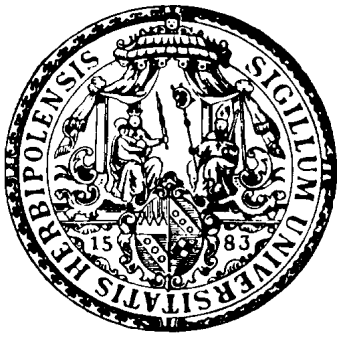
Evaluation von Open-Source OCR-Software auf deutscher Frakturschrift

Bachelorarbeit im Fach Informatik

vorgelegt von

Kevin Fuchs

11.09.2015



Julius-Maximilians-Universität Würzburg

Lehrstuhl für Informatik VI

Künstliche Intelligenz und Angewandte Informatik

betreut von

Prof. Dr. Frank Puppe
M.Sc. Markus Krug

Inhaltsverzeichnis

Vorwort v

1 Optical Character Recognition 1
- 1.1 Definition . . . 1
- 1.2 Schwierigkeiten der Zeichenerkennung . . . 1
- 1.3 Phasen des OCR-Prozesses . . . 2
- 1.4 Essentielle OCR-Techniken . . . 4
- 1.5 Evaluation . . . 5

2 Verwandte Arbeiten 9
- 2.1 State of the Art . . . 9
- 2.2 Marktübersicht für nichtkommerzielle OCR-Software . . . 13

3 Fraktur 17

4 Übersicht über ausgewählte OCR-Software 19
- 4.1 Tesseract . . . 19
- 4.2 Ocropy . . . 20
- 4.3 GOCR . . . 22
- 4.4 Asprise OCR . . . 23
- 4.5 Google Docs . . . 23

5 Evaluation 25
- 5.1 Erkennung von deutsch- und englischsprachiger Literatur . . . 25
  - 5.1.1 Ausgangsmaterial . . . 25
  - 5.1.2 Testszenario . . . 26
  - 5.1.3 Evaluation . . . 26
- 5.2 Erkennung von Frakturschrift . . . 29
  - 5.2.1 Ausgangsmaterial . . . 29
  - 5.2.2 Testszenario . . . 29
  - 5.2.3 Evaluation . . . 30
- 5.3 Erkennung der lateinischen Übersetzung eines deutschsprachigen Werkes aus dem 15.Jahrhundert . . . 32
  - 5.3.1 Ausgangsmaterial . . . 32
  - 5.3.2 Testszenario . . . 32
  - 5.3.3 Evaluation . . . 34

6 Fazit **37**

7 Anhang **39**

Vorwort

Diese Bachelorarbeit beschäftigt sich mit der Qualität der Erkennung von Texten deutscher Frakturschrift basierend auf Open-Source Softwarelösungen. Durch den Einsatz von „Optical Character Recognition“ (OCR) kann die Erkennung von Texten und Büchern automatisiert und so der Arbeits- und Kostenaufwand für die Digitalisierung minimiert werden. Eine automatisierte Digitalisierung ist essentiell, um historische Werke vor dem Zerfall bzw. dem Qualitätsverlust zu retten. Weiterhin bietet es eine schnelle und einfache Option der Verbreitung der jeweiligen Dokumente an, welche per Onlinedienste zur Verfügung gestellt werden können. Diese Bereitstellung realisiert interessierten Bürgern Zugang zu wissenschaftlichen und kulturellen Texten, welche mangels Auflagenzahl oder durch die Ortsabhängigkeit der Werke nicht erreichbar wären. Neben der Möglichkeit des Neudrucks und Eingliederung in Bibliotheken kann durch die Digitalisierung eine Konvertierung in verschiedene Formate wie z.B. XML, PDF oder EPUB verwirklicht werden. Dadurch wird die Informationsgewinnung innerhalb der Texte mittels Verwendung von zusätzlichen Features wie z.B. einer Suchfunktion wesentlich vereinfacht. Weiterhin können in der digitalen Version Kommentare bzw. Notizen eingefügt werden, ohne das Werk zu beschädigen. Ein weiterer Aspekt ist, dass sich altdeutsche Texte durch die Grammatik und Zeichenform stark von modernen Schriften differenzieren können, wodurch der ungeübte Leser den Inhalt schwerer erfassen kann. Durch die Digitalisierung kann der Text in die gewünschte Schriftart konvertiert und so das Lesen vereinfacht werden. [4]

Zuerst wird im Kapitel „Optical Character Regocnition“ der Begriff „OCR“ definiert und die Schwierigkeiten der Zeichenerkennung erläutert. Folgend wird der Ablauf des OCR-Prozesses und essentielle Techniken erörtert. Abschließend werden die einzelnen Fehlerarten der Zeichenerkennung erklärt, welche die Basis für die darauffolgende Evaluation darstellt. Im zweiten Kapitel „Verwandte Arbeiten„ wird der aktuelle Stand der OCR-Technik diskutiert und einen Überblick über nicht kommerzielle OCR-Software gegeben. Das Kapitel „Fraktur“ beschäftigt sich mit den Klassifikationsmerkmalen und der Einordnung der Frakturschrift. Anschließend werden im Kapitel „Übersicht über ausgewählte OCR-Software“ die verwendeten Programme der Testszenarien beschrieben. Im Kapitel „Evaluation“ wird die Genauigkeit des OCR-Prozesses in verschiedenen Testszenarien durch die Standard- bzw. mittels Training generierter Sprachmodelle ermittelt. Abgerundet wird die Arbeit im letzten Kapitel „Fazit“ durch eine Zusammenfassung der Ergebnisse und weitere Möglichkeiten zur Optimierung des Erkennungsprozesses.

1 Optical Character Recognition

Im folgenden Kapitel wird eine Defintion über den Begriff „OCR“ präsentiert und im Anschluss aktuelle Problemstellungen des Prozesses aufgeführt. Weiterhin gliedert sich der Ablauf des OCR-Prozesses in verschiedene Verarbeitungsschritte, welche näher erläutert werden. Um eine Evaluierung durchführen zu können, werden die einzelnen Fehlerarten bzw. Fehlerraten anhand von Beispielen erklärt.

1.1 Definition

Unter dem Begriff „Optical Character Recognition“ (OCR) wird der Vorgang beschrieben, bei dem ein gescanntes Bild, welches maschinen- bzw. handschriftliche Inhalte wie zum Beispiel Zeichen und Symbole enthält, in einen maschinenlesbaren Zeichenstream konvertiert wird. [16, S.3]

1.2 Schwierigkeiten der Zeichenerkennung

Die fehlerhafte Erkennung eines Zeichens kann aus der schlechten Qualität des Bildes bzw. den unzureichenden Erkennungsfähigkeiten des Klassifizierers resultieren. Einerseits existieren eine Vielzahl von weiteren Faktoren wie z.B. eine mangelhafte Qualität des Originaldokumentes, wodurch Verschmutzungen als Zeichen oder als ein Teilsegment erkannt werden können und so der OCR-Prozess negativ beeinflusst wird. Zusätzlich können unzureichende Vorverarbeitungsschritte oder eine schlechte Segmentierung des Bildes die Fehlerrate vervielfachen.

Auf der anderen Seite kann die Methode, welche für die Erkennung der Zeichen zuständig ist, das Resultat verschlechtern, da nur ein begrenztes Training oder das Lernen durch die beschränkten Fähigkeiten des Klassifizierers die Ursache sein können.

Zudem können ähnliche Zeichen wie der Buchstabe „ l “ und die Zahl „ 1 “ sehr schwierig differenziert werden. Durch die unterschiedliche Typografie des Textes durch Einsatz von Kursiv- oder Fettdruck, Schattierungen von Zeichen oder Unterstreichungen können die Unterschiede zwischen ähnlichen Objekten nicht mehr ersichtlich sein. Ebenfalls stellen verformte Grundlinien ein Problem dar, weil durch diese weitere Vorverarbeitungsschritte nötig werden oder eine falsche Segmentierung der einzelnen Zeichen erfolgen kann. [16][S.8 ff] Der Erkennungsprozess auf historischen Dokumenten, welche in dieser Arbeit näher untersucht

wird, gestaltet sich durch die eben erläuterten Problemstellungen schwieriger als bei modernen Dokumenten.

1.3 Phasen des OCR-Prozesses

Der typische OCR-Prozess besteht aus einer Pipeline aus mehreren Phasen und kann anhand der folgenden Auflistung und der Abbildung 7[20] nachvollzogen werden:

- **Digitalisierung**

  Mithilfe eines Scanners wird das vorhandene Dokument als Bild erfasst und abhängig von der weiteren Verarbeitung wird eine Auflösung von mindestens 200 dpi benötigt, da featurebasierende Verarbeitungsmethoden von der Skalierung des Grauwertes des Dokumentes profitieren können. Falls die Auflösung zu gering ist, können in der Phase der Merkmalsextraktion und Klassifikation einzelne Buchstaben nicht korrekt identifiziert werden, da essentielle Features nicht mehr vorhanden sein könnten. [16][S.3]

- **Binarisierung**

  Um den Rechenaufwand zu minimieren und die Komplexität für den Analysekomponente zu reduzieren , erfolgt eine Vorverarbeitung mittels Binarisierung, welcher das Bild in einen Vordergrund bzw. Hintergrund einteilt.

  Es existieren eine Vielzahl von Binarisierungsverfahren, wobei die meisten Algorithmen mit einem Threshold bzw. einem Grenzschwellwert arbeiten, welche auf einem Farb- bzw. Grauwert als Klassifizierungskriterium basieren. Unterschieden wird zwischen dem „global Threshold“, welcher den Grauwert anhand des ganzen Bildes festlegt und dem „local Threshold“, welcher für individuelle Pixelgruppen einen entsprechenden Grenzwert ermittelt. Die Berechnung eines globalen Grenzwertes für ein Dokument erfolgt schnell, jedoch kann durch verschiedene Farbverteilungen bzw. Farbverläufe dies zu mangelhaften Ergebnissen führen. Der Einsatz von lokalen Grenzwerten ist somit zuverlässiger, allerdings auch anfälliger für Verschmutzungen und rechenintensiver. Weiterhin existieren parameterlose Methoden, welche den lokalen Grenzwert basierend auf dem Mittelwert errechnen. Eine weitere Möglichkeit ist die Berechnung eines gemeinsamen Mittelwertes, dieser errechnet sich aus einem gewichtetem lokalem Grenzwert, einem globalem Grenzwert und dem globalen Otsu Grenzwert. [14] Zusätzliche Problemstellungen bei der Binarisierung können zu geringer Kontrast, Verschmutzungen, durchscheinende Zeichen von der Rückseite des Dokumentes und die elektronische Dokumentenkonvertierung sein, welche durch schlechte Kalibrierung von Geräten Fluktuationen von Licht in Bereichen der Seite

begünstigen. [15][S.2] Eine detaillierte Betrachtung des Binarisierungsprozesses kann im Paper „The image binarization problem revisited: perspectives and approaches“[14] bzw. „Methods of bitonal image conversion for modern and classic documents“[15] nachvollzogen werden.

- **Segmentierung**

  In dieser Phase erfolgt eine Zerlegung des Bildes in mehrere Zusammenhangskomponenten, welche verschiedene Objekte wie z.B.Textblöcke, Zeichen, Symbole oder Bilder beinhalten. Hierbei kann zwischen folgenden Ansätzen differenziert werden:

  – **Bottom Up** Gruppierung von Zusammenhangskomponenten, welche eine logische Einheit z.B. ein Wort darstellen.

  – **Top Down** Einsatz von Mustern, welche Standardlayouts erkennen oder rekursiv Zusammenhangskomponenten unterteilen. [31]

- **Merkmalsextraktion und Klassifikation**

  Die Klassifikation eines Zeichens beruht auf der Merkmalsextraktion der segmentierten Zeichen. Diese lässt sich durch geometrische und strukturelle Eigenschaften sowie von Transformationen im Merkmalsraumes unterscheiden. Beispiele für geometrische bzw. strukturelle Merkmale können aus Momenten, Histogrammen eines Bildsegmentes, der Kontur von Zeichen oder bestimmten Kreuzungs- bzw. Gabelungspunkte von Zeichen dargestellt sein. [31][S.11 ff] Transformationen werden benötigt um Variationen von Zeichen in Dokumenten zu eliminieren und diese richtig zu erkennen. Beispielsweise kann durch lineare Transformation, welche aus einer Kombination von Rotationen, Verschiebungen oder Größenänderungen besteht kann, das Zeichen angepasst werden. Weitere Methoden zur Merkmalsextraktion können im Paper „Character recognition systems: a guide for students and practitioners “[18]S.64 ff nachgelesen werden. Die Klassifikation wird mittels der gesammelten Daten und eines Klassifikators wie z.B. durch den Einsatz von Support Vektor Maschinen oder künstlichen neuronalen Netzen, realisiert. [31][S.11 ff]

- **Optionale Nachbearbeitung**

  Fehlerhafte Klassifikationen der OCR-Engine können durch eine Nachbereitung ausgebessert werden. Beispielhaft kann durch den Einsatz von Lexika die Fehlerrate weiter minimiert werden, da ein Wortabgleich mit dem Wörterbuch einzelne Klassifikationsfehler erkennt und diese anschließend von der Anwendung bereinigt werden können. [29][S.133 ff]

- **Ausgabe des Resultates**

  Nach Abschluss des OCR-Prozesses erfolgt die Kommunikation mit dem Benutzer mithilfe des Ausgabeinterfaces anhand der ermittelten Daten und stellt diesem die Ergebnisse der Texterkennung bereit. Abhängig von den Konfigurationsmöglichkeiten des Programmes kann das Ergebnis in verschiedenen Formaten wie z.B als Text- oder HTML-Datei ausgegeben werden. [16]

1.4 Essentielle OCR-Techniken

Obwohl mittlerweile sehr vielseitige Erscheinungen von Verarbeitungstechniken innerhalb verschiedenster OCR-Software verwendet werden, existieren bestimmte Grundkomponenten wie die Extraktion anhand von Features, als auch eine Komponente für die Klassifizierung der Zeichen, welche in jedem Algorithmus implementiert sind.

Anhand eines Bildelementes erkennt der Merkmalsextraktor bestimmte Eigenschaften eines Zeichens bzw. Elementes und leitet diese als Eingabe an den Klassifizierer weiter. Dieser bestimmt an einem meist bekannten Zeichensatz die Zugehörigkeit des Elementes basierend auf den ermittelten Features.

Die meist verbreitetste und gewöhnlichste Klassifizierungsmethode stellt das Template- bzw. Matrix Matching dar. Hierzu wird eine Musterschablone der einzelnen Zeichen und die dazugehörige A-Priori Wahrscheinlichkeit des Auftretens als Basis genommen. Der Merkmalsextraktor benutzt einzelne Pixel des Bildes als Merkmale, welche nun mit dem Prototyp jedes Zeichens verglichen wird. Bei jedem Vergleich wird anhand einer Distanzfunktion die Ähnlichkeit zwischen der Eingabe und dem Zeichen des Templates ermittelt. Der Wert der Ähnlichkeit vergrößert sich, wenn ein Pixel der Eingabe mit dem Pixel der Schablone übereinstimmt (match) und verringert sich, wenn diese nicht identisch sind (mismatch). Die Zuordnung des Zeichen wird anhand des größten Ähnlichkeitswertes festgelegt.

Bei der strukturierten Klassifizierungsmethode werden Zeichen anhand von Merkmalen und Entscheidungsregeln bestimmt. Die Beschreibung eines Zeichens kann durch eine Kombination von Termen, welches auf dieses Zeichen zutreffen, beschrieben werden. Beispielsweise werden Zeichenstriche, Löcher und weitere Zeichenattribute wie Kanten und Wölbungen ermittelt. So kann der Buchstabe P durch einen vertikalen Strich bei dem oben ein weiterer Strich anliegt und eine Wölbung bildet, sodass ein Loch entsteht, beschrieben werden. Basierend auf den ermittelten Merkmalen und einem regelbasiertem System wird das entsprechende Zeichen berechnet.

Die beiden Methoden sind in ihrer einfachen Grundform beschrieben, da es eine Vielzahl an abgeleiteten Variationen und auch Hybridmethoden existieren.

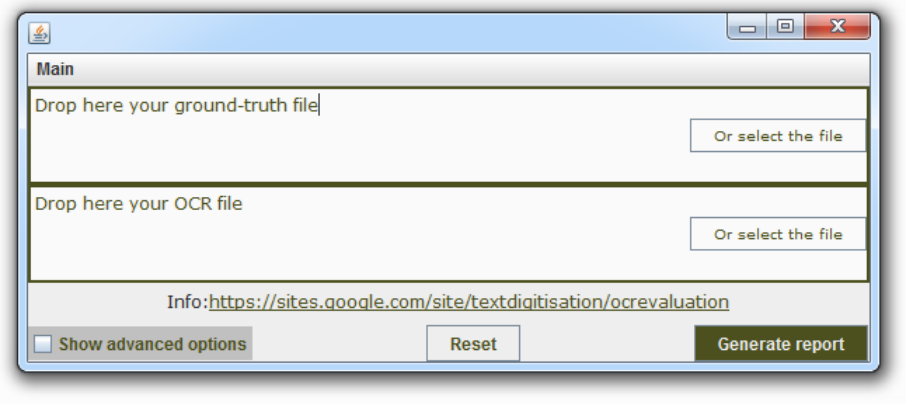


Abbildung 1.1: Evaluationsprogramm “ocrevalUation„

Anschließend folgt eine Auflistung weiterer Techniken:

- **Discriminant function classifiers**

  Verwendet Hyperflächen in multi- dimensionalen Merkmalsräumen um die Featurebeschreibungen in verschiedene semantische Klassen zu separieren. Das Ziel solcher Klassifikatoren ist die Reduktion von dem „mean-squared classification error“.

- **Bayesian classifiers**

  Die Klassifizierung erfolgt anhand einer Kostenfunktion und durch den Einsatz der Wahrscheinlichkeitstheorie, welche mit der Missklassifikation im Zusammenhang steht.

- **Artificial Neural Nets**

  Verwendet „error back-propagation“-Techniken, welche nicht-triviale Zeichenklassifikationen erlernen und mathematische Optimierungstechniken verwenden um die Fehlerrate zu minimieren.[16][S.5 ff]

1.5 Evaluation

Die Evaluation des OCR-Prozesses wird in dieser Bachelorarbeit mithilfe des Open-Source Programmes „ocrevalUation“[1]realisiert, welches, wie in Abbildung 1.1 aufgeführt, das Ergebnis des OCR-Programmes mit dem korrekten Inhalt (Ground Truth) des Dokumentes, ins Verhältnis setzt.

Die produzierten Fehlerarten lassen sich, wie in Tabelle 1.1 ersichtlich, in drei Kategorien klassifizieren. Im Falle der Fehlerart „Insertion“ wird in der OCR-

[1]https://github.com/impactcentre/ocrevalUAtion

Tabelle 1.1: Klassifizierung der Fehlerarten eines OCR-Prozesses

Fehlerart	**Ground Truth**	**OCR**
Insertion	others	others*
Substitution	in	m
Deletion	yours	ours

Datei ein zusätzliches Zeichen hinzugefügt. Im angegebenen Beispiel wurde das Symbol * an das Wort „others“ angehängt.

Eine Substitution beinhaltet das Ersetzen von einem bzw. mehreren Zeichen durch ein anderes. So kann aufgrund einer schlechten Druckqualität der Originaldatei die OCR-Software den Punkt im Buchstaben i nicht als zugehöriges Zeichen erkennen und identifiziert statt „in“ ein „m“.

Die dritte Fehlerklasse wird als „Deletion“ bezeichnet, hier wird ein Zeichen nicht als solches erkennt und fehlt im Resultat des OCR-Programmes. Basierend auf diesen Fehlerklassifikationen lassen sich folgende Fehlerraten definieren:

- *Character error rate (CER)*
- *Word error rate (WER)*

Die „Character error rate“ berechnet sich nach folgender Formel, wobei i die Anzahl der Zeichen darstellt, welche eingefügt worden sind (Insertions), s die Anzahl der Zeichen die substituiert worden sind (Substitutions) und d die Anzahl der Zeichen, welche gelöscht worden sind (Deletions), um das erzeugte OCR-Resultat in das Zieldokument zu transformieren. Die Anzahl aller Zeichen im Referenztext sind durch die Variable n deklariert.

$$CER = \frac{i + s + d}{n}$$

Anhand der Abbildung 1.2 soll der Wert des CER-Wertes von der Ground Truth-Datei mit dem Inhalt „ernest“ und dem OCR-Resultat „nester“ begründet werden. Betrachtet man nun die Positionen der einzelnen Zeichen wird ersichtlich, dass kein Zeichen der zwei Dateien übereinstimmt. Weiterhin wird innerhalb des Wortes nach längsten gemeinsamen Substrings gesucht, um so die minimale Anzahl von Operationen zu finden, welche nötig sind, um das OCR-Resultat in den Referenztext zu transformieren. Dies wird auch als Levenshtein-Distanz bezeichnet. In dem Beispiel ist dieser Substring „nest“ und lässt sich mit 2 Einfügeoperationen („er“) vor den String „nester“und zwei Löschoperationen („er“) am Schluss des Wortes „nester“ in den Referenztext transformieren. Die CER ergibt durch Berechnung der Formel folgendes Resultat:

$$CER = \frac{2 + 0 + 2}{6} = 66,67\%$$

General results

CER	66,67
WER	100,00
WER (order independent)	100,00

Difference spotting

ernest.txt	nester.txt
ernest	nester

Error rate per character and type

Character	Hex code	Total	Spurious	Confused	Lost	Error rate
e	65	2	1	0	1	100,00
n	6e	1	0	0	0	0,00
r	72	1	1	0	1	200,00
s	73	1	0	0	0	0,00
t	74	1	0	0	0	0,00

Abbildung 1.2: Begründung der CER anhand eines Beispieles

Die „Word error rate“ berechnet sich nach folgender Formel, wobei i_w die Anzahl der Wörter darstellt, welche eingefügt worden sind (Insertions), s_w die Anzahl der Wörter die substituiert worden sind(Substitutions), d_w die Anzahl der Wörter, welche gelöscht worden sind (Deletions), um das erzeugte OCR-Resultat in den Referenztext zu transformieren. Die Variable n_w bezeichnet die Anzahl aller Wörter, welche in der Ground Truth-Datei vorhanden sind.

$$WER = \frac{i_w + s_w + d_w}{n_w}$$

Die Anzahl der korrekt erkannten Wörter c_w errechnet sich nach folgender Formel:

$$c_w = n_w - s_w - d_w$$

Durch Einfügeoperationen kann der Wert der CER über 100% ansteigen.

Eine Ausnahme stellen Leerzeichen dar. Die Aneinanderreihung mehrerer Leerzeichen wird als einzelnes Zeichen gewertet. Weiterhin ist es nicht gestattet, dass man Leerzeichen durch druckbare Zeichen substituiert. Der String „abc“, welcher durch den OCR-Prozess als „a c“ erkannt wurde, benötigt somit eine Löschoperation des Leerzeichens und eine Einfügeoperation des Buchstabens „b“. Daraus folgert sich eine CER von 66,67%. Zudem erfolgt eine Differenzierung zwischen

Groß- und Kleinschreibung von Wörtern, welche jedoch in der Anwendung unter dem Punkt „Advanced Options“ bei Bedarf deaktiviert werden kann. Weiterhin ist auch die Zeichenkodierung zu berücksichtigen, da sonst die Qualität des OCR-Prozesses beinträchtigt werden kann.

Nachdem zwei Dokumente verglichen worden sind, werden die einzelnen Operationen unterschiedlich markiert, um die Fehler detailliert betrachten zu können. Substitutionen werden in beiden Texten mit roter Schriftfarbe gekennzeichnet, Einfügeoperationen bzw. Löschoptionen werden aquamarin im OCR-Text bzw. Ground Truth-Text hinterlegt. Neben der CER und WER wird zusätzlich noch eine Übersicht über die Anzahl und Art des Fehlers für jedes Zeichen angegeben. [11]

2 Verwandte Arbeiten

Innerhalb dieses Kapitels soll der aktuelle Stand der OCR-Technik und die Qualität der Erkennung erläutert werden. Weiterhin erfolgt eine Übersicht und eine kurze Beschreibung über nicht-kommerzielle Software.

2.1 State of the Art

Im Jahr 2013 erfolgte ein Auswertung von englischsprachigen Dokumenten und Frakturschriften, welche die Leistungsfähigkeit von LSTM-Architekturen auf OCR-Aufgaben im Verhältnis zu den Anwendungen Tesseract und ABBY demonstrieren sollte. Für den englischen Text wurde von der Universität von Washington ein Datensatz verwendet, welcher 1600 Seiten von naturwissenschaftlichen Zeitschriften und weiteren ähnlichen Quellen beinhaltete. Insgesamt wurden 95.338 Textzeilen für das Training und 1.020 Zeilen für das Testset benutzt. Mit Ocropus, welches die LSTM-Architektur verwendet, konnte eine CER von 0,6% erreicht werden. Damit diese Ergebnisse vergleichbar sind, wurde im weiteren Verlauf das gleiche Testset verwendet. Tesseract erreichte eine CER von 1,299% mit zeilenweisem Einlesen des Textes und dem englischem Sprachmodell. Der Recognition Server von ABBYY [1] erreichte mit der Einstellung Englisch eine Zeichenfehlerrate von 0,85%. Zudem ist anzumerken, dass all diese sprachmodellbasierten Systeme das OCR-Resultat, im Gegensatz zu dem LSTM-Netzwerk, mittels Nachbearbeitung von Sprachmodelltechniken verbessern.

Im nächsten Testszenario wurde eine Evaluation auf deutscher Frakturschrift auf Theodor Fontanes „Wanderungen durch die Mark Brandenburg“ und der Ersch-Gruber Enzyklopädie durchgeführt. Text, welcher die Schriftart Antiqua beinhaltete, wurde von der Auswertung ausgeschlossen. Auf zufällig gewählten Seiten von Fontane, welche insgesamt 8.988 Zeichen in Frakturschrift verwendeten, wurde mittels LSTM eine Fehlerrate von 0,16% ermittelt. Auf Ersch-Gruber betrug die CER 0,82% bei 10.881 Frakturzeichen. Bei Tesseract betragen die zu vergleichenden Fehlerraten 0,898% (Fontane) und 1,47% (Ersch- Gruber), wobei die Anwendung zusätzlich ein deutsches Wörterbuch verwendet, als auch eine Anpassung der Schriftart vornimmt. Das kommerzielle Produkt ABBY lieferte CERs von 1,23% auf Fontane und 0,43% auf Ersch-Gruber. In der folgenden Tabelle 2.1 werden die gesammelten Daten kurz präsentiert.

[1] http://www.abbyy.com/recognition-server/

Tabelle 2.1: Auswertung von Ocropus-LSTM, Tesseract und ABBYY auf englischen Dokumenten und zwei deutschsprachigen Büchern in Frakturschrift

Dokument	OCRopus-LSTM	Tesseract	ABBY Server
Englisch	0,6 %	1,3 %	0,85 %
Fontane	0,15 %	0,9 %	1,23 %
Ersch-Gruber	1,37 %	1,47 %	0,43 %

Tabelle 2.2: Auswertung deutscher Frakturschrift unterschiedlicher Jahrhunderte mittels Zeichenpräzision in Prozent

Jahr	ABBYY FineReader 11.1	Tesseract 3.03	OCRopus 0.7
1544	83,14 %	70,32 %	74,59 %
1649	88,07 %	84,87 %	78,98 %
1779	82,13 %	80,77 %	75,46 %

Durch den Einsatz von LSTM-Netzwerken kann die Fehlerrate ohne Verwendung von Sprachmodellen mit aktuellen OCR-Anwendungen konkurrieren und teilweise bessere Resultate als kommerzielle Produkte auf deutscher Frakturschrift erzielen.[17]

Ein weiterer Beleg für die Qualität von LSTM-Technik findet sich in einer Veröffentlichung der Ludwigs-Maximilians-Universität München aus dem Jahr 2014. In diesem Testszenario wurden die Programme ABBYY Finereader 11.1, Tesseract 3.03 und OCRopus 0.7 mittels Erkennungsgenauigkeit von Zeichen auf deutsche Texte in Fraktur aus unterschiedlichen Jahrhunderten evaluiert. Hierbei wurde vorerst kein Sprachmodell oder das default-Modell (Englisch) gewählt. Auf allen drei Dokumenten konnte sich laut Tabelle 2.2 ABBYY gegen Tesseract und OCRopus durchsetzen.

Im nächsten Testszenario wurden fünf Seiten aus „Pontanus, Progymnasmata Latinitatis“ aus dem Jahr 1589 für die Evaluation der Programme ausgewählt. Neben den vorherigen Konfigurationen der Anwendungen, wurden OCRopus und Tesseract anhand von künstlich generierten Seiten der Schriftart trainiert. Zusätzlich wurde ermittelt, ob sich durch den Einsatz eines Wörterbuches zur Nachkorrektur die Qualität von Tesseract verbessert. Vergleicht man nun ABBY mit den untrainierten Anwendungen, so erreichte OCRopus auf einer von den fünf Seiten einen minimal besseren Wert. Die trainierten Modelle zeigen jedoch eine starke Verbesserung der Erkennung des OCR-Prozesses. Wenn man beispielhaft die Zeichengenauigkeit einer Seite betrachtet, bei der ABBY eine Präzision von 87,79% realisierte und mit OCRopus vergleicht, konnte sich OCRopus durch Training von 80,70% auf 92,55% steigern und so ein besseres Resultat realisieren. Tesseract konnte ebenfalls eine Verbesserung aufweisen, jedoch unterlag das Pro-

Tabelle 2.3: Auswertung von „Pontanus, Progymnasmata Latinitatis“ aus dem Jahr 1589

Seite	ABBYY FR 11.1	Tesseract 3.03	OCRopus 0.7	Tesseract (font)	Tesseract (font+lex.)	OCRopus (font)
15	87,79 %	80,88 %	80,70 %	91,02 %	93,90 %	92,55 %
16	82,94 %	77,41 %	76,94 %	80,12 %	85,65 %	80,47 %
17	85,25 %	75,98 %	86,07 %	85,41 %	91,56 %	93,93 %
18	85,93 %	79,51 %	85,53 %	88,29 %	92,68 %	89,67 %
19	87,94 %	80,09 %	79,09 %	86,06 %	90,15 %	87,83 %

Tabelle 2.4: Auswertung von Thanner, Petronij Arbitri Sathyra aus dem 15. Jahrhundert

Seite	**Tesseract 3.03**	**OCRopus 0.7**	**OCRopus (trained)**
13	41,59 %	44,59 %	93,15 %
14	52,38 %	57,77 %	94,61 %
15	53,09 %	62,38 %	95,17 %
16	59,09 %	61,45 %	93,27 %

gramm weiterhin in drei von fünf Seiten gegenüber ABBY. Durch den Einsatz des Wörterbuches konnten die Ergebnisse allerdings signifikant verbessert werden, sodass nun Tesseract gegenüber ABBY als auch OCRopus auf nahezu allen Seiten dominierte. So konnte sich Tesseract mit dem Standardmodell von 80,88% auf 91,02% mit Training der Schriftart und auf 93,90% mithilfe des zusätzlichen Wörterbuches steigern. Eine Übersicht der einzelnen Ergebnisse soll die Tabelle 2.3 verdeutlichen.

Im letzten Szenario erfolgte ein Training anhand von 12 Seiten des Dokumentes Thanner, Petronij Arbitri Sathyra aus dem 1500 Jahrhundert. Anhand von vier ausgewerteten Seiten wird sehr deutlich, dass die Zeichenerkennung durch Training auf historischen Texten mittels LSTM erheblich verbessert werden kann. Während Tesseract 3.03 und OCRopus 0.7 ohne Training ähnliche Ergebnisse erzielen, wurde die Zeichenerkunnung bei beiden Programmen durch das Training um mindestens 30% verbessert. Anhand der folgenden Tabelle 2.4 wird die starke Diskrepanz zu Tesseract 3.03 und OCRopus ersichtlich.[28]

Im September 2014 wurde im Rahmen einer Masterarbeit der Julius-Maximilians Universität die Leistungsfähigkeit des vorläufigen Releasekandidaten von Tesseract 3.04 auf dem zweibändigen Nachdrucks der Würzburger Bischofs-Chronik von Lorenz Fries von 1924 evaluiert. Hierbei wurde neben dem deutschen Frakturmodell (deu-frak), das fränkische (frk) sowie dänische (tesseract-dan-fraktur) Paket und das Ergebnis der Eigenleistung der Arbeit berücksichtigt. Auch in diesem Test bestätigte sich, dass die vortrainierten Pakete deutliche Defizite in der Erkennung aufweisen. Das Paket frk erreichte hierbei die schlechtesten Resultate mit einer CER von 6,27%. Die beiden Modelle deu-frak und tesseract-dan-fraktur erzielten mit 2,5% bzw. 2,44% bessere Ergebnisse, welche jedoch mit einer CER

Tabelle 2.5: Auswertung von verschiedenen Sprachmodellen von Tesseract auf der Würzburger Bischofs-Chronik von Lorenz Fries

Sprachpaket	**Zeichenfehlerrate**
frk	6,27 %
deu-frak	2,5 %
tesseract-dan-fraktur	2,44 %
Eigenleistung Vorbach	1,85 %

Tabelle 2.6: Auswertung von GOCR und Tesseract mit unterschiedlichen Helligkeitswerten

Helligkeit	Anzahl extrahierter Zeichen von Tesseract	Anzahl korrekt erkannter Zeichen von Tesseract	Anzahl extrahierter Zeichen von GOCR	Anzahl korrekt erkannter Zeichen von GOCR	Tesseract Genauigkeit in %	Tesseract Präzision in %	GOCR Genauigkeit in %	GOCR Präzision in %
25	39	37	28	23	94,8	94,8	58,9	82,1
50	39	38	27	26	97,4	97,4	66,6	96,2
100	39	37	1	1	94,8	94,8	2,5	100

von 1,85% nicht mit dem Ergebnis der Arbeit konkurrieren können. Diese Werte können auch in der Tabelle 2.5 nachvollzogen werden. [31]

In einem direkten Vergleich der Open-Source Programme GOCR und Tesseract aus dem Jahre 2013 resultierte, dass GOCR nur in seltenen Fällen mit der Qualität von Tesseract mithalten kann. Hierbei wurden verschieden Dokumente verwendet, welche u.a. verschiedene Auflösungen, Helligkeitswerte oder Schriftarten beinhalteten. Die Tabelle 2.6 stellt einen Ausschnitt des Tests dar, welches den Vergleich zwischen unterschiedlichen Werten für die Helligkeit des Dokumentes und den Einfluss auf die Qualität verdeutlicht. [19]

Weiterhin findet im zweijährlichen Turnus eine Konferenz mit dem Titel „ International Conference on Document Analysis and Recognition“ (ICDAR) statt, welche sich u.a. mit Problemstellungen der OCR beschäftigt. Bei der letzten Konferenz aus dem Jahr 2013 wurde ein Paper veröffentlicht, welches sich mit der Problemstellung der OCR-Technik auf unterschiedlichen Schriftarten und Schriftgrößen beschäftigt. In diesem Testszenario wurden 17 verschiedene Schriftarten für englische Dokumente und 14 für Gurumukhi verwendet. Weiterhin wurde zwischen elf Schriftgrößen differenziert. Die Featureextraktion wurde durch zwei Sets realisiert. Das erste Set besteht aus 189 Merkmalen, welche auf Gabor Filter basieren. Das zweite Set verwendet 200 Features durch den Einsatz des Gradienten. Für die Klassifikation wird eine „Support Vector Machine“ (SVM) eingesetzt, welche die Leistung von einem linearen, polynomialen und Gaussian (RBF)Kernel evaluiert. Die Mehrfachklassifikation ist durch die Kombination von mehreren binären SVM möglich. Hierbei unterscheidet man zwischen der Kombinationsart „One versus All“ (OVA) und „One versus One“ (OVO). Um die Qualität der Klassifizierer zu validieren, wurden verschiedene Experimente mittels OVO durchgeführt. Zuerst wurde die Genauigkeit der Texterkennung für das komplette Skript basierend auf einer zehnfachen Kreuzvalidation des ganzen Datensets

Tabelle 2.7: Auswertung von SVM-Kernels mit Gabor- und Grandientenfeaturen

Verwendete Features		**Linear**	**Polynomial**	**RBF**
Durchschnittliche Genauigkeit	Gabor Features	97,85 %	98,89 %	98,9 %
	Gradient Features	97,17 %	99,23 %	99,45 %
Standardabweichung	Gabor Features	0,22	0,20	0,23
	Gradient Feartures	0,57	0,20	0,19

Tabelle 2.8: Auswertung unterschiedlicher Methoden und Ausgangsbasen

Methode veröffentlicht durch	**Ausgangsbasis**	**Methode**	**Genauigkeit**
Zhang et al.[33]	Chinesisch, Englisch	Sructural Features und SVM	99,3 %
Sanguansat et al.[24]	Thai, Englisch	Hidden Markov Model	99,31%
Zhu et al.	Chinesisch, Englisch	Feature Selection and Cascade Classifier	99,25%
Rani et al.[23]	Gurmukhi, Englisch	Gabor Features and SVM Gradient Feartures and SVM	98,90% 99,45%

durchgeführt. Die beste Leistung konnte der RBF-Kernel mit dem Einsatz von Gradientenfeaturen mit einer maximalen Durchschnittsgenauigkeit von 99,45% und der niedrigsten Standardabweichung von 0,19 erzielen. Anhand Tabelle 2.7 soll verdeutlicht werden, dass die SVM mit RBF und Polynomial Kernelfunktionen sowohl mittels Gabor- und Grandientenmerkmalsextraktion bessere Leistung erzielen als der lineare Kernel.

Im Abschluss dieses Papers wird ein Vergleich zu anderen Methoden aufgelistet, welche in der Tabelle 2.8 ersichtlich ist. Diese haben jedoch eine andere Daten als Ausgangsbasis und können nicht direkt ins Verhältnis zu den ermittelten Leistungen dieser Experimente gesetzt werden. [23]

2.2 Marktübersicht für nichtkommerzielle OCR-Software

Die folgende Auflistung liefert einen Überblick über kostenfreie OCR-Programme.

- **CuneiForm**[2] Das russische Unternehmen Cognitive Technologies entwickelte das mittlerweile kostenfreie Programm und ermöglicht der Open Source Community Zugriff auf den Sourcecode. Neben über 20 verschiedenen Sprachpaketen verwendet CuneiForm ein Wörterbuch um die Zeichenerkennung weiter zu verbessern. Das kommandozeilenbasierte Programm wird mittlerweile in diversen Frontends wie z.B. YAGF verwendet. Als Ausgabeformat werden u.a. HTML, HOCR oder RTF angeboten. [1] [2]

[2] http://cognitiveforms.com/products_and_services/cuneiform

- **FreeOCR**[3] Die auf Tesseract basierende Engine vereinfacht die Bedienung durch eine graphische Oberfläche und unterstützt die folgenden Formate: BMP, JPEG, GIF, PNG, PDF und TIFF. Als Ausgabeformat können TXT,DOC oder RTF konfiguriert werden. Fehlende Sprachpakete können von der Homepage von Tesseract[4] heruntergeladen und hinzugefügt werden. [5]

- **GOCR**[5] ermöglicht eine Verarbeitung einer Vielzahl von Eingabeformaten wie z.B. GZIP, BZIP2, PNG oder NETPBM. Als mögliche Ausgabeformate sind ISO8859, TeX, HTML, XML, UTF-8 oder ASCII konfigurierbar. Weitere detaillierte Information werden im Kapitel 4.3 erörtert. [26]

- **Google Docs via Google Drive**[6] Nach der Registrierung in Google Drive kann der Benutzer seine Daten mit der Cloud synchronisieren und Online den OCR-Prozess durchführen. Analog zu GOCR erfolgt eine kurze Erläuterung zu Limitierungen und Durchführung im Kapitel 4.5. [12]

- **OCRAD**[7] basiert auf der Extraktion von Features und liest PPM Dateien ein. Die Ausgabe erfolgt als Textdatei im byte- oder UTF-8 Format. Auch für dieses kommandozeilenbasierte Programm exisitieren Frontends, wie z.b. ocrfeeder[8] [6]

- **Ocrocis**[9] stellt ein Managerinterface für Ocropy für das Betriebssystem Linux und Mac OS X dar, welches es ermöglicht den Trainingsprozess zu vereinfachen, indem weniger Parameter benötigt werden. Weiterhin erfolgt unter Linux durch Verwendung von Hardlinks eine Reduktion von redundanten Dateien. Eine detaillierte Anleitung wird von der Universität München (LMU) bereitgestellt.[10] [8]

- **Ocropy**[11] Die Anwendung basiert auf Recurrent Neural Networks (RNN) und ermöglicht das Generieren von neuen Paketen durch Training. Der OCR-Prozess lässt sich in drei Phasen gliedern und wird schrittweise per Kommandozeile realisiert. Die einzelnen Schritte der Binarisierung, Segmentierung bzw. Erkennung des Textes sowie des Trainings werden bei Auswahl der Anwendungen für die Testszenarien näher beschrieben. Weitere Informationen in Kapitel 4.2. [30]

[3] https://www.heise.de/download/freeocr-1149486.html
[4] https://code.google.com/p/tesseract-ocr/
[5] http://jocr.sourceforge.net/
[6] https://www.google.com/intl/de_de/drive/
[7] http://www.gnu.org/software/ocrad/
[8] https://code.google.com/p/ocrfeeder/
[9] https://github.com/kmnns/ocrocis
[10] http://cistern.cis.lmu.de/ocrocis/
[11] https://github.com/tmbdev/ocropy

- **Puma.NET**[12] Hierbei handelt es sich um einen Wrapper, für die OCR-Engine „Cognitive Technologies CuneiFrom recognition“[13] und in in allen .NET Framework 2.0 oder aktuelleren Applikationen integriert werden kann. Insgesamt werden 27 Sprachen unterstützt. Akzeptiert werden die Formate BMP, GIF, EXIG, JPG, PNG und TIFF und können als TXT, RTF oder HTML mittels OCR-Prozess übersetzt werden. [9]

- **Tesseract**[14] Die ursprünglich von der Firma Hewlett-Packard im Jahre 1994 entwickelte Software, unterliegt seit dem Jahre 2006 der Finanzierung und somit Förderung von Google. Die aktuelle Version von Tesseract 3.02[15] stammt aus dem Jahr 2012 und bietet eine Vielzahl von Sprachmodellen an. Durch den Einsatz von Leptonica-Bibliotheken[16] wird eine Vielzahl von Eingabeformaten aktzeptiert. Der komplexe Ablauf des Trainings und die Erkennung von Dokumenten kann durch graphische Frontends stark vereinfacht werden. Eine ausführliche Beschreibung folgt im Kapitel 4.1. [32] [31]

[12] http://pumanet.codeplex.com
[13] http://cognitiveforms.com/products_and_services/cuneiform
[14] https://code.google.com/p/tesseract-ocr/
[15] https://code.google.com/p/tesseract-ocr/downloads/list
[16] http://www.leptonica.com/

3 Fraktur

Laut der DIN 15618 stellt die Frakturschrift eine Subkategorie der gebrochenen Schriften neben Gotisch, Rundgotisch, Schwabacher und weitere Fraktur-Varianten dar. Die Frakturschrift war vor allem im deutschsprachigen Raum anfangs des 16.Jahrhunderts etabliert und wurde zu Beginn des 20.Jahrhunderts durch die Schriftart Antiqua verdrängt. [31] Die Eingliederung einer Schriftart, welche für gebrochene Schrift anhand der Abbildung 3.1 erfolgt, wird durch typische Klassifikationsmerkmale bzw. Eigenschaften der jeweiligen Schrift wie zum Beispiel durch das Auftreten von Serifen (1), dem Querstrich des Buchstabens „e“ (2) oder dem Winkel oder der Strichstärke des k-Schenkels (3) bestimmt. [21][S.252]

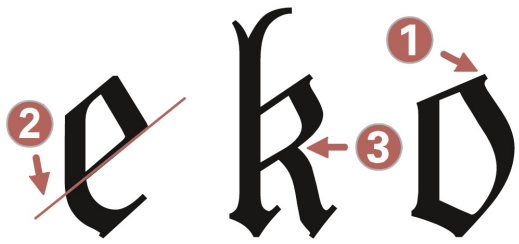


Abbildung 3.1: Beispiel von Merkmalen anhand gebrochener Schrift[21][S.252]

Zu den Gemeinsamkeiten der gebrochenen Schriften zählt, dass die Bögen innerhalb von Zeichen, welche durch eine ungleichmäßige Schreibbewegung hervorgerufen wird, gebrochen wirken. [31]Die Abbildung 7.2 illustriert die Bogenbrechungen der gebrochenen Schriften im Verhältnis zur runden und nicht gebrochenen Schriftart Antiqua. [22] In der Frakturschrift existieren gesonderte Charakteristika, wie die Differenzierung von langem ſ und rundem s, welche nach bestimmten Regeln eingesetzt werden.

Ersteres wird im Anlaut oder innerhalb einer Silbe verwendet, wie zum Beispiel in dem Wort „ſagen“ oder „Manuſkript“. Zusätzlich wird das ſ statt dem runden s, welches auch als Schluss-s bezeichnet wird, benutzt, wenn ein unbetontes e als Auslaut auf den Silbenanlaut s, wie zum Beispiel „ich preiſ“ für „ich preiſe“, folgt. Typische Buchstabenkombination in Wörtern wie ſch in „ſchaden“ , ſp in „Knoſpe“ oder ſt in „geſtern“ begründen das lange ſ. Sollten die Buchstabenvariationen allerdings durch Zusammensetzung resultieren, ist das runde s, wie in „Zirkuschef“ , „transparent“ oder „Dienstag“, zu verwenden. Neben dem Einsatz des runden s als Auslaut wie in „das“,

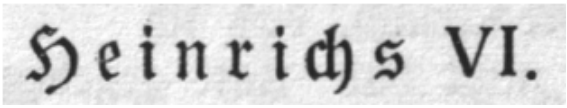


Abbildung 3.2: Beispiel einer Sperrung mit der ausgeschlossenen Ligatur ch und einer römischen Ziffer
[31][S.7]

wird dieses noch in manchen Fremdwörtern als Kombination in sk verwendet, wie zum Beispiel bei „groteſk“ oder „Obeliſk“.

Weiterhin werden Ligaturen eingesetzt, welche eine Verkettung von bestimmten Zeichen, wie zum Beispiel von „ch“ nach „ch“, zu einem Ersatz dieser durch ein neues, meist besser lesbares Zeichen umfasst. Ligaturen werden ebenfalls für die Verwendung von Initialen zu Beginn von Kapiteln bzw. Abschnitten benutzt. Eine besondere Berücksichtigung erfolgt bei der Zeichenfolge von einem langem ſ und einem ʒ, welches von ſʒ zu dem Einzelbuchstaben ß transformiert wird. Um Wörter hervorzuheben wird in der Frakturschrift eine Sperrung, die den Abstand der einzelnen Zeichen innerhalb des Wortes erhöht, eingesetzt. Ausgeschlossen von der Sperrung sind die Ligaturen ch, ck oder ß. Sollten in dem Text römische Ziffern oder Fremdwörter, welche nicht eingedeutscht wurden, eingesetzt werden, so wird die Schriftart Antiqua verwendet. Ein Beispiel für eine Sperrung mit einer ausgeschlossenen Ligatur und einer römischen Ziffer ist in Abbildung 3.2 illustriert.

Weitere Beispiele zur Frakturschrift und Besonderheiten sind aus dem Paper „Analysen und Heuristiken zur Verbesserung von OCR-Ergebnissen bei Frakturtexten“ zu finden.[31]

4 Übersicht über ausgewählte OCR-Software

Dieses Kapitel soll einen Einblick über die Funktionsweise und Verwendung der ausgewählten Programme für die Testszenarien geben. Durch die beträchtliche Vielfalt an Softwarelösungen für Texterkennung, wurde die Auswahl auf die nachfolgenden Programme limitiert.

4.1 Tesseract

Die Anwendung Tesseract ermöglicht neben dem englischen und deutschen Sprachmodell eine Vielzahl von weiteren Sprachmodellen, unter anderem auch deutsche Frakturschrift und Fränkisch. Nach den Vorverbreitungsschritten wie Binarisierung und Layoutanalyse folgt die Merkmalsextraktion basierend auf normalisierten Polygon-Approximationen des Zeichenumrisses.

Die Klassifizierung ist durch den „k-nächste-Nachbar-Algorithmus“ realisiert. [31][S.13 ff] Durch die Etablierung auf dem Markt und den signifikanten Vorsprung zu kommerziellen Produkten im Jahr 1995, entstanden bis heute diverse graphische Oberflächen und Tools wie zum Beispiel gImageReader, OCRFeeder oder SunnyPage OCR, welche die ursprüngliche Bedienung über die Kommandozeile vereinfachen.[32]

Weiterhin lassen sich neue Sprachmodelle über eigenes Training erstellen bzw. verbessern. Das Training wird mit der Trialversion des kostenpflichtigen Programmes Sunnypage OCR[1] in der Version 2.1 durchgeführt. Die neueste Version ist 2.7, allerdings wird angegeben, dass diese Tesseract Version 3.04 verwendet, obwohl noch kein offizieller Releasekandidat exisitert. Durch die Verwendung von Sunnypage wird der Trainingsprozess stark vereinfacht. Nach dem Start des Programmes wird das zu trainierende Bild geöffnet und via den Button „Training“ eine neue Sprache erstellt. Weiterhin erfolgt die Benennung des Sprachmodells, die Erstellung eines individuellen Zeichensets und die Auswahl, dass kein Wörterbuch verwendet werden soll. Nach Abschluss der Konfiguration erfolgt eine automatisierte Layoutanalyse des Bildes und im Anschluss beginnt der Trainingsprozess. Das zu trainierende Zeichen ist einer Box markiert, welches bei falscher Segmentierung bzw. Überlappung mit anderen Zeichen bearbeitet werden kann. Bei jeder

[1] http://www.sunnypage.ge/en

Box besteht die Möglichkeit dieses Zeichen zu erlernen, in dem das Zeichen annotiert und mit dem „Train“-Button bestätigt wird. Alternativ können erfasste Zeichen mit „Skip“ übersprungen werden. Nach der Bearbeitung aller erkannter Zeichen oder dem Abbruch des Trainingsprozesses wird das Sprachmodell erstellt und das Programm beendet sich. Mit dem Neustart des Programmes lassen sich selbst erstellte Sprachmodelle über den “Training„-Button mit der Auswahl des erstellten Sprachmoduls und der Option „Verbesserung der aktuellen Sprache“ erweitern. Nachdem die Sprachmodelle mit der Demoversion erstellt wurden, werden diese im Tesseract Verzeichnis in den Unterordner „tessdata“ kopiert und umbenannt, da der Erkennungsprozess bzw. die Verwertung des erkannten Textes nicht im Umfang der Demoversion liegt. Im Anschluss erfolgt der OCR-Prozess der Seiten kommandozeilenbasiert über Tesseract 3.02 mit folgendem Befehl:

tesseract.exe [Bilddatei] [Ausgabedatei] -l [Sprache]

4.2 Ocropy

Im Verhältnis zu Tesseract handelt es sich bei dem linuxbasierenden Programm Ocropy[2] (früher Ocropus) nicht um ein segmentierungsbasierendes OCR-System, sondern um segmentierungsfreie Technik. Durch den Einsatz von Recurrent Neural Networks (RNN) mittles Long Short Term Memory (LSTM) wurden viele Probleme bzw. Limitierungen vorheriger Architekturen beseitigt. Recurrent Neutral Networks wurden als gut für textbezogene Verarbeitungen und die Erkennung von Mustern, welche im Zeitverlauf auftreten, erachtet. Dennoch konnten traditionale RNN die erwartete Leistung in aufwändigen Aufgaben wie in der OCR oder der Spracherkennung u.a. wegen dem „vanishing gradient problem“ nicht bestätigen.[17]

Anfang des Jahres 2015 wurde das Framework Ocrocis[3] veröffentlicht, welches die Abfolge der einzelnen Schritte des Erkennungs- bzw. Trainingsprozesses stark vereinfacht. Zum besseren Verständnis wird die Konfiguration meist anhand der Ocropy-Befehle aufgezeigt. Im ersten Schritt erfolgt eine Binarisierung und Entzerrung des Bildes durch folgende Anweisung:

ocropus-nlbin <image-dir>/*.png -o book

Anschließend wird eine Zerlegung des binären Bildes in Textzeilen vorgenommen, welche für jede Datei einen Ordner anlegt und jede erkannte Textzeile dieses Bildes in das entsprechende Verzeichnis mit Dateiendung bin.png erstellt. [27] Falls das Bild zu klein ist, kann die Überprüfung mit dem Parameter **-n** unterdrückt bzw. mittels –**maxcolseps 0** kann Ocropy so konfiguriert werden, dass

[2] https://github.com/tmbdev/ocropy
[3] https://github.com/kmnns/ocrocis

dieses Bild für das Programm nur eine Spalte beinhaltet.[30] Dies wird durch folgenden Befehl realisiert:

ocropus-gpageseg book/*.bin.png

Folgend werden die Seiten ausgewählt, welche für Trainings- bzw. Testzwecke verwendet werden sollen und eine correction.html erstellt, welche eine Annotierung der einzelnen Textzeilen ermöglicht. Falls Segmentierungsprobleme z.B. durch zwei Zeilen als eine Erkannte oder unvollständige Zeichen beinhalten, sind diese Trainingszeichen leer zu lassen und werden beim Training ignoriert.

ocropus-gtedit html book/00[1-2][0-9]/*.bin.png

Der nächste Befehl ist für die Generierung der Ground Truth aus den annotierten Zeilen anhand der correction.html zuständig und speichert diese in das Verzeichnis des zugehörigen Bildes als gt.txt:

ocropus-gtedit extract correction.html

Anschließend wird manuell ein Subset für das Training bzw. zum Test ausgewählt und separat in Verzeichnissen abgespeichert. Danach ermöglicht der ocrocis-Befehl nun sehr einfach die Vorbereitung für das Training der ersten Iteration durch Erzeugung eines Verzeichnisses Iterations/01 mit einer Correction.html. Diese gilt es ebenfalls zu annotieren. Falls diese in vorherigen Schritten schon durch die correction.html beschriftet worden sind, können diese kopiert werden. Beispielhaft für die Vorbereitung der ersten Iteration der Seiten 10-29:

ocrocis next 10..29

Eine Durchführung des Trainings wird durch das Kopieren der Ground Truth Daten in das Verzeichnis iterations/01/annotation/ ,deren Verlinkung zu einem neu erstellten Trainingsverzeichnisses und durch die Bestimmung eines individuellen Zeichensets, angelegt. Im folgenden Beispiel beinhaltet das Training 30.000 Schritte, bei dem im Intervall von 1.000 Schritten eine Sicherung des aktuellen Modelles in eine Datei im Ordner model erfolgt. Das Training beinhaltet eine Seite aus dem Trainingspool. Die Realisierung erfolgt durch die Anweisung:

cropus-rtrain –ntrain 30000 –savefreq 1000 –codec ./book/charset.txt –output ./iterations/01/models/model ./training/0001/*.bin.png 2<&1

Der Prozess kann, wie in Abbildung 4.1 ersichtlich, im Terminal verfolgt werden. In der ersten Zeile des Trainingsschrittes ist der aktuelle Schritt, gefolgt von einem Wert für die Unsicherheit der Ausgabe. Dieses Maß ist selbst bei perfekter Erkennung bei OUT niemals null, da es sich um eine A-posteriori-Wahrscheinlichkeit zwischen 0 und 100% für jedes Zeichen handelt und das Maß suggeriert die Summe aller Werte der jeweiligen Zeile. Anschließend folgt die Breite bzw. Höhe in Pixel des Zeichens in Klammern und der Titel des verwendeten Bildes.

Die nächsten drei Zeilen beinhalten die Groundtruth (TRU), die Netzwerk-

```
16013 5.98 (520, 48) train/0014/010015.bin.png
   TRU: u'of Pirats and A\u017f\u017fa\u017f\u017fins. Therefore'
   ALN: u'of Pirats and A\u017f\u017fa\u017f\u017fins. Therefore'
   OUT: u'of Pirats and A\u017f\u017fa\u017f\u017fins. Therefore'
```

Abbildung 4.1: Terminal während des Trainingsprozesses

ausgabe, nachdem die Daten gesichtet worden sind (ALN) und die tatsächliche Netzwerkvorhersage (OUT). Ein perfekt trainiertes Netzwerk würde eine Übereinstimmung aller drei Zeilen voraussetzen. Die Ausgabe in der Kommandozeile ist im ASCII Code, da nicht alle Terminals UTF-8 unterstützen. Spezielle UTF-8 Zeichen werden mit nicht-ASCII Zeichen dargestellt. Nachdem die Modelle erstellt worden sind und die Testseiten binarisiert bzw. segmentiert sind, können diese über folgenden Befehl auf die zu erkennenden Seiten angewendet werden:

ocropus-rpred -m model.pyrnn.gz book/0001/*.png

Zuletzt werden die generierten Textdateien der einzelnen Zeilen des Bildes, welche unter book/0001/ erzeugt worden sind, zu einer Datei mit folgendem Befehl vereinigt. [27]

cat book/0001/??????.txt > ocr.txt

Eine ausführlichere und praxisbezogene Anleitung wurde von Ludwig-Maximilians-Universität aus Munich veröffentlicht.[4]

4.3 GOCR

Das Programm GOCR[5] von Joerg Schulenburg wurde im Jahre 2000 erstmals veröffentlicht und bis zum Jahre 2013 kontinuierlich verbessert. Der aktuelle Release GOCR 0.50 vom März 2013 wurde für die Auswertung der Tests herangezogen. [26] Seit Version 0.37 verwendet GOCR einen Wahrscheinlichkeitswert in Verbindung mit dem jeweiligen Zeichen und ermittelt anhand des besten Wertes das Ausgabezeichen. Es handelt sich um eine regelbasierende Engine, welche durch Merkmalsextraktion wie z.B. horizontalen oder sich kreuzenden Linie das Zeichen ermittelt. [25] Nach erfolgreicher Installation lässt sich das Programm über Kommandozeile zur Auswertung mit folgendem Befehl nutzen:

gocr [OPTION] [-i] Datei

Zusätzlich besteht die Möglichkeit GOCR zu Trainieren. Dies wird ermöglicht, indem man einen Unterordner db mit einer Datei namens db.lst im Hauptverzeichnis anlegt. Innerhalb dieses Ordners werden sämtliche erlernten Zeichen als pbm-Datei hinterlegt und mithilfe der db.lst einem Buchstaben zugewiesen. Durch den optionalen Parameter „-m 130“ wird der interaktive Modus gestar-

[4] http://cistern.cis.lmu.de/ocrocis/tutorial.pdf
[5] http://www.gocr.de

tet und das Programm stellt innerhalb der Konsole das erfasste Zeichen dar. Dies kann nun dauerhaft in der Datenbank gespeichert, nur im Arbeitsspeicher für den nächsten OCR-Prozess genutzt oder bei falscher Segmentierung ignoriert werden. Bei Neutraining einer Sprache besteht die Möglichkeit mittels „-m 2“ den Abgleich mit der Datenbank zu nutzen und durch „-m 256“ die integrierte OCR-Engine zu deaktivieren, um nur trainierte Zeichen zur Erkennung zu benutzen. [26]

4.4 Asprise OCR

Das kommerzielle OCR-Tool Asprise OCR[6] wird in der aktuellen Version 15.3 für die Tests verwendet. Neben den Standardmodellen wie Deutsch und Englisch, ist auch ein Modul für deutsche Frakturschrift integriert. Die Auswertung erfolgt mittels einer Demoversion, welche nur einen Bruchteil der zwanzig geworbenen Sprachen enthält. Die Durchführung des OCR-Prozesses wird durch eine GUI intuitiv bedienbar. [7]

4.5 Google Docs

Das Unternehmen Google ermöglicht ein integriertes OCR-Feature in Google Docs durch die Verwendung von Google Drive[7]. Nachdem Bilddateien im Format JPG,GIF oder PNG in Google Drive hochgeladen worden sind, können diese via Google Docs geöffnet werden. Eine Limitierung stellt eine maximale Dateigröße von 2MB dar. In diesem Textverarbeitungsprogramm wird neben dem Bild auch der erkannte Text angefügt. Die Auswahl des Sprachmodelles erfolgt automatisiert. [12] Über die verwendete OCR-Engine konnten keine Informationen ermittelt werden. Dennoch ist bekannt, dass der Entwickler Ray Smiths, welcher die Engine von Tesseract OCR entwickelt hat, mittlerweile für Google an Tesseract arbeitet. [10]

[6] http://asprise.com/royalty-free-library/java-ocr-api-overview.html
[7] https://drive.google.com/drive/my-drive

5 Evaluation

In diesem Kapitel wird mithilfe der zuvor erläuterten Programme die Qualität der Texterkennung basierend auf unterschiedlichen Dokumenten miteinander verglichen. Zu Beginn der Testreihe liegt ein deutsch- bzw. englischsprachiger Text zugrunde, bei dem die Standardoptionen bzw. die verfügbaren Sprachmodelle der jeweiligen Software verwendet werden. Im Anschluss werden die Programme, sofern möglich, anhand eines erstellten Dokumentes trainiert und mit den zuvor erhaltenen Ergebnissen ins Verhältnis gesetzt. Im nächsten Testszenario erfolgt ein ausführliches Training anhand eines Schriftstückes mit deutscher Frakturschrift. Anschließend folgt eine Evaluation der trainierten und den bereits integrierten Modellen anhand eines Testsets. Die gewonnen Sprachmodelle werden auf ein Dokument mit deutscher Frakturschrift eines anderen Jahrzehntes angewandt. Im letzten Testszenario wird ein lateinisches Ausgangsdokument in Frakturschrift für die Evaluation zu Grunde gezogen und neue Sprachmodelle durch Training generiert. Zu jedem Testszenario werden die erzeugten OCR-Resultate mithilfe des Auswertungsprogrammes „ocrevalUAtion“ mit der Ground Truth-Datei ausgewertet und die ermittelten Werte im Anschluss miteinander verglichen.

5.1 Erkennung von deutsch- und englischsprachiger Literatur

5.1.1 Ausgangsmaterial

Seitens der Tests für die deutschsprachige Texterkennung wurden die Seiten fünf bis neun aus aus dem PDF-Dokument mit dem Titel „George R. R. Martin Die Herren von Winterfell Das Lied von Eis und Feuer 1“ ausgewählt. Zusätzlich zu diesem Buchtitel wurde aus der Zeitschrift „Der Spiegel“ aus der Ausgabe 21 des Jahres 2015 die Seite 92 benutzt. Da diese Seite neben einem zweispaltigen Fließtext auch mehrere Grafiken beinhaltet, wurde diese in zwei Dokumente separiert. Dadurch wurde gewährleistet, dass Probleme mit der Layouterkennung der jeweiligen Programme nicht berücksichtigt werden müssen.

Das Buch „Game of Thrones and Philosophy: Logic Cuts Deeper than Swords“ wurde als Basis für die Erkennung des englischsprachigen Texte zugrunde gelegt. Analog zur deutschsprachiger Literatur wurden ebenfalls fünf Seiten (neun bis 13) ausgewählt. Aus der Zeitschrift „English Language Learning Magazine Contact“ aus dem Jahre 2014 wurden die Seiten drei und neun verwendet.

Da nicht alle Programme die Weiterverarbeitung eines PDF-Dokumentes beherrschen, wurden die Seiten mit einem Konvertierungsprogrammes wie zum Beispiel „Multi-Page TIFF Editor“[1] in ein adäquates Format transformiert, um so eine identische Ausgangssituation zu schaffen.

5.1.2 Testszenario

Zuerst erfolgt der Einsatz der Programme mit den Standardoptionen auf den deutschen Seiten des Buches bzw. der Zeitschrift. Da GOCR kein spezielles Sprachmodell zur Auswahl zulässt, erfolgt die Erkennung über die integrierte Engine ohne Verwendung der Datenbank. Bei Tesseract und Asprise wird das Modell für deutsche Sprache ausgewählt. Ocropus benutzt das englische Standardmodell und Google Drive ermöglicht keine Einstellungsmöglichkeiten. In den englischsprachigen Dokumenten wurden die Sprachmodelle bei Tesseract und Asprise auf Englisch geändert.

Im Anschluss der Auswertung der bereits vorhandenen Sprachpaketen wird jeweils ein neues Modell der Programme GOCR, Tesseract und Ocropy trainiert. Hierzu wird mittels Textverarbeitungsprogramm ein Dokument erstellt, welches jedes Zeichen zweimal beinhaltet. Diese Zeichen sind zusätzliches durch Leerzeichen von einander getrennt. Zuerst ist jedes Zeichen sortiert und anschließend randomisiert angeordnet. Bei GOCR wird die Erkennungs-Engine deaktiviert und die Nutzung der hinzugefügten Zeichen in der Datenbank aktiviert. Das Training bei Tesseract wird über das Programm Sunnypage realisiert, bei dem auf die Verwendung eines Wörterbuches verzichtet wird. Seitens Ocropus wird der Trainingsprozess durch 9.000 Steps beschränkt.

5.1.3 Evaluation

Im Durchschnitt der Erkennungsraten der deutschsprachigen Literatur konnten sich vor allem Google Docs und Asprise mit sehr guten CER-Werten von 1,87% bzw. 2,50% profilieren. Anzumerken ist, dass ein Großteil der Fehler seitens Google Docs auf der Löschung von dem angewinkelten Anführungszeichen rechts „»“ bzw. links „«“ beruht. Durch die Verwendung des englischen Sprachmoduls bei Ocropy wurden nur mangelhafte Werte von 23,40% erreicht. Auch die Auswertung der WER mit 76,10% kann nur schlecht mit den anderen Programmen mithalten. Im Verhältnis zur besten WER von Google Docs mit 5,36% beträgt der Fehlerfaktor von Ocropy über das 14-fache. Die Werte der CER von Tesseract (7,55%) und GOCR (25,04%) weisen ebenfalls einen deutlichen Qualitätsunterschied auf. Die Abbildung 5.1 visualisiert die erläuterte Auswertung anhand der durchschnittlichen CER und WER aller fünf Seiten.

[1] http://www.heise.de/download/multi-page-tiff-editor-1138025.html

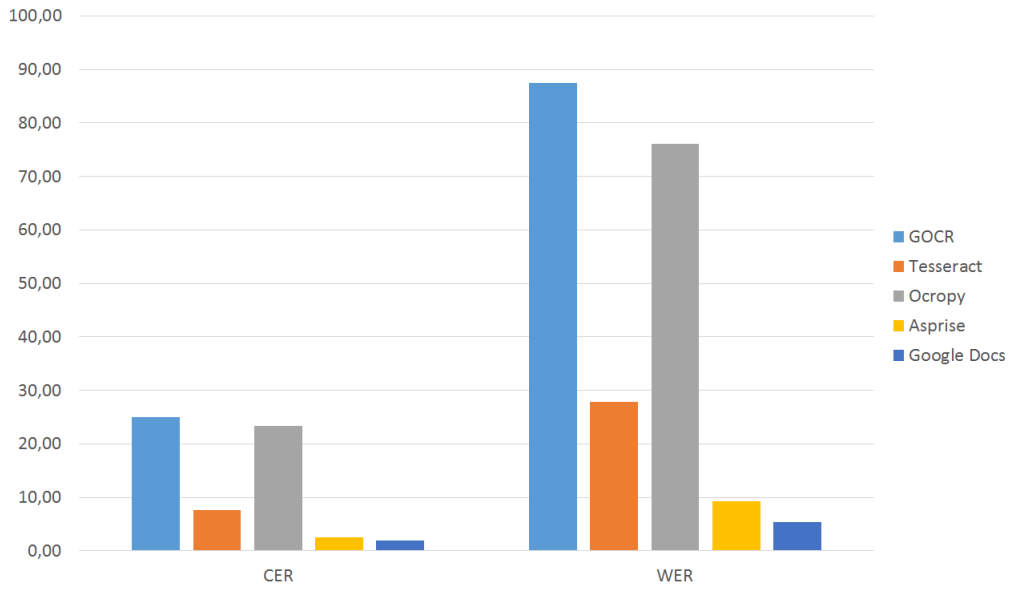


Abbildung 5.1: Visualisierung des deutschsprachigen Buches mit Standardmodellen

Bei der Zeitschrift ist besonders auffällig, dass jene Seite ohne Bilddruck von allen Programmen sehr gut interpretiert wurde. Asprise erreichte hierbei den Spitzenwert der CER mit 1,13%. Die restlichen unterliegen im Bereich bis zu 4,04% (Tesseract).

Die Folgeseite enthielt ein großes Foto innerhalb des Fließtextes, welches durch fehlerhafte Layouterkennung von GOCR teilweise als Text erkannt wurde und somit zu einer über zehnfachen CER von 35,62% im Vergleich zur ersten Seite führt. Steigern konnten sich dagegen Google Docs und Tesseract. Die CER liegt bei Google Docs bei 0,65% und beinhaltet als einzige Fehlerquelle die Löschung von Anführungszeichen und Bindestrichen. Der Wert von Tesseract ist mit 0,89% schlechter, jedoch ist die Erkennung qualitativ besser als bei Google Docs, weil zum Einem die Anführungszeichen erkannt wurden und zudem der Text so wie er gedruckt auch erkannt wurde. D.h. bei einem Zeilenbruch wurde das Wort wie im ursprünglichen Text getrennt und mittels Leerzeichen zum restlichen Wortteil erkannt. Hierbei muss man ebenfalls berücksichtigen, dass in der Ground Truth die Wörter ohne Bindestrich bei Trennung aufgeführt sind. Eine vollständige Übersicht der Einzelseiten des Buches sind in Tabelle 7.1 und für die Zeitschrift in Tabelle 7.4 ersichtlich.

Die Auswertung des englischsprachigen Buches offenbart, dass GOCR unbefriedigende Ergebnisse im Vergleich zum deutschsprachigen Buch liefert. Sowohl die CER als auch die WER steigerte sich jeweils um über das Doppelte auf 43,96% bzw. 112,58%. Trotz den englischen Sprachmoduls bei Ocropy konnte es bei einer CER von 28,98% nicht mit den Programmen Tesseract (4,43%), Asprise (2,04%)

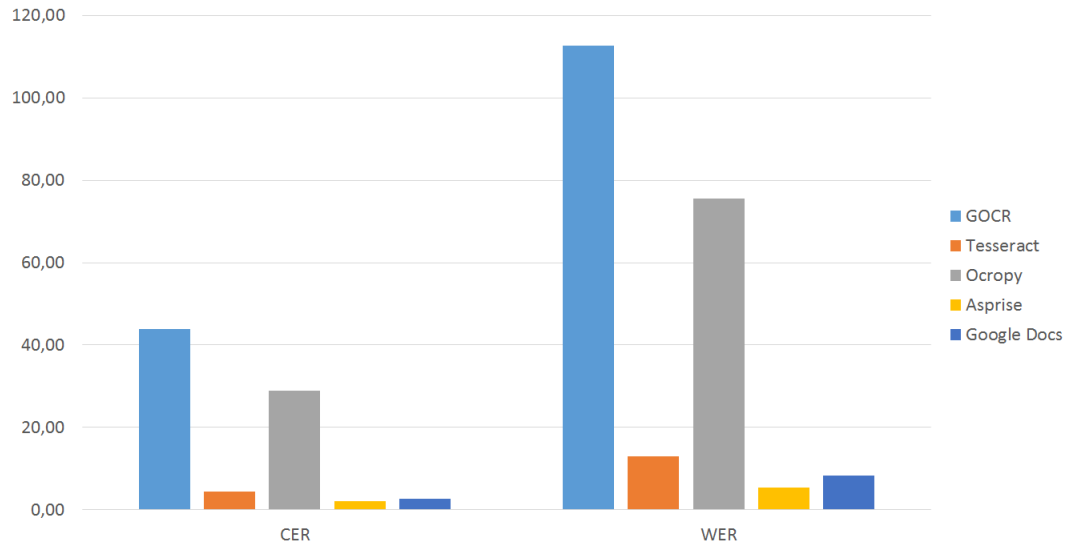


Abbildung 5.2: Visualisierung des englischsprachigen Buches mit Standardmodellen

und Google Docs (2,62%) mithalten. Die Abbildung 5.2 visualisiert die erläuterte Auswertung der Einzelseiten anhand der durchschnittlichen CER und WER.

Durch ein kleineres Bild in der Zeitschrift wurden die Layoutfehler von der CER von GOCR auf 31,05% minimiert. Allerdings stellt GOCR im Verhältnis zu den übgrigen Tools wie Asprise (1,27%) und Tesseract (3,23%) keine Alternative beim Einsatz in der englischsprachigen Literatur dar. Im Durchschnitt der zwei Testseiten erreichte Asprise mit einer CER von 1,07% das beste Resultat. Eine vollständige Übersicht der Einzelseiten des Buches sind in Tabelle 7.3 und für die Zeitschrift in Tabelle 7.4 ersichtlich.

Nachdem die Programme GOCR, Tesseract und Ocropy anhand des Trainingsdokumentes trainiert worden sind, vervielfachte sich die Fehlerrate im Vergleich zu den Standardmodellen und es konnte sich keines der Anwendungen beim kurzweiligen Training bei deutsch- und englischsprachen Dokumenten durchsetzen. Bei GOCR stieg beim deutschsprachigen Buch die CER von 25.04% auf 81,82% und bei der Zeitschrift von 19,54% auf 112,88% an. Zusätzlich ist bei Ocropy zu erwähnen, dass diese Art des Trainings nicht für den Einsatz von neuronalen Netzen empfehlenswert ist. Das OCR-Resultat beinhaltet nur durch Leerzeichen isolierte Zeichen und ist auch inhaltlich mit einer CER von 86,06% im Durchschnitt auf dem deutschsprachigen Buch unbrauchbar. Eine vollständige Übersicht der Einzelseiten des deutsch- bzw. englischsprachigen Buches sind in Tabelle 7.5 bzw. 7.7 und für die Zeitschrift in Tabelle 7.6 bzw. 7.8 ersichtlich.

5.2 Erkennung von Frakturschrift

5.2.1 Ausgangsmaterial

Basierend auf dem 1867 erschienenen illustrierten Familienblatt „Die Gartenlaube“ erfolgte eine Auswahl von fünf Seiten. Auf der Webseite[2] sind neben der digitalen Version der Buchseite auch die dazugehörige Ground Truth verfügbar. Hiervon wurden vier Seiten zum Training und eine Seite zum Evaluieren deklariert. Diese enthalten im Durchschnitt 9.795 Zeichen ohne Leerzeichen. Der Aufbau einer Seite gliedert sich in zwei separate Spalten und wurde für das Testszenario aufgeteilt, um Probleme bei der Segmentierung bzw. Layouterkennung zu minimieren.

Zusätzlich wurde zur Evaluation der vorhandenen und erzeugten Modelle eine Seite von Karl Gutzkow: Die neuen Serapionsbrüder[3] aus dem Jahre 1877 heruntergeladen und evaluiert. [3]

5.2.2 Testszenario

Zu den bereits im vorherigen deutschsprachigen Test verwendeten Standardmodellen wird nun seitens Tesseract sowohl das fränkische Modell als auch das Sprachpaket für deutsche Fraktur hinzugefügt. Bei Ocropy erfolgt die Erkennung nun mit dem Modul für deutsche Frakturschrift. Auch in der Demoversion von Asprise steht ein gesondertes Frakturmodell zur Verfügung und wird neben dem deutschen Modul evaluiert.

Für das Training neuer Sprachmodule für Tesseract und Ocropy wurde der menschliche Workload als Kriterium gesetzt. GOCR konnte leider aufgrund der miserablen Segmentierung der Zeichen nicht berücksichtigt werden. Die gesetzten Zeitintervalle sind von 10, 15, 30, 45, 60, 75, 90 bis 120 Minuten festgelegt. Die Anzahl der Annotationen innerhalb eines Intervalles unterscheidet sich zwischen den Anwendungen sehr stark. Während bei Tesseract im Durchschnitt innerhalb von 15 Minuten circa 500 Zeichen mittels Sunnypage annotiert werden konnten, ermöglicht die zeilenweise Beschriftung von Ocropy mittels kopieren der Ground Truth einen Durchsatz von über 3.500 Zeichen in der selben Zeit. Aufgrund einer frühzeitigen Stagnierung der Evaluationswerte wurden für Tesseract noch die Zeitintervalle 0, 1, 2, 3, 150 und 180 hinzugefügt.

Für jedes Zeitintervall wurden für Ocropy 30.000 Steps angesetzt und eine aktuelle Sicherung des Fortschrittes während des Trainings in 1.000er Schritten erstellt. Die Trainingsdauer nach der Annotation ist von mehreren Faktoren abhängig wie z.B. von der Hardware des Computers. Für eine Modellberechnung mit 30.000 Steps wurden circa 15 Stunden benötigt, wobei währendessen keine

[2] https://de.wikisource.org/wiki/Index:Die_Gartenlaube_(1867)
[3] http://www.deutschestextarchiv.de/book/view/gutzkow_serapionsbrueder02_1877?p=9

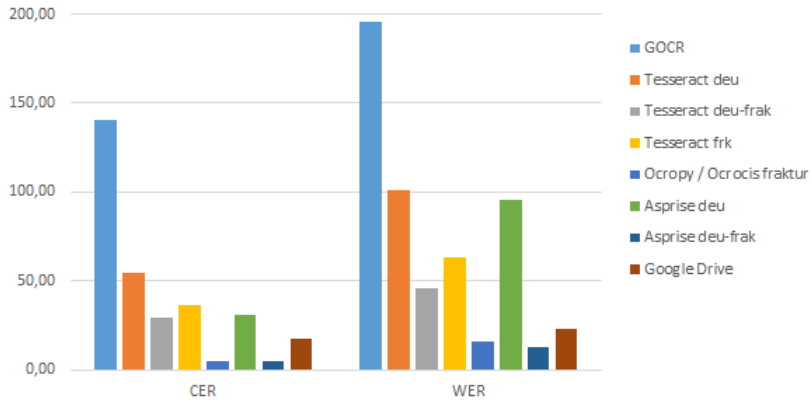


Abbildung 5.3: Visualisierung der Standardmodelle auf der Gartenlaube

weiteren Eingaben seitens des Benutzers erforderlich waren. Aus diesen 30 erzeugten Modellen jedes Intervalles wurde das beste Modell als Vertreter des jeweiligen Zeitabschnittes anhand der Auswertung der Testseiten ermittelt. Im Anschluss sollen die ermittelten Werte von Tesseract und Ocropy miteinander verglichen werden.

Weiterhin werden sämtliche in diesem Test verwendeten und erzeugten Modelle auf zwei Seiten von Karl Gutzkows: Die neuen Serapionsbrüder ausgewertet.

5.2.3 Evaluation

Nach der Auswertung der Standardmodelle zeigte sich, dass die Programme GOCR, Tesseract und Google Docs mit dem sich wechselnden Kontrast bzw. vermehrten Auftreten von Störungen gegen Abschluss der beiden Seiten vermehrt Fehler generieren als im Verhältnis zu Beginn der Seite. Im Gegensatz hierzu haben Asprise deu-frak und Ocropy fraktur wesentlich weniger Schwierigkeiten und verzeichnen im Durchschnitt die besten Werte mit einer CER von 4,48% bzw. 4,98%. Anhand der Abbildung 5.3 wird ersichtlich, dass sich GOCR als ungeeignet für die Verwendung von OCR auf Fraktur erweist.

Zwischen den einzelnen Sprachmodulen bei Tesseract sind Abstufungen der Qualität ersichtlich. Das deutsche Sprachmodell konnte sich mit einer CER von 54,67% gegenüber fränkisch (36,06%) oder deutscher Fraktur (29,41%) nicht behaupten. Auch das deutsche Sprachmodul von Asprise erreicht nahezu mit einer CER 31,19% die gleiche Qualität wie Tesseracts bestes Modell. Eine vollständige Übersicht der Auswertung der Standardmodelle ist in Tabelle 7.9 ersichtlich.

Im nächsten Schritt erfolgt die Evaluation der in Ocropy trainierten Modelle. Mittels der Abbildung 5.4 soll anhand eines Auszuges die Auswahl des besten Modelles für die Eingabe von 15 Minuten basierend auf den Testseiten suggeriert werden:

Die erste Zeile enthält die Bezeichnung des betroffenen Modelles, gefolgt von 5

1/model-00001000.pyrnn.gz		1/model-00002000.pyrnn.gz		▪▪▪	1/model-00024000.pyrnn.gz		▪▪▪	1/model-00029000.pyrnn.gz		1/model-00030000.pyrnn.gz	
errors	1592	errors	416		errors	92		errors	134	errors	115
missing	0	missing	0		missing	0		missing	0	missing	0
total	4898	total	4898		total	4898		total	4898	total	4898
err	32.503 %	err	8.493 %		err	1.878 %		err	2.736 %	err	2.348 %
errnomiss	32.503 %	errnomiss	8.493 %		errnomiss	1.878 %		errnomiss	2.736 %	errnomiss	2.348 %

Abbildung 5.4: Auszug der Modellliste nach 15 Minuten Training auf den Testseiten

Werten, welche die Qualität des OCR-Prozesses beschreibt. Die Zeile „errors“ beschreibt die Summe aller Fehler über alle Zeilen. Der Wert für „missing“ schließt alle Zeilen aus, welche zu kurz für zuverlässige Ergebnisse sind. Der Wert ist bei allen Modellen null, da im Vorfeld beim Annotieren betroffene Zeilen aus dem Testset für den Modellvergleich entfernt worden sind. Weiterhin folgt die Anzahl der Zeichen, welche herangezogen worden sind. Die Zeile „err“ gibt den prozentualen Wert für die Fehlerrate basierend auf der Anzahl aller Zeichen an. Die letzte Zeile „errnomiss“ verrechnet zusätzlich die eliminierten Zeile, welche in diesem Fall zu keiner Änderung führen. [27] Im direkten Vergleich setzte sich das Modell nach 24.000 Steps mit der niedrigsten Fehlerrate von 1.878% durch. Diese Methode wird nun für jedes Zeitintervall durchgeführt und im Anschluss erfolgt die Evaluation der Testseiten. Die Tabellen 7.10, 7.11, 7.12 und 7.13 zeigen alle Modelle der jeweiligen Zeitintervalle, wobei die Bezeichnung in diesem Testszenario die Anzahl der 15-minütigen Annotierungsdauer und die Steps verdeutlicht. So suggeriert die Bezeichnung 5/model5-00003000.pyrnn.gz dem Leser, dass 75 Minuten Text annotiert wurde und 3.000 Steps durchgeführt wurden. Die Zeilen „missing“ und „errormiss“ wurden aufgrund von Redundanz und Platzersparnis aus der Übersicht entfernt.

Die Auswertung der Abbildung 5.5 verdeutlicht, dass schon nach sehr wenig Trainingsaufwand eine Stagnation der Fehlerrate seitens Tesseract eintritt. Nach bereits drei Minuten fluktuiert diese zwischen 40 und 50 Prozent. Ocropy liefert durchgehend gute Resultate bis zu einer durchschnittlichen CER von 3,99% nach 120 Minuten, welche sich mit der Abhängigket der Dauer der Eingabe nur minimal ändert. Eine vollständige Übersicht der einzelnen Werte für jedes Zeitintervall sind in Tabelle 7.14 aufgeführt.

Nutzt man nun die erstellten Modelle zur Evaluation auf den Seiten des Dokumentes „Die neuen Serapionsbrüder“ offenbart sich, dass sich die Qualität von Tesseract signifikant verbessern konnte. Die Fehlerrate fällt nach zehn Minuten im Durchschnitt auf 19,21% und erreicht nach einer Eingabedauer von 120 Minuten den Tiefpunkt bei 16,92%. Dies stellt eine Differenz von über 25% im Verhältnis zum besten Modell auf den Testseiten der Gartenlaube, welche aus dem gleichem Buch wie die Trainingsseiten stammen, dar. Das beste Modell von Ocropy hat eine Fehlerrate von 11,08% nach 90 Minuten. Die Abbildung 5.6 visualisiert, dass sich das Resultat von Ocropy auf einem Dokument eines anderen Jahrzehntes verschlechtert und minimiert so die Diskrepanz zu der Leistung von Tesseract.

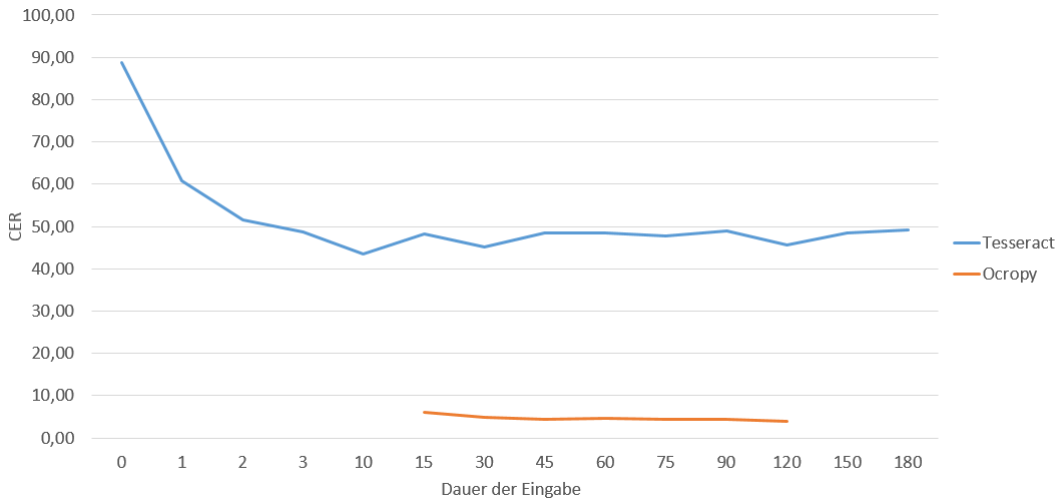


Abbildung 5.5: Modellvergleich in Minuten zwischen Tesseract und Ocropy auf der Gartenlaube

Die Tabelle 7.16 ermöglicht eine Übersicht über die einzelnen Fehlerraten der besten Modelle der Zeitintervalle.

5.3 Erkennung der lateinischen Übersetzung eines deutschsprachigen Werkes aus dem 15.Jahrhundert

5.3.1 Ausgangsmaterial

Das ursprünglich deutschsprachige Buch „Das Narrenschiff“ wurde im Jahre 1494 von Sebastianus Brant veröffentlicht und im Jahre 1497 ins lateinische übersetzt. [13] Es wurden 24 Seiten für das Trainingsset und zwei Seiten, eine mit Marginalien und eine ohne, für die Evaluation aus den von der Universität Würzburg bereitgestellten Dokumenten herangezogen. Es wurden ausschließlich Seiten ohne Bilder verwendet.

5.3.2 Testszenario

Zusätzlich zu der Auswertung der Standardmodelle wie auch im zweiten Szenario wird seitens Tesseract das Sprachmodell Latein hinzugefügt.

Die Transkriptionen, welche ebenfalls durch die Universität Würzburg bereitgestellt wurden, benötigen aufgrund von Sonderzeichen den Palemonas MUFI[4]

[4]http://folk.uib.no/hnooh/mufi/fonts/#PalemonasMUFI

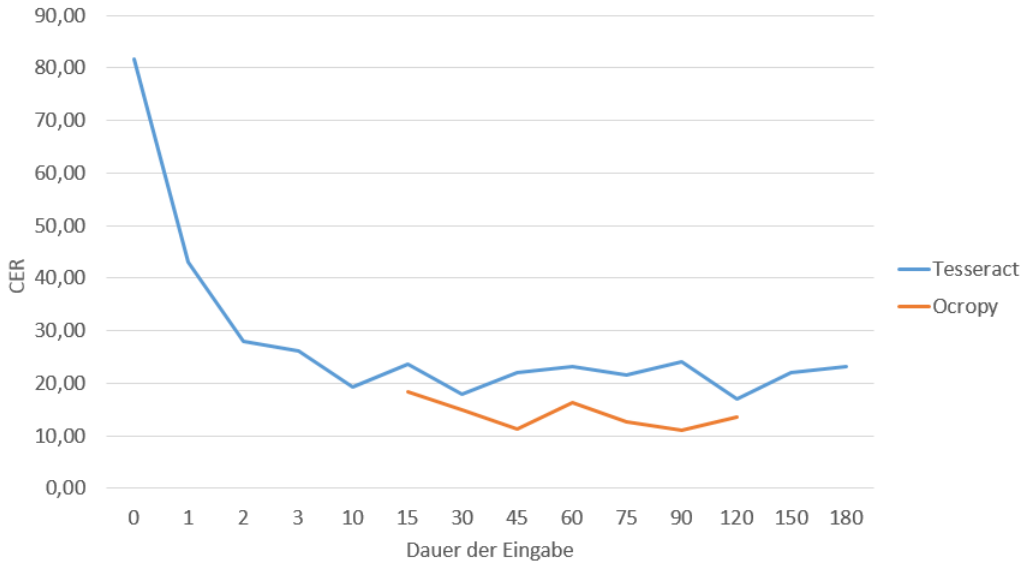


Abbildung 5.6: Modellvergleich in Minuten zwischen Tesseract und Ocropy auf den Serapionsbrüdern

Zeichensatz. Nach der Installation der Schriftart wurde jedes Sonderzeichen der Trainings- und Testseiten durch unterschiedliche, stellvertretende Standardzeichen bzw. Symbole, welche nicht im Text vorkommen, ersetzt. Weiterhin wurden eine gesonderte Ground Truth erstellt, welche diese Stellvertreter integriert hat. Durch den Austausch sollen Zeichenkodierungsprobleme vermieden und der Trainingsprozess optimiert werden.

Im Durchschnitt konnten so in 15 Minuten drei Seiten bei Ocropy annotiert werden. Bei Tesseract lässt sich keine zuverlässige Aussage treffen, da die Eingabegeschwindigkeit stark aufgrund der Segmentierung variiert. Die einzelnen Zeitintervalle wurden analog zum zweiten Testszenario gesetzt.

Bei Ocropy wurde die Anzahl der maximalen Steps auf 30.000 und die Sicherungsfrequenz auf 1.000 Steps festgelegt. Das beste Modell des jeweiligen Zeitblockes wurde ausgewählt und für die weitere Evaluation eingesetzt. Eine vollständige Übersicht der Auswertung der Modelle der jeweiligen Zeitintervalle liefern die Tabellen 7.18, 7.19, 7.20 und 7.21. Durch die präzise und vollständige Erkennung der Textzeilen in diesem Szenario konnte eine konstante Anzahl von drei Seiten innerhalb von 15 Minuten annotiert werden. Anhand der Modellbezeichnung wird die Anzahl der trainierten Seiten in Zeitraum und der Schritte beschrieben. Das Modell mit der Bezeichnung model15-00016000.pyrnn.gz zeigt auf, dass 15 Seiten trainiert und 16.000 Steps durchgeführt worden sind.

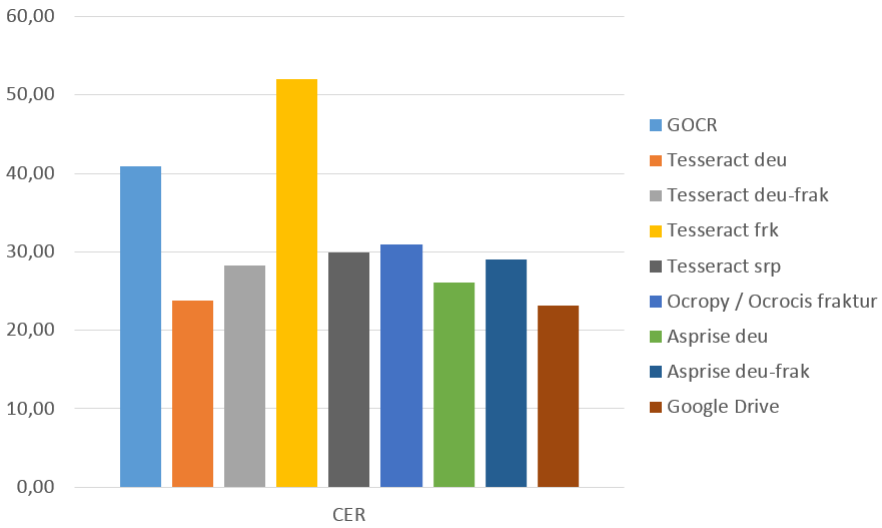


Abbildung 5.7: Visualisierung der Standardmodelle auf dem Narrenschiff

5.3.3 Evaluation

Durch die höhere Bildqualität und dem geringerem Rauschen der eingescannten Dokumente erzielte GOCR im Verhältnis zur Auswertung bei der Gartenlaube ein besseres Resultat mit einer durchschnittlichen CER von 40,85%. Bei Tesseract konnte sich das deutsche Sprachmodell mit einer CER von 23,83% gegen die Modelle deu-frak (28,22%), frk (52,05%) und dem lateinischen Modell srp (29,91%) durchsetzen. Das beste Ergebnis erzielte Google Drive mit 23,14%. Die Abbildung 5.7 verdeutlicht die Resultate der einzelnen Standardmodelle.

Weiterhin ist eine deutliche Diskrepanz zwischen den Werten der ersten Testseite ohne Marginalien und der Folgeseite mit Marginalen zu verzeichnen. Diese erscheinen in der Ground Truth innerhalb des Fließtextes auf der jeweiligen Zeilenhöhe und werden sofern erkannt von den OCR-Programmen an das Ende des kompletten Blocktextes angehängt. Eine vollständige Übersicht der Auswertung der Standardmodelle ist in Tabelle 7.17 ersichtlich.

Mit selbstständigem Training von Tesseract ist eine kontinuierliche Verbesserung innerhalb der ersten zehn Minuten der CER, wie in Abbildung 5.8 verdeutlicht wird, zu verzeichnen.

Hierbei sinkt diese von 96,05% auf 24,01% und erreicht nach den Minimalwert nach 180 Minuten bei 17,63%. Bei Ocropy ist der zeitliche Einfluss der Eingabe der trainierten Zeichen bzw. Zeilen wesentlich geringer. Nach 15 Minuten beträgt die CER 16,1%, welche sich im Verlauf stetig fallend dem Wert 14,48% nach 75 Minuten annähert.

Bei der Auswertung der ersten Seite, welche keine Marginalien enthält, konnte Ocropus eine CER von 3,36% erreichen. Jedoch muss berücksicht werden, dass unter den 40 Operationen, welche durchgeführt werden müssen um das OCR-

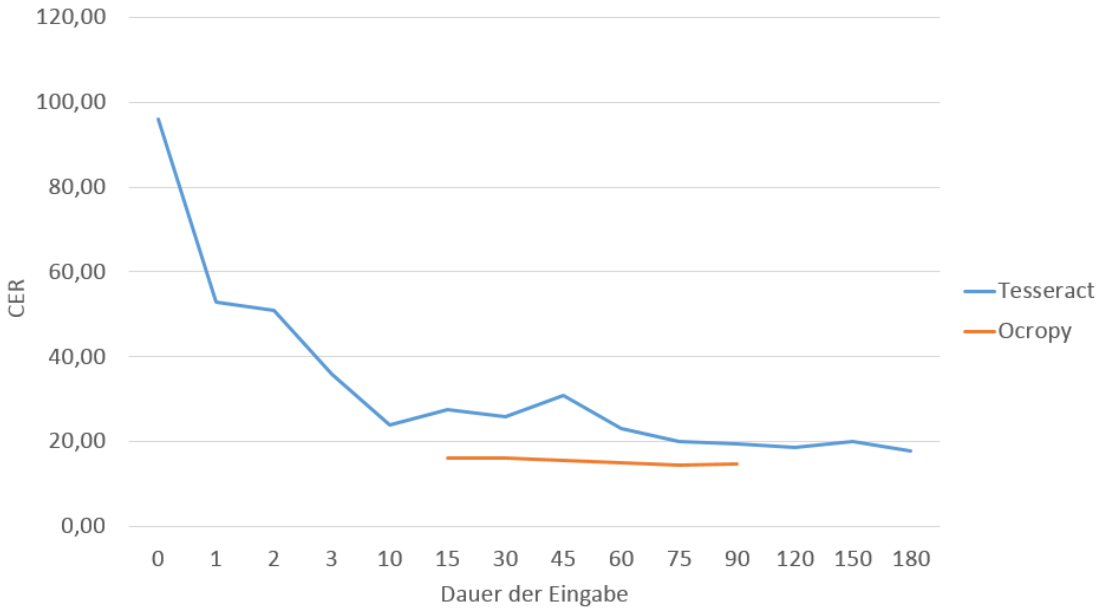


Abbildung 5.8: Modellvergleich in Minuten zwischen Tesseract und Ocropy auf dem Narrenschiff

Resultat in die Ground Truth zu wandeln, 28 Substitutionen beinhaltet, welche das einfache „ s “ in das ursprüngliche „ ſ “ konvertieren müssen. Weitere neun Fehler erzeugten das Einfügen von Leerzeichen, gehäuft vor und nach „&“. Die übrigen drei Fehler basieren auf einmaligen Substitutionen. Somit konnte sich das Neutraining sowohl gegenüber dem Standardmodell von Ocropy für Fraktur (21,24%) als auch der CER von dem besten Standardmodell von Google Drive mit 13,27% deutlich abgrenzen.

Bei Tesseract sind auf der ersten Seite über 60 Operationen nötig um die Ground Truth zu generieren, wobei mehr als ein Drittel auf das Löschen von eingefügten Leerzeichen im OCR-Text zurückzuführen sind. Die Leerzeichen wurden hauptsächlich vor bzw. nach Satzpunkten, Doppelpunkten und dem „&“-Zeichen eingefügt. Auch Tesseract konnte sich im Verhältnis zu allen Standardmodellen mit einer CER von 5,54% signifikant verbessern. Die Diskrepanz zu den programminternen Sprachmodellen liegt bei über 8%.

Betrachtet man folgend die Seite mit Marginalien steigt die Fehlerrate aller Modelle deutlich. Die Programme fügen diese, falls erkannt, am Ende des Fließtextes ein.

Durch Eigentraining bei Ocropus konnte der beste Wert von 25,37% bei dem Modell nach 120 Minuten ermittelt werden. Die Abweichung der zweiten Seite beträgt bei dem durchschnittlich besten Modell nach 75 Minuten 0,15%. Neben den zuvor erläuterten Fehlerursachen bei der Seite ohne Marginalien erscheinen diese am Ende des Textes als Einfügeoperation und in der Ground Truth als

gelöscht. Im direkten Vergleich zum Standardmodell zu Ocropy Fraktur liegt die Differenz bei über 15%. Problematisch erweist sich die Erkennung des „ſ“, welches nicht als s, sondern meist als „ l “ erkannt wird. Die Tabelle 7.22 ermöglicht eine Übersicht über die einzelnen Fehlerraten der besten Modelle der Zeitintervalle.

6 Fazit

In der Einleitung dieser Bachelorarbeit wurde darauf hingewiesen, dass die Weiterentwicklung des OCR-Prozesses und die damit verbundene Qualitätssteigerung der Erkennung eine essentielle Voraussetzung für die automatisierte Digitalisierung von historischen Texten darstellt. Nachdem die Problemstellungen der OCR-Technik näher analysiert und der typische Ablauf eines OCR-Prozesses anhand der verschiedenen Phasen erläutert wurde, erfolgte eine Erläuterung einer Möglichkeit die Fehlerrate des Erkennungsprozesses bzw. die Qualität der Anwendung mithilfe des Evaluationstools „ocrevalUation“ zu ermitteln. Folgend wird dem Leser der aktuellen Stand der OCR-Technik basierend auf Testergebnissen diverser OCR-Anwendungen aufgezeigt und einen Ausblick über nichtkommerzielle Anwendungen präsentiert. Anschließend erfolgt eine Erläuterung der Frakturschrift anhand von typischen Merkmalen und einem direkten Vergleich zu anderen gebrochenen Schriften. Eine Zielsetzung dieser Arbeit war es, präzise und kostenfreie OCR-Anwendungen auszuwählen und diese mit vorhandenen und selbst trainierten Sprachmodulen auf deutsche Frakturschrift anzuwenden. Diese Resultate sollten innerhalb von diversen Testszenarien miteinander evaluiert werden und folgend ein Ausblick auf die zukünftige Verwendung und Entwicklung des OCR-Prozesses gegeben werden.

Durch die Evaluation der Programme auf aktuellen deutsch- und englischsprachigen Dokumenten wurde gezeigt, dass ein Teil der Open-Source mit kommerziellen Anwendungen konkurrieren können. Im Rahmen des selbstständigen Trainings zeigt sich, dass GOCR große Probleme mit der Segmentierung der Zeichen auf gebrochener Schrift bzw. mit Verschmutzungen und Rauschen der Dokumente aufweist. Bei dem Eigentraining von Tesseract und Ocropy wurde ersichtlich, dass ein minimales Training mittels Generierung neuer Sprachmodelle, bereits die Fehlerrate der Standardmodelle in kürzester Zeit unterbietet. Im direkten Vergleich der beiden Programme konnte sich Ocropy bei den Tests im Einsatz für Frakturschrift durchsetzen. Sehr gute Resultate lieferte die in Google Docs integrierte OCR-Engine, welche in vielen Tests im Vergleich mit den anderen Anwendungen deutliche Qualitätsvorteile liefert.

Negativen Auswirkungen auf die Erkennungsrate wurde durch ein Testszenario deutlich, welche Dokumente mit leichter Verschmutzung und einen Farbverlauf im Hintergrund beinhalteten. Um den OCR-Prozess in Folgeprojekten weiter zu verbessern, können zusätzliche Vorverarbeitungsschritte wie das Entfernen von Rauschen und Störungen in Dokumenten als auch Nachbearbeitungsmethoden wie ein Wörterbuch eingesetzt werden, um die Fehlerraten weiter zu senken.

7 Anhang

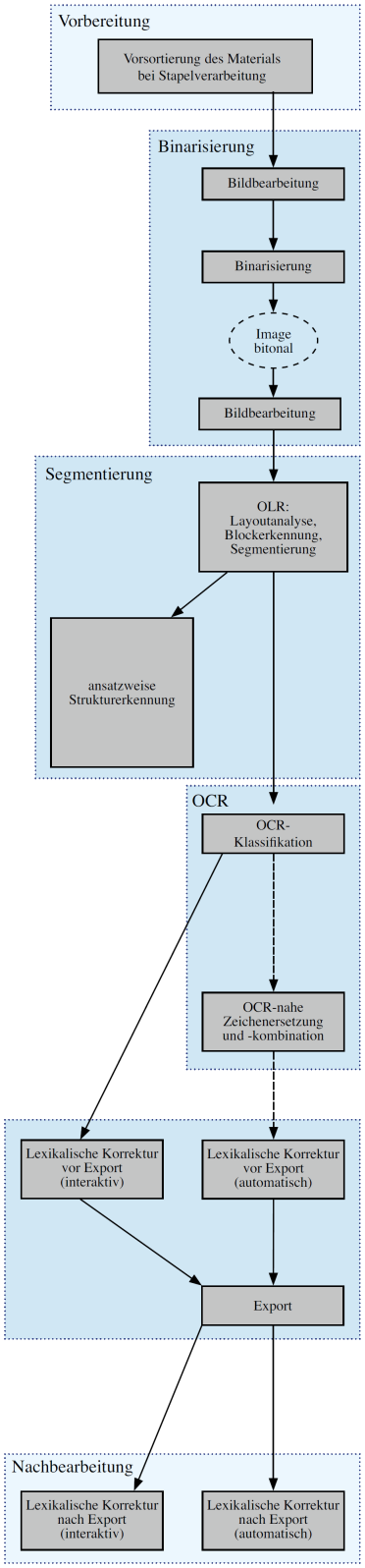


Abbildung 7.1: Typischer Workflow der OCR[20]

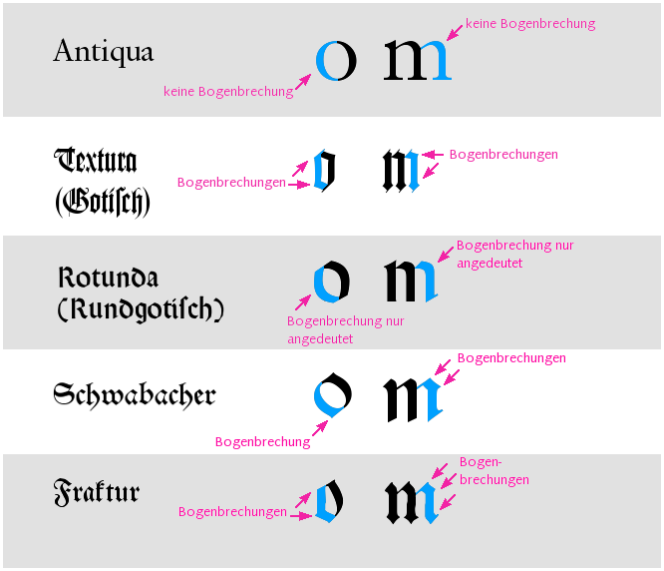


Abbildung 7.2: Differenzierung von Antiqua und gebrochenen Schriften[22]

Tabelle 7.1: Auswertung der Standardmodelle auf den deutschsprachigen Buchseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy	Asprise	Google Docs
1	CER	30,24	11,97	20,61	4,39	3,26
	WER	119,67	36,40	62,34	15,48	7,95
	WER (order independent)	118,83	35,56	60,67	15,48	7,95
2	CER	21,93	5,47	21,73	1,61	0,10
	WER	80,00	25,08	81,90	9,21	0,32
	WER (order independent)	74,92	24,76	81,27	9,21	0,32
3	CER	21,12	4,51	23,99	1,54	1,03
	WER	78,16	18,35	81,33	5,38	3,16
	WER (order independent)	72,78	18,04	80,06	5,06	3,16
4	CER	26,04	7,52	23,69	2,77	3,04
	WER	78,69	29,51	78,69	9,18	9,18
	WER (order independent)	68,85	28,85	77,05	8,20	9,18
5	CER	25,89	8,28	26,98	2,18	1,91
	WER	81,11	29,97	76,22	7,17	6,19
	WER (order independent)	74,59	29,32	69,38	6,84	6,19
Ø	CER	25,04	7,55	23,40	2,50	1,87
	WER	87,53	27,86	76,10	9,28	5,36
	WER (order independent)	81,99	27,31	73,69	8,96	5,36

Tabelle 7.2: Auswertung der Standardmodelle auf den deutschsprachigen Zeitschriftsseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy	Asprise	Google Docs
1	CER	3,45	4,04	2,79	1,13	4,39
	WER	10,21	7,29	6,88	4,79	6,25
	WER (order independent)	10,21	4,17	6,88	4,79	3,13
2	CER	35,62	0,89	7,47	4,16	0,65
	WER	31,32	5,79	16,05	11,32	2,89
	WER (order independent)	31,32	5,79	16,05	11,32	2,89
Ø	CER	19,54	2,47	5,13	2,65	2,52
	WER	20,77	6,54	11,47	8,06	4,57
	WER (order independent)	20,77	4,98	11,47	8,06	3,01

Tabelle 7.3: Auswertung der Standardmodelle auf den englischsprachigen Buchseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy	Asprise	Google Docs
1	CER	45,16	6,90	28,09	3,70	1,77
	WER	99,08	23,39	67,89	10,09	12,39
	WER (order independent)	93,12	19,27	66,06	9,17	12,39
2	CER	49,81	3,61	29,19	1,75	1,36
	WER	108,10	13,13	78,77	6,42	6,42
	WER (order independent)	105,03	11,45	75,14	5,59	6,42
3	CER	34,97	3,97	29,15	2,31	1,56
	WER	104,13	12,39	79,06	4,42	9,73
	WER (order independent)	100,29	10,03	74,93	4,13	9,73
4	CER	44,83	4,83	28,97	1,38	4,28
	WER	148,94	6,38	72,34	2,13	4,26
	WER (order independent)	144,68	6,38	72,34	2,13	4,26
5	CER	45,02	2,84	29,50	1,07	4,15
	WER	102,65	9,93	79,47	3,97	9,27
	WER (order independent)	100,66	9,27	78,81	3,31	9,27
Ø	CER	43,96	4,43	28,98	2,04	2,62
	WER	112,58	13,04	75,51	5,41	8,41
	WER (order independent)	108,76	11,28	73,46	4,87	8,41

Tabelle 7.4: Auswertung der Standardmodelle auf den englischsprachigen Zeitschriftsseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy	Asprise	Google Docs
1	CER	31,05	3,23	16,85	1,27	3,39
	WER	82,91	8,23	45,25	4,43	6,65
	WER (order independent)	75,00	6,96	42,09	4,11	6,65
2	CER	15,41	6,95	21,15	0,86	1,83
	WER	55,13	23,21	61,61	3,35	7,81
	WER (order independent)	55,13	21,43	60,49	3,35	7,81
Ø	CER	23,23	5,09	19,00	1,07	2,61
	WER	69,02	15,72	53,43	3,89	7,23
	WER (order independent)	65,07	14,20	51,29	3,73	7,23

Tabelle 7.5: Auswertung der Trainingsmodelle auf den deutschsprachigen Buchseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy
1	CER	96.60	32,01	86,90
	WER	100,00	84,52	159,83
	WER (order independent)	100,00	84,10	159,83
2	CER	78,83	30,26	89,46
	WER	98,41	80,63	174,60
	WER (order independent)	97,46	80,63	174,29
3	CER	83,39	34,85	87,39
	WER	99,37	84,49	156,96
	WER (order independent)	97,47	84,18	156,96
4	CER	82,76	33,40	87,78
	WER	99,34	86,23	155,74
	WER (order independent)	99,02	84,59	155,74
5	CER	82,29	32,81	88,77
	WER	99,02	82,74	168,08
	WER (order independent)	97,72	82,08	168,08
Ø	CER	81,82	32,67	88,06
	WER	99,23	83,72	163,04
	WER (order independent)	98,33	83,12	162,98

Tabelle 7.6: Auswertung der Trainingsmodelle auf den deutschsprachigen Zeitschriftsseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy
1	CER	88,59	14,42	90,25
	WER	100,00	44,58	188,96
	WER (order independent)	100,00	42,71	188,96
2	CER	137,16	12,48	94,88
	WER	100,00	50,53	217,11
	WER (order independent)	100,00	50,00	217,11
Ø	CER	112,88	13,45	92,57
	WER	100,00	47,56	203,04
	WER (order independent)	100,00	46,36	203,04

Tabelle 7.7: Auswertung der Trainingsmodelle auf den englischsprachigen Buchseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy
1	CER	86,65	41,63	85,11
	WER	99,68	91,74	133,03
	WER (order independent)	99,37	89,91	128,90
2	CER	82,80	43,86	86,65
	WER	99,44	101,96	145,53
	WER (order independent)	98,32	101,68	143,30
3	CER	82,71	41,11	88,99
	WER	99,41	98,53	155,46
	WER (order independent)	99,41	96,76	152,51
4	CER	86,90	40,69	87,24
	WER	100,00	108,51	157,45
	WER (order independent)	97,87	108,51	157,45
5	CER	84,72	42,42	85,90
	WER	100,00	97,35	142,38
	WER (order independent)	99,34	96,03	141,06
Ø	CER	84,76	41,94	86,78
	WER	99,71	99,62	146,77
	WER (order independent)	98,86	98,58	144,64

Tabelle 7.8: Auswertung der Trainingsmodelle auf den englischsprachigen Zeitschriftsseiten

Seite	Fehlerbezeichnung	GOCR	Tesseract	Ocropy
1	CER	92,58	30,68	85,11
	WER	99,37	72,47	138,92
	WER (order independent)	99,37	70,89	137,97
2	CER	83,93	33,52	89,94
	WER	99,78	81,25	161,16
	WER (order independent)	99,78	79,91	160,94
Ø	CER	88,26	32,10	87,53
	WER	99,58	76,86	150,04
	WER (order independent)	99,58	75,40	149,46

Tabelle 7.9: Auswertung der Standardmodelle auf der Gartenlaube

Seite	Fehlerbezeichnung	GOCR	Tesseract deu	Tesseract deu-frak	Tesseract frk	Ocropy frak	Asprise deu	Asprise deu-frak	Google Docs
1	CER	126,59	55,07	30,11	35,27	4,83	33,75	4,29	17,95
	WER	190,83	98,96	47,49	60,65	16,72	99,56	12,28	24,11
	WER (order independent)	189,94	89,50	41,42	56,21	15,68	97,93	11,54	23,67
2	CER	154,01	54,26	28,70	36,85	5,13	28,63	4,66	17,63
	WER	201,11	103,00	44,08	64,93	15,48	91,31	13,59	21,64
	WER (order independent)	199,84	99,21	38,70	60,35	14,69	90,68	13,27	20,54
Ø	CER	140,30	54,67	29,41	36,06	4,98	31,19	4,48	17,79
	WER	195,97	100,98	45,79	62,79	16,10	95,44	12,94	22,88
	WER (order independent)	194,89	94,36	40,06	58,28	15,19	94,31	12,41	22,11

Tabelle 7.10: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (1),30 (2),45 (3) und 60 (4) Minuten von 1.000 bis 15.000 Steps

1/model-00001000.pyrnn.gz		2/model2-00001000.pyrnn.gz		3/model3-00001000.pyrnn.gz		4/model4-00001000.pyrnn.gz	
errors	1592	errors	1000	errors	483	errors	1030
total	4898	total	12574	total	4898	total	4898
err	32.503 %	err	7.953 %	err	9.861 %	err	21.029 %
1/model-00002000.pyrnn.gz		**2/model2-00002000.pyrnn.gz**		**3/model3-00002000.pyrnn.gz**		**4/model4-00002000.pyrnn.gz**	
errors	416	errors	592	errors	738	errors	528
total	4898	total	12522	total	4898	total	4898
err	8.493 %	err	4.728 %	err	15.067 %	err	10.780 %
1/model-00003000.pyrnn.gz		**2/model2-00003000.pyrnn.gz**		**3/model3-00003000.pyrnn.gz**		**4/model4-00003000.pyrnn.gz**	
errors	205	errors	581	errors	311	errors	222
total	4898	total	12815	total	4898	total	4898
err	4.185 %	err	4.534 %	err	6.350 %	err	4.532 %
1/model-00004000.pyrnn.gz		**2/model2-00004000.pyrnn.gz**		**3/model3-00004000.pyrnn.gz**		**4/model4-00004000.pyrnn.gz**	
errors	545	errors	233	errors	178	errors	174
total	4898	total	13167	total	4898	total	4898
err	11.127 %	err	1.770 %	err	3.634 %	err	3.552 %
1/model-00005000.pyrnn.gz		**2/model2-00005000.pyrnn.gz**		**3/model3-00005000.pyrnn.gz**		**4/model4-00005000.pyrnn.gz**	
errors	483	errors	171	errors	148	errors	138
total	4898	total	13265	total	4898	total	4898
err	9.861 %	err	1.289 %	err	3.022 %	err	2.817 %
1/model-00006000.pyrnn.gz		**2/model2-00006000.pyrnn.gz**		**3/model3-00006000.pyrnn.gz**		**4/model4-00006000.pyrnn.gz**	
errors	715	errors	142	errors	124	errors	125
total	4898	total	13294	total	4898	total	4898
err	14.598 %	err	1.068 %	err	2.532 %	err	2.552 %
1/model-00007000.pyrnn.gz		**2/model2-00007000.pyrnn.gz**		**3/model3-00007000.pyrnn.gz**		**4/model4-00007000.pyrnn.gz**	
errors	311	errors	127	errors	125	errors	119
total	4898	total	13309	total	4898	total	4898
err	6.350 %	err	0.954 %	err	2.552 %	err	2.430 %
1/model-00008000.pyrnn.gz		**2/model2-00008000.pyrnn.gz**		**3/model3-00008000.pyrnn.gz**		**4/model4-00008000.pyrnn.gz**	
errors	175	errors	115	errors	116	errors	126
total	4898	total	13320	total	4898	total	4898
err	3.573 %	err	0.863 %	err	2.368 %	err	2.572 %
1/model-00009000.pyrnn.gz		**2/model2-00009000.pyrnn.gz**		**3/model3-00009000.pyrnn.gz**		**4/model4-00009000.pyrnn.gz**	
errors	146	errors	114	errors	132	errors	101
total	4898	total	13317	total	4898	total	4898
err	2.981 %	err	0.856 %	err	2.695 %	err	2.062 %
1/model-00010000.pyrnn.gz		**2/model2-00010000.pyrnn.gz**		**3/model3-00010000.pyrnn.gz**		**4/model4-00010000.pyrnn.gz**	
errors	130	errors	141	errors	117	errors	102
total	4898	total	13325	total	4898	total	4898
err	2.654 %	err	1.058 %	err	2.389 %	err	2.082 %
1/model-00011000.pyrnn.gz		**2/model2-00011000.pyrnn.gz**		**3/model3-00011000.pyrnn.gz**		**4/model4-00011000.pyrnn.gz**	
errors	123	errors	99	errors	111	errors	109
total	4898	total	13351	total	4898	total	4898
err	2.511 %	err	0.742 %	err	2.266 %	err	2.225 %
1/model-00012000.pyrnn.gz		**2/model2-00012000.pyrnn.gz**		**3/model3-00012000.pyrnn.gz**		**4/model4-00012000.pyrnn.gz**	
errors	116	errors	101	errors	104	errors	93
total	4898	total	13337	total	4898	total	4898
err	2.368 %	err	0.757 %	err	2.123 %	err	1.899 %
1/model-00013000.pyrnn.gz		**2/model2-00013000.pyrnn.gz**		**3/model3-00013000.pyrnn.gz**		**4/model4-00013000.pyrnn.gz**	
errors	124	errors	98	errors	95	errors	115
total	4898	total	13352	total	4898	total	4898
err	2.532 %	err	0.734 %	err	1.940 %	err	2.348 %
1/model-00014000.pyrnn.gz		**2/model2-00014000.pyrnn.gz**		**3/model3-00014000.pyrnn.gz**		**4/model4-00014000.pyrnn.gz**	
errors	113	errors	94	errors	110	errors	86
total	4898	total	13351	total	4898	total	4898
err	2.307 %	err	0.704 %	err	2.246 %	err	1.756 %
1/model-00015000.pyrnn.gz		**2/model2-00015000.pyrnn.gz**		**3/model3-00015000.pyrnn.gz**		**4/model4-00015000.pyrnn.gz**	
errors	116	errors	106	errors	111	errors	106
total	4898	total	13332	total	4898	total	4898
err	2.368 %	err	0.795 %	err	2.266 %	err	2.164 %

Tabelle 7.11: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (5),90 (6),105 (7) und 120 (8) Minuten von 1.000 bis 15.000 Steps

5/model5-00001000.pyrnn.gz		6/model6-00001000.pyrnn.gz		7/model7-00001000.pyrnn.gz		8/model8-00001000.pyrnn.gz	
errors	706	errors	710	errors	503	errors	406
total	4898	total	4898	total	4898	total	4898
err	14.414 %	err	14.496 %	err	10.269 %	err	8.289 %
5/model5-00002000.pyrnn.gz		**6/model6-00002000.pyrnn.gz**		**7/model7-00002000.pyrnn.gz**		**8/model8-00002000.pyrnn.gz**	
errors	272	errors	266	errors	217	errors	198
total	4898	total	4898	total	4898	total	4898
err	5.553 %	err	5.431 %	err	4.430 %	err	4.042 %
5/model5-00003000.pyrnn.gz		**6/model6-00003000.pyrnn.gz**		**7/model7-00003000.pyrnn.gz**		**8/model8-00003000.pyrnn.gz**	
errors	177	errors	182	errors	174	errors	148
total	4898	total	4898	total	4898	total	4898
err	3.614 %	err	3.716 %	err	3.552 %	err	3.022 %
5/model5-00004000.pyrnn.gz		**6/model6-00004000.pyrnn.gz**		**7/model7-00004000.pyrnn.gz**		**8/model8-00004000.pyrnn.gz**	
errors	161	errors	150	errors	135	errors	143
total	4898	total	4898	total	4898	total	4898
err	3.287 %	err	3.062 %	err	2.756 %	err	2.920 %
5/model5-00005000.pyrnn.gz		**6/model6-00005000.pyrnn.gz**		**7/model7-00005000.pyrnn.gz**		**8/model8-00005000.pyrnn.gz**	
errors	125	errors	129	errors	122	errors	133
total	4898	total	4898	total	4898	total	4898
err	2.552 %	err	2.634 %	err	2.491 %	err	2.715 %
5/model5-00006000.pyrnn.gz		**6/model6-00006000.pyrnn.gz**		**7/model7-00006000.pyrnn.gz**		**8/model8-00006000.pyrnn.gz**	
errors	126	errors	106	errors	113	errors	119
total	4898	total	4898	total	4898	total	4898
err	2.572 %	err	2.164 %	err	2.307 %	err	2.430 %
5/model5-00007000.pyrnn.gz		**6/model6-00007000.pyrnn.gz**		**7/model7-00007000.pyrnn.gz**		**8/model8-00007000.pyrnn.gz**	
errors	121	errors	99	errors	129	errors	130
total	4898	total	4898	total	4898	total	4898
err	2.470 %	err	2.021 %	err	2.634 %	err	2.654 %
5/model5-00008000.pyrnn.gz		**6/model6-00008000.pyrnn.gz**		**7/model7-00008000.pyrnn.gz**		**8/model8-00008000.pyrnn.gz**	
errors	130	errors	128	errors	106	errors	108
total	4898	total	4898	total	4898	total	4898
err	2.654 %	err	2.613 %	err	2.164 %	err	2.205 %
5/model5-00009000.pyrnn.gz		**6/model6-00009000.pyrnn.gz**		**7/model7-00009000.pyrnn.gz**		**8/model8-00009000.pyrnn.gz**	
errors	121	errors	87	errors	106	errors	106
total	4898	total	4898	total	4898	total	4898
err	2.470 %	err	1.776 %	err	2.164 %	err	2.164 %
5/model5-00010000.pyrnn.gz		**6/model6-00010000.pyrnn.gz**		**7/model7-00010000.pyrnn.gz**		**8/model8-00010000.pyrnn.gz**	
errors	113	errors	95	errors	116	errors	98
total	4898	total	4898	total	4898	total	4898
err	2.307 %	err	1.940 %	err	2.368 %	err	2.001 %
5/model5-00011000.pyrnn.gz		**6/model6-00011000.pyrnn.gz**		**7/model7-00011000.pyrnn.gz**		**8/model8-00011000.pyrnn.gz**	
errors	104	errors	90	errors	97	errors	98
total	4898	total	4898	total	4898	total	4898
err	2.123 %	err	1.837 %	err	1.980 %	err	2.001 %
5/model5-00012000.pyrnn.gz		**6/model6-00012000.pyrnn.gz**		**7/model7-00012000.pyrnn.gz**		**8/model8-00012000.pyrnn.gz**	
errors	97	errors	104	errors	133	errors	97
total	4898	total	4898	total	4898	total	4898
err	1.980 %	err	2.123 %	err	2.715 %	err	1.980 %
5/model5-00013000.pyrnn.gz		**6/model6-00013000.pyrnn.gz**		**7/model7-00013000.pyrnn.gz**		**8/model8-00013000.pyrnn.gz**	
errors	112	errors	85	errors	89	errors	87
total	4898	total	4898	total	4898	total	4898
err	2.287 %	err	1.735 %	err	1.817 %	err	1.776 %
5/model5-00014000.pyrnn.gz		**6/model6-00014000.pyrnn.gz**		**7/model7-00014000.pyrnn.gz**		**8/model8-00014000.pyrnn.gz**	
errors	109	errors	87	errors	113	errors	117
total	4898	total	4898	total	4898	total	4898
err	2.225 %	err	1.776 %	err	2.307 %	err	2.389 %
5/model5-00015000.pyrnn.gz		**6/model6-00015000.pyrnn.gz**		**7/model7-00015000.pyrnn.gz**		**8/model8-00015000.pyrnn.gz**	
errors	93	errors	81	errors	99	errors	110
total	4898	total	4898	total	4898	total	4898
err	1.899 %	err	1.654 %	err	2.021 %	err	2.246 %

Tabelle 7.12: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (1),30 (2),45 (3) und 60 (4) Minuten von 16.000 bis 30.000 Steps

1/model-00016000.pyrnn.gz		2/model2-00016000.pyrnn.gz		3/model3-00016000.pyrnn.gz		4/model4-00016000.pyrnn.gz	
errors	103	errors	104	errors	89	errors	95
total	4898	total	13352	total	4898	total	4898
err	2.103 %	err	0.779 %	err	1.817 %	err	1.940 %
1/model-00017000.pyrnn.gz		**2/model2-00017000.pyrnn.gz**		**3/model3-00017000.pyrnn.gz**		**4/model4-00017000.pyrnn.gz**	
errors	94	errors	89	errors	115	errors	100
total	4898	total	13353	total	4898	total	4898
err	1.919 %	err	0.667 %	err	2.348 %	err	2.042 %
1/model-00018000.pyrnn.gz		**2/model2-00018000.pyrnn.gz**		**3/model3-00018000.pyrnn.gz**		**4/model4-00018000.pyrnn.gz**	
errors	115	errors	104	errors	103	errors	85
total	4898	total	13356	total	4898	total	4898
err	2.348 %	err	0.779 %	err	2.103 %	err	1.735 %
1/model-00019000.pyrnn.gz		**2/model2-00019000.pyrnn.gz**		**3/model3-00019000.pyrnn.gz**		**4/model4-00019000.pyrnn.gz**	
errors	111	errors	94	errors	101	errors	83
total	4898	total	13371	total	4898	total	4898
err	2.266 %	err	0.703 %	err	2.062 %	err	1.695 %
1/model-00020000.pyrnn.gz		**2/model2-00020000.pyrnn.gz**		**3/model3-00020000.pyrnn.gz**		**4/model4-00020000.pyrnn.gz**	
errors	97	errors	93	errors	95	errors	78
total	4898	total	13359	total	4898	total	4898
err	1.980 %	err	0.696 %	err	1.940 %	err	1.592 %
1/model-00021000.pyrnn.gz		**2/model2-00021000.pyrnn.gz**		**3/model3-00021000.pyrnn.gz**		**4/model4-00021000.pyrnn.gz**	
errors	116	errors	96	errors	102	errors	96
total	4898	total	13363	total	4898	total	4898
err	2.368 %	err	0.718 %	err	2.082 %	err	1.960 %
1/model-00022000.pyrnn.gz		**2/model2-00022000.pyrnn.gz**		**3/model3-00022000.pyrnn.gz**		**4/model4-00022000.pyrnn.gz**	
errors	108	errors	90	errors	91	errors	95
total	4898	total	13374	total	4898	total	4898
err	2.205 %	err	0.673 %	err	1.858 %	err	1.940 %
1/model-00023000.pyrnn.gz		**2/model2-00023000.pyrnn.gz**		**3/model3-00023000.pyrnn.gz**		**4/model4-00023000.pyrnn.gz**	
errors	106	errors	96	errors	103	errors	95
total	4898	total	13370	total	4898	total	4898
err	2.164 %	err	0.718 %	err	2.103 %	err	1.940 %
1/model-00024000.pyrnn.gz		**2/model2-00024000.pyrnn.gz**		**3/model3-00024000.pyrnn.gz**		**4/model4-00024000.pyrnn.gz**	
errors	92	errors	86	errors	102	errors	99
total	4898	total	13363	total	4898	total	4898
err	1.878 %	err	0.644 %	err	2.082 %	err	2.021 %
1/model-00025000.pyrnn.gz		**2/model2-00025000.pyrnn.gz**		**3/model3-00025000.pyrnn.gz**		**4/model4-00025000.pyrnn.gz**	
errors	94	errors	114	errors	125	errors	105
total	4898	total	13386	total	4898	total	4898
err	1.919 %	err	0.852 %	err	2.552 %	err	2.144 %
1/model-00026000.pyrnn.gz		**2/model2-00026000.pyrnn.gz**		**3/model3-00026000.pyrnn.gz**		**4/model4-00026000.pyrnn.gz**	
errors	95	errors	109	errors	99	errors	84
total	4898	total	13362	total	4898	total	4898
err	1.940 %	err	0.816 %	err	2.021 %	err	1.715 %
1/model-00027000.pyrnn.gz		**2/model2-00027000.pyrnn.gz**		**3/model3-00027000.pyrnn.gz**		**4/model4-00027000.pyrnn.gz**	
errors	110	errors	102	errors	88	errors	97
total	4898	total	13374	total	4898	total	4898
err	2.246 %	err	0.763 %	err	1.797 %	err	1.980 %
1/model-00028000.pyrnn.gz		**2/model2-00028000.pyrnn.gz**		**3/model3-00028000.pyrnn.gz**		**4/model4-00028000.pyrnn.gz**	
errors	121	errors	92	errors	88	errors	92
total	4898	total	13369	total	4898	total	4898
err	2.470 %	err	0.688 %	err	1.797 %	err	1.878 %
1/model-00029000.pyrnn.gz		**2/model2-00029000.pyrnn.gz**		**3/model3-00029000.pyrnn.gz**		**4/model4-00029000.pyrnn.gz**	
errors	134	errors	86	errors	89	errors	84
total	4898	total	13358	total	4898	total	4898
err	2.736 %	err	0.644 %	err	1.817 %	err	1.715 %
1/model-00030000.pyrnn.gz		**2/model2-00030000.pyrnn.gz**		**3/model3-00030000.pyrnn.gz**		**4/model4-00030000.pyrnn.gz**	
errors	115	errors	96	errors	99	errors	95
total	4898	total	13370	total	4898	total	4898
err	2.348 %	err	0.718 %	err	2.021 %	err	1.940 %

Anzahl trainierter Seiten	1	2	3	4
Bestes Modell	1.878 %	0.644 %	1.797 %	1.592 %
Steps in 1.000	24	29,24	27,28	20

Tabelle 7.13: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (5),90 (6),105 (7) und 120 (8) Minuten von 16.000 bis 30.000 Steps

5/model5-00016000.pyrnn.gz		6/model6-00016000.pyrnn.gz		7/model7-00016000.pyrnn.gz		8/model8-00016000.pyrnn.gz	
errors	112	errors	96	errors	115	errors	102
total	4898	total	4898	total	4898	total	4898
err	2.287 %	err	1.960 %	err	2.348 %	err	2.082 %
5/model5-00017000.pyrnn.gz		**6/model6-00017000.pyrnn.gz**		**7/model7-00017000.pyrnn.gz**		**8/model8-00017000.pyrnn.gz**	
errors	108	errors	80	errors	102	errors	91
total	4898	total	4898	total	4898	total	4898
err	2.205 %	err	1.633 %	err	2.082 %	err	1.858 %
5/model5-00018000.pyrnn.gz		**6/model6-00018000.pyrnn.gz**		**7/model7-00018000.pyrnn.gz**		**8/model8-00018000.pyrnn.gz**	
errors	103	errors	82	errors	94	errors	95
total	4898	total	4898	total	4898	total	4898
err	2.103 %	err	1.674 %	err	1.919 %	err	1.940 %
5/model5-00019000.pyrnn.gz		**6/model6-00019000.pyrnn.gz**		**7/model7-00019000.pyrnn.gz**		**8/model8-00019000.pyrnn.gz**	
errors	93	errors	75	errors	93	errors	93
total	4898	total	4898	total	4898	total	4898
err	1.899 %	err	1.531 %	err	1.899 %	err	1.899 %
5/model5-00020000.pyrnn.gz		**6/model6-00020000.pyrnn.gz**		**7/model7-00020000.pyrnn.gz**		**8/model8-00020000.pyrnn.gz**	
errors	99	errors	86	errors	108	errors	110
total	4898	total	4898	total	4898	total	4898
err	2.021 %	err	1.756 %	err	2.205 %	err	2.246 %
5/model5-00021000.pyrnn.gz		**6/model6-00021000.pyrnn.gz**		**7/model7-00021000.pyrnn.gz**		**8/model8-00021000.pyrnn.gz**	
errors	95	errors	79	errors	111	errors	107
total	4898	total	4898	total	4898	total	4898
err	1.940 %	err	1.613 %	err	2.266 %	err	2.185 %
5/model5-00022000.pyrnn.gz		**6/model6-00022000.pyrnn.gz**		**7/model7-00022000.pyrnn.gz**		**8/model8-00022000.pyrnn.gz**	
errors	107	errors	83	errors	103	errors	109
total	4898	total	4898	total	4898	total	4898
err	2.185 %	err	1.695 %	err	2.103 %	err	2.225 %
5/model5-00023000.pyrnn.gz		**6/model6-00023000.pyrnn.gz**		**7/model7-00023000.pyrnn.gz**		**8/model8-00023000.pyrnn.gz**	
errors	118	errors	86	errors	121	errors	113
total	4898	total	4898	total	4898	total	4898
err	2.409 %	err	1.756 %	err	2.470 %	err	2.307 %
5/model5-00024000.pyrnn.gz		**6/model6-00024000.pyrnn.gz**		**7/model7-00024000.pyrnn.gz**		**8/model8-00024000.pyrnn.gz**	
errors	133	errors	91	errors	133	errors	108
total	4898	total	4898	total	4898	total	4898
err	2.715 %	err	1.858 %	err	2.715 %	err	2.205 %
5/model5-00025000.pyrnn.gz		**6/model6-00025000.pyrnn.gz**		**7/model7-00025000.pyrnn.gz**		**8/model8-00025000.pyrnn.gz**	
errors	116	errors	90	errors	93	errors	90
total	4898	total	4898	total	4898	total	4898
err	2.368 %	err	1.837 %	err	1.899 %	err	1.837 %
5/model5-00026000.pyrnn.gz		**6/model6-00026000.pyrnn.gz**		**7/model7-00026000.pyrnn.gz**		**8/model8-00026000.pyrnn.gz**	
errors	96	errors	76	errors	101	errors	79
total	4898	total	4898	total	4898	total	4898
err	1.960 %	err	1.552 %	err	2.062 %	err	1.613 %
5/model5-00027000.pyrnn.gz		**6/model6-00027000.pyrnn.gz**		**7/model7-00027000.pyrnn.gz**		**8/model8-00027000.pyrnn.gz**	
errors	88	errors	82	errors	98	errors	85
total	4898	total	4898	total	4898	total	4898
err	1.797 %	err	1.674 %	err	2.001 %	err	1.735 %
5/model5-00028000.pyrnn.gz		**6/model6-00028000.pyrnn.gz**		**7/model7-00028000.pyrnn.gz**		**8/model8-00028000.pyrnn.gz**	
errors	95	errors	92	errors	93	errors	92
total	4898	total	4898	total	4898	total	4898
err	1.940 %	err	1.878 %	err	1.899 %	err	1.878 %
5/model5-00029000.pyrnn.gz		**6/model6-00029000.pyrnn.gz**		**7/model7-00029000.pyrnn.gz**		**8/model8-00029000.pyrnn.gz**	
errors	92	errors	85	errors	90	errors	88
total	4898	total	4898	total	4898	total	4898
err	1.878 %	err	1.735 %	err	1.837 %	err	1.797 %
5/model5-00030000.pyrnn.gz		**6/model6-00030000.pyrnn.gz**		**7/model7-00030000.pyrnn.gz**		**8/model8-00030000.pyrnn.gz**	
errors	89	errors	73	errors	77	errors	79
total	4898	total	4898	total	4898	total	4898
err	1.817 %	err	1.490 %	err	1.572 %	err	1.613 %

Anzahl trainierter Seiten	5	6	7	8
Bestes Modell	1.797 %	1.490 %	1.572 %	1.613 %
Steps in 1.000	27	30	30	26

Tabelle 7.14: Auswertung der Trainingsmodelle unterschiedlicher Zeitintervalle auf der Gartenlaube

Dauer der Eingabe in Minuten	Seite	Fehlerbezeichnung	Tesseract	Ocropy
0	1	CER	88,66	
		WER	100,00	
		WER (order independent)	100,00	
	2	CER	88,83	
		WER	100,00	
		WER (order independent)	100,00	
1	1	CER	59,54	
		WER	88,76	
		WER (order independent)	85,50	
	2	CER	61,96	
		WER	87,36	
		WER (order independent)	84,68	
2	1	CER	50,95	
		WER	85,50	
		WER (order independent)	81,51	
	2	CER	51,98	
		WER	81,36	
		WER (order independent)	77,41	
3	1	CER	48,67	
		WER	82,10	
		WER (order independent)	78,40	
	2	CER	48,66	
		WER	78,67	
		WER (order independent)	74,72	
10	1	CER	42,74	
		WER	75,74	
		WER (order independent)	71,89	
	2	CER	44,43	
		WER	72,83	
		WER (order independent)	68,72	
15	1	CER	46,62	6,02
		WER	77,07	22,49
		WER (order independent)	72,63	21,15
	2	CER	49,75	5,99
		WER	76,46	17,85
		WER (order independent)	72,04	17,06
30	1	CER	43,73	4,87
		WER	77,07	14,64
		WER (order independent)	72,63	13,76
	2	CER	46,61	5,13
		WER	75,20	12,95
		WER (order independent)	70,14	11,85
45	1	CER	47,47	4,01
		WER	86,24	13,46
		WER (order independent)	82,10	12,28
	2	CER	49,43	4,95
		WER	81,20	13,43
		WER (order independent)	76,78	12,64
60	1	CER	46,91	4,65
		WER	82,99	14,79
		WER (order independent)	78,85	13,76
	2	CER	49,88	4,75
		WER	80,25	13,11
		WER (order independent)	77,73	12,64
75	1	CER	46,73	4,15
		WER	81,51	12,57
		WER (order independent)	76,48	11,39
	2	CER	48,86	4,68
		WER	76,30	12,01
		WER (order independent)	73,78	11,06
90	1	CER	47,41	4,06
		WER	84,91	12,72
		WER (order independent)	79,73	11,54
	2	CER	50,45	4,78
		WER	81,36	12,01
		WER (order independent)	77,73	11,06
120	1	CER	44,25	3,54
		WER	76,48	11,69
		WER (order independent)	70,71	10,80
	2	CER	46,83	4,43
		WER	75,20	11,06
		WER (order independent)	70,46	9,95
150	1	CER	47,18	
		WER	83,88	
		WER (order independent)	78,40	
	2	CER	49,60	
		WER	80,73	
		WER (order independent)	76,94	
180	1	CER	48,06	
		WER	83,14	
		WER (order independent)	77,22	
	2	CER	50,20	
		WER	81,99	
		WER (order independent)	79,78	

Tabelle 7.15: Auswertung der Standardmodelle auf den neuen Serapionsbrudern

Seite	Fehlerbezeichnung	GOCR	Tesseract deu	Tesseract deu-frak	Tesseract frk	Ocropy frak	Asprise deu	Asprise deu-frak	Google Docs
1	CER	44,58	33,39	4,71	8,17	2,66	29,04	3,82	1,78
	WER	176,33	78,70	13,02	30,18	8,28	78,70	11,83	4,73
	WER (order independent)	175,74	76,33	12,43	28,99	8,28	77,51	11,83	4,73
2	CER	41,72	32,43	2,36	9,21	1,35	30,49	2,20	1,44
	WER	184,44	83,33	7,78	35,56	6,67	82,78	8,33	1,67
	WER (order independent)	183,33	82,78	7,78	35,56	6,11	82,22	7,78	1,67
Ø	CER	43,15	32,91	3,54	8,69	2,01	29,77	3,01	1,61
	WER	180,39	81,02	10,40	32,87	7,48	80,74	10,08	3,20
	WER (order independent)	179,54	79,56	10,11	32,28	7,20	79,87	9,81	3,20

Tabelle 7.16: Auswertung der Trainingsmodelle von der Gartenlaube auf den neuen Serapionsbrudern

Dauer der Eingabe in Minuten	Seite	Fehlerbezeichnung	Tesseract	Ocropy
0	1	CER	81,35	
		WER	104,14	
		WER (order independent)	104,14	
	2	CER	81,84	
		WER	101,67	
		WER (order independent)	101,67	
1	1	CER	45,20	
		WER	89,94	
		WER (order independent)	89,35	
	2	CER	40,79	
		WER	87,22	
		WER (order independent)	86,67	
2	1	CER	27,98	
		WER	78,70	
		WER (order independent)	77,51	
	2	CER	28,13	
		WER	80,00	
		WER (order independent)	80,00	
3	1	CER	24,42	
		WER	72,78	
		WER (order independent)	71,01	
	2	CER	27,79	
		WER	79,44	
		WER (order independent)	79,44	
10	1	CER	18,65	
		WER	57,99	
		WER (order independent)	55,62	
	2	CER	19,76	
		WER	62,22	
		WER (order independent)	62,22	
15	1	CER	23,09	18,29
		WER	62,13	62,72
		WER (order independent)	60,36	61,54
	2	CER	24,16	18,41
		WER	64,44	67,22
		WER (order independent)	64,44	66,67
30	1	CER	17,23	16,52
		WER	52,66	62,72
		WER (order independent)	50,89	62,13
	2	CER	18,58	13,43
		WER	62,22	55,56
		WER (order independent)	62,22	55,00

Dauer der Eingabe in Minuten	Seite	Fehlerbezeichnung	Tesseract	Ocropy
45	1	CER	21,31	12,08
		WER	62,13	53,85
		WER (order independent)	60,95	53,25
	2	CER	22,89	10,56
		WER	71,67	46,67
		WER (order independent)	71,67	46,11
60	1	CER	20,78	16,96
		WER	64,50	66,27
		WER (order independent)	63,31	64,50
	2	CER	25,68	15,71
		WER	75,00	64,44
		WER (order independent)	75,00	63,89
75	1	CER	19,89	14,21
		WER	57,99	53,85
		WER (order independent)	56,21	53,25
	2	CER	23,40	10,98
		WER	67,22	46,11
		WER (order independent)	67,22	45,56
90	1	CER	21,67	10,83
		WER	64,50	46,15
		WER (order independent)	61,54	45,56
	2	CER	26,60	11,32
		WER	75,56	48,33
		WER (order independent)	75,56	46,67
120	1	CER	16,43	13,59
		WER	51,48	56,80
		WER (order independent)	49,70	56,21
	2	CER	17,40	13,68
		WER	58,33	61,11
		WER (order independent)	58,33	60,00
150	1	CER	21,40	
		WER	61,54	
		WER (order independent)	60,36	
	2	CER	22,80	
		WER	68,33	
		WER (order independent)	68,33	
180	1	CER	22,02	
		WER	55,03	
		WER (order independent)	53,85	
	2	CER	24,49	
		WER	67,22	
		WER (order independent)	66,67	

Tabelle 7.17: Auswertung der Standardmodelle auf dem Narrenschiff

Seite	Fehlerbezeichnung	GOCR	Tesseract deu	Tesseract deu-frak	Tesseract frk	Tesseract srp	Ocropy	Asprise deu	Asprise deu-frak	Google Docs
1	CER	33,50	14,27	20,74	44,00	20,74	21,24	17,21	20,74	13,27
	WER	119,57	53,80	65,76	111,96	68,48	75,00	59,24	62,50	53,26
	WER (order independent)	117,93	46,74	59,24	110,33	63,04	70,65	52,17	56,52	52,72
2	CER	48,20	33,38	35,70	60,10	39,07	40,57	34,88	37,35	33,01
	WER	155,61	66,34	70,24	111,71	80,00	87,32	66,83	74,63	60,00
	WER (order independent)	153,66	44,88	56,10	107,32	65,85	77,07	51,71	60,49	44,88
Ø	CER	40,85	23,83	28,22	52,05	29,91	30,91	26,05	29,05	23,14
	WER	137,59	60,07	68,00	111,84	74,24	81,16	63,04	68,57	56,63
	WER (order independent)	135,80	45,81	57,67	108,83	64,45	73,86	51,94	58,51	48,80

Tabelle 7.18: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (model3),30 (model6),45 (model9) und 60 (model12) Minuten von 1.000 bis 15.000 Steps

model3-00001000.pyrnn.gz		**model6-00001000.pyrnn.gz**		**model9-00001000.pyrnn.gz**		**model12-00001000.pyrnn.gz**	
errors	778	errors	1104	errors	1090	errors	1549
total	2446	total	2446	total	2446	total	2446
err	31.807 %	err	45.135 %	err	44.563 %	err	63.328 %
model3-00002000.pyrnn.gz		**model6-00002000.pyrnn.gz**		**model9-00002000.pyrnn.gz**		**model12-00002000.pyrnn.gz**	
errors	300	errors	331	errors	356	errors	303
total	2446	total	2446	total	2446	total	2446
err	12.265 %	err	13.532 %	err	14.554 %	err	12.388 %
model3-00003000.pyrnn.gz		**model6-00003000.pyrnn.gz**		**model9-00003000.pyrnn.gz**		**model12-00003000.pyrnn.gz**	
errors	247	errors	243	errors	277	errors	234
total	2446	total	2446	total	2446	total	2446
err	10.098 %	err	9.935 %	err	11.325 %	err	9.567 %
model3-00004000.pyrnn.gz		**model6-00004000.pyrnn.gz**		**model9-00004000.pyrnn.gz**		**model12-00004000.pyrnn.gz**	
errors	264	errors	214	errors	236	errors	198
total	2446	total	2446	total	2446	total	2446
err	10.793 %	err	8.749 %	err	9.648 %	err	8.095 %
model3-00005000.pyrnn.gz		**model6-00005000.pyrnn.gz**		**model9-00005000.pyrnn.gz**		**model12-00005000.pyrnn.gz**	
errors	240	errors	238	errors	205	errors	179
total	2446	total	2446	total	2446	total	2446
err	9.812 %	err	9.730 %	err	8.381 %	err	7.318 %
model3-00006000.pyrnn.gz		**model6-00006000.pyrnn.gz**		**model9-00006000.pyrnn.gz**		**model12-00006000.pyrnn.gz**	
errors	247	errors	187	errors	197	errors	169
total	2446	total	2446	total	2446	total	2446
err	10.098 %	err	7.645 %	err	8.054 %	err	6.909 %
model3-00007000.pyrnn.gz		**model6-00007000.pyrnn.gz**		**model9-00007000.pyrnn.gz**		**model12-00007000.pyrnn.gz**	
errors	252	errors	187	errors	197	errors	244
total	2446	total	2446	total	2446	total	2446
err	10.303 %	err	7.645 %	err	8.054 %	err	9.975 %
model3-00008000.pyrnn.gz		**model6-00008000.pyrnn.gz**		**model9-00008000.pyrnn.gz**		**model12-00008000.pyrnn.gz**	
errors	247	errors	202	errors	189	errors	171
total	2446	total	2446	total	2446	total	2446
err	10.098 %	err	8.258 %	err	7.727 %	err	6.991 %
model3-00009000.pyrnn.gz		**model6-00009000.pyrnn.gz**		**model9-00009000.pyrnn.gz**		**model12-00009000.pyrnn.gz**	
errors	224	errors	222	errors	179	errors	139
total	2446	total	2446	total	2446	total	2446
err	9.158 %	err	9.076 %	err	7.318 %	err	5.683 %
model3-00010000.pyrnn.gz		**model6-00010000.pyrnn.gz**		**model9-00010000.pyrnn.gz**		**model12-00010000.pyrnn.gz**	
errors	255	errors	178	errors	153	errors	144
total	2446	total	2446	total	2446	total	2446
err	10.425 %	err	7.277 %	err	6.255 %	err	5.887 %
model3-00011000.pyrnn.gz		**model6-00011000.pyrnn.gz**		**model9-00011000.pyrnn.gz**		**model12-00011000.pyrnn.gz**	
errors	229	errors	181	errors	181	errors	159
total	2446	total	2446	total	2446	total	2446
err	9.362 %	err	7.400 %	err	7.400 %	err	6.500 %
model3-00012000.pyrnn.gz		**model6-00012000.pyrnn.gz**		**model9-00012000.pyrnn.gz**		**model12-00012000.pyrnn.gz**	
errors	232	errors	220	errors	164	errors	159
total	2446	total	2446	total	2446	total	2446
err	9.485 %	err	8.994 %	err	6.705 %	err	6.500 %
model3-00013000.pyrnn.gz		**model6-00013000.pyrnn.gz**		**model9-00013000.pyrnn.gz**		**model12-00013000.pyrnn.gz**	
errors	225	errors	186	errors	161	errors	157
total	2446	total	2446	total	2446	total	2446
err	9.199 %	err	7.604 %	err	6.582 %	err	6.419 %
model3-00014000.pyrnn.gz		**model6-00014000.pyrnn.gz**		**model9-00014000.pyrnn.gz**		**model12-00014000.pyrnn.gz**	
errors	228	errors	178	errors	162	errors	141
total	2446	total	2446	total	2446	total	2446
err	9.321 %	err	7.277 %	err	6.623 %	err	5.765 %
model3-00015000.pyrnn.gz		**model6-00015000.pyrnn.gz**		**model9-00015000.pyrnn.gz**		**model12-00015000.pyrnn.gz**	
errors	250	errors	159	errors	155	errors	147
total	2446	total	2446	total	2446	total	2446
err	10.221 %	err	6.500 %	err	6.337 %	err	6.010 %

Tabelle 7.19: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (model15),90 (model18),105 (model21) und 120 (model24) Minuten von 1.000 bis 15.000 Steps

model15-00001000.pyrnn.gz		**model18-00001000.pyrnn.gz**		**model21-00001000.pyrnn.gz**		**model24-00001000.pyrnn.gz**	
errors	1597	errors	1269	errors	1176	errors	1292
total	2446	total	2446	total	2446	total	2446
err	65.290 %	err	51.881 %	err	48.078 %	err	52.821 %
model15-00002000.pyrnn.gz		**model18-00002000.pyrnn.gz**		**model21-00002000.pyrnn.gz**		**model24-00002000.pyrnn.gz**	
errors	317	errors	360	errors	322	errors	318
total	2446	total	2446	total	2446	total	2446
err	12.960 %	err	14.718 %	err	13.164 %	err	13.001 %
model15-00003000.pyrnn.gz		**model18-00003000.pyrnn.gz**		**model21-00003000.pyrnn.gz**		**model24-00003000.pyrnn.gz**	
errors	219	errors	239	errors	238	errors	217
total	2446	total	2446	total	2446	total	2446
err	8.953 %	err	9.771 %	err	9.730 %	err	8.872 %
model15-00004000.pyrnn.gz		**model18-00004000.pyrnn.gz**		**model21-00004000.pyrnn.gz**		**model24-00004000.pyrnn.gz**	
errors	212	errors	178	errors	187	errors	177
total	2446	total	2446	total	2446	total	2446
err	8.667 %	err	7.277 %	err	7.645 %	err	7.236 %
model15-00005000.pyrnn.gz		**model18-00005000.pyrnn.gz**		**model21-00005000.pyrnn.gz**		**model24-00005000.pyrnn.gz**	
errors	166	errors	157	errors	176	errors	159
total	2446	total	2446	total	2446	total	2446
err	6.787 %	err	6.419 %	err	7.195 %	err	6.500 %
model15-00006000.pyrnn.gz		**model18-00006000.pyrnn.gz**		**model21-00006000.pyrnn.gz**		**model24-00006000.pyrnn.gz**	
errors	169	errors	154	errors	177	errors	155
total	2446	total	2446	total	2446	total	2446
err	6.909 %	err	6.296 %	err	7.236 %	err	6.337 %
model15-00007000.pyrnn.gz		**model18-00007000.pyrnn.gz**		**model21-00007000.pyrnn.gz**		**model24-00007000.pyrnn.gz**	
errors	161	errors	130	errors	159	errors	173
total	2446	total	2446	total	2446	total	2446
err	6.582 %	err	5.315 %	err	6.500 %	err	7.073 %
model15-00008000.pyrnn.gz		**model18-00008000.pyrnn.gz**		**model21-00008000.pyrnn.gz**		**model24-00008000.pyrnn.gz**	
errors	159	errors	147	errors	175	errors	135
total	2446	total	2446	total	2446	total	2446
err	6.500 %	err	6.010 %	err	7.155 %	err	5.519 %
model15-00009000.pyrnn.gz		**model18-00009000.pyrnn.gz**		**model21-00009000.pyrnn.gz**		**model24-00009000.pyrnn.gz**	
errors	150	errors	136	errors	166	errors	141
total	2446	total	2446	total	2446	total	2446
err	6.132 %	err	5.560 %	err	6.787 %	err	5.765 %
model15-00010000.pyrnn.gz		**model18-00010000.pyrnn.gz**		**model21-00010000.pyrnn.gz**		**model24-00010000.pyrnn.gz**	
errors	218	errors	123	errors	162	errors	117
total	2446	total	2446	total	2446	total	2446
err	8.913 %	err	5.029 %	err	6.623 %	err	4.783 %
model15-00011000.pyrnn.gz		**model18-00011000.pyrnn.gz**		**model21-00011000.pyrnn.gz**		**model24-00011000.pyrnn.gz**	
errors	136	errors	120	errors	138	errors	126
total	2446	total	2446	total	2446	total	2446
err	5.560 %	err	4.906 %	err	5.642 %	err	5.151 %
model15-00012000.pyrnn.gz		**model18-00012000.pyrnn.gz**		**model21-00012000.pyrnn.gz**		**model24-00012000.pyrnn.gz**	
errors	124	errors	114	errors	132	errors	154
total	2446	total	2446	total	2446	total	2446
err	5.070 %	err	4.661 %	err	5.397 %	err	6.296 %
model15-00013000.pyrnn.gz		**model18-00013000.pyrnn.gz**		**model21-00013000.pyrnn.gz**		**model24-00013000.pyrnn.gz**	
errors	122	errors	129	errors	160	errors	125
total	2446	total	2446	total	2446	total	2446
err	4.988 %	err	5.274 %	err	6.541 %	err	5.110 %
model15-00014000.pyrnn.gz		**model18-00014000.pyrnn.gz**		**model21-00014000.pyrnn.gz**		**model24-00014000.pyrnn.gz**	
errors	124	errors	131	errors	140	errors	112
total	2446	total	2446	total	2446	total	2446
err	5.070 %	err	5.356 %	err	5.724 %	err	4.579 %
model15-00015000.pyrnn.gz		**model18-00015000.pyrnn.gz**		**model21-00015000.pyrnn.gz**		**model24-00015000.pyrnn.gz**	
errors	127	errors	131	errors	145	errors	122
total	2446	total	2446	total	2446	total	2446
err	5.192 %	err	5.356 %	err	5.928 %	err	4.988 %

Tabelle 7.20: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (model3),30 (model6),45 (model9) und 60 (model12) Minuten von 16.000 bis 30.000 Steps

model3-00016000.pyrnn.gz		**model6-00016000.pyrnn.gz**		**model9-00016000.pyrnn.gz**		**model12-00016000.pyrnn.gz**	
errors	238	errors	162	errors	149	errors	165
total	2446	total	2446	total	2446	total	2446
err	9.730 %	err	6.623 %	err	6.092 %	err	6.746 %
model3-00017000.pyrnn.gz		**model6-00017000.pyrnn.gz**		**model9-00017000.pyrnn.gz**		**model12-00017000.pyrnn.gz**	
errors	256	errors	159	errors	177	errors	164
total	2446	total	2446	total	2446	total	2446
err	10.466 %	err	6.500 %	err	7.236 %	err	6.705 %
model3-00018000.pyrnn.gz		**model6-00018000.pyrnn.gz**		**model9-00018000.pyrnn.gz**		**model12-00018000.pyrnn.gz**	
errors	312	errors	182	errors	148	errors	177
total	2446	total	2446	total	2446	total	2446
err	12.756 %	err	7.441 %	err	6.051 %	err	7.236 %
model3-00019000.pyrnn.gz		**model6-00019000.pyrnn.gz**		**model9-00019000.pyrnn.gz**		**model12-00019000.pyrnn.gz**	
errors	238	errors	176	errors	145	errors	170
total	2446	total	2446	total	2446	total	2446
err	9.730 %	err	7.195 %	err	5.928 %	err	6.950 %
model3-00020000.pyrnn.gz		**model6-00020000.pyrnn.gz**		**model9-00020000.pyrnn.gz**		**model12-00020000.pyrnn.gz**	
errors	231	errors	173	errors	152	errors	154
total	2446	total	2446	total	2446	total	2446
err	9.444 %	err	7.073 %	err	6.214 %	err	6.296 %
model3-00021000.pyrnn.gz		**model6-00021000.pyrnn.gz**		**model9-00021000.pyrnn.gz**		**model12-00021000.pyrnn.gz**	
errors	234	errors	180	errors	158	errors	155
total	2446	total	2446	total	2446	total	2446
err	9.567 %	err	7.359 %	err	6.460 %	err	6.337 %
model3-00022000.pyrnn.gz		**model6-00022000.pyrnn.gz**		**model9-00022000.pyrnn.gz**		**model12-00022000.pyrnn.gz**	
errors	249	errors	170	errors	158	errors	167
total	2446	total	2446	total	2446	total	2446
err	10.180 %	err	6.950 %	err	6.460 %	err	6.827 %
model3-00023000.pyrnn.gz		**model6-00023000.pyrnn.gz**		**model9-00023000.pyrnn.gz**		**model12-00023000.pyrnn.gz**	
errors	243	errors	193	errors	146	errors	148
total	2446	total	2446	total	2446	total	2446
err	9.935 %	err	7.890 %	err	5.969 %	err	6.051 %
model3-00024000.pyrnn.gz		**model6-00024000.pyrnn.gz**		**model9-00024000.pyrnn.gz**		**model12-00024000.pyrnn.gz**	
errors	224	errors	178	errors	147	errors	163
total	2446	total	2446	total	2446	total	2446
err	9.158 %	err	7.277 %	err	6.010 %	err	6.664 %
model3-00025000.pyrnn.gz		**model6-00025000.pyrnn.gz**		**model9-00025000.pyrnn.gz**		**model12-00025000.pyrnn.gz**	
errors	229	errors	174	errors	144	errors	145
total	2446	total	2446	total	2446	total	2446
err	9.362 %	err	7.114 %	err	5.887 %	err	5.928 %
model3-00026000.pyrnn.gz		**model6-00026000.pyrnn.gz**		**model9-00026000.pyrnn.gz**		**model12-00026000.pyrnn.gz**	
errors	214	errors	180	errors	155	errors	156
total	2446	total	2446	total	2446	total	2446
err	8.749 %	err	7.359 %	err	6.337 %	err	6.378 %
model3-00027000.pyrnn.gz		**model6-00027000.pyrnn.gz**		**model9-00027000.pyrnn.gz**		**model12-00027000.pyrnn.gz**	
errors	247	errors	174	errors	158	errors	147
total	2446	total	2446	total	2446	total	2446
err	10.098 %	err	7.114 %	err	6.460 %	err	6.010 %
model3-00028000.pyrnn.gz		**model6-00028000.pyrnn.gz**		**model9-00028000.pyrnn.gz**		**model12-00028000.pyrnn.gz**	
errors	246	errors	184	errors	159	errors	1640
total	2446	total	2446	total	2446	total	2446
err	10.057 %	err	7.522 %	err	6.500 %	err	67.048 %
model3-00029000.pyrnn.gz		**model6-00029000.pyrnn.gz**		**model9-00029000.pyrnn.gz**		**model12-00029000.pyrnn.gz**	
errors	242	errors	175	errors	144	errors	181
total	2446	total	2446	total	2446	total	2446
err	9.894 %	err	7.155 %	err	5.887 %	err	7.400 %
model3-00030000.pyrnn.gz		**model6-00030000.pyrnn.gz**		**model9-00030000.pyrnn.gz**		**model12-00030000.pyrnn.gz**	
errors	235	errors	168	errors	150	errors	142
total	2446	total	2446	total	2446	total	2446
err	9.608 %	err	6.868 %	err	6.132 %	err	5.805 %

Anzahl trainierter Seiten	3	6	9	12
Bestes Modell	8.749 %	6.500 %	5.887 %	5.683 %
Steps in 1.000	26	15,17	25,29	9

Tabelle 7.21: Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (model3),90 (model6),105 (model9) und 120 (model12) Minuten von 16.000 bis 30.000 Steps

model15-00016000.pyrnn.gz		**model18-00016000.pyrnn.gz**		**model21-00016000.pyrnn.gz**		**model24-00016000.pyrnn.gz**	
errors	131	errors	132	errors	135	errors	111
total	2446	total	2446	total	2446	total	2446
err	5.356 %	err	5.397 %	err	5.519 %	err	4.538 %
model15-00017000.pyrnn.gz		**model18-00017000.pyrnn.gz**		**model21-00017000.pyrnn.gz**		**model24-00017000.pyrnn.gz**	
errors	137	errors	130	errors	162	errors	122
total	2446	total	2446	total	2446	total	2446
err	5.601 %	err	5.315 %	err	6.623 %	err	4.988 %
model15-00018000.pyrnn.gz		**model18-00018000.pyrnn.gz**		**model21-00018000.pyrnn.gz**		**model24-00018000.pyrnn.gz**	
errors	123	errors	158	errors	157	errors	127
total	2446	total	2446	total	2446	total	2446
err	5.029 %	err	6.460 %	err	6.419 %	err	5.192 %
model15-00019000.pyrnn.gz		**model18-00019000.pyrnn.gz**		**model21-00019000.pyrnn.gz**		**model24-00019000.pyrnn.gz**	
errors	133	errors	133	errors	132	errors	109
total	2446	total	2446	total	2446	total	2446
err	5.437 %	err	5.437 %	err	5.397 %	err	4.456 %
model15-00020000.pyrnn.gz		**model18-00020000.pyrnn.gz**		**model21-00020000.pyrnn.gz**		**model24-00020000.pyrnn.gz**	
errors	126	errors	120	errors	146	errors	124
total	2446	total	2446	total	2446	total	2446
err	5.151 %	err	4.906 %	err	5.969 %	err	5.070 %
model15-00021000.pyrnn.gz		**model18-00021000.pyrnn.gz**		**model21-00021000.pyrnn.gz**		**model24-00021000.pyrnn.gz**	
errors	134	errors	126	errors	140	errors	121
total	2446	total	2446	total	2446	total	2446
err	5.478 %	err	5.151 %	err	5.724 %	err	4.947 %
model15-00022000.pyrnn.gz		**model18-00022000.pyrnn.gz**		**model21-00022000.pyrnn.gz**		**model24-00022000.pyrnn.gz**	
errors	136	errors	130	errors	136	errors	108
total	2446	total	2446	total	2446	total	2446
err	5.560 %	err	5.315 %	err	5.560 %	err	4.415 %
model15-00023000.pyrnn.gz		**model18-00023000.pyrnn.gz**		**model21-00023000.pyrnn.gz**		**model24-00023000.pyrnn.gz**	
errors	135	errors	126	errors	132	errors	114
total	2446	total	2446	total	2446	total	2446
err	5.519 %	err	5.151 %	err	5.397 %	err	4.661 %
model15-00024000.pyrnn.gz		**model18-00024000.pyrnn.gz**		**model21-00024000.pyrnn.gz**		**model24-00024000.pyrnn.gz**	
errors	126	errors	130	errors	129	errors	102
total	2446	total	2446	total	2446	total	2446
err	5.151 %	err	5.315 %	err	5.274 %	err	4.170 %
model15-00025000.pyrnn.gz		**model18-00025000.pyrnn.gz**		**model21-00025000.pyrnn.gz**		**model24-00025000.pyrnn.gz**	
errors	118	errors	125	errors	120	errors	107
total	2446	total	2446	total	2446	total	2446
err	4.824 %	err	5.110 %	err	4.906 %	err	4.374 %
model15-00026000.pyrnn.gz		**model18-00026000.pyrnn.gz**		**model21-00026000.pyrnn.gz**		**model24-00026000.pyrnn.gz**	
errors	124	errors	134	errors	128	errors	117
total	2446	total	2446	total	2446	total	2446
err	5.070 %	err	5.478 %	err	5.233 %	err	4.783 %
model15-00027000.pyrnn.gz		**model18-00027000.pyrnn.gz**		**model21-00027000.pyrnn.gz**		**model24-00027000.pyrnn.gz**	
errors	116	errors	127	errors	119	errors	104
total	2446	total	2446	total	2446	total	2446
err	4.742 %	err	5.192 %	err	4.865 %	err	4.252 %
model15-00028000.pyrnn.gz		**model18-00028000.pyrnn.gz**		**model21-00028000.pyrnn.gz**		**model24-00028000.pyrnn.gz**	
errors	115	errors	134	errors	123	errors	116
total	2446	total	2446	total	2446	total	2446
err	4.702 %	err	5.478 %	err	5.029 %	err	4.742 %
model15-00029000.pyrnn.gz		**model18-00029000.pyrnn.gz**		**model21-00029000.pyrnn.gz**		**model24-00029000.pyrnn.gz**	
errors	124	errors	142	errors	121	errors	111
total	2446	total	2446	total	2446	total	2446
err	5.070 %	err	5.805 %	err	4.947 %	err	4.538 %
model15-00030000.pyrnn.gz		**model18-00030000.pyrnn.gz**		**model21-00030000.pyrnn.gz**		**model24-00030000.pyrnn.gz**	
errors	114	errors	138	errors	145	errors	111
total	2446	total	2446	total	2446	total	2446
err	4.661 %	err	5.642 %	err	5.928 %	err	4.538 %

Anzahl trainierter Seiten	15	18	21	24
Bestes Modell	4.661 %	4.661 %	4.865 %	4.170 %
Steps in 1.000	30	12	27	24

Tabelle 7.22: Auswertung der Trainingsmodelle unterschiedlicher Zeitintervalle auf dem Narrenschiff

Dauer der Eingabe in Minuten	Seite	Fehlerbezeichnung	Tesseract	Ocropy
0	1	CER	95,39	
		WER	100,00	
		WER (order independent)	99,46	
	2	CER	96,71	
		WER	99,51	
		WER (order independent)	99,51	
1	1	CER	42,74	
		WER	90.76	
		WER (order independent)	86,41	
	2	CER	63,02	
		WER	98,05	
		WER (order independent)	94,15	
2	1	CER	43,41	
		WER	91,85	
		WER (order independent)	89,67	
	2	CER	58,31	
		WER	93,66	
		WER (order independent)	91,22	
3	1	CER	26,36	
		WER	79,35	
		WER (order independent)	75,00	
	2	CER	45,66	
		WER	85,37	
		WER (order independent)	75,61	
10	1	CER	12,17	
		WER	44,02	
		WER (order independent)	42,39	
	2	CER	35,85	
		WER	68,29	
		WER (order independent)	55,61	
15	1	CER	18,47	5,63
		WER	59,24	27,72
		WER (order independent)	54,89	27,17
	2	CER	36,75	26,57
		WER	74,15	46,83
		WER (order independent)	60,00	37,07
30	1	CER	14,53	4,95
		WER	57,07	29,35
		WER (order independent)	54,89	28,80
	2	CER	36,90	27,10
		WER	78,54	41,95
		WER (order independent)	63,90	30,24

Dauer der Eingabe in Minuten	Seite	Fehlerbezeichnung	Tesseract	Ocropy
45	1	CER	21,33	4,70
		WER	76,09	26,63
		WER (order independent)	73,91	26,09
	2	CER	40,19	26,20
		WER	84,88	40,49
		WER (order independent)	75,12	29,27
60	1	CER	12,17	4,28
		WER	45,65	22,83
		WER (order independent)	42,93	22,83
	2	CER	33,98	25,67
		WER	74,15	41,46
		WER (order independent)	60,49	28,78
75	1	CER	8,56	3,36
		WER	32,61	18,48
		WER (order independent)	29,89	18,48
	2	CER	31,66	25,60
		WER	58,05	36,10
		WER (order independent)	40,98	24,39
90	1	CER	7,64	3,86
		WER	25,54	23,91
		WER (order independent)	23,37	22,83
	2	CER	30,99	25,52
		WER	54,15	37,07
		WER (order independent)	38,05	23,41
120	1	CER	6,97	3,61
		WER	20,11	21,20
		WER (order independent)	18,48	21,20
	2	CER	30,01	25,37
		WER	46,83	35,12
		WER (order independent)	31,22	21,95
150	1	CER	8,73	
		WER	30,43	
		WER (order independent)	28,80	
	2	CER	31,36	
		WER	50,73	
		WER (order independent)	37,07	
180	1	CER	5,54	
		WER	19,57	
		WER (order independent)	17,39	
	2	CER	29,72	
		WER	48,29	
		WER (order independent)	33,66	

Tabellenverzeichnis

1.1 Klassifizierung der Fehlerarten eines OCR-Prozesses 6

2.1 Auswertung von Ocropus-LSTM, Tesseract und ABBYY auf englischen Dokumenten und zwei deutschsprachigen Büchern in Frakturschrift 10
2.2 Auswertung deutscher Frakturschrift unterschiedlicher Jahrhunderte mittels Zeichenpräzision in Prozent 10
2.3 Auswertung von „Pontanus, Progymnasmata Latinitatis“ aus dem Jahr 1589 11
2.4 Auswertung von Thanner, Petronij Arbitri Sathyra aus dem 15. Jahrhundert 11
2.5 Auswertung von verschiedenen Sprachmodellen von Tesseract auf der Würzburger Bischofs-Chronik von Lorenz Fries 12
2.6 Auswertung von GOCR und Tesseract mit unterschiedlichen Helligkeitswerten 12
2.7 Auswertung von SVM-Kernels mit Gabor- und Grandientenfeaturen 13
2.8 Auswertung unterschiedlicher Methoden und Ausgangsbasen . . . 13

7.1 Auswertung der Standardmodelle auf den deutschsprachigen Buchseiten 41
7.2 Auswertung der Standardmodelle auf den deutschsprachigen Zeitschriftsseiten 42
7.3 Auswertung der Standardmodelle auf den englischsprachigen Buchseiten 42
7.4 Auswertung der Standardmodelle auf den englischsprachigen Zeitschriftsseiten 42
7.5 Auswertung der Trainingsmodelle auf den deutschsprachigen Buchseiten 43
7.6 Auswertung der Trainingsmodelle auf den deutschsprachigen Zeitschriftsseiten 43
7.7 Auswertung der Trainingsmodelle auf den englischsprachigen Buchseiten 44
7.8 Auswertung der Trainingsmodelle auf den englischsprachigen Zeitschriftsseiten 44
7.9 Auswertung der Standardmodelle auf der Gartenlaube 45

7.10 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (1),30 (2),45 (3) und 60 (4) Minuten von 1.000 bis 15.000 Steps . 46
7.11 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (5),90 (6),105 (7) und 120 (8) Minuten von 1.000 bis 15.000 Steps 47
7.12 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (1),30 (2),45 (3) und 60 (4) Minuten von 16.000 bis 30.000 Steps . 48
7.13 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (5),90 (6),105 (7) und 120 (8) Minuten von 16.000 bis 30.000 Steps 49
7.14 Auswertung der Trainingsmodelle unterschiedlicher Zeitintervalle auf der Gartenlaube 50
7.15 Auswertung der Standardmodelle auf den neuen Serapionsbrudern 50
7.16 Auswertung der Trainingsmodelle von der Gartenlaube auf den neuen Serapionsbrudern 51
7.17 Auswertung der Standardmodelle auf dem Narrenschiff 51
7.18 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (model3),30 (model6),45 (model9) und 60 (model12) Minuten von 1.000 bis 15.000 Steps 52
7.19 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (model15),90 (model18),105 (model21) und 120 (model24) Minuten von 1.000 bis 15.000 Steps 53
7.20 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 15 (model3),30 (model6),45 (model9) und 60 (model12) Minuten von 16.000 bis 30.000 Steps 54
7.21 Auswertung der Fehlerraten der Modelle für die Zeitintervalle 75 (model3),90 (model6),105 (model9) und 120 (model12) Minuten von 16.000 bis 30.000 Steps 55
7.22 Auswertung der Trainingsmodelle unterschiedlicher Zeitintervalle auf dem Narrenschiff 56

Abbildungsverzeichnis

1.1 Evaluationsprogramm “ocrevalUation„ 5
1.2 Begründung der CER anhand eines Beispieles 7

3.1 Beispiel von Merkmalen anhand gebrochener Schrift[21][S.252] . . 17
3.2 Beispiel einer Sperrung mit der ausgeschlossenen Ligatur ȸ und einer römischen Ziffer 18

4.1 Terminal während des Trainingsprozesses 22

5.1 Visualisierung des deutschsprachigen Buches mit Standardmodellen 27
5.2 Visualisierung des englischsprachigen Buches mit Standardmodellen 28
5.3 Visualisierung der Standardmodelle auf der Gartenlaube 30
5.4 Auszug der Modellliste nach 15 Minuten Training auf den Testseiten 31
5.5 Modellvergleich in Minuten zwischen Tesseract und Ocropy auf der Gartenlaube 32
5.6 Modellvergleich in Minuten zwischen Tesseract und Ocropy auf den Serapionsbrüdern 33
5.7 Visualisierung der Standardmodelle auf dem Narrenschiff 34
5.8 Modellvergleich in Minuten zwischen Tesseract und Ocropy auf dem Narrenschiff 35

7.1 Typischer Workflow der OCR[20] 40
7.2 Differenzierung von Antiqua und gebrochenen Schriften[22] 41

Literaturverzeichnis

[1] CuneiForm. URL: http://cognitiveforms.com/products_and_services/cuneiform. (abgerufen am 26.08.2015).

[2] Cuneiform-Linux. URL: http://wiki.ubuntuusers.de/cuneiform-linux. (abgerufen am 27.08.2015).

[3] Die Gartenlaube (1867). URL: https://de.wikisource.org/wiki/Die_Gartenlaube_(1867). (abgerufen am 11.08.2015).

[4] Die Herausforderung: Digitalisierung alter Dokumente. URL: http://www.frakturschrift.com/de:start. (abgerufen am 14.08.2015).

[5] FreeOCR 5.4.1. URL: https://www.heise.de/download/freeocr-1149486.html. abgerufen am 26.08.2015.

[6] GNU Ocrad Manual. URL: http://www.gnu.org/software/ocrad/manual/ocrad_manual.html. (abgerufen am 27.08.2015).

[7] Java OCR and Barcode Recognition. URL: http://asprise.com/royalty-free-library/java-ocr-api-overview.html. (abgerufen am 08.07.2015).

[8] Ocrocis. URL: https://github.com/kmnns/ocrocis. (abgerufen am 27.08.2015).

[9] Puma.NET. URL: http://pumanet.codeplex.com/. (abgerufen am 26.08.2015).

[10] Research at Google, Ray Smith. URL: http://research.google.com/pubs/author4479.html. (abgerufen am 08.07.2015).

[11] Text Digitisation - The ocrevalUAtion tool. URL: https://sites.google.com/site/textdigitisation/ocrevaluation. (abgerufen am 14.08.2015).

[12] Über die optische Zeichenerkennung in Google Drive. URL: https://support.google.com/drive/answer/176692?hl=de. (abgerufen am 08.07.2015).

[13] Wikipedia - Das Narrenschiff. URL: https://de.wikipedia.org/wiki/Das_Narrenschiff_(Brant). (abgerufen am 16.08.2015).

[14] Costin-Anton Boiangiu, Ion Bucur, and Andrei Tigora. The image binarization problem revisited: perspectives and approaches. The Proceedings of Journal ISOM, 6(2):419–427, 2012.

[15] Costin-Anton Boiangiu and Andrei Iulian Dvornic. Methods of bitonal image conversion for modern and classic documents. WSEAS Transactions on Computers, 7(7):1081–1090, 2008.

[16] Eugene Borovikov. A survey of modern optical character recognition techniques. arXiv preprint arXiv:1412.4183, 2014.

[17] Thomas M Breuel, Adnan Ul-Hasan, Mayce Ali Al-Azawi, and Faisal Shafait. High-performance OCR for printed English and Fraktur using LSTM networks. In Document Analysis and Recognition (ICDAR), 2013 12th International Conference on, pages 683–687. IEEE, 2013.

[18] Mohamed Cheriet, Nawwaf Kharma, Cheng-Lin Liu, and Ching Suen. Character recognition systems: a guide for students and practitioners. John Wiley & Sons, 2007.

[19] MS Dhiman and P Dr AJ Singh. Tesseract vs gocr a comparative study. International Journal of Recent Technology and Engineering, 2(4):80, 2013.

[20] Maria Federbusch and Christian Polzin. Volltext via OCR - Möglichkeiten und Grenzen. URL: `http://staatsbibliothek-berlin.de/fileadmin/user_upload/zentrale_Seiten/historische_drucke/pdf/SBB_OCR_STUDIE_WEBVERSION_Final.pdf`.

[21] Norbert Hammer. Mediendesign für Studium und Beruf: Grundlagenwissen und Entwurfssystematik in Layout, Typografie und Farbgestaltung. Springer-Verlag, 2008.

[22] Claas Kalwa. Gebrochene Schriften. URL: `https://schriftgestaltung.com/schriftgestaltung/schriftklassen/gebrochene-schriften.html`, Mai 2015. (abgerufen am 11. 05. 2015).

[23] Rajneesh Rani, Renu Dhir, and Gurpreet Singh Lehal. Script identification of pre-segmented multi-font characters and digits. In Document Analysis and Recognition (ICDAR), 2013 12th International Conference on, pages 1150–1154. IEEE, 2013.

[24] Parinya Sanguansat, Patcharin Yanwit, Paisam Tangwiwatwong, Widhyakorn Asdornwised, and Somchai Jitapunkul. Language-based hand-printed character recognition: a novel method using spatial and temporal informative features. In Neural Networks for Signal Processing, 2003. NNSP'03. 2003 IEEE 13th Workshop on, pages 527–536. IEEE, 2003.

[25] Jörg Schulenburg. GOCR-documentation. URL: `http://mcs.une.edu.au/doc/gocr/gocr.html`, 2002. (abgerufen am 08.07.2015).

[26] Heinrich Schwietering. Wiki/GOCR. URL: `https://wiki.ubuntuusers.de/GOCR`, 2014. (abgerufen am 08.07.2015).

[27] David Springmann, Uwe und Kaumanns. ocrocis – a high accuracy OCR method to convert early printings into digital text, 2015.

[28] Uwe Springmann, Dietmar Najock, Hermann Morgenroth, Helmut Schmid, Annette Gotscharek, and Florian Fink. OCR of historical printings of Latin texts: problems, prospects, progress. In Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage, pages 71–75. ACM, 2014.

[29] Christian M Strohmaier. Methoden der lexikalischen Nachkorrektur OCR-erfasster Dokumente. PhD thesis, LMU, 2005.

[30] Dan Vanderkam. Extracting text from an image using Ocropus. URL: `http://www.danvk.org/2015/01/09/extracting-text-from-an-image-using-ocropus.html`. (abgerufen am 05.08.2015).

[31] Paul Vorbach. Analysen und Heuristiken zur Verbesserung von OCR-Ergebnissen bei Frakturtexten. 2014.

[32] Z. Podobnỳ. 3rdParty, GUIs and Other Projects using Tesseract OCR. URL: `https://code.google.com/p/tesseract-ocr/wiki/3rdParty`. (abgerufen am 18.07.2015).

[33] Yefeng Zheng, Changsong Liu, and Xiaoqing Ding. Single-character type identification. In Electronic Imaging 2002, pages 49–56. International Society for Optics and Photonics, 2001.

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Bachelorarbeit selbständig und ohne unzulässige, fremde Hilfe verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Die bildlichen Darstellungen, Tabellen, Quelltexte und Graphen habe ich selbst angefertigt.

Würzburg, den 11. September 2015