technische universität
dortmund

Diplomarbeit

Generierung synthetischer Daten für das Word-Spotting-Verfahren

Oksana Bessonov
20. Dezember 2018

Überarbeitete Fassung

Supervisors:
Dipl. Inf. Leonard Rothacker
Prof. Dr.-Ing. Gernot A. Fink

Fakultät für Informatik
Technische Universität Dortmund
http://www.cs.uni-dortmund.de

INHALTSVERZEICHNIS

1 EINLEITUNG

Historische handgeschriebene Dokumente und Manuskripte beinhalten wertvolle Informationen, die insbesondere für Forschungen von Bedeutung sind. Die Dokumente werden häufig eingescannt oder abfotografiert und in einer digitalen Form als Bilder zur Verfügung gestellt. Um die Bilddaten zu verarbeiten oder nach bestimmten Informationen zu durchsuchen, müssen darin zuerst Texte erkannt und dann in eine maschinenlesbare Form umgewandelt werden.

Standardisierte OCR-Techniken, die sich mit Verfahren der automatischen Texterkennung oder der optischen Zeichenerkennung (engl. Optical Character Recognition, OCR) beschäftigen, funktionieren bei modernen maschinengeschriebenen und in guter Qualität eingescannten Textseiten weitgehend fehlerfrei. Sie stoßen jedoch bei handgeschriebenen und vor allem historischen handgeschriebenen Dokumenten und Manuskripten an ihre technischen Grenzen [GSGN17].

Im Gegensatz zu gedruckten Dokumenten zeichnen sich handschriftliche Dokumente durch eine höhere Varianz der Schreibweise nicht nur aufgrund von mehreren Schreibern, sondern auch bei den einzelnen Schreibern aus. Wortabbilder eines textuellen Wortes sind nicht einheitlich und unterscheiden sich visuell voneinander. Jede Handschrift ist individuell, denn die Buchstaben werden unterschiedlich geschrieben und miteinander verbunden. Bei einem geschriebenen Wort können sich die einzelnen Buchstaben überlappen und zusammen oder getrennt geschrieben werden. Durch eine fehlende oder inkonsistente Trennung der einzelnen Buchstaben ist es schwierig und fehleranfällig, die handgeschriebenen Wörter in Buchstaben zu segmentieren. Bei einer engen Schreibweise lassen sich auch Textzeilen und Wörter nur aufwändig voneinander trennen [GSGN17].

Die nächste Herausforderung beim automatischen Erkennungsprozess ist der Zustand der Textdokumente. Insbesondere historische handgeschriebene Dokumente und Manuskripte befinden sich in den meisten Fällen in einem sehr schlechten Zustand. Durch den Alterungsprozess sind Textseiten vergilbt und die Tinte ist verblasst. Verunreinigungen und Beschädigungen des Papiers erschweren die Erkennung zusätzlich. Des Weiteren ist die Beschriftung der Rückseiten zu erkennen [GSGN17].

Neben der höheren Schreibvarianz und der schlechten Qualität der Textseiten sind bei historischen handgeschriebenen Dokumenten und Manuskripten auch Wörter zu finden, die heutzutage nicht mehr verwendet werden. Es ist sehr aufwändig

Abbildung 1.0.1: Beispiele für die Varianz der Schreibweise von mehreren Schreibern und den schlechten Zustand von historischen handgeschriebenen Textdokumenten. Die zwei oberen Textzeilen sind historische und die zwei unteren moderne Handschriften. Die Textzeilen wurden aus [ES] [GW] [IAM] entnommen.

eine ausreichende Menge von Beispielen für das Training eines OCR-Modells zur Verfügung zu stellen [GSGN17]. Die Abbildung 1.0.1 veranschaulicht die Unterschiede zwischen historischen und modernen handgeschriebenen Textdokumenten und die Schreibvariationen von mehreren Schreibern.

Eine alternative Lösung zum Durchsuchen von handgeschriebenen und vor allem historischen handgeschriebenen Dokumenten bietet das *Word-Spotting-Verfahren* an. Es beruht auf der Idee, nach bestimmten Schlüsselwörtern in digitalisierten Textdokumenten zu suchen. Die Suche erfolgt auf der Basis der Ähnlichkeitsberechnung ohne eine explizite Erkennung eines Wortabbildes. Die Abbildung 1.0.2 veranschaulicht die Grundidee des Word-Spotting-Ansatzes. Bei einer benutzerdefinierten Suchanfrage werden visuell ähnliche Wortabbilder anhand ihrer charakteristischen visuellen Merkmale in einer Dokumentensammlung identifiziert. Als Antwort wird eine Retrieval-Liste der Wortabbilder, die anhand ihrer Ähnlichkeit zum Anfragebild sortiert ist, zurückgeliefert [GSGN17].

In den letzten Jahren erhielt das Word Spotting viel Aufmerksamkeit im Bereich der Dokumentenanalyse von historischen handgeschriebenen Dokumenten. Im Rahmen der Forschung haben sich viele Methoden entwickelt. Eine ausführliche Übersicht ist in [GSGN17] zu finden.

Dank des technischen Fortschritts sowie einer wachsenden Rechenleistung und verbesserten Algorithmen lassen sich Word-Spotting-Modelle in der letzten Zeit mit Deep-Learning-Lösungen implementieren. Faltungsnetze repräsentieren den Stand der Technik und ermöglichen eine effiziente Suche in großen Sammlungen von Bilddokumenten [SF18].

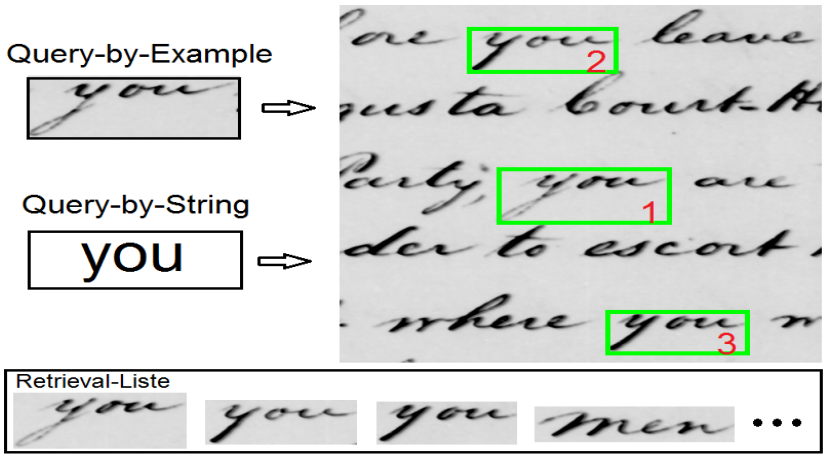

Abbildung 1.0.2: Visualisierung der Grundidee des Word-Spotting-Ansatzes. Das Bild wurde in Anlehnung an [Fin18] erstellt.

Grundsätzlich kann der Ablauf eines CNN-basierten Word-Spotting-Modells in die Bearbeitungsschritte „Segmentierung“, „Gestaltung der Suchanfrage“, „Merkmalsextraktion und Repräsentation“ und „Ähnlichkeitsberechnung“ unterteilt werden. Die allgemeinen Bearbeitungsschritte eines Word-Spotting-Modells werden in der Abbildung 1.0.3 verdeutlicht [GSGN17].

- Segmentierung

  Word-Spotting-Ansätze werden in zwei Kategorien eingeteilt: *segmentierungsbasierte* und *segmentierungsfreie* [GSGN17]. Bei einem segmentierungsbasierten Word-Spotting-Ansatz werden Dokumentabbilder in einzelne Wörter oder Textzeilen segmentiert. Die Segmentierung der handgeschriebenen und degradierten Textdokumente ist keine triviale Aufgabe. Alternativ dazu können segmentierungsfreie Word-Spotting-Verfahren, die mit vollständigen Textdokumenten arbeiten, eingesetzt werden [RSR+17].

- Gestaltung der Suchanfrage

  Die zwei verbreitetsten Möglichkeiten, um eine Suchanfrage an das Modell zu stellen, sind *Query-by-Example* und *Query-by-String*.

  Mit Query-by-Example wird eine Suchanfrage durch einen Bildausschnitt des zu suchenden Wortes beschrieben. Das gewünschte Wortabbild muss zuerst manuell gefunden werden. In größeren Dokumentensammlungen ist das eine

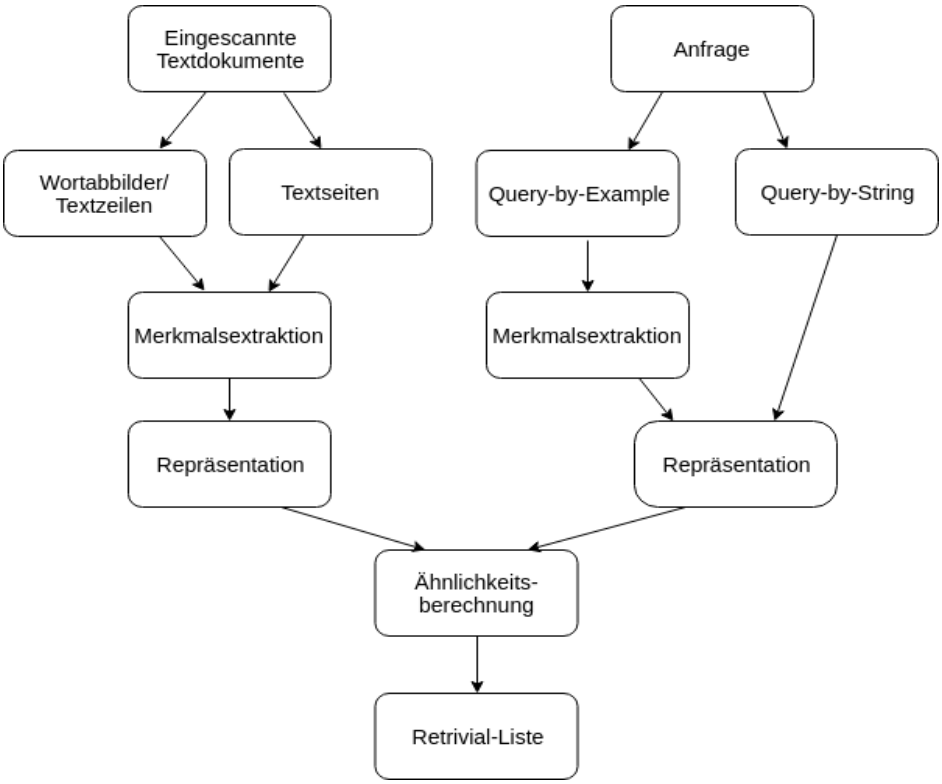

Abbildung 1.0.3: Verallgemeinerung des Ablaufs eines Word-Spotting-Modells [GSGN17].

.

sehr langwierige und zeitaufwändige Aufgabe. In einem ungünstigen Fall wird das gewünschte Wortabbild nicht gefunden [GSGN17].

Mit Query-by-String wird eine Suchanfrage in Form einer Zeichenkette direkt per Tastatur eingegeben.

Um Query-by-Example- oder Query-by-String-Anfragen durchzuführen, wird zuerst das CNN-basierte Word-Spotting-Modell trainiert. Die Abbildung von textuellen und visuellen Repräsentationen wird anhand von Trainingsbeispielen erlernt. Für das Training eines robusten und schreiberunabhängigen Word-Spotting-Modells wird eine ausreichend große Menge von Trainingsbeispielen benötigt [GSGN17].

- Merkmalsextraktion und Repräsentation

Merkmalsextraktion und Repräsentation sind erhebliche Bestandteile eines Word-Spotting-Ansatzes und eine wesentliche Voraussetzung für die Leistungsfähigkeit des Modells. Wortabbilder werden durch ihre charakteristischen Merkmale repräsentiert, die wesentliche visuelle Bildeigenschaften beschreiben. Merkmale, die aus dem gesamten Wortabbild extrahiert werden, werden als *globale* Merkmale bezeichnet. Im Gegensatz dazu werden *lokale* Merkmale aus einer lokalen Umgebung des Wortabbildes extrahiert [GSGN17]. Zudem wird zwischen *holistischen* und *sequenziellen* Merkmalsrepräsentationen unterschieden. Bei holistischen Repräsentationen werden aus dem ganzen Wortabbild extrahierte Merkmale zu einem Merkmalsvektor zusammengefasst [AGFV14]. Bei sequenziellen Merkmalsrepräsentationen handelt es sich um Sequenzen von Merkmalsvektoren, die innerhalb der in Schreibrichtung bewegten Zeitfenster berechnet werden [RRF13] [RF15].

- Ähnlichkeitsberechnung

  Die in den vorherigen Bearbeitungsschritten extrahierten Merkmale werden zur Ähnlichkeitsberechnung verwendet, indem die Merkmale der Anfrage mit den Merkmalen des Wortabbildes verglichen werden. Die Ähnlichkeitsberechnung hängt von der ausgewählten Merkmalsrepräsentation ab. Die sequenziellen Modelle werden mit *Dynamic Time Warping* [RM07], *Hidden Markov Models* [RRF13] und *Long Short Term Memory Netzen* [FFM12] implementiert. Bei holistischen Modellen wird die Ähnlichkeit der Merkmalsvektoren über das Ähnlichkeitsmaß bewertet [SF18] [RSR+17]. Die Wortabbilder werden nach ihrer Ähnlichkeit mit der Suchanfrage sortiert und als Retrieval-Liste zurückgegeben [GSGN17].

Die Gewinnung von Merkmalen sowie die anschließende Ähnlichkeitsberechnung spielen eine entscheidende Rolle für ein Word-Spotting-Verfahren. In der letzten Zeit haben sich holistische Repräsentationen der Wörter als *Pyramidal Histogram of Characters* (PHOC, vgl. 3.1) als erfolgreich erwiesen [AGFV14]. Dabei werden Wortabbilder mit Attributen kodiert, die zu einem binären d-dimensionalen PHOC-Vektor konkateniert werden. Die Dimensionalität des PHOC-Vektors wird über die Anzahl der Attribute bestimmt. Die Attribute beschreiben semantische Charakteristika eines Wortes und können als eine Antwort auf die Frage „Welche Zeichen befinden sich in einer bestimmten lokalen Region eines Wortes?“ interpretiert werden. Durch die Abbildung der Suchanfragen und Wortabbilder in einem Vektorraum wird die Berechnung der Ähnlichkeiten zweier Vektoren anschließend durch das Kosinus-Ähnlichkeitsmaß berechnet [AGFV14].

1.1 PROBLEMSTELLUNG

In dieser Diplomarbeit geht es um einen segmentierungsbasierten Word-Spotting-Ansatz, bei dem Dokumentseiten in einem Vorverarbeitungsschritt in einzelne Wortabbilder segmentiert werden. Der Ansatz wird mit einem PHOC-Netz realisiert. Das PHOC-Netz ist eine sehr erfolgreiche tiefe CNN-Architektur, die für Word-Spotting-Aufgaben entwickelt wird [SF16]. Das Training des tiefen Netzes erfordert eine ausreichend große Menge von Trainingsbeispielen. Aufgrund der großen Schreibvarianz bei Handschriften werden die Trainingsbeispiele spezifisch für jede Dokumentsammlung erstellt. Des Weiteren lassen sich Trainingsbeispiele bei manchen handgeschriebenen Textdokumenten und vor allem bei historischen handgeschriebenen Textdokumenten nur schwer und/oder in einer eingeschränkten Anzahl gewinnen [GSGN17].

Die Erstellung der manuell annotierten Wortabbilder ist eine sehr langwierige Aufgabe, die mit einem hohen Aufwand verbunden ist. Es ist wünschenswert, nur eine kleine Anzahl von Hand annotierter Wortbilder zu benötigen. Dieser Wunsch und der in [GSF17] vorgestellte Ansatz zur Reduzierung der Anzahl der nötigen manuell annotierten Beispieldaten unter Verwendung des synthetischen Datensatzes motivieren diese Diplomarbeit.

Das Ziel ist die Generierung synthetischer Trainingsdaten, um den manuellen Aufwand zu reduzieren. Zu diesem Zweck wird das in [KJ16] vorgeschlagene Verfahren zur Erstellung computergenerierter Wortabbilder mit Handschriften ähnelnden Computerschriften betrachtet. Mit dem vorgestellten Verfahren lassen sich synthetische Datensätze in unterschiedlichen Größen generieren. Unter der Annahme, dass passende Computerschriften vorhanden sind, kann ein beliebiges Alphabet verwendet werden. Es werden mehrere synthetische Datensätze für das lateinische Alphabet in verschiedenen Größen generiert und es wird untersucht, welche Anzahl an computergenerierten Wortabbildern bezüglich der Leistungsfähigkeit für die Zwecke dieser Arbeit optimal ist.

Im nächsten Schritt werden Verbesserungen der synthetischen Trainingsmenge vorgenommen, um die Unterschiede zwischen synthetischen und realen Wortabbildern zu verringern. Zu diesem Zweck werden elastisch deformierte und verrauschte computergenerierte Wortabbilder verwendet. Das Hinzufügen von elastisch deformierten [SSP03] und verrauschten [ZSC17] Beispieldaten hat einen positiven Effekt auf die Erkennungsleistung von Faltungsnetzen bei handgeschriebenen Zahlen. Außerdem werden computergenerierte Wortabbilder dadurch etwas realistischer abgebildet. Anschließend werden kleine Mengen der realen Wortabbilder elastisch deformiert und verrauscht, um eine bessere Leistungsfähigkeit zu erreichen.

Wie in [GSF17] vorgeschlagen wurde das PHOC-Netz auf der Basis von synthetischen Wortabbildern vortrainiert und an kleine Mengen manuell annotierter handgeschriebener Wortabbilder angepasst.

1.2 GLIEDERUNG DER ARBEIT

Die vorliegende Arbeit gliedert sich wie folgt: In Kapitel 2 werden zunächst die theoretischen Grundlagen, die für das Verständnis der Arbeit notwendig sind, erläutert. Hierbei werden im Abschnitt 2.1 relevante Verfahren der Bildbearbeitung und im Abschnitt 2.2 das Grundprinzip, der Aufbau und die Funktionsweise der Faltungsnetze diskutiert. Verwandte Arbeiten, die sich mit ähnlichen Problemstellungen beschäftigen und diese Arbeit inspiriert haben, werden in Kapitel 3 untersucht. In Kapitel 4 wird die Methodik zur Generierung synthetischer Daten beschrieben, auf deren Basis mehrere synthetische Datensätze erstellt werden. Das Kapitel 5 präsentiert die Ergebnisse der verschiedenen Experimente. Schließlich wird diese Arbeit in Kapitel 6 zusammengefasst.

2 GRUNDLAGEN

Das tiefe Faltungsnetz (engl. Deep Convolutional Neural Network, CNN) wird für die Implementierung des segmentierungsbasierten Word-Spotting-Ansatzes in dieser Diplomarbeit verwendet. Eine der bedeutendsten Eigenschaften des CNN ist die lineare Faltungsoperation. Um die Funktionsweise der Faltung darzustellen, wird im Abschnitt 2.1 zuerst auf die theoretischen Grundlagen der Bildverarbeitung eingegangen. Danach werden weitere Bildoperationen erläutert, die für die Generierung der synthetischen Daten und die Datenaugmentierung in dieser Arbeit von Bedeutung sind. Die Synthese der Wortabbilder und die Datenaugmentierung dienen dazu, die Anzahl der manuell annotierten realen Trainingsbeispiele und dementsprechend den manuellen Aufwand zu reduzieren. Im Abschnitt 2.2 werden der Aufbau und die Funktionsweise des tiefen Faltungsnetzes beschrieben sowie das Trainingsverfahren und das Prinzip der Feinabstimmung erläutert. Im Abschnitt 2.2.5 werden zwei Regularisierungstechniken *Dropout* und *Datenaugmentierung* kurz dargestellt.

2.1 BILDBEARBEITUNG

Ein *digitales Bild I* der Größe M x N ist definiert als eine zweidimensionale Funktion der ganzzahligen Koordinaten $\mathbb{N}$ x $\mathbb{N}$ auf ein Intervall von Bildwerten $\mathbb{P}$[BB06, S. 10]:

$$I(u,v) \in \mathbb{P} \quad \text{und} \quad u,v,\ M,\ N \in \mathbb{N} \tag{2.1.1}$$

Die Koordinaten $u \in \{0,1,...,M-1\}$, $v \in \{0,1,...,N-1\}$ bezeichnen die Spalten M bzw. die Zeilen N des Bildes. Pixelwerte $\mathbb{P}$ von Grauwertbildern liegen im Bereich [0, 1, ..., 255], wobei der minimale Wert 0 der schwarzen und der maximale Wert 255 der weißen Farbe entspricht [BB06, S. 13].

2.1.1 *Lineare Filter und Faltung*

Lineare Filter sind lokale Bildoperationen, um bestimmte charakteristische visuelle Merkmale eines Bildes zu erkennen, zu modifizieren oder zu unterdrücken. Grundsätzlich werden lineare Filter zur Erkennung von Kanten, Glättung oder Weichzeichnen und Rauschunterdrückung verwendet. Eine wesentliche Eigenschaft der lokalen Bil-

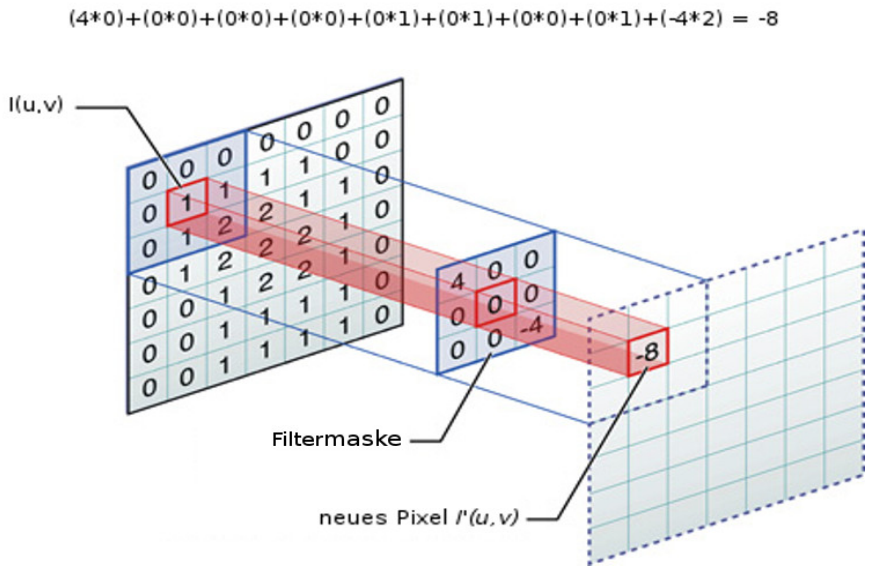

Abbildung 2.1.1: Die Filtermaske wird auf das Eingabebild I gelegt. Dabei befindet sich der Mittelpunkt der Filtermaske $K(0,0)$ an der Position (u,v). Die Koeffizienten der Filtermaske werden mit den entsprechenden Bildpixeln innerhalb der Filterregion multipliziert. Die Ergebnisse werden summiert und der neue Wert wird an der Position (u, v) des Ergebnisbildes I' gespeichert. Dann wird die Filtermaske über das Eingabebild um ein Bildpixel nach rechts verschoben und das nächste Bildpixel wird für das Ergebnisbild berechnet. Das Bild wurde aus [Hue] entnommen.

doperationen ist, dass der neue Pixelwert eines Ausgabebildes aus dem Bildpixel an derselben Position und seinen benachbarten Bildpixeln im Eingabebild berechnet wird. Dabei entsteht eine 1:1-Abbildung der Bildkoordinaten und die Größe des Ausgabebildes bleibt unverändert [BBo6, S. 90].

Die Operation eines linearen Filters entspricht einer linearen Faltung, die zwei Funktionen gleicher Dimensionalität kontinuierlich oder diskret verknüpft [BBo6, S. 101]. Die *zweidimensionale Faltung* eines Eingabebildes I mit einer 3x3-Filtermaske K wird in [BBo6, S. 93] wie folgt definiert:

$$I'(u,v) = I(u,v) * K(i,j) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} I(u+i, v+j) \cdot K(i,j) \tag{2.1.2}$$

dabei wird der Mittelpunkt der Filtermaske $K(0,0)$ über der Bildkoordinate (u, v) zentriert.

Wie das Bild selbst ist auch die Filtermaske K eine zweidimensionale Funktion und wird in Form einer Matrix gegeben. Die Matrixkoeffizienten spezifizieren die Gewichte des Filters.

Die Größe der Filtermaske ist ein bedeutsamer Parameter, denn sie legt fest, wie viele ursprüngliche Bildpixel zur Berechnung eines neuen Bildpixels beitragen. In der Praxis werden meistens quadratische Filtermasken von ungerader Größe verwendet, da der Bezugspunkt der Filtermaske im Zentrum liegt [BBo6, S. 90].

Die Filtermaske wird auf dem Eingabebild so positioniert, dass sich ihr Koordinatenursprung $K(0,0)$ auf dem aktuellen Bildpixel *I(u,v)* befindet. Die Bildpixel innerhalb dieser Filterregion werden mit entsprechenden Gewichten der Filtermaske multipliziert und die Ergebnisse werden summiert. Diese Summe wird an der entsprechenden Position (u,v) im Ergebnisbild I' gespeichert. Die Abbildung 2.1.1 veranschaulicht die Berechnung eines neuen Pixelwertes I'(u,v) mit einer 3x3-Filtermaske.

Bei der Filterung an den Filterregionen, die über das Eingabebild hinausgehen, ergeben sich Schwierigkeiten an den Bildrändern. Da die Filterung über benachbarte Pixel ausgeführt wird, gibt es keine Bildwerte zu den zugehörigen Gewichten [BBo6, S. 93]. Für das Problem der Randbehandlung gibt es keine mathematisch korrekte Lösung. Mögliche Lösungen sind die ursprünglichen Pixelwerte des Eingabebildes beizubehalten oder den Pixeln außerhalb des Eingabebildes einen konstanten Wert zuzuweisen [BBo6, S. 113]. Die negativen Werte, die das Ergebnisbild nach der Filterung enthält, werden auf das Grauwertintervall [0...255] normiert [Erho8, S. 154].

2.1.2 *Gaußfilter*

Der *Gaußfilter* ist ein linearer Filter, der zu der Klasse der Tiefpassfilter gehört. Sie kennzeichen sich durch die Unterdrückung der hohen Frequenzanteile im Bild und werden zur Glättung, Weichzeichnung und Rauschunterdrückung eines Bildes verwendet [Erho8, S. 144].

Der Gaußfilter wird durch eine diskrete zweidimensionale Gaußfunktion G(x,y) definiert [BBo6, S. 100]:

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-(x^2+y^2)/2\sigma^2} \tag{2.1.3}$$

wobei σ als *Standardabweichung* und σ^2 als *Varianz* bezeichnet werden.

Die Standardabweichung σ bestimmt die Weite der glockenförmigen Filterfunktion. Für große σ-Werte ist die Gaußfunktion flach und breit. Dies entspricht einer starken

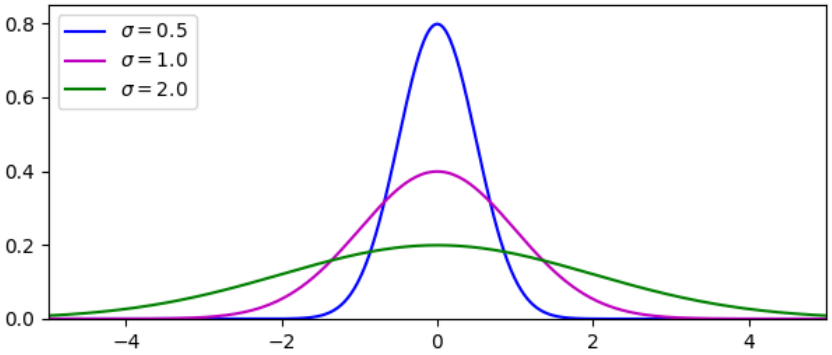

Abbildung 2.1.2: Visualisierung der Verläufe der eindimensionalen Gaußfunktion für die unterschiedlichen σ-Werte.

Glättung. Dagegen wird eine schwache Glättung über kleine σ-Werte reguliert, für die die Gaußfunktion steil und schmal verläuft. Die Abbildung 2.1.2 visualisiert den Einfluss der unterschiedlichen σ-Werte auf den Verlauf der eindimensionalen Gaußfunktion.

In der Literatur werden auch die Koeffizienten der Gaußfiltermaske über die Binomialkoeffiziente definiert. Die Gaußfiltermaske G mit der Größe 3x3 Pixeln ist in Form von

$$G_{3x3} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.1.4}$$

gegeben.

Das Ergebnis der Filterung wird durch die Summe der Koeffizienten dividiert, um die Änderung des Grauwertbereiches zu verhindern [Erho8, S. 149].

2.1.3 *Affine Abbildung*

Eine *affine Abbildung* ist eine geometrische Transformation, die die räumliche Beziehung zwischen Pixeln in einem Bild verändert. Charakteristische geometrische Transformationen sind *Translation*, *Skalierung*, *Rotation*, *Scherung* sowie deren Kombinationen. In der Abbildung 2.1.3 sind Beispiele der geometrischen Transformationen zu sehen.

Abbildung 2.1.3: Veranschaulichung geometrischer Transformationen. 1. Eingabebild 2. Skalierung 3. Rotation 4. horizontale Scherung 5. vertikale Scherung.

Jede affine Abbildung kann durch Matrixmultiplikation definiert werden [GW08, S. 88]:

$$[x\ y\ 1] = [v\ w\ 1]\,\mathbf{T} = [v\ w\ 1] \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{33} & 1 \end{bmatrix} \tag{2.1.5}$$

Elemente der Matrix $\mathbf{T}$ beschreiben eine elementare geometrische Abbildung [GW08, S. 88].

- Translation:

  Bei der Translation wird jede Pixelkoordinate (v, w) um den Vektor (t_x, t_y) auf die neue Pixelkooridinate $(x, y) = (v + t_x, w + t_y)$ verschoben. Die Matrix $\mathbf{T}$ ist definiert als

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix} \tag{2.1.6}$$

- Skalierung:

Um ein Eingabebild zu skalieren, wird jede Pixelkoordinate (v, w) um den Faktor c auf die neue Pixelkoordinate $(x, y) = (c_x v, c_y w)$ berechnet. Die Matrix $\mathbf{T}$ ist definiert als

$$\mathbf{T} = \begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1.7}$$

- Rotation:

  Die Rotation sorgt dafür, dass ein Eingabebild um den Koordinatenursprung $(0, 0)$ mit dem Winkel Θ gedreht wird. Die neue Pixelkoordinate (x, y) wird aus der Pixelkoordinate (v, w) als $(x, y) = (v \cos\Theta - w \sin\Theta,\ v \sin\Theta + w \cos\Theta)$ berechnet. Die Matrix $\mathbf{T}$ ist definiert als

$$\mathbf{T} = \begin{bmatrix} \cos\Theta & \sin\Theta & 0 \\ -\sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1.8}$$

- Scherung:

  Bei der Scherung wird das Eingabebild in der vertikalen oder horizontalen Richtung geschert. Die neue Pixelkoordinate (x, y) wird als $(x, y) = (v + s_v w, w)$ für die vertikale und als $(x, y) = (v, s_h v + w)$ für die horizontale Scherung bestimmt. Die Matrix $\mathbf{T}$ für die vertikale 2.1.9 und die horizontale 2.1.10 Scherung ist definiert als

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1.9}$$

$$\mathbf{T} = \begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1.10}$$

2.2 FALTUNGSNETZE

Tiefe Faltungsnetze werden erfolgreich im Bereich der Mustererkennung und für die Implementierung von Word-Spotting-Verfahren eingesetzt [SF18] [KDJ18]. Obwohl sie erst in der letzten Zeit einen Aufschwung im Bild- und Sprachverstehen erfahren haben, geht die Grundidee auf die 90er Jahre zurück, als das tiefe Faltungsnetz zur Erkennung handgeschriebener Ziffern in [LBD+90] vorgestellt wurde. Der technische Fortschritt sowie die wachsende Rechenleistung, optimierte Algorithmen und die Verfügbarkeit ausreichend großer Datensätze erlauben erstmals, tiefe Faltungsnetze mit mehreren Millionen Parametern zu trainieren. Da tiefe Faltungsnetze eine spezielle Form der künstlichen neuronalen Netze sind, wird zuerst das Grundprinzip der künstlichen neuronalen Netze und dann das der Faltungsnetze erklärt.

2.2.1 *Grundprinzip*

Künstliche neuronale Netze, im Folgenden auch Netze genannt, sind informationsverarbeitende Systeme, die in Anlehnung an das biologische Vorbild des Gehirns von Menschen und Tieren gebildet wurden [KBB+15, S. 7].

Ein Netz besteht aus hierarchisch aufeinander aufgebauten Schichten. Die erste Schicht ist die *Eingabeschicht*, die die Eingabedaten annimmt. Die letzte Schicht ist die *Ausgabeschicht*, die das Ergebnis des Netzes repräsentiert. Dies kann zum Beispiel eine Zuordnung zu einer Klassenkategorie sein [Nie03, S. 384]. Zwischen den Eingabe- und Ausgabeschichten befinden sich *versteckte Schichten*, da sie keinen direkten Zugang zur Umgebung des Netzes haben. Die Anzahl der versteckten Schichten bestimmt die *Tiefe* des Netzes. Die direkt nach der Eingabeschicht folgende versteckte Schicht wird als *niedrige Schicht* bezeichnet und ist mit der *nächsthöheren Schicht* verbunden [GBC16, S. 169].

Jede Schicht besteht aus mehreren Neuronen. Neuronen sind signalverarbeitende Bauelemente, die Informationen über gewichtete Verbindungen versenden [Dö18, S. 89]. Die Art der Verbindung zwischen einzelnen Neuronen ist vom Netztyp und bei Faltungsnetzen vom Schichtentyp abhängig.

Die Abbildung 2.2.1 stellt den schematischen Aufbau eines künstlichen Neurons dar. Ein Neuron verfügt über mehrere Eingänge und einen Ausgang. Um den Ausgangswert u eines Neurons zu berechnen, wird zuerst der Wert y über die Gleichung 2.2.1 bestimmt. Dafür werden die Eingabewerte $x_1, x_2, ..., x_n$ mit den Gewichten

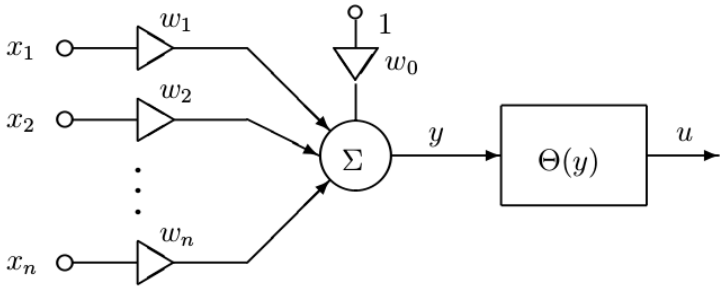

Abbildung 2.2.1: Einfaches Modell eines künstlichen Neurons. Das Bild wurde aus [Nie03] genommen.

$w_1, w_2, ..., w_n$ multipliziert und aufsummiert. Zusätzlich wird die Konstante w_0, die als Bias bezeichnet wird, zu der gewichteten Summe hinzuaddiert.

$$y = \sum_{i=1}^{n} w_i \cdot x_i + w_0 \cdot x_0 \tag{2.2.1}$$

Mit Hilfe einer nichtlinearen Aktivierungsfunktion $\Theta(y)$ wird der Wert y in die Ausgabe des Neurons u transformiert [Nie03, S. 384].

Die wesentliche Aufgabe einer Aktivierungsfunktion ist es, den Ausgabewert eines Neurons in ein fest definiertes Intervall zu überführen. In der Praxis werden unterschiedliche Aktivierungsfunktionen angewendet. Um Netze trainieren zu können, müssen die Aktivierungsfunktionen differenzierbar sein [Dö18, S. 107].

Für diese Arbeit sind zwei Aktivierungsfunktionen relevant:

- *Sigmoidfunktion*

  $$\Theta(y) = \frac{1}{1+e^{-y}}$$

- *Rectified Linear Unit (ReLU)*

  $$\Theta(y) = max(0, y)$$

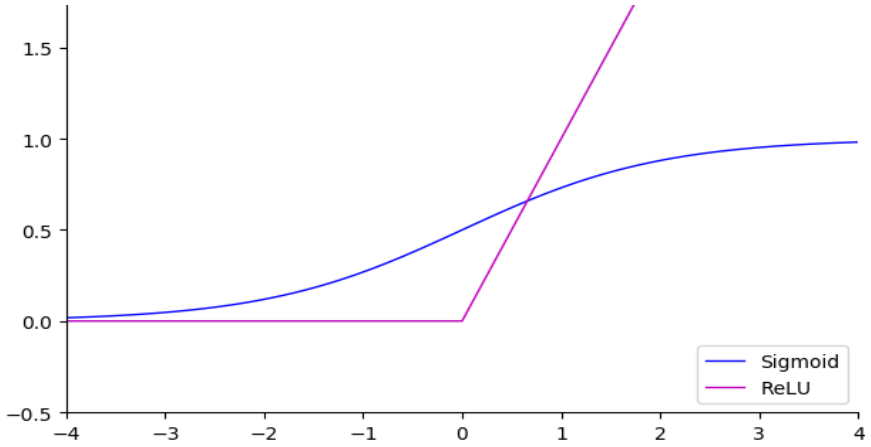

Abbildung 2.2.2: Visualisierung der Verläufe von Sigmoidfunktion und ReLU-Funktion für die Eingabewerte aus dem Intervall $[-4, 4]$ [Dö18, S. 107].

Die Sigmoidfunktion bildet Eingabewerte auf das Intervall $(0, 1)$ ab und wird als Aktivierungsfunktion für die Neuronen der Ausgabeschicht verwendet.

Für die Neuronen der versteckten Schichten wird die ReLU-Funktion als Aktivierungsfunktion verwendet. Nach der Aktivierung liegen die Ausgabewerte von Neuronen im Bereich $[0, \infty)$, ðdie Werte, die kleiner als 0 sind, werden auf 0 gesetzt und Werte, die größer als 0 sind, bleiben erhalten. In der Abbildung 2.2.2 sind die Verläufe der Sigmoidfunktion und der ReLU-Funktion dargestellt.

Der wesentliche Vorteil der ReLU-Funktion im Vergleich zu den anderen Aktivierungsfunktionen ist, dass beim Training der tiefen Netze kein Verschwinden oder kein Explodieren des Gradienten auftritt. Mit der ReLU-Aktivierungsfunktion wird die Trainingszeit kürzer und das Training liefert bessere Ergebnisse [MHN13].

2.2.2 *Architektur*

Faltungsnetze sind tiefe vorwärtsgerichtete Netze, die aus einer Eingabeschicht, einer oder mehreren Faltungsschichten (engl. convolutional layer), Pooling-Schichten (engl. pooling layer) und vollständig verbundenen Schichten (engl. fully connected layer) bestehen. Das Faltungsnetz erwartet Bilder als Eingabe. Dabei wird jedem Bildpixel ein Eingabeneuron zur Kodierung der Pixelintensitäten zugewiesen. Die Neuronenanzahl der Eingabeschicht entspricht der Anzahl der Pixel des Eingabebildes [Dö18, S. 130].

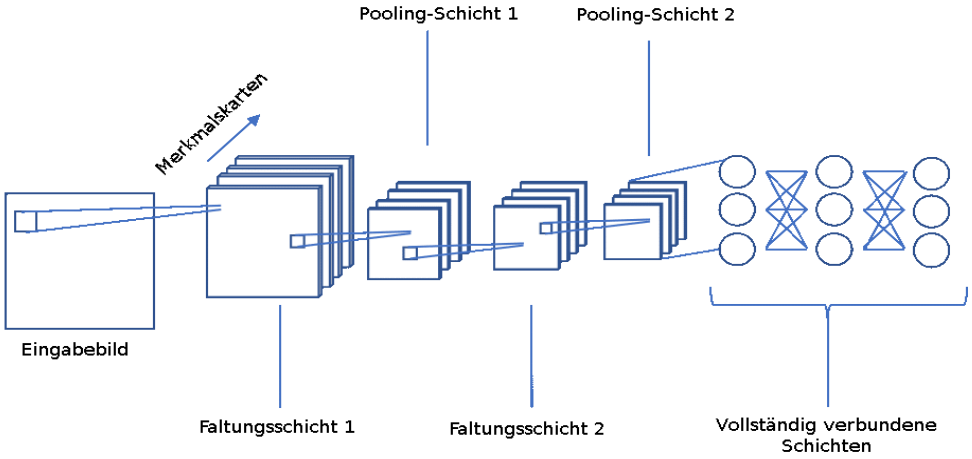

Abbildung 2.2.3: Schematische Darstellung einer einfachen tiefen CNN-Architektur [Mor17]. Ein Eingabebild wird mit trainierbaren Filtermasken in der Faltungsschicht gefaltet. Die entstandenen Merkmalskarten werden in der Pooling-Schicht komprimiert. Die komprimierten Merkmalskarten werden in der nächsten Faltungsschicht erneut gefaltet und danach in einer Pooling-Schicht komprimiert. Anschließend werden die extrahierten Merkmale in Form eines Vektors in einen vollständig verbundenen MLP übergeben.

Die Abbildung 2.2.3 veranschaulicht eine einfache tiefe CNN-Architektur, die aus einer Eingabeschicht, zwei Faltungsschichten, zwei Pooling-Schichten und drei vollständig verbundenen Schichten besteht.

Faltungsschicht

In der Faltungsschicht wird die Eingabematrix parallel mit mehreren Filtermasken gefaltet. Jede Filtermaske erlernt selbstständig ein lokales Merkmal und kann es an unterschiedlichen Positionen des Bildes wiedererkennen. In niedrigen Faltungsschichten werden einfache visuelle Merkmale wie Kanten und Ecken erkannt. In höheren Faltungsschichten werden die erkannten Merkmale zu komplexeren und abstrakteren Formen kombiniert [GBC16, S. 6].

Über die Parameter *Schrittweite* und *Padding* kann die Dimension der Ausgabematrix reguliert werden. Die zweidimensionale Faltung, die über die Gleichung 2.1.2 definiert ist, reduziert die Dimension der Ausgabematrix. Um die gleiche Dimension der Ausgabematrix wie der Eingabematrix zu behalten, wird Padding angewendet. In der Praxis wird häufig das *Zero-Padding-Verfahren* eingesetzt. Dabei werden an den Seiten

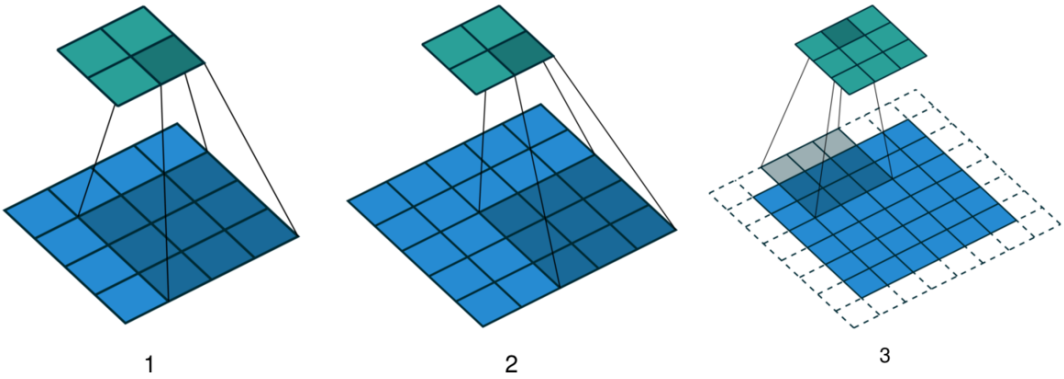

Abbildung 2.2.4: Visualisiert die Berechnungen einer zweidimensionalen Faltung mit einer 3x3-Filtermaske. 1. Das 4x4-Eingabebild 1 wird mit einer 3x3-Filtermaske und einer Schrittweite gleich 1 gefaltet. Nach der Faltungsoperation entsteht ein 2x2-Ergebnisbild. 2. Das 5x5-Eingabebild 2 wird mit einer 3x3-Filtermaske und einer Schrittweite gleich 2 gefaltet. Nach der Faltungsoperation entsteht ein 2x2-Ergebnisbild. 3. Das 6x6-Eingabebild wird mit einer 3x3-Filtermaske, einer Schrittweite 2 und dem Padding 1 gefaltet. Nach der Faltungsoperation entsteht ein 3x3-Ergebnisbild. Das Bild entnommen aus [DV18].

fehlende Zeilen und Spalten mit Nullen hinzugefügt. Zum Beispiel wird bei einer 3x3-Filtermaske an allen Seiten der Eingabe jeweils eine zusätzliche Spalte bzw. Zeile hinzugefügt. Neben der gleichbleibenden Dimension ist auch eine deutlich kleinere Ausgabematrix möglich. Die Dimension der Ausgabematrix kann über eine größere Schrittweite der Filtermaske reduziert werden [Cho18].

In der Abbildung 2.2.4 werden die Berechnungen einer zweidimensionalen Faltung mit unterschiedlichen Parametern dargestellt. Für eine detaillierte Beschreibung der Faltungsoperationen wird auf [DV18] verwiesen.

Die Gewichte der Filtermaske werden mit zufälligen Werten initialisiert und während des Trainings des Netzes erlernt.

Während einer Faltungsoperation benutzen alle Neuronen der Faltungsschicht die gleichen Gewichte. Diese werden nicht für jedes einzelne Neuron, sondern für eine Menge von Verbindungen, die sich das gleiche Gewicht teilen, modifiziert. Damit reduzieren sich der Rechenaufwand und der Speicherbedarf [GBC16, S. 335].

Die durch eine Faltung transformierte Eingabematrix wird als Merkmalskarte (engl. feature map) bezeichnet und mit nichtlinearen ReLU-Aktivierungsfunktionen aktiviert.

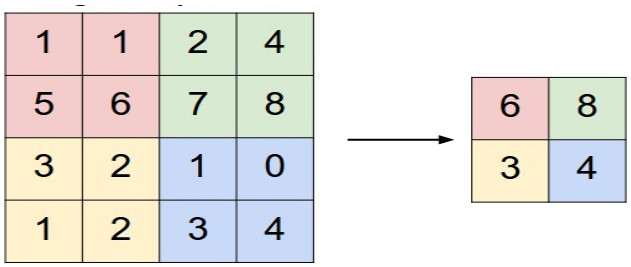

Abbildung 2.2.5: Beispielrechnung mit der Max-Pooling-Strategie sowie einer 2x2-Filtermaske und einer Schrittweite gleich 2. Die neuen Pixelwerte werden durch den maximalen Wert für jede abgedeckte Region bestimmt. Aus der Eingabebildgröße von 4x4 Pixeln ergibt sich ein neues Bild mit der Größe von 2x2 Pixeln. Das Bild wurde aus [LKJ] entnommen.

Die Gesamtzahl der Merkmalskarten korrespondiert mit einer festdefinierten Anzahl von Filtermasken pro Faltungsschicht [GBC16, S. 332].

Jedes Neuron einer Faltungsschicht ist mit einer räumlichen Region der vorherigen Schicht verbunden. Diese Region wird als rezeptives Feld (engl. receptive field) bezeichnet [GBC16, S. 337].

Pooling-Schicht

Die Pooling-Schicht reduziert die räumliche Auflösung der Merkmalskarten der vorherigen Schicht. Dementsprechend werden die Anzahl der Parameter sowie der Rechenaufwand reduziert. Über jede Merkmalskarte wird eine kleine Filtermaske bewegt und für jede abgedeckte Region wird nur ein Wert bestimmt. Die Filtermaske enthält dabei keine Gewichte [GBC16, S. 339].

Die am häufigsten verwendete Pooling-Strategie ist die *Max-Pooling-Strategie*. Hierbei wird jede 2x2-Teilmatrix jeder Merkmalskarte durch ihren größten Wert ersetzt. Da eine Schrittweite gleich 2 verwendet wird, überlappen sich einzelne Pixel nicht und die Dimension der Eingabematrix reduziert sich um 75%. Die Abbildung 2.2.5 veranschaulicht das Prinzip der Max-Pooling-Strategie.

Durch die Pooling-Layer werden die Faltungsnetze robuster gegen kleine Rotationen und Translationen in den Eingabebildern[GBC16, S. 342].

Vollständig verbundene Schicht

Am Ende des tiefen Faltungsnetzes befinden sich eine oder mehrere vollständig verbundene Schichten. Die Architektur dieser Schichten entspricht den mehrschich-

tigen Perzeptronen (engl. Multi Layer Perceptron, MLP), bei denen alle Neuronen vollständig mit Neuronen der vorherigen Schicht verbunden sind.

Die extrahierten Merkmale werden durch die vollständig verbundenen Schichten bearbeitet. Die Neuronenanzahl der Ausgabeschicht korrespondiert mit der Anzahl der Klassifikationsklassen [GBC16, S. 186].

Es wird zwischen *Single-Label-* und *Multi-Label-Klassifikationen* unterschieden [BDW18, S. 221]. Bei der Single-Label-Klassifikation handelt es sich um eine Vorhersage der Zugehörigkeit des Eingabebildes zu einer bestimmten Klasse. Als Aktivierungsfunktion wird in der Regel die *Softmax-Funktion* verwendet, die für jedes Ausgabeneuron die Wahrscheinlichkeit der entsprechenden Klassenzugehörigkeit berechnet. Es wird dasjenige Neuron ausgewählt, für das die Wahrscheinlichkeit am größten ist. Für das Training werden Beispielbilder mit einem sogenannten *One-Hot-Vektor* kodiert. Dabei ist der One-Hot-Vektor ein n-dimensionaler binärer Vektor, bei dem jede Dimension eine Klasse repräsentiert und nur eine Dimension den Wert 1 enthält.

Bei der Multi-Label-Klassifikation handelt es sich um eine Vorhersage der gleichzeitigen Zugehörigkeit eines Eingabebildes zu mehreren Klassen. Hierbei wird hauptsächlich die Sigmoidfunktion als Aktivierungsfunktion der Ausgabeschicht verwendet. Im Gegensatz zur Softmax-Funktion können alle Ausgabeneuronen mit Sigmoidaktivierung gleichzeitig Werte aus dem Intervall $(0, 1)$ enthalten [SF18].

2.2.3 *Training*

Das Error-Backpropagation-Verfahren ist ein am häufigsten eingesetzte Lehrverfahren für das Training der tiefen neuronalen Netze [KBB+15, S. 62]. Da es ein überwachtes Lehrverfahren ist, wird das Netz auf der Basis von annotierten Trainingsbeispielen trainiert. Jedes annotierte Trainingsbeispiel besteht aus einem Paar aus Beispielbild und erwarteten Ausgabe.

Das Modell lernt aus den gegebenen Trainingsbeispielen eine Repräsentation der Daten in Form einer Funktion. Mit Hilfe dieser Funktion können unbekannte Beispielbilder der gleichen Problemklasse wie die Trainingsbeispiele klassifiziert werden [Ert16, S. 206]. Die Fähigkeit, das Gelernte auf neue Beispielbilder anzuwenden, wird als Generalisierung bezeichnet. Die Qualität der Generalisierungsfähigkeit des Lernverfahrens wird durch Testdaten überprüft [Ert16, S. 192].

Die Trainingsphase wird in zwei Schritten ausgeführt. In dem ersten Schritt berechnet das Netz zu einem Beispielbild eine Ausgabe. Der berechnete Wert wird mit dem erwarteten Ausgabewert des Beispielbildes verglichen. Stimmen diese beiden Werte nicht überein, wird versucht, diese Differenz im zweiten Schritt zu verkleinern. In

dem zweiten Schritt werden Netzgewichte korrigiert und von der Ausgabeschicht zur Eingabeschicht zurückpropagiert, sodass der Fehler zwischen den berechneten und den erwarteten Werten minimiert wird. Dafür wird eine Kostenfunktion gewählt. Eine der möglichen Kostenfunktionen ist die Summe der quadratischen Abweichungen (engl. mean square error) von berechneten und erwarteten Ausgabewerten über alle Ausgabeneuronen für ein Trainingsbeispiel p, die folgendermaßen definiert wird:

$$E_p(w) = \frac{1}{2}\sum_{i=1}^{n}(a_i - \widehat{a_i})^2 \tag{2.2.2}$$

Der Term $\frac{1}{2}$ vereinfacht die Berechnung der Ableitung.

Eine weitere Variante ist es, die Kostenfunktion als Kreuzentropie zwischen einer tatsächlichen und einer erwarteten Wahrscheinlichkeitsverteilung (engl. cross entropy loss) zu definieren [SF16]:

$$E_p = -\frac{1}{n}\sum_{i=1}^{n}[a_i \log \widehat{a}_i + (1 - a_i)\log(1 - \widehat{a}_i)], \tag{2.2.3}$$

wobei n die Anzahl der Ausgabeneuronen, a_i die erwarteten und $\widehat{a_i}$ die tatsächlichen Ausgabewerte beschreiben.

Das Ziel des Trainings ist es, ein Minimum der Kostenfunktion zu erreichen. Dies wird durch einen Gradientenabstieg angestrebt. Dazu werden die partiellen Ableitungen der Kostenfunktion nach allen Gewichtsparametern bestimmt. Das daraus resultierende Ergebnis wird als Gradient bezeichnet. Er gibt die Richtung des steilsten Abstiegs der Kostenfunktion an. Um den Fehler zu verringern, werden die Gewichte in die entgegengesetzte Richtung des Gradienten geändert [KBB+15, S. 58]. Die Gewichte werden in jedem Schritt $i+1$ für jedes Trainingsbeispiel nach der Gleichung 2.2.4 angepasst.

$$w_{i+1} = w_i + \Delta w_i, \qquad \Delta w = -\eta\frac{\partial E_p(w_i)}{\partial w_i} \tag{2.2.4}$$

Durch die Lernrate η wird bestimmt, wie schnell das Verfahren konvergiert. Wird η zu klein gewählt, dann konvergiert das Verfahren sehr langsam. Wird η zu groß gewählt, dann kann das Minimum übersprungen werden und das Verfahren konvergiert nicht oder divergiert sogar. Die Lösung kann darin bestehen, dass die Lernrate η in Abhängigkeit von der Anzahl der Iterationen verringert wird [KBB+15, S. 67].

Um den Einfluss der Lernrate auf das Konvergierungsverhalten zu verringern, wurden Erweiterungen des Gradientenabstiegsverfahrens entwickelt. Die Gewichtsäderungsregel, die häufig bei tiefen Faltungsnetzen verwendet wird [KSH12] [SF16] [GSF17], lautet für den Schritt $i+1$ folglich

$$w_{i+1} = w_i + v_{i+1}, \qquad v_{i+1} = \alpha v_i - \gamma\eta w_i - \eta \frac{\partial E_p(w_i)}{\partial w_i}, \tag{2.2.5}$$

wobei α ein Momentum-Parameter, γ der Gewichtsverfall (engl. weight decay) und η die Lernrate ist. Der Momentum-Parameter α muss kleiner als 1 und größer als 0 sein, um das Verfahren stabil zu halten [KBB+15, S. 70].

Der Momentum-Parameter kann das Lernen im Parameterraum, in dem die Kostenfunktion flach verläuft und in eine einheitliche Richtung fällt, beschleunigen. Außerdem wird die Schrittweite je nach Verlauf der Kostenfunktion durch den Momentum-Parameter verkleinert oder vergrößert. Bei einer zu großen oder zu kleiner Lernrate hat das Momentum-Parameter keinen Einfluss auf die Schrittweite [KBB+15, S. 70]. Die Aufgabe des Gewichtsverfalls γ ist es, große Gewichte zu vermeiden. In jedem Schritt werden die Gewichte um kleine Anteile verringert [KBB+15, S. 73].

Zusätzlich wird zwischen *Online-* und *Offline-Training* unterschieden. Beim Online-Training wird der Gradient nach dem Durchlaufen eines Trainingsbeispiels berechnet. Beim Offline-Training wird der durchschnittliche Gradient über eine kleine Menge von zufällig ausgewählten Trainingsbeispielen bestimmt. Dieser Ansatz wird als *Mini-Batch-Training* oder *stochastischer Gradientenabstieg* (engl. Stochastic Gradient Descent, SGD) bezeichnet. Da ein tiefes Netz anhand einer großen Menge von Trainingsbeispielen trainiert wird, ist eine Berechnung des durchschnittlichen Gradienten über alle Trainingsbeispiele unerwünscht oder sogar unmöglich.

Das Durchlaufen aller Lernbeispiele wird als *Epoche* oder *Trainingsiteration* bezeichnet [KBB+15, S. 61]. Das Netz wird in mehreren Epochen trainiert.

2.2.4 *Feinabstimmung*

Das Training eines tiefen Faltungsnetzes von Grund auf ist sehr zeitintensiv und mit einem hohen Rechenaufwand verbunden. Aufgrund einer komplexen und tiefen Architektur enthält ein Faltungsnetz eine enorme Anzahl von erlernbaren Parametern. Die Trainingsdauer kann sich auf mehrere Tage oder Wochen belaufen. Außerdem ist eine große Menge an annotierten Trainingsbeispielen erforderlich. Eine kleine Menge von Trainingsbeispielen führt sonst zu einer Überanpassung (siehe Unterabschnitt 2.2.5) [BDW18, S. 111].

Um die oben genannten Probleme umzugehen, erfolgt das Training eines tiefen Faltungsnetzes nicht von Grund auf, sondern auf der Basis eines vortrainierten Faltungsnetzes. Diese Lerntechnik wird als Feintuning (engl. fine-tuning) bezeichnet. Das Training des Faltungsnetzes mit neuen Trainingsbeispielen beginnt ab dem Punkt, an dem das vortrainierte Faltungsnetz das Training abgeschlossen hat [BDW18, S. 111].

Die Übertragung des erlernten Wissens für eine bestimmte Aufgabe zum Erlernen einer neuen Aufgabe wird als Transfer Learning bezeichnet. Es werden drei Szenarien für das Transfer Learning definiert [BDW18, S. 111]:

- Merkmalsextraktion

  Das vortrainierte Faltungsnetz wird verwendet, um Merkmale aus einem neuen Datensatz zu extrahieren. Dafür wird die letzte vollständig verbundene Schicht des vortrainierten Faltungsnetzes entfernt. Dann werden die Merkmalsvektoren für das Training eines linearen Klassifikators, zum Beispiel Support Vector Machine, benutzt [BDW18, S. 111].

- Adaption

  Das vortrainierte Faltungsnetz wird auf eine neue Problemstellung adaptiert, indem die letzte Ausgabeschicht durch die korrekte Anzahl an Neuronen ersetzt wird. Die erlernten Gewichte des vortrainierten Faltungsnetzes werden für das Training des neuen Faltungsnetzes übernommen. Anschließend wird das neue Faltungsnetz mit dem Error-Backpropagation-Verfahren (siehe Unterabschnitt 2.2.3) trainiert und die Gewichte werden an neue Trainingsbeispiele angepasst. Die Anpassung der Gewichte kann für alle oder alternativ nur für die höheren Schichten erfolgen, indem die Gewichte der niedrigeren Schichten fixiert werden. Die Adaption des vortrainierten Modells funktioniert auch bei einer kleinen Menge von Trainingsbeispielen, wenn das vortrainierte Modell anhand eines ausreichend großen Datensatzes vortrainiert wurde und die Problemstellungen ähnlich sind [BDW18, S. 111].

- Retraining

  Das Retraining eines Faltungsnetzes wird empfohlen, wenn ein Modell von Grund auf trainiert werden muss und das Training des Faltungsnetzes mehrere Tage dauert. Um den Trainingsaufwand zu reduzieren, kann ein vortrainiertes Modell für eine neue Problemstellung nachtrainiert werden. Die erlernten Gewichte des vortrainierten Faltungsnetzes werden für das Training des neuen Faltungsnetzes übernommen. Diese Vorgehensweise hat den Vorteil, dass das

vortrainierte Modell bereits erlernte Filtermasken enthält, die sich zur Unterscheidung einfacher allgemeiner Muster wie zum Beispiel Kanten und Ecken anwenden lassen. Das Modell muss nur eine neue Zuordnung von einfachen Mustern lernen. Da die erlernten Gewichte übernommen und nicht zufällig initialisiert werden, verkürzt sich die Trainingszeit [BDW18, S. 111].

Die Größe und die Ähnlichkeit der Datensätze spielen eine bedeutsame Rolle, ob die Feinabstimmung wirksam angewendet werden kann. Bei einem viel zu kleinen Datensatz (weniger als 2000 Bilddaten) ist es sinnvoller, das vortrainierte Modell als Merkmalsextraktor zu verwenden. Wenn die beiden Bilddomains sich zu stark unterscheiden, ist das Training von Grund auf effizienter als das Retraining [BDW18, S. 223]. Bei der Feinabstimmung wird die Lernrate im Vergleich zu der Lernrate des vortrainierten Faltungsnetzes in der Regel kleiner ausgewählt [BDW18, S. 111].

2.2.5 *Überanpassung*

Mit Überanpassung (engl. Overfitting) wird eine Situation beschrieben, bei der das Netz eine gute Erkennungsleistung bei den Trainingsdaten und eine schlechte Erkennungsleistung bei den Testdaten aufweist. Ein tiefes Faltungsnetz besteht aus Millionen von Parametern und es kann vorkommen, dass sich das Faltungsnetz an Trainingsbeispiele überanpasst, insbesondere wenn die Menge von Trainingsbeispielen nicht ausreichend groß ist. Um dies zu vermeiden, werden Regularisierungstechniken wie zum Beispiel *Dropout* und *Data Augmentation* angewendet.

Dropout ist eine Regularisierungstechnik, bei der Neuronen während des Trainings zufällig deaktiviert werden. Die aktiven Neuronen müssen ihre Gewichte anpassen, um gute Ergebnisse zu erzielen. Während jeder Trainingsiteration werden unterschiedliche Kombinationen von aktiven Neuronen trainiert. Dadurch wird das Netz robuster, da es sich nicht auf bestimmte Neuronenaktivierungen verlassen kann, und es wird dazu gezwungen, eine größere Anzahl von abstrakte Merkmalen zu lernen. Nach dem Training stehen alle Neuronen mit den erlernten Parametern wieder zur Verfügung. Der Nachteil der Dropout-Technik ist die längere Trainingsdauer [SKS+14].

Die Datenaugmentierung ist eine Regularisierungstechnik, um mehr Trainingsdaten zu erhalten, ohne neue Daten manuell zu annotieren. Die neuen Trainingsbeispiele werden aus vorhandenen Bilddaten künstlich erzeugt, indem verschiedene Bildoperationen angewendet werden. Einige möglichen Bildoperationen sind Bildbeschnitt, Spiegelung, Hinzufügen von Rauschen und Transformationen wie Skalierung, Verzerrung und Drehung (siehe Unterabschnitt 2.1.3).

Mit künstlich erzeugten Daten wird die Anzahl der verschiedenen Klassen ausgeglichen, weil sonst die Wahrscheinlichkeit groß ist, dass das Netz sich an eine Klasse mit einer kleinen Anzahl von Bilddaten schneller anpasst, als an eine Klasse mit einer größeren Anzahl [SF16], [SSP03], [ZSC17].

3 VERWANDTE ARBEITEN

Die vorliegende Diplomarbeit befasst sich mit einem segmentierungsbasierten Word-Spotting-Ansatz, mit dem Query-by-Example- und Query-by-String-Szenarien abgebildet werden können. Zur Realisierung der beiden Anwendungsszenarien werden textuelle und visuelle Wortrepräsentation mit Hilfe der Einbettung von PHOC-Attributen in einen gemeinsamen Vektorraum transformiert. Die PHOC-Attribute werden im Abschnitt 3.1 detailliert erläutert. Um ein Wortabbild durch einen PHOC-Vektor in einem n-dimensionalen Vektorraum zu repräsentieren, wird ein tiefes PHOC-Netz verwendet, dessen Aufbau und Funktionsweise im Abschnitt 3.2 beschrieben wird.

Der CNN-basierte Word-Spotting-Ansatz funktioniert gut bei handgeschriebenen und historischen handgeschriebenen Dokumenten, mit denen das Modell trainiert wurde. Das wurde an einem modernen handgeschriebenen Datensatz sowie einer Sammlung von historischen handgeschriebenen Datensätzen in [SF18] demonstriert. Bei neuen handgeschriebenen Dokumentsammlungen, deren Handschriften keine ähnlichen visuellen Charakteristika mit den Handschriften der Trainingsbeispiele des bereits trainierten Modells aufweisen, ist ebenfalls eine ausreichend große Menge von manuell annotierten Trainingsbeispielen erforderlich, um das Modell an neue handgeschriebene Textdokumente anzupassen. Diese Voraussetzung erfordert viel Zeit und einen großen manuellen Aufwand. Als Alternative zu den von Hand annotierten Wortabbildern bietet sich die Generierung synthetischer Trainingsbeispiele an. Im Abschnitt 3.3 wird ausführlich über die Erstellung synthetischer Trainingsbeispiele diskutiert. Ein Verfahren zur Reduzierung des manuellen Aufwandes unter Verwendung eines synthetischen Datensatzes, das im Fokus dieser Diplomarbeit liegt, wird im Abschnitt 3.4 beschrieben.

3.1 PHOC-REPRÄSENTATION

PHOC ist eine gemeinsame Repräsentation eines textuellen und visuellen Wortes, die in [AGFV14] vorgestellt wird. Ein Wort wird als ein binärer PHOC-Vektor in einem d-dimensionalen Vektorraum dargestellt, der sich aus der Konkatenation von binären Vektoren ergibt.

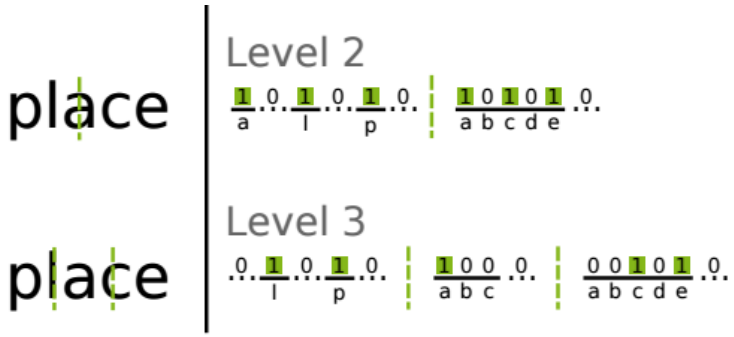

Abbildung 3.1.1: Berechnung der binären Vektoren auf Ebene 2 und Ebene 3. Das Bild wurde aus [SF18] entnommen.

Um ein Wort möglichst diskriminativ und robust zu repräsentieren, wird es in mehrere Regionen aufgeteilt. Für jede Wortregion wird ein binärer Vektor bestimmt, der kennzeichnet, ob ein bestimmtes Zeichen in dieser Wortregion vorkommt. Die Anzahl der Regionen, in die das Wort aufgeteilt wird, ist über die Nummer der Ebene zu bestimmen. Ein Wort auf der Ebene 2 wird also in zwei Regionen aufgeteilt. Die Anzahl der Ebenen kann beliebig ausgewählt werden [AGFV14].

Die Länge eines binären Vektors entspricht der Anzahl aller Zeichen eines Alphabetes. Beispielsweise setzt sich ein binärer Vektor für das lateinische Alphabet aus 26 Buchstaben und 10 Ziffern zusammen und hat somit eine Länge von 36 Zeichen. Dabei wird nicht zwischen großen und kleinen Buchstaben unterschieden.

In der Abbildung 3.1.1 ist als Beispiel das Wort „place“ für die Berechnung der binären Vektoren auf den Ebenen 2 und 3 zu sehen. Das Zeichen „a“ wird den beiden Regionen der Ebene 2 zugewiesen, da sein Bereich genau zu 50% mit der linken und der rechten Region überlappt.

In dieser Arbeit werden PHOC-Repräsentationen für die Ebenen 2 bis 5 bestimmt. Jedes mit lateinischen Zeichen geschriebene Wortabbild wird als ein 504-dimensionaler PHOC-Vektor dargestellt. Dabei entspricht jede Dimension des PHOC-Vektors einem Attribut, das kennzeichnet, ob und in welchem Teil des Wortes ein bestimmtes Zeichen vorkommt.

Der PHOC-Vektor lässt sich direkt aus einer textuellen Repräsentation eines Wortes berechnen. Um ein Wortabbild im PHOC-Vektor abzubilden, muss zuerst das Word-Spotting-Modell anhand von Trainingsbeispielen trainiert werden. Die Trainingsbeispiele bestehen aus einer Menge von Wortabbildern, für die Annotationen bekannt sind. Die PHOC-Vektoren zu den Wortabbildern werden aus den Annotationen abgeleitet. Da die Attribute während der Trainingsphase voneinander unabhängig

gelehrt werden, ist das trainierte Modell in der Lage Suchanfragen durchzuführen, für die keine Trainingsbeispiele der gleichen Wortklassen wie die der Suchanfragen mittrainiert wurden [AGFV14].

In der Arbeit von [AGFV14] erfolgt das Lernen von PHOC-Attributen mit *Support Vector Machines* (SVMs). Zunächst werden wichtige visuelle Merkmale von Wortabbildern in Fisher-Vektoren kodiert. Für die Bestimmung jedes Attributes wird jeweils eine SVM mit den Merkmalsvektoren und den PHOC-Vektoren trainiert.

Eine weitere Möglichkeit zum Lernen von PHOC-Attributen mit einem tiefen Faltungsnetz wird in [SF16] vorgestellt und im nächsten Abschnitt 3.2 beschrieben.

3.2 PHOC-NETZ

Das *PHOC-Netz* ist eine erfolgreiche CNN-Architektur für Word-Spotting-Aufgaben. Es wurde zum ersten Mal in [SF16] vorgestellt und mit dem „Best Paper Award ICFHR2016“ ausgezeichnet. Die Autoren übertrafen konkurrierende Methoden für den segmentierungsbasierten Word-Spotting-Ansatz. Derzeit repräsentiert das PHOC-Netz den aktuellen Stand der Technik [SF18]. Im Gegensatz zu den anderen CNN-basierten Ansätzen, die vortrainierte tiefe Faltungsnetze für die Word-Spotting-Aufgaben adaptieren [KJ18] [KDJ18], wird das PHOC-Netz von Grund auf trainiert. Der Ausgangsvektor des PHOC-Netzes wird als holistische Repräsentation eines Wortabbildes verwendet.

Die PHOC-Netz-Architektur setzt sich aus 13 Faltungsschichen, 2 Max-Pooling-Schichten, die jeweils nach der 2-ten und 4-ten Faltungsschichten folgen, einer räumlichen Spatial-Pyramid-Schicht sowie vier vollständig verbundenen Schichten zusammen [SF16]. Die Abbildung 3.2.1 stellt die schematische Darstellung der PHOCNet-Architektur dar.

Die Faltungsoperation wird mit Hilfe einer kleinen 3x3-Filtermaske mit einer Schrittweite von 1 durchgeführt. Der Vorteil kleiner Filter gegenüber größeren ist, dass sie weniger lernbare Parameter benötigen und damit zu einer besseren Resistenz gegenüber der Überanpassung führen. Um die Auflösung der Wortabbilder nach der Faltungsoperation zu erhalten, wird das *Zero-Padding*-Verfahren (siehe 2.2.2) verwendet. Die Anzahl der Filtermasken wird von niedrigen zu höheren Faltungsschichten erhöht, um mehr abstrakte Merkmale auf den höheren Faltungsschichten zu erlernen. Sämtliche Faltungsschichten sind mit der ReLU-Aktivierungsfunktion versehen [SF16].

Im Gegensatz zur im Abschnitt 2.2.2 beschriebenen gängigen Architektur der Faltungsnetze enthält die Architektur des PHOC-Netzes eine zusätzliche Spatial-Pyramid-Schicht. Diese Schicht wird vor den vollständig verbundenen Schichten eingebaut und

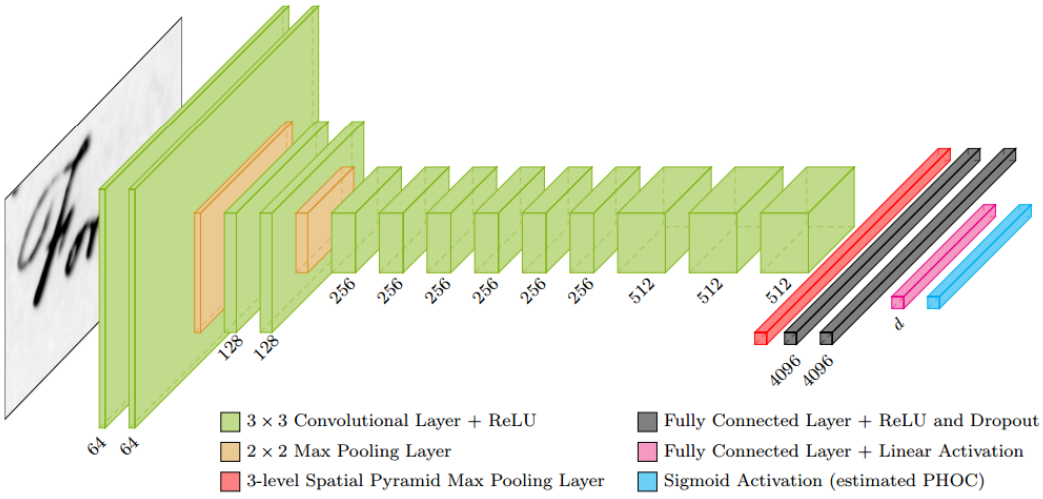

Abbildung 3.2.1: Visualisierung der Architektur des PHOC-Netzes. Das Bild wurde aus [SF18] entnommen.

dient dazu, dass das PHOC-Netz Eingabebilder in verschiedenen Größen verarbeiten kann. Dies ist insbesondere für Handschriftdaten bedeutsam, da das visuelle Aussehen von Wörtern beim Zuschneiden oder durch Verformungen beim Skalieren beeinträchtigt wird. Zudem können Verformungen oder das Zuschneiden eines Wortabbilders zum Verlust der erheblichen semantischen Charakteristika von einzelnen Zeichen führen [SF16].

Um auch kleine Wortabbilder bearbeiten zu können, werden bei der PHOC-Net-Architektur nur zwei Max-Pooling-Schichten mit der Schrittweite von 2 zur Reduktion von freien Parametern eingesetzt. Die Merkmalskarten der letzten Faltungsschicht sind die Eingabe für die Spatial-Pyramid-Pooling-Schicht, die dazu dient, dass die darauf folgende voll verbundene Schicht als Eingabe einen 4096-dimensionalen Merkmalsvektor bekommt [SF16].

Die ersten zwei vollständig verbundenen Schichten, die auf der räumlichen Pyramiden-Pooling-Schicht folgen, werden durch die ReLU-Aktivierungsfunktion aktiviert. Während des Trainings wird die Hälfte der Neuronen dieser beiden Schichten zufällig deaktiviert. Um eine Überanpassung zu vermeiden, wird neben *Dropout* die Anzahl von Trainingsbeispielen mit Hilfe von geometrischen Bildtransformationen vergrößert (vgl. 2.2.5).

Die Neuronenanzahl der letzten beiden vollständig verbundenen Schichten korrespondiert mit der Anzahl von Attributen, die Wortabbilder repräsentieren. Jedes

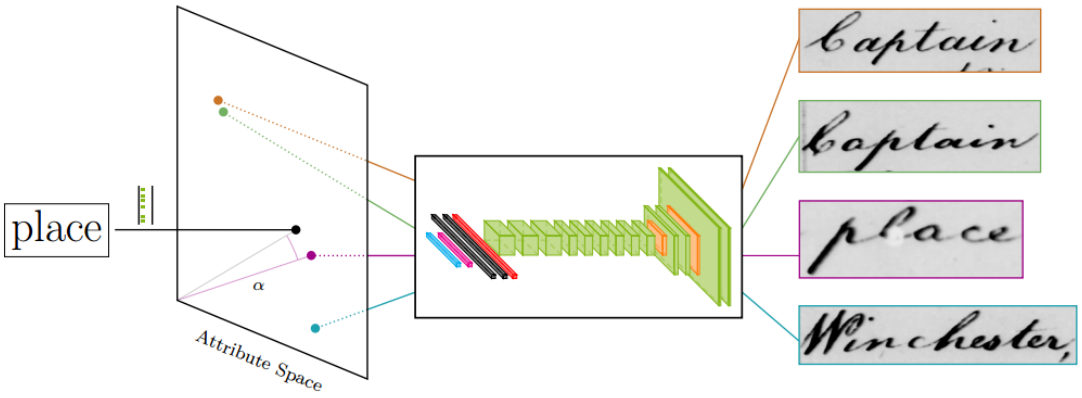

Abbildung 3.2.2: Visualisierung der Berechnung einer Query-by-String-Anfrage. Wortabbilder und Anfrage werden anhand der Einbettung der PHOC-Attribute in einem gemeinsamen Vektorraum abgebildet. Die Ähnlichkeit zweier PHOC-Vektoren wird durch das Kosinus-Ähnlichkeitsmaß bestimmt. Das Bild wurde aus [SF18] entnommen.

Neuron der Ausgabeschicht wird mit Sigmoidfunktion aktiviert und gibt aus, ob ein bestimmtes Zeichen in einer bestimmten lokalen Region eines Wortabbildes vorkommt. Dabei handelt es sich um die *Multi-Label-Klassifizierung* (vgl. 2.2.2), bei der jedes Attribut unabhängig voneinander bestimmt wird [SF16].

Bei der Suche wird ein binärer PHOC-Vektor aus der Suchanfrage berechnet. Die Suchanfrage in Form einer Zeichenkette wird direkt über die Berechnung der binären Attribute in einem d-dimensionalen Vektorraum abgebildet. Bei einer Bildanfrage wird ein PHOC-Vektor durch das trainierte PHOC-Netz in einem d-dimensionalen Vektorraum berechnet. Die Berechnung der Ähnlichkeit zwischen dem binären PHOC-Vektor der Suchanfrage und den PHOC-Vektoren von Wortabbildern erfolgt anschließend durch das Kosinus-Ähnlichkeitsmaß [SF16]. Die Abbildung 3.2.2 verdeutlicht das Prinzip der Berechnung einer Query-by-String-Anfrage durch das mit dem PHOC-Netz implementierte Word-Spotting-Modell.

3.3 SYNTHETISCHE TRAININGSDATEN

Neben der richtigen Auswahl einer CNN-Architektur und den Trainingsparametern ist eine ausreichend große Menge von Trainingsbeispielen für die Leistungsfähigkeit eines CNN-basierten Word-Spotting-Modells entscheidend. Aufgrund einer hohen

Varianz der Schreibweise und eines in der Regel schlechten Zustandes der historischen handgeschriebenen Textdokumente ist die Schaffung von ausreichend vielen Trainigsbeispielen eine große Herausforderung. Außerdem ist eine manuelle Erstellung von Trainingsbeispielen ein aufwändiger Prozess, da die Wortabbilder von Hand annotiert werden müssen. Des Weiteren enthalten manche historische handgeschriebene Dokumente und Manuskripte nicht ausreichend viele Textseiten, um das Trainingsmaterial zu erstellen. In solchen Fällen entspricht die Erzeugung von Trainingsbeispielen dem Transkribieren eines Textdokumentes.

Um das Problem des Fehlens eines ausreichend großen Trainingsdatensatzes zu umgehen, bietet sich die Generierung von synthetischen Wortabbildern an. Eine der am häufigsten benutzte Methode bei CNN-basierten Ansätzen, um einen Datensatz künstlich zu vergrößern, ist die Datenaugmentierung (siehe Unterabschnitt 2.2.5).

In der Literatur werden unterschiedliche Augmentierungsmethoden verwendet und untersucht. Die Verbesserung der Erkennungsleistung des tiefen Faltungsnetzes wird durch elastische Transformationen von Beispielbildern der handgeschriebenen Zahlen erreicht [SSP03]. In der Arbeit von [ZSC17] wird die Trainingsmenge durch Deformierungen der realen Bilddaten erweitert. Eine Modellierung von Störungen und ein Hinzufügen von Rauschen bei den Bildern führen zu einem positiven Effekt auf die Erkennungsleistung der tiefen Faltungsnetzen.

Der Nachteil der Datenaugmentierung besteht darin, dass die Erstellung neuer künstlicher Trainingsbeispiele auf der Basis manuell annotierter realer Wortabbilder erfolgt.

Ein weiteres Verfahren zur Erstellung eines synthetischen Datensatzes wird in [KJ16] beschrieben. Der synthetische IIIT-HWS-Datensatz enthält 9 Millionen künstlicher Wortabbilder, die mit der Handschrift ähnelnden Computerschriften generiert wurden. Insgesamt wurden 750 Computerschriften zur Erstellung von künstlichen Wortabbildern ausgewählt. Das aus dem englischen Hunspell-Wörterbuch ausgewählte Vokabular des synthetischen Datensatzes besteht aus 90.000 Wörtern und Zahlen. Jede Wortklasse besteht aus 100 künstlichen Wortabbilder, die mit zufällig ausgewählten Computerschriften generiert werden. Dabei werden Wörter entweder klein, groß oder der erste Buchstabe wird groß geschrieben. Die Schriftstärke und der Buchstabenabstand des jeweiligen Wortes werden zufällig geändert. Die Größe der synthetischen Wortabbilder in dem Datensatz wurde auf 128x48 normiert [KJ16].

Um die computergenerierten Wortabbilder realistischer darzustellen, werden zusätzlich Pixelwerte im Bildvorder- und -hintergrund als Gaußverteilung um einen Erwartungswert mit einer Varianz modelliert. Der Erwartungswert und die Varianz werden aus den handgeschriebenen Wortabbildern des IAM-Datensatzes automa-

tisch geschätzt. Anschließend werden die computergenerierten Wortabbilder mit dem Gauß-Filter geglättet [KJ16].

Eine der großen Herausforderungen bei der Generierung synthetischer Bilddaten ist die Nachbildung der realitätsnahen charakteristischen visuellen Merkmale realer Wortabbilder. Aus diesem Grund werden manuell annotierte reale Wortabbilder benötigt um ein auf der Basis synthetischer Trainingsdaten trainiertes Modell an handgeschriebene Textdokumente anzupassen und eine angemessene Erkennungsleistung zu erzielen.

Der Literatur ist zu entnehmen, dass der synthetische IIIT-HWS-Datensatz zu einem großen Teil für das Vortraining von tiefen Faltungsnetzen angewendet wird. Um die vortrainierten Modelle an reale Daten anzupassen, wird weiterhin eine ausreichend große Menge von manuell annotierten Trainingsbeispielen benötigt [KJ18] [KDJ18].

Ein Verfahren, indem das Word-Spotting-Verfahren als Wortklassenproblem definiert wird, wird in [KJ18] vorgestellt. In dem Verfahren werden die Suchanfragen einer Wortklasse zugeordnet. Das vorgeschlagene tiefe Faltungsnetz HW-Netz v2 wird zuerst auf der Basis des synthetischen Datensatz vortrainiert. Nach dem Vortraining wird das HW-Netz v2 mit einem realen und vollständigen Trainingsdatensatz feingetunt. Synthetische Bilddaten werden zusätzlich skaliert, rotiert, geschert und elastisch verformt, um die synthetischen Handschriften den realen Handschriften anzunähern.

Ein anderes Verfahren wird zur Vorhersage von PHOC-Attributen in [KDJ18] beschrieben. Das vorgeschlagene Verfahren verwendet das auf der Basis synthetischer Wortabbilder vortrainierte HW-Netz v2. Das Faltungsnetz wird als Merkmalsextraktor auf realen Daten trainiert. Die Ausgabe der vorletzten Schicht des Netzes wird als Merkmalsvektor für das Training der in [AGFV14] vorgestellten AttributSVMs verwendet. In der Testphase wird für jede QbS-Anfrage zusätzlich ein synthetisches Wortabbild generiert. Dadurch wird eine hohe Leistungsfähigkeit des Word-Spotting-Modelles für das Query-by-String-Anwendungsszenario mit einem handgeschriebenen historischen Datensatz und einem modernen handgeschriebenen Datensatz erreicht. Leider lässt sich das Verfahren aufgrund eines sehr hohen Trainingsaufwandes und hoher manueller Kosten für eine praktische Anwendung nicht realisieren.

3.4 SCHWACH ÜBERWACHTES LERNEN

Ein Erfolg versprechendes Verfahren wird für die Reduzierung des manuellen Aufwandes in [GSF17] vorgestellt. Es beruht auf der Idee, die Anzahl von manuell erstellten Trainingsbeispielen unter der Verwendung von synthetischen Trainingsbeispielen zu reduzieren. Die Autoren zeigten, dass schon mit relativ kleinen Mengen von realen Trai-

ningsbeispielen eine konkurrenzfähige Leistungsfähigkeit des Word-Spotting-Modells erreicht werden kann. Demzufolge reduziert sich der manuelle Aufwand.

Das vorgeschlagene Verfahren wird mit dem PHOC-Netz implementiert [SF16]. Dafür wird das Training des Modells in zwei Phasen durchgeführt. In der ersten Trainingsphase wird das PHOC-Netz von Grund auf anhand des synthetischen HW-SYNTH-Trainingsdatensatzes vortrainiert. Der HW-SYNTH-Trainingsdatensatz ist eine Teilmenge des synthetischen IIIT-HWS-Datensatzes und wird im Abschnitt 3.3 beschrieben. Um das PHOC-Netz vorzutrainieren, werden 750.000 synthetische Wortabbilder verwendet, die 10.000 Wortklassen repräsentieren. Jede Wortklasse enthält 75 Wortabbilder. Die synthetischen Wortabbilder werden zufällig um einen Faktor aus dem Intervall $[1, 2)$ skaliert, um die Variabilität von Wortlängen abzubilden [GSF17].

In der Testphase wird der Nutzen von synthetischen Bilddaten zur PHOC-Vorhersage von realen Bilddaten ausgewertet. Aufgrund der deutlichen visuellen Unterschiede zwischen computergenerierten und realen Wortabbildern wird eine geringe Leistungsfähigkeit des Word-Spotting-Modells erzielt.

Um die visuellen Unterschiede zwischen synthetischen und realen Bilddaten zu verringern, wird das vortrainierte PHOC-Netz in der zweiten Trainingsphase an eine Menge von manuell annotierten realen Wortabbildern angepasst. Für das Experiment werden vier Trainingsmengen mit 100, 250, 500 und 1000 zufällig ausgewählten annotierten handschriftlichen Wortabbildern definiert [GSF17]. Jede Trainingsmenge wird zusätzlich über geometrische Bildtransformationen augmentiert, wie es in [SF16] vorgeschlagen wird.

Das vorgestellte Verfahren wird als schwach überwachtes Lernverfahren durch die Reduktion des manuellen Aufwandes bezeichnet, da nur eine relativ kleine Menge von manuell annotierten Trainingsdaten für die Adaption des PHOC-Netzes benötigt wird, um eine konkurrenzfähige Erkennungsleistung zu erreichen. Bei der Reduzierung der Anzahl von realen Trainingsbeispielen um 86% für einen historischen handgeschriebenen Datensatz und um 98% für einen modernen handgeschriebenen Datensatz werden vergleichbare Ergebnisse mit dem AttributSVMs-Verfahren[AGFV14] erreicht. Das vorgestellte Verfahren bietet sich als Alternative bei handgeschriebenen und historischen handgeschriebenen Textdokumenten an, für die keine oder nur eine begrenzte Anzahl von Trainingsbeispielen vorhanden sind [GSF17].

4 METHODIK

Das Ziel dieser Diplomarbeit ist die Generierung synthetischer Daten für einen CNN-basierten Word-Spotting-Ansatz, um den manuellen Aufwand zu reduzieren. Aufgrund einer hohen Varianz der Schreibweise und eines oft schlechten Zustandes der handgeschriebenen und historischen handgeschriebenen Textdokumente ist es erforderlich, das Modell an die jeweilige Sammlungen von Textdokumenten anzupassen. Dafür wird eine ausreichend große Menge von Trainingsbeispielen benötigt, die sich in der Regel schwer erstellen lässt und großen manuellen Aufwand und Zeit kostet. Deshalb ist es wünschenswert, ein vortrainiertes Modell zu haben, das sich mit einem geringen Aufwand an neue Sammlungen handgeschriebener Textdokumente anpassen lässt. Das in [GSF17] vorgestellte Verfahren zur Senkung der Anzahl der nötigen manuell annotierten Beispieldaten unter Verwendung der synthetischen Daten (vgl. Unterabschnitt 3.4) dient als Grundlagen für die Arbeit. Nach diesem Verfahren werden mehrere CNN-basierte Word-Spotting-Modelle anhand unterschiedlich großer synthetischer Datensätze vortrainiert und anschließend mit kleinen Mengen von manuell annotierten Wortabbildern angepasst. Im vorliegenden Kapitel werden die verwendeten Methoden zur Generierung synthetischer Daten ausführlich beschrieben. Zu diesem Zweck wird das in [KJ16] vorgestellte Verfahren zur Generierung synthetischer Datensatz angewendet (vgl. 3.3) und erweitert.

4.1 VORGEHENSWEISE

Für eine gute Leistungsfähigkeit eines lernbasierten Word-Spotting-Modells sind die Güte und die Anzahl von Trainingsbeispielen von Bedeutung. Um ein robustes und schreiberunabgängiges Word-Spotting-Modell zu trainieren, ist eine vielfältige Anzahl von Trainingsbeispielen erforderlich, die eine hohe Schreibvarianz und einen in der Regel schlechten Qualitätszustand von historischen handgeschrieben Textdokumenten repräsentiert. Für die von Hand geschriebenen Wörtern ist eine enorme Variabilität charakteristisch. Dazu gehören Variationen der Schreibweise nicht nur von verschiedenen Schreibern, sondern auch von jedem einzelnen Schreiber. Die Erscheinungsform eines Wortes kommt in mehreren Varianten vor, die sich in Buchstabenformungen, Wortlänge, Neigung und Dichte der Handschrift unterscheiden. Außerdem ist das visuelle

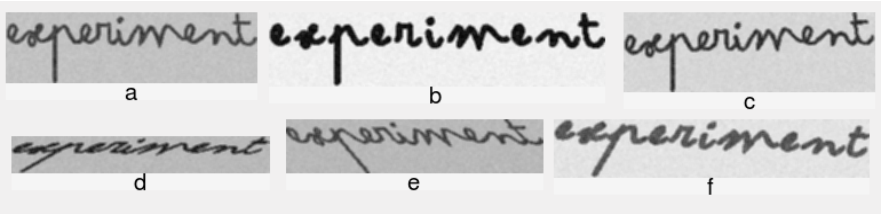

Abbildung 4.1.1: Beispiele für ein gedrucktes Wort, das mit einer Computerschrift, die einer Handschrift ähnelt, generiert und angepasst wird.
(a) Generiertes Wort (b) Änderung der Schriftstärke und des Buchstabenabstandes (c) Rotation gegen den Uhrzeigersinn (d), (e) Neigung der Buchstaben (f) Rotation im Uhrzeigersinn, Änderung von Schriftstärke, Buchstabenabstand und Neigung

Aussehen eines Wortes davon abhängig, ob Buchstaben groß oder klein geschrieben werden oder der erste Buchstabe des Wortes groß geschrieben wird. Abhängig von unterschiedlichen Schreibstilen sind Zeichen bei einem handgeschriebenen Wort getrennt oder miteinander verbunden.

Um synthetische Wortabbilder möglichst realistisch zu gestalten, werden die charakteristischen Merkmale der Handschriften berücksichtigt. Die Erstellung der synthetischen Daten basiert auf dem im Unterabschnitt 3.3 beschriebene Verfahren zur Generierung des synthetischen IIIT-HWS-Datensatzes. Für die Annotation synthetischer Wortabbilder wird das gleiche Vokabular von 10.000 Wörtern und Zahlen angewendet, welches im Unterabschnitt 3.4 für das schwach überwachte Lehrverfahren beim synthetischen HW-SYNTH-Datensatz verwendet wird. Es handelt sich um das Lexikon, das aus dem englischen Hunspell-Wörterbuch ausgewählt wird. Wörter werden zufällig klein, groß und der erste Buchstabe groß geschrieben. Neben der zufälligen Änderung des Buchstabenabstandes und der Schriftstärke werden Buchstaben zusätzlich nach links oder rechts geneigt. Wörter werden leicht im Uhrzeigersinn bzw. gegen den Uhrzeigersinn rotiert. Dadurch wird eine größere Variabilität mit einer Computerschrift erreicht. Die Abbildung 4.1.1 visualisiert Variationen eines mit einer Computerschrift generierten Wortes.

Für den Druck der Textdokumente verwendete *Standard-Schriftfonts* wie zum Beispiel *Arial* und *Times* weisen keine ausgeprägten visuellen Charakteristika von Handschriften auf. Deshalb werden für die Generierung synthetischer Wortabbilder die *Handschrift-Fonts* verwendet, die Handschriften ähneln. Der in dieser Arbeit verwendete Begriff *„Computerschrift“* bezieht sich stets auf die Handschrift-Fonts.

Um den Qualitätszustand der realen Wortabbilder nachzuahmen, werden Pixel im Bildvorder- und -hintergrund als Gaußverteilung um einen Erwartungswert mit einer Varianz modelliert. Diese Werte werden wie im [KJ16] vorgestellten Verfahren automatisch aus den Bilddaten des IAM-Datensatzes geschätzt. Anschließend werden computergenerierte Wortabbilder mit dem Gaußfilter geglättet. Der Gaußfilter wird im Unterabschnitt 2.1.2 näher beschrieben.

4.1.1 *Technische Realisierung*

Die Generierung synthetischer Daten erfolgt in zwei Schritten. Im ersten Schritt wird ein Bild zu jedem Wort aus einem vorgegebenen Vokabular generiert.

Zur Generierung synthetischer Wortabbilder wird *ImageMagick* verwendet. ImageMagick ist ein freies Softwarepaket zur Erstellung und Bearbeitung von Bildern und verfügt über Bibliotheken, die sich in vielen Programmiersprachen verwenden lassen.

Über die Kommandozeile kann der Befehl **convert** von ImageMagick angesprochen werden [mag]. Folgende Parameter sind für die Generierung eines synthetischen Wortabbildes bedeutsam:

- **-label** Wortname
- **-font** Computerschriftname
  Der Computerschriftname wird zufällig aus einer vorgegebenen Menge von verschiedenen Computerschriften ausgewählt.
- **-size** Computerschriftgröße
  Zuerst wird die Größe der Computerschriften auf 52 Pixel für die Generierung der synthetischen Datensätzen eingestellt. Dann wird die Computerschriftgröße auf 75 Pixel erhöht. Dieser Wert wurde empirisch ausgewählt und hat zu besseren Evaluierungsergebnissen geführt. (Siehe Tabelle 5.4.5)
- **-strokewidth** Schriftstärke
  Der Wert wird zufällig aus dem Intervall $[0, 1]$ ausgewählt. Ein positiver Wert verstärkt die Stärke der Schrift im Vergleich zu der ursprünglichen Schriftstärke.
- **-kerning** Zeichenabstand
  Der Wert definiert den Pixelabstand zwischen Buchstaben des generierten Wortabbildes und wird zufällig aus dem Intervall $[0, 7]$ ausgewählt.
- **-annotate** {Xdegrees}x{Ydegrees}
  Xdegrees definiert einen Winkel, um den ein Wort rotiert wird. Positive Werte

bestimmen die Rotierung im Uhrzeigersinn. Negative Werte bestimmen die Rotierung gegen den Uhrzeigersinn. Der Wert wird zufällig aus dem Intervall $[-2, 2]$ ausgewählt.
Ydegrees bestimmt die Neigung einer Schrift. Ein Wert, außer 0, definiert die Neigung der Schrift nach rechts bzw. links, wobei die kleinen Werte einer geringeren und die größeren Werte einer stärkeren Neigung entsprechen. Der Wert wird zufällig aus dem Intervall $[0, 40]$ für die rechte oder aus dem Intervall $[320, 340]$ für die linke Neigung ausgewählt.

Alle vorgeschlagenen Parameterwerte werden nach informellen Experimenten ausgewählt. Bei den Experimenten wird auf die optische Ähnlichkeit synthetischer Wortabbilder geachtet, um charakteristische visuelle Eigenschaften der Handschriften zu repräsentieren.

Die generierten Wortabbilder werden als Grauwertbilder abgespeichert, wobei die Länge und die Höhe der des jeweiligen generierten Wortabbildes entspricht. Auf diese Weise werden mehrere Wortabbilder für eine Wortklasse mit zufälligen Parameterwerten aus einem festgelegten Intervall generiert. Insgesamt werden 134 Computerschriften, die ähnliche Charakteristika wie Handschriften bei modernen und historischen Textdokumenten aufweisen, aus [fona] [fonb] verwendet.

Im zweiten Schritt werden Pixelwerte im Bildvorder- und -hintergrund des computergenerierten Wortabbildes in Anlehnung an [KJ16] als Gaußverteilung um einen Erwartungswert mit einer Varianz modelliert. Anschließend wird das Ergebnisbild auf das Intervall [0...255] normiert und mit der 3x3-Gaußfiltermaske geglättet. Die Gaußfiltermarke wird im Unterabschnitt 2.1.2 erläutert.

4.2 SYNTHETISCHE DATENSÄTZE

Im Verlauf dieser Arbeit werden mehrere synthetische Datensätze generiert und in diesem Abschnitt beschrieben.

4.2.1 *I-SYNTH-750K- und II-SYNTH-750K-Datensätze*

Der synthetische I-SYNTH-750K-Datensatz enthält 10.000 Wortklassen. Jede Wortklasse beinhaltet 75 Wortabbildern. Synthetische Daten werden nach dem im Unterabschnitt 4.1.1 erläuterten Verfahren generiert.

Die Erstellung des II-SYNTH-750K-Datensatzes dient der Verbesserung des I-SYNTH-750K-Datensatzes. Dabei werden Abbilder generiert, dessen Wörter kleingeschrieben

werden oder der erste Buchstabe groß geschrieben wird. In der Regel werden handgeschriebene Wörter ebenfalls klein oder der erste Buchstabe wird groß geschrieben.

Der synthetische II-SYNTH-750K-Datensatz enthält das gleiche Lexikon, die gleiche Anzahl an Wortabbildern, Wortklassen und Wortabbildern pro Klasse wie der synthetische I-SYNTH-750K-Datensatz.

4.2.2 *Größe der synthetischen Trainigsmenge*

Bei diesem Experiment wird untersucht, welchen Effekt die Größe eines synthetischen Datensatzes auf die Leistungsfähigkeit des auf der Basis von synthetischen Wortabbildern vortrainierten Modells bei der Auswertung realer Wortabbilder hat. Dafür werden vier Trainingsmengen SYNTH-100K, SYNTH-400K, SYNTH-700K und SYNTH-1500K erstellt, die jeweils 1.000, 4.000, 7.000 und 15.000 Wortklassen beinhalten. Jede Wortklasse enthält 100 Wortabbilder.

Die Wörter zur Generierung synthetischer Abbilder werden aus den Annotationen des synthetischen IIIT-HWS-Datensatzes [KJ16] entnommen. Die ausgewählte Annotationsmenge beinhaltet 90.000 Wörter und Zahlen, die in [KJ16] aus dem englischen Hunspell-Wörterbuch ausgewählt werden. Um ein generisches Modell zu erzeugen, wird auf die Verteilung der PHOC-Attribute (siehe Unterabschnitt 3.1) geachtet. Damit eine möglichst gleiche Verteilung der PHOC-Attribute erreicht wird, wird für die Auswahl der Wörter folgendes Verfahren definiert:

Sei v ein Vektor, der für jedes PHOC-Attribut die Anzahl von Wörtern beschreibt, die durch das jeweilige Zeichen repräsentiert werden. Dabei werden für jedes Wort und jede Iteration Kosten berechnet. Folgender Algorithmus wird für die Berechnung der Kosten für jedes Wort eines einzelnen Schrittes festgelegt:

1. Skaliere den aktuellen Vektor v durch die Anwendung der natürlichen Exponentialfunktion, damit in den nächsten Schritten die Wörter niedrigere Kosten haben, deren PHOC-Attribute möglichst weniger belegte Attribute in dem Vektor v füllen würden.

2. Multipliziere den entstandenen Vektor mit dem binären PHOC-Vektor für das Wort.

3. Bilde eine Summe aus den Elementen des resultierenden Vektors.

4. Normalisiere das Ergebnis durch eine Division mit der Anzahl der repräsentierten Attribute in dem binären PHOC-Vektor, um auszuschließen, dass nur die

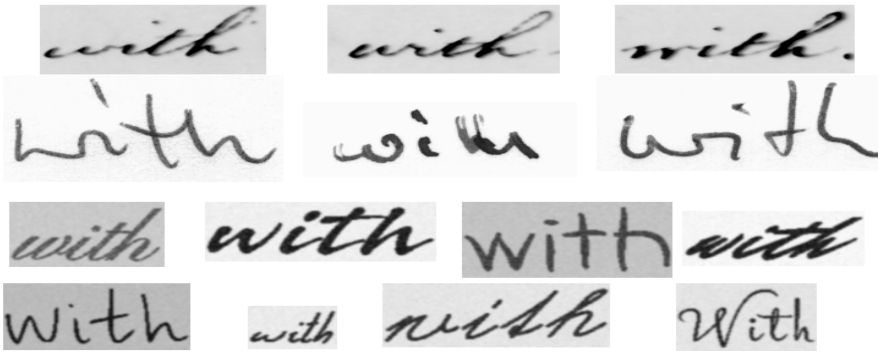

Abbildung 4.2.1: Visuelle Unterschiede zwischen Abbildern für ein textuelles Wort. Die zwei oberen Zeilen sind reale Wortabbilder, wobei die erste Zeile die Variationen der Schreibweise von einem Schreiber und die zweite Zeile die Variationen der Schreibweisen von mehreren Schreibern repräsentiert. Die Bilder sind aus [GW] und [IAM] entnommen. Die zwei unteren Zeilen sind synthetische Wortabbilder.

kürzesten Wörter ausgewählt werden. Das Resultat sind die Kosten für das Wort in der aktuellen Iteration.

5. Wähle die Wörter aus, die die niedrigsten Kosten haben.
6. Wähle aus der Menge der ausgewählten Wörter zufällig ein Wort **W** aus.
7. Bilde einen neuen Vektor v' durch die Summe von Vektor v und dem binären PHOC-Vektor für das ausgewählte Wort **W**.
8. Entferne das Wort **W** aus der Menge der möglichen Wörter und füge es zu der Menge der Wortklassen hinzu.
9. Wiederhole die Schritte, bis die gewünschte Anzahl an Wortklassen erreicht ist.

4.3 DEFORMIERUNG SYNTHETISCHER DATEN

Eine große Herausforderung bei der Generierung synthetischer Daten ist es, die realen Wortabbilder nachzubilden. Ein von Hand geschriebenes Wort und ein in Computerschrift geschriebenes Wort weisen deutliche Unterschiede auf. Ein handgeschriebenes Wort wird selbst vom demselben Schreiber selten mehrmals identisch geschrieben. Im Gegensatz dazu ist ein mehrmals in Computerschrift geschriebenes Wort immer identisch bezüglich der Länge und der Höhe des Wortes und der einzelnen Zeichen,

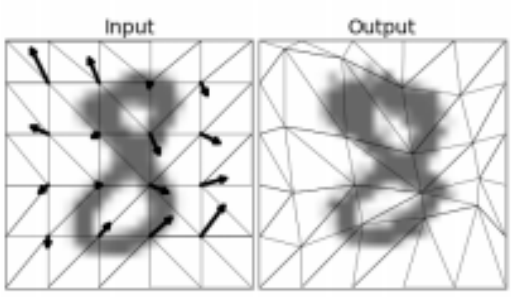

Abbildung 4.3.1: Visualisierung elastischer Transformationen. Rechts ist das Eingabebild. Die Größe der Gitter bestimmt die Granularität der elastischen Transformationen. Die Länge der Pfeile zeigt die Stärke der Verschiebung innerhalb der Gitter. Links ist das elastisch transformierte Ausgabebild. Das Bild wurde aus [BSH17] entnommen.

aus denen es besteht. In der Abbildung 4.2.1 sind mehrere Beispiele für ein Wort zu sehen, die von einem Schreiber, mehreren Schreibern und mit verschiedenen einer Handschrift ähnelnden Computerschriften generiert werden. Durch eine elastische Transformation synthetischer Bilddaten können die Unterschiede von charakteristischen visuellen Merkmalen zwischen synthetischen und realen Wortabbilder verringert werden. Das Prinzip der elastischen Transformation wird in dem Unterabschnitt 4.3.1 beschrieben.

Weitere Unterschiede zwischen realen und synthetischen Wortabbildern sind oft eine verblasste Tinte und ein schlechter Zustand der Textseiten, was insbesondere für historische handgeschriebene Textdokumente charakteristisch ist. Der Bildhintergrund handgeschriebener Wortabbilder kann unterschiedliche Arten von Verschmutzungen wie zum Beispiel Tintenflecken oder die Beschriftung der Rückseite beinhalten. Die Erstellung der synthetischen Daten mit Störungen stellt eine weitere Möglichkeit dar, die Differenzen von charakteristischen visuellen Merkmalen zwischen synthetischen und realen Wortabbilder zu verringern. Die Modellierung der Störungen wird im Unterabschnitt 4.3.2 beschrieben.

4.3.1 *Elastische Transformationen*

Um die oben genannten Unterschiede zwischen realen und synthetischen Wortabbildern zu verringern und synthetische Bilddaten etwas realistischer zu repräsentieren, bietet sich die elastische Transformation synthetischer Wortabbilder an. Im Gegensatz

Abbildung 4.3.2: Das erste Wortabbild in jeder Zeile ist das synthetische Eingabebild. Die anderen Wortabbilder sind elastisch deformiert.

zur affinen Transformation, bei der jede gerade Linie des Eingabebildes auf eine grade Linie im Ausgabebild abgebildet und die Parallelität behalten wird, können elastische Transformationen eine gerade Linie des Eingabebildes auf eine Kurve im Ausgabebild abbilden [Fra09, S. 16].

Die zugrunde liegende Idee der elastischen Transformation besteht darin, leichte Deformierungen von Zeichen, die bei einem Schreibprozess durch Handbewegung entstehen, bei synthetischen Wortabbildern nachzubilden. Zufällige elastische Transformationen synthetischer Daten werden mit *Augmentator* [BSH17] implementiert. Dabei wird die Granularität elastischer Transformationen über die Gittergröße und die Stärke der Verschiebung innerhalb der Gitter reguliert. In der Abbildung 4.3.1 ist ein Beispiel einer elastisch deformierten Ziffer zu sehen.

SYNTH-700K-elastic-Datensatz

Der synthetische SYNTH-700K-elastic-Datensatz enthält 7.000 Wortklassen. Jede Wortklasse besteht aus 100 synthetischen Wortabbildern. Davon werden 75 Prozent, also 525.000, synthetische Wortabbilder elastisch deformiert.

Zuerst werden 25 Wortabbilder pro Wortklasse nach dem in Unterabschnitt 4.1.1 beschriebenen Verfahren generiert. Dann werden synthetische Wortabbilder mit dem *Augmentator* elastisch transformiert. In der Abbildung 4.3.2 sind Beispiele aus dem synthetischen SYNTH-700K-elastic-Datensatz zu sehen.

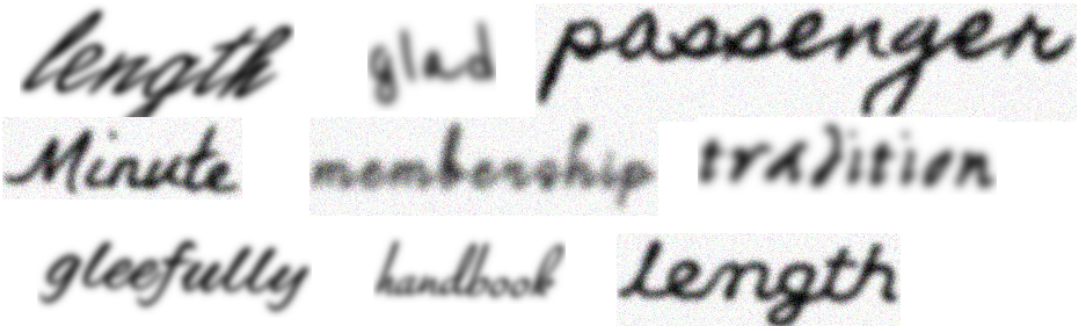

Abbildung 4.3.3: Synthetische Wortabbilder mit Störungen.

4.3.2 *Modellierung der Störungen*

Die nächste Änderung, um synthetische Wortabbilder an reale Wortabbilder anzugleichen, ist das Hinzufügen von zwei zusätzlichen Parametern, die die Qualität der synthetischen Wortabbilder verschlechtern:

- **-blur** Tiefpass
  Das Bild wird mit einer Gaußfiltermaske geglättet (vgl. 2.1.2). Nach der Glättung wird die Computerschrift eines synthetischen Wortabbildes unschärfer. Die Stärke der Unschärfe wird mit der Standardabweichung σ reguliert. Der σ-Wert wird zufällig aus dem Intervall $[2, 4]$ ausgewählt. Je größer der σ-Wert ist, desto unschärfer wird das Wortabbild.

- **+noise** Gaußsches Rauschen
  Pixelwerte eines computergenerierten Bildes werden gleichmäßig nach *der Gaußschen Normalverteilung*, deren Dichtefunktion glockenförmig verläuft (siehe Abb. 2.1.2), verfälscht. Der Parameter **noise** darf nicht vor dem Parameter **blur** angewendet werden, da sonst das definierte Rauschen aus dem synthetischen Wortabbild durch das Weichzeichnen entfernt wird.

I-SYNTH-700K-distorted-Datensatz

Der synthetische I-SYNTH-700K-distorted-Datensatz enthält 7.000 Wortklassen. Jede Wortklasse besteht aus 100 synthetischen Wortabbildern. Davon werden 75 Prozent, also 525.000, synthetische Wortabbilder mit Störungen generiert.

II-SYNTH-700K-distorted-Datensatz

Der synthetische II-SYNTH-700K-distorted-Datensatz enthält 7.000 Wortklassen. Jede Wortklasse besteht aus 100 synthetischen Wortabbildern. Davon wird 20 Prozent, also 140.000, synthetische Wortabbilder mit Störungen generiert.

In der Abbildung 4.3.3 sind Beispiele aus den synthetischen I-SYNTH-700K-distorted- und II-SYNTH-700K-distorted-Datensätzen zu sehen.

SYNTH-700K-elastic-distorted-Datensatz

Der synthetische SYNTH-700K-elastic-Datensatz enthält 7.000 Wortklassen. Jede Wortklasse besteht aus 100 synthetischen Wortabbildern. Davon werden 490.000 synthetische Wortabbilder mit elastischen Transformationen und Störungen generiert.

5 EXPERIMENTE

In diesem Kapitel werden Experimente beschrieben, die auf der Grundlage der synthetischen Abbilder (Kapitel 4) und der drei realen Datensätze (Abschnitt 5.1) durchgeführt werden. Die Details zum Training werden im Abschnitt 5.2 beschrieben. Die Evaluierung der trainierten Modelle erfolgte nach dem im Abschnitt 5.3 erläuterten Protokoll. Zunächst wird das Baseline-Verfahren im Abschnitt 5.4.1 nachgebildet, um die Korrektheit des in dieser Arbeit implementierten Verfahrens festzustellen. Die in den Abschnitten 5.4.2 und 5.4.3 beschriebenen Experimente dienen der Einstellung einiger Trainingsparameter und der Korrektheit der implementierten Methode zur Generierung synthetischer Daten. Im Abschnitt 5.4.4 wird untersucht, wie sich die Größe des synthetischen Datensatzes auf die Leistungsfähigkeit der Word-Spotting-Modelle auswirkt. Der Effekt von Deformierungen synthetischer und realer Daten wird in den Abschnitten 5.4.5 bzw. 5.4.6 analysiert. In Abschnitt 5.4.7 werden zwei Auswahlstrategien der realen Trainingsdaten diskutiert. Anschließend werden die erhaltenen Ergebnisse mit Ansätzen des Standes der Technik im Abschnitt 5.4.8 verglichen.

5.1 DATENSÄTZE

Die Durchführung der Experimente erfolgt auf der Basis der in Kapitel 4 vorgestellten synthetischen Datensätze. Für die Feinabstimmung werden drei Datensätze mit handgeschriebenen Wortabbildern verwendet, die im Folgenden beschriebenen werden.

Der *George-Washington-(GW)*-Datensatz [GW] ist ein historischer handgeschriebener Datensatz. Er enthält Auszüge über 20 Textseiten aus einer digitalisierten Sammlung von Briefen, die von George Washington und seinen Assistenten geschrieben worden sind. In dem GW-Datensatz sind insgesamt 4860 Wortabbilder annotiert. Da die visuellen Erscheinungsformen der annotieren Wörter relativ ähnliche Charakteristika aufweisen, wird dieser Datensatz als „geschrieben von einem Schreiber“ angesehen [AGFV14]. Aufgrund des Fehlens der offiziellen Einteilung von Daten in eine Trainings- und eine Testmenge wird eine vier-fach Kreuzvalidierung verwendet [SF18], [AGFV14], [GSF17]. Dafür werden die Daten in vier Mengen partitioniert, wie das in [AGFV14] beschrieben ist. Aus den vier Mengen werden vier Datensätze

erstellt, die jeweils drei unterschiedliche Mengen für das Training und eine Menge für das Testen beinhalten. Die PHOC-Netze werden viermal mit jeweils einer der vier unterschiedlichen Trainingsmengen trainiert und mit der entsprechenden Testmenge getestet. Die durchschnittlichen Ergebnisse der Evaluierung werden in die Tabellen (siehe Unterabschnitt 5.4) eingetragen. Aufgrund einer niedrigen Variabilität in der Schreibweise und eines guten Zustandes der Textseiten stellt der GW-Datensatz keine großen Schwierigkeiten für die Suche mit dem Word-Spotting-Verfahren dar.

Der *IAM*-Datensatz [IAM] ist ein moderner handgeschriebener Datensatz. Er enthält 1.539 digitalisierte englische Textseiten, die von 657 unterschiedlichen Schreibern geschrieben worden sind. Insgesamt sind 115.320 Wortabbilder annotiert. Es wird die offizielle Partitionierung der Daten für das schreiberunabhängige Erkennungsszenario der Textzeilen verwendet. Durch die Kombination der Trainings- und Validierungsmengen ergaben sich 60.453 annotierte Wortabbilder für das Training und 13.752 annotierte Wortabbilder für das Testen. Der IAM-Datensatz zeichnet sich durch eine hohe Variabilität in der Schreibweise aus. Die Abbilder der gleichen Wörter weisen unterschiedliche charakteristische visuelle Eigenschaften auf und stellen somit eine Herausforderung für die Word-Spotting-Aufgaben dar [SF16].

Der *Esposalles*-Datensatz [ES] ist ein historischer handgeschriebener Datensatz. Er enthält eine digitalisierte Sammlung von Heiratsurkunden, die aus dem Archiv der Kathedrale von Barcelona stammen. Hier wird die offizielle Partitionierung der Daten mit 32.052 annotierten Wortabbildern für das Training und 13.048 annotierten Wortabbildern für das Testen verwendet. Die Textseiten der Heiratsurkunden befinden sich in einem schlechten Zustand. Sie sind mit Tintenflecken verschmiert und die Beschriftung der Rückseite ist zu erkennen. Die oben genannten charakteristischen Eigenschaften des Esposalles-Datensatzes erschweren die Word-Spotting-Aufgaben [SF16].

5.2 TRAININGSDETAILS

Vor dem Training und der Evaluierung von PHOC-Netzen wird aus der Annotation jedes Wortabbildes ein PHOC-Vektor bestimmt. Für das Training werden die gleichen Parameter wie in [GSF17] und [SF18] für das PHOC-Netz-Training mit der *Binary-Cross-Entropy*-Kostenfunktion angewandt. Die Kostenfunktion wird mittels des stochastischen Gradientenverfahrens mit dem Momentum gleich 0.9 und dem Gewichtsverfall gleich $5 \cdot 10^{-5}$ optimiert. Jeder Batch enthält 10 Trainingsbeispiele.

Das Training aller Modelle erfolgt in zwei Schritten. In dem ersten Schritt wird ein PHOC-Netz anhand eines synthetischen Datensatzes von Grund auf vortrainiert.

Dabei werden die in Kapitel 4 implementierten synthetischen Datensätze verwendet. In dem zweiten Schritt wird das auf der Basis synthetischer Daten vortrainierte Modell anhand einer Menge von realen Trainingsbeispielen nachtrainiert. Für jeden realen Datensatz werden zufällig jeweils vier Trainingsmengen mit 100, 250, 500 und 1000 Trainingsbeispielen aus den jeweiligen realen Trainingsmengen ausgewählt. Die Anzahl der ausgewählten realen Trainingsbeispiele wird durch geometrische Transformationen nach [SF16] vergrößert.

Der PHOC-Vektor wird für die Ebenen $2, 3, 4$ und 5 gebildet. Die PHOC-Repräsentationen sind Schriftzeichen, die aus der Trainingsmenge jedes realen Datensatzes abgeleitet werden. Dabei wird kein Unterschied zwischen großen und kleinen Buchstaben gemacht. Modelle, die anhand synthetischer Daten vortrainiert werden, lernen die gleichen PHOC-Embeddings wie nachtrainierte Modelle.

Das Vortraining erfolgt über 100.000 Iterationen. PHOC-Netze werden mit den ersten 70.000 Iterationen mit einer Lernrate von 10^{-4} und mit den letzten 30.000 Iterationen mit einer Lernrate von 10^{-5} vortrainiert. Angelehnt an [HZRS15] werden die Gewichte zufällig mit einer Normalverteilung mit dem Mittelwert 0 und einer Standabweichung von $\frac{2}{n}$ initialisiert. Wobei n die Anzahl von lernbaren Parametern der jeweiligen Schicht ist. Das Bias wird mit 0 initialisiert.

Das Nachtraining erfolgt über 40.000 Iterationen. Die gelernten Parameter des vortrainierten Modells werden übernommen und mittels des stochastischen Gradientenverfahrens optimiert. Die Lernrate wird empiristisch in Abhängigkeit vom betrachteten realen Datensatz und der Anzahl der realen Trainingsbeispiele ausgewählt (siehe Unterabschnitt 5.4.2). Die Übersicht zur Auswahl von Lernraten ist in der Tabelle 5.2.1 gegeben.

Alle experimentellen Untersuchungen erfolgten auf der Grafikkarte Nvidia GeForce GTX 1080 GPU.

5.3 WORD-SPOTTING-PROTOKOLL

Der in dieser Arbeit implementierte segmentierungsbasierte Word-Spotting-Ansatz hat die Aufgabe, nach allen einer Suchanfrage relevanten Wortabbildern in der Testmenge von annotierten Wortabbildern des entsprechenden Datensatzes zu suchen. Als Antwort wird eine nach der Relevanz mit der Anfrage sortierte Retrieval-Liste mit allen Wortabbildern der Testmenge zurückgegeben. Dabei ist ein Wortabbild der Retrieval-Liste *relevant*, wenn es das angefragte Wort repräsentiert, andernfalls ist es *irrelevant*.

Anzahl der realen Trainingsbeispiele	GW	IAM, Esposalles
100	$\eta = 10^{-6}$	$\eta = 10^{-6}$
250	$\eta = 10^{-5}$	$\eta = 10^{-6}$
500	$\eta = 10^{-5}$	$\eta = 5 \cdot 10^{-6}$
1000	$\eta = 10^{-5}$	$\eta = 5 \cdot 10^{-6}$

Tabelle 5.2.1: Angewendete Lernraten für das Nachtraining mit unterschiedlich großen Mengen von handgeschriebenen Trainingsbeispielen.

Alle Modelle werden für QbE- und QbS-Szenarien evaluiert, wie in [GSF17]. Für das QbS-Szenario wird die Annotation jeder in der Testmenge vorhandenen Wortklasse als Anfrage verwendet. Für das QbE-Szenario wird ein Wortabbild als Anfrage verwendet, wenn mindestens ein anderes Wortabbild innerhalb derselben Wortklasse vorhanden ist.

Im Gegensatz zu einem QbS-Szenario, für das ein PHOC-Vektor direkt aus einer Zeichenkette der Anfrage bestimmt wird, erfolgt die Berechnung eines PHOC-Vektors für ein QbE-Szenario mit Hilfe eines trainierten Modells.

Der PHOC-Vektor der Anfrage wird mit jedem PHOC-Vektor des Wortabbildes der Testmenge verglichen. Die Ähnlichkeitsberechnung zweier Vektoren erfolgt anschließend durch das Kosinus-Ähnlichkeitsmaß [SF18].

Um die Güte der Retrieval-Liste zu bestimmen, wird *mean Average Precision* (mAP) über alle Anfragen einer Evaluierung als Bewertungsmaß berechnet. Die Average Precision einer Anfrage ist definiert als

$$AP = \frac{\sum_{k=1}^{n} \mathrm{Precision}(k) \cdot \mathrm{rel}(k)}{\text{Anzahl relevanter Wortabbilder,}} \tag{5.3.1}$$

wobei n die Länge der Retrieval-Liste ist, Precision(k) das Verhältnis des Treffers zu der k-Stelle definiert und rel(k) die binäre Relevanz des k-Elements ist. rel(k) ist gleich 1, wenn das Element an der Stelle k ein zu der Anfrage relevantes Wortabbild ist, und sonst gleich 0 [SF18].

5.4 EXPERIMENTE UND ERGEBNISSE

Alle Experimente sind nach demselben Schema aufgebaut. Zunächst werden Modelle anhand eines synthetischen Datensatzes vortrainiert und auf Testdaten des jeweiligen

Verfahren	Anzahl der realen Trainings-beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
Baseline	0	39.89	48.92	26.21	36.57	34.92	10.30
		40.65	**49.57**	**28.12**	**37.12**	**39.15**	**11.32**
	100	83.05	86.69	38.45	56.47	89.67	71.15
		84.88	**85.95**	**38.58**	**53.60**	**90.27**	**67.87**
	250	90.76	92.39	43.78	60.90	94.06	82.43
		90.94	**89.78**	**48.24**	**61.66**	**93.55**	**74.98**
	500	93.86	94.82	52.41	68.33	95.14	85.42
		93.60	**93.13**	**51.66**	**67.63**	**94.43**	**79.98**
	1000	95.74	96.59	57.79	74.03	95.67	83.16
		94.52	**95.79**	**53.11**	**73.00**	**94.89**	**81.28**

Tabelle 5.4.1: Vergleich der Baseline-Implementierung mit Angaben in [GSF17]. Die fett markierten Angaben stellen die Ergebnisse nach der Rekonstruktion des Baseline-Verfahrens dar. Die nicht markierten Angaben stammen aus [GSF17]. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben.

realen Datensatzes evaluiert. Danach werden die vortrainierten Modelle mit relativ kleinen Mengen von realen Trainingsbeispielen des jeweiligen Datensatzes nachtrainiert. Anschließend werden die nachtrainierten Modelle auf Testdaten des jeweiligen realen Datensatzes evaluiert [GSF17].

5.4.1 *Baseline-Verfahren*

Das Ziel dieses Experimentes ist es, die Arbeit von Gurjar et al. zu rekonstruieren und die Korrektheit des implementierten Verfahrens festzustellen. Das Nachtraining des Modelles erfolgt anhand des synthetischen HW-SYNTH-Datensatzes. Der HW-SYNTH-Datensatz wird im Abschnitt 3.4 vorgestellt. Insbesondere ist die Evaluierung nachtrainierter Modelle interessant, da die Mengen von realen Trainingsbeispielen zufällig für das Nachtraining ausgewählt werden. Es ist eher unwahrscheinlich, dass identische Trainingsbeispiele bei der Arbeit von Gurjar et al. und bei der vorliegenden Arbeit verwendet werden.

PHOC-Embeddings	Anzahl der realen Trainingsbeispiele	IAM		Esposalles	
		QbE	QbS	QbE	QbS
synth	0	**25.24**	**36.64**	38.69	16.21
real	0	24.54	35.12	**40.49**	**19.04**
synth	100	37.84	52.37	**91.22**	75.12
real	100	**40.88**	**60.94**	90.99	**75.63**

Tabelle 5.4.2: Vergleich der Leistungsfähigkeit von Modellen mit den verschiedenen PHOC-Embeddings. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben.

Die Modelle werden mit den in [GSF17] empfohlenen Trainingsdetails implementiert. Einige Trainingsparameter unterscheiden sich von den in Abschnitt 5.2 beschriebenen Trainingsparametern. In der Arbeit von Gurjar et al. werden 80.000 Iterationen für das Vortraining verwendet. Für das Nachtraining wird die Lernrate 10^{-5} verwendet. Als PHOC-Repräsentationen werden nur die Zeichen des lateinischen Alphabets verwendet.

Die Tabelle 5.4.1 zeigt die Ergebnisse der Auswertung des Baseline-Verfahrens und die Ergebnisse aus [GSF17]. Die Abweichungen der Ergebnisse der GW- und IAM-Datensätze sind nicht stark ausgeprägt. Größere Unterschiede sind bei dem QbS-Szenario für die Evaluierung der nachtrainierten Modelle mit dem Esposalles-Datensatzes zu beobachten. Die mAP-Werte liegen bei $4\% \pm 2\%$. Das kann an der Auflösung der synthetischen Bilddaten liegen, da die synthetischen Wortabbilder vor dem Vortraining zufällig skaliert werden (siehe Unterabschnitt 3.4). Ein anderer Grund könnte in repräsentativen Stichproben der realen Datenbasis für das Nachtraining liegen, da die handgeschriebenen Wortabbilder zufällig ausgewählt werden.

5.4.2 *I-SYNTH-750K-Datensatz*

Das Ziel des Experimentes ist die Korrektheit der in Kapitel 4 angewendeten Methode zur Generierung synthetischer Daten festzustellen. Des Weiteren werden einige Trainingsparameter eingestellt, mit denen für die in der vorliegenden Arbeit durchgeführten Experimenten bessere Ergebnisse erreicht werden können. Da der Lernerfolg des PHOC-Netzes von vielen Parametern abhängig ist, werden einige Parameter geändert und analysiert. Zu diesem Zweck werden die Modelle anhand des syntheti-

Lernrate	Anzahl der realen Trainings-beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
$\eta = 10^{-5}$	100	87.35	90.05	40.72	54.51	91.99	72.13
$\eta = 10^{-6}$	100	88.34	91.03	40.88	60.94	90.99	75.63
$\eta = 10^{-5}$	250	92.57	92.79	-*	-*	-*	-*
$\eta = 10^{-6}$	250	90.55	92.60	46.53	66.82	92.06	77.36
$\eta = 10^{-5}$	500	94.88	95.28	52.24	70.93	95.00	80.52
$\eta = 10^{-6}$	500	-*	-*	48.81	68.99	92.77	81.42
$\eta = 5 \cdot 10^{-6}$	500	-*	-*	52.04	70.96	94.65	81.48

Tabelle 5.4.3: Vergleich der Ergebnisse von mit unterschiedlichen Lernraten nachtrainierten Modellen. Für die -*-Szenarien wird kein Vergleich durchgeführt, da die erhaltenen mAP-Werte der nachtrainierten Modelle nur mit einer von den vorgegebenen Lernrate akzeptabel sind. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben.

schen I-SYNTH-750K-Datensatzes vortrainiert. Der I-SYNTH-750K-Datensatz wird im Abschnitt 4.2.1 vorgestellt.

Um eine bessere Leistungsfähigkeit zu erzielen, werden folgende Parameter geändert:

PHOC-Repräsentationen: Eine bedeutende Wirkung zeigte die Auswahl von Zeichen für die PHOC-Repräsentationen. Die Leistungsfähigkeit von Modellen, für die PHOC-Repräsentationen aus den Zeichen des Alphabets der entsprechenden handgeschriebenen Trainingsmenge berechnet werden, kann verbessert werden. Die Tabelle 5.4.2 zeigt die Unterschiede zwischen Modellen, für die PHOC-Repräsentationen aus den Zeichen der synthetischen und handgeschriebenen Trainingsmenge berechnet werden. Es wird kein Vergleich für den GW-Datensatz vorgenommen, da die GW-Trainingsmenge die gleichen Zeichen wie eine synthetische Trainingsmenge enthält.

Eine Verbesserung um 3% wird für das QbS-Szenario des Esposalles-Datensatzes bei PHOC-Embeddings-Synth des vortrainierten Modells erreicht. Trotz eines Unterschieds um 1% bei PHOC-Embeddings-Synth des vortrainierten Modells im Vergleich zu PHOC-Embeddings-Real für den beiden QbE- und QbS-Szenarien auf dem IAM-Datensatz wird eine Steigerung um 3% für das QbE-Szenario und

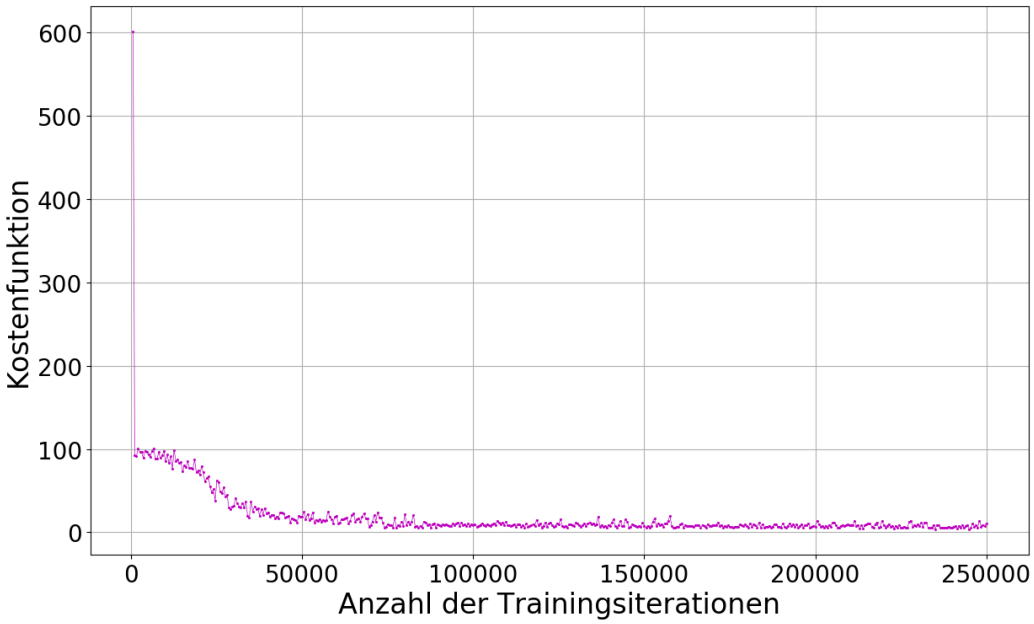

Abbildung 5.4.1: Der Verlauf der Kostenfunktion beim Training des PHOC-Netzes anhand des synthetischen I-SYNTH-750K-Datensatzes.

um 8% für das QbS-Szenario bei PHOC-Embeddings-Real des nachtrainierten Modells erreicht.

Lernrate: Eine wesentliche Voraussetzung für den Lernerfolg eines Faltungsnetzes ist die richtige Auswahl der Lernrate. Um bessere mAP-Ergebnisse zu erzielen, wird die Anpassung der vortrainierten Modelle mit unterschiedlichen Lernraten zwischen 10^{-5} und 10^{-6} durchgeführt.

Die auf der Basis des synthetischen I-SYNTH-750K-Datensatzes vortrainierten Modelle werden mit unterschiedlichen Lernraten nachtrainiert. In der Tabelle 5.4.3 werden die Ergebnisse der Evaluierung einiger ausgewählten Modelle verglichen, die mit unterschiedlichen Lernraten nachtrainiert wurden. Eine detaillierte Übersicht zur Auswahl der Lernrate für das Nachtraining mit unterschiedlich großen Mengen von handgeschriebenen Trainingsbeispielen ist in 5.2.1 gegeben.

Die Experimente haben gezeigt, dass eine kleinere Lernrate für das Nachtraining eines Modells mit einer relativ kleinen Anzahl von realen Trainingsbeispielen

sinnvoller ist. Bei der Erhöhung der Anzahl von realen Trainingsdaten werden bessere Ergebnisse mit einer größeren Lernrate erreicht.

Anzahl der Trainingsiterationen: Um den Lernerfolg des PHOC-Netzes anhand synthetischer Daten zu analysieren, wird das Modell anhand des synthetischen I-SYNTH-750K-Datensatzes über 250.000 Iterationen mit einer Lernrate von 10^{-4} trainiert. Die Lernrate wird nach 70.000 Iterationen auf 10^{-5} reduziert. Die Abbildung 5.4.1 zeigt den Verlauf der Kostenfunktion. Da die Kostenfunktion zwischen 85.000 und 120.000 Iterationen ohne wesentliche Verbesserungen verläuft, werden 100.000 Iterationen für das Vortraining aller Modelle anhand synthetischer Daten im Rahmen dieser Arbeit ausgewählt.

In der Tabelle 5.4.4 sind Ergebnisse der ausgewerteten Modelle zusammengefasst. Die Leistungsfähigkeit der vortrainierten und nachtrainierten Modelle hat sich für einige QbE- und QbS-Szenarien im Vergleich zum Baseline-Verfahren durch die oben beschriebenen Parameterkonfigurationen verbessert. Bei der Auswertung vortrainierter Modelle sind die mAP-Werte für die beiden QbE- und QbS-Szenarien um 6% bzw. um 5% auf dem GW-Datensatz höher. Für den Esposalles-Datensatz ist der mAP-Wert für das QbS-Szenario um 9% höhrer. Für den IAM-Datensatz sind Ergebnisse bei vortrainierten Modellen im Vergleich zum Baseline-Verfahren um 2% niedriger. Dagegen wird eine deutliche Verbesserung bei der Auswertung nachtrainierter Modelle erzielt. Beim Experiment I-SYNTH-750K mit 100 handgeschriebenen Wortabbildern werden die mAP-Werte für das QbE-Szenario um 5% (GW), 8% (IAM), 3% (Esposalles) und für das QbS-Szenario um 4% (GW), 8% (IAM), 4% (Esposalles) verbessert. Bei einer Vergrößerung der Anzahl von handgeschriebenen Wörtern auf 500 und 1000 werden die ähnlichen mAP-Werte zum Baseline-Verfahren auf den GW und Esposalles Datensätzen erreicht. Außerdem wird eine Verbesserung der mAP-Werte für das QbE-Szenario um 4% bzw. 3% und für das QbS-Szenario um 5% bzw. 4% auf dem IAM-Datensatz erzielt.

5.4.3 *II-SYNTH-750K-Datensatz*

Das Ziel dieses Experiments ist eine Verbesserung gegenüber den im Unterabschnitt 5.4.2 erhaltenen Ergebnisse. Dabei wird eine Reihe von weiteren synthetischen Datensätzen generiert und analysiert. Der synthetische Datensatz mit der besten Leistungsfähigkeit für die Auswertung der realen Daten wird als II-SYNTH-750K-Datensatz bezeichnet und im Abschnitt 4.2.1 vorgestellt. Im Gegensatz zu dem I-SYNTH-750K-Datensatz werden Wörter des synthetischen II-SYNTH-750K-Datensatzes, die aus-

Verfahren	Anzahl der realen Trainings-beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
I-SYNTH-750K	0	46.39	53.42	24.54	35.12	40.69	19.04
	100	88.34	91.03	40.88	60.94	90.99	75.63
	250	92.57	92.79	46.53	66.82	92.06	77.36
	500	94.88	95.28	52.04	70.96	94.65	81.48
	1000	95.87	96.86	57.05	75.29	94.55	83.91

Tabelle 5.4.4: Die Resultate der Experimente für QbE und QbS sind in mAP % eingegeben.

schließlich klein geschrieben werden oder bei denen der erste Buchstabe groß geschrieben wird, abgebildet. Außerdem wird die Schriftgröße bei der Generierung synthetischer Abbilder geändert (vgl. Abschnitt 4.1.1).

Da Untersuchungen mit einer kleinen Anzahl von manuell annotierten Wortabbildern für das Ziel der vorliegenden Arbeit von Interesse sind, werden die Modelle bei weiteren Experimenten nur mit Mengen bestehend aus 100 und 250 realen Trainingsbeispielen nachtrainiert. Bei der Auswertung der vortrainierten Modelle in Bezug auf die Testmengen konnte eine Verbesserung der Ergebnisse erreicht werden. Für das QbS-Szenario sind haben sich Ergebnisse um 5% (GW), 13% (IAM) und 8% (Esposalles) verbessert. Für das QbE-Szenario verbesserten sich die Ergebnisse um 6% (IAM) und 18% (Esposalles). Die Verbesserung der Leistungsfähigkeit der nachtrainierten Modelle wird mit dem IAM-Datensatz erreicht. Beim Experiment II-SYNTH-750K mit 100 und 250 handgeschriebenen Wortabbildern verbesserten sich die mAP-Werte für das QbE-Szenario um 6% bzw. um 8% und für das QbS-Szenario um 4% bzw. um 1% auf dem IAM-Datensatz.

Trotz einer wesentlichen Verbesserung der Ergebnisse von vortrainierten Modellen wird keine Verbesserung der Ergebnisse bei nachtrainierten Modellen mit GW- und Esposalles-Datensätzen erreicht. Es wird vermutet, dass die Erkennungsdifferenz zwischen den I-SYNTH-750K-Datensatz und II-SYNTH-750K-Datensatz bei vortrainierten Modellen zu gering ist und deshalb durch das Feintuning stärker überlagert wird.

5.4.4 *Größe der synthetischen Trainingsmenge*

Mit dem in [KJ16] vorgestellten Verfahren zur Generierung synthetischer Daten ist es möglich, beliebig viele Trainingsbeispiele zu generieren. Damit kann untersucht

Verfahren	Anzahl der realen Trainings-beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
II-SYNTH-750K	0	44.12	58.85	30.65	48.72	58.49	27.80
	100	88.87	90.58	46.48	64.45	92.49	75.03
	250	92.94	92.74	54.01	67.59	94.21	80.10

Tabelle 5.4.5: Die Resultate der Experimente für QbE und QbS sind in mAP % eingegeben.

werden, welche Auswirkung die Größe eines synthetischen Datensatzes auf die Leistungsfähigkeit des Modells hat. Dafür werden vier unterschiedlich große synthetische Datensätze erstellt, die jeweils 100.000, 400.000, 700.000 und 1.500.000 Wortabbilder enthalten. Die Erstellung der SYNTH-100K-, SYNTH-400K-, SYNTH-700K- und SYNTH-1500K-Datensätze wird im Unterabschnitt 4.2.2 beschrieben.

Das Lexikon der vier Datensätze wird möglichst nach der gleichen PHOC-Attributsverteilung ausgewählt. Die Abbildung 5.4.2 repräsentiert die Häufigkeitsverteilung von PHOC-Attributen in den vier unterschiedlich großen synthetischen Trainingsmengen. Das Häufigkeitsvorkommen von Zeichen des lateinischen Alphabets wird für unterschiedliche Wortregionen bestimmt. Auf der x-Achse sind 504-Dimensionen des PHOC-Vektors definiert, auf der y-Achse die Häufigkeit, mit der ein Attribut in einer bestimmten Wortregion vorkommt.

Die Tabelle 5.4.6 zeigt die Ergebnisse der vor- und nachtrainierten Modelle für die drei realen GW-, IAM- und Esposalles-Datensätze. Im Vergleich mit den 100.000 synthetischen Trainingsbeispielen wird mit den 400.000 synthetischen Trainingsbeispielen eine deutliche Verbesserung der Leistungsfähigkeit von vor- und nachtrainierten Modellen für alle drei Datensätze erreicht. Auch zwischen 400.000 und 700.000 können kleine Verbesserungen bei den IAM- und Esposalles-Datensätzen beobachtet werden. Bei 100 realen Trainingsbeispielen wird eine Verbesserung der mAP-Werte für QbS für den IAM-Datensatz um 4% und für den Esposalles-Datensatz um 5% erreicht.

Bei einer weiteren Vergrößerung der Anzahl von synthetischen Trainingsdaten auf 1.500.000 Wortabbilder sind keine Verbesserungen festzustellen. Für den IAM-Datensatz haben sich die Ergebnisse von QbE und QbS sogar verschlechtert.

Beobachtungen zufolge werden die besten Ergebnisse zwischen 400.000 und 700.000 synthetischen Trainingsbeispiele erzielt.

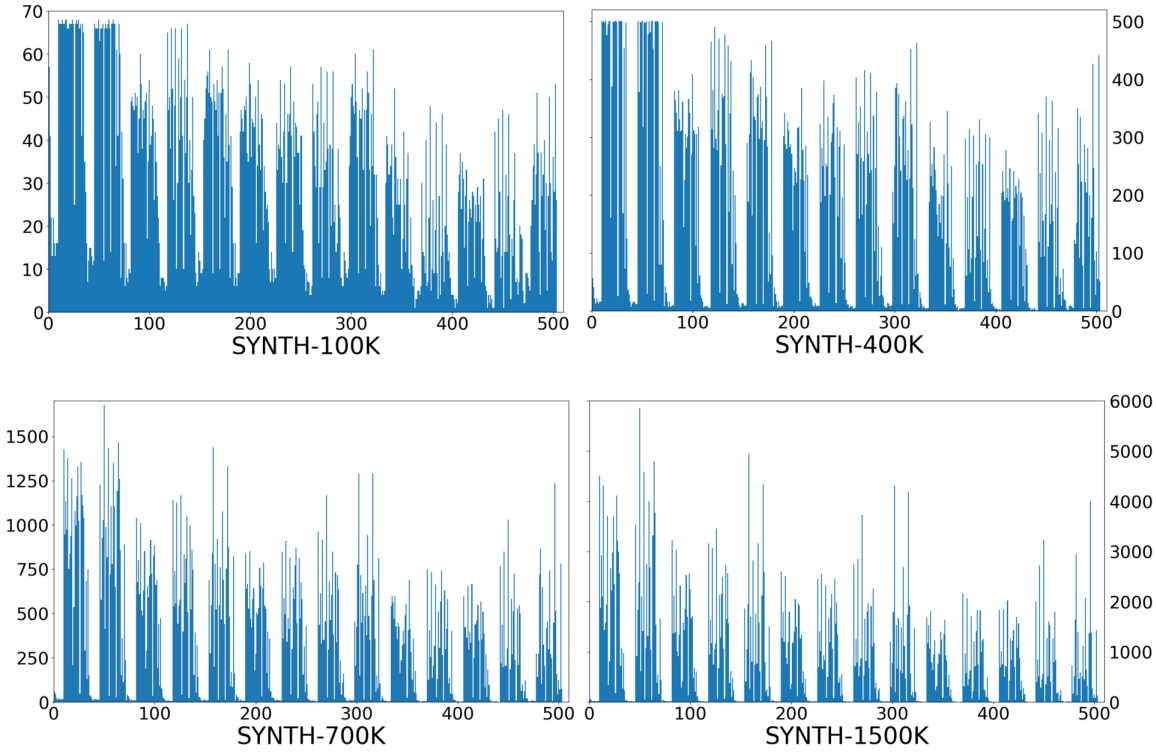

Abbildung 5.4.2: Histogramme der Häufigkeitsverteilung von PHOC-Attributen in den vier unterschiedlichgroßen synthetischen Datensätzen.

5.4.5 *Deformierte synthetische Daten*

Als Ziel der folgenden Experimente wird untersucht, inwiefern sich deformierte synthetische Daten für die realen Anwendungen eignen. Deformierte synthetische Daten sollen visuelle charakteristische Eigenschaften der realen Wortabbilder nachbilden und zu einer verbesserten Leistungsfähigkeit der vortrainierten Modelle führen. Die im 4.3 Abschnitt generierten deformierten synthetischen Datensätze beinhalten jeweils 700.000 Trainingsbeispiele.

SYNTH-700K-elastic-Datensatz

Das Ziel des Experimentes ist die Auswirkung von Hinzufügen der elastisch transformierten synthetischen Daten auf die Leistungsfähigkeit von Modellen zu untersuchen.

Verfahren	Anzahl der realen Trainings-beispielen	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
SYNTH-100K	0	43.14	48.39	30.16	36.45	55.81	20.82
SYNTH-400K	0	**50.57**	60.42	**36.48**	51.84	62.94	31.47
SYNTH-700K	0	48.28	**60.93**	35.48	**54.11**	**62.99**	**32.52**
SYNTH-1500K	0	47.39	60.57	33.03	50.22	60.40	31.86
SYNTH-100K	100	86.82	85.80	44.77	44.99	89.15	53.22
SYNTH-400K	100	90.23	**92.07**	**49.36**	63.04	91.82	70.38
SYNTH-700K	100	89.62	92.03	49.27	**67.12**	92.30	75.63
SYNTH-1500K	100	**90.24**	91.31	44.18	65.93	**92.39**	**77.21**
SYNTH-100K	250	91.66	89.36	50.99	57.68	92.41	67.94
SYNTH-400K	250	92.91	93.05	53.52	70.75	93.64	78.67
SYNTH-700K	250	92.57	93.78	**55.06**	**73.34**	**94.20**	**80.12**
SYNTH-1500K	250	**93.19**	**93.89**	49.85	72.10	94.04	80.10

Tabelle 5.4.6: Vergleich der vier unterschiedlich großen synthetischen Datensätze.Die Resultate der Experimente für QbE und QbS sind in mAP % eingegeben.

Zu diesem Zweck werden die Modelle anhand des SYNTH-700K-elastic-Datensatzes vortrainiert. Der SYNTH-700K-elastic-Datensatz wird im Abschnitt 4.3.1 vorgestellt. Zunächst wird die Auswahl von Parameterkonfigurationen, die zur Generierung elastischer Transformationen dienen, erläutert. Danach wird die Evaluierung der Modelle diskutiert.

Im Folgenden werden zwei Experimente beschrieben, die der empirischen Auswahl der Parameter für die elastischen Transformationen synthetischer Daten dienen. Bei der elastischen Transformation wird die Granularität über die Gittergröße und eine Magnitude bestimmt. Für das erste Experiment werden ein Gitter von 8x8 Pixeln und eine Magnitude von 2 Pixeln ausgewählt. Für das zweite Experiment werden ein Gitter von 4x4 Pixeln und eine Magnitude von 3 Pixeln ausgewählt.

Die Ergebnisse sind in der Tabelle 5.4.7 aufgeführt. Im Vergleich zu dem Experiment mit dem SYNTH-700K-Datensatz wird eine Verbesserung der Ergebnisse mit dem Gitter-4x4 und der Magnitude gleich 2 erreicht. Für das QbE-Szenario werden die

Verfahren	Anzahl der realen Trainings beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
Gitter 8x8, Magnitude=2	0	**51.66**	**63.75**	**37.59**	**54.35**	**70.09**	**38.13**
Gitter 4x4, Magnitude=3	0	50.18	62.92	36.57	53.35	67.65	36.44
SYNTH-700K	0	48.28	60.93	35.48	54.11	62.99	32.52

Tabelle 5.4.7: Vergleich der Parameterkonfigurationen zur Erstellung des SYNTH-700K-elastic-Datensatzes (die zwei ersten Zeilen der Tabelle) und der Leistungsfähigkeit der auf der Basis synthetischer Daten nachtrainierten Modelle. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben.

mAP-Werte um 3% (GW), 2% (IAM), 7% (Esposalles) und für das QbS-Szenario um 3% (GW) und 6% (Esposalles) verbessert.

I-SYNTH-700K- und II-SYNTH-700K-distorted Datensätze

Ein weiteres Experiment untersucht die Auswirkungen der synthetischen Trainingsdaten mit Störungen auf die Leistungsfähigkeit der Modelle bei realen Anwendungen. Zu diesem Zweck werden die zwei synthetische I-SYNTH-700K-distorted und II-SYNTH-700K-distorted Datensätze analysiert. Die beiden synthetischen Datensätze werden im Abschnitt 4.3.2 dargestellt.

Die Tabelle 5.4.8 zeigt die Evaluierungswerte der Modelle, die anhand der synthetischen I-SYNTH-700K-distorted und II-SYNTH-700K-distorted Datensätze vortrainiert werden. Das Experiment mit dem I-SYNTH-700K-distorted-Datensatz zeigte einen negativen Effekt auf die Leistungsfähigkeit der vortrainierten Modelle, da es 75% synthetische Wortabbilder mit Störungen beinhaltet. Das Experiment mit dem II-SYNTH-700K-distorted-Datensatz, der 20% synthetische Wortabbilder beinhaltet, erzielte eine deutliche Verbesserung bei der Evaluierung der Testdaten. Für das QbE-Szenario werden mAP-Werte von 56.09% (GW), 67.51% (Esposalles) und für das QbS-Szenario 66.66% (GW), 38.00% (Esposalles) erreicht. Die Ergebnisse bei dem IAM-Datensatz sind schlechter im Vergleich mit dem Experiment mit dem SYNTH-700K-Datensatz,

Verfahren	Anzahl der realen Trainings beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
I-SYNTH-700K-distorted	0	47.49	58.61	29.28	49.53	57.39	35.73
II-SYNTH-700K-distorted	0	**56.09**	**66.66**	32.35	53.49	**67.51**	**38.00**
SYNTH-700K	0	48.28	60.93	**35.48**	**54.11**	62.99	32.52

Tabelle 5.4.8: Vergleich der Leistungsfähigkeit der auf der Basis synthetischer Daten nachtrainierten Modelle. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben

da seine Textseiten im Vergleich mit den historischen GW- und Esposalles-Datensätzen weniger Störungen im Bildhintergrund beinhalten.

SYNTH-700K-elastic-distorted-Datensatz

Das Ziel dieses Experimentes ist die Auswirkung von elastisch deformirten Wortabbildern und Wortabbildern mit Störungen auf die Leistungsfähigkeit der Modlle zu untersuchen. Zu diesem Zweck werden die Modelle anhand des synthetischen SYNTH-700K-elastic-distorted-Datensatzes vortrainiert. Der SYNTH-700K-elastic-distorted-Datensatz wird im Abschnitt 4.3.2 dargestellt.

Die Ergebnisse des Vergleichs der SYNTH-700K, SYNTH-700K-elastic- und II-SYNTH-700K-distorted Datensätze sind in der Tabelle 5.4.9 zu sehen. Das Experiment zeigt, dass mit dem SYNTH-700K-elastic-distorted-Datensatz ähnliche Ergebnisse wie mit den Experimenten mit den SYNTH-700K-elastic- und II-SYNTH-700K-distorted-Datensätzen erreicht werden. Insgesamt zeigen die experimentellen Untersuchungen, dass während einer Vortrainingsphase mit deformierten synthetischen Daten, zusätzliche Informationen gewonnen werden, die die Leistungsfähigkeit der vortrainierten Modelle mit realen Daten verbessern.

Verfahren	Anzahl der realen Trainings-beispielen	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
SYNTH-700K-elastic	0	51.66	63.75	**37.59**	54.35	**70.09**	38.13
II-SYNTH-700K-distorted	0	**56.09**	**66.66**	32.35	53.49	67.51	38.00
SYNTH-700K-elastic-distorted	0	55.28	65.19	35.02	**56.95**	69.54	**43.48**
SYNTH-700K	0	48.28	60.93	35.48	54.11	62.99	32.52
SYNTH-700K-elastic	100	89.63	92.00	48.96	67.31	90.14	74.57
II-SYNTH-700K-distorted	100	**90.22**	92.64	46.18	65.08	92.19	73.55
SYNTH-700K-elastic-distorted	100	89.46	**92.75**	46.85	**67.88**	**93.06**	74.61
SYNTH-700K	100	89.62	92.03	**49.27**	67.12	92.30	**75.63**
SYNTH-700K-elastic	250	93.02	93.56	**55.58**	**74.02**	94.18	**81.24**
II-SYNTH-700K-distorted	250	**93.52**	93.97	53.29	73.29	94.04	80.00
SYNTH-700K-elastic-distorted	250	93.24	**94.16**	53.51	73.43	94.07	79.58
SYNTH-700K	250	92.57	93.78	55.06	73.34	**94.20**	80.12

Tabelle 5.4.9: Vergleich der synthetischen und deformierten synthetischen Daten. Die Resultate der Experimente für QbE und QbS sind in mAP % eingegeben.

5.4.6 *Deformierte reale Daten*

Nach dem erfolgreichen Versuchsaufbau von Deformierungen synthetischer Daten wird bei diesem Experiment untersucht, ob Deformierungen realer Daten zur Verbesserung der Leistungsfähigkeit der nachtrainierten Modelle führt. Zu diesem Zweck

Verfahren	Anzahl der realen Trainings beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
distorted-1000	100	87.23	90.04	45.26	67.05	92.51	75.29
elastic-1000	100	87.12	91.44	45.63	68.06	91.62	74.05
SYNTH-700K-elastic-distorted	100	89.46	92.75	46.85	67.88	93.06	74.61

Tabelle 5.4.10: Vergleich der Leistungsfähigkeit von nachtrainierten Modellen mit unterschiedlich augmentierten Datensätzen. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben.

werden drei ausgewählte Mengen mit 100 realen Wortabbildern für GW-, IAM- und Esposalles-Datensätze wie folgt augmentiert:

distorted-1000:
Die Anzahl der Wortabbilder pro Wortklasse wird auf 1000 Trainingsbeispiele durch die Modellierung von Störungen (siehe Unterabschnitt 4.3.2) erhöht.

elastic-1000:
Die Anzahl der Wortabbilder pro Wortklasse wird auf 1000 Trainingsbeispiele durch die elastischen Transformationen (siehe Unterabschnitt 4.3.1) erhöht.

Das auf der Basis des SYNTH-700K-elastic-distorted-Datensatzes vortrainierte Modell wird mit distorted-1000- und elastic-1000-Datensätzen nachtrainiert. Die Tabelle 5.4.10 zeigt die Ergebnisse der Experimente. Insgesamt zeigte sich, dass nach der Auswertung der nachtrainierten Modelle sehr ähnliche Ergebnisse mit den GW-, IAM- und Esposalles-Datensätzen erhalten werden. Keine der angewendeten Datenaugmentierungsmethoden kann mehr zusätzliche Informationen als eine andere Datenaugmentierungsmethode bereitstellen, um die Leistungsfähigkeit der nachtrainierten Modelle zu erhöhen.

5.4.7 *Auswahl realer Trainingsbeispiele*

Bei allen vorherigen Experimenten wurden zufällig ausgewählte Mengen von handgeschriebenen Wortabbildern verwendet, bei denen eine Wortklasse mehrere Abbilder enthalten kann. Eine andere Möglichkeit besteht darin, dass genau ein Abbild pro Wortklasse zufällig ausgewählt wird. Es werden zwei Teilmengen unique-100 und

Verfahren	Anzahl der realen Trainings-beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
unique-100	100	**89.51**	92.71	**51.06**	**73.15**	89.97	**79.40**
SYNTH-700K-elastic-distorted	100	89.46	**92.75**	46.85	67.88	**93.06**	74.61
unique-250	250	92.79	**95.11**	52.62	**75.41**	94.05	**83.93**
SYNTH-700K-elastic-distorted	250	93.24	94.16	**53.51**	73.43	**94.07**	79.58

Tabelle 5.4.11: Vergleich der Ergebnisse für die zwei Auswahlstrategien der Stichproben von handgeschriebenen Wortabbildern. Die Resultate der Experimente für QbE und QbS sind in mAP % eingegeben.

unique-250 aus der entsprechenden Trainingsmenge des jeweiligen Datensatzes gebildet. Die Tabelle stellt den Vergleich der Ergebnisse für die zwei Auswahlstrategien dar.

Für die Feinabstimmung werden die Modelle verwendet, die auf der Basis des synthetischen SYNTH-700K-elastic-distorted-Datensatzes vortrainiert wurden. Bei dem Experiment mit dem unique-100-Datensatz werden wesentliche Verbesserungen der mAP-Werte für das QbS-Szenario um 5% (IAM) und 5% (Esposalles) sowie für das QbE-Szenario um 4% (IAM) im Vergleich zum Experiment mit dem SYNTH-700K-elastic-distorted-Datensatz mit 100 zufällig ausgewählten handgeschriebenen Wortabbildern erzielt. Bei dem QbE-Szenario für den Esposalles-Datensatz verschlechterte sich das Ergebnis um 3%.

Die Experimente mit dem unique-250-Datensatz und dem SYNTH-700K-elastic-distorted mit 250 zufällig ausgewählten handgeschriebenen Wortabbildern zeigen sehr ähnliche Ergebnisse.

Insgesamt zeigte sich bei den experimentellen Untersuchungen, dass bei der Annotation von bis zu 100 Trainingsbeispielen die Auswahl eines Abbildes pro Wortklasse zu empfehlen ist.

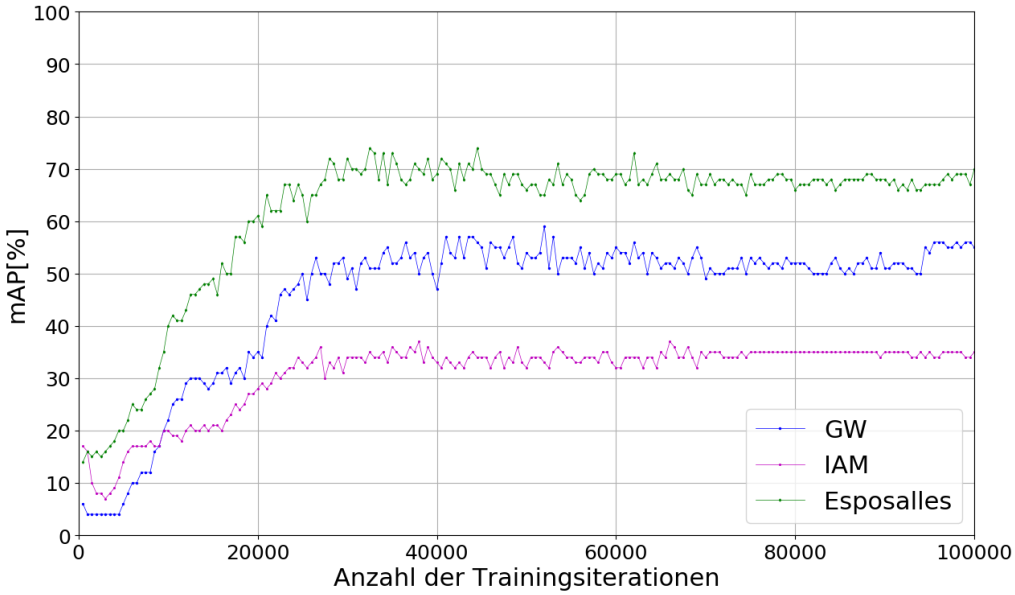

Abbildung 5.4.3: Evaluierung der Testdaten der GW, IAM und Esposalles Datensätze für das QbE-Szenario beim Training anhand synthetischer Daten.

5.4.8 *Diskussion*

Im Folgenden werden die Ergebnisse des Experimentes mit dem SYNTH-700K-elastic-distorted-Datensatz diskutiert und mit der Arbeit vom Gurjar et al. sowie verwandten Arbeiten verglichen. Die Abbildung 5.4.3 zeigt die Evaluierung der Testdaten der GW-, IAM- und Esposalles-Datensätze beim Training der PHOC-Netze anhand des SYNTH-700K-elastic-distorted-Datensatzes. Die mAP-Werte werden für das QbE-Szenario für jede 500. Iteration ausgewertet. Die Sprünge in den mAP-Werten weisen auf eine Überanpassung hin.

Auf der Basis synthetischer Daten vortrainierte Modelle zeigen bei der Evaluierung bessere mAP-Werte als im [GSF17].

Im Vergleich zu dem Baseline-Verfahren sind die mAP-Werte für das QbE-Szenario um 15% (GW), 9% (IAM), 34% (Esposalles) und für das QbS-Szenario um 16% (GW), 20% (IAM) und 33% (Esposalles) höher.

Verfahren	Anzahl der realen Trainings-beispiele	GW		IAM		Esposalles	
		QbE	QbS	QbE	QbS	QbE	QbS
SYNTH-700K-elastic-distorted	0	55.28	65.19	35.02	56.95	69.54	43.48
	100	89.46	92.75	46.85	67.88	93.06	74.61
	250	93.24	94.16	53.51	73.43	94.07	79.58
Baseline[GSF17]	0	39.89	48.92	26.21	36.57	34.92	10.30
	100	83.05	86.69	38.45	56.47	89.67	71.15
	250	90.76	92.39	43.78	60.90	94.06	82.43
AttributeSVMs[AGFV14]	Alle	93.04	91.29	55.73	73.72	-	-
PHOCNet [SF18]	Alle	97.58	95.58	85.50	92.38	**97.40**	**93.67**
HWNet v2 [KJ18]	Alle	96.01	-	**90.65**	-	-	-
DeepEmbed [KDJ18]	Alle	**98.01**	98.86	90.38	94.04	-	-
Synth-DeepEmbed [KDJ18]	Alle	-	**98.98**	-	**95.09**	-	-

Tabelle 5.4.12: Übersicht über die Ergebnisse des Experimentes mit dem SYNTH-700K-elastic-distorted-Datensatz und die verwandten Arbeiten. Die Resultate der Experimente für QbE und QbS sind in mAP % angegeben.

Beim Nachtraining der Modelle anhand handgeschrieber Wortabbilder werden die fast erzielten Werten ab 2.000 Iterationen erhalten, und verbessert sich um 2-5% während der Nachtrainingszeit bis 40.000 Iterationen.

Die mAP-Werte des Baseline-Verfahrens werden für alle drei GW-, IAM- und Esposalles-Datensätze sowohl für das QbE- als auch für das QbS-Szenario bereits mit einer Menge von 100 und 250 realen Trainingsbeispiele übertroffen. Beim Experiment SYNTH-700K-elastic-distorted mit 100 handgeschriebenen Wortabbildern werden die mAP-Werte für das QbE-Szenario um 6% (GW), 8% (IAM), 3% (Esposalles) und für das QbS-Szenario um 6% (GW), 11% (IAM), 3% (Esposalles) verbessert.

Die mit AttributeSVWs implementierten Ergebnisse werden schon mit 250 manuell annotierten handgeschriebenen Wortabildern erreicht, was der Senkung der gesamten Anzahl von manuell annotierten Wortabbildern um 93% für den GW-Datensatz und um 99.6% für den IAM-Datensatz entspricht.

Im Vergleich zu den [SF18], [KJ18], [KDJ18] liegen die Unterschiede für das QbE-Szenario bei 5% (GW) und für das QbS-Szenario bei 3% (GW). Für den Esposalles-Datensatz sind die mAP-Werte um 2% für das QbE-Szenario und 15% für das QbS-

Szenario niedrieger im Vergleich zum [SF18]. Die erhaltenen mAP-Werte für die beiden QbE- und QbS-Szenarien auf dem IAM-Datensatz im Vergleich zu den [SF18], [KJ18], [KDJ18] sind deutlich schlechter.

Es kann beobachtet werden, dass die Leistungsfähigkeit der Modelle von der visuellen Charakteristika und der Große des Datensatzes abhängt. Sind die visuellen Unterschiede zwischen Handschriften gering, reicht dann wenige hundert reale annotierte Wortabbilder aus, um eine konkurrenzfähige Ergebnisse zu erzielen. Bei den großen visuellen Unterschieden zwischen Handschriften werden wesentlich mehr annotierte Wortabbilder benötigt.

6 FAZIT

Während der Digitalisierung wächst die Anzahl an Sammlungen von modernen und wertvollen alten historischen handgeschriebenen Textdokumenten und Manuskripten, die eingescannt oder abfotografiert werden. Dadurch können die Dokumente in einer digitalen Form als Bilder zur Verfügung gestellt werden. Eine hohe Varianz der Schreibweise und Qualitätsdefizite erschweren die automatische Texterkennung. Um in den Dokumentsammlungen nach bestimmten Schlüsselwörtern zu suchen, wird das Word-Spotting-Verfahren angewendet.

Die Implementierung des Word-Spotting-Verfahrens mit tiefen Faltungsnetzen repräsentiert den Stand der Technik. Insbesondere sind die tiefen PHOC-Netze erfolgreich, die speziell für die Word-Spotting-Aufgaben entwickelt wurden. Eine der wichtigsten Eigenschaften des PHOC-Netzes ist, dass es bereits mit einigen tausend annotierten Wortabbildern von Grund auf trainiert werden kann [SF18]. Für ein auf der Basis synthetischer Daten vortrainiertes PHOC-Netz reichen einhundert manuell annotierte reale Wortabbilder aus, um dessen Gewichte zu adaptieren und brauchbare Ergebnisse mit anspruchsvollen Datensätzen zu erzielen [GSF17].

Forscher haben bei vielen anspruchsvollen Datensätzen bislang sehr gute Ergebnisse mit Word-Spotting-Modellen erreicht [SF18] [KJ18] [KDJ18]. Trotzdem ist die Implementierung eines Word-Spotting-Verfahrenes für eine praktische Anwendung schwierig, denn jede Sammlung von handgeschriebenen Textdokumenten unterscheidet sich zu einem großen Teil durch eigene visuelle Charakteristika und macht deshalb ein Training von Modellen in Bezug auf die neuen Handschriften erforderlich. Wenn nur eine geringe Anzahl von annotierten Trainingsbeispielen zur Verfügung steht oder das Trainingsmaterial ganz fehlt, ist das Training eines tiefen Faltungsnetzes eine Herausforderung.

Um den Bedarf an von Hand annotierten Beispielen und den damit verbundenen manuellen Annotationsaufwand zu reduzieren, werden auf Grundlage der Arbeit [KJ16] im Rahmen der vorliegenden Diplomarbeit mehrere synthetische Datensätze generiert und in Kapitel 4 vorgestellt. Wie in der Arbeit von [GSF17] vorgeschlagen, wird das PHOC-Netz im Anschluss auf der Basis eines synthetischen Datensatzes vortrainiert, an kleine Mengen manuell annotierte handgeschriebene Wortabbilder angepasst und die Ergebnisse werden ausgewertet.

Bei der Generierung synthetischer Daten ergaben sich große Herausforderungen in Bezug auf die Nachbildung der realitätsnahen charakteristischen visuellen Merkmale realer Wortabbilder. Die Experimente zeigten, dass sich Unterschiede zwischen synthetischen und realen Daten durch elastische Transformationen (siehe Unterabschnitt 4.3.1) und modellierte Störungen (siehe Unterabschnitt 4.3.2) reduzieren lassen. Die bereits erlernten visuellen Merkmale synthetischer Wortabbilder konnten durch die PHOC-Netze zum großen Teil bei den unbekannten handgeschriebenen Wortabbildern wieder erkannt werden. Für das QbE-Szenario werden mAP-Werte von 55.28% (GW), 35.02% (IAM), 69.54% (Esposalles) und für das QbS-Szenario von 65.19% (GW), 56.95% (IAM), 43.48% (Esposalles) erreicht.

Grundsätzlich kann das vortrainierte Modell bei einer Sammlung von handgeschriebenen Dokumenten für eine Suche verwendet werden, wenn kein Trainingsmaterial vorhanden ist. Im Gegensatz zu Word-Spotting-Verfahren, die außer einem Anfragewortabbild keine annotierten Daten verwenden, kann eine Suche zusätzlich in Form einer Zeichenkette eingegeben werden. Dadurch kann eine manuelle und gegebenenfalls langwierige Suche nach einem Anfrageabbild vermieden werden. Zusätzlich kann die Generalisierungsfähigkeit der PHOC-Netze während eines Nachtrainingsprozesses mit einer relativ geringen Anzahl von manuell annotierten Wortabbildern erheblich verbessert werden. Die Steigerung der Leistungsfähigkeit der nachtrainierten Modelle ist dabei von den visuellen Charakteristika der Handschriften abhängig.

Auf Grundlage der experimentellen Untersuchungen lässt sich zusammenfassen, dass synthetische Datensätze die Probleme des Fehlens einer ausreichend großen Menge von annotierten realen Wortabbildern mildern können und dementsprechend den manuellen Annotationsaufwand reduzieren. Insbesondere ist das Verfahren für historische handgeschriebene Dokumente und Manuskripte vorteilhaft, da bei diesen in der Regel kein Trainingsmaterial vorhanden ist und OCR-Verfahren nicht funktionieren. Aufgrund der visuellen Unterschiede zwischen synthetischen und realen Wortabbildern ist ein Nachtraining von Modellen mit manuell annotierten handgeschriebenen Wortabbildern von Bedeutung. Bereits mit einer relativ kleinen Menge von annotierten realen Trainingsbeispielen werden zum großen Teil relevante Ergebnisse erzielt und somit der manuelle Annotationsaufwand im Vergleich zum Training von Grund auf deutlich reduziert. Der weitere Vorteil ist, dass ein vortrainiertes Modell als Basis für die Feinabstimmung verwendet werden kann. Dadurch reduziert sich der Trainingsaufwand [GSF17].

Mit der Methode in Anlehnung an [KJ16] zur Generierung der synthetischen Daten lassen sich unterschiedlich große synthetische Datensätze mit verschiedenen Alphabetzeichen generieren. Eine Herausforderung für das Verfahren stellen historische

und moderne handgeschriebene Textdokumente dar, für die keine den Handschriften ähnelnden Computerschriften vorhanden sind.

LITERATURVERZEICHNIS

[AGFV14] Almázan, Jon ; Gordo, Albert ; Fornés, Alicia ; Valveny, Ernest: Word Spotting and Recognition with Embedded Attributes. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on 36*, 2014, S. 2552–2566

[BB06] Burge, Mark J. ; Burger, Wilhelm: *Digitale Bildverarbeitung. Eine Einführung mit Java und ImageJ. 2., überarbeitete Auflage*. Berlin Heidelberg : Springer-Verlag, 2006

[BDW18] Bhardwaj, Anurag ; Di, Wei ; Wei, Jianing: *Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling*. Packt Publishing Ltd, 2018. – ISBN 978–1–785–88777–2

[BSH17] Bloice, Marcus D. ; Stocker, Christof ; Holzinger, Andreas: Augmentor: An Image Augmentation Library for Machine Learning. In: *arXiv* (2017)

[Cho18] Chollet, François: *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP Verlags GmbH, 2018. – ISBN 978–3–9584–5838–3

[DV18] Dumoulin, Vincent ; Visin, Francesco: A guide to convolution arithmetic for deep learning. In: *arXiv* (2018)

[Dö18] Dörn, Sebastian: *Programmieren für Ingenieure und Naturwissenschaftler*. Springer Vieweg, 2018. – ISBN 978–3–662–54303–0

[Erh08] Erhardt, Angelika: *Einführung in die Digitale Bildverarbeitung Grundlagen, Systeme und Anwendungen*. Wiesbaden: Vieweg+Teubner, 2008. – ISBN 978–3–519–00478–3

[Ert16] Ertel, Wolfgang: *Grundkurs Künstliche Intelligenz: Eine praxisorientierte Einführung. 4., überarbeitete Auflage*. Springer Vieweg, 2016. – ISBN 978–3–658–13548–5

[ES] http://www.cvc.uab.es/5cofm/competition/

[FFM12] Frinken, Volkmar ; Fischer, Andreas ; Manmatha, R.: A Novel Word Spotting Method Based on Recurrent Neural Networks. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012

[Fin18] Fink, Gernot A.: *Deep Learning for Word Spotting*. Tutorial presented at Int. Conf. on Frontiers in Handwriting Recognition (ICFHR), Niagara Falls, USA, 2018. – http://http://patrec.cs.tu-dortmund.de/pubs/papers/Kolkata1801-Fink.pdf; abgerufen am 2. Juni 2018.

[fona] http://https://fonts.google.com; abgerufen am 12. Juni 2018.

[fonb] https://www.1001fonts.com; abgerufen am 12. Juni 2018.

[Fra09] Frank, Kreuder: *2D-3D-Registrierung mit Parameterentkopplung für die Patientenlagerung in der Strahlentherapie*. KIT Scientific Publishing, 2009

[GBC16] Goodfellow, Ian ; Bengio, Yoshua ; Courville, Aaron: *Deep Learning*. MIT Press, 2016. – http://www.deeplearningbook.org

[GSF17] Gurjar, Neha ; Sudholt, Sebastian ; Fink, Gernot A.: Learning Deep Representations for Word Spotting Under Weak Supervision. In: *arXiv* (2017)

[GSGN17] Giotis, Angelos ; Sfikas, Giorgos ; Gatos, Basilis ; Nikou, Christophoros: A survey of document image word spotting techniques. In: *Pattern Recognition* 68 (2017), S. 310–332

[GW] http://ciir.cs.umass.edu/downloads/old/data_sets.html/

[GW08] Gonzalez, Rafael C. ; Woods, Richard E.: *Digital image processing. 3. ed.* Upper Saddle River, NJ: Pearson Prentice Hall, 2008. – ISBN 978–0–131–68728–8

[Hue] Huew: *Introduction to Convolution Neural Networks*. – http://engineering.huew.co/introduction-to-convolution-neural-networks-18981d1cd09a; abgerufen am 2. Juni 2018.

[HZRS15] He, Kaiming ; Zhang, Xiangyu ; Ren, Shaoqing ; Sun, Jian: Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In: *Proc. of the Int. Conf. on Computer Vision*, 2015, S. 1026–1034

[IAM] http://www.fki.inf.unibe.ch/databases/iam-handwriting-database/

[KBB+15] Kruse, Rudolf ; Borgelt, Christian ; Braune, Christian ; Klawonn, Frank ; Moewes, Christian ; Steinbrecher, Matthias: *Computational Intelligence.*

Eine methodische Einführung in Künstliche Neuronale Netze, Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze. 2., überarbeitete und erweiterte Auflage. Springer Vieweg, 2015. – ISBN 978–3–658–10903–5

[KDJ18] Krishnan, Praveen ; Dutta, Kartik ; Jawahar, C.V.: Word Spotting and Recognition using Deep Embedding. In: *arXiv* (2018)

[KJ16] Krishnan, Praveen ; Jawahar, C.V.: Generating Synthetic Data for Text Recognition. In: *arXiv* (2016)

[KJ18] Krishnan, Praveen ; Jawahar, C.V.: HWNet v2: An Efficient Word Image Representation for Handwritten Documents. In: *arXiv* (2018)

[KSH12] Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E.: ImageNet Classification with Deep Convolutional Neural Networks. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, Curran Associates Inc., 2012 (NIPS'12), 1097–1105

[LBD+90] LeCun, Y. ; Boser, B. ; Denker, J. S. ; Henderson, D. ; Howard, R. E. ; Hubbard, W. ; Jackel, L. D.: Handwritten Digit Recognition with a Back-Propagation Network. NIPS, 1990

[LKJ] Ly, Fei-Fei ; Karpathy, Andrej ; Johnson, Justin: *CS231n Convolutional Neural Networks for Visual Recognition*. – http://cs231n.github.io/convolutional-networks/; abgerufen am 2. Juni 2018.

[mag] https://www.imagemagick.org/; abgerufen am 11. September 2018.

[MHN13] Maas, Andrew L. ; Hannun, Awni Y. ; Ng, Andrew Y.: Rectifier Nonlinearities Improve Neural Network Acoustic Models. In: *IJML* 30 (2013)

[Mor17] Moreau, Christian: *Bilder lernen mit Neuronalen Netzen*. 2017. – http://www.statworx.com/de/blog/bilder-lernen-mit-neuronalen-netzen/; abgerufen am 2. Juni 2018.

[Nie03] Niemann, Heinrich: *Klassifikation von Mustern*. Universität Erlangen, 2003. – http://www5.informatik.uni-erlangen.de/fileadmin/Persons/NiemannHeinrich/klassifikation-von-mustern/m00-www.pdf

[RF15] Rothacker, Leonard ; Fink, Gernot A.: Segmentation-free Query-by-String Word Spotting with Bag-of-Features HMMs. In: *Proc. Int. Conf. on Document Analysis and Recognition*. Nancy, France, 2015

[RM07] Rath, Tony M. ; Manmatha, R.: Word spotting for historical documents. In: *Proc. Int. Conf. on Document Analysis and Recognition*, 2007

[RRF13] Rothacker, Leonard ; Rusinol, Marcal ; Fink, Gernot A.: Bag-of-Features HMMs for Segmentation-Free Word Spotting in Handwritten Documents. In: *Proc. Int. Conf. on Document Analysis and Recognition*. Washington DC, USA, 2013

[RSR+17] Rothacker, Leonard ; Sudholt, Sebastian ; Rusakov, Eugen ; Kasperidus, Matthias ; Fink, Gernot A.: Word Hypotheses for Segmentation-free Word Spotting in Historic Document Images. In: *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017

[SF16] Sudholt, Sebastian ; Fink, Gernot A.: PHOCNet: A Deep Convolutional Neural Network for Word Spotting in Handwritten Documents. In: *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition*, 2016, S. 277–282

[SF18] Sudholt, Sebastian ; Fink, Gernot A.: Attribute CNNs for Word Spotting in Handwritten Documents. In: *Int. Journal on Document Analysis and Recognition* (2018). – to appear

[SKS+14] Srivastava, Nitish ; Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E. ; Salakhutdinov, Ruslan: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. In: *IJML* 15 (2014), S. 1929–1958

[SSP03] Simard, Patrice Y. ; Steinkraus, Dave ; Platt, John C.: Best practices for convolutional neural networks applied to visual document analysis. In: *Proceedings of the 7th International Conference on Document Analysis and Recognition*, 2003, S. 958–964

[ZSC17] Zhou, Yiren ; Song, Sibo ; Cheung, Ngai-Man: On classification of distorted images with deep convolutional neural networks. In: *arXiv* (2017)