapter 6

InvoiceFormats

During the technologization of companies over the world, electronic invoices (also known as e-invoice) have become more and more important. E-Invoicing offers companies the possibility to improve their business processes, making invoicing faster and more efficient and enables a direct connection to other tools like ERP-Software.

To enable companies these benefits and in order to make the communication between companies even possible a comprehensive standard has to be defined. With an invoice standard at hand, companies can use invoices from their business partners and read them into their systems (in case of B2B).

There are several invoice standards in action at the moment. The next section will deal with the most important ones and describes them as well as pointing out the benefits and drawbacks of the format. After that, the next section defines criteria that are relevant for the application and how to measure them. In the last section, these criteria are applied on the formats defined in section **??** and compared against each other. Eventually a decision regarding the usage of one of these formats is made.

6.1 Description of leading formats

6.1.1 UN/EDIFACT

EDIFACT is a well-established [**?**] subset of standards from CEFACT regarding the electronic interchange of structured data. It is developed and maintained by the United Nations Economic Comission for Europe (UNECE) [**?**].

The European Commission states that the UN/EDIFACT INVOIC message has been a corner-

stone in electronic invoicing over the past years [**?**, **?**].

There are several subsets of EDIFACT that have been developed for different industries. For instance, the chemical industry uses CEFIC/ESCom[1] as their standard, while automotive industry is in charge with ODETTE/FTP2[2].

EDIFACT has different message types such as ORDCHG for a request to change an order or PAYORD which contains a payment order. In the context of this thesis, the message type INVOIC (containing an invoice) is the most interesting one.

6.1.2 XCBL

The XML Common Business Library is an extension of the CBL which originally has been devloped by Veo Systems Inc. [**?**]. The company has been bought by Commerce One Inc. in 1999 [**?**], page 29.

xCBL currently exists in version 4.0 (since 2003)[3]. Since the company has gone bancrupt in 2004 **??** it is not very likely that this format gains more interest in the future.

6.1.3 OASIS/UBL

UBL stands for Universal Business Language and is being developed by OASIS. The current version is 2.1 and is normed by the international standardization organization[4].

Several countries developed their own subset of this format. Especially interesting in this case is a project called PEPPOL (Pan-European Public Procurement Online project) that aims at developing a format for public sectors in the whole European Union[5].

Also interesting in the context of invoice interchange is the UBL-based project called *simplerinvoicing* that aims at connecting ERP systems with accounting and e-invoicing software by providing an own invoicing standard[6].

6.1.4 ZugFerd

This invoice format has been published initially in 2014 [**?**]. The name is a german acronym, containing the name of the corresponding forum (FeRD). It can be translated to ”Central User

[1] see also: https://www.cefic.org/Industry-support/Implementing-reach/escom/
[2] see also: https://www.odette.org/services/oftp2
[3] see also: https://www.xcbl.org
[4] see ISO/IEC 19845:2015
[5] see also: https://www.peppol.eu/about_peppol/about-openpeppol-1
[6] see also: www.simplerinvoicing.org/en/

Guide of the Forum for electronic Invoicing in Germany". Although this invoice format is rather young it tries to fulfill the directive 2014/55/EU of the european parliament [**?**] while still being flexible and simple. This directive states that the use of electronic invoice formats should be adopted by all member states of the european union until the 27. of November 2018 [**?**, **?**].

The approach of the ZugFerd-format enables not only big companies to work with that format, but also smaller and medium companies (SME's) that are in need of such a format but are normally not able to implement a complex electronic invoice standard. Furthermore three levels of conformance are defined: Basic, Comfort and Extended. Each of those levels have a different amount of required information fields, that have to be set in order to be a valid ZugFerd-format. Nevertheless, in all of the three formats, it is possible to define more information in free text fields.

This enables extensibility of the format and the possible business areas in which this standard can be used.

The German Forum for electronic invoice (FeRD) states that this format has been accepted as a core standard in Germany to be used in the future such that every company, that wants to start business relations with a german company, has to use this standard **??**. Furthermore, the possibility to extend this standard to all European Countries is in sight, as stated in **??**.

6.2 Definition of decision criteria

While the standards defined in the section before focus on specific areas or try to combine multiple fields, this section defines the criteria that are most relevant for the application that is developed.

6.2.1 Future Potential

One of the major criteria for a suitable invoice standard should be its future potential. Developing an application that deals with a standard that is not being used 10 years later does not make sense. Therefore, any standard that is going to be replaced should not be considered useful.

6.2.2 Relevance in Germany (and Europe)

As this thesis is being written at a German university, the chosen standard should be relevant in Germany. Even better if it is relevant in Europe as well. On the other Hand, standards that are not of interest for Europe should be excluded.

6.2.3 Extendability to more countries

The possibilities of a standard to be used in other countries will also affect its importance over the next decades. Standards that only suits the requirements of one country are not important enough. The focus lies on standards with a wide (possible) range of countries to be affected, instead.

6.2.4 Extendability of the standard itself

Last but not least, the extendability of the standard itself is an important criterion. The world is changing and new requirements are coming while older ones are getting broken up. A valuable standard should be able to deal with these changes and should be extensible towards new requirements, or special requirements in specific business areas.

6.2.5 Complexity

The complexity of the standard is important for this thesis too. Not only is the development of the application limited by time, but also makes a complex standard it hard to understand it and less error-prone.

6.3 Comparison and decision finding

6.3.1 Application of the criteria

6.3.2 Decision and explanation

Appendix A

Glossary

B2B - Business to Business

- ERP - Enterprise Resource Planning
- EDIFACT - Electronic Data Interchange For Administration, Commerce and Transport
- FeRD - Forum für elektronische Rechnung Deutschland
- SME - Small and medium enterprises
- UBL - Universal Business Language
- xCBL - XML Common Business Library
- ZugFerd - Zentraler User Guide des Forums für elektronische Rechnung Deutschland

List of Figures

List of Tables

Bibliography

[Bre02] Thomas M. Breuel. *Two Geometric Algorithms for Layout Analysis*, pages 188–199. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[Bre03] Thomas M. Breuel. High performance document layout analysis, 2003.

[Bre07] Thomas M. Breuel. The hocr microformat for ocr workflow and results. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 1063–1067, Sept 2007.

[Bre08] Thomas M. Breuel. The ocropus open source ocr system, 2008.

[Che08] Qiang Chen, Quan-sen Sun, Pheng Ann Heng, and De-shen Xia. A double-threshold image binarization method based on edge detector. *Pattern Recogn.*, 41(4):1254–1267, April 2008.

[Che13] Shuhan Chen, Weiren Shi, and Wenjie Zhang. An efficient universal noise removal algorithm combining spatial gradient and impulse statistic. *Mathematical Problems in Engineering*, 2013:1–12, 2013.

[CO00] Inc. Commerce One. Annual report of 2000, 2000. https://www.media.corporate-ir.net/media_files/NSD/CMRC/reports/10_k.pdf, last visited on 09.11.2016.

[Cov01] Robin Cover. Xml common business library (xcbl), 2001. https://www.xml.coverpages.org/cbl.html, last visited on 09.11.2016.

[Den14] Andreas Dengel and Faisal Shafait. *Analysis of the Logical Layout of Documents*, pages 177–222. Springer London, London, 2014.

[fE16] UN Economic Commission for Europe. Introducing un/edifact, 2016. https://www.unece.org/cefact/edifact/welcome.html, last visited on 08.11.2016.

[Ham07] Hatem Hamza, Yolande Belaïd, and Abdel Belaïd. Case-based reasoning for invoice analysis and recognition. In Rosina O. Weber and Michael M. Richter, editors, *7th International Conference on Case-based Reasoning - ICCBR 2007*, volume 4626, pages 404–418, Belfast, United Kingdom, August 2007. Springer Berlin / Heidelberg. The original publication is available at www.springerlink.com , ISBN 978-3-540-74138-1, ISSN 0302-9743 (Print) 1611-3349 (Online).

[Kau15] Achim Kauffmann. 5 punkte, die sie über zugferd wissen sollten, 2015. https://www.basware.de/blog/2015-07-10/ZUGFeRD-5-punkte-die-sie-wissen-sollten, last visited on 09.11.2016.

[Kle04] Bertin Klein, Stevan Agne, and Andreas Dengel. *Results of a Study on Invoice-Reading Systems in Germany*, pages 451–462. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[Men04] Kurt Menges. Commerce one declares bankruptcy: Does this foretell the fate of b2b e-commerce?, 2004. https://www.supplychainmarket.com/doc/commerce-one-declares-bankruptcy-does-this-fo-0001, last visited on 09.11.2016.

[Nag95] George Nagy. *Document image analysis: What is missing?*, pages 576–587. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

[Ram12] Cartic Ramakrishnan, Abhishek Patnia, Eduard Hovy, and Gully APC Burns. Layout-aware text extraction from full-text pdf of scientific articles. *Source Code for Biology and Medicine*, 7(1):7, 2012.

[Wan15] Chenyang Wang, Yanhong Xie, Kai Wang, and Tao Li. *OCR with Adaptive Dictionary*, pages 611–620. Springer International Publishing, Cham, 2015.

[Zhu05] Li Zhuang and Xiaoyan Zhu. *An OCR Post-processing Approach Based on Multi-knowledge*, pages 346–352. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.