Segmentierung von historischen Zeitungsseiten

Christian Schultze

Geboren am 31.08.1998, München

Bachelorarbeit Informatik

Betreuer: Dr. Moritz Wolter
Fach-Betreuer: PD Dr. Felix Selgert
Zweitgutachter: Prof. Dr. Reinhard Klein

HPC/A-Lab und Institut für Informatik II

Mathematisch-Naturwissenschaftliche Fakultät
Rheinische Friedrich-Wilhelms-Universität Bonn

7. Dezember 2023

Inhaltsverzeichnis

1 Einleitung **1**
- 1.1 Allgemein . . . 1
- 1.2 Zielsetzung . . . 2
- 1.3 Aufbau . . . 3

2 Literatur **4**
- 2.1 Allgemein . . . 4
- 2.2 Algorithmische Verfahren . . . 4
- 2.3 Faltungs-Modelle . . . 6
  - 2.3.1 Faltung . . . 6
  - 2.3.2 CNN Encoder-Decoder . . . 7
  - 2.3.3 U-Net . . . 9
  - 2.3.4 Einbeziehung semantischer Informationen . . . 10
  - 2.3.5 Evaluation von Deep Learning Modellen . . . 11
- 2.4 Transformer . . . 11
  - 2.4.1 Attention . . . 11
  - 2.4.2 Vision Transformer . . . 14
  - 2.4.3 Convolution Block Attention Module . . . 16
- 2.5 Daten . . . 17

3 Implementierung **19**
- 3.1 Training . . . 19
  - 3.1.1 Allgemein . . . 19
  - 3.1.2 Daten . . . 20
  - 3.1.3 Validation und Test . . . 21
- 3.2 Vorhersage . . . 22
  - 3.2.1 Allgemein . . . 22
  - 3.2.2 Transkribus-Export . . . 22
  - 3.2.3 Slicing-Export . . . 24
- 3.3 Modelle . . . 24
  - 3.3.1 DhSegment . . . 24
  - 3.3.2 DhSegment CBAM . . . 25
  - 3.3.3 TransUnet . . . 25

4 Experimente **26**
4.1 Allgemein . . . 26
4.2 Daten . . . 27
4.3 Laufzeit . . . 28
4.3.1 Optimierung durch Torch Implementationen . . . 28
4.3.2 Optimierung durch Multithreading . . . 29
4.3.3 Optimierung des Batch-Loadings . . . 30
4.3.4 Vorhersage . . . 31
4.4 Initiales Training . . . 33
4.5 Pretraining . . . 36
4.6 Hyperparameter Suche . . . 37
4.6.1 Loss Funktion . . . 37
4.6.2 Lernrate und Weight Decay . . . 40
4.6.3 Skalierung und Klassen Reduzierung . . . 41
4.6.4 Finales Training . . . 42

5 Ergebnisse **45**

6 Schlussfolgerungen und Ausblick **48**

A Anhang **60**
A.1 Trainingsgeschwindigkeit . . . 60
A.2 Verlauf Initiales Training . . . 61
A.3 Vorhersage Initiales Training . . . 63
A.4 Ergebnisse reduziertes Modell . . . 67
A.5 Ergebnisse Skalierung . . . 67
A.6 Vorhersage des finalen Modells . . . 68
A.7 1924 Zeitungsseiten . . . 74
A.8 Ergebnis Slice Export . . . 76
A.9 XML Beispiel . . . 82
A.10 Slurm Beispiel . . . 82
A.11 Datensatz Kölnische Zeitung . . . 82

Danksagung

Ich möchte mich für die Unterstützung, die ich im Laufe der Bachelorarbeit bekommen habe, bedanken. Ich danke meinem Betreuer Dr. Moritz Wolter für die technische Hilfe und die Möglichkeit diese Arbeit in diesem Umfang durchführen zu können. Ich danke meinem Betreuer PD Dr. Felix Selgert für die Organisation und Anregungen aus der geschichtswissenschaftlichen Perspektive. Ich bedanke mich bei Kara und Jonas, die sich die Mühe gemacht haben, in ihrer Freizeit meine Arbeit Korrektur zu lesen. Zuletzt danke ich dem Annotationsteam aus dem Institut für Geschichtswissenschaft, die diese Arbeit erst möglich gemacht haben.

Kapitel 1

Einleitung

1.1 Allgemein

Durch die fortschreitende Digitalisierung von Archivalien und anderen historischen Dokumenten eröffnen sich neue Möglichkeiten für die historische Forschung. Dabei wird auch die automatisierte Verarbeitung historischer Texte immer wichtiger. Zeitungen sind wichtige Quellen historischer Forschung, es reicht jedoch nicht, digitale Bilder einer Zeitung vorliegen zu haben. Eine korrekte Transkription und Lesereihenfolge ist zur effizienten Arbeit nötig. Transkribierte Zeitungsseiten würden es ermöglichen, Artikel einer Zeitung nach bestimmten Kriterien zu durchsuchen, Schlagwörter im Text oder in Überschriften zu finden und automatisierte Analysen über Jahrzehnte hinweg durchzuführen.

Es existieren bereits Modelle für Texterkennung (Optical Character Recognition) (OCR) , welche Text von historischen Zeitungen erkennen können. Allerdings verstehen diese das Layout einer Zeitung nicht. Das führt zum Beispiel dazu, dass nicht innerhalb einer Spalte gelesen wird, sondern über Spaltengrenzen hinweg Satzteile aneinandergereiht werden. Die resultierende Transkription ist dadurch nicht verwertbar. Die Projektgruppe Segmentierung von Zeitungsseiten [33] setzt an diesem Punkt an und trainiert ein Segmentierungsmodell auf einem Datensatz der Kölnischen Zeitung, welches das Layout der Zeitung erkennen soll. Die Layouterkennung als erster Schritt und eine OCR als zweiten Schritt, soll es dann in Zukunft ermöglichen, eine Zeitungsseite vollständig zu transkribieren.

In der Projektgruppe wurden 140 Seiten der Kölnischen Zeitung von Hilfskräften am geschichtswissenschaftlichen Institut annotiert. Die Daten enthalten 10 Klassen, welche stark unterschiedliche Häufigkeiten aufweisen. Seltene Klassen wie Tabellen, Überschriften und Titelzeilen stellen das Modell vor Herausforderungen. Der Datensatz der Kölnischen Zeitung wurde während dieser Arbeit erweitert, sodass 402 Seiten für die Experimente zur Verfügung stehen. [33]

Die seltenen Klassen sind besonders schwer zu lernen, weil zum Beispiel

Überschriften und Tabellen die gleiche Schriftart aufweisen, wie der restliche Text. Überschriften sind meist zentriert und etwas größer oder dicker als der restliche Text, allerdings kommen auch im Text fettgedruckte Stellen vor. Tabellen sind nur an der Struktur und eventuell vorhandenen Trennlinien zu erkennen. Für die Erkennung dieser Regionen kann es von Vorteil sein, die grobe Struktur besser einbeziehen zu können, anstatt eine begrenzte lokale Umgebung. Deswegen sollen in dieser Arbeit Modelle evaluiert werden, die auf Aufmerksamkeit (Attention) basieren. Im Bereich der Verarbeitung natürlicher Sprache (Natural Language Processing) (NLP) werden Transformer verwendet, um den Kontext von menschlicher Sprache erfassen zu können. Dieser Ansatz kann auch auf Bildverarbeitung angewendet werden und soll hier als Alternative zu reinen Faltungsnetzen (Abschnitt 2.3) untersucht werden.

1.2 Zielsetzung

In dieser Arbeit sollen mehrere Modelle zur Segmentierung von historischen Zeitungsseiten untersucht werden. Neben dem bereits erfolgreich trainierten DhSegment Modell sollen 2 Abwandlungen von DhSegment implementiert und evaluiert werden. Einerseits soll nach dem Vorbild des Trans-Unet [6] ein Transformer in der letzten Schicht des Encoders eingebaut werden (Unterabschnitt 3.3.3). Dieser soll besser als die Faltungsschichten Kontext-Informationen beachten und diese an den Decoder weiterleiten. Als dritte Variante sollen Convolutional Block Attention Module (CBAM) nach den unteren 4 Encoder Schichten von DhSegment eingebaut werden (Unterabschnitt 3.3.2). Die CBAM Module verwenden eine auf Pooling basierende Art der Aufmerksamkeit, die kompatibel mit der Faltung ist und lokale Teile der Eingabe beachtet.

Außerdem soll diese Arbeit dazu beitragen, historische Zeitungsseiten im Allgemeinen und die Kölnische Zeitung im Speziellen transkribieren zu können. Um die Ausgabe des Modells weiterverarbeiten zu können, ist es nötig, die Vorhersage für jedes Pixel in ein Format zu überführen, welches zum Beispiel Transkribus[44] verarbeiten kann. Transkribus ist die Software, mit der die Annotationen erstellt werden. Das Importieren von Annotationen ist über das, auch für den Export, verwendete Page XML Format [48] möglich. In diesem Format sind Layout-Regionen in Form von Polygon-Koordinaten gespeichert. Die Ausgabe des Modells muss also in Polygone übersetzt und in eine solche XML-Datei geschrieben werden.

Ein besonders interessanter Teil des Layouts sind Überschriften, weil anhand dieser Artikel identifiziert und voneinander unterschieden werden könnten. Dies würde es erlauben, die Daten nach Artikeln sortiert, abzuspeichern und automatische Analysen, zum Beispiel über Inhalt und Frequenz von Artikeln durchzuführen. Während der Evaluation wird deswegen ein größerer Wert auf die Überschriften gelegt.

1.3 Aufbau

In Kapitel 2 werden die theoretischen Grundlagen von Faltungsnetzen und Transformern, sowie den Convolutional Block Attention Module (CBAM) dargelegt, sowie ein Überblick darüber gegeben, welche Vorarbeiten es im Kontext von historischen Dokumenten und Zeitungen gibt. Kapitel 3 enthält Beschreibungen der Implementation bezüglich des Trainings, der 3 verwendeten Modelle, sowie der Vorhersage dieser Modelle für unbekannte Daten. In Kapitel 4 werden die durchgeführten Experimente und dafür verwendeten Daten beschrieben. Darunter fällt das initiale Training auf dem Bender GPU Cluster, welches vor allem zum Testen der Implementation durchgeführt wurde, sowie die Experimente auf dem JUWELS (Jülich Wizard for European Leadership Science) Supercomputer. Diese Experimente umfassen die Verbesserung der Laufzeit, um die zur Verfügung stehende Rechenzeit möglichst effizient zu nutzen, sowie eine für das DhSegment Modell durchgeführte Hyperparametersuche. Die Ergebnisse dieser Arbeit werden schließlich in Kapitel 5 erläutert und in Kapitel 6 sind die Schlussfolgerungen sowie ein Ausblick zu finden.

Kapitel 2

Literatur

2.1 Allgemein

Die automatische Transkription von historischen Dokumenten und Zeitungen ist für das Erfassen von mikro- und makroskopische Details in der historischen Forschung wichtig. Mikroskopische Arbeit benötigt detaillierte Informationen für einen bestimmten Zeitraum. Die Untersuchung eines bestimmten historischen Moments, erfordert es, in möglichst vielen Dokumenten nach einem bestimmten Datum oder Ereignis suchen zu können. Makroskopische Arbeit bedeutet, größere Trends und Zusammenhänge aufzudecken. Dafür müssen Informationen aus vielen Quellen zusammengefasst werden, um zum Beispiel ein Stimmungsbild über mehrere Jahre anhand von Zeitungsartikeln zusammenzutragen. [20]

Der Kern der automatisierten Transkription von Dokumenten ist die Texterkennung (Optical Character Recognition) (OCR). Im englischen ist die Identifizierung von einzelnen Zeichen, die gemeinsam Wörter und Sätze bilden, namensgebend. Dies kann jedoch nicht der erste Schritt sein, da das Layout von den meisten Zeitungen keine einfache Heuristik erlaubt, welche Teile des Textes zusammengehören. Layoutelemente wie Überschriften, Header und Footer Elemente, sowie Trennlinien müssen erkannt werden, um keine unzusammenhängenden Texte zu erzeugen. Die Ergebnisse der OCR hängen maßgeblich von der vorherigen Layout-Analyse ab. [25]

2.2 Algorithmische Verfahren

Bevor maschinelles Lernen effektiv für große Datenmengen anwendbar war, wurden bereits algorithmische Ansätze für die Segmentierung von historischen Zeitungen verwendet. Schon 1982 stellen Wong et al. [57] algorithmische Methoden vor, um ein Dokument in Blöcke aufzuteilen, die jeweils Text enthalten oder frei von Text sind. Dafür werden weiße und schwarze Pixel in einer Liste als Einsen und Nullen aufgeführt. Jede Null wird zu einer Eins, falls die Anzahl der umgebenden Nullen unterhalb eines festgelegten Limits ist. Auf diese Weise

können die Regionen identifiziert werden, die für die OCR relevant wären. [57]

Gatos et al. [18] verwenden 1999 ein sequentielles Vorgehen, um Zeitungsseiten zu segmentieren und Artikel zu identifizieren. Dafür werden grundlegende Regionen allmählich aus den Bildern extrahiert. Die Reihenfolge ist so gewählt, dass Regionen, die die Erkennung anderer Regionen erschweren, zuerst identifiziert werden . Zum Beispiel können Linien zwischen Textblöcken es erschweren, die Textblöcke voneinander zu unterscheiden, wenn die Linien nicht zuvor identifiziert wurden. Zusätzlich zu den Vordergrund-Regionen werden auch Hintergrund-Linien berücksichtigt. Das sind zum Beispiel die weißen Stellen zwischen Artikeln. Diese bilden die Grundlage zur Identifikation von Artikeln. [18]

Im Zuge des ersten Newspaper Segmentation Contest, durchgeführt von dem Department of Digital Technologies des Labrakis Press Archives und dem Department of Computer Science der Universität von Liverpool in Seattle im Jahr 2001, wurden 3 algorithmische Ansätze von Gatos et al. [17] ausgewertet. Es werden Definitionen von Regionen vorgegeben, die Text, Überschriften, Linien und Grafiken voneinander unterscheiden. Bei 2 Ansätzen handelt es sich um Bottom-Up und bei einem um Split and Merge Ansätze. Die Bottom-Up Ansätze identifizieren zunächst entweder Textblöcke oder Linien im Dokument, um aus diesen Informationen das größere Bild zusammenzubauen. Auch beim Split and Merge Ansatz werden Linien identifiziert, um das Dokument in Regionen zu unterteilen. Diese Regionen werden dann aber wieder zusammengefasst, wenn bestimmte Kriterien erfüllt sind. Auch Mitchel et al. [30] verwenden einen Bottom-Up Ansatz, der rechteckige Regionen wie Wörter erkennt und dann zu größeren Regionen zusammenfasst. [30, 17]

Die Herausforderung bei historischen Zeitungen sind die komplexen Layouts, für die keine einfachen Regeln definiert werden können. Sowohl die Identifizierung von einfachen Regionen wie Überschriften oder Tabellen, als auch die Lesereihenfolge der Artikel in Zeitungen stellen algorithmische Ansätze vor Herausforderungen. Mit der besseren Verfügbarkeit von Systemen des maschinellen Lernens wurden auch bei historischen Dokumenten selbstlernende Algorithmen zur Layout-Erkennung angewendet. Williams et al. [56] haben schon 1996 ein Neurales Netz verwendet, um

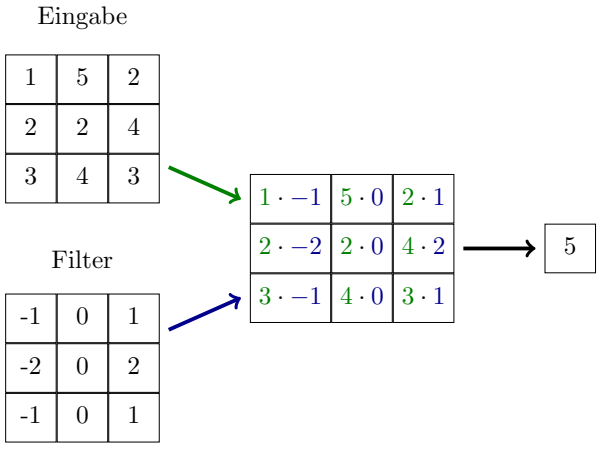


Abbildung 2.1: Einfache Faltungs-Operation als gewichtete Summe

Bilder und Text in Zeitungsseiten zu segmentieren. Verwendet wird ein quadratisches neurales Netz mit einer verdeckten Schicht. Die Eingaben des Modells sind extrahierte und parametrisierte Featuremaps. Dieser Schritt wird heutzutage oft von einem selbstlernenden Encoder durchgeführt, anstatt eine Verarbeitung zu implementieren. Die Verarbeitung basiert zum Beispiel auf Helligkeitswerten und Abständen zwischen hellen und dunklen Pixeln. [56]

2.3 Faltungs-Modelle

2.3.1 Faltung

Für die Verarbeitung von Bildern haben sich Faltungsnetze (convolutional neural network) (CNN)s durchgesetzt. CNNs sind von der Grundidee her wie Neuronale Netze (NN). Es gibt auch hier Neuronen, die Eingaben erhalten und diese verarbeiten. Die Verarbeitung besteht aus einer gewichteten Summe, sowie einer nicht-linearen Aktivierungsfunktion. Auch gibt es bei beiden mehrere Schichten von parallelen Neuronen.

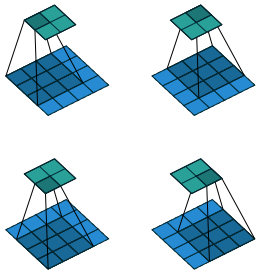


Abbildung 2.2: Die 4 Schritte einer 3x3 Faltung auf einer 5x5 Eingabe [14]

NNs arbeiten normalerweise mit vollständig vernetzten Schichten. Es gibt also von jeder Eingabe zu jedem Neuron ein Gewicht. Bei Bildern sind diese vollständig vernetzten Schichten sehr rechenaufwendig, weil jedes Neuron der ersten verdeckten Schicht Eingaben von allen Pixeln bekommt. Bei dem einfachen MNIST Datensatz[59] (Bilder von handschriftlichen Ziffern geringer Auflösung) sind es zum Beispiel 28x28 Pixel. Bei 100 Neuronen der verdeckten Schicht wären das 28x28x100 Gewichte. Das wären 78400 Gewichte für die erste Schicht, obwohl es sich um kleine Bilder handelt, die keine Farben enthalten. Dazu kommt, dass das Neurale Netz hier jeweils eine Funktion für jedes Pixel des Bildes separat lernt. Es ist aber nicht nötig, für die obere linke Ecke eines Bildes etwas anderes zu lernen, als für die untere rechte Ecke. In beiden Fällen soll die gleiche Funktion gelernt werden. Hier kommen die geteilten Gewichte von Faltungs-Operatoren ins Spiel. [35]

Faltungs-Kerne oder -Filter werden schrittweise auf das ganze Bild angewendet. Ein Filter besteht aus $n \times n$ Gewichten und wird als lokale gewichtete Summe angewendet. In Abbildung 2.1 wird ein 3x3 Filter mit 9 Gewichten auf eine 3x3 Eingabe angewendet. Das Ergebnis ist ein einzelner Wert, analog zu der gewichteten Summe bei vollständig vernetzten Modellen. Wenn die Eingabe aber größer ist, als der Filter, wird der Filter wiederholt auf unterschiedliche Berei-

che des Bildes angewendet. In Abbildung 2.2 werden die Gewichte zunächst auf die 3x3 Eingabe-Werte eines Bildes angewendet, die in der oberen linken Ecke sind. Die gewichtete Summe aus den 9 Eingabe-Werten und den 9 Gewichten im Filter ist der erste Pixelwert der Ausgabe. Danach wird der Filter um ein Pixel verschoben und erneut die gewichtete Summe berechnet, bis das gesamte Bild bearbeitet wurde. Die Ausgabe ist wiederum ein Bild, bestehend aus allen Summen, die bei diesem Prozess entstanden sind. Durch dieses Verfahren sind die Werte auch in der Ausgabe noch geometrisch korrekt angeordnet. Werte, die also in der Nähe voneinander liegen, enthalten auch Informationen von benachbarten Bereichen des Eingabe-Bildes. Wie bei NNs werden nach den gewichteten Summen nicht-lineare Aktivierungsfunktionen angewendet. Bei CNNs hat sich die ReLU Aktivierungsfunktion durchgesetzt. Diese bildet alle negativen Werte auf 0 ab, im positiven Bereich entspricht ReLU der Identität. Die Funktion wird elementweise auf alle Pixel angewendet. [35]

2.3.2 CNN Encoder-Decoder

Analog zu mehreren parallelen Neuronen in einer Schicht eines NNs, werden bei CNNs mehrere Faltungen mit unterschiedlichen lernbaren Filtern parallel durchgeführt. Jeder Filter erzeugt ein eigenes Ausgabe-Bild, welches Informationen zu einem bestimmten Feature gespeichert hat (Feature Maps). Die Ausgabe einer Faltungs-Schicht ist also eine 3-Dimensionale Matrix, bestehend aus Höhe, Breite und Anzahl der Feature Maps (Kanäle). Auch die Eingabe kann mehrere Kanäle besitzen. Abbildung 2.3 zeigt ein Beispiel, bei dem die Eingabe aus 3 Kanälen eines RGB Bildes besteht. Alle 3 Kanäle werden von der Faltungs-Operation beachtet, was dazu führt, dass für jeden der 3 Eingabe Kanäle eigene Faltungen mit eigenen Filtern durchgeführt werden. Die Ausgabe der Faltungs-Schicht besteht aus 8 Feature Maps. Für jede der 8 Ausgabe Feature Maps wird ebenfalls eigene Faltungen durchgeführt. Aus dem ursprünglichen 2-dimensionalen 3x3 Filter, den 3 Eingabe- und 8 Ausgabe-Kanälen ergibt sich damit ein 4-dimensionaler Faltungs-Filter mit den Dimensionen 3x8x3x3. Danach wird in Abbildung 2.3 eine Pooling Operation mit einer Größe von 2x2 und Stride 2 durchgeführt, um die Größe der Feature Maps zu reduzieren. Die Höhe und Breite jedes Bildes wird dabei halbiert, wodurch die Anzahl der Pixel um den Faktor 4 reduziert wird. Durch die Reduktion der Bild-Größen muss das CNN lernen, eine effiziente Komprimierung des Bildes durchzuführen und sich auf relevante Eigenschaften zu fokussieren. Gleichzeitig mit der Reduktion der Größe wird die Anzahl der Feature Maps erhöht, was

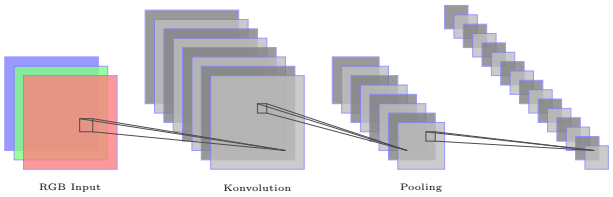


Abbildung 2.3: Darstellung einer Faltungs-Schicht mit RGB Eingabe

zu einer großen Anzahl Bildern führt, die jeweils Informationen über eine bestimmte Eigenschaft speichern. Diese Bilder besitzen deutlich weniger Pixel als das Eingabe-Bild. Jeder Pixel in der Ausgabe entspricht also einem größeren Bereich im Eingabe-Bild. Das ist sinnvoll, weil hier komplexe Features identifiziert werden sollen, wie beispielsweise Ohren oder Nase eines Gesichts. Für die Unterscheidung zwischen verschiedenen Tieren könnte dann die grobe Position der Ohren und deren Größe ausreichen, ohne dass detaillierte Information nötig sind. Bei selbstlernenden Modellen kodieren die Bilder innerhalb des Modells in der Regel keine für Menschen nachvollziehbaren Eigenschaften. [35]

Die Ausgabe eines Segmentierungsmodells ist ein Bild, welches jedem Pixel des Eingabe-Bildes einen Wert zuweist. Jedes Feature in der Ausgabe entspricht einer Klasse, die identifiziert werden soll. Jeder Klasse ist ein Label-Wert zugeordnet. Das Ausgabebild hat genauso viele Pixel wie das Eingabe-Bild, welche jeweils den Label-Wert enthalten, der zu der zugeordneten Klasse gehört. Für diese Aufgabe werden Modelle verwendet, die vollständig aus Faltungs-Schichten aufgebaut sind. Die Architektur in Abbildung 2.4 reduziert die Feature Map Dimensionen zunächst durch Faltungs- und Pooling-Schichten. Über den einzelnen Schichten steht immer die Anzahl der Feature Maps(Kanäle) einer Schicht. Die Pfeile dazwischen stellen die Operationen dar, die durchgeführt werden. Die roten Pooling Operationen halbieren die Bilddimensionen, was auch anhand der Höhe der Schichten zu erkennen ist. Die 3 Kanäle der Eingabe werden durch die erste Konvolution auf 64 Kanäle angehoben. Danach werden Bilddimensionen 2 mal halbiert(rote Pfeile) und in der darauffolgenden Faltung wird die Anzahl der Kanäle verdoppelt. In der Mitte steigt die Anzahl der Feature-Dimension auf 256. Dieser Teil (bis vor den ersten grünen Pfeil) wird Encoder genannt. [29, 33]

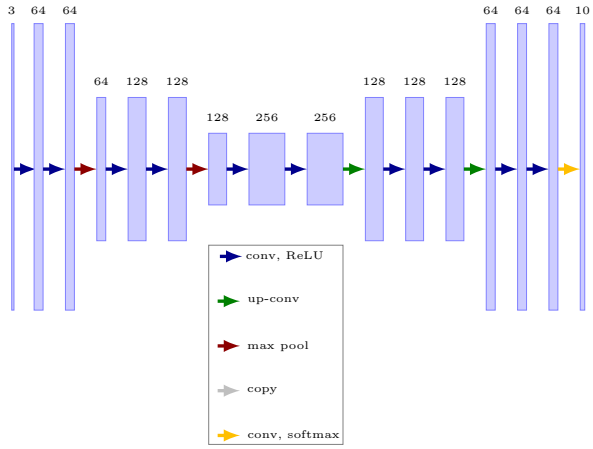


Abbildung 2.4: Beispiel einer Decoder - Encoder Architektur mit 10 Ausgabeklassen

Nach dem Encoder werden die Feature Maps im Decoder über mehrere Schichten hochskaliert. Das geschieht durch transponierte Faltungen. Dabei handelt es sich um die Anwendung eines Faltungs-Filters als Gewichtung eines einzelnen Werts. Wie in Abbildung 2.5 zu sehen, wird aus einem Pixel ein 3x3 Pixel großes Bild generiert. Analog zur Faltung wird diese Operation schrittweise über das gesamte Bild ausgeführt. Am Ende des Decoders in Abbildung 2.4 stehen wieder Feature Maps, welche die gleiche Größe haben wie das Ausgabe-Bild. Die

10 Feature Maps am Ende des Encoders kodieren die 10 verschiedenen Klassen, die identifiziert werden sollen. Abbildung 2.6 zeigt zur Verdeutlichung die Ausgabe eines Modells mit 10 Klassen. Jedes Features der Ausgabe entspricht einer Farbe im Ausgabe-Bild. Der Vorteil von vollständigen Faltungs-Netzen ist die Fähigkeit, unterschiedlich große Bilder zu verarbeiten. Aufgrund der Pooling-Operation gibt es eine Mindestgröße, aber auch größere Eingaben können von dem Modell verarbeitet werden. Die Ergebnisse werden sich in dem Fall nur dann deutlich unterscheiden, wenn es Unterschiede in der Auflösung gibt. [29, 33]

2.3.3 U-Net

Weiter verbessern lässt sich die Architektur, indem zwischen den Schichten des Encoder und Decoder Skip-Connections hinzugefügt werden. Die Decoder-Schichten von U-Nets wie in Abbildung 2.7 haben durch Skip-Connections(Graue Pfeile) nicht nur die Informationen zur Verfügung, die am Ende des Encoders hochskaliert werden, sondern auch die Ausgabe der gegenüberliegenden Encoder-Schicht. Diese beiden Teile werden im Decoder aufaddiert und durch Faltungen weiter verarbeitet. Bei der letzten Schicht des Decoders können also auch die Informationen aus dem Original-Bild miteinbezogen werden. Die Encoder-Schichten werden im besten Fall nach dem Transferlernen-Verfahren auf großen Datensätzen vortrainiert. Die annotierten Datensätze für historische Zeitungen sind begrenzt, aber grundlegende Feature-Extraktionen können von anderen Bildern gelernt werden. Wenn der Encoder zum Beispiel auf dem Image-Net Datensatz [10] vortrainiert wurde, reicht es, den Decoder mit wenigen hundert Zeitungsseiten zu trainieren. Transferlernen kann auch auf den Decoder angewendet werden, indem auf einem möglichst großen Zeitungsdatensatz vortrainiert wird. Das Netz kann nun einen weiteren Zeitungsdatensatz schneller lernen, weil es bereits auf ähnlichen Daten trainiert wurde. [33, 55]

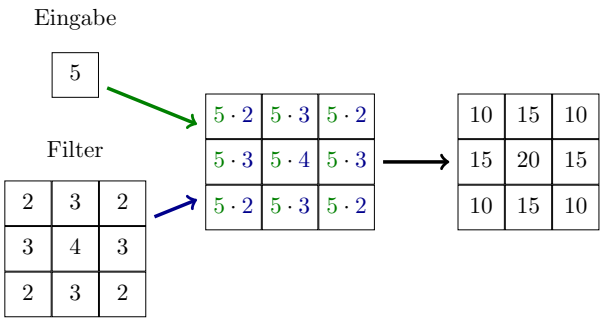


Abbildung 2.5: Einfaches Beispiel einer transponierten Faltung

Eine Alternative zu vortrainierten Modellen ist es, der Eingabe zusätzliche Informationen hinzuzufügen. Meier et al. [29] verwenden ein CNN Encoder - Decoder Modell, dessen Eingabe nicht nur aus Grauwert-Bildern von den Artikeln besteht, sondern auch aus einer Texterkennung (Optical Character Recognition) (OCR)-Version des Bildes. Die Verwendung dieser zusätzlichen Eingabe hat den Vorteil, dem Netz zusätzliche Informationen zur Verfügung zu stellen. Die-

se Informationen müssen nicht von dem Netz selber ermittelt werden. Dadurch ist es möglich, ein kleineres Modell zu verwenden. Dieser Ansatz hat aber den Nachteil, dass es deutlich schwieriger ist, vortrainierte Modelle zu finden, die für das Transferlernen verwendet werden können. Die Gewichte von auf Image-Net [10] vortrainierten Modellen sehen beispielsweise keine weitere Eingabe als das Bild vor. Transferlernen kann allerdings nur schwer angewendet werden, wenn die Architekturen nicht übereinstimmen. [29]

Abbildung 2.6: Beispiel für die Ausgabe des Modells mit 10 Klassen. Jeder Ausgabe Kanal des Modells entspricht einer Farbe in der Ausgabe. Nicht alle 10 möglichen Klassen sind hier abgebildet. Die volle Ausgabe ist in Abschnitt A.3 zu finden

2.3.4 Einbeziehung semantischer Informationen

Ein weiterer Weg, dem Modell zusätzliche Informationen zur Verfügung zu stellen, ist, semantische Informationen bezüglich der Wörter zu der Eingabe hinzuzufügen. Barman et al. [4] fügen im Encoder eines U-Nets noch für jedes Pixel einen Wort-Vektor hinzu. Dieser Vektor repräsentiert semantische Informationen, die von einem NLP Modell wie word2vec [42] gelernt wurden. Je ähnlicher die Vektoren von zwei Wörtern sind, desto größer ist die semantische Ähnlichkeit zwischen diesen beiden Wörtern. Durch das Hinzufügen dieser Information an die Pixel-Daten der Eingabe kann jedem Pixel auch eine semantische Information zugeordnet werden und im Lernprozess berücksichtigt werden. Dazu ist es allerdings nötig, die Bilder vorher per OCR zu verarbeiten. Und semantische Informationen zu extrahieren. [4]

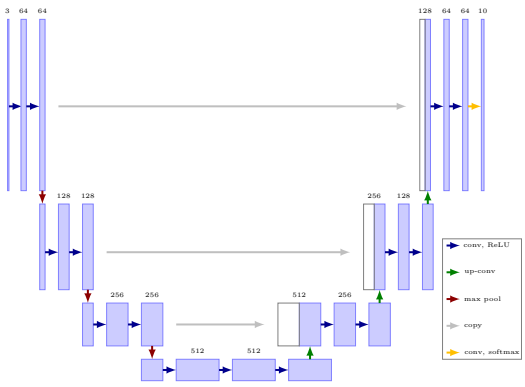


Abbildung 2.7: Beispiel einer U-Net Architektur im Stil von [43]

Furmaniak et al. [16] teilen die Eingabe zunächst in Textblöcke auf, indem

auf jeder Seite ein OCR-Modell ausgeführt wird. Um aus dieser groben Segmentierung ganze Artikel zu erkennen, verwenden sie zwei Ansätze. Die erkannten Textblöcke können nach semantischen Kriterien verglichen werden, wonach semantisch ähnliche Texte dem gleichen Artikel zugeordnet werden. Eine andere Möglichkeit ist das Einbeziehen geometrischer Eigenschaften der Textblöcke, wie die Position und Ausrichtung, um Textblöcke zu identifizieren, die zum gleichen Artikel gehören. Furmaniak et al. kombinieren beide Ansätze, indem das geometrische Modell sowohl auf annotierten Daten trainiert wird als auch auf den Resultaten der semantischen Ähnlichkeit. Diese ist zwar weniger genau, liefert aber genug Informationen, um das geometrische Modell zu trainieren. [16]

2.3.5 Evaluation von Deep Learning Modellen

Liebl et al. [25] evaluieren Deep Learning Architekturen. Diese Erkenntnisse sind für diese Arbeit relevant, weil auch hier mit reinen Bildeingaben ohne zusätzliche semantischen Informationen oder ähnliches gearbeitet wird. Untersucht werden die Hyperparameter Modell, Kostenfunktion, Optimierer und Auflösung der Bilder. Unterschieden werden Aufgaben zur Unterscheidung von Text und Hintergrund sowie Separatoren und Hintergrund. Die Kreuz Entropie Verlustfunktion [60] liefert die besten Ergebnisse für Text-Erkennung. Dice-Loss (Intersection over Union) hingegen ist besser bei der Erkennung von Separatoren. [25, 12, 49]

Als Optimierer eignet sich Adam mit dekoppeltem weight decay. Die Architekturtypen U-Net und FPN (Feature Pyramid Networks) wurden verglichen, es ist hier allerdings kein signifikanter Unterschied zu erkennen. Diese beiden liefern aber bessere Ergebnisse als andere Architekturen, die noch vor der Evaluation aussortiert wurden. Das Modell Inception-ResNet-v2 zeigt die besten Ergebnisse über alle Aufgaben. Das in dieser Arbeit verwendete Modell DhSegment(Unterabschnitt 3.3.1) wird ebenfalls untersucht. Die Ergebnisse von dhSegment sind in dieser Evaluation für Text-Erkennung schlechter und für Separator-Erkennung besser als die der meisten anderen Modelle. Die Untersuchungen zeigen bei den meisten Modellen bessere Ergebnisse bei hoher Auflösung der Bilder. Es werden verschiedene Versionen des Cropping untersucht, die keine quadratischen Crops ausschneiden, sondern rechteckige, die sich in der vertikalen oder horizontalen Richtung Überlappen (oder beides). Es wird festgestellt, dass rein vertikales Cropping am robustesten ist. Die Untersuchungen zeigen, dass schon ein Datensatz mit 20-30 Bildern gute Ergebnisse liefert, wenn der Encoder vortrainiert ist. Das final verwendete Modell erreicht eine Pixel-Genauigkeit von 98 %. [25]

2.4 Transformer

2.4.1 Attention

CNN Deep-Learning Architekturen zeigen immer wieder gute Ergebnisse für die Verarbeitung von Bildern. Der Vorteil der Faltungen besteht darin, dass lokale

Zusammenhänge stets erhalten bleiben und dadurch eine grundlegende Eigenschaft von Bildern nicht erst gelernt werden muss. CNNs besitzen ein rezeptives Feld fester Größe. Zusammenhänge, die über die Größe des rezeptiven Feldes hinausgehen können dadurch nicht erfasst werden. Das kann bei stark variierenden Eingaben wie kurzen und langen Texten bei der Verarbeitung natürlicher Sprache (Natural Language Processing) (NLP) zu Problemen führen. Die Herausforderung bei menschlicher Sprache besteht darin, dass sich die Bedeutung von Wörtern oder Sätzen abhängig vom Kontext ändern kann. Kontext muss bei langen Texten auch über große Distanz berücksichtigt werden können. Auf Aufmerksamkeit (Attention) basierende Modelle dominieren im NLP Bereich. [29, 33, 25, 24, 53]

Vaswani et al. stellen im Artikel Attention is all you Need [53] Transformer Modelle vor, die es ermöglichen, Abhängigkeiten über weite Entfernung zu beachten. Der Kern von Transformer-Modellen ist der Aufmerksamkeits-Mechanismus. Dieser verwendet Ähnlichkeiten zwischen Vektor-Repräsentationen von Wörtern, um allen Wörtern der Eingabe eine Relevanz zuzuordnen. Um zum Beispiel eine Frage zu beantworten, ist es hilfreich die Wörter aus der Frage zu identifizieren, die besonders relevant für die Frage sind. Um zu ermitteln, welche Wörter am relevantesten sind, wird die Frage oder ein Satz mit sich selbst verglichen. Dabei sollen die Vektoren, die die Wörter repräsentieren, einander ähnlich sein, wenn zum Beispiel der Kontext oder die Bedeutung ähnlich ist. Der Vergleich von „Geld“ mit „Bank“ sollte beispielsweise nur dann einen hohen Ähnlichkeitswert ergeben, wenn es sich nicht um eine Parkbank handelt. Das Vergleichen geschieht durch eine Matrix-Multiplikation, die auch als eine Reihe von Skalarprodukten interpretiert werden kann. Dadurch ergeben sich Ähnlichkeitswerte zwischen allen Vektoren von Wörtern einer Frage oder eines Satzes. [53]

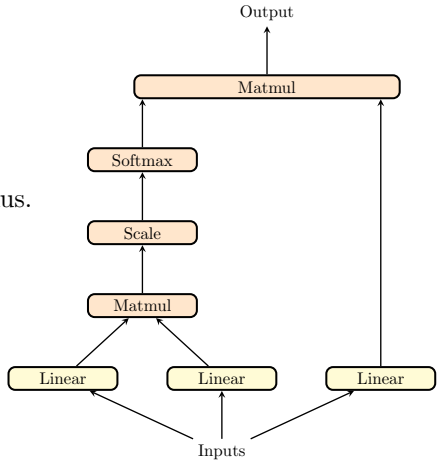


Abbildung 2.8: Architektur eines einzelnen Self-Attention Heads [53]

Abbildung 2.8 zeigt die Architektur eines Attention Heads. Die Eingabe wird 3-mal verwendet, wobei eine vollständig vernetzte Schicht die Eingabe auf eine feste Länge abbildet und gelernte Gewichtungen ermöglicht. Diese 3 Versionen der Eingabe werden Query, Key und Values genannt. Query und Key werden für die Matrixmultiplikation(Matmul) verwendet. Die Zeilen der resultierenden Matrix enthalten für jeweils ein Query-Wort Ähnlichkeitswerte aus dem Skalarprodukt mit allen Keys. Die Softmax Funktion bildet diese Werte pro Zeile auf

Wahrscheinlichkeiten ab, sodass für jede Query Prozentzahlen bekannt sind, welche Keys wie relevant für die Bedeutung der Query-Wörter sind. Vor der Anwendung der Softmax Funktion wird die Matrix skaliert. Der Hintergrund hierfür ist ähnlich der Batch-Normalisierung bei CNNs. Abhängig von der Dimension der Keys, also der Länge der Vektorrepräsentationen der Key-Wörter können die Werte der Skalarprodukte so groß werden, dass die Softmax Funktion gesättigt wird. Das führt dazu, dass der Gradient klein ist und Training per Backpropagation nur schwer möglich ist. Deswegen werden die Zeilen der Matrix abhängig von der Dimension d_k der Key-Vektoren mit $\frac{1}{\sqrt{d_k}}$ skaliert. Die prozentualen Relevanz-Werte werden zum Schluss verwendet, um die Value-Wörter zu gewichten. Dadurch ergibt sich eine Matrix, die nicht mehr Prozentwerte enthält, sondern die nach dieses Werten gewichteten Vektorrepräsentationen der Value-Wörter. Es lassen sich also Query-Wörtern direkt relevante Value-Wörter zuordnen. [53]

Analog zu mehreren Neuronen in einer Schicht oder mehreren Feature Maps in einer Faltungs-Schicht, werden mehrere Attention-Heads pro Transformer Schicht verwendet. In Abbildung 2.9 ist ein Beispiel mit 3 parallelen Attention-Heads zu sehen. In diesem Fall kann die Aufmerksamkeit also nach 3 verschiedenen Kriterien auf relevante Teile der Eingabe fokussiert werden. Die Ergebnisse der Attention-Heads werden zusammengefügt und wieder mit einer vollständig vernetzten Schicht auf eine feste Größe projiziert. Eine Transformer-Schicht (Abbildung 2.10) besteht aus einem Multi-Head-Attention Block und einer vollständig vernetzten Schicht. Analog zu Resnet [21] sind beide durch Skip-Connections umgehbar. Außerdem werden die Werte nach dem Addieren mit den Werten aus der Skip-Connection normalisiert. Ein Transformer-Encoder besteht aus mehreren solchen Schichten. Ein Transformer-Decoder bezieht zusätzlich zu den Eingaben auch die bisherige Ausgabe mit ein. Das ist bei NLP nötig, weil die Ausgabe Wort für Wort erstellt wird und der Satz Sinn ergeben muss. Im ersten Teil des Encoders wird eine maskierte Multi-Head Attention durchgeführt, weil die Wörter der Ausgabe nur von den Wörtern abhängen dürfen, die zu einem vorherigen Zeitpunkt ausgegeben wurden. In jedem Zeitschritt wird aber die gesamte bisherige Ausgabe per Self-Attention verarbeitet. Das Ergebnis des Encoders wird nun per Cross-Attention auf die Ausgabe angewendet, die durch den Masked-Multi-Head-Self-Attention Block verarbeitet wurde. Cross-Attention bedeutet, dass die Wahrscheinlichkeiten wie

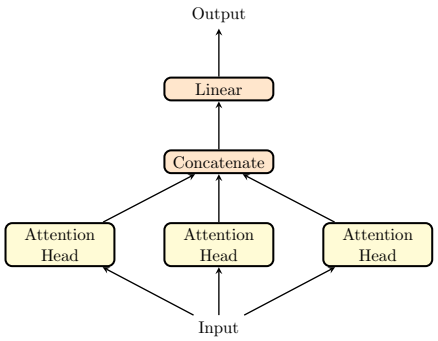


Abbildung 2.9: Multi-Attention-Head aufgebaut aus mehreren Attention-Heads

bisher durch Queries und Keys per Matrix-Multiplikation berechnet werden. Diese werden allerdings auf Values angewendet, die von der Ausgabe kommen. [53]

2.4.2 Vision Transformer

Statt Multi-Head-Attention zu verwenden ist es auch möglich, Single-Head-Attention für Vision Tasks einzusetzen. Wang et al. [54] verwenden Single-Head Attention Layers als Bauteil eines CNNs. Diese werden zwischen ResNet-Schichten eingebaut. Dies verbessert die Performance im Vergleich zu ResNet ohne Attention Schicht. [54]

Anstatt Faltungen mit Aufmerksamkeit zu unterstützen, kann sie auch durch Self-Attention ersetzt werden. Ramachandran et al. [39] verwenden statt Faltungen, lokale Self-Attention Blöcke. Die Berechnung der Attention läuft also nicht über die gesamte Eingabe, sondern nur in einem lokalen Bereich. Das hat den Vorteil, dass nicht alle Kombinationen der gesammten Eingabe beachtet werden, sondern jeweils nur die lokale Umgebung des aktuellen Pixels. Dadurch können größere Bilder verarbeitet werden, ohne die asymptotische Laufzeit in Abhängigkeit der Bildgröße mehr als Linear wachsen zu lassen. [39]

Die Verwendung von Multi-Head-Attention-Layern in Transformern hat sich auch bei Vision Tasks durchgesetzt. Dosovitskiy et al. stellen den ersten Vision Transformer (ViT) vor [13]. Vision Transformer benötigen im Gegensatz zu NLP Modellen keinen Decoder mit einem maskierten Aufmerksamkeits-Modul, weil die Ausgabe nur vom Eingabe-Bild abhängt und keine Teile der Ausgabe voneinander abhängen. Ein Vision-Transformer für die Klassifikation von Bildern besteht aus einem Transformer Encoder und einem Klassifikator, welcher zum Beispiel ein herkömmliches Neuronales Netz sein kann. Das Eingabe-Bild wird vor dem Encoder in Teilbilder zerlegt (Cropping). Jeder Crop wird so behandelt wie ein Wort bei NLP Transformern. Das bedeutet, dass die Ähnlichkeit zwischen allen Crops eines Bildes so verglichen wird wie die Wörter eines Satzes. Der Transformer ist also in der Lage, die Aufmerksamkeit auf bestimmte Crops zu fokussieren. Jeder Crop wird in einen Vektor umgewandelt. Dieser wird Token genannt und vom Transformer verarbeitet. Anders als CNNs, muss ein Transformer Strukturelle Informationen über Bilder erst lernen. CNN können durch die 2d Filter die Struktur von Bildern natürlicherweise miteinbeziehen. Deswegen müssen Vision Transformer auf sehr großen Datenmengen vortrainiert werden. Diese werden verwendet um unüberwacht grundlegendes Verständnis über Bilder zu lernen. Ein Beispiel dafür ist, dass Teile des Bildes geschwärzt werden und das Modell diese wiederherstellen soll. Durch solche Verfahren können grundlegende Dinge gelernt werden, ohne auf manuell annotierte Daten angewiesen zu sein. Danach kann auf einem Datensatz wie ImageNet [10] trainiert werden. Auch ist es möglich, einen Transformer zu trainieren, indem ein CNN Modell als Lehrer eingesetzt wird. Danach wird noch auf annotierten Daten trainiert. [24, 13]

Im Gegensatz zu NLP Modellen ist bei Bildern die Auflösung wichtig. Die Vektorrepräsentationen von Wörtern werden als Tokens bezeichnet. Ein Token besteht aus den Werten eines Vektors, der ein Wort repräsentiert. Bei Vision-Transformern stammen die Werte eines Tokens allerdings aus den Pixeln eines Bildausschnitts. Wenn die Anzahl und Dimension der Token in den verschiedenen Transformer-Schichten gleich bleibt, ist es nicht möglich, feine Details zu erfassen. Für detaillierte Segmentierung haben sich Modelle als effektiv erwiesen, bei denen die Anzahl der Tokens pro Schicht allmählich verringert und die Dimension erhöht wird. Die Anzahl der Token wird reduziert, indem benachbarte Tokens verschmolzen werden. Das hat einen ähnlichen Effekt, wie bei CNNs, wo in späteren Schichten wenige Pixel pro Feature Map vorhanden sind, aber dafür viele Featuremaps verwendet werden. Diese Multi-Stage Architekturen werden sowohl als vollständige Transformer Architekturen verwendet als auch als Teil eines hybriden Designs aus Transformern und Faltungen. Der Nachteil von Multi-Stage Architekturen ist die hohe benötigte Rechenzeit. [13]

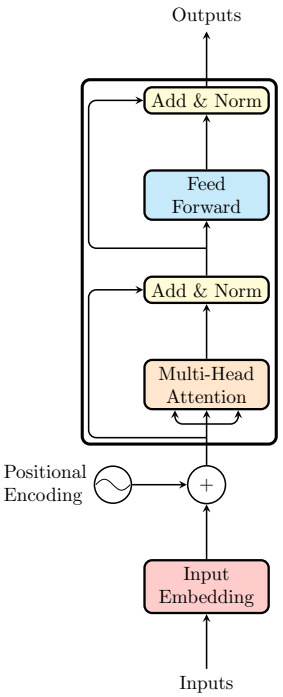


Abbildung 2.10: Aufbau eines Transformer-Encoders.

Um Transformern Informationen über die Position der Teilbilder zur verfügung zu stellen, werden den Repräsentationen Positional Embeddings hinzugefügt. Es gibt sowohl gelernte als auch feste Embeddings. Gelernte Embeddings bestehen aus einer Matrix fester Größe, die für jedes Teilbild einen Vektor enthält, der Informationen über die Position enthält. Diese Matrix wird auf die verdeckten Repräsentationen des Transformers aufaddiert und beim Training angepasst. Feste Embeddings werden zum Beispiel aus Sinus Funktionen unterschiedlicher Frequenz berechnet. Der Nachteil an einer Matrix fester Größe ist, dass die Anzahl der Teilbilder immer so groß sein muss wie im Training. [53]

Für diese Arbeit kommen nur Architekturen infrage, die in der Lage sind, Segmentierungs-Aufgaben zu lernen. Der ViT Transformer von Dosovitskiy et al. [13] verwendet ein Multi Layer Perzeptron (MLP), um das Bild zu klassifizieren und ist damit nicht geeignet. (MLP)s sind geeignet einem Bild eine Klasse zuzuweisen, allerdings ist die Zuweisung einer Klasse für jeden Pixel mit (MLP)s nicht praktikabel. Strudel et al. [47] verwenden einen vollständigen Transformer, um Bilder zu segmentieren. Dabei wird ein Transformer-Encoder verwendet, dessen Ergebnisse von einem Mask Transformer verarbeitet werden. Der Mask Transformer erhält sowohl die Ergebnisse für jeden Bild-Ausschnitt vom Encoder als auch eine lernbare Vektorrepräsentation für jede Klasse als

Eingabe. Die Ausgabe besteht aus der gleichen Anzahl an Token. Um Masken für jede Klasse zu erhalten, werden die Tokens der Klassen per Skalarprodukt auf die Tokens der Bild -Ausschnitte angewendet. Diese Masken werden hoch skaliert um die Vorhersage pro Bildausschnitt zu erhalten.[47]

Eine andere Variante ist die Verwendung eines hybriden Modells aus CNN und Transformer Teilen. Der Encoder eines U-Nets wird durch einen Transformer Block mit 12 Schichten ergänzt. Das hat den Vorteil, dass der Decoder sowohl Zugriff auf globale Kontextinformationen hat, die der Transformer extrahiert, als auch Zugriff auf die durch den CNN Encoder extrahierten lokalen Features, die beim Transformer verloren gehen. [6]

Ein Einsatzgebiet von Vision Transformern ist der Bereich der medizinischen Bild-Verarbeitung, um zum Beispiel Krebszellen automatisch zu erkennen. In diesem Feld werden hybride Architekturen bevorzugt, weil sowohl detaillierte lokale Informationen als auch globale Kontextinformationen wichtig sind. Außerdem sind reine Transformer-Modelle rechenaufwendiger als hybride Modelle. Aufgrund der geringen Komplexität werden Single-Scale Architekturen Multi-Scale Architekturen vorgezogen. Single-Scale Architekuren verstärken die Notwendigkeit von detaillierter Feature Extraktion durch CNN Schichten, weil die Auflösung von Single-Scale Transformern begrenzt ist. Die meisten hybriden Modelle verwenden Transformer im Encoder oder zwischen Encoder und Decoder. Transformer im Decoder sind noch nicht gut untersucht. [46]

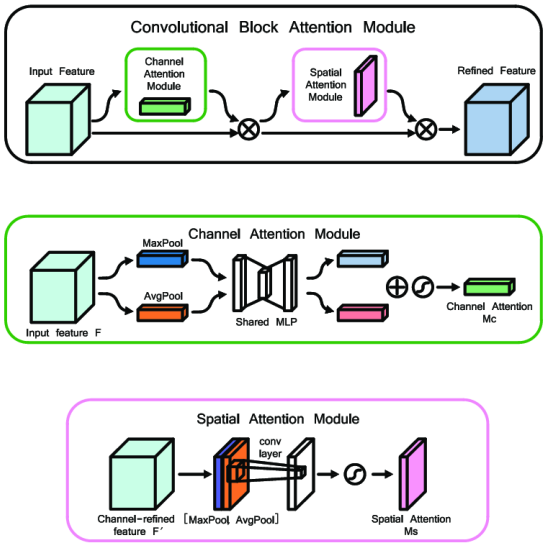


Abbildung 2.11: Aufbau des Convolutional Block Attention Module (CBAM) [58]

2.4.3 Convolution Block Attention Module

Woo et al. stellen das Convolutional Block Attention Module (CBAM) vor, welches das Konzept der Aufmerksamkeit einerseits durch das Channel Attention Module (CAM) auf die verschiedenen Kanäle der Eingabe und andererseits mit dem Spatial Attention Module (SAM) auf bestimmte räumliche Regionen anwendet. Das CAM berechnet für jeden Kanal eine Gewichtung und das SAM berechnet für jedes Pixel über alle Kanäle eine Gewichtung, die dann verwendet werden, um sich auf die relevantesten Stellen zu fokussieren. Beide Module verwenden dafür, im Gegensatz zu Attention Heads in Transformern, Pooling Ope-

rationen statt dem Skalar Produkt. Das Ergebnis des Pooling wird mit einem MLP (CAM) oder einer Faltung (SAM) verarbeitet, um die Gewichte zu berechnen. Beide verwenden im ersten Schritt sowohl Max- als auch Avg-Pooling. [58]

Das in Abbildung 2.11 dargestellte CAM wendet die beiden Pooling Operationen für jeden Kanal separat an. Dadurch entsteht ein Vektor, der den maximalen Wert jedes Kanals enthält und ein weiterer, der den Durchschnitt der Pixel in jedem Kanal enthält. Um daraus Gewichte für die Kanäle zu berechnen, wird ein einschichtiges MLP verwendet, das sowohl für den Max- als auch für den Avg-Pooling Vektor verwendet wird. Die beiden Ergebnisse werden dann addiert. [58]

Das SAM (Abbildung 2.11) wendet Pooling entlang der Kanäle an, sodass eine Feature Map entsteht, deren Pixel entweder den maximalen Wert aus allen Kanälen enthalten oder den Durchschnitt. Diese beiden Feature Maps werden mit einer Faltung verarbeitet, um eine einzelne Feature Map zu erhalten. Im CBAM Modul wird zunächst die Aufmerksamkeits-Gewichtung der Kanäle (CAM) berechnet und auf die Eingabe per elementweise Multiplikation angewendet. Dann wird die räumliche Gewichtung (SAM) berechnet und angewendet.[58]

2.5 Daten

Entscheidend für das Training sind, neben der Architektur, vor allem die Daten. Bei der Segmentierung von Zeitungsseiten muss zu dem Bild einer Seite eine annotierte Version vorliegen, in der die Regionen markiert sind. Mit diesem Ziel-Bild kann ein Modell dann trainiert werden, indem der Fehler der Vorhersage zur gewünschten Ausgabe berechnet wird. Der Erfolg des Trainings hängt stark von der Qualität der Annotationen ab. Wenn Widersprüche in den Annotationen zu finden sind, kann das Modell keinen Lerneffekt aus diesen Annotationen ziehen. Zum Beispiel Überschriften, die fälschlicherweise als normaler Text markiert wurden, wirken dem gewünschten Lerneffekt, diese beiden Regionen zu unterscheiden, entgegen. Die dieser Arbeit vorhergegangene Projektarbeit von Kerkfeld et al. [33] verwendet zwei Datensätze, um ein Modell für die kölnische Zeitung (zeitpunkt.nrw) zu trainieren. Der erste Datensatz vom Europeana Newspaper Project umfasst 500 annotierte Zeitungsseiten aus europäischen Zeitungen in unterschiedlichen Sprachen [8]. Dieser wurde im Sinne des Transferlernens zum Pretraining verwendet, bevor auf dem während der Projektarbeit manuell annotierten Datensatz der Kölnischen Zeitung trainiert wurde. Die beiden Datensätze mussten getrennt voneinander trainiert werden, weil sich die Regeln für die Annotationen zwischen den Datensätzen unterscheiden. Die Annotationen des Europeana Newspaper Project erfassen einige Elemente nicht, die für die Weiterverarbeitung bei der Kölnischen Zeitung für nötig erachtet werden. Nur die Klassen Artikel, Überschrift und Separator (Vertikal) aus den Annotationen der Kölnischen Zeitung finden sich im Europeana Datensatz in der Form wieder, dass diese für das Training übernommen werden

können. [33]

Der ENP Datensatz für historische Zeitungen repräsentiert europäische Zeitungen mit Zeitungen in 15 unterschiedlichen Sprachen. Martinek et al. [28] verwenden den deutschsprachigen Teil des Datensatzes, um Segmentierungsnetze vorzutrainieren. Diese bilden die Grundlage für das Durchführen einer OCR, die trotz einer geringen Menge von annotierten Seiten erfolgreich sein soll. Sie testen die OCR an einem Datensatz, der lediglich 10 transkribierte Seiten in Fraktur-Schrift enthält. Dafür werden synthetischen Daten für das Pretraining verwendet. Einerseits werden Bilder verwendet, die nur ein einzelnes Zeichen enthalten und andererseits Bilder einer künstlich generierten Zeitungsseite. Die Experimente zeigen, dass das Pretraining auf synthetischen Daten mit anschließendem Training auf den echten Daten eine Verbesserung in der Performance zur Folge hat. [28]

Pack et al. nutzen den ENP Datensatz, um Augmentierungs-Techniken zu evaluieren. Dafür werden OCR Modelle auf augmentierten Versionen des ENP Datensatzes trainiert. Diese werden dann mit dem Training auf dem Original-Datensatz verglichen.[38]

Der Deutsche Reichsanzeiger erschien bis 1945 als amtliches Presseorgan des Deutschen Reichs und Preußen. Die UB-Mannheim stellt 197 annotierte Seiten bereit. Die Annotationen liegen im PAGE-XML Format[48] vor und enthalten Transkriptionen [40].

Eine weitere Möglichkeit an Trainingsdaten zu gelangen ist, diese synthetisch herzustellen. Monier et al. verwenden einen synthetischen Datensatz aus 10k Bildern, die Seiten von historischen Dokumenten ähnlich sehen. Dafür werden zufällige Hintergründe von leeren Seiten verwendet, die zufällig mit Elementen wie Text und Bildern gefüllt werden. Das erlaubt die automatische Erstellung detaillierter Annotationen. [31]

Kapitel 3

Implementierung

3.1 Training

3.1.1 Allgemein

Das Training wird mit Pytorch [15] durchgeführt. Es erfolgt in Epochen, wobei jede Epoche das Verarbeiten des gesamten Trainings-Datensatzes umfasst. Die Daten sind im Verhältnis 90% 5% 5% in Trainings-, Validations- und Test-Datensätze aufgeteilt. Der in dieser Arbeit verwendete Datensatz mit 402 Bildern der Kölnischen Zeitung(Abschnitt 4.2) wird in Batches aufgeteilt. Bei Teilbildern von 512x512 Pixeln und einer Batchgröße von 64 Teilbildern, besteht der Trainingsdatensatz aus 318 Batches und Validations- wie Test-Datensatz aus jeweils 17 Batches. Für jeden Batch wird die Ausgabe des Models berechnet. Abhängig vom Fehler dieser Ausgabe im Vergleich mit den Targets wird das Model schrittweise angepasst. Verwendet werden die Verlustfunktionen Kreuz Entropie und Focal Loss. Beide erhalten eine Gewichtung für die verschiedenen Klassen. Weil der Focal Loss in diesem Fall keine guten Ergebnisse liefert, wird für das finale Training die Kreuz Entropie verwendet(Unterabschnitt 4.6.1).

3 Mal pro Epoche wird eine Validationsrunde durchgeführt. Das bedeutet, den gesamten Validationsdatensatz vom Model vorhersagen zu lassen und den Fehler, sowie Jaccard Score(Intersection over Union), Genauigkeit(Accuracy) und Multi Class CSI zu berechnen(Unterabschnitt 3.1.3). Im Laufe des Trainings wird nach dem Early Stopping Verfahren, anhand von den in der Validation errechneten Loss-, Genauigkeit- und Jaccard-Werten, das beste Modell gespeichert und aktualisiert. Das geschieht immer, wenn die Evaluation des aktuellen Modells bessere Werte produziert als bisher(Genaueres dazu in Kapitel 3.3 vom Projektbericht [33]). Am Ende des Trainings wird eine Testrunde durchgeführt. Das entspricht einer Validation auf dem Test-Datensatz.

3.1.2 Daten

Die Daten der Kölnischen Zeitung liegen als Bilder im jpg Format und Annotations-Daten im PAGE XML Format [48] vor. Beim Europeana Datensatz handelt es sich um tif Bilder und ebenfalls XML Dateien (Siehe auch Abschnitt 4.2). Bevor das Training durchgeführt werden kann, müssen die XML-Daten umgewandelt werden. Die XML-Dateien enthalten Koordinaten von Polygon Ecken, die jeweils einer bestimmten Klasse zugeordnet sind. Vor dem Training werden alle Polygone in ein 2d numpy array [34] gezeichnet, welches die gleiche Größe hat wie das ursprüngliche Bild. Die Implementierung davon stammt aus der Projektgruppe Segmentierung von Zeitungsseiten (Siehe Abschnitt 3.2.1 des Projektberichts). [33]

Das Laden der Daten wird von einer eigens implementierten Dataset Klasse durchgeführt. Die Bilder werden mithilfe von PIL[7] geladen. Die aus den XML-Dateien umgewandelten Ziel-Daten (Targets) werden als numpy Dateien abgespeichert und aus diesen am Anfang des Trainings wieder geladen. Die Daten werden dann mit der Preprocessing Klasse in mehreren Schritten für das Training vorbereitet:

- Die Eingabe-Bilder werden in numpy arrays umgewandelt und die letzte Dimension wird an die erste Stelle verschoben. Die Eingabe-Bilder liegen nach dem Laden mit PIL im Format [H,W,C] (Höhe, Breite, Kanäle) vor, Pytorch benötigt jedoch das Format [C,H,W].

- Scaling: Die Bilder können um einen vorgegebenen Faktor herunterskaliert werden. Das geschieht mit der resize Methode von PIL. Ein Faktor von 1 bedeutet, dass die Größe nicht verändert wird, während 0,5 beide Seiten um die Hälfte herunterskaliert. Für die Bilder wird der PIL BICUBIC Algorithmus verwendet und für die Targets PIL NEAREST.

- Cropping: Die Bilder werden in Crops einer vorgegebenen Größe unterteilt, um größere Batch-Größen zu ermöglichen. Dafür wird die view_as_window Methode von scikit-image verwendet [51]. Wichtig ist dabei, dass die Seitenlängen 2er-Potenzen sind. Die Verarbeitung durch die Modelle enthält maxpooling Operationen, die das Bild um die Hälfte herunterskaliert. Nur wenn die Dimension in diesem Schritt durch 2 teilbar ist, kann dies korrekt durchgeführt werden.(Genaueres zum Cropping, siehe Projektbericht Abschnitt 3.2.2 [33])

Die Bilder der Zeitungsseiten liegen als RGB Bilder vor, obwohl für die Layoutanalyse Grauwerte ebenfalls ausreichen würden. Eine Umwandlung in Grauwert Bilder ist nicht sinnvoll, weil die von ResNet[21] übernommenen Gewichte 3 Eingabekanäle vorsehen (Abschnitt 3.1). Es ist nicht sinnvoll, zum Beispiel die Gewichte für den roten Kanal auf die Grauwerte anzuwenden. Deswegen werden die Bilder im RGB Format verarbeitet. Außerdem wäre der Gewinn durch eine Grauwertumwandlung gering. Lediglich die erste Schicht der Modelle Abschnitt 3.3 würde davon profitieren, indem weniger Parameter nötig sind.

Die verarbeiteten Daten werden in ein array gespeichert, dass zu Beginn des Trainings den gesamten Datensatz in Form von Crops enthält. Bei dem verwendeten Datensatz ist der Arbeitsspeicher ausreichend, um den gesamten Datensatz zu laden. Die ursprüngliche Vorgehensweise, die vorverarbeiteten Daten getrennt abzuspeichern, wurde wegen begrenztem Speicherplatz verworfen. Beim Laden der Bilder während des Trainings werden zufällig Augmentierung auf die Daten angewendet. Dabei handelt es sich um horizontale und vertikale Spiegelungen, um Rotationen, Schwärzung zufälliger Regionen, Blurring und dem hoch- oder Herunterskalieren der Daten, um das Modell robust gegen Veränderungen der Auflösung im Vergleich zu den Trainingsdaten zu machen.

3.1.3 Validation und Test

Während des Trainings werden 3 Mal pro Epoche Validationsrunden durchgeführt. Das enthält die Evaluation auf dem gesamten Validationsdatensatz. Am Ende des Trainings wird eine Validation auf dem Testdatensatz durchgeführt. Für eine Validation werden die Eingabe-Bilder durch das aktuelle Modell verarbeitet und mithilfe der Targets wird der Loss, die Genauigkeit, der Jaccard Score und der Multi Class CSI berechnet. Genauigkeit und Jaccard Score werden mit torchmetrics [32] berechnet. Bei beiden ist die Option „weighted“ eingetragen, die dafür sorgt, dass die seltenen Klassen in dem unausgeglichenen Datensatz gewichtet werden. Andernfalls erreicht die Validation gute Werte, weil die beiden häufigsten Klassen (Hintergrund und Artikel) gute Werte erhalten. Die dann noch verbleibenden Klassen nehmen einen geringen Teil der markierten Pixel ein. Um die einzelnen Klassen getrennt voneinander zu betrachten, ist die Genauigkeit (Gleichung 3.1) ungeeignet. Das liegt daran, dass bei einer selten vorkommenden Klasse hohe True Negative Werte zu sehr hohen Genauigkeit Werten führen, obwohl es vergleichsweise wenige True Positives gibt. Wenn zum Beispiel eine Klasse 1% der Daten ausmacht, würde es ein Ergebnis von 99% Genauigkeit geben, wenn einfach alle Pixel als nicht zugehörig markiert werden. Aufgrund des unausgeglichenen Datensatzes ist dieser Wert aber nicht aussagekräftig. Der Critical Sucess Index (Gleichung 3.2) ignoriert die True Negatives. Dadurch entstehen verwertbare Werte für die einzelnen Klassen. Wenn es nur True Negatives in der Evaluation eines Batches gibt, entsteht dadurch aber eine Division durch 0. Dies wird abgefangen und der entsprechende Wert wird ignoriert. Nur True Negatives kommen vor, wenn in dem fraglichen Batch kein einziges Pixel dieser Klasse im Target und der Vorhersage vorkommt.

$$Genauigkeit = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

$$CSI = \frac{TP}{TP + FP + FN} \tag{3.2}$$

Um mit zeitlichen Begrenzungen auf dem Server umzugehen, werden sowohl die Logs von Tensorboard [27], als auch das aktuelle Modell, sowie die Anzahl der Epochen und bisher abgearbeiteten Trainings-Batches am Ende jeder Epoche

abgespeichert. Das Training wird dann in einem neuen Prozess fortgesetzt, wobei die abgespeicherten Informationen und das Modell geladen werden. Dadurch wird das Logging in einem Ordner fortgesetzt, obwohl ein neuer Prozess das Training fortsetzt. Andernfalls wäre es nötig gewesen, mehrere Logdateien zu verbinden, was ein erheblicher Aufwand gewesen wäre.

3.2 Vorhersage

3.2.1 Allgemein

Ein trainiertes Modell soll dazu verwendet werden, unbekannte Zeitungsseiten zu verarbeiten. Die Layoutanalyse kann dann verwendet werden, um OCR Erkennung zu verbessern. Im Gegensatz zum Training werden bei der Vorhersage vollständige Bilder geladen und verarbeitet. Diese werden, wie auch in Unterabschnitt 3.1.2 beschrieben, geladen, jedoch nicht in Crops aufgeteilt. Im Training werden 2er-Potenzen als Seitenlängen verwendet, um Fehler beim Pooling zu verhindern. Das verwendete maxpooling reduziert die Bilddimensionen um die Hälfte, die Dimensionen müssen also durch 2 teilbar sein. In den Experimenten zur Vorhersage hat sich herausgestellt, dass die Eingabe für DhSegment mit und ohne CBAM mindestens 6 Mal durch 2 teilbar sein muss. Das Trans-Unet benötigt Bild-Dimensionen, die 9 Mal durch 2 teilbar sind. Entsprechend wird ein Padding durchgeführt, welches gegenüber des Koordinaten-Ursprungs(oben links) Pixel mit Wert (0,0,0) einfügt. Dies vereinfacht die spätere Berechnung der Polygon-Werte beim Slicing Export(Unterabschnitt 3.2.3). Die durch Softmax in Wahrscheinlichkeiten vorliegende Vorhersage des Modells wird durch torch.argmax verarbeitet, um eine Entscheidung herbeizuführen. Pro Pixel wird dadurch die Klasse mit der höchsten Wahrscheinlichkeit ermittelt. Pixel, bei denen die maximalen Wahrscheinlichkeiten unterhalb eines vorgegebenen Schwellwerts liegen, werden allerdings immer als Hintergrund interpretiert. Die Vorhersage kann dann zu Testzwecken als Bild mit Legende generiert werden.

3.2.2 Transkribus-Export

Transkribus ist die Software, in der die Annotationen für die Trainingsdaten ursprünglich erstellt werden. Die Vorhersage des Modells kann auch dorthin exportiert werden. Transkribus arbeitet mit Polygon Daten, die in einer XML-Datei abgespeichert sind. Um die Daten exportieren zu können, werden die Pixel-Daten in Polygone umgewandelt. Dazu wurde eine schon vorhandene Implementation angepasst, die Pixeldaten in Polygon-Daten überführt [2]. Zunächst werden Sub-Masken erstellt, die für jede Klasse aus Pixeln der Werte 1 oder 0 bestehen. 1 bedeutet der Pixel gehört zu der Klasse und 0 bedeutet der Pixel gehört nicht dazu. Bei 10 Klassen werden also auch 10 Sub-Masken erstellt. Mithilfe der skimage Funktion find countours [50] werden die Daten in jeder Submaske in Polygone umgewandelt. Zusätzlich vereinfacht die shapely polygon simplify Funktion [19] die Polygone. Das hat den Vorteil, dass die erkannten

Regionen geglättet werden. [2]

Für den Transkribus Export müssen Bilder, die ohne Annotationen in Transkribus vorhanden sind, exportiert werden. Der von Tranksribus erstellte Ordner enthält die Bilder, zu denen eine Vorhersage durchgeführt werden soll, sowie einen Ordner mit leeren XML-Dateien für jedes Bild. Die Polygon-Daten der Vorhersage werden mithilfe von BeautifulSoup [41] in ein entsprechendes XML-Format übertragen. Für jede erkannte Region wird ein Tag erstellt. Dabei handelt es sich entweder um eine „TextRegion“ oder eine „SeparatorRegion“ für Trennlinien. Diese speichern die Information über die Klasse der Region und enthalten den Tag „Coords“, welcher eine Liste von Koordinaten-Paaren der Polygon-Ecken enthält. Die so erstellten XML-Daten werden in die schon vorhandenen XML-Dateien eingefügt. Im Anhang Abschnitt A.9 ist ein Beispiel für Regionen im XML-Format.

Zusätzlich ist es möglich, über die XML-Tags eine Lese-Reihenfolge festzulegen. Um das Layout der Zeitungsseite vollständig zu verstehen, ist es auch wichtig, die Lesereihenfolge der erkannten Regionen zu ermitteln. Dazu wurde ein Algorithmus implementiert, der Bounding Box Informationen der Polygone nutzt, um eine Lese-Reihenfolge zu bestimmen. Voraussetzung für das erfolgreiche Ermitteln der Lese-Reihenfolge ist eine gute Erkennung von Regionen und insbesondere von großen Separatoren.

Der Algorithmus für die Lesereihenfolge arbeitet mit einer Liste von Bounding-Boxes aller erkannten Regionen. Diese enthält das Label der Klasse, die minx, miny, maxx und maxy Werte, der Bounding Box sowie den vertikalen Mittelwert der Region (Also die Mitte zwischen dem höchsten und niedrigsten Punkt). Der Algorithmus teilt die Zeitungsseite in Regionen zwischen großen horizontalen Separatoren ein. Diese wirken immer wie ein Seitenumbruch, weswegen das Problem darauf reduziert werden kann, Regionen dazwischen zu analysieren. Dazu werden alle Regionen aus der Liste identifiziert, die einerseits die korrekte Klasse haben und andererseits breit genug sind, um die gesamte Seite zu teilen. Solche Separatoren kommen in den Daten auch in kürzerer Form vor, die nicht in dieser Art verwendet werden sollen. Nun werden alle Regionen, deren vertikaler Mittelpunkt oberhalb des erkannten großen Separators ist, rekursiv weitergegeben. Sollte sich in diesen Regionen kein großer Separator mehr befinden, wird die Lesereihenfolge bestimmt. Ansonsten wird der Prozess so lange rekursiv durchgeführt, bis eine Liste ohne große Separatoren vorhanden ist.

Listen, die keine großen Separatoren mehr enthalten, werden nach der Summe des minx und miny Wertes sortiert. Dies wird als Heuristik verwendet, um die Region zu identifizieren, die sich am weitesten oben links befindet. Die Zeitungsdaten sind größtenteils nach Spalten aufgebaut. Die erste Region in der Lesereihenfolge wird also gefolgt von Regionen, die in der gleichen Spalte weiter unten liegen. Deswegen werden alle Regionen identifiziert, deren kleinster x-Wert kleiner ist, als der größte x-Wert der ersten Region dieser Spalte. Durch den Koordinatenursprung oben links bedeutet das, dass die fragliche Region sich nicht rechtsseitig von der ersten Region befinden darf, um zur gleichen Spalte gezählt zu werden. Alle Regionen in einer solchen Spalte werden nach der vertikalen Position sortiert und der Prozess wird mit allen verbleibenden Regionen

so lange wiederholt, bis alle Regionen in Spalten unterteilt wurden. Die so ermittelte Lese-Reihenfolge wird in die XML-Dateien für den Transkribus-Export eingefügt.

3.2.3 Slicing-Export

Um auch unabhängig von Transkribus arbeiten zu können, ist es möglich, erkannte Text-Regionen in Form einzelner Bilder zu exportieren. Die ausgeschnittenen Text-Regionen ermöglichen es, einen OCR-Prozess auf einzelnen Text-Regionen durchführen zu lassen. So ist sichergestellt, dass nur durch die Layout-Analyse als zusammengehörig erkannter Text von der OCR gelesen wird. Konkret kann die OCR nicht in die nächste Spalte springen, weil sich diese nicht mit auf dem Bild befindet, sondern separat ausgeschnitten wird.

Es sollen nur Textregionen ausgeschnitten werden, die eine bestimmte Größe haben. Alle Polygone, die groß genug sind, werden also in eine Maske gezeichnet. Anhand der Bounding Box wird nun für jedes Polygon der Bereich der Bounding Box in der Maske und dem Original-Bild ausgeschnitten. Die Maske wird auf den Ausschnitt des Original-Bildes angewendet, um Pixel auszusortieren, die nicht als Text-Region erkannt wurden. Alle Pixel, die in der Maske einen Wert von 1 haben, erhalten den originalen Farbwert. Alle Pixel, die in der Maske den Wert 0 haben, werden in der Farbe des Hintergrundes eingefärbt. Die so ausgeschnittenen Bilder werden nach dem Originalbild, aus dem sie stammen, sortiert und nach der Lesereihenfolge nummeriert. Die Ausschnitte werden dann, entsprechend in Ordnern pro Bild aufgeteilt, exportiert.

3.3 Modelle

3.3.1 DhSegment

Ausgangslage dieser Arbeit war die Projektgruppe Segmentierung von Zeitungsseiten [33]. Das dort verwendete DhSegment Modell hat bereits gute Ergebnisse erzielt und wird deswegen hier übernommen. Die Modelle Trans-Unet und DhSegment CBAM(Unterabschnitt 3.3.3 und Unterabschnitt 3.3.2) bauen auf DhSegment auf. Bei DhSegment handelt es sich um ein U-Net Modell, welches von Oliveira et al. [36] erfolgreich auf historischen Zeitungsseiten eingesetzt wurde. Der Encoder entspricht Resnet50 [21], bestehend aus 6 Teilen. Dementsprechend besitzt Dh Segment 6 Schichten. Zwischen dem Encoder und Decoder Teil liegt in den ersten 5 Schichten eine Skip-Connection. Die letzte Schicht überträgt ihre Ergebnisse direkt an den untersten Teil des Decoders. [36, 33, 21]

Der ResNet Encoder wird mit vortrainierten Gewichten aus einem Training mit dem ImageNet Datensatz initialisiert und im ersten Schritt während des Trainings nicht angepasst. Das dient dazu, die schon gelernten Gewichte nicht durch das Training abzuändern, weil diese bereits gute Resultate erzielen. Erst wenn die Gewichte des Decoders ebenfalls trainiert wurden, und das Training konvergiert, kann in einem weiteren Schritt auch der Encoder mit angepasst

werden. Das reduziert auch die nötigen Berechnungen bei der Backpropagation. Entsprechend werden auch die Daten bei der Vorhersage und dem Training mit dem Mittelwert und der Standartabweichung der ImageNet Daten [10] normalisiert.

3.3.2 DhSegment CBAM

Wie in Unterabschnitt 2.4.3 beschrieben, handelt es sich bei CBAM um Aufmerksamkeits Blöcke, die an dem Prinzip der Faltung angelehnt sind. Das CBAM besteht aus einem Aufmerksamkeits Modul für die Kanal-Dimension und einem für die räumlichen Dimensionen. Die daraus berechneten Gewichtungen werden auf die Eingabe angewendet. In Abschnitt 4.5 werden 2 Versionen verglichen. Die eine fügt eine Skip Connection ein, die die Eingabe auf das Ergebnis des CBAM addiert. Das entspricht einer Skip-Connection wie bei ResNet Modellen. [58]

Das DhSegment CBAM Modell fügt nach den unteren 4 Schichten des Encoders von DhSegment jeweils einen CBAM Block ein. Die CBAM Blöcke wurden von einer bestehenden Implementation übernommen [23]. Die bestehende Skip-Connection wurde allerdings entfernt und die beiden MLPs im CAM wurden auf ein MLP reduziert.

3.3.3 TransUnet

Das in Unterabschnitt 2.4.2 beschriebene TransUnet [6] wird in dieser Arbeit in abgewandelter Form eingesetzt. Der dort verwendete Transformer wird in das DhSegment Modell [37] eingebaut. Dafür wird der Block 4 nach dem max-pooling entfernt und stattdessen der Transformer eingebaut. Die Patches für den Transformer sind 8x8 Teilbilder mit 1024 Kanälen. Diese werden mit einer Konvolution mit Filter-Größe 8x8 zu jeweils einem Token der Länge 768 konvertiert. Bei der Ausgabe wird jeder Token mit einer transponierten Konvolution(Unterabschnitt 2.3.3) wieder zu einem 8x8 Teilbild hochskaliert. Diese werden wieder auf die ursprüngliche Größe, aber mit 512 Kanälen zusammengesetzt und an den ersten Decoder Block übergeben. Die Implementierung von Vaswani et al. verwendet gelernte Positional Embeddings mit einer Matrix fester Größe. In dieser Arbeit wird stattdessen ein Sinusoidal Embedding [1] verwendet, um nicht auf eine festgelegte Anzahl an Patches beschränkt zu sein. [6]

Die Trainingsdaten werden in Teilbilder der Größe 1024 Pixel aufgeteilt. Das ist deutlich größer, als die 512 Pixel großen Teilbilder beim Training von DhSegment, weil der Transformer Informationen aus einer größeren Umgebung als Kontext verwenden soll, um die Vorhersagen im Gegensatz zum reinen Faltungsnetz zu verbessern. Dadurch sind die Batchgrößen deutlich kleiner. Sollten die Bilder aus den Trainingsdaten zu klein sein, werden sie mittels Padding auf die Größe 1024 angehoben. [6]

Kapitel 4

Experimente

4.1 Allgemein

In diesem Kapitel werden die durchgeführten Experimente und deren Ergebnisse vorgestellt. Ziel dieser Arbeit ist es, die Segmentierung von Zeitungsseiten mithilfe maschinellen Lernens zu verbessern. Auf den Ergebnissen der vorangegangenen Projektgruppe [33] aufbauend, wird das bisher verwendete DhSegment Modell abgewandelt. Die erste Version verwendet CBAM (Unterabschnitt 3.3.2) und die zweite Version baut nach dem Vorbild des Trans Unets einen Transformer in den Encoder ein (Unterabschnitt 3.3.3). Diese 2 Modelle werden im Laufe der Experimente mit dem ursprünglichen DhSegment Modell verglichen. In Abschnitt 4.4 wird das initiale Training vorgestellt. Dieses wurde durchgeführt, um die Implementierung der beiden neuen Modellen zu testen und lieferte vorläufige Ergebnisse. Abschnitt 4.6 behandelt die durchgeführte Hyperparametersuche. Neben einem generellen Verständnis über das Layout, welches in der Projektgruppe und dem initialen Training bereits erreicht werden konnte, wäre die zuverlässige Erkennung von Überschriften ein großer Gewinn. Durch die Überschriften können einzelne Artikel identifiziert werden und die transkribierten Seiten zur weiteren Verarbeitung in Artikel aufgeteilt werden. Die Hyperparametersuche (Abschnitt 4.6) sollte dazu dienen, insbesondere diesen Teil der Ergebnisse zu verbessern. Wie in Abschnitt 4.4 beschrieben, produziert das initiale Training ein Modell, welches zu viele Pixel als Überschrift erkennt und eines, welches zu wenige erkennt.

Für die Experimente steht der Bender GPU Cluster des HCP der Uni Bonn, sowie der JUWELS (Jülich Wizard for European Leadership Science) Supercomputer zur Verfügung [5, 22]. Bei Bender handelt es sich um einen GPU-Cluster mit 4 A40 GPU Nodes, die jeweils über 2 CPUs mit 8 Kernen und 4 Nvidia GA102GL Grafikkarten, mit 48 GB internem Speicher, verfügen. Außerdem stehen 2 A100 GPU Nodes zur Verfügung, die über 2 CPUs mit 64 Kernen sowie 8 Nvidia GA100 Grafikkarten, mit 80 GB internem Speicher, verfügen. Bender wird für Lehrzwecke, aber auch Forschungsprojekte verwendet. Die initialen

Trainings (Abschnitt 4.4) zum Testen der Implementation sowie der größte Teil der Entwicklung wurde auf Bender durchgeführt. Außerdem wurden alle Trainings auf dem Europeana Datensatz (Abschnitt 4.5) auf Bender durchgeführt. Nodes können auf Bender höchsten 24 Stunden am Stück belegt werden, die Verteilung der Rechenzeit findet automatisch anhand aller vorhandenen eingereichten Jobs statt. Dafür wird der Slurm Workload Manager [45] verwendet. Dieser erlaubt es, Jobs einzureichen, denen automatisch eine Node (oder mehrere) zugewiesen werden, sobald Ressourcen frei sind. Das Beispiel einer Slurm Datei ist im Anhang Abschnitt A.10 zu finden. [5]

Bei JUWELS handelt es sich um einen Teil des Jülich Supercomputing Centre (JSC) und gehört zu den größten Supercomputern Europas. JUWELS besteht aus 2 Modulen, einmal dem Cluster, der CPU Nodes bereitstellt und dem hier verwendeten BOOSTER, welcher GPU Nodes bietet. Jede BOOSTER Node verfügt über 2 CPUs mit 24 Kernen, 512 GB Ram sowie 4 NVIDIA A100 Tensor Core GPUs mit 40 GB. Auch Juwels verwendet den SLURM Workload Manager, um Jobs auf die 936 GPU Nodes aufzuteilen, allerdings gibt es hier vorgeschriebene Rechenzeit Limits, die pro Monat zur Verfügung stehen. Die Reservierung einer Node belegt immer alle 48 Kerne und verbraucht damit pro Stunde 48 Kernstunden. [22]

4.2 Daten

In dieser Arbeit werden zwei Datensätze verwendet. Clausner et al. [9] veröffentlichten im Rahmen des Europeana Newspapers Project einen Datensatz, der aus 528 Seiten historischer Zeitungen aus dem 17. bis 20.Jahrhundert besteht. Dieser wird hier im Sinne des Transferlernens für das Pretraining verwendet. Auf dem Europeana Newspaper Datensatz vortrainierte Modelle werden dann auf dem Datensatz der Kölnischen Zeitung weiter trainiert. [9]

In Zusammenarbeit mit dem Geschichtswissenschaftlichen Institut wurden am Anfang der Projektgruppe Newspaper Segmentation, Richtlinien für die Annotation der Seiten der Kölnischen Zeitung aus dem Jahr 1866 erstellt. Daraus hervorgegangen sind 9 Regionen, die im Programm Transkribus annotiert werden sollten. Die Verteilung der Klassen im Datensatz sowie der Leitfaden zum Erstellen der Annotationen findet sich im Anhang Abschnitt A.11. Hilfskräfte im Geschichtswissenschaftlichen Institut haben 400 Seiten annotiert. Diese werden für das Training in dieser Arbeit verwendet. Genauere Informationen zu den Annotationen finden sich im Projektbericht, Abschnitt 4.1 [33]. In Unterabschnitt 3.1.2 sind Informationen zum Format der Annotationen und deren Verarbeitung zu finden.

4.3 Laufzeit

4.3.1 Optimierung durch Torch Implementationen

Welche und wie viele Experimente durchgeführt werden können, hängt maßgeblich von der Laufzeit des Trainings ab. Die Rechenleistung, mit der Experimente durchgeführt werden, ist begrenzt. Je rechenaufwendiger eine Epoche ist, desto mehr Zeit ist auch nötig, um Experimente durchzuführen, die beispielsweise das Konvergenz-Verhalten in Abhängigkeit von Hyperparametern untersuchen sollen. Deswegen wurde die Laufzeit von verschiedenen Teilen des Trainings vor den eigentlichen Experimenten untersucht. Dafür wurde das DhSegment Modell mit den bisherigen Parametern auf 100 der 400 Bilder im Datensatz trainiert. Initialisiert wurde das Modell mit den Gewichten, die von dem Pretraining auf den Europeana Daten stammen. Das Training erfolgte auf 4 Grafikkarten mit jeweils einer Batchgröße von 64 3x512x512 Pixel Batches. Zusammen ist die Batchgröße damit effektiv 256.

Die erste Untersuchung zeigte 2 Stellen, die eine hohe Laufzeit aufweisen.

- Das Laden des Batches sowie Berechnung und Anwendung der Gradienten beim allerersten Batch. Aufgrund der Initialisierung des Dataloaders dauert der erste Batch-Loading-Schritt deutlich länger als nachfolgende. (Über 20 Sekunden beim ersten Schritt vs. 5-10 Sekunden in allen weiteren.)

- Die Laufzeit der Validations-Runde (Unterabschnitt 3.1.3) mit ca. 30 Sekunden pro Batch spielte eine große Rolle.

Weil die Laufzeit des ersten Batches keine großen Auswirkungen auf den gesamten Trainingsprozess hat, wurde zunächst die Validation genauer untersucht. Die Berechnung des Jaccard Score, sowie die für die Metriken nötige Berechnung des Argmax der Featuredimension der Modellvorhersage machen den Hauptteil der Laufzeit aus. Jaccard Score mit 14 Sekunden pro Batch und Argmax mit 8 Sekunden. Die Metriken wurden, anders als in (Unterabschnitt 3.1.3) beschrieben, vor den Experimenten mit scikit [11] berechnet. Der Multi Class CSI wurde vorher statt mit torch mit numpy implementiert. Die Laufzeit der Metrik-Berechnung konnte durch die Verwendung von torchmetrics Implementationen [32] verbessert werden. Die Berechnung des Jaccard Score konnte dadurch von 14 auf eine Sekunde reduziert werden. Die Laufzeit der Genauigkeit und des Multi Class CSI wurden durch die torch Implementation von 2 und 3 Sekunden auf jeweils eine Sekunde pro Batch reduziert. Damit sinkt die Laufzeit von über 30 Sekunden pro Batch auf 12 Sekunden. Zusätzlich konnte die Laufzeit des Optimierungsschritts durch die Verwendung von Automated Mixed Precision. (AMP) mit torch.autocast, von durchschnittlich 2 Sekunden auf 0,4 Sekunden pro Batch verbessert werden.

4.3.2 Optimierung durch Multithreading

Auch nach der initialen Optimierung der Validation machen die 12 Sekunden pro Batch immer noch einen großen Teil der Trainingszeit aus. Nach einer vorläufigen Optimierung des Batch-Loadings (Unterabschnitt 4.3.3), betrug die Laufzeit für eine Epoche ca. 6 Minuten. Davon entfallen bei 3 Validationsrunden mit jeweils 4 Batches ($3 \cdot 4 \cdot 12s = 144s$) über 2 Minuten alleine auf das Berechnen der Metriken in der Validation. Das setzt sich zusammen aus ca. 9 Sekunden für die Berechnung des Argmax auf der CPU und jeweils eine Sekunde für die 3 Metriken.

Die Erklärung für diese hohe Laufzeit liegt in der Menge der Daten, die dort verarbeitet werden müssen. Die Vorhersage des Modells läuft auf 4 Grafikkarten gleichzeitig, die jeweils einen 64-er Batch verarbeiten. Bei 10 Ausgabeklassen resultiert aus der Vorhersage ein Tensor mit Dimensionen [256, 10, 512, 512] des Typ float16. Das entspricht 1,28 GB, die von Argmax verarbeitet werden müssen. Der Target Tensor mit [256, 512, 512] uint8 Werten und 64 MB ist dagegen vernachlässigbar.

Um diese Berechnung zu beschleunigen, ist es naheliegend, die Berechnung auf mehrere CPU Kerne auszulagern. Dies ist innerhalb von reinem Python Code effektiv nur mit Multiprocessing möglich, weil der Global Interpreter Lock (GIL)[3] verhindert, dass mehr als ein Thread gleichzeitig den Python Interpreter kontrolliert. Das bedeutet, Threads können zwar auf unterschiedlichen Kernen laufen, jedoch darf nur ein Thread zu einem bestimmten Zeitpunkt aktiv sein. Effektiv verhindert dies, Berechnungen parallel auf mehreren Kernen durchzuführen. Der Nachteil von Prozessen gegenüber Threads ist, dass es deutlich aufwendiger ist, einen neuen Prozess zu erstellen, als einen Thread. Das liegt daran, dass Threads den gleichen Speicherbereich verwenden, während für einen Prozess in Python ein zusätzlicher Python Interpreter gestartet wird, der einen eigenen Speicherbereich hat. Dadurch wird das Transferieren von Daten zwischen den Speicherbereichen zu einem wichtigen Faktor für die Performance. Die multiprocessing guidelines warnen davor, zu große Datenmengen einem Prozess direkt zu übergeben (multiprocessing guidelines). Tatsächlich führt das Anwenden von Multiprocessing auf die Berechnung der Metriken durch die 1,28 GB aus der Vorhersage dazu, dass die Prozesse nie abgeschlossen werden und das Programm in der Validation ohne Fehlermeldung aufhört zu arbeiten. Das von den Guidlines vorgeschlagene Verwenden einer Queue hilft nur bei kleineren Datenmengen. Die Berechnung der Metriken konnte mithilfe von Python Multiprocessing und Queues unter Aufteilung der Daten in Pakete von 40 KB beschleunigt werden. Wenn in die Queue deutlich mehr als diese 40 KB eingefügt werden, stoppt das Programm wie bisher und die Prozesse werden nie fertig. Nichtsdestotrotz verbessert dies die Laufzeit von 12 auf 7 Sekunden pro Batch. Der Flaschenhals in diesem Fall ist, dass die Pakete von 40 KB zu klein sind, um die komplette Rechenkapazität der vorhandenen Prozessor-Kerne auszunutzen. [3, 52]

Das GIL kann durch Implementierungen in C umgangen werden. Nur Teile des Programms, die tatsächlich in Python durchgeführt werden, sind immer

vom GIL betroffen. Libraries wie numpy und torch setzen das GIL aus, sobald sie C Code ausführen. Deswegen können in Python gestartete Threads zum Beispiel von torch.argmax auf mehrere Kerne ausgelagert werden. Die Berechnung der Metriken wurde nach dieser Erkenntnis mithilfe von multiprocessing.pool.ThreadPool und threadPool.map in 64 Teile auf 64 Threads aufgeteilt. Das führt zu einer Laufzeit von ca. 3 Sekunden pro Batch. Noch weiter verbessert werden, konnte das ganze, indem die Berechnung auf der Grafikkarte durchgeführt werden. Dadurch wird bei einer Aufteilung in 32 Teile auf 32 Threads eine Laufzeit von 0,3 Sekunden erreicht. Um die Laufzeit zu messen, wurde ein Training über 2 Epochen mit 400 Bildern aus dem Datensatz der Kölnischen Zeitung, mit 10 torch.Dataloader Workern und einem Prefetch Faktor von 2, sowie 32 Threads für die Berechnung der Metriken auf der GPU durchgeführt. Mit dieser Konfiguration beträgt die Laufzeit für eine Epoche ca. 2 Minuten. [52]

Um den Gewinn durch Multithreading zu verdeutlichen, wurde aufbauend auf den Ergebnissen aus Unterabschnitt 4.3.3, mit 7 Workern und einem Prefetch Faktor von 6, Trainingseinheiten von 5 Epochen in Abhängigkeit von der Anzahl der Threads und der Aufteilung der Daten durchgeführt. Es stellt sich jedoch heraus, dass auch die Verwendung von einem einzigen Thread kaum zu einem Nachteil führt. Im Gegensatz zum ursprünglichen Vorgehen ist hier einfach wichtig, dass die Metriken auf der Grafikkarte berechnet werden. Lediglich bei der Berechnung auf CPU Kernen spielte der Einsatz des Multithreading eine größere Rolle.

4.3.3 Optimierung des Batch-Loadings

Das Laden der Daten für jeden Batch stellt einen zeitintensiven Teil des Trainings dar. Wenn die Berechnung eines Batches abgeschlossen ist, kann die Vorhersage des Modells für den nächsten Batch erst dann erfolgen, wenn die Daten vom torch DataLoader zur Verfügung gestellt wurden. Wenn dies zu lange dauert, führt das zu einer unnötigen Wartezeit. Um das Laden zu beschleunigen, kann der DataLoader mehrere Prozesse parallel verwenden, um Daten zu laden. Außerdem kann jeder dieser Prozesse eine Anzahl an Batches im Voraus laden, damit diese bei Bedarf schneller zur Verfügung stehen.

Um die optimale Anzahl an Workern und Prefetch-Batches zu ermitteln, wurde die Trainingszeit von 3 Epochen auf dem gesamten Datensatz der Kölnischen Zeitung gemessen. Dies geschieht in Abhängigkeit von der Anzahl an Workern und Prefetch-Batches und nach den Optimierungen in Unterabschnitt 4.3.1 und Unterabschnitt 4.3.2. Es ist zu erwarten, dass der Prefetch Faktor nicht besonders hoch liegen sollte. Da das Training kontinuierlich Daten benötigt und es keine Stellen gibt, an denen plötzlich sehr viele Daten angefordert werden, würde das Laden von 20 Batches im Vorhinein einfach nur Speicherplatz aufbrauchen. Abhängig von der Anzahl der Worker wäre es viel mehr zu erwarten, dass niemals 20 Batches geladen werden, weil neue Batches schneller vom Training angefordert werden als sie geladen werden. Deswegen liegt der Parameter Prefetch-Batches im Bereich von 1 bis 10 und der Parameter Workern wird im

Bereich von 1 bis 40 untersucht.

In Abhängigkeit von der Anzahl an Workern und dem Prefetch Faktor, wird die Geschwindigkeit des Trainings in Batches pro Sekunde gemessen. Abbildung 4.2 zeigt den Durchschnitt bezüglich der beiden Dimensionen (3-D Grafik im Anhang Abschnitt A.1).Der Prefetch Faktor hat keine großen Auswirkungen. Bezüglich der Worker liegt das Maximum bei 7 Workern. Das allgemeine Maximum liegt bei 7 Workern und einem Prefetch Faktor von 6. Bei einer geringeren oder höheren Anzahl an Workern nimmt die Geschwindigkeit stark ab. Im Bereich von 30 - 40 Workern steigt die Geschwindigkeit wieder. Dort wurden keine weiteren Untersuchungen durchgeführt, weil der mögliche Zeitgewinn gering ist. Bei dem Maximum von 7 Workern und einem Prefetch Faktor von 6, wird die meiste Zeit auf die Vorhersage(ca. 31 Sekunden pro Epoche), die Backpropagation (ca. 22 Sekunden pro Epoche) und die Validation (ca. 18 Sekunden pro Epoche) verwendet. Das Laden der Daten dagegen benötigt ca. 12 Sekunden pro Epoche. Die durchgeführten 5 Epochen benötigen dabei 520 Sekunden, dabei ist aber zu beachten, dass der Start der 1. Epoche, wie in Unterabschnitt 4.3.1 erwähnt, besonders lange dauert. Es werden hier also bereits mit geringem Ressourcen-Verbrauch gute Werte erreicht, damit ist es nicht nötig oder sinnvoll, die Möglichkeit von kleinen Verbesserungen mit deutlich mehr Ressourcen-Aufwand zu untersuchen.

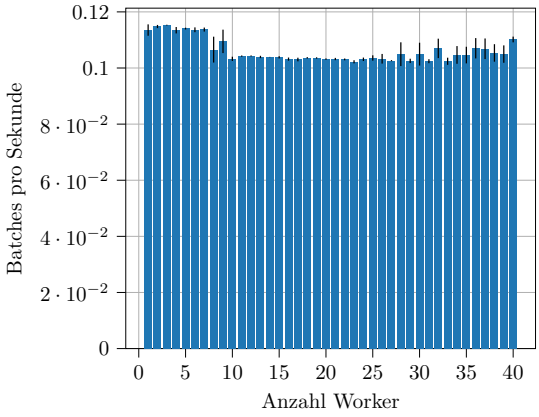


Abbildung 4.1: Trainingsgeschwindigkeit gemessen über 3 Epochen in Abhängigkeit von Anzahl Workern. Höhere Werte bedeuten geringere Laufzeit. Die Fehlerbalken stellen die Standardabweichung über 3 Durchläufe dar.

Für das CBAM Modell wurde ebenfalls eine Untersuchung zu der Anzahl der Worker durchgeführt, um die Laufzeit zu optimieren. In diesem Fall gibt es keinen großen Unterschied im Bereich von 1-40 Workern. Das Maximum liegt bei 3 Workern und dieser Wert wird für das finale Training verwendet (Unterabschnitt 4.6.4). Die Ergebnisse sind in Abbildung 4.1 zu finden. Es gibt in diesem Fall kein klares Muster, wie die Laufzeit mit der Anzahl an Workern zusammenhängt. Es kann lediglich festgestellt werden, dass 1-7 Worker deutlich besser sind als 8 und mehr.

4.3.4 Vorhersage

Die Erkenntnisse aus der Performancesteigerung des Trainings wurden auch auf die Vorhersage angewendet. Ursprünglich konnten 100 Bilder des Datensatzes,

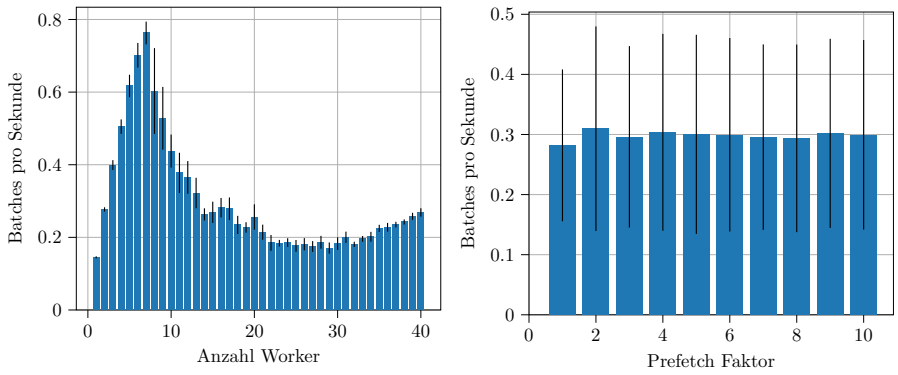


Abbildung 4.2: Trainingsgeschwindigkeit gemessen über 5 Epochen in Abhängigkeit von Anzahl Workern (Links) und dem Prefetch Faktor (Rechts) im torch Dataloader. Höhere Werte bedeuten geringere Laufzeit. Die Fehlerbalken stellen die Standardabweichung bezüglich der anderen Dimension dar. Die Daten sind entlang der jeweiligen Dimension zusammengefasst, ausgehend von den Daten in Abbildung A.1.

die um die Hälfte herunterskaliert werden, innerhalb einer Stunde verarbeitet werden. Um das zu beschleunigen, wurde der Ladevorgang auf einen Dataloader umgestellt und die Vorhersage des Modells auf 4 Grafikkarten parallelisiert. Die Verwendung des Dataloaders hat den Vorteil, dass nun mehrere Worker die Daten parallel laden können und nicht ein Bild nach dem anderen geladen wird. Durch die Nutzung von 4 Grafikkarten wurde die Vorhersage und die Berechnung des Argmax beschleunigt. Jedoch stellte sich in diesem Fall heraus, dass das größtenteils in Python durchgeführte Post-Processing den überwiegenden Teil der Zeit benötigt. Der Zeit-Gewinn durch 4 Grafikkarten hatte damit keinen großen Einfluss auf die Laufzeit.

Die Laufzeit konnte maßgeblich verbessert werden, indem die Berechnung von Submasken von Python auf numpy umgestellt wurde. Die ursprüngliche Version verwendete Python Schleifen, um für jedes Pixel des Bildes zu überprüfen, welcher Klasse er angehört. Für jede Klasse wird so eine Submaske erstellt, deren Pixel entweder den Wert 0 oder den Wert 1 haben. Pixel, die dieser Klasse angehören, haben den Wert 1 alle anderen 0. Diese Operation ist in Python sehr langsam und kann ohne großen Aufwand mit numpy durchgeführt werden. Diese Änderung senkt die Laufzeit für die Berechnung der Submasken von über 10 Sekunden auf 0,1 Sekunde pro Bild.

Trotz dieser Änderung werden zu viele Berechnungen durchgeführt, die durch das GIL eingeschränkt werden, um einen großen Nutzen mit Multithreading zu erzielen. Die Umstellung von einem Thread auf bis zu 40 Threads, senkt die Laufzeit des Slicing Export (Unterabschnitt 3.2.3) für 210 Bilder von 16 auf 13 Minuten. Der Engpass in diesem Fall ist die Berechnung der Polygone aus

den Masken, die durch sklearn durchgeführt wird und nicht von Multithreading profitiert. Wenn nur bestimmte Klassen für die Ausgabe relevant sind, kann die Laufzeit dadurch verbessert werden, dass die Berechnung der Polygone für alle anderen Klassen nicht durchgeführt wird. Für das Ausschneiden von Textregionen und die Bestimmung der Lesereihenfolge sind zum Beispiel nur 2 Klassen relevant(Artikel und große Separatoren). Durch das Abschalten der Berechnungen für die anderen Klassen konnte die Laufzeit für 210 Bilder auf 5min gesenkt werden.

4.4 Initiales Training

Um die Implementierungen der neuen Modelle zu testen, wurden vor der systematischen Suche nach Hyperparametern und der Verbesserung der Performance, initiale Trainings durchgeführt. Jeweils ein längeres Training wurde von jedem Modell durchgeführt. In Abbildung 4.3 werden die Ergebnisse dieser 3 Durchläufe dargestellt. Das Training vom reinen DhSegment beläuft sich auf 150 Epochen, während die beiden anderen 300 Epochen lang trainiert wurden. Bei allen Versionen wurde der Encoder im ersten Teil des Trainings eingefroren. Am Ende des Trainings wird das dann aufgehoben, so dass auch für die Parameter des Encoders Gradienten berechnet werden und diese optimiert werden. Im Fall von DhSegment wurde der Encoder bei 90 Epochen freigegeben und bei CBAM und Trans-Unet passiert das bei 150 Epochen. Im Anhang Abschnitt A.2 ist der Verlauf der Trainings dargestellt.

Die in Abbildung 4.3 dargestellte Auswertung auf dem Test-Datensatz ergibt, dass das Trans Unet insgesamt etwas schlechtere Ergebnisse erzielt. Der Jaccard Score ist eindeutig geringer als bei den anderen beiden Modellen und bei den Klassen Heading (Überschrift) und Separator Short ist die Precision deutlich geringer als bei den anderen beiden Modellen. Das schlechte Ergebnis bei den Überschriften ist besonders relevant, weil diese zur Identifizierung von Artikeln gut erkannt werden sollen. Die UnknownRegion Klasse enthält Regionen, die keiner anderen Klasse zugeordnet werden sollten sowie fehlerhafte Daten wie Regionen, die zwar markiert sind, aber keiner Klasse zugeordnet wurden. Deswegen ist es zu erwarten, dass diese nicht erkannt werden kann. Im Idealfall sollte kein Pixel in den Daten als UnknownRegion markiert sein.

Die Vorhersage der DhSegment und CBAM Modelle auf einer Zeitungsseite ist in Abbildung 4.4 dargestellt. Wie aus den Ergebnissen in Abbildung 4.3 hervorgeht, sind die Ergebnisse ähnlich. Es gibt jedoch einen entscheidenden Unterschied in der Vorhersage der für die Identifizierung der Artikel nötigen Überschriften. Das DhSegment Modell erkennt nur wenige Überschriften, oder nur Teile einer Überschrift. CBAM dagegen erkennt zu viele Stellen als Überschrift. Dieser Unterschied findet sich in Abbildung 4.3 wieder. DhSegment hat bei Überschriften (Heading) eine etwas höhere Precision als CBAM. Eine höhere Precision bedeutet, dass Pixel, die als Überschrift erkannt werden, wahrscheinlicher tatsächlich eine Überschrift sind. Das deckt sich mit der Beobachtung aus Abbildung 4.4, weil CBAM einige Stellen als Überschrift markiert, die kei-

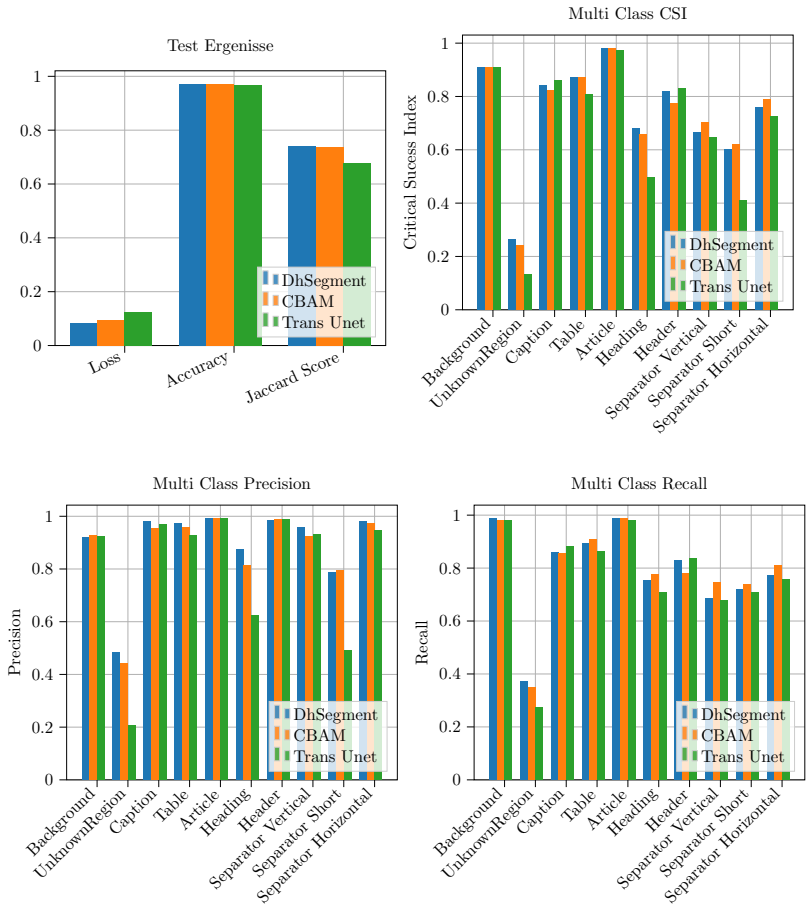


Abbildung 4.3: Testergebnisse für das initiale Training der 3 Modelle. Es sind keine Fehler-Balken vorhanden, weil jedes Training einmal durchgeführt wurde.

ne sind. Weiterhin hat CBAM einen etwas höheren Recall bei Überschriften. Das bedeutet, CBAM erkennt mehr der Pixel, die eine Überschrift sein sollen als DhSegment. Auch das deckt sich mit Abbildung 4.4, weil DhSegment einige Überschriften als Text erkennt. Zusammengefasst erkennt DhSegment zu wenige und CBAM zu viele Pixel als Überschriften. Kleine Änderungen in den Precision und Recall Werten können wie an diesem Beispiel demonstriert große Auswirkungen haben. Eine Besonderheit von DhSegment ist es, Bereiche zwischen 2 verschiedenen Klassen teilweise als UnknownRegion zu markieren. Diese Klasse enthält Fehler in den Daten und sollte in der Vorhersage gar nicht vorkommen.

Im Training sollen alle 10 Klassen erkannt werden, die im Datensatz annotiert sind. Je nach Anwendungsfall ist die Unterscheidung dieser 10 Klassen aber nicht nötig. Soll die Seite nur transkribiert werden, ohne Unterscheidung

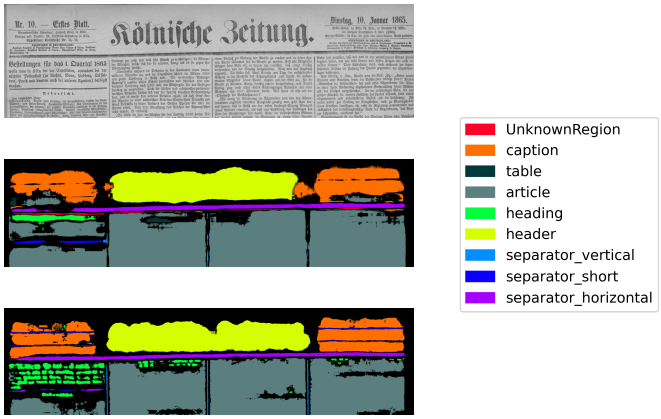


Abbildung 4.4: Vorhersage des Modells über einen Ausschnitt Zeitungsseite. Oben ist das Original zu sehen, in der Mitte das Ergebnis von DhSegment und unten das CBAM Ergebnis. Pixel werden nur einer Klasse zugeordnet, wenn das Modell eine Wahrscheinlichkeit von über 75 % angibt. Die vollständigen Seiten sind im Anhang Abschnitt A.3 zu finden.

der Text-Regionen, können alle Text-Regionen zu einer Klasse zusammengefasst werden. Das ist eine Aufgabe, die deutlich leichter zu lernen ist, weil nur Text vom Hintergrund unterschieden werden muss. Wenn die Lesereihenfolge berechnet werden soll, ist es zusätzlich noch wichtig, große Separatoren zu erkennen, die eine Seite trennen. Deswegen wurde aufbauend auf dem Trainierten DhSegment Modell, ein Training durchgeführt, welches nur die 3 Klassen Article(Alle Textregionen), Separator Vertical und Separator Horizontal (große Separatoren) unterscheidet. Dies verbessert die Erkennung der übrigbleibenden Klassen. In Abbildung 4.5 ist zu sehen, dass sich die Precision der kombinierten Separator-Klasse im Vergleich zur Separator Short Klasse in Abbildung 4.4 deutlich verbessert hat. Das liegt daran, dass die Modelle gerade die beiden kleinen Separator Typen öfters miteinander verwechseln. Dieses Modell ist so gut, dass es bereits für die Aufbereitung von Zeitungsseiten verwendet wird. Der Slicing Export Unterabschnitt 3.2.3 exportiert Textregionen einer vorgegebenen Größe als ausgeschnittene Bilder und ordnet sie nach der berechneten Lesereihenfolge. Diese Ausschnitte werden dann einem OCR-Modell zur Transkription übergeben. In naher Zukunft sollen so 130k Bilder der Kölnischen Zeitung transkribiert und für die Sentiment-Analysis aufbereitet werden. 2 Beispiele dafür sind im Anhang Abschnitt A.8

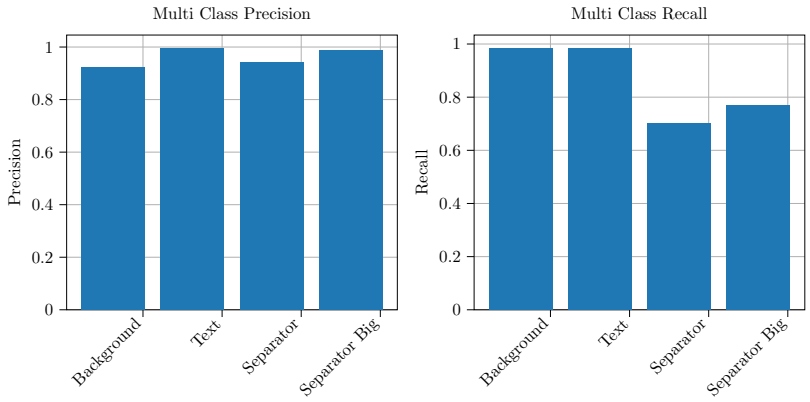


Abbildung 4.5: Precision und Recall Werte für das Training mit reduzierten Klassen. Die Textklasse enthält alle Textregionen und die Separator-Klasse enthält alle kleinen Separatoren, die von den großen unterschieden werden. Weitere Ergebnisse im Anhang Abschnitt A.4

4.5 Pretraining

Wie in der vorangegangenen Projektgruppe stehen 2 Datensätze zu Verfügung. 140 Seiten des Datensatzes der Kölnischen Zeitung wurden während der Projektgruppe annotiert. Während dieser Arbeit ist der Datensatz auf 400 Seiten angewachsen. Außerdem wird der Europeana Datensatz verwendet (Siehe auch Abschnitt 4.2). Die resultierenden Segmentierungs-Modelle sollen Seiten der Kölnischen Zeitung verarbeiten können, deswegen ist der Datensatz der Kölnischen Zeitung der primäre Datensatz. Aber auch die Europeana Daten werden im Sinne des Transferlernens eingesetzt. Modelle des maschinellen Lernens profitieren davon, auf möglichst vielen Daten trainiert worden zu sein. Das geschieht in diesem Projekt das erste Mal beim Initialisieren des ResNet Encoder. Der Encoder von allen 3 Modellen besteht aus ResNet 50 Blöcken. Diese Gewichte werden mit den Werten initialisiert, die aus einem Training auf dem ImageNet Datensatz [10] herrühren. Der zweite Punkt, an dem Transferlernen verwendet wird, ist das Pretraining auf dem Europeana Datensatz. Die Modelle, die in der Hyperparametersuche (Abschnitt 4.6) verwendet werden, sind auf dem Europeana Datensatz vortrainiert und werden mit den entsprechenden Gewichten initialisiert. In der Projektgruppe Segmentierung von Zeitungsseiten wurde festgestellt, dass das Verwenden von Transferlernen mit vortrainierten Modellen auf dem Europeana Datensatz bessere Ergebnisse erzielt, als ein Modell zu verwenden, welches nur mit auf ImageNet vortrainierten ResNet Gewichten initialisiert wird (Siehe Abschnitt 4.2 [33]).

Für jedes Modell wurden 3 Versionen des Pretrainings mit unterschiedlichen Pytorch Seeds durchgeführt. Abbildung 4.6 zeigt die Ergebnisse des Vergleichs der 3 Modelle. DhSegment CBAM liefert leicht bessere Ergebnisse als die ande-

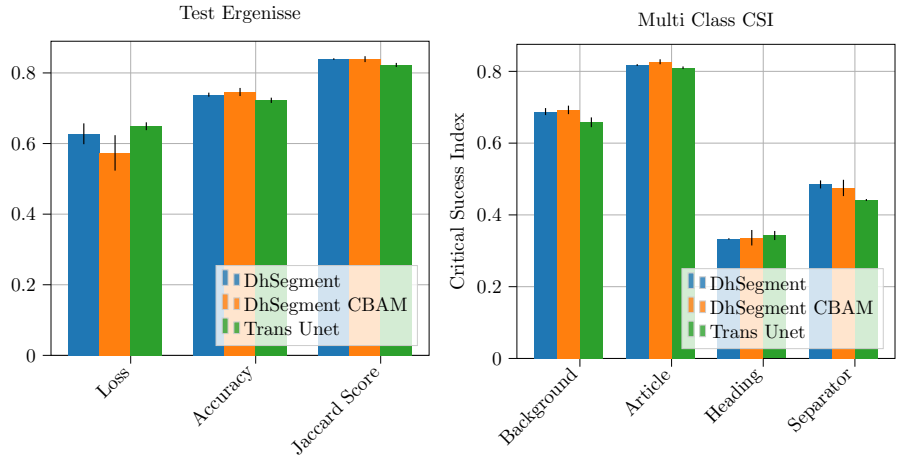


Abbildung 4.6: Testergebnisse für das Training auf den Europeana Daten über 200 Epochen. Die Annotationen dieses Datensatzes unterscheidet andere Klassen, welche in den 4 dargestellten Klassen zusammengefasst werden.

ren beiden Modelle, das ist aber nicht weiter relevant, weil diese Modelle nur die Grundlage für das weitere Training sind. Die Ergebnisse sind insgesamt deutlich schlechter, als die Ergebnisse des initialen Trainings (Abschnitt 4.4). Das ist aber auch zu erwarten, weil das initiale Training auf vortrainierten Modellen aufbaut und der Datensatz der Kölnischen Zeitung eine größere Menge Daten enthält.

4.6 Hyperparameter Suche

4.6.1 Loss Funktion

In allen bisherigen Trainings wurde die Verlustfunktion Kreuz Entropie verwendet. Diese ist nicht gut dafür geeignet, mit unausgeglichenen Datensätzen zu arbeiten. Unausgeglichene Datensätze sind solche, in denen die verschiedenen Klassen stark unterschiedlich oft vertreten sind. Lin et al. stellen den Focal Loss [26] vor, der besser mit unausgeglichenen Datensätzen umgehen kann. Weil das initiale Training bestimmte seltene Klassen wie die Überschriften und einen bestimmten Separator Typ nicht gut erkennen kann, sollte als erster Hyperparameter untersucht werden, ob die Erkennung von seltenen Klassen durch Verwendung des Focal Loss verbessert werden kann. [26]

Es wurde eine Laufzeituntersuchung durchgeführt, bei der jeweils 3 Trainings von 5 Epochen für die beiden Loss Funktionen einmal mit eingeschaltetem Automated Mixed Precision. (AMP) Training und einmal ohne AMP. Die Kreuz Entropie profitiert im Gegensatz zum Focal Loss stark von AMP (Abbildung 4.8). Die Laufzeit des Trainings ist bei Verwendung der Kreuz Entropie

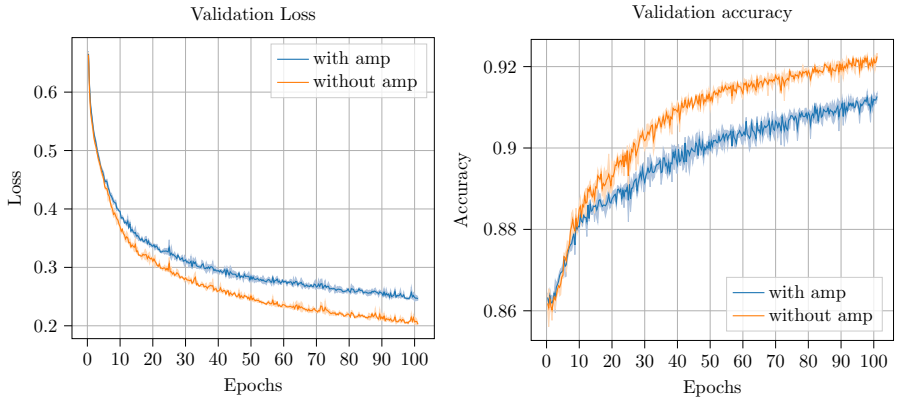


Abbildung 4.7: Validationsverlauf für zwei Versionen eines Trainings von 100 Epochen mit Kreuzentropie. Der einzige Unterschied zwischen den Versionen besteht in der Aktivierung des AMP Trainings. Für beide Versionen wurden 3 Trainingsdurchläufe durchgeführt und die Ergebnisse gemittelt. Der schattierte Bereich stellt die Standardabweichung dar.

und AMP am geringsten. Wenn der Focal Loss ein besseres Konvergenz Verhalten ermöglicht, kann es sich aber lohnen, die höhere Laufzeit in Kauf zu nehmen. Zu beachten ist allerdings, dass der Focal Loss ohne AMP eine kleinere Batch-Größe benötigt. Mit der Batchgröße von 64 pro Grafikkarte übersteigt Focal Loss ohne AMP den verfügbaren Grafikkarten Speicher. Deswegen wurde dieses Training mit Batchgröße 32 durchgeführt. Um die beiden vergleichbar zu halten, werden in Abbildung 4.8 für Focal Loss ohne AMP 2 Batches zusammengerechnet. Das entspricht dann wieder einer Berechnung von 64 Crops pro Grafikkarte.

Die Verwendung von AMP Training reduziert die Laufzeit, verändert aber auch das Konvergenz-Verhalten. Für den Focal Loss wurde deswegen ein Training mit und ein Training ohne AMP durchgeführt. Die Laufzeit der Kreuz Entropie profitiert sehr vom AMP, weswegen es für die weiteren Experimente nicht sinnvoll wäre, AMP abzuschalten. In Abbildung 4.7 ist zu sehen, dass das Training ohne AMP bessere Ergebnisse erzielt. Das Training mit AMP benötigt ca. zweieinhalb Stunden, während die Laufzeit ohne AMP über 10 Stunden beträgt. Die leichte Verbesserung der Ergebnisse bezogen auf die 100 Epochen rechtfertigen diesen Laufzeit-Unterschied nicht. Deswegen wurde im nächsten Schritt die Kreuz Entropie mit AMP mit dem Focal Loss verglichen, einmal mit und einmal ohne AMP. Abbildung 4.9 zeigt die Ergebnisse dieses Vergleiches über 400 Epochen. Es ist zu erkennen, dass die Genauigkeit und Jaccard Score Werte der Kreuz Entropie besser sind als bei den Focal Loss Trainings. Auch hier ist zu erkennen, dass das AMP Training negative Auswirkungen auf das Training hat. Im Falle des Focal Loss würde es sich lohnen, ohne AMP zu arbeiten, weil die Beschleunigung des Trainings deutlich geringer ist als bei der Kreuz Entro-

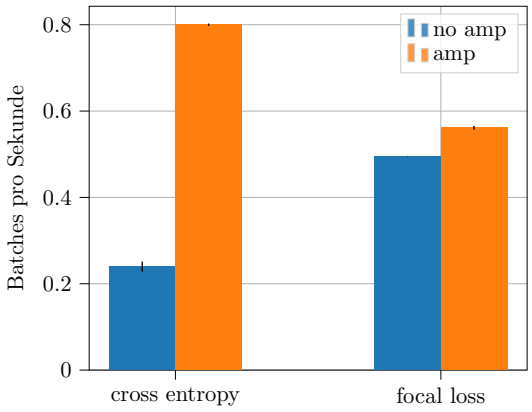


Abbildung 4.8: Trainingsgeschwindigkeit gemessen über 5 Epochen in Abhängigkeit von der Loss Funktion und der Aktivierung des AMP Trainings. Jede Option wurde 3 Mal für ein Training verwendet und gemittelt. Die Fehlerbalken stellen die Standardabweichung bezüglich dieser 3 Trainings dar. Batches pro Sekunde bedeutet die Verarbeitung von 256 Crops, welche ein 79-tel einer Epoche entspricht.

pie. Die Kreuz Entropie verhält sich wie erwartet, während der Focal Loss über das gesamte Training bei ca. 0.98 verweilt (Abbildung 4.9). Der Focal Loss wird in dieser Implementation so berechnet, dass er immer zwischen 0 und 1 liegt, wobei der Fehler minimiert werden soll. Eine Erklärung für dieses Verhalten wäre, dass der Focal Loss zu große Anteile des Gradienten in Klassen investiert, die schwer zu lernen sind, weil sie in den Daten zu selten vorkommen. Allerdings fördert das Training mit reduzierten Klassen wie in Unterabschnitt 4.6.3 beschrieben, auch keine besseren Ergebnisse zu Tage. In beiden Versionen hat das Focal Loss Training ein Exploding Gradient Problem, weswegen Gradient Clipping betrieben wird. Das bedeutet, dass die Gradienten zu groß werden. In der Praxis entstehen dadurch NaN (NotaNumber) Werte im Loss, mit denen nicht weiter gearbeitet werden kann. Durch Gradient Clipping kann das zwar verhindert werden, die Tatsache, dass der Fehlerwert sich während des Trainings aber nicht verbessert, deutet darauf hin, dass es ein ernstes zugrundeliegendes Problem gibt. Zu beachten ist noch, dass wegen eingeschränkten Grafikkarten-Speicher das Training mit Focal Loss mit 32-er Batches durchgeführt werden musste. Im Gegensatz dazu verwendet die Kreuz Entropie 64-er Batches. Aufgrund des recht deutlichen Ergebnisses und der Tatsache, dass das Training pro Epoche bei der Kreuz Entropie deutlich weniger Zeit benötigt, werden die nachfolgenden Experimente mit Kreuz Entropie durchgeführt.

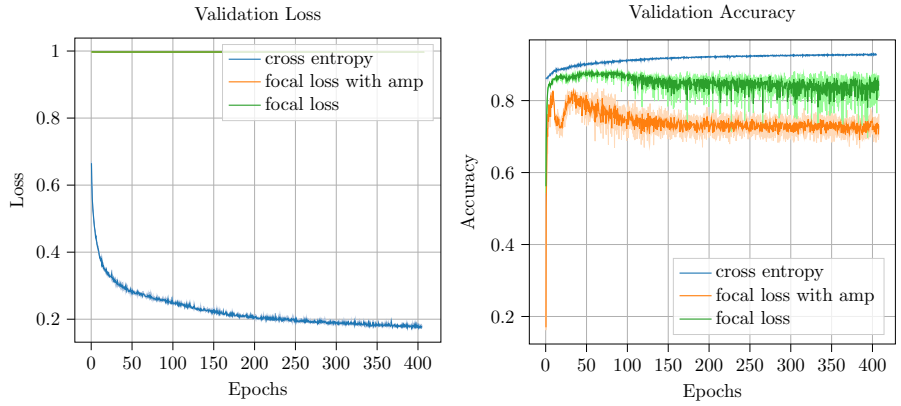


Abbildung 4.9: Validationsverlauf für drei Versionen eines Trainings von 400 Epochen. Verglichen wurden Versionen mit Kreuzentropie(Cross Entropy) und Focal Loss. Für den Focal Loss ist eine Version mit AMP und eine ohne vorhanden. Für aller drei Versionen wurden 3 Trainingsdurchläufe durchgeführt und die Ergebnisse gemittelt. Der schattierte Bereich stellt die Standardabweichung dar.

4.6.2 Lernrate und Weight Decay

Der zweite Teil der Hyperparameter Suche bezieht sich auf die Lernrate und das Weight Decay. Weight Decay ist das Addieren einer euklidischen Norm zur Kostenfunktion. Für beide Parameter wurden separate Experimente durchgeführt, um festzustellen, ob der Parameter überhaupt einen messbaren Einfluss hat und wenn ja bei welchem Wert das beste Ergebnis erzielt wird. Dafür wurden jeweils 3 Trainingsrunden über 200 Epochen mit verschiedenen Werten für den jeweiligen Parameter durchgeführt. Um die Ergebnisse zu bewerten, wird das per Early Stopping bestimmte beste Modell des Trainings auf dem Testdatensatz ausgewertet. Die Genauigkeit und der Jaccard Score werden zu einem Wert zusammengefasst (score). Dafür wird einfach der Mittelwert genommen. Außerdem werden die Critical Sucess Index(CSI) Werte der einzelnen Klassen beachtet. Die Werte für alle 10 Klassen werden gemittelt und im class score zusammengefasst. In Abbildung 4.10 ist zu sehen, dass das Weight Decay keinen Einfluss auf die Ergebnisse hat. Auch das Abschalten des Weight Decays führt zu den gleichen Ergebnissen. Trotzdem wird hier der ursprüngliche Wert für das Weight Decay für die Zukunft beibehalten.

Bei der Lernrate werden die Ergebnisse bei zu hoher Lernrate deutlich schlechter. Auch die Abweichung der Ergebnisse nimmt bei zu hoher Lernrate deutlich zu. Das ist ein Hinweis auf zu große Schritte beim Gradienten-Abstieg, die von der hohen Lernrate herrühren. Das Maximum der Summe beider Scores bei der Lernrate liegt bei $6 \cdot 10^{-4}$. Damit ist dieser Wert für die Lernrate sowie 10^{-5} für das Weight Decay Ergebnis der Hyperparameter Suche.

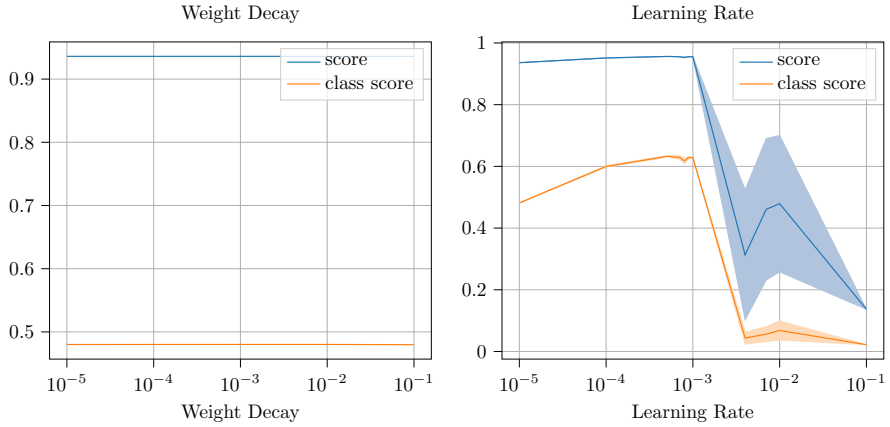


Abbildung 4.10: Ergebnisse für die Hyperparameter Suche für die Lernrate und den Weight Decay. Ergebnisse sind gemittelt über 3 Trainingsdurchläufe pro Wert. Schattierte Bereiche stellen die Standardabweichung dar. Score ist der Mittelwert von Genauigkeit und Jaccard Score und Class Score ist der Mittelwert aller 10 Multi Class CSI Werte. Beide haben Werte zwischen 0 und 1, wobei 1 ein perfektes Ergebnis wäre. Bei beiden Graphen beträgt der jeweils andere Wert 10^{-5}.

4.6.3 Skalierung und Klassen Reduzierung

Wie in Unterabschnitt 3.1.2 beschrieben, werden die Trainingsdaten bezüglich der Skalierung augmentiert. Das ist dazu da, dass das Modell auch Bilder verarbeiten kann, die eine andere Auflösung besitzen, als die Trainingsdaten. In der Projektgruppe Segmentierung von Zeitungsseiten [33] ist es als Problem aufgefallen, dass herunterskalierte Bilder sehr schlechte Ergebnisse bei der Vorhersage produzieren. Der Grund dafür scheint zu sein, dass zum Beispiel unterschiedliche Separator Typen anhand der Pixel-Anzahl identifiziert werden. Wird das Bild nun herunterskaliert, ist der gleiche Separator nur noch halb so breit und wird deswegen anders behandelt. [33]

Da hier aber möglichst gute Ergebnisse erreicht werden sollen, ist es auch eine Option diese Augmentierungen abzuschalten und eine feste Auflösung für die Eingaben eines Modells festzulegen. Eine weitere Option zur Verbesserung der Ergebnisse ist es, die beiden seltensten Klassen im Datensatz mit einer häufigeren zu vereinigen. Einerseits gibt es die UnknownRegion Klasse, die alle Regionen enthält, die nicht zugeordnet werden konnten. Diese wird in einer reduzierten Version mit dem Hintergrund vereint. Außerdem wird die Separator Short Klasse mit der Separator Vertical Klasse vereinigt, weil es sich um den gleichen Separator Typ handelt, nur in 2 unterschiedlichen Orientierungen. In Abbildung 4.11 sind die Precision und Recall Werte für den Vergleich dieser 2 Varianten mit den bisherigen Trainingsergebnissen. Weil die Variante ohne Skalierungs-Augmentierung (no augment) bessere Ergebnisse liefert, als

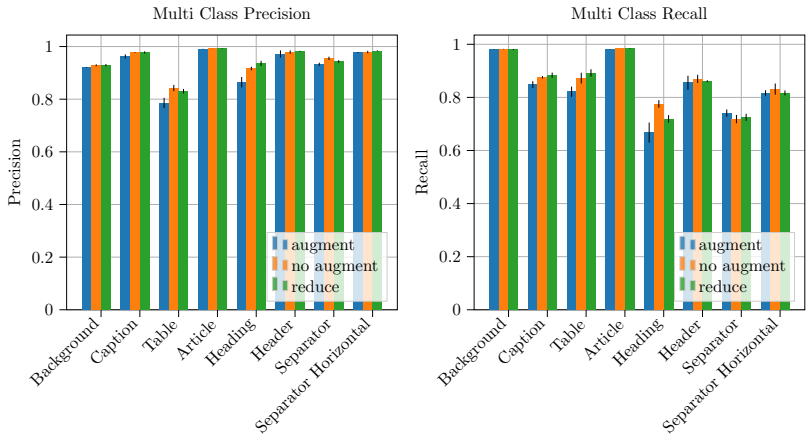


Abbildung 4.11: Precision und Recall Werte für das Training ohne Skalierungs-Augmentierung und mit reduzierten Klassen. Weitere Ergebnisse sind im Anhang Abschnitt A.5

die Default Variante (augment) wurde die Version mit reduzierten Klassen (reduced) ebenfalls ohne Augmentation durchgeführt. Bei der besonders relevanten Überschriften Klasse (Heading) ist der Recall Wert deutlich besser, wenn keine Skalierungs-Augmentation durchgeführt wird. Das Reduzieren der Klassen führt dagegen an dieser Stelle wieder zu einem schlechteren Ergebnis. Der leicht bessere Precision Wert ist dagegen nicht so relevant, weil die Precision bereits sehr gut ist. Weil das Hauptaugenmerk hier auf der Identifizierung der Überschriften liegt, wird für das finale Training die Version ohne Skalierungs-Augmentation, aber mit allen Klassen verwendet.

4.6.4 Finales Training

Die in der Hyperparametersuche (Abschnitt 4.6) ermittelten Parameter werden für das weitere Training von DhSegment und DhSegment CBAM(Abschnitt 3.3) verwendet. Die Ergebnisse des Trans-Unets sind über das initiale Training(Abschnitt 4.4) besonders bei den relevanten Überschriften deutlich schlechter gewesen. Deswegen und wegen begrenzter Rechenkapazität wird in der abschließenden Untersuchung das Trans-Unet nicht trainiert. Aufgrund mangelnder Zeit und Rechenkapazität wird auf eine separate Hyperparametersuche für CBAM verzichtet. Lediglich eine Untersuchung zur Ermittlung der optimalen Anzahl an Workern wurde separat durchgeführt. Die für DhSegment optimale Zahl von 7 Workern führt zum Beispiel bei DhSegment CBAM dazu, dass pro Batch 3 Sekunden für das Laden der Daten verbraucht wird. Bei DhSegment und 7 Workern dauert das ganze 0,3 Sekunden. Mit 3 Workern reduziert sich das Laden bei CBAM auf 0,8 Sekunden. Genaueres dazu ist in Unterabschnitt 4.3.3 zu finden.

Der ursprüngliche Plan für das finale Training umfasst 400 Epochen in 3

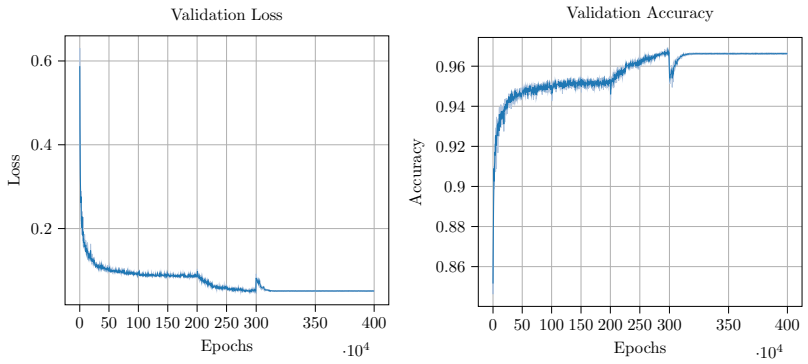


Abbildung 4.12: Verlauf der Validation für die ursprüngliche Version des finalen Trainings von DhSegment. Alle Werte sind über 3 Durchläufe gemittelt. Ab Epoche 300 musste die Batchgröße halbiert werden. Deswegen verdoppelt sich die Anzahl der Schritte und damit der eingenommene Platz auf der x-Achse pro Epoche.

Schritte unterteilt:

- Training über 200 Epochen mit eingefrorenem Encoder.
- Training über 100 Epochen ohne eingefrorenem Encoder. Nach dem ersten Trainingsschritt werden auch die Gewichte des Encoders verändert, um bessere Ergebnisse zu erhalten.
- Training über 100 Epochen mit eingeschaltetem Scheduler. Der Scheduler verringert die Lernrate immer dann, wenn sich der Validations Loss über mehr als eine Epoche nicht verbessert hat. Im Falle von DhSegment wird in diesem Schritt auch das AMP abgeschaltet. In der Untersuchung dazu wurde festgestellt, dass die Ergebnisse ohne AMP besser sind (Unterabschnitt 4.6.1). Für die Durchführung der Hyperparametersuche war die geringere Laufzeit mit AMP wichtiger.

Im Falle von DhSegment musste im letzten Schritt die Batchgröße auf 32 gesenkt werden, um den vorhandenen Grafikkartenspeicher nicht zu überschreiten. Dies ist bei den letzten 100 Epochen in Abbildung 4.12 dadurch zu sehen, dass diese doppelt so viel Platz einnehmen, weil doppelt so viele Schritte für eine Epoche durchgeführt werden. Das Miteinbeziehen des Encoders bezüglich Gradienten Gewichtung ab Epoche 200 ist deutlich dadurch zu erkennen, dass die Werte nicht mehr stagnieren, sondern sich verbessern. Die Abschaltung des AMP Training bei Epoche 300 führt zu einer deutlichen Verschlechterung der Werte. Dies hängt mutmaßlich mit großen Gradienten zusammen, die durch diese Änderung entstehen. Bei einem Trainingsprozess sollte also von Anfang an festgelegt sein, ob AMP verwendet wird oder nicht. Die Werte verbessern sich im nachfolgenden Training wieder, allerdings erreichen sie nicht mehr das vorherige

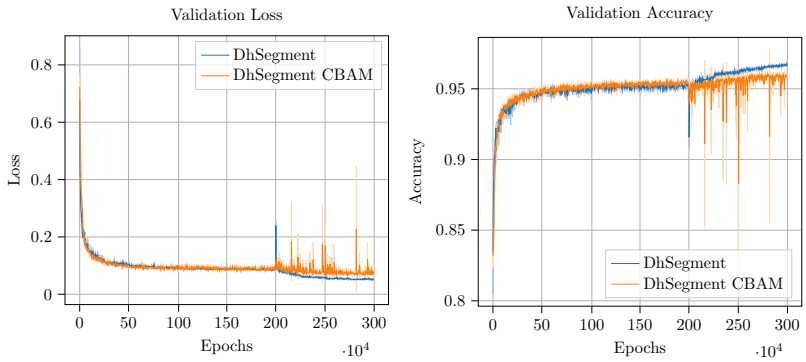


Abbildung 4.13: Verlauf der Validation für die abschließende Version des finalen Trainings von DhSegment und DhSegment CBAM. Alle Werte sind über 3 Durchläufe gemittelt.

Niveau. Ein Grund dafür scheint zu sein, dass die Lernrate durch den Scheduler zu schnell gesenkt wird, was weitere Verbesserungen verhindert. Es wird darauf verzichtet, im weiteren Training einen Scheduler zu verwenden, weil nicht die Zeit vorhanden ist, um diesen korrekt einzustellen. Aufgrund dieser Erkenntnisse wird der 3. Schritt für das Finale Training der 3 Modelle gestrichen.

Abbildung 4.13 zeigt den Verlauf des finalen Trainings mit den übrigen 2 Schritten für DhSegment und DhSegment CBAM. In den ersten 200 Epochen, mit eingefrorenem Encoder verhalten sich beide sehr ähnlich. Danach steigen die Werte bei DhSegment deutlich an, bei DhSegment CBAM verbessern sich die Werte zwar, das Training ist aber deutlich unruhiger und es gibt wiederholt negative Ausschläge. Das könnte auf eine zu hohe Lernrate hindeuten. Diese müsste bei CBAM im 2. Teil des Trainings angepasst werden. Auch ist bei beiden Modellen zu sehen, dass es sich lohnen würde deutlich früher den Encoder mitzutrainieren. Etwa ab Epoche 50 steigt die Leistung der Modelle nur noch in geringem Maße.

Kapitel 5

Ergebnisse

Im Vergleich der 3 Modelle DhSegment, DhSegment CBAM und dem auf DhSegment angepasstem Trans-Unet(Abschnitt 3.3) stellt sich im initialen Training heraus, dass das Trans-Unet deutlich schlechtere Ergebnisse liefert als die anderen beiden Modelle. Deswegen werden abschließend nur DhSegment und DhSegment CBAM verglichen. Abbildung 5.1 zeigt die Ergebnisse des besten Modells beider Trainings, evaluiert mit dem Test-Datensatz. Die allgemeinen Loss, Genauigkeit und Jaccard Werte von DhSegment sind geringfügig besser, als die von DhSegment CBAM. Der Unterschied wird aber durch die Betrachtung der einzelnen Klassen deutlich. Bei Precision und Recall sind die meisten Werte von DhSegment besser als von DhSegment CBAM, deutlich wird das aber erst bei der Betrachtung des CSI Score. Dieser ist deutlich empfindlicher als Recall und Precision alleine. Beide Modelle haben deutlich höhere Precision-, als Recallwerte. Das bedeutet, dass die Pixel, die einer Klasse zugeordnet wurden, auch mit hoher Wahrscheinlichkeit dieser Klasse angehören. Die geringen Recallwerte deutet darauf hin, dass dafür aber einige Stellen nicht als die jeweilige Klasse identifiziert werden. Zu beachten ist dabei, dass die Annotationen nie perfekt sind. Separatoren sind zum Beispiel nie Pixel-Genau markiert, sondern zusammen mit einem Bereich um den eigentlichen Separator herum. Aufgrund der manuellen Annotation gibt es dabei eine gewisse Varianz. Damit ist es nicht überraschend, dass die Recallwerte geringer sind. Für die Klasse UnkownRegion sind die Werte überraschend hoch. Es handelt sich bei diesen Regionen um Fehler in den Daten. Dass diese einen Precision Wert von etwa 60% und einen Recall Wert von über 50% erreichen ist unerwartet. In Abschnitt A.6 ist die reine Vorhersage von DhSegment dargestellt, in welcher zu sehen ist, dass kleine Bereiche zwischen Regionen als UnknownRegion markiert werden. Dies führt anscheinend zu relativ guten Ergebnissen, weswegen das Modell sich auch dieser Zuordnung sicher ist(75% Schwellwert).

Die Ergebnisse dieser Arbeit sind deutlich besser, als die Ergebnisse der Projektgruppe (Projektbericht[33] Abschnitt 4.2). In dieser waren die CSI Werte der meisten Klassen unterhalb von 50 %. Insbesondere die relevanten Überschriften (Heading) wurden mit unter 40 % CSI deutlich schlechter erkannt als in dieser

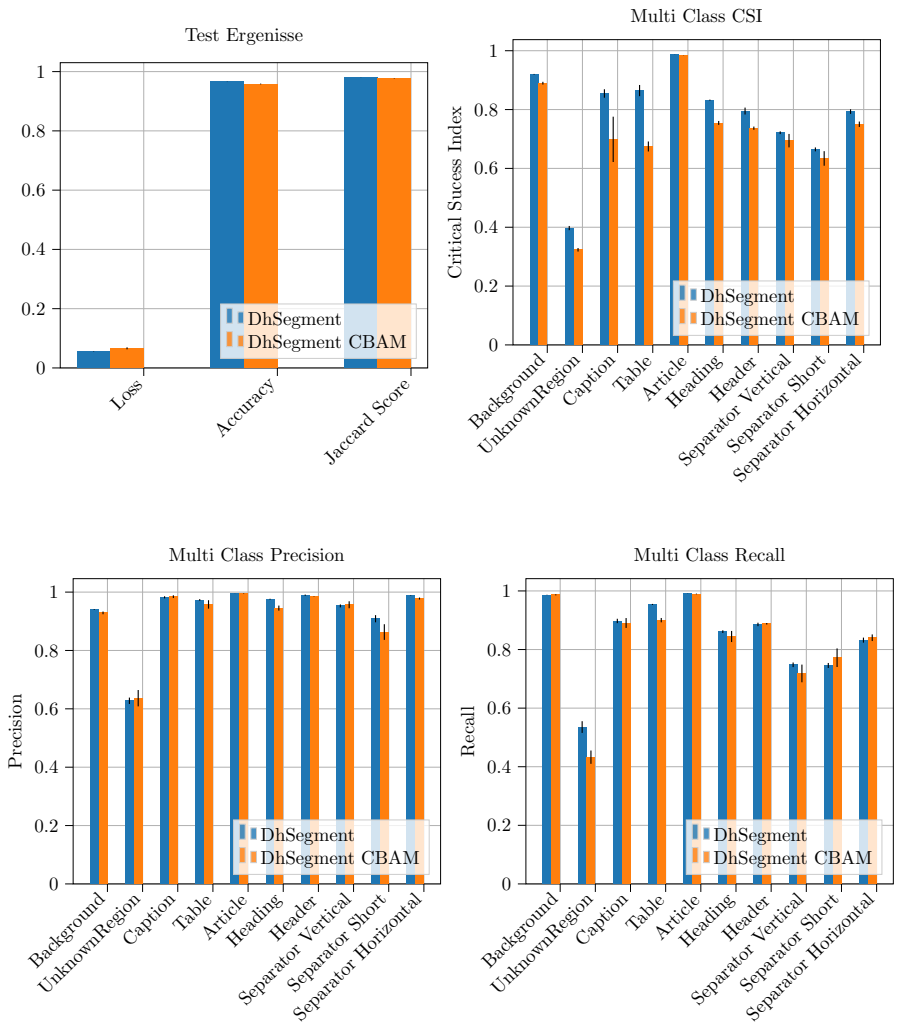


Abbildung 5.1: Testergebnisse für das abschließende Training von DhSegment und DhSegment CBAM. Alle Werte sind über 3 Durchläufe gemittelt.

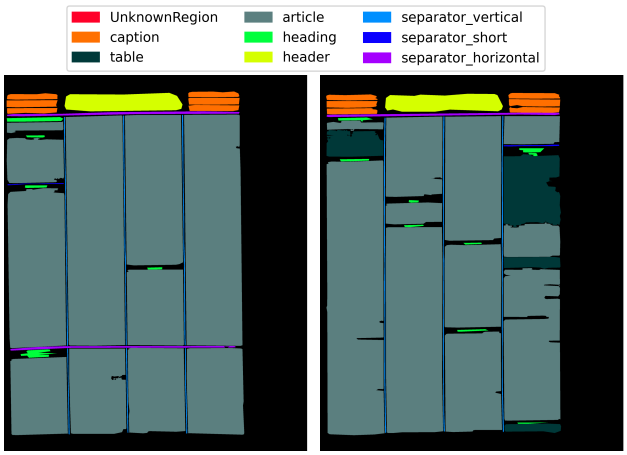


Abbildung 5.2: Polygon Vorhersage von dem finalen DhSegment Modell. Alle gefärbten Pixel liegen über dem Schwellwert von 75 %. Pixel, bei denen die Wahrscheinlichkeit geringer ist, werden als Hintergrund gefärbt. Siehe Anhang Abschnitt A.6 für große Version

Arbeit.

Die guten Ergebnisse bei Precision und Recall bestätigen sich bei der Vorhersage unbekannter Seiten der Kölnischen Zeitung. Abbildung 5.2 zeigt das Ergebnis der in Polygone umgerechneten Vorhersage. Dabei werden Regionen, die zu klein sind oder der Klasse UnkownRegion angehören, ignoriert. Im Anhang Abschnitt A.6 finden sich die originalen Versionen der Vorhersage. Insbesondere die Überschriften (Heading) und Tabellen werden in diesem Beispiel zuverlässig erkannt. Das stellt einen entscheidenden Fortschritt gegenüber allen bisherigen Ergebnissen dar. Sowohl Überschriften als auch Tabellen waren immer die beiden Klassen, die nicht zuverlässig erkannt wurden.

Die Lesereihenfolge(Unterabschnitt 3.2.3) hat noch Probleme mit dem Unterteilen der Seite in Regionen oberhalb und unterhalb der großen Separatoren, weil diese teils schräg über die Seite verlaufen. Auch die verwendete Heuristik zur Identifizierung der nächsten Spalte ist fehleranfällig. Im Anhang Abschnitt A.6 sind Beispiele für den Transkribus Export inklusive der berechneten Lesereihenfolge zu finden. Außerdem wird das am Ende von Abschnitt 4.4 beschriebene reduzierte Modell vom geschichtswissenschaftlichen Institut verwendet, um mithilfe des Slice Exports Zeitungsseiten zu transkribieren und für die Sentiment Analysis aufzubereiten.

Kapitel 6

Schlussfolgerungen und Ausblick

Diese Arbeit leistet einen Beitrag dazu, historische Zeitungen in Zukunft zuverlässig transkribieren zu können. Um die dafür wichtige Layouterkennung zu verbessern, wird der Datensatz der Kölnischen Zeitung verwendet. Dieser wird in Zusammenarbeit mit dem geschichtswissenschaftlichen Institut erstellt.

Der Datensatz enthält Seiten der Kölnischen Zeitung von 1866 und wird in Zukunft um Seiten von 1924 erweitert. Außerdem wäre es sinnvoll, die Stellen, die bisher als UnknownRegion markiert sind, zu korrigieren, sowie Regionen zu identifizieren die bisher schlecht identifiziert werden und die Annotationsanweisungen diesbezüglich zu überprüfen. Ein Beispiel dafür ist bei Abbildung 5.2 im rechten Bild die Überschrift genau über der Tabelle. Beispiele wie die 3. Textzeile ist in den Daten inkonsistent annotiert, weil es fraglich ist, ob es sich dabei um einen Teil der Überschrift, normalen Text oder um einen Teil der Tabelle handelt (Siehe auch Abschnitt A.6 letztes Bild).

Eine gute Layouterkennung ist für eine erfolgreiche Transkription unerlässlich. Bisherige OCR Programme setzen oft unzusammenhängende Textabschnitte zusammen, wodurch die Ergebnisse nicht zu gebrauchen sind. Die in Abschnitt 3.3 beschriebenen Modelle DhSegment, DhSegment CBAM und das angepasste Trans-Unet wurden für diese Aufgabe verglichen. Alle 3 Modelle basieren auf dem in der vorangegangenen Projektgruppe bereits verwendeten DhSegment Modell. Dieses wird dann einerseits mit CBAM Blöcken und andererseits mit einem Transformer erweitert. Trans-Unet zeigte bereits im initialen Training (Abschnitt 4.4) schlechtere Ergebnisse als die anderen beiden Modelle und wurde deswegen nicht in die finale Auswertung einbezogen.

Für DhSegment wurde dank der auf dem JUWELS Supercomputer verfügbaren Rechenkapazität eine Hyperparametersuche (Abschnitt 4.6) durchgeführt. Weil das AMP Training für das CBAM Modell zu schlechten Ergebnissen geführt hat, benötigt das Training deutlich länger als für DhSegment. Deswegen reichte die verfügbare Rechenkapazität nicht für eine Hyperparametersuche für das CBAM

Modell aus.

Die finale Auswertung ergab deutlich bessere Ergebnisse für das ursprüngliche DhSegment Modell (Kapitel 5). Die Ursache dafür scheint in der fehlenden Optimierung des CBAM Modells zu liegen. Die Vorhersage von DhSegment erkennt alle relevanten Klassen zuverlässig genug, um Artikel anhand von Überschriften zu identifizieren und Tabellen, sowie Titelzeilen vom restlichen Text unterscheiden zu können. Das stellt eine entscheidende Verbesserung im Vergleich zu allen bisherigen Ergebnissen auf dem Datensatz der Kölnischen Zeitung dar. Ebenfalls stellt diese Arbeit Exportoptionen zur Verfügung, die die Vorhersage des Modells in Transkribus [44] übertragen und Textregionen für die weitere Verarbeitung ausschneiden können. Diese Exportoptionen können in Zukunft verwendet werden, um die Vorhersage dieses Modells weiterzuverarbeiten. In Zukunft kann auf diesen Ergebnissen aufbauend, der Slice Export so angepasst werden, dass die Textregionen nach Artikeln sortiert exportiert werden. Außerdem sollte die Berechnung der Lesereihenfolge verbessert werden. Dadurch ergeben sich eine Vielzahl an Möglichkeiten für automatische Analysen der Kölnischen Zeitung. Ausgenommen hiervon sind Seiten der Kölnischen Zeitung, die Anzeigen enthalten. Diese sind deutlich anders aufgebaut und wurden bisher nicht ins Training miteinbezogen. In Zukunft könnte ein Modell für diese noch speziell trainiert werden. Außerdem ändert sich das Layout der Kölnischen Zeitung im 20. Jahrhundert (Siehe Angang Abschnitt A.7). Auch dort könnte es nötig sein, ein Modell für diese Seiten zu trainieren.

Diese Arbeit zieht keine abschließende Bewertung bezüglich der 3 verwendeten Modelle, weil nur das DhSegment Modell per Hyperparametersuche optimiert wurde. Es ist nicht klar, ob in der Zukunft die anderen beiden Modelle abschließend bewertet werden, weil die bisherigen Ergebnisse zufriedenstellend sind. Mit den Erkenntnissen aus der Hyperparametersuche sollte es möglich sein, DhSegment auf weiteren Daten zu trainieren und vergleichbare Ergebnisse zu erzielen.

Abkürzungen

AMP Automated Mixed Precision.. 28, 37–39, 43, 48

CBAM Convolutional Block Attention Module. 2, 3, 16, 22, 25, 26, 31, 33, 34, 36, 42, 44, 45, 48, 49

CNN Faltungsnetze (convolutional neural network). 6, 7, 9, 11–16

CSI Critical Sucess Index. 19, 21, 28, 40, 45

GIL Global Interpreter Lock. 29, 30, 32

MLP Multi Layer Perzeptron. Andere Bezeichnung für ein vollständig vernetztes Neurales Netz. 15, 17

NLP Verarbeitung natürlicher Sprache (Natural Language Processing). 2, 10, 12, 14, 15

NN Neuronale Netze. 6, 7

OCR Texterkennung (Optical Character Recognition). 1, 4, 5, 9–11, 18, 22, 24, 35

ReLU Rectified Linear Unit. Weit verbreitete Aktivierungsfunktion für Neuronale Netze. Hat den Vorteil besondern einfach berechnenbar zu sein und trotzdem eine nicht lineare Aktiverungsfunktion darzustellen. Die Verwendung von linearen Aktiverungsfunktionen würde das Verwenden von mehreren Schichten mit gewichteter Summe überflüssig machen. 7

Glossar

Aufmerksamkeit Kernelement von Transformer Modellen. Ist an das menschliche Konzept der Aufmerksamkeit angelehnt. Elementare Teile der Eingabe, wie Pixel oder Wörter, werden in einem Aufmerksamkeitsmodul, zum Beispiel anhand von Ähnlichkeiten, gewichtet. Dadurch kann die Aufmerksamkeit auf bestimmte Teile der Eingabe fokussiert werden. Siehe Unterabschnitt 2.4.1. 2, 12, 14, 25

Augmentierung Eingabe-Daten werden während des Trainings zufällig leicht modifiziert, um die Vielfalt der Daten zu erhöhen. Kann Overfitting entgegenwirken. 21

Backpropagation Berechnet die Gradienten für mehrere Schichten eines Modells. Die Gradienten können dann verwendet werden um die Parameter anzupassen. 25, 31

Bottom-Up Ansätze, die sich von Grundlegenden Eigenschaften zu abstrakterem vorarbeiten. 5

Bounding Box Rechteck, dass alle Pixel einer Region enthält und dabei minimale Ausdehnung hat. 23, 24

Cropping Unterteilung von Eingabe Bildern in Teil-Bilder, die dann für das Training verwendet werden. Das hat den Vorteil, die Daten eines Trainingsschritts als representative Stichprobe aus dem gesamten Datensatz zusammenstellen zu können. Eine größere Anzahl an vollständigen Eingabe-Bildern auf einmal zu verarbeiten ist aufgrund von beschränkten Grafikkartenspeicherplatz meist nicht möglich. Siehe auch Unterabschnitt 3.1.2. 11, 14

Early Stopping Verfahren, welches über das Training das beste Modell speichert. Die Evaluierung dafür erfolgt mit dem Validationsdatensatz. Dient als Mittel gegen Overfitting. 19, 40

Exploding Gradient Große Gradienten können den Fortschritt des Trainings verhindern, weil dadurch in zu großen Schritten Anpassungen an den Parametern durchgeführt werden. Dadurch wird über das Ziel hinausgeschossen und das Resultat des Schritts ist ein deutlich schlechteres Ergebnis als

zuvor. Dem kann entgegengewirkt werden, indem eine maximale Größe (Norm) von Gradienten festgelegt wird und diese entsprechend abgeschnitten werden. (Gradient Clipping) . 39

Feature Map Filter von Faltungsnetzen erzeugen Ausgabe-Bilder, die jeweils Informationen zu einem bestimmten Feature enthalten, auch Feature Maps genannt. 7, 17

Focal Loss Modifizierte Version der Kreuzentropie, die Vorhersagen gewichtet, abhängig davon, wie sicher sich das Modell bei dieser Vorhersage ist. Ist ein Pixel zum Beispiel mit 90% Wahrscheinlichkeit einer Klasse zugeorndet worden, so wird der Fehler von 10% deutlich niedriger gewichtet als ein anderer Pixel, der eine geringerer Wahrscheinlichkeit hat [26]. 19, 37–39

Hyperparametersuche Evaluation von Hyperparametern über mehrere Trainingsdurchläufe. Im Gegensatz zu den Parametern des trainierten Modells beinflussen Hyperparameter das gesamte Training und werden nicht durch das Training angepasst. Zu Hyperparametern gehören zum Beispiel die Lernrate und Wahl der Kostenfunktion. 3

Intersection over Union Metrik für die Qualität von Segmentierungs Modellen. Die Schnittmenge zweier Datensätze wird durch deren Vereinigung geteilt. Im Fall von einer Vorhersage und einer gewünschten Ausgabe, entspricht der Schnitt allen korrekt Idintifizierten Beispielen (True Positive) und die Vereinigung besteht aus allen Daten, die entweder als richtig markiert wurden(True Positives + False Positives) oder richtig sein sollen(True Positives + False Negatives). Je größer die Übereinstimmung zwischen den Mengen der Vorhersage und der gewünschten Ausgabe ist, desto kleiner sind Intersection over Union Verlustfunktionen. Dazu gehören der Dice-Loss und Jaccard-Score [49]. 11, 19

Kanäle Die Kanal-Dimension der Ausgabe von Faltungsschichten umfasst die verschiedenen Feature Maps. Jeder Kanal entspricht einer Feature Map. 20

Kostenfunktion Berechnet die Kosten, die im Training minimiert werden sollen. Dazu gehört der Fehler zwischen Vorhersage des Modells und der gewünschten Ausgabe, sowie gegebenenfalls Normterme, die Gewichte klein halten sollen. 11

Kreuz Entropie Bei der Kreuzentropie handelt es sich um eine Verlustfunktion, welche die Übereinstimmung von Wahrscheinlichkeiten der Vorhersage mit den gewünschten Wahrscheinlichkeitswerten bewertet. Bei Modellen des maschinellen Lernens ist es üblich, dass die Ausgabe eines Modells Wahrscheinlichkeiten für bestimmte Klassen enthält. Je weiter diese Vorhersage von der gewünschten Ausgabe abhängt, desto größer wird der Wert der Kreuzentropie [12]. 11, 19, 37–39

Neurales Netz Ein auf dem Konzept von biologischen Neuronen basierendes Modell des maschinellen Lernens. Künstliche Neuronen werden meist in Schichten angeordnet und die Gewichte zwischen den Schichten werden beim Training angepasst, um Daten zu verarbeiten. 5

Optimierer Verantwortlich für die Anpassung der Parameter eines Modells während des Trainings. Die berechneten Gradienten werden je nach gewählten Optimierer angepasst um ein besseres Konvergenz Verhalten zu erhalten. 11

Pooling Pooling ist eine Operation, die die Dimension der Eingabe reduzieren soll ohne dabei zu viele Informationen zu verlieren. Dafür wird zum Beispiel das Maximum von einem 2x2 Bereich in einem Bild in einem Pixel abgespeichert. Es wird also nicht nur der Wert des maximalen Pixels übernommen, sondern auch die Information, dass keines der anderen 3 Pixel einen höheren Wert hat. 2, 8, 9, 16, 17, 22

Precision Metrik für die Korrektheit der erkannten Pixel. Je höher der Precision Wert ist, desto wahrscheinlicher ist es, dass die markierten Pixel auch tatsächlich der markierten Klasse angehören. 33–36, 41, 42, 45, 47

Recall Metrik für die Erkennung von Pixeln einer Klasse. Je höher der Recall Wert ist, desto höher ist die Wahrscheinlichkeit, einen Pixel, der einer Klasse zugehört, auch als solche zu markieren. 34, 36, 41, 42, 45, 47

rezeptives Feld Angelehnt an das rezeptive Feld von biologischen Neuronen. Faltungsnetze haben für die Vorhersage eines Pixels der Ausgabe einen begrenzten Teil der Eingabe zu Verfügung. Eingabe-Pixel, die außerhalb des rezeptiven Feldes liegen können nicht verarbeitet werden. 12

Scheduler Reduziert die Lernrate im laufe des Trainings. Das ist dazu da, die Schrittgröße zu verringern, sobald das Training in der Nähe des Minimums angelangt ist. 43, 44

Sinusoidal Embedding Positional Embedding, welches keine feste Eingabegröße hat, sondern Sinus Wellen unterschiedlicher Frequenz auf die Eingabe anwendet und dadurch Informationen über die Position liefert. 25

Skip-Connection Ein Element in Modellen des maschinellen Lernens, wodurch die Eingabe eines Moduls an dessen Ausgabe addiert wird. Dadurch überspringt dieser Teil der Summe das Modul, was zu einer besseren Propagation der Gradienten beitragen kann und dem Verschwinden von Gradienten entgegenwirkt. 9

Softmax Nicht lineare Aktivierungsfunktion, die Eingaben entlang einer bestimmten Dimension auf Wahrscheindlichkeiten abbildet. Das bedeutet, die Werte summieren zu Eins und können in Prozeptwerte umgerechnet werden. 22

Split and Merge Ansätze, die zu verarbeitende Daten zunächst aufteilen um diese nach bestimmten Kriterien kategorisieren. Danach werden die Daten anhand der Kriterien wieder zusammengefügt. 5

unüberwacht Unüberwachtes Training benötigt keine von Menschen annotierten Ziel-Daten sondern verwendet die Eingabe als Ziel-Ausgabe oder generiert die Ziel-Ausgaben selber. Dadurch können deutlich größere Datenmengen genutzt werden als bei überwachtem lernen. 14

Literatur

[1] *1D, 2D, and 3D Sinusoidal Postional Encoding (Pytorch and Tensorflow).* https://github.com/tatp22/multidim-positional-encoding. 2023.

[2] Kayla Kelly Adam Kelly. *Create COCO Annotations From Scratch.* URL: `https://www.immersivelimit.com/tutorials/create-coco-annotations-from-scratch/#create-custom-coco-dataset` (besucht am 11. 11. 2023).

[3] Abhinav Ajitsaria. *What Is the Python Global Interpreter Lock (GIL)?* URL: `https://realpython.com/python-gil/` (besucht am 07. 11. 2023).

[4] Raphaël Barman u. a. „Combining visual and textual features for semantic segmentation of historical newspapers". In: *Journal of Data Mining & Digital Humanities* (2021).

[5] HCP Uni Bonn. *Bender – GPU Cluster.* URL: `https://www.hpc.uni-bonn.de/en/systems/bender` (besucht am 25. 11. 2023).

[6] Jieneng Chen u. a. „Transunet: Transformers make strong encoders for medical image segmentation". In: *arXiv preprint arXiv:2102.04306* (2021).

[7] Alex Clark. *Pillow.* URL: `https://python-pillow.org/` (besucht am 11. 11. 2023).

[8] Christian Clausner u. a. „The ENP image and ground truth dataset of historical newspapers". In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR).* 2015, S. 931–935. DOI: `10.1109/ICDAR.2015.7333898`.

[9] Christian Clausner u. a. „The ENP image and ground truth dataset of historical newspapers". In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR).* 2015, S. 931–935. DOI: `10.1109/ICDAR.2015.7333898`.

[10] Jia Deng u. a. „Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition.* Ieee. 2009, S. 248–255.

[11] scikit-learn developers. *scikit-learn-metrics.* URL: `https://scikit-learn.org/stable/modules/model_evaluation.html` (besucht am 05. 11. 2023).

[12] Read the Docs Team. *Cross-Entropy.* URL: `https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html` (besucht am 23. 11. 2023).

[13] Alexey Dosovitskiy u. a. „An image is worth 16x16 words: Transformers for image recognition at scale“. In: *arXiv preprint arXiv:2010.11929* (2020).

[14] Vincent Dumoulin und Francesco Visin. „A guide to convolution arithmetic for deep learning“. In: *arXiv preprint arXiv:1603.07285* (2016).

[15] Linux Foundation. *Pytorch*. URL: `https://pytorch.org/`.

[16] Ralph Furmaniak. „Unsupervised newspaper segmentation using language context“. In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. Bd. 2. IEEE. 2007, S. 1263–1267.

[17] Basilios Gatos, SL Mantzaris und Apostolos Antonacopoulos. „First international newspaper segmentation contest“. In: *Proceedings of Sixth International Conference on Document Analysis and Recognition*. IEEE. 2001, S. 1190–1194.

[18] Basilios Gatos u. a. „Automatic page analysis for the creation of a digital library from newspaper archives.“ In: *International Journal on Digital Libraries* 3.1 (2000).

[19] Sean Gillies und Shapely contributors. *object.simplify*. URL: `https://shapely.readthedocs.io/en/stable/manual.html#object.simplify` (besucht am 11. 11. 2023).

[20] Shawn Graham u. a. *Exploring big historical data: the historian's macroscope*. World Scientific, 2016.

[21] Kaiming He u. a. „Deep Residual Learning for Image Recognition“. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, S. 770–778. DOI: `10.1109/CVPR.2016.90`.

[22] HELMHOLTZ. *Jülich Wizard for European Leadership Science*. URL: `https://www.fz-juelich.de/en/ias/jsc/systems/supercomputers/juwels` (besucht am 25. 11. 2023).

[23] Jae Kyu Im. *CBAM: Convolutional Block Attention Module (PyTorch)*. URL: `https://github.com/Peachypie98/CBAM` (besucht am 07. 12. 2023).

[24] Salman Khan u. a. „Transformers in vision: A survey“. In: *ACM computing surveys (CSUR)* 54.10s (2022), S. 1–41.

[25] Bernhard Liebl und Manuel Burghardt. „An evaluation of DNN architectures for page segmentation of historical newspapers“. In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, S. 5153–5160.

[26] Tsung-Yi Lin u. a. „Focal loss for dense object detection“. In: *Proceedings of the IEEE international conference on computer vision*. 2017, S. 2980–2988.

[27] Martín Abadi u. a. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[28] Jiří Martínek, Ladislav Lenc und Pavel Král. „Building an efficient OCR system for historical documents with little training data“. In: *Neural Computing and Applications* 32 (2020), S. 17209–17227.

[29] Benjamin Meier u. a. „Fully convolutional neural networks for newspaper article segmentation“. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Bd. 1. IEEE. 2017, S. 414–419.

[30] Phillip E Mitchell und Hong Yan. „Newspaper document analysis featuring connected line segmentation“. In: *Proceedings of sixth international conference on document analysis and recognition*. IEEE. 2001, S. 1181–1185.

[31] Tom Monnier und Mathieu Aubry. „docExtractor: An off-the-shelf historical document element extraction“. In: *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE. 2020, S. 91–96.

[32] Nicki Skafte Detlefsen u. a. *TorchMetrics - Measuring Reproducibility in PyTorch*. Feb. 2022. DOI: 10.21105/joss.04101. URL: https://github.com/Lightning-AI/torchmetrics.

[33] Christian Schultze Niklas Kerkfeld. *Segmentierung von Zeitungsseiten*. Feb. 2023.

[34] Inc. NumFOCUS. *Numpy*. URL: https://numpy.org/ (besucht am 11. 11. 2023).

[35] Keiron O'Shea und Ryan Nash. „An introduction to convolutional neural networks“. In: *arXiv preprint arXiv:1511.08458* (2015).

[36] Sofia Ares Oliveira, Benoit Seguin und Frédéric Kaplan. „dhSegment: A generic deep-learning approach for document segmentation“. In: *CoRR* abs/1804.10371 (2018). arXiv: 1804.10371. URL: http://arxiv.org/abs/1804.10371.

[37] Sofia Ares Oliveira, Benoit Seguin und Frederic Kaplan. „dhSegment: A generic deep-learning approach for document segmentation“. In: *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE. 2018, S. 7–12.

[38] Chulwoo Pack u. a. „Augmentation-based Pseudo-Ground truth Generation for Deep Learning in Historical Document Segmentation for Greater Levels of Archival Description and Access“. In: *Journal on Computing and Cultural Heritage (JOCCH)* 15.3 (2022), S. 1–21.

[39] Prajit Ramachandran u. a. „Stand-alone self-attention in vision models“. In: *Advances in neural information processing systems* 32 (2019).

[40] *reichsanzeiger-gt*. https://github.com/UB-Mannheim/reichsanzeiger-gt. Accessed: 2023-05-28.

[41] Leonard Richardson. *BeautifulSoup*. URL: https://www.crummy.com/software/BeautifulSoup/ (besucht am 11. 11. 2023).

[42] Xin Rong. „word2vec parameter learning explained“. In: *arXiv preprint arXiv:1411.2738* (2014).

[43] Olaf Ronneberger, Philipp Fischer und Thomas Brox. „U-net: Convolutional networks for biomedical image segmentation“. In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer. 2015, S. 234–241.

[44] READ-COOP SCE. *Transkribus*. URL: `https://readcoop.eu/de/transkribus/` (besucht am 28. 11. 2023).

[45] SCHEDMD. *SLURM workload manager*. URL: `https://slurm.schedmd.com/documentation.html` (besucht am 25. 11. 2023).

[46] Fahad Shamshad u. a. „Transformers in medical imaging: A survey“. In: *Medical Image Analysis* (2023), S. 102802.

[47] Robin Strudel u. a. „Segmenter: Transformer for Semantic Segmentation“. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Okt. 2021, S. 7262–7272.

[48] Vineeth S Subramanyam. *Herzog August Bibliothek Wolfenbüttel*. URL: `https://ocr-d.de/de/gt-guidelines/trans/impressum.html` (besucht am 28. 11. 2023).

[49] Vineeth S Subramanyam. *IOU (Intersection over Union)*. URL: `https://medium.com/analytics-vidhya/iou-intersection-over-union-705a39e7acef` (besucht am 23. 11. 2023).

[50] scikit-image team. *Contour finding*. URL: `https://scikit-image.org/docs/stable/auto_examples/edges/plot_contours.html` (besucht am 11. 11. 2023).

[51] scikit-image development team. *View-as-windows*. URL: `https://scikit-image.org/docs/stable/api/skimage.util.html?highlight=view%20window#skimage.util.view_as_windows` (besucht am 11. 11. 2023).

[52] Turner-Trauring. *Python's multiprocessing performance problem*. URL: `https://pythonspeed.com/articles/faster-multiprocessing-pickle/` (besucht am 07. 11. 2023).

[53] Ashish Vaswani u. a. „Attention is all you need“. In: *Advances in neural information processing systems* 30 (2017).

[54] Xiaolong Wang u. a. „Non-local neural networks“. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, S. 7794–7803.

[55] Karl Weiss, Taghi M Khoshgoftaar und DingDing Wang. „A survey of transfer learning“. In: *Journal of Big data* 3.1 (2016), S. 1–40.

[56] Paul Stefan Williams und Michael D Alder. „Generic texture analysis applied to newspaper segmentation“. In: *Proceedings of International Conference on Neural Networks (ICNN'96)*. Bd. 3. IEEE. 1996, S. 1664–1669.

[57] Kwan Y. Wong, Richard G. Casey und Friedrich M. Wahl. „Document analysis system“. In: *IBM journal of research and development* 26.6 (1982), S. 647–656.

[58] Sanghyun Woo u. a. „Cbam: Convolutional block attention module“. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, S. 3–19.

[59] Corinna Cortes Yann LeCun und Christopher J.C. Burges. *object.simplify.* URL: `http://yann.lecun.com/exdb/mnist/` (besucht am 23. 11. 2023).

[60] Zhilu Zhang und Mert Sabuncu. „Generalized cross entropy loss for training deep neural networks with noisy labels“. In: *Advances in neural information processing systems* 31 (2018).

Anhang A

Anhang

A.1 Trainingsgeschwindigkeit

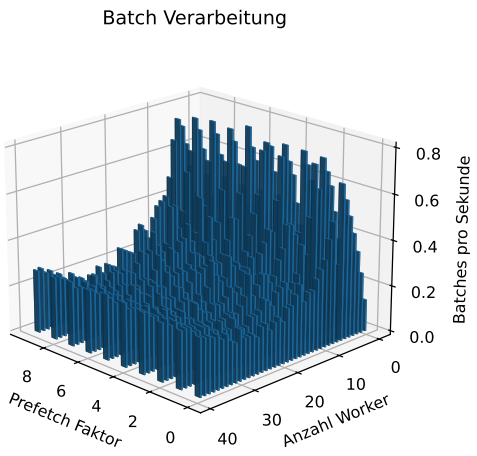


Abbildung A.1: Trainingsgeschwindigkeit gemessen über 5 Epochen in Abhängigkeit von Anzahl Workern und Prefetch Faktor im torch Dataloader. Das Training wurde durchgeführt mit 79 Batches pro Epoche, wobei jeder Batch 256 512x512 Crops entspricht. Die vorhersage und loss Berechnung findet auf 4 Grafikkarten mit jeweils 64 Crops und aktiviertem autoamted mixed precision statt. Zusätzlich gibt es 3 Validationsrunden pro Epochen, deren Dauer zum Training zählt.

A.2 Verlauf Initiales Training

Es folgen Graphen, die den Verlauf des initialen Trainings darstellen. Weil der Datensatz sehr hochauflösende Bilder enthält, werden diese vor dem Training herunterskaliert. Diese Umstellung ist während des Trainings von DhSegment geschehen. Deswegen sieht man bei 87 Epochen eine plötzliche Verschlechterung der Ergebnisse. Das liegt daran, dass die Trainingsdaten ab dem Zeitpunkt um die Hälfte herunterskaliert sind und das Modell sich erst den veränderten Daten anpassen muss. Ebenfalls reduziert sich durch das Skalieren die Anzahl der Batches pro Epoche deutlich. Deswegen nehmen die Epochen ab 87 deutlich weniger Platz auf der x-Achse ein. Beim Trans-Unet nehmen die Epochen ab 150 mehr Platz ein. Durch die zusätzliche Berechnung der Gradienten des Encoders steigt der Bedarf an Grafikkarten Speicher. In diesem Fall hat das dazu geführt, dass die Batchgröße von 32 auf 16 reduziert werden musste. Deswegen nehmen die nachfolgenden Epochen doppelt so viel Platz ein, entsprechend der doppelten Anzahl an Batches.

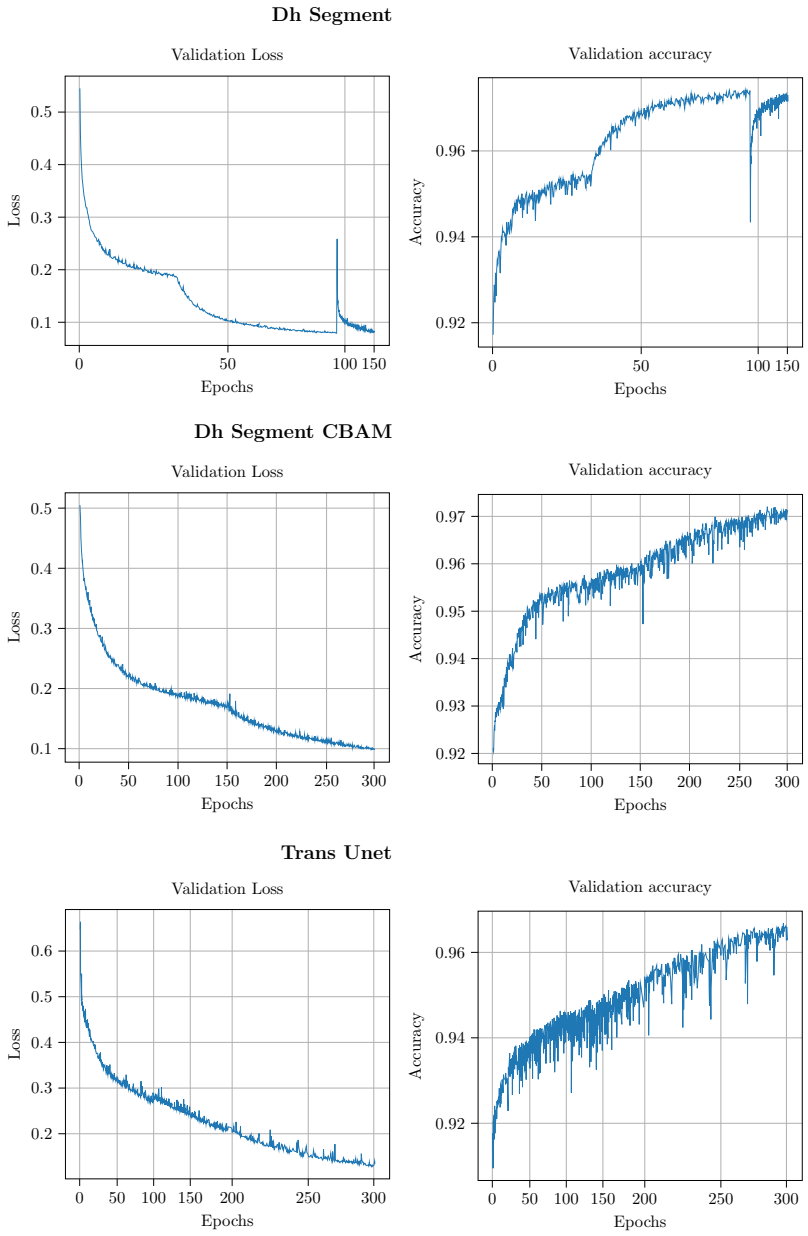


Abbildung A.2: Validations-Ergebnisse für das initiale Training der 3 Modelle. Die Auflösung der Daten wurde während des Trainings von DhSegment geändert und während des Trainings von Trans Unet wurde die Batchgröße verändert. Deswegen verändern sich der Platz, den eine Epoche auf der X Achse einnimmt.

A.3 Vorhersage Initiales Training

Es folgt ein Bild einer Seite der Kölnischen Zeitung, sowie die Vorhersage der DhSegment und DhSegment CBAM Modelle. Alle gefärbten Pixel liegen über dem Schwellwert von 75 %. Pixel, bei denen die Wahrscheinlichkeit geringer ist, werden als Hintergrund gefärbt.

Nr. 10. — Erstes Blatt.

Kölnische Zeitung.

Dinstag, 10. Januar 1865.

Bestellungen für das 1. Quartal 1865 wolle man in Köln bei der Expedition, auswärts bei der nächsten Postanstalt (in Aachen, Bonn, Coblenz, Düsseldorf, Paris und London auch bei unseren Agenten) baldigst machen.

Uebersicht.

Zur ungarischen Frage.

Deutschland.

Frankreich.

Italien.

Schweiz.

Belgien.

Niederlande.

Dänemark.

Schweden.

Amerika.

Zur ungarischen Frage.

* Köln, 9. Januar.

Man schreibt uns aus Wien unterm 7. Jan.:

Die altconservative Partei in Ungarn hat zur Aufrechterhaltung ihrer Politik, deren traurige Resultate bekannt, zu allen ein Gesetz (das Provisorium kann man es nicht nennen) unter dem Titel: «Zehn Jahre Verfassungsfriede» erdichtet lassen, in welchem der Wortführer dieser hochgeborenen und reichen Herren mit einem Aufwand von Humor glaubt, der ein mit den ungarischen Verhältnissen nicht vertrauten Publicum leicht irre führen könnte. Wer hat unter den Neuzehn Jahren Verfassungsfriede, das gegenwärtige Ausnahmsjahr nicht kennen gelernt, das in einer und der andern Provinz zwei bis drei Jahre hindurch, das in einer und der andern Provinz, wie z. B. in Siebenbürgen, fast ununterbrochen fortdauert hat? Wer hat unter den Neunzehn Jahren Verfassungsfriede nicht die Wahrnehmung gemacht, daß die Verwaltung, die Justiz, die Gesetzgebung, die Finanzen, das Militärwesen, kurz alle Zweige des Staatslebens, ja selbst die Gemeinden und Privatverhältnisse, ausschließlich und lediglich von den Befehlen des Wiener Cabinets abhängig waren? Wer hat unter den Neunzehn Jahren Verfassungsfriede nicht die Wahrnehmung gemacht, daß die Gesetze, welche die Wiener Regierung für Ungarn und Siebenbürgen gab, nicht nur nicht in Kraft traten, sondern daß sie nicht einmal publicirt wurden

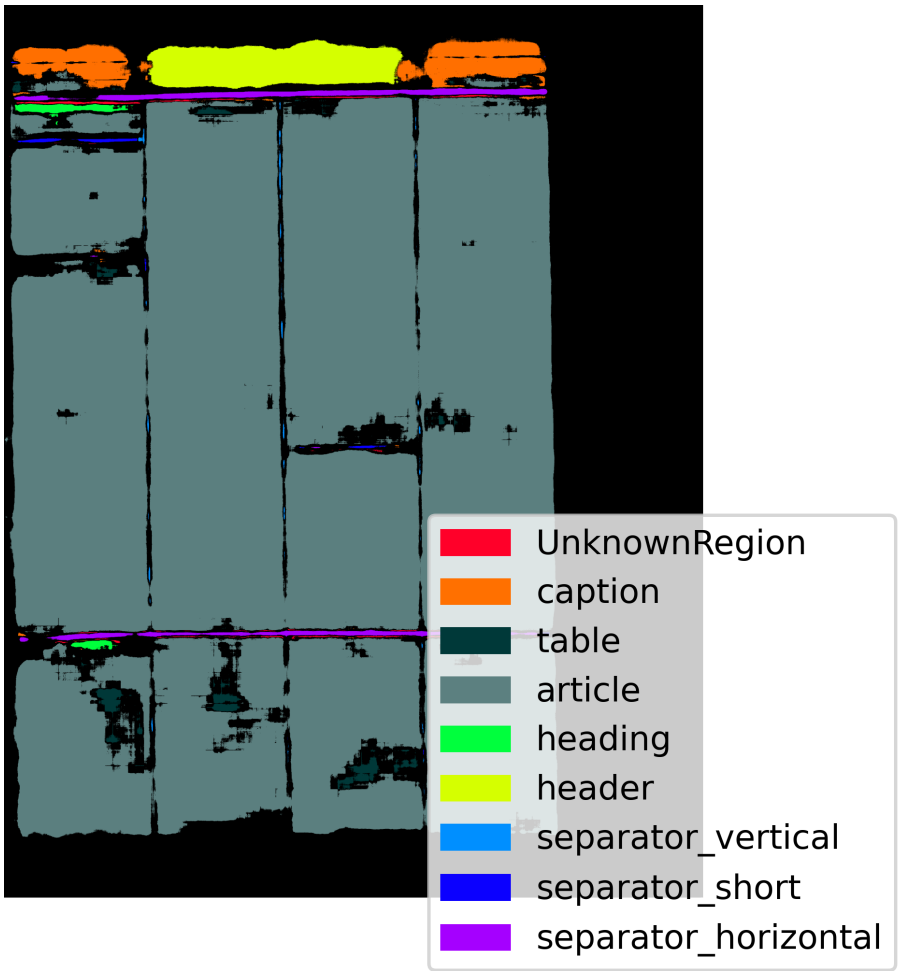


Abbildung A.4: Vorhersage DhSegment

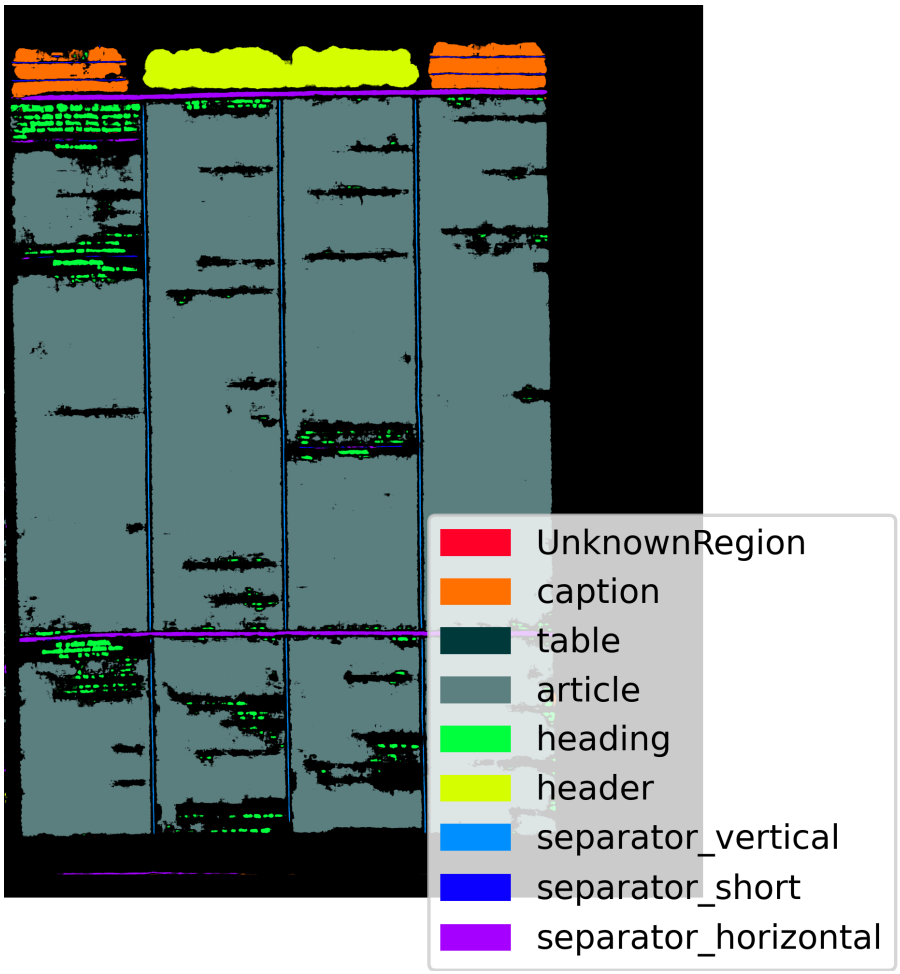


Abbildung A.5: Vorhersage DhSegment CBAM

A.4 Ergebnisse reduziertes Modell

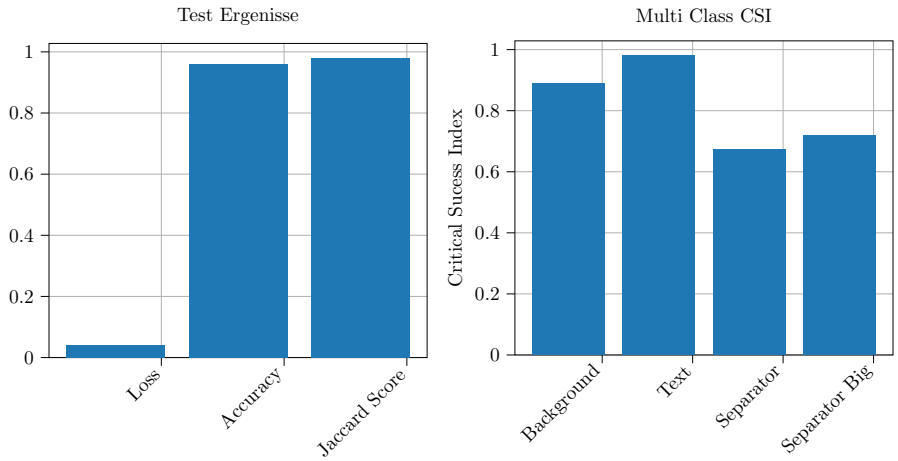


Abbildung A.6: Testergebnisse für das Training mit reduzierten Klassen. Die Textklasse enthält alle Textregionen und die Separator-Klasse enthält alle kleinen Separatoren, die von den großen Unterschieden werden.

A.5 Ergebnisse Skalierung

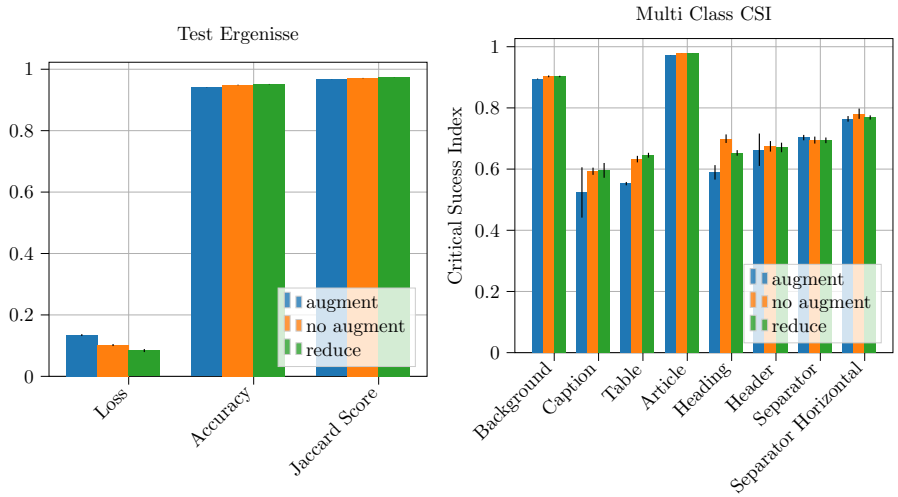


Abbildung A.7: Testergebnisse für das Training ohne Skalierungs-Augmentierung und mit reduzierten Klassen.

A.6 Vorhersage des finalen Modells

Es folgen Original Bilder der Kölnischen Zeitung, die dazugehörige rohe Vorhersage sowie der dazugehörigen Bilder vom Transkribus Export mit Lesereiheinfolge. Alle gefärbten Pixel liegen über dem Schwellwert von 75 %. Pixel, bei denen die Wahrscheinlichkeit geringer ist, werden als Hintergrund gefärbt.

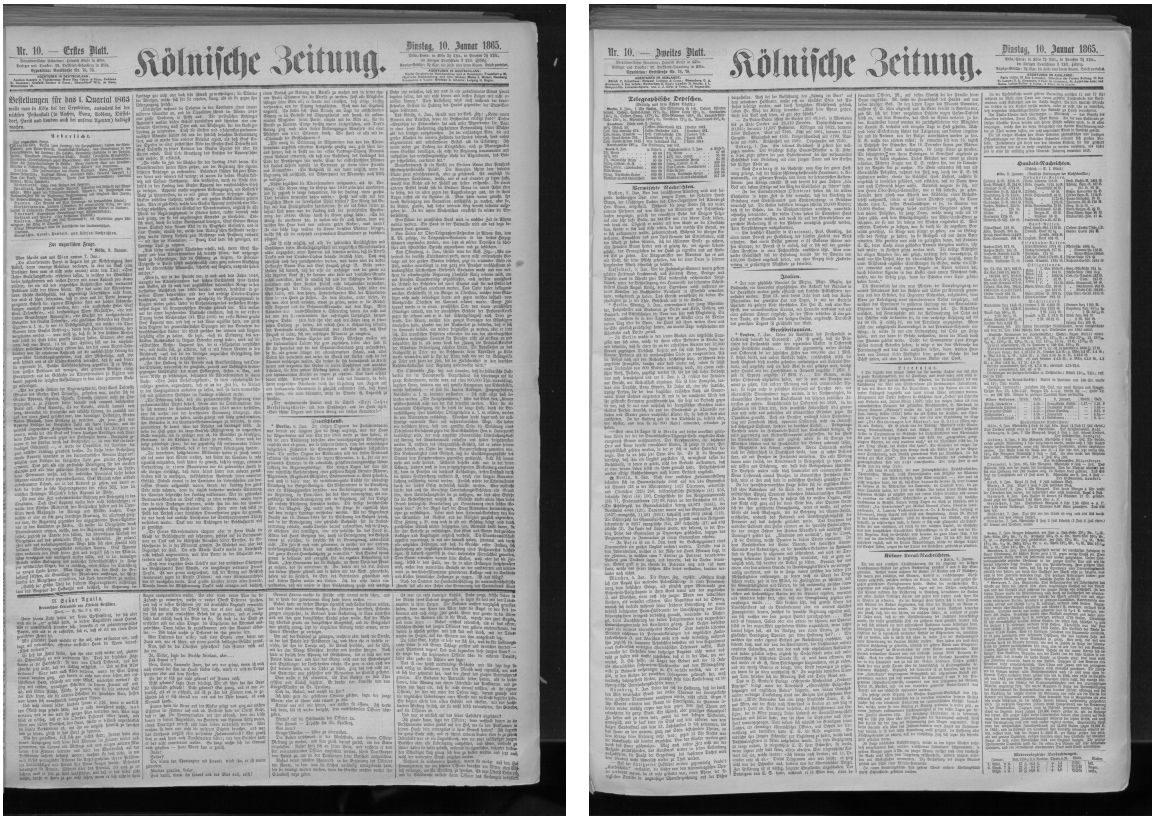


Abbildung A.8: Originale der Kölnischen Zeitung von 1865

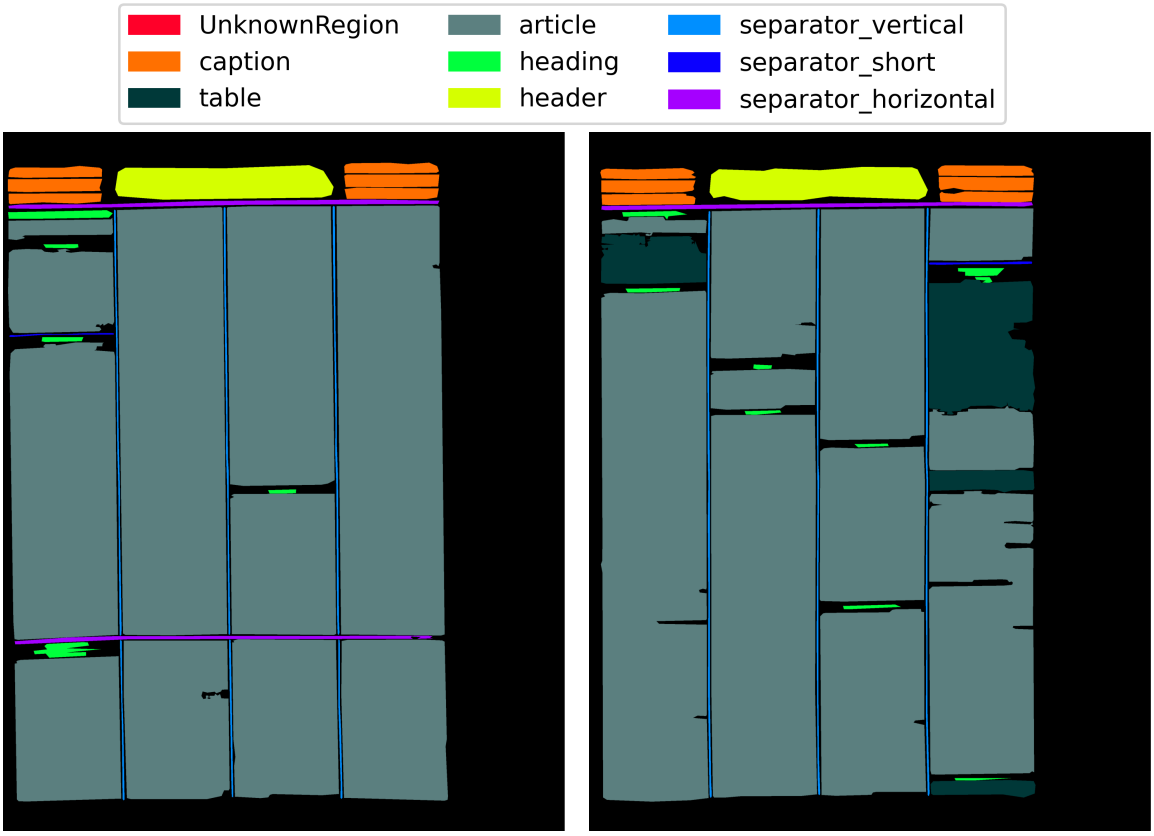


Abbildung A.9: Polygon Vorhersage von dem finalen DhSegment Modell. Alle gefärbten Pixel liegen über dem Schwellwert von 75 %. Pixel, bei denen die Wahrscheinlichkeit geringer ist, werden als Hintergrund gefärbt.

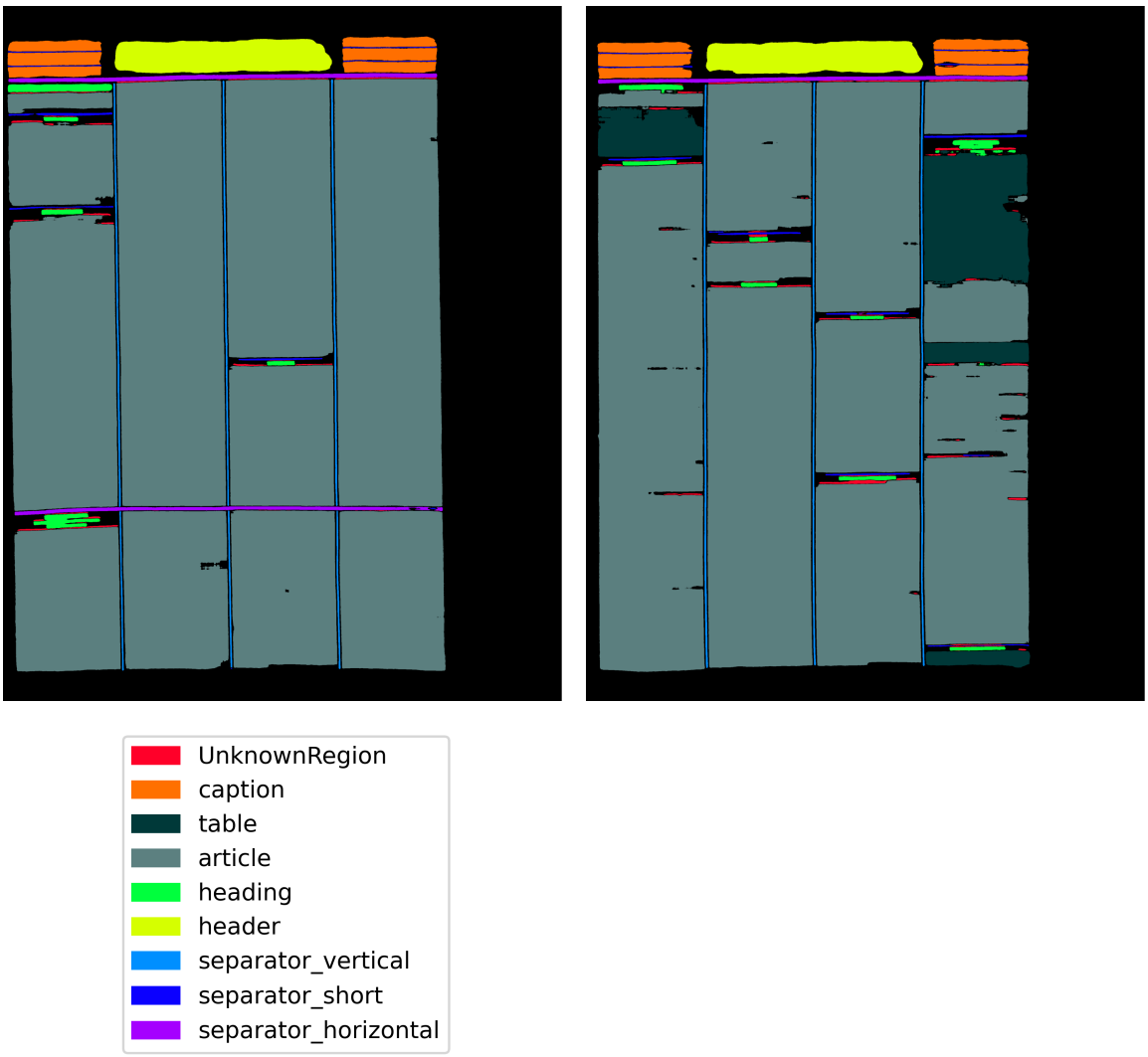


Abbildung A.10: Rohe Vorhersage von dem finalen DhSegment Modell.

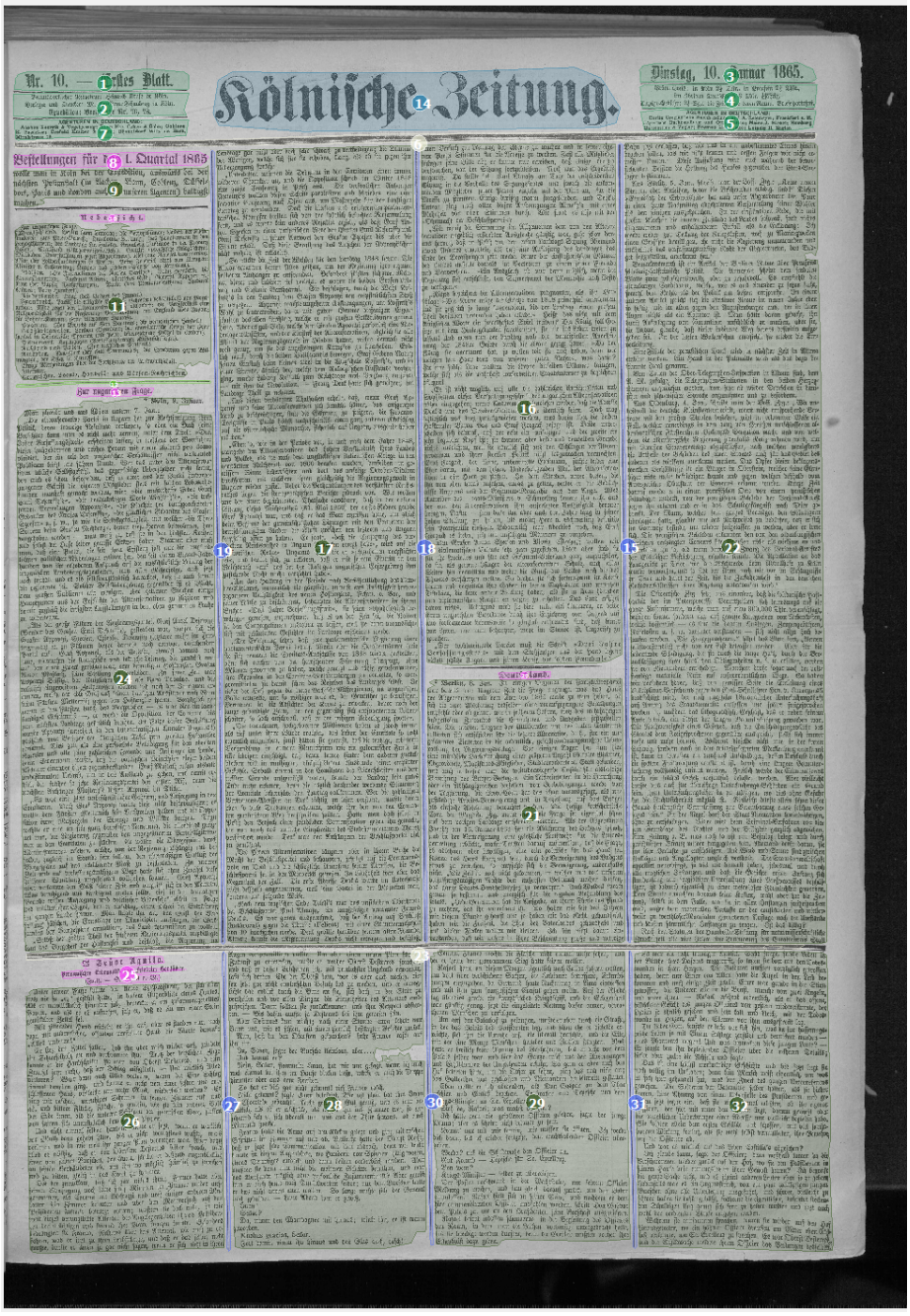


Abbildung A.11: Transkribus Export Titelseite

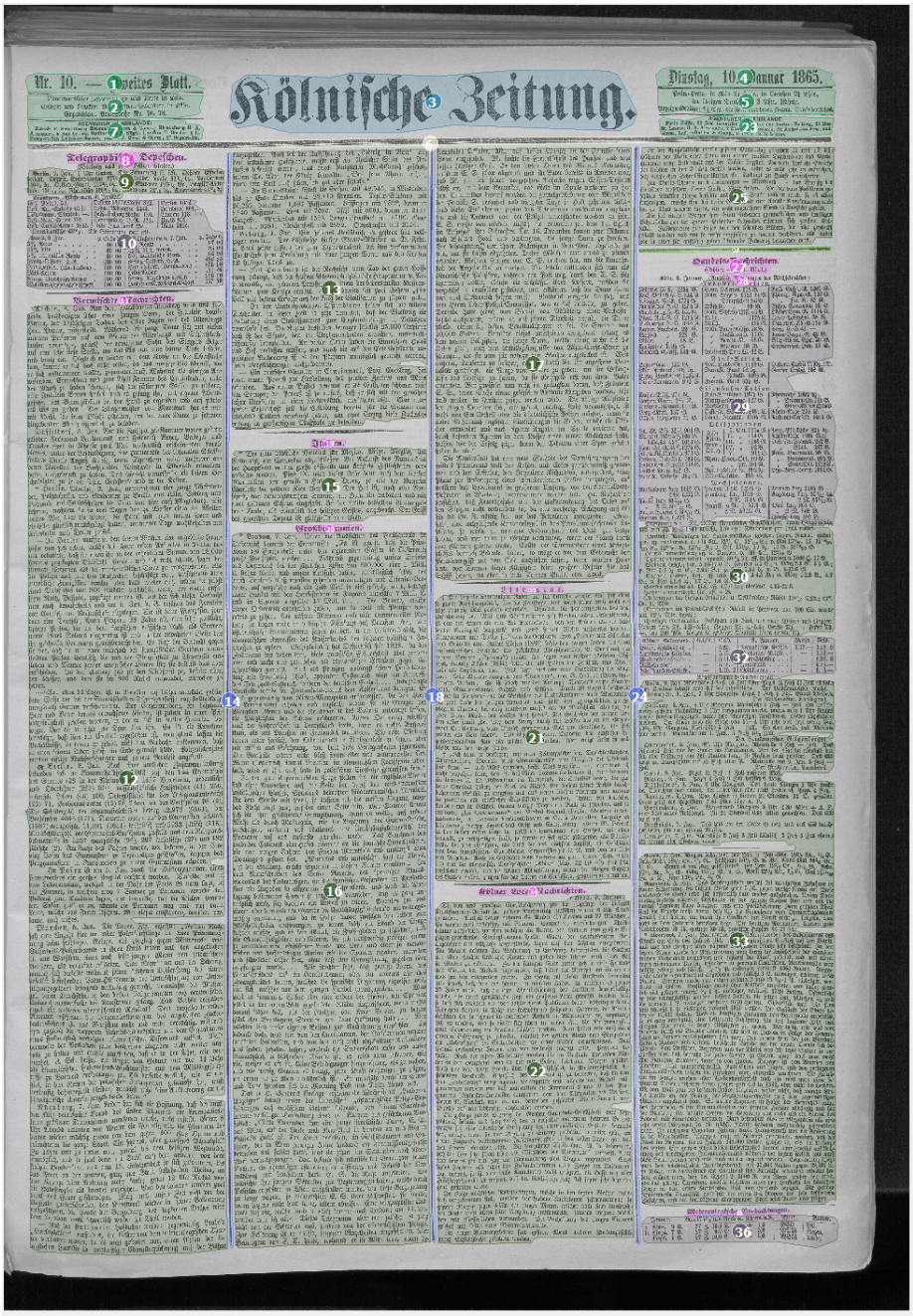


Abbildung A.12: Transkribus Export Seite Titelseite 2

A.7 1924 Zeitungsseiten

Dienstag 1. Januar

Kölnische Zeitung

Morgen-Ausgabe

Einzelverkaufspreis 200 Milliarden Mark

1924 – Nr. 1

Eine Umdeutung der deutschen Absicht.

Die Schuld des Scheiterns soll Deutschland zugeschoben werden.

WTB **Berlin**, 31. Dezember. (Lesrg.)

Die Botschafterkonferenz hat am 29. Dezember folgende Mitteilung veröffentlicht:

Während die Regierungen von Frankreich und Belgien im Einverständnis mit der deutschen Regierung es für angebracht gehalten haben, die am Montag überreichte Denkschrift der Sachverständigen nicht zu übergeben, um in die eingeleiteten Verhandlungen möglichst ja erleichternd zu wirken, hat der Botschafterrat der Deutschheit in der englischen Presse zu seiner Überraschung die Nachricht gefunden, daß die Note bei dem Empfang, einen so unfreundlichen und schroffen Charakter gehabt habe, daß sie jede Aussicht, sie zum Ausgangspunkt ernsthafter Verhandlungen zu machen, illusorisch mache. Auf diese Nachrichten hin hat der Botschafterrat sich veranlaßt gesehen, eine amtliche Feststellung darüber zu veranlassen, daß weder der Ton noch der Charakter, noch der Inhalt der Note irgendwie zu solchen Befürchtungen Anlaß gegeben haben. Es bedeutet es sich für das Reich nicht leisten, eine Ablehnung an den Tag der Beratungen aufzuführen oder Lust zu suchen. Es sind einige und schwere Bedingungen, die hinsichtlich der Sachlage und der Verpflichtungen, die Deutschland bezüglich der Lösung der Reparationsfrage übernommen hat, den deutschen Gegnern zu machen sind. Es ist aber nicht angängig, daß eine Umdeutung der Absichten, die Deutschland in seiner Note vom 29. Dezember ausgesprochen hat, in der Weise vorgenommen wird, daß durch sie die Verantwortung für das Scheitern der Verhandlungen Deutschland zugeschoben werden kann. Deutschland hat sich bereit erklärt, seine Vorschläge zu erläutern und zu ergänzen, und es ist nicht seine Schuld, wenn diese Erklärungen nicht gegeben worden sind.

Die vorgestrige Veröffentlichung der deutschen Denkschrift, die bei den Besprechungen in Paris und Brüssel getroffenen Vereinbarung zuwiderläuft, wird auch deutscherseits bedauert, zumal sie in einer Zeit der französischen Wahlen vorgenommen worden ist, die kaum geeignet sind, hier in Ruhe sachlicher Atmosphäre zu führen. Die deutschen amtlichen Feststellungen sind bis jetzt einem amtlichen Bericht dieser Sachverständigenkonferenz nicht hat erfolgt, daß man in Paris die in Berlin in solcher Form vorgenommene Umdeutung der Absichten und Vorschläge der Sachverständigen ablehnen werde. Es ist daher nicht zu verstehen, daß die französische Presse sich über den Inhalt der deutschen Note in einer Weise äußert, die geeignet ist, die Stimmung der Öffentlichkeit in Frankreich und Belgien gegen Deutschland aufzupeitschen.

In dieser Meldung wird uns von unterrichteter Seite folgendes mitgeteilt:

Die vorgestrige Veröffentlichung der deutschen Denkschrift, die bei den Besprechungen in Paris und Brüssel getroffenen Vereinbarung zuwiderläuft, wird auch deutscherseits bedauert, zumal sie in einer Zeit der französischen Wahlen vorgenommen worden ist, die kaum geeignet sind, hier in Ruhe sachlicher Atmosphäre zu führen. Die deutschen amtlichen Feststellungen sind bis jetzt einem amtlichen Bericht dieser Sachverständigenkonferenz nicht hat erfolgt, daß man in Paris die in Berlin in solcher Form vorgenommene Umdeutung der Absichten und Vorschläge der Sachverständigen ablehnen werde. Es ist daher nicht zu verstehen, daß die französische Presse sich über den Inhalt der deutschen Note in einer Weise äußert, die geeignet ist, die Stimmung der Öffentlichkeit in Frankreich und Belgien gegen Deutschland aufzupeitschen.

Die Revolution in Mexiko.

Drei Jahre erlebte sich Mexiko einer seit 1911, als der Rätselhafte Porfirio Diaz zurücktrat, nicht mehr gekannten Ruhe im Innern. Die Regierung des gewählten Präsidenten Obregon hatte die Macht fest in der Hand, und trotz einer nicht zu leugnenden schwierigen wirtschaftlichen Lage, die aber in der Hauptsache durch den Mangel genügender Verkehrsmittel, fehlende Bergwerke, fehlende Industrien und einen Ausgleich der Kräften verursacht wurde, hatte sich das Land wieder einigermaßen erholt. Jener Mann, der große politische Gegensätze und den politischen Mist seines in Mai 1920 ermordeten Vorgängers Carranza, der bei der Staatsform niederwarf und einen Ausgleich der Kräften herbeiführte, verstand es, aber es gab doch noch genug Unzufriedene und auch eine genügende Energie in seiner Partei hat, und er konnte als ganzer Erfolg der Jahren die Ruhe im Innern sichern. Die Regierung mit den Vereinigten Staaten, Vorher Bergen er herrlichen Besitzen, beim Carranza zum Opfer fiel, schien sich mit der Zeit zu bessern. Die Aufhebung der Verhaftung ausgeschriebenen ist, einen amerikanischen Bergwerksbesitzer, der sich beim Obregon-Regime beschwerte, hat er persönlich empfangen und ihm Genugtuung gegeben. Die Ruhe schien wieder hergestellt zu sein, und es schien, als ob Mexiko wieder in die Reihe der geordneten Staaten eintreten würde. Da kommt plötzlich eine Nachricht, die wie ein Donnerschlag wirkt: Obregon ist ermordet worden! Und nicht einmal ein politischer Gegner hat ihn getötet, sondern ein Mann, der ihm persönlich nahestand, der sein Freund war, der ihm seine Wahl zum Präsidenten verdankte, der sein Vertrauen besaß, hat ihn ermordet. Und nicht nur ihn, sondern auch den Präsidenten Calles, der ihn zum Präsidenten gemacht hatte, der sein Freund war, der ihm seine Wahl zum Präsidenten verdankte, der sein Vertrauen besaß, hat ihn ermordet. Und nicht nur ihn, sondern auch den Präsidenten Calles, der ihn zum Präsidenten gemacht hatte, der sein Freund war, der ihm seine Wahl zum Präsidenten verdankte, der sein Vertrauen besaß, hat ihn ermordet.

Zu neuen Ufern lockt ein neuer Tag.

Atempause. — Der Kampf um das Rheingold. — Trostzeit!

© **Köln**, 31. Dezember.

Die Silvesternachtstunde, die auch in diesen Jahre wie immer viel Menschengedanken an heiligen Recht der Erbenfinder: der Ausdruck der Hoffnung auf eine bessere Zukunft ist, wird in Deutschland diesmal begleitet durch die politische Entspannung der letzten Wochen. Daher wird sich vorgestrichen in einer Atempause, deren gute Wirkung auf die Nerven uns die Möglichkeit bot, in den letzten schweren Jahren oft gekosteten Geschäften beruhigt, daß bei außerordentlich höheren Zahlenden die Ordnung im Innern Deutschlands ist von jetzt wieder wurde. Nicht daß wir diese uns so heimgeld nötige außerordentliche Sicherheit nun als selbstverständlich hinnehmen. Aber der Kampf um Ruhe und Ruhe, der zurzeit das Hauptproblem der innerpolitischen Lage ist, hat durch die deutsch-französischen Besprechungen im besonderen, ja, selbst im Hauptziel der Ruhe des Siedlungsgebiets noch einen gewissen Großzügigkeit. Jahren und drüben werden Fühler ausgestreckt, ob man das Bedürfnis nach Verhandlungen und Aussprachen irgendwo fördern; hüben und drüben gibt die Hoffnung, daß man sich jetzt auch so friedlich auf entdecken werden. Der Sinn des Versammelnens ist das Hauptkennzeichen dieser Tage: über die gegenwärtige Hast ist ein Ruhebedürfnis gegen Jahrhunderte, dessen unsichtbare Gebärde unter Umständen mehr bringen bringt als der Ausdruck der nackten Tatsache und der Zahl. Die Nerven ruhen, die in gespannt wurden ist, um in Deutschland zu Ruhe und Ruhe, der zurzeit das Hauptproblem der innerpolitischen Lage ist, hat durch die deutsch-französischen Besprechungen im besonderen, ja, selbst im Hauptziel der Ruhe des Siedlungsgebiets noch einen gewissen Großzügigkeit. Jahren und drüben werden Fühler ausgestreckt, ob man das Bedürfnis nach Verhandlungen und Aussprachen irgendwo fördern; hüben und drüben gibt die Hoffnung, daß man sich jetzt auch so friedlich auf entdecken werden. Der Sinn des Versammelnens ist das Hauptkennzeichen dieser Tage: über die gegenwärtige Hast ist ein Ruhebedürfnis gegen Jahrhunderte, dessen unsichtbare Gebärde unter Umständen mehr bringe bringt als der Ausdruck der nackten Tatsache und der Zahl. Die Nerven ruhen, die in gespannt wurden ist, um in Deutschland zu

Ein ganz Ähnliches unter Einbeziehung Rußlands zu einem schwierigen Kontinentalproblem Frankreichs zu machen; zu einer Weltfrage, die nach genseitiger und politischer Art wirken, ein Ende der alle Gegensätze Kenia war, die nun in der Kampfatmosphäre der Geschäfte wurden. Um einen gestürzten Weltmarkt als Ausgleich reiche heute der Herr und die Spionage sozialistischen, die Engländer und Amerikaner mit Belgien gemacht, ist heute unentwirrbar Engländer Willen belagte zum anderen Mal erreicht; die Entscheidung, die bei Deutschland in letzten Weltschwester kann. Die Unklugheit seiner Staatsmänner auf England Wunsch und Willen hat gelehrt helfen müßte, wurde bis langsam zum Instrument. Heute schon ist Deutschland in seinen europäischen Beziehungen wieder selbständig. Die großen und doch kleinen Bewegungen, wenn man bedenkt, daß Deutschland heute nicht Schach, sondern nur Objekt der Politik ist und die Lösungen der englischen Konferenzen das großzügigste Weltwerk in seinem europäischen Interesse nicht vorübergehend hat. Das Reich, das so ganz Mitte Deutschlandes ist, um den Krieg gegen den deutschen Wiederaufbau und dem Wettwerk zu gewinnen, in Wahrheit den Frieden Deutschlands als Weltfrage gewandelt hat, braucht nicht überschätzt heute die deutsche nun einsehen.

Die Chronik der Ereignisse zeigt noch mehr: wenn man den Weg betrachtet, den Deutschland im Jahre 1923 zum Gegenteil zurückgelegt hat, so ist es ein Weg, der Deutschland in seinen Beziehungen zu den maßgebenden Mächten in Europa und Amerika wieder in die Mitte der Ereignisse gestellt hat. Die Politik, die Deutschland heute nicht Schach, sondern nur Objekt der Politik ist und die Lösungen der englischen Konferenzen das großzügigste Weltwerk in seinem europäischen Interesse nicht vorübergehend hat. Das Reich, das so ganz Mitte Deutschlandes ist, um den Krieg gegen den deutschen Wiederaufbau und dem Wettwerk zu gewinnen, in Wahrheit den Frieden Deutschlands als Weltfrage gewandelt hat, braucht nicht überschätzt heute die deutsche nun einsehen.

Die Chronik der Ereignisse zeigt noch mehr: wenn man den Weg betrachtet, den Deutschland im Jahre 1923 zum Gegenteil zurückgelegt hat, so ist es ein Weg, der Deutschland in seinen Beziehungen zu den maßgebenden Mächten in Europa und Amerika wieder in die Mitte der Ereignisse gestellt hat. Die Politik, die Deutschland heute nicht Schach, sondern nur Objekt der Politik ist und die Lösungen der englischen Konferenzen das großzügigste Weltwerk in seinem europäischen Interesse nicht vorübergehend hat. Das Reich, das so ganz Mitte Deutschlandes ist, um den Krieg gegen den deutschen Wiederaufbau und dem Wettwerk zu gewinnen, in Wahrheit den Frieden Deutschlands als Weltfrage gewandelt hat, braucht nicht überschätzt heute die deutsche nun einsehen.

Weltes Antlitz hat diese letzte Tagesseite, der man in Deutschland heute über den Wert der Vergangenheit bis zum Gestade der Ereignisse trennt? Im Parlament der europäischen Völker ist es doch vorgestern im Jahre des Unheils 1923. Im Parlamentssaal sitzt und damit die Führung der Regierungsgeschäfte Deutschlands bei den Händen Frankreichs und seiner Geistesgehilfen gegeben; man weg hat sich Deutschland nicht gewandt, man kennt nicht Deutschland noch den Weg des zerfallenen Jahres kennen gelernt. Der 11. Januar war der „Marsch“ Tage Ruhrgas, der mit dem Marsch Frankreichs in das Industriegebiet Deutschlands die englische Führung und das Ministerium der Hoffnung, die nicht verbrechen werden ist — zu neuen Ufern lockt ein neuer Tag! Und wie ist es gekommen, obligt und rafft hier, man spürt den Frühjahr und den Weg und Ruhe, Ruhe, Ruhe,

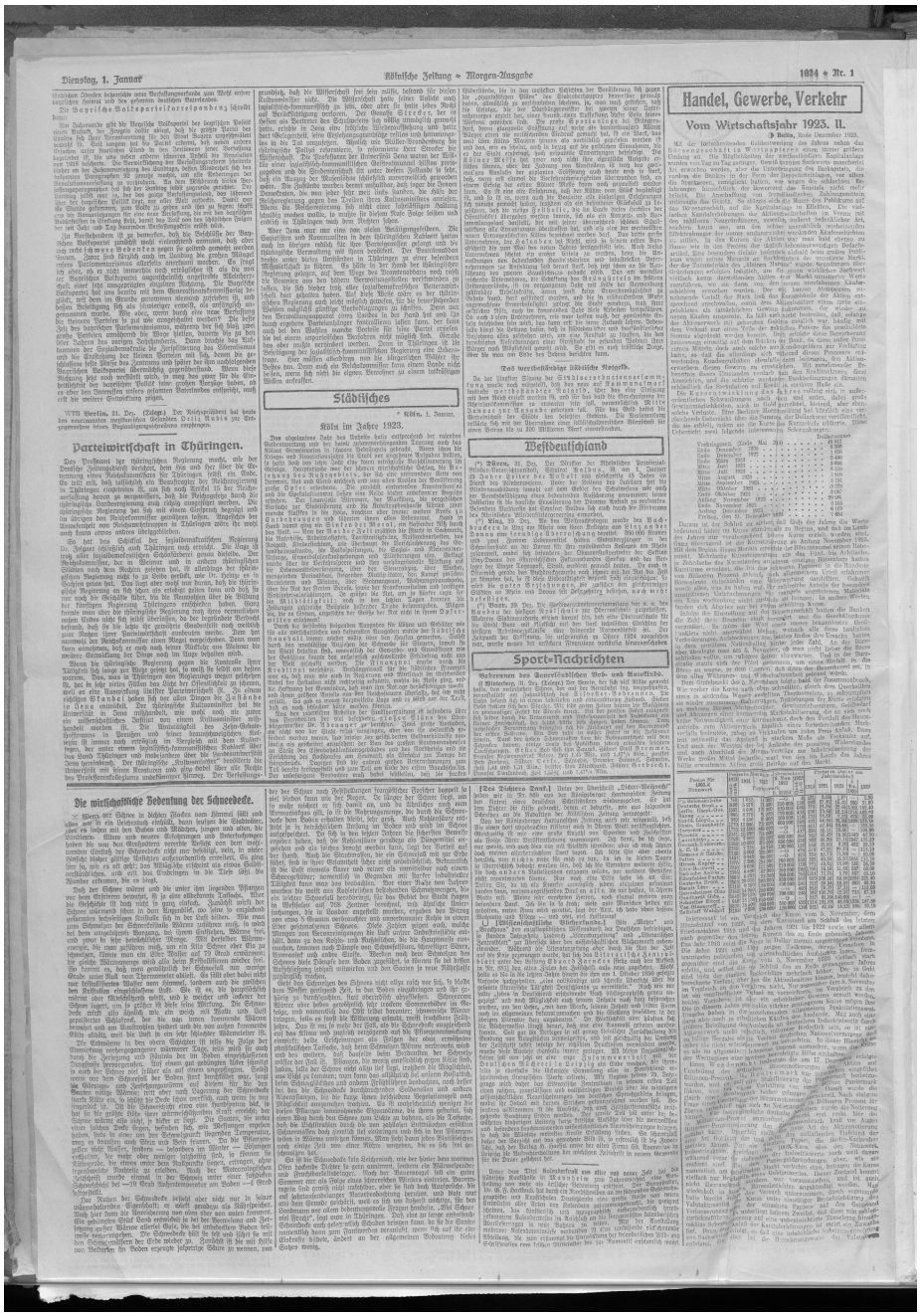


Abbildung A.14: 1924 Seite 1

A.8 Ergebnis Slice Export

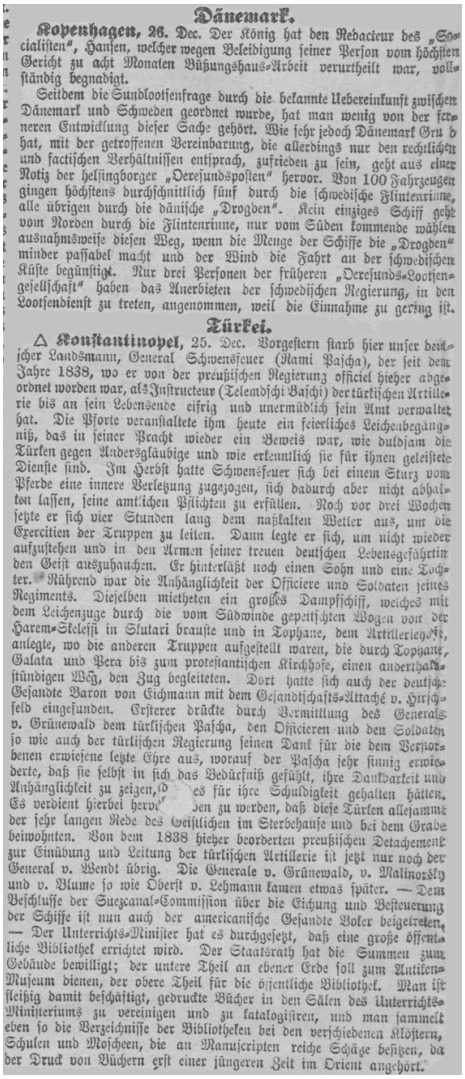
Im Folgenden sind Die Ergebnisse des Slice Export dargestellt. Bildausschnitte sind mit dem dazugehörendem aus der OCR stammenden Text dargestellt. Danach folgt die dazugehörende originale Seite. Das erste Beispiel ist ein gutes Ergebnis. Das 2. Beispiel umfasst mehr als eine Spalte, weswegen die OCR nicht zufriedenstellend funktioniert. Das liegt daran, dass es sich um eine Seite mit Anzeigen handelt, die nicht in den Trainingsdaten enthalten sind.

Dänemark.

Kopenhagen, 28. Der. Der Ftönig hat den Redacteur des „So- cialisten“, Hansen, welcher wegen Beleidigung seiner Perfon vom höchsten Gericht zu acht Monalen Bueßungshaus-Arbeit verurtheilt war, vollständig begnadigt. .

Seiidem die Sundlootsenfrage durch die bekannte Uebereinkunft zwischen Dänematk und Schweden geordnet wurde, hat man wenig von der ser- neren Entwicklung dieser Sache gehört. Wie sehr jedoch Dänemart Gru 5 hat, mit der getroffenen Vereinbarung, die allerdings nur den rechtlichen und factischen Verhältnissen entsprach, zufrieden zu sein, geht aus einer Notiz der helsingborger „Oeresundsposten“ hervor. Von 100 Fahrzeugen gingen höchstens durchschnittlich fünf durch die schwedische Flintenrinte, alle übrigen durch die däuische „Drogden“. Kein einziges Schiff gehe vom Norden durch die Flintenrinne, nur vom Süden kommende wäählen ausnahmsweise diesen Weg, wenn die Meuge der Schiffe die „Drogden- minder passabel macht und der Wind die Fahrt au der sihwedischen s begünstigt. Nur drei Personen der früheren „Oeresunds-Loctsen- d geellschast“ haben das Auerbieten der schwedischen Regierung, in den — Lootsendienst zu treten, angenommen, weil die Einnahme zu gering ist. — Türkei.

A Konstantinopel, 25. Dec. Vorgestern starb hier unser deit- her Landsmann, General Schwensseuer (ami Pascha), der jeit dem Jahre 1838, wo er von der preußischen Regierung officiel hieher abge- ordnet worden



Dänemark.

Kopenhagen, 26. Dec. Der König hat den Redacteur des „Socialisten“, Hansen, welcher wegen Beleidigung seiner Person vom höchsten Gericht zu acht Monaten Büßungshaus-Arbeit verurtheilt war, vollständig begnadigt.

Seitdem die Sundlootsenfrage durch die bekannte Uebereinkunft zwischen Dänemark und Schweden geordnet wurde, hat man wenig von der ferneren Entwicklung dieser Sache gehört. Wie sehr jedoch Dänemark Gru 5 hat, mit der getroffenen Vereinbarung, die allerdings nur den rechtlichen und factischen Verhältnissen entsprach, zufrieden zu sein, geht aus einer Notiz der helsingborger „Oeresundsposten“ hervor. Von 100 Fahrzeugen gingen höchstens durchschnittlich fünf durch die schwedische Flintenrinne, alle übrigen durch die dänische „Drogden“. Kein einziges Schiff geht vom Norden durch die Flintenrinne, nur vom Süden kommende wählen ausnahmsweise diesen Weg, wenn die Menge der Schiffe die „Drogden“ minder passabel macht und der Wind die Fahrt an der schwedischen Küste begünstigt. Nur drei Personen der früheren „Oeresunds-Lootsengesellschaft“ haben das Anerbieten der schwedischen Regierung, in den Lootsendienst zu treten, angenommen, weil die Einnahme zu gering ist.

Türkei.

△ **Konstantinopel**, 25. Dec. Vorgestern starb hier unser deutscher Landsmann, General Schwensfeuer (Rami Pascha), der seit dem Jahre 1838, wo er von der preußischen Regierung officiel hieher abgeordnet worden war, als Instructeur (Telemdschi Baschi) der türkischen Artillerie bis an sein Lebensende eifrig und unermüdlich sein Amt verwaltet hat. Die Pforte veranstaltete ihm heute ein feierliches Leichenbegängniß, das in seiner Pracht wieder ein Beweis war, wie duldsam die Türken gegen Andersgläubige und wie erkenntlich sie für ihnen geleistete Dienste sind. Im Herbst hatte Schwensfeuer sich bei einem Sturz vom Pferde eine innere Verletzung zugezogen, sich dadurch aber nicht abhalten lassen, seine amtlichen Pflichten zu erfüllen. Noch vor drei Wochen setzte er sich vier Stunden lang dem naßkalten Wetter aus, um die Exercitien der Truppen zu leiten. Dann legte er sich, um nicht wieder aufzustehen und in den Armen seiner treuen deutschen Lebensgefährtin den Geist auszuhauchen. Er hinterläßt noch einen Sohn und eine Tochter. Rührend war die Anhänglichkeit der Officiere und Soldaten seines Regiments. Dieselben miethelen ein großes Dampfschiff, welches mit dem Leichenzuge durch die vom Südwinde gepeitschten Wogen von der Harem-Skelessi in Stutari brauste und in Topkane, dem Artilleriehafen, anlegte, wo die anderen Truppen aufgestellt waren, die durch Topkane, Galata und Pera bis zum protestantischen Kirchhofe, einen anderthalbstündigen Weg, den Zug begleiteten. Dort hatte sich auch der deutsche Gesandte Baron von Eichmann mit dem Gesandtschafts-Attaché v. Hirschfeld eingefunden. Ersterer drückte durch Vermittlung des Generals v. Grünewald dem türkischen Pascha, den Officieren und den Soldaten so wie auch der türkischen Regierung seinen Dank für die dem Verstorbenen erwiesene letzte Ehre aus, worauf der Pascha sehr sinnig erwiederte, daß sie selbst in sich das Bedürfniß gefühlt, ihre Dankbarkeit und Anhänglichkeit zu zeigen, da es für ihre Schuldigkeit gehalten hätten. Es verdient hierbei hervorgehoben zu werden, daß diese Türken allesammt der sehr langen Rede des Geistlichen im Sterbehause und bei dem Grabe beiwohnten. Von dem 1838 hieher beorderten preußischen Detachement zur Einübung und Leitung der türkischen Artillerie ist jetzt nur noch der General v. Wendt übrig. Die Generale v. Grünewald, v. Malinorsky und v. Blume so wie Oberst v. Lehmann kamen etwas später. — Dem Beschlusse der Suezcanal-Commission über die Eichung und Besteuerung der Schiffe ist nun auch der americanische Gesandte Boker beigetreten. — Der Unterrichts-Minister hat es durchgesetzt, daß eine große öffentliche Bibliothek errichtet wird. Der Staatsrath hat die Summen zum Gebäude bewilligt; der untere Theil an ebener Erde soll zum Antiken-Museum dienen, der obere Theil für die öffentliche Bibliothek. Man ist fleißig damit beschäftigt, gedruckte Bücher in den Sälen des Unterrichts-Ministeriums zu vereinigen und zu katalogisiren, und man sammelt eben so die Verzeichnisse der Bibliotheken bei den verschiedenen Klöstern, Schulen und Moscheen, die an Manuscripten reiche Schätze besitzen, da der Druck von Büchern erst einer jüngeren Zeit im Orient angehört.

Abbildung A.15: Erstes Slice Export 1

war, als Instructeur (Telemdschi Baschi) der türtischen Artille- rie bis an sein Lebensende eifrig und unermüdlich sein Amt verwaltet hat. Die Pforte veranltaltete ihm heute ein feierliches Leichenbegaug- niß, das in seiner Pracht wieder ein Beweis war, wie duldsam die Türken gegen Andersgläubige und wie erkenntlich sie für ihnen geleistete Dienste sind. Im Herbst hatte Schwensfeuer sich bei einem Sturz vom Pferde eine innere Verlezung zugszogen, sich dadurch aber nicht abhal- tenn lassen, seine amt;ichen Pllichten zu erfüllen. Noch vor drei Wochen setztte er sich vier Stunden lang dem naßkalten Wetter aus, um die Erercitien der Truppen zu leiten. Dann legte er sich, um nicht wieder aufzustehen und in den Armen seiner treuen deutschen Lebensgefähetin den Geist auszuhauchen. Er hinterläßt noch einen Sohn und eine Toch- ter. Rührend war die Anhaänglichteit der Ofüciere und Soldaten seines Regiments. Dieselben mietheteu ein grohes Dainpkschiff, weiches mit dem Leichenzuge durch die vom Südwinde gepeitschten Wogen von der Harem-Skelessi in Stutati brauste und in Tophane, dem Artällerichofi, anlegte, wo die anderen Truppen aufgestellt waren, die durch Tophane, Galata und Pera bis zum protestantijchen Kirchhofe, einen auderthaki- stündigen Wen, den Zug begleiteten. Dort hatte sich auch der deutsch- Gesandte Baron von Eichntann mit dem Gesandtschafts-Attaché v. Hirsch- feld eingekunden. Ersterer drückte durch Vermitttung des Generals v. Grünewald dem türlischen Pascha, den Officteren und den Soldaten so wie auch der türtischen Regierung seinen Dank für die dem Verstor- benen erwiesene letzte Ehre aus, worauf der Vascha sehr sinnig erwie- derte, baß sie selbst in sich das Bedücfniß Elh, ihre Dankvarteit und Anhänglichkeit zu zeigen, d. es für ihre Schuldigieit gehalten hätten. Es verdient hierbei herve! zen zu werden, daß diese Tüerken allesammt der sehr laugen Rede des Geistlichen im Sterbehause und hei dem Grabe beiwohnten. Von dem 1838 hieher beorderten preußischen Detachemenk zur Einübung und Leitung der türlischen Artillerie ist jetzt nur noch der General v. Wendt übrig. Die Generale v. Grünewald, v. Malinorstz und v. Blume so wie Oberst v. Lehmann kamen etwas später. — Dem Beschlusse der Suezcanal-Commission über die Eichung und Besteuerung der Schiffe ist nun auch der americanische Gesandte Voler beigetr.

— Der Unterrichts-Minister hat es durchgesetzt, daß eine große öffent- liche Bibliothel errichtet wird. Der Staatsrath hat die Summen zum Gebäude bewilligt: der untere Theil an ebener Erde soll zum Antilen- Museum dienen, der obere Theil für die öefjentliche Bibliothek. Man ist fleibig damit beschäftigt, gedruckte Bücher in den Sälen des Unterrichts- Ministeriums zu vereinigen und zu latalogistren, und man sammelt eben so die Verzeichnisse der Bibliotheken bei den verschiedenen Klöftern, Schulen und Moscheen, die an Manuscripten reiche Schäge besizen, da der Druck von Büchern erst einer jüngeren Zeit im Orient angeho

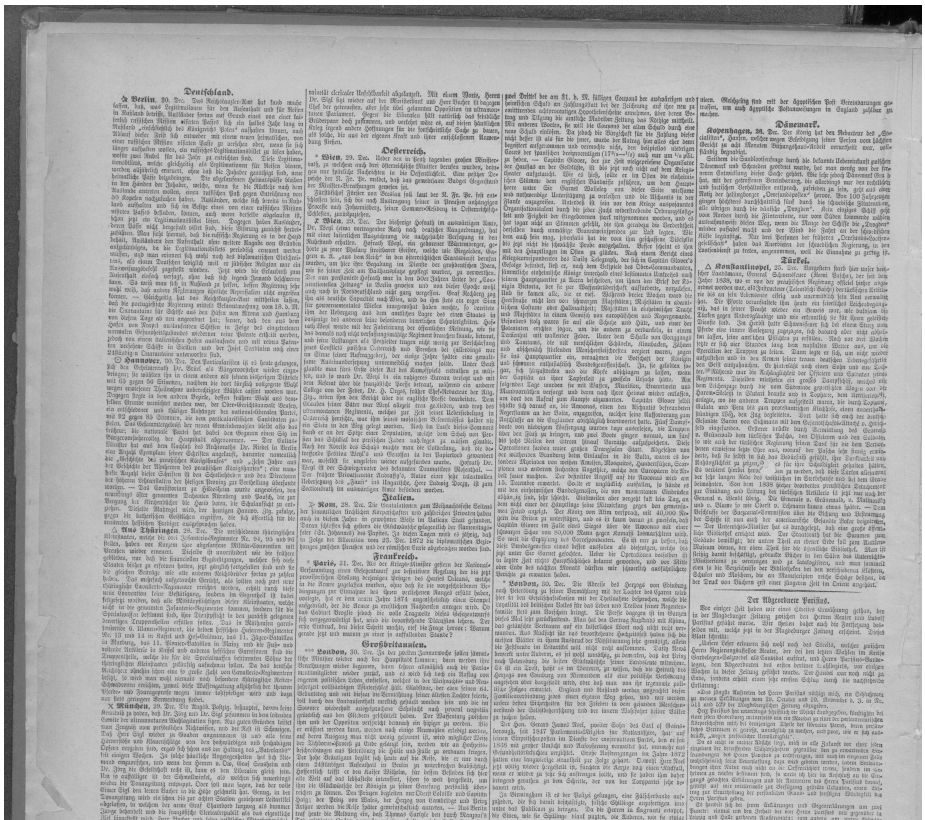


Abbildung A.16: Original Seite Slice Export 1

Sichere Hülfe
fü allet nn
Schwäche-Zuständen,
amnentlich burch Sthwäche und Erkrankung ves Nervensystems, der weiteren hieraus entstehenden Leiben, wie Nbspannung, Mattigkeit, Erschlastung, Nerven und Gedächtnißschwäche, Rücenschmerzen, Blutan: drang nach Kopf und Herz, Verdauungsbefchwerden, Lähmungen bietet in vollkommen reeller Weise mur das Buch:
Dr. Retau's Selbstbewahrung,.-
Mit 23 Abbildungen. Preis « Thlr.
so wie
2 fü. db. W.
Berlag von G+. Poenicke's Scmulbuchhandlung in Leipzig und
dort, fo wie vurch jede Buchhandiung, in Köln bei I« Kreuter, . Höhie 17, zu bekommen, .
5 Dieses Werk zeichnet sich vor anderen dadurch vortheil:
hait aus, daß es nicht theure Geheimmittel empfichlt,
joudertn ein in allet FJ;
ällen erprobtes,
billiges Heilverfahren
namhaft macht. C5 wurde defihatb anch sribst von Regierungen und Wohlfahrt8:Behörden empfohlen. Visheriger Abiaß: 24 Auflagen
oder 225,909 Exemplare.
Avis für Fabricauten
und Kaufleute,
Gute Rimessen werden billigst offe- eiet unter strengster Discretion,
Franco-Osferten ünter 24 Y 12 bes forat die Erved d. Bl.
Frische Fische. Ein botländisches Cngros-Geschäft uptjüchlich in Schellfiichen,
ngen, sucht in Köln ter, un gegen
im
Malzdarren von auerkannt zwomäßigster u. dauer Betester Constkuetionliesert billigst die
htwgarens Fabrik von . C. A. Kruse, Dnisbura a, Nh. Zur Notiz für
Bruchleideude.
Nachdom ich lange Zahre für mein AI der Nähe umb
schweres Bruchleiden in ve Ül
KE DOCH Ze
neu und comfortable eingerichtet, wird den Herren Geschaeftaereijenden beion ders entpfohlen, -
Oymnätns au Baknvoi.
Original-Loofe

Abbildung A.17: Erstes Slice Export 2

M 1. Llasse
inb noch zu beziehen durch ÄK Molling
in Braunschweig.
YYmeren viöuereranten oze wer: theilung, dasz fofort mit
Eisfahren
begonnen werden kann. Rheinische Brayerei- Geseilj¡ast Alteburg.*
Adolph Rick, Köln, Brüvkenstrasee 2, empfichlt
Geschästsbücher
2-0 Fabrikproiagm Yhilständig asSortirtes Lager. Vailbertrefsllehe Purpurin-Schreib und Copir-Dinte, Copirpreszen, Papiere, Cou- verts, Stahtsedern, Fatzmappen und Hestmaschinon zN0- tizbücuer etc.

Abbildung A.18: Original Seite Slice Export 2

A.9 XML Beispiel

```
<TextRegion id="region_1667474143077_318" custom="readingOrder {index:0;} structure {type:article_;}">
        <Coords points="3498,5612 4581,5626 4594,6971 3506,6955"/>
</TextRegion>
<SeparatorRegion id="Separator_1667474121191_313" custom="readingOrder {index:1;} structure {type:separator_short;}">
        <Coords points="3904,5602 4178,5605 4177,5613 3901,5612"/>
</SeparatorRegion>
<TextRegion type="heading" id="region_1667474001019_266" custom="readingOrder {index:4;} structure {type:article;}">
        <Coords points="3786,4688 4281,4702 4284,4741 3785,4736"/>
</TextRegion>
<TextRegion type="heading" id="region_1667474001019_266" custom="readingOrder {index:4;} structure {type:heading;}">
        <Coords points="3786,4688 4281,4702 4284,4741 3785,4736"/>
</TextRegion>
<TextRegion id="region_1667473763741_236" custom="readingOrder {index:10;} structure {type:table;}">
        <Coords points="3496,3312 4580,3312 4585,3496 3501,3496"/>
</TextRegion>
<TextRegion id="Separator_1667473666019_216" custom="readingOrder {index:14;} structure {type:header;}">
        <Coords points="106,821 4580,814 4579,850 107,857"/>
</TextRegion>
<SeparatorRegion id="Separator_1667473666019_216" custom="readingOrder {index:14;} structure
{type:separator_horizontal;}">
        <Coords points="106,821 4580,814 4579,850 107,857"/>
</SeparatorRegion>
```

A.10 Slurm Beispiel

```
#SBATCH --gres gpu:4
#SBATCH --ntasks-per-node=10
#SBATCH --mem=400G
#SBATCH --partition=booster
#SBATCH --job-name=final_reduced
#SBATCH --error=logs/final/job.%J.err
#SBATCH --output=logs/final/job.%J.out
#SBATCH --mail-user=s6cnscul@uni-bonn.de
#SBATCH --mail-type=ALL
#SBATCH --time=24:00:00

#ln -sf logs/job.$SLURM_JOB_ID.4294967291.err latest.err
#ln -sf logs/job.$SLURM_JOB_ID.4294967291.out latest.out

ml Stages/2023
ml Python
source ../venv/pytorch/bin/activate
cd /p/project/holistic-vid-westai/schultze1/repositories/NewspaperImageSegmentation/
PYTHONPATH=.
python src/news_seg/train.py -e 400 -n final_reduced -b 64 -d ../../data/ -g 4 -ts 1 -s 0.5 -l
final_dh_segment_1.0 -ls -w 7 -pf 6 -np 16 -nss 16 -wd 1e-05 --result-path final_dh_segment/ -lr 6e-04
--no-scale-aug --no-freeze --amp -r
```

A.11 Datensatz Kölnische Zeitung

Es folgen die Tabelle für die Verteilung der Klasse sowie der Leitfaden für die Annotationen. Der Leitfaden wurde in Zusammenarbeit mit den Hilfskräften im geschichtswissenschaftlichen Institut der Uni Bonn erstellt.

Label	Klassen	Häufigkeit
0	Background	26.1%
1	UnknownRegion	0.46%
2	caption	0.75%
3	table	1.98%
4	article	67.82%
5	heading	0.43%
6	header	1.0%
7	separator_vertical	0.84%
8	separator_short	0.16%
9	separator_horizontal	0.46%

Tabelle A.1: Tabelle der Label und ihrer Verteilung im Datensatz

Leitfaden: Kölnische Zeitung (Stand: 10.05.2023)

Schritt 1a: Textregionen (TR) setzen

Als Textregion werden markiert:

- Zeitungsüberschrift → „header“
- Elemente der Kopfzeile → „caption“
- Artikelüberschriften → „heading“
- Artikeltext (meint den Artikel ohne die Überschrift der Artikel)→ „article“
- Tabellen → „table“
- Gefallenenlisten → „article“
- Wenn der Scan krumm ist, dann kann ein Artikel durch Polygone getagt werden (genaue Erklärung weiter unten)

Textregionen werden spaltenweise und der „logischen Lesereihenfolge“ entsprechend gesetzt

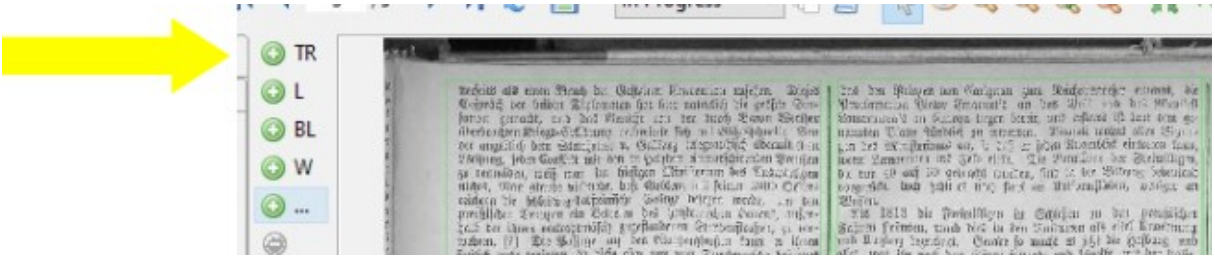
Textregionen müssen möglichst eng an die Schrift gesetzt werden

Vorgehensweise:

- Das Icon „TR“ anklicken
- Einfacher Linksklick dort hinsetzen, wo man die Textregion beginnen lassen möchte. Ein rechteckiges Feld ziehen und einen einfachen Klick dort hinsetzen, wo die Textregion beendet werden soll

Wichtig:

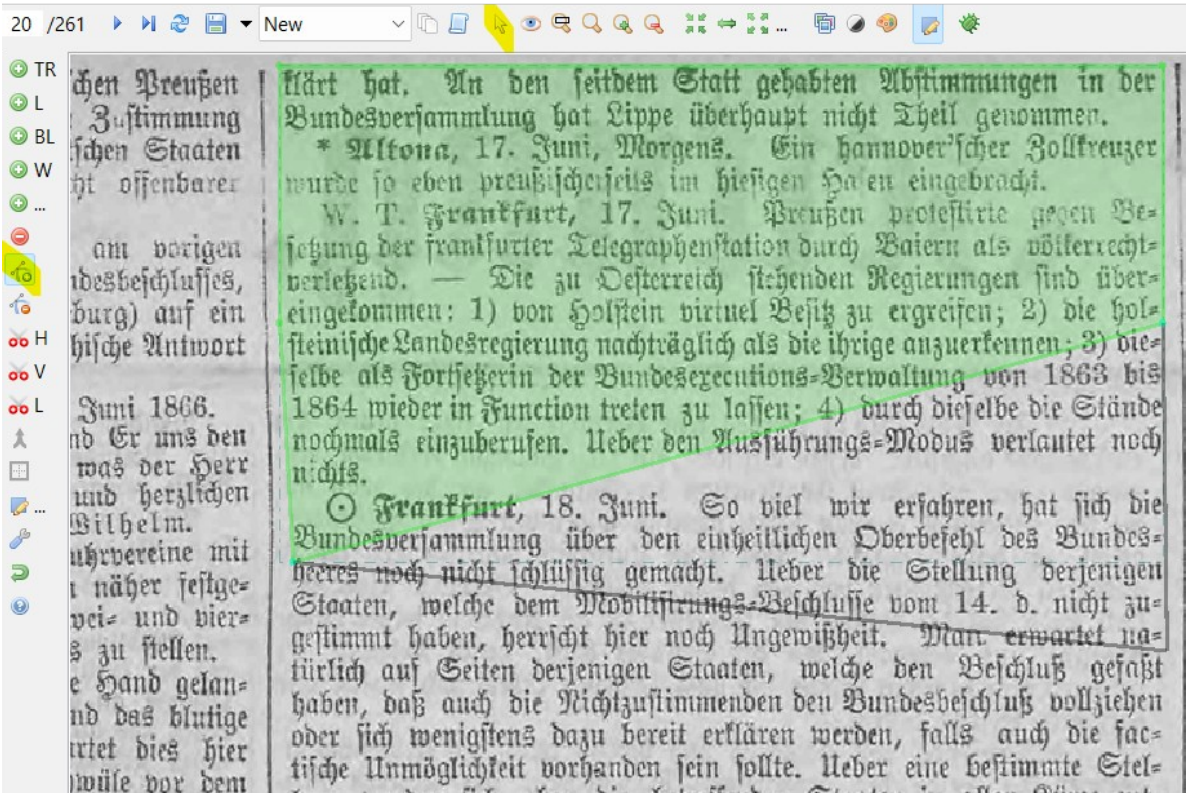
- **genaues und effizientes Arbeiten, d.h.:**
  - Textregionen sollen sich mit anderen Textregionen möglichst nicht überschneiden
  - Überschriften müssen so genau wie möglich annotiert werden
  - Separatoren müssen so genau wie möglich annotiert werden
  - Header können als einfache 4-Ecke annotiert werden
- generell gilt bei allen Arbeitsschritten: Zwischenspeichern nicht vergessen. Lieber zu viel als zu wenig, da das Programm gerne mal abstürzt



Verwendung von Polygonen, um Textregionen zu setzen

Vorgehensweise:

- Eine Textregion ziehen
- Diese dann anklicken, damit in der Toolbar auf der linken Seite folgendes Symbol anwählbar wird
- Symbol anklicken, dann kann für die zuvor ausgewählte Textregion eine Linie erstellt werden, die die Textregion besser umfasst
- Durch erneutes links klicken wird ein Ankerpunkt erstellt, von dem aus weitere Linien gezogen werden können.
- Zum Beenden der Funktion auf das Cursorsymbol auf der oberen Toolbar klicken

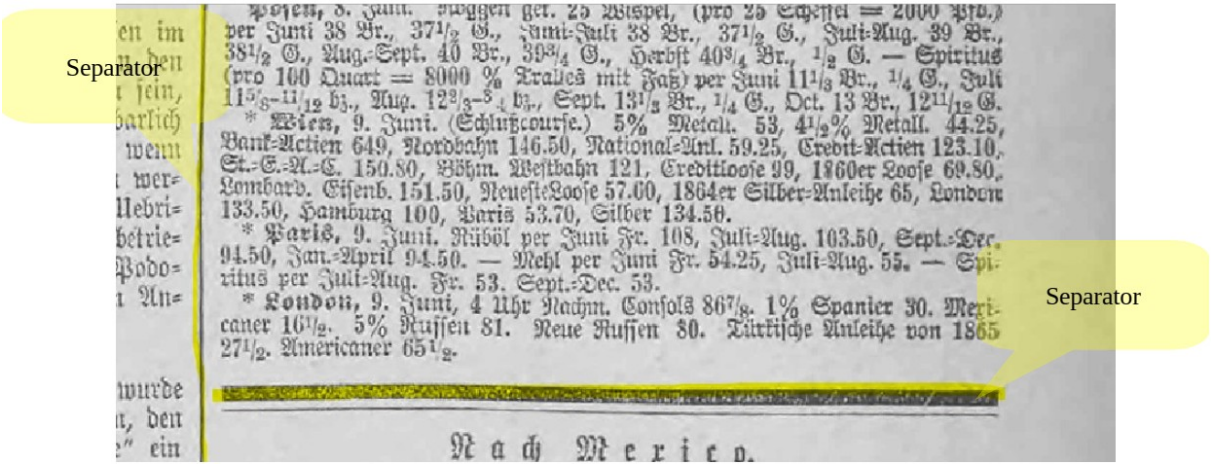


Merke:

- Für genaues Arbeiten lohnt es sich diese Funktion zu nutzen, sowohl bei dem taggen der Textregionen als auch bei Separatoren

Schritt 1b: Separatoren setzen

Separatoren sind alle optischen „Streifen“ bzw. „Striche“, die die Artikel voneinander trennen. Dazu zählen auch die Streifen, die die einzelnen Spalten voneinander trennen



Vorgehensweise:

- Das Icon „+…“ anklicken
- „Separator“ auswählen
- Wie beim Setzen von Textregionen verfahren, d.h.:
- Einfacher Linksklick dort hinsetzen, wo man die Textregion beginnen lassen möchte. Ein rechteckiges Feld ziehen und einen einfachen Klick dort hinsetzen, wo die Textregion beendet werden soll

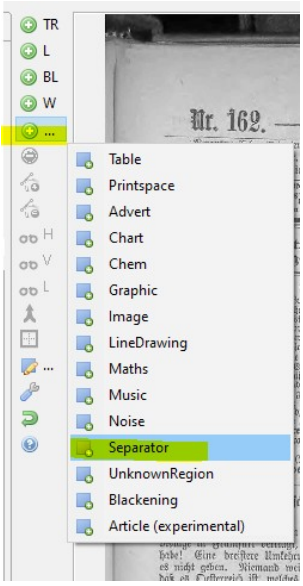
Wichtig:

- genaues Arbeiten
- Die Textregionen sollen sich mit anderen Textregionen nicht überschneiden. Das ist wichtig damit die Baselines vom Programm gut erkannt werden

Separatoren Tags

Horizontale, kurze Separatoren innerhalb einer Spalte	separator_short
Separatoren, die Spalten voneinander trennen	separator_vertical
horizontale Separatoren über die gesamte Seite	separator_horizontal

Vgl. Video: Separatoren



Schritt 2: Baselines erkennen lassen

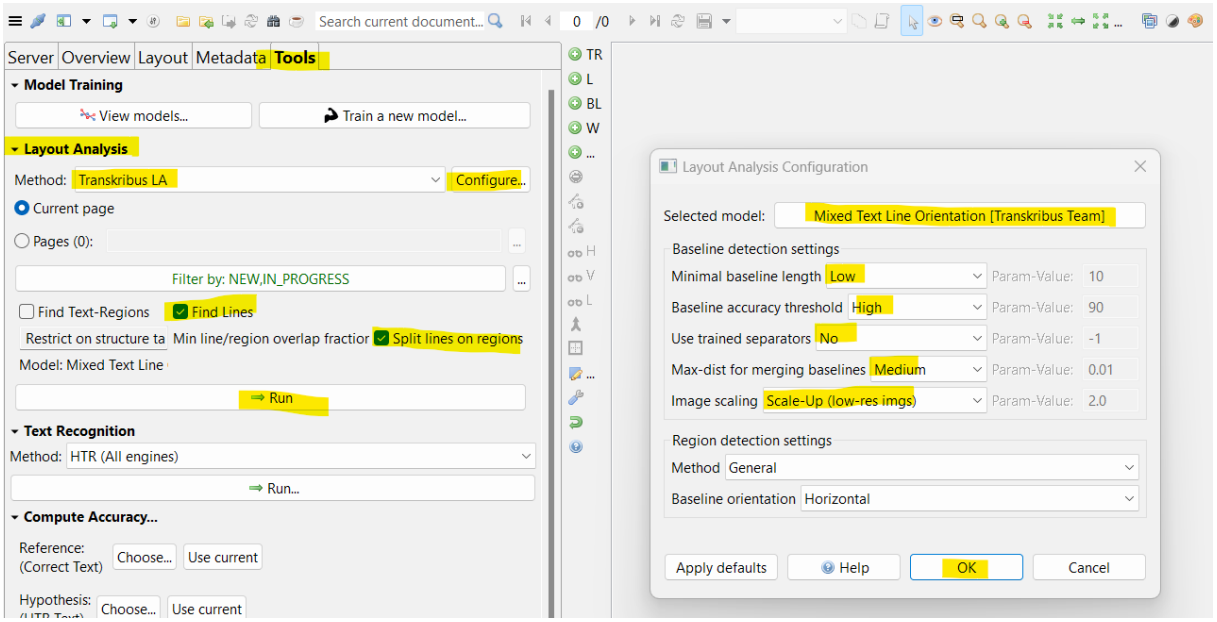
Wichtig:

- Beim aktuellen Stand (24.05.23) müssen vorab alle Textregionen markiert werden, bei denen der Text erkannt werden soll
- Auf „Tools“ klicken
- Unter „Layout Analysis“ bei „Method“ folgendes Einstellen:
  - Transkribus LA
- auf „Configuration“ klicken und folgendes Einstellen:

Try using the following settings:

Layout Model	Mixed Text Line Orientation
(click on Configure)	
Generation of Text Regions (Layout Blocks)	Keep existing
Image Scaling	Upscale
(click on Baseline Options)	
Minimal Baseline Lengh	Low
Baseline Accuracy Threshold	High
Use Trained Separators	No
Max distance for merging baselines	Medium
Split Lines on Regions border	Yes

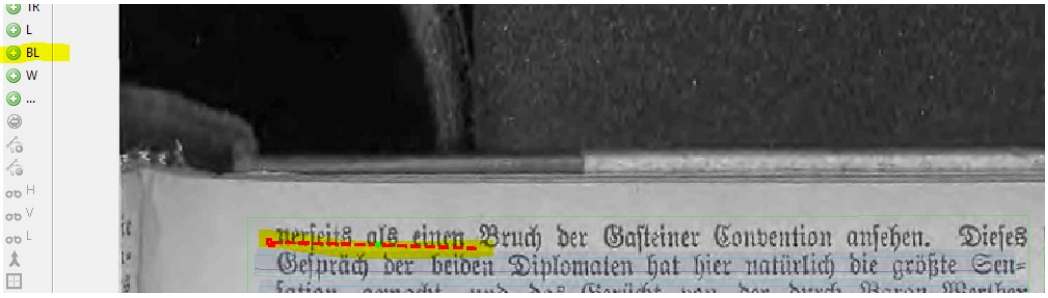
- „Current page“ auswählen
- Darauf achten, dass „Find Text Regions“ **nicht** ausgewählt ist
- Darauf achten, dass „Find lines“ und „split lines on regions“ ausgewählt sind
- auf das obere „Run“ klicken
- ein neuer „Job“ entsteht, der unter dem Icon „Kaffeetasse“ beobachtet werden kann
- Das Programm braucht Zeit, um die Baselines zu ziehen
- Nachdem die Baselines erkannt wurden, öffnet sich ein Fenster, welches die Frage beinhaltet, ob die Seite neu geladen werden soll, dies wird bejaht



Vgl. Video: Baselines_erkennen1

Tipp: Baselines händisch ziehen

- Icon „BL“ klicken
- Linksklick auf die gewünschte Stelle
- Linie ziehen mit „Zwischenklicks“ und „Doppelklick“ um Baseline ziehen zu lassen



Tipp: Löschen von Baselines

- Gewünschte Baseline auswählen

- Rechtsklick
- Delete wählen

Video: Baselines_erkennen2

Schritt 3: Taggen

- Tagvergabe an die verschiedenen Textregionen und Baselines

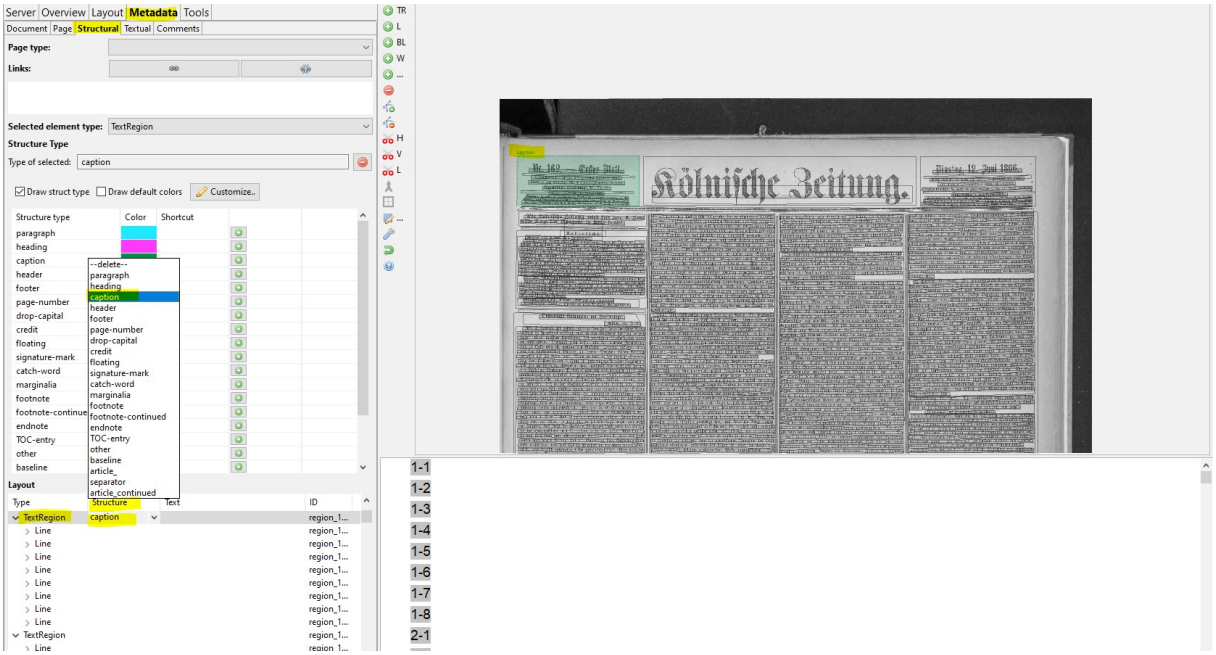
Mögliche Tags:

Zeitungsname „Kölsche Zeitung“	Header
Textblöcke links/rechts neben der Zeitungsüberschrift	Caption
Einzelne Artikel	Article
Überschrift der Artikel	Heading
Baseline	Baseline

Vorgehensweise:

- Auf den Reiter „Metadata“ gehen
- „Structural“ auswählen
- Daraufhin die gewünschte Textregion unter Layout auswählen und in „Structure“ klicken, dadurch öffnet sich ein neues Fenster
- Den gewünschten Tag auswählen
- Tipp:
    - Bei der Vergabe des Tags „Baseline“ in „Layout“ „Shift“ gedrückt halten und möglichst viele Lines markieren → spart Zeit
    - Wichtig: hierbei immer Zwischenspeichern, da bei mir das Programm sich manchmal aufhängt/abstürzt

Vgl. Video: taggen

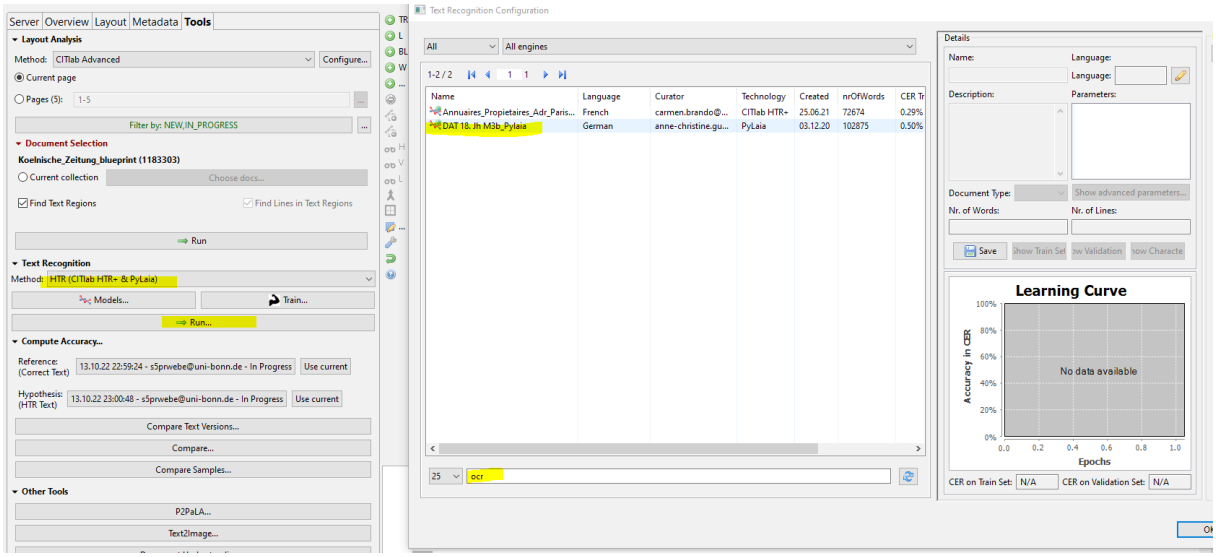


Schritt 4: Ocr-Erkennung → dieser Schritt wird erst mal ausgelassen

Vorgehensweise:

- Auf „Tools“ klicken
- „Current page“ auswählen
- Unter „Text Recognition“ bei „Method“ „HTR (CITlab HTR + & Pylaia) auswählen
- Auf „Run“ klicken
- Ein Fenster öffnet sich und „Select HTR model“ auswählen
- In die Suche „OCR“ eingeben und das Modell „DAT 18. Jh M3b_Pylaia“ auswählen
- „Ok“ auswählen

Vgl. Video: ocr_Erkennung



Zeitungsannoncen annotieren

Vorgehensweise: