Karlsruher Institut für Technologie

Neuronale Textklassifikation mittels Wissen aus der maschinellen Übersetzung

Masterarbeit von

Dominik Kleiser

an der Fakultät für Informatik
Institut für Anthropomatik und Robotik (IAR)
Interactive Systems Labs (ISL)

Erstgutachter:	Prof. Dr. A. Waibel
Zweitgutachter:	Prof. Dr. T. Asfour
Betreuender Mitarbeiter:	Dr.-Ing. Jan Niehues
Zweiter betreuender Mitarbeiter:	Dr.-Ing. Matthias Keller (extern)

01. Februar 2018 – 31. Juli 2018

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Änderungen entnommen wurde.

Karlsruhe, den 15.10.2018

. . . . . .

(Dominik Kleiser)

Zusammenfassung

Die Textklassifikation (TK) ist eine der fundamentalen Problemstellungen der Sprachverarbeitung, bei der die Aufgabe darin besteht, Texte in Abhängigkeit ihres semantischen Inhaltes einer oder mehreren vordefinierten Kategorien oder Klassen zuzuordnen. Die TK wird nicht nur von vielen wissenschaftlichen Arbeiten thematisiert, sondern ist auch Grundlage vieler praktischer Anwendungen in Industrie und Wirtschaft. Eingesetzt werden Textklassifikatoren beispielsweise in Suchmaschinen, bei Spamfiltern, zur automatisierten Analyse von Persönlichkeitsprofilen und zur Vorhersage von Aktienkursen. Zur Lösung der Problemstellung der TK werden Methoden des maschinellen Lernens (ML) eingesetzt, die automatisiert durch die Analyse von vielen gelabelten Beispieltexten selbständige statistische Zusammenhänge in den Daten erkennen können. Das *Labeling* von textuellen Daten ist häufig mit einem hohen zeitlichen und finanziellen Aufwand verbunden. Aus Kostengründen ist es daher nicht immer praktikabel, ausreichend viele Beispiele zu generieren. Noch schwieriger ist die Situation bei kurzlebigen Aufgaben, die während ihrer Lebensdauer mehrfach an veränderte Anforderungen angepasst werden müssen und dadurch ein regelmäßiges *relabeling* notwendig machen. Bei vielen TK-Aufgaben ist folglich die Datenknappheit ein beschränkender Faktor für die Leistung der Systeme.

Die vorliegende Arbeit beschäftigt sich mit Methoden, um dieser Datenknappheit entgegenzutreten. Es werden Ansätze entwickelt, mit denen das in maschinellen Übersetzern gespeicherte Wissen auf TK-Systeme übertragen werden kann. Dafür werden zwei unterschiedliche Problemszenarien besprochen:

- Zum einen wird untersucht, wie mit Methoden der neuronalen maschinellen Übersetzung Systeme zur crosslingualen Textklassifikation (CTK) aufgebaut werden können. Dadurch wird es ermöglicht, Trainingsdaten in eine andere Sprache zu Übertragen, und somit Klassifikatoren für Sprachen zu trainieren, für die keine Daten vorhanden sind.

- Zum anderen werden Methoden diskutiert, wie mit dem Wissen aus maschinellen Übersetzern die Fehlerrate eines einsprachigen TK-Systems reduziert werden kann, wenn für dieses nicht ausreichend viele Trainingsdaten vorhanden sind.

Die in der Arbeit schwerpunktmäßig behandelten Lernverfahren sind die neuronalen Netze (NNs), die aufgrund ihres hierarchischen Aufbaus und der Fähigkeit, Strukturen für verschiedene Lerntasks wiederzuverwenden, ein großes Potential für Wissenstransfermethoden eröffnen. In der vorliegenden Arbeit kann gezeigt werden, dass NNs auch bei wenigen Trainingsdaten mit der Leistung der traditionellen Verfahren in Wettbewerb treten können und diese mit einer geringen Optimierung der Hyperparameter teilweise bereits übertreffen.

Bei der Evaluation der CTK-Verfahren wird festgestellt, dass die Klassifikatoren annähernd die Leistung eines Orakels der Zielsprache erreichen. Mit den heutigen maschinellen Übersetzern ist

demnach beim Übergang zwischen den Sprachen nur noch mit einem geringen Leistungsabfall zu rechnen.

Zum reduzieren der Fehlerrate von Klassifikatoren wird ein Verfahren zur Generierung zusätzlicher Trainingsdaten beschrieben (Trainingsdatenerweiterung), dass auf neun von elf Datensätze die Baselinemodelle übertrifft. Es erzielt durchschnittlich eine Verbesserung von 0.3 Prozentpunkten. Ein zweites Verfahren verwendet Methoden des neuronalen Lerntransfer zur Übertragung von internem Wissen eines Übersetzungssystems auf einen Textklassifikator. Es erzielt auf acht von elf Datensätzen neue Bestwerte und führt zu einer durchschnittlichen Leistungssteigerung von 1.0 Prozentpunkten.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Die menschliche Sprache ist komplex, mehrdeutig und ständigen Veränderungen unterworfen. Die Verarbeitung natürlicher Sprache (engl. natural-language processing; kurz: NLP) ist ein interdisziplinärer Teilbereich der Informatik, der sich mit Techniken und Methoden zur maschinellen Verarbeitung natürlicher Sprache in Form von Text- und Audiodaten beschäftigt. Das erklärte Ziel ist es, Erkenntnisse der Linguistik mit statistischen und regelbasierten Verfahren der Informatik zu kombinieren, um Computern ein „Verständnis“ der menschlichen Sprache zu verschaffen und aus ihr die für eine weitere Verarbeitung relevanten Informationen zu extrahieren.

Das Aufkommen des *World Wide Webs* und die bis heute andauernde Digitalisierung sämtlicher Lebensbereiche führte in den letzten Jahrzehnten zu einem explosionsartigen Wachstum des global verfügbaren Wissens. Insbesondere mit der Verbreitung von sozialen Netzwerken, Shopping- und News-Portalen, Bewertungsplattformen und anderen Online-Datenbanken stehen heute nahezu unbegrenzte Textressourcen frei zur Verfügung. Der Wunsch, diese unstrukturierten Ressourcen automatisiert zu analysieren und zu verarbeiten, verlieh der Forschung im Bereich Natürliche Sprachverarbeitung (NLP) einen großen Aufschwung. Neben dem gestiegenen Interesse aus der Wissenschaft konnte die NLP in jüngster Zeit aber auch zahlreiche neue Anwendungen in Industrie und Wirtschaft hervorbringen. In nahezu allen Wirtschaftssektoren sind Unternehmen heutzutage mit großen Mengen unstrukturierter Dokumente wie Notizen, Briefe, E-Mails, PR-Meldungen oder auch Patientenakten konfrontiert. Mit den Verfahren der NLP können diese Daten automatisiert nach relevanten Informationen durchsucht oder zusammengefasst werden.

1.1. Textklassifikation

Die Textklassifikation (TK) ist eine der fundamentalen Problemstellungen der Sprachverarbeitung, bei der die Aufgabe darin besteht, Texte in Abhängigkeit ihres semantischen Inhaltes einer oder mehreren vordefinierten Kategorien oder Klassen zuzuordnen. Die durch die Klassifkation extrahierten Informationen helfen dabei, ein abstraktes Verständnis über den Inhalt von unstrukturierten Texten zu entwickeln. Anwendung finden die Textklassifikatoren in einem breitem Spektrum an Disziplinen des Text Minings[1]. Beispielsweise können sie von Unternehmen dazu verwendet werden, um ein besseres Verständnis davon zu bekommen, wie in sozialen Medien oder den Nachrichten über sie berichtet wird. Weite Anwendungsszenarien sind das Filtern und Organisieren von Nachrichtentexten[54], die Erkennung von Spam-Mails[85] oder auch die Vorhersage der Entwicklung von Aktienkursen[11].

Problemstellung der TK können mit regelbasierten Verfahren oder aber mit den Methoden des maschinellen Lernens (ML) gelöst werden. Hängt die Klasse eines Texts von komplexen semantischen Eigenschaften ab, ist es oftmals selbst für Experten schwierig, entsprechende

Regeln manuell festzulegen. Aus diesem Grund werden heutzutage vorwiegend automatische Lernverfahren eingesetzt, deren Vorteil es ist, durch die Analyse von vielen gelabelten Beispieltexten selbständige statistische Zusammenhänge in den Daten zu erkennen. Die Fehlerrate der Systeme wird maßgeblich von Umfang und Qualität der vorhandenen Trainingsdaten bestimmt. Für das *Labeling* der Trainingsbeispiele sind die Verfahren üblicherweise auf menschliches Feedback angewiesen, ein Prozess, der häufig mit einem hohen zeitlichen und finanziellen Aufwand verbunden ist, weswegen es aus Kostengründen nicht immer praktikabel ist, ausreichend viele Beispiele zu generieren. Noch schwieriger ist die Situation bei kurzlebigen Aufgaben, die während ihrer Lebensdauer mehrfach an veränderte Anforderungen angepasst werden müssen und dadurch ein regelmäßiges *relabeling* erfordern. In vielen Fällen ist Datenknappheit ein beschränkender Faktor für die Leistung der Klassifikatoren.

1.2. Maschineller Wissenstransfer

Die wissenschaftliche Fachwelt konzentriert sich bei der Entwicklung neuer ML-Verfahren hauptsächlich auf die überwachten (engl. supervised) Lernverfahren. Beim klassischen Vorgehen des maschinellen Lernens werden Modelle für das Lösen von Lernaufgaben (engl. Tasks) auf bestimmten Domänen trainiert. Der *Task* definiert die Problemstellung (z.B. Bilderkennung, Textklassifikation oder maschinelle Übersetzung). Die Domäne legt fest, aus welchem Umfeld die Daten stammen (z.B. Bilder von deutschen Straßen oder Texte aus Nachrichtenartikeln).

Die steigende Rechenleistung, das Aufkommen des GPGPU-Comptutings und das mit der Verbreitung des Internets einhergehende Wachstum von global verfügbaren Datenquellen sind nur ein paar der Einflussfaktoren, die die Entwicklungen von überwachten Lernmethoden in den letzten Jahren positiv begünstigten.

ML-Modelle sind sehr gut darin geworden, mithilfe von gelabelten Trainingsdaten selbständig Zuordnungen zwischen Eingangssignalen und Ausgangssignalen zu finden. Bedingung für die Effektivität von überwachten Lerntechniken sind jedoch große Mengen gelabelter Trainingsdaten. Wenn nicht genügend Ressourcen zur Verfügung stehen, versagt das klassische Lernparadima[97]. Das überwachte Lernen erfordert es, dass Trainings- und Testdaten dem selben Merkmalsraum entstammen und der gleichen Bedingungen unterliegen. Ändern sich die Gegebenheiten in der realen Welt oder wird ein Modell mit bisher ungesehen Szenarien konfrontiert, so können von ihm keine verlässlichen Prädiktion mehr geliefert werden. Der Datensatz muss angepasst und das Modell erneut trainiert werden.

Soll ein Modell für die Erkennung von Autos auf deutschen Straßen trainiert werden, so kann es schon zu Problemen kommen, wenn die Trainingsdaten auf amerikanischen Straßen aufgenommen wurden. Zwar sind die beiden Domänen eng miteinander verwandt, jedoch unterscheiden sich beispielsweise die Landschaften und die Fahrzeugtypen auf den Straßen. Es kommt zu Leistungseinbrüchen, da das Modell wegen eines erlernten Bias nicht ausreichend auf die geänderte Domäne generalisiert. Noch schwieriger ist die Situation, wenn das Wissen zwischen verschiedenen Tasks geteilt werden soll. Ein Detektor für Autos und ein Detektor für Fußgänger müssen intern ein ähnliches Weltmodell erlernen. Wegen des unterschiedlichen Labelraums kann das gemeinsame Wissen aber nicht ohne weitere Vorkehrungen zwischen den Tasks geteilt werden.

Die Fähigkeit des Menschen, eine Problemlösung intelligent auf eine andere, vergleichbare Situation zu übertragen, um neue Probleme schneller zu lösen oder eine bessere Lösung für sie zu finden, bezeichnet man in der Psychologie als Lerntransfer. Das wissenschaftliche Interesse, Wissenstransfermethoden auch im maschinellen Lernen einzusetzen, hat eine lange Historie und geht zurück bis in die Anfangszeiten der künstlichen Intelligenz (KI). 2005 definierte die amerikanische Defense Advanced Research Projects Agency (DARPA) den Lerntransfer als die Fähigkeit eines Systems, das gemeinsame Wissen zwischen Aufgaben zu erkennen und das Wissen vorher gelernter Aufgaben auf neue Aufgaben zu übertragen[27]. Nach dieser Definition werden für den Lerntransfer also immer ein oder mehrere Quelltasks benötigt, deren Wissen auf einen Zieltasks übertragen werden soll[73].

1.3. Maschinelle Übersetzung

Eine weitere Problemstellung der NLP ist die maschinelle Übersetzung (MT), bei der die Aufgabe darin besteht, Texte durch Substitution von Wörtern und Phrasen von einer Sprache in eine andere zu übersetzen. Die MT beschäftigt die Wissenschaft schon mindestens seit den 80er Jahren und konnte in der jüngeren Zeit beachtliche Erfolge erzielen. MT-Systeme werden ebenso wie TK-Modelle mit überwachten ML-Verfahren trainiert. Maschinelle Übersetzungssysteme wie *Google Translate*[30], *Microsoft Translator*[9] oder der *DeepL Translator*[21] erzielen beeindruckende Ergebnisse und sind mit ihrer Übersetzungsqualität nur noch schwer von menschlichen Übersetzungen zu unterscheiden[52]. Ermöglicht wurde die Leistungssteigerung der letzten Jahre durch neue Übersetzungstechniken. Mit der Einführung der *Sequence-to-Sequence*-Architektur[93, 15] konnte die neuronale maschinelle Übersetzung (NMT) endgültig die bis dahin überlegene statistische maschinelle Übersetzung (SMT) ablösen. Neuronale Netze sind dann besonders effektiv, wenn ein umfangreicher Trainingsdatensatz für deren Training vorhanden ist. Für MT-Systeme werden parallele Trainingsdaten, also Zuordnung von Sätzen der einen Sprache auf entsprechende Übersetzungen der anderen Sprache, benötigt. Die jüngsten Erfolge der NMT-Systeme wurden nicht zuletzt auch dadurch möglich, dass über die Jahre große Trainingsdatensätze aus unterschiedlichen Domänen aufgebaut werden konnten.

1.4. Übertragung des Wissens von maschinellen Übersetzern

Das Thema dieser Arbeit ist es, den Einsatz von Wissenstransfertechniken in der Sprachverarbeitung zu untersuchen. Der Wissenstransfer ist ein in der Wissenschaft momentan intensiv diskutiertes Forschungsfeld. Die Fähigkeit, eine erlernte Aufgabe auf andere, vergleichbare Situationen zu übertragen, ist eine wichtige Voraussetzung für die Entwicklung intelligenter Systeme. Während diese Fähigkeit - die in der Psychologie unter dem Begriff Lerntransfer bekannt ist - dem Menschen intuitiv gelingt, fällt es maschinellen Lernverfahren oft noch schwierig, sich auf andere Tasks zu adaptieren. In dieser Arbeit wird im speziellen betrachtet, wie das in maschinellen Übersetzungssystemen gespeicherte Wissen auf Textklassifikatoren übertragen werden kann.

Dazu werden zwei unterschiedliche Problemszenarien besprochen, bei denen der Wissenstransfer genutzt werden kann:

- Zum einen wird untersucht, wie mit modernen maschinellen Übersetzern Systeme zur crosslingualen Textklassifikation (CTK) aufgebaut werden können. Die Aufgabe bei der CTK ist es, die Sprachbarriere beim Erlernen von TK-Tasks zu überwinden und aus den einsprachigen Trainingsdaten einer Quellsprache *A* Klassifikatoren für Texte einer anderen Zielsprache *B* zu trainieren.
- Zum anderen werden Methoden diskutiert, wie mit maschinellen Übersetzern die Fehlerrate eines (einsprachigen) TK-Systems reduziert werden kann.

Der Fokus der Arbeit liegt dabei auf der Klassifikation mit neuronalen Netzen (NNs), die aufgrund dessen, dass die gleichen Strukturen für verschiedene Tasks verwendet werden können und aufgrund ihres hierarchischen Aufbaus ein großes Potential für Transfermethoden eröffnen. Trotz des weiträumigen Erfolges von Neuronales Netz (NN)s in der Bildverarbeitung, insbesondere auch in der Bildklassifikation, werden nach dem Kenntnisstand des Autors für die Textklassifikation noch überwiegend traditionelle Lernverfahren eingesetzt. Dies lässt sich dadurch erklären, dass bei NNs eine Vielzahl von Hyperparametern festgelegt und für jeweilige Datensätze optimiert werden müssen. Aufgrund des relativ großen Hypothesenraums tendieren tiefe Netze bei wenigen Trainingsbeispielen zur Überanpassung (engl. Overfitting) und generalisieren folglich schlechter als Support Vektor Maschinen, naive Bayes- oder Nächste-Nachbarn-Klassifikatoren. In der Arbeit wird evaluiert, ob die oben genannten Transfermethoden einen Ausweg aus dem Overfitting-Problem aufzeigen und mit tiefen NNs auch bei spärlichen Trainingsdaten die Performance traditioneller Lernalgorithmen übertroffen werden kann. Der Einsatz von NN für die TK ist auch von praktischer Bedeutung, da diese aufgrund ihrer Fähigkeit zum selbständigen Lernen von Repräsentationen (engl. *representation learning*) weite Teile des *Feature-Engineerings* obsolet machen.

1.5. Beitrag

Der Beitrag dieser Arbeit ist es, verschiedene Methoden zur Übertragung des in maschinellen Übersetzern gespeicherte Wissen vorzustellen. Die verschiedenen Verfahren werden implementiert und im Anschluss daran einer quantitativen und qualitativen Evaluation unterzogen.

- Es wird ein Überblick über die bisherigen Arbeiten zum Thema CTK und Lerntransfer mit maschinellen Übersetzern gegeben.
- Es werden drei verschiedene Verfahren zur CTK vorgestellt, bei denen entweder zur Trainingszeit oder vor jeder Inferenz Übersetzungen generiert werden.
- Es werden zwei Verfahren zur Reduktion der Fehlerrate von (einsprachigen) Klassifikatoren präsentiert:
  - Ein neuartiges Trainingsdatenerweiterungsverfahren (TDE-Verfahren) für semantisch gelabelte Datensätze. Das Verfahren nutzt einen maschinellen Übersetzer, um Paraphrasen der Trainingsbeispiele zu generieren. Auf diese Art kann die Größe des Trainingsdatensatzes vervielfacht werden.

– Ein neuronales Verfahren, bei dem die internen Gewichte eines NMT-Systems auf die Eingangsschichten eines neuronalen Klassifikators übertragen werden.

Als Ausgangspunkt für die Klassifikation werden zunächst verschiedene neuronale Baselinemodelle ausgewertet und deren Leistung mit anderen, traditionellen Verfahren verglichen. Zur Evaluation der CTK-Verfahren wird eigens ein Datensatz mit deutsch- und englischsprachigen Trainings- und Testdaten erstellt. Es wird evaluiert, mit welchem der drei Verfahren die beste Leistung erzielt werden kann und wie hoch der Leistungsabfall beim Übergang der Sprachen ist. Die beiden Verfahren zur Reduktion der Fehlerrate werden auf 11 verschiedenen Datensätzen ausgewertet. Ziel der Analyse ist es, herausfinden, welche Leistungssteigerung durch den Wissenstransfer erzielt werden kann.

1.6. Aufbau

Im nächsten Kapitel (Kapitel 2) werden die grundlegenden Konzepte eingeführt, die für das weitere Verständnis der Arbeit benötigt werden. Dazu wird zunächst der Wissenstransfer formal definiert, die Funktionsweise künstlich neuronaler Netze (KNNs) erläutert und eine Überblick über die State-of-the-Art Techniken zur Textklassifikation und maschinellen Übersetzung gegeben. Danach wird in Kapitel 3 die zum Thema dieser Arbeit relevante Literatur vorgestellt. Im einzelnen werden die bedeutendsten Publikationen zu den Themenbereichen neuronale TK, Lerntransfer, crosslinguale Textklassifikation und Paraphrasierung besprochen. In Kapitel 4 werden schließlich Ansätze für den Wissenstransfer von maschinellen Übersetzern auf Textklassifikatoren diskutiert und in Kapitel 5 experimentell evaluiert. Abschließend werden in Kapitel 6 die Ergebnisse der Arbeit zusammengefasst und ein Ausblick auf noch offene Forschungsfragen gegeben.

2. Hintergrund

In diesem Kapitel werden die grundlegenden Konzepte für das Verständnis dieser Arbeit vermittelt. In Abschnitt 2.1 wird zunächst eine Einführung in die Grundlagen der neuronalen Netze gegeben. Der Themenschwerpunkt der Arbeit ist der Wissenstransfer von der maschinellen Übersetzung auf die Textklassifikation. Entsprechend wird in Abschnitt 2.2 zunächst der theoretische Hintergrund des maschinellen Lerntransfers vorgestellt, um im Anschluss in Abschnitt 2.3 und Abschnitt 2.4 die Problemstellungen und Verfahren der Textklassifikation und maschinelle Übersetzungen vorzustellen.

2.1. Künstliche Neuronale Netze

Künstliche neuronale Netze (KNNs oder kurz: NNs) sind ein durch die Funktionsweise des menschlichen Gehirns inspiriertes Berechnungsmodell. Sie sind Forschungsgegenstand des maschinellen Lernens (ML) und finden im Allgemeinen Anwendung bei der nichtlinearen Approximation von Funktionen. Sie werden als überwachtes Lernverfahren genutzt, um automatisiert mithilfe von Beispielen Lösungen für Probleme zu erlernen, ohne dass aufgabenspezifische Regeln programmiert werden müssen. Formal kann ein NN als eine Struktur mit internen Parametern θ aufgefasst werden:

$$y = KNN(x, \theta) \tag{2.1}$$

Beim überwachten Lernen, ein Teilgebiet des maschinellen Lernens, werden die Parameter θ durch die sukzessive Anpassung auf gelabelte Trainingsdaten trainiert. Dem Netz wird ein beliebiges Beispiel x aus der Menge der Trainingsdaten zugeführt und die Differenz zwischen der erwarteten Ausgabe z und der tatsächlichen Ausgabe y gemessen. Diese Differenz wird als Fehler des Netzes erachtet. Die Parameter werden dann so verändert, dass der Fehler minimiert wird. Der Vorgang wird so oft wiederholt bis das Netz mit der gewünschten Genauigkeit generalisiert, das heißt, die Zielfunktion auch für ungesehene Eingaben approximiert wird[24]. Es folgt nun ein Überblick über die historische Entwicklung der neuronalen Netze und deren internen Aufbau.

Ihren Ursprung haben die NN in der Arbeit „A Logical Calculus of the Ideas Immanent in Nervous Activity“[65] von Warren McCulloch und Walter Pitts aus dem Jahre 1943, die nach einem vereinfachten Modell zur mathematischen Beschreibung von Vorgängen im menschlichen Gehirn suchten. Sie zeigten, dass mit ihrem Netzwerk aus Neuronen ähnlichen Elementen praktisch jede logische Funktion berechnet werden kann. Das Modell inspirierte daraufhin viele weitere Forscher auf dem Gebiet der künstlichen Intelligenz (KI). Einer von ihnen war der Psychologe und Informatiker Frank Rosenblatt, der schließlich 1958 das Perzeptron-Modell vorstellte[81], dass das Modell von McCulloch und Pits um eine nichtlineare Ausgabefunktion erweiterte. Sein Modell ist bis heute der Grundbaustein der neuronalen Netze. Nach dem

anfänglichen Enthusiasmus für das Perzeptron stellte die mathematische Analyse von Marvin Minsky und Seymour Papert in [68] einen gravierenden Einschnitt dar. Sie konnten darlegen, dass mit der Deltaregel, dem Perzeptron-Lernverfahren, nur linear separierbare Probleme gelöst werden können, woraufhin die Erforschung der NNs erst einmal zum Erliegen kam. Mit neuen Lernverfahren, insbesondere dem Backpropagation-Algorithmus von Paul Werbos[80], und den Fortschritten in der Hardwaretechnik, durchlebten NNs in der Wissenschaft seither mehrere Höhen und Tiefen. Heute spielt das biologische Vorbild der Netze oft nur noch eine untergeordnete Rolle. In der jüngeren Zeit erleben neuronale Verfahren eine Wiedergeburt, da sie bei herausfordernden Anwendungen in der Spracherkennung und Text- und Bildverarbeitung bessere Ergebnisse als konkurrierende Lernverfahren liefern.

2.1.1. Feedforward Netze

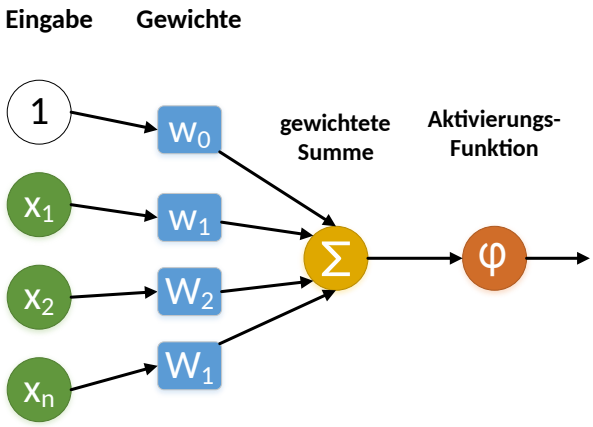


Abbildung 2.1.: Modell eines einfachen Neurons (Perzeptron).

Neuronale Netze werden aus mehreren Knotenpunkten zusammengesetzt. Die Knoten imitieren biologische Nervenzellen und werden entsprechend als Neuronen bezeichnet. Die Neuronen werden nach dem Perzeptron-Modell modelliert (siehe Abbildung 2.1). Jedes Neuron nimmt mehrere Eingabewerte entgegen und erzeugt aus ihnen einen Ausgabewert, der an nachfolgende Neuronen weitergegeben wird. Die Relevanz der Eingabewerte für den Ausgabewert wird durch Gewichte festgelegt. Die Ausgabe ist das Ergebnis der gewichteten Eingangssignale angewandt auf eine Aktivierungsfunktion. Gegeben sei ein Neuron mit n Eingänge $x_1, \ldots, x_n \in \mathbb{R}$ mit Gewichten $w_1, \ldots, w_n \in \mathbb{R}$ und Aktivierungsfunktion φ ergibt sich die Ausgabe o aus:

$$o = \varphi(\sum_{i=0}^{n} x_i w_i) = \varphi(x \cdot w) \tag{2.2}$$

Mit $x_0 \equiv 1$ wird w_0 als Schwellwert (engl. *Bias*) bezeichnet. Die Aktivierungsfunktion φ ist eine nichtlineare Funktion und wird benötigt, damit mit dem Netz auch nichtlineare Probleme gelöst werden können. Für den Backpropagation-Algorithmus werden differenzierbare Funktionen vorausgesetzt. Es kommen eine ganze Reihe unterschiedlicher Aktivierungsfunktionen in Frage und die Entscheidung wird zumeist anhand von empirischen Erfahrungswerten festgemacht. Häufig verwendet werden beispielsweise die Sigmoidfunktion oder der Tangens hyperbolicus.

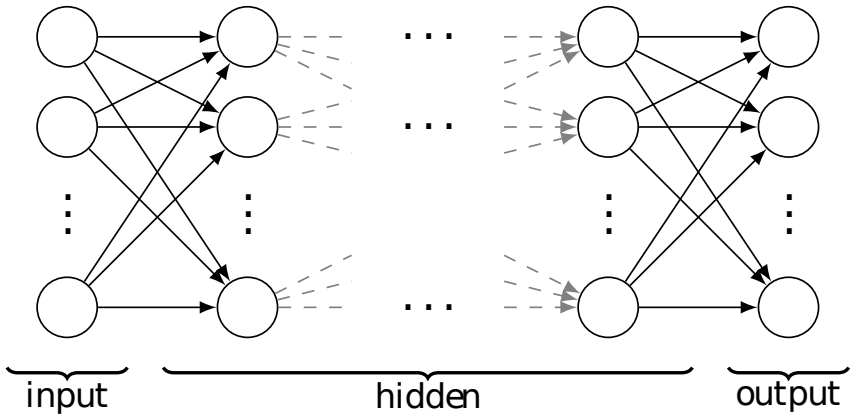


Abbildung 2.2.: Aufbau eines Feedforward Netzes.

Die Neuronen werden in einer festen Hierarchie miteinander verbunden und sind schichtweise in sogenannten *Layern* angeordnet. Die für ein Netz zu trainierenden Parameter θ sind die Gewichtsvektoren und Biases der einzelnen Neuronen. In der Regel werden nur die Neuronen aufeinanderfolgender Schichten verknüpft (Inter-Neuronlayer-Connection), in selteneren Fällen aber auch innerhalb eines Layers (Intra-Neuronlayer-Connection). Man unterscheidet zwischen drei verschiedenen Arten von Layern. Die *Input-Layer* enthält keine Neuronen im eigentlichen Sinne, sondern trägt lediglich die Netzeingabe. Die *Output-Layer* präsentiert die Netzausgabe. *Hidden-Layer* bezeichnen die verdeckten Schichten zwischen Input- und Output-Layer. Abbildung 2.2 zeigt den Aufbau eines einfachen NNs. Da Neuronen immer nur mit nachfolgenden Schichten verbunden werden, das heißt keine Rückwärtskanten vorhanden sind, wird das Netz als *Feedforward Network* bezeichnet. Wird in einer Schicht jedes Neuron mit jedem Neuron der vorhergehenden Schicht vermascht, spricht man von einer Fully-Connected-Layer (FC-Layer). In den letzten Jahren ist die Anzahl der Schichten von neuronalen Architekturen rasant angestiegen. Man spricht deshalb auch von *Deep Neural Networks* und für das Training dieser ist der Begriff des *Deep Learnings* gebräuchlich. Neben den Feedforward Netzen existieren noch weitere Modell-Architekturen wie die Convolutional Neural Networks (CNNs) und die Recurrent Neural Networks (RNN). Sie werden später in Abschnitt 3.3 beschrieben.

2.1.2. Backpropagation

Das am weitesten verbreitete Verfahren zum Trainieren der Parameter von neuronalen Netzen ist die Fehlerrückführung (engl. *Backpropagation*, kurz: BP). Der Algorithmus wurde 1974 von Paul Werbos im Zuge seiner Dissertation entwickelt[80]. Seinen heutigen Stellenwert erlangte Backpropagation (BP) aber erst durch [84], wo experimentell gezeigt werden konnte, dass durch das Training von Netzen mit BP nützliche Repräsentationen der Eingabedaten generiert werden. Zudem stellten die Autoren fest, dass das Verfahren effizienter lernt als frühere Lernverfahren, wodurch Aufgaben, die zuvor noch zu komplex waren, lösbar wurden.

Im Laufe der Zeit haben sich viele Varianten und Optimierungen des BP-Algorithmus etabliert. Die Standardttechnik ist die stochastische Fehlerrückführung (engl. *Backpropagation with Stochastic Gradient Descent*, kurz: SGD), welche in [24] detailliert beschrieben wird. Die

Gewichte werden zufällig initialisiert und während der Trainingsroutine angepasst. Der Algorithmus läuft in den folgenden Phasen ab:

1. Ein beliebiges Eingabemuster x aus der Trainingsmenge wird angelegt und vorwärts durch das Netz propagiert.

2. Die Ausgabe des Netzes y wird mit der gewünschten Ausgabe z verglichen. Die Differenz der beiden Werte wird als Fehler des Netzes erachtet.

3. Der Fehler wird nun wieder über die Ausgabe- zur Eingabeschicht zurück propagiert. Dabei werden die Gewichtungen der Neuronenverbindungen abhängig von ihrem Einfluss auf den Fehler geändert, so dass bei einem erneuten Anlegen der Eingabe eine Annäherung an die gewünschte Ausgabe eintritt.

Für die Berechnung des Fehler wird eine Fehlerfunktion (engl. *Loss function*) l festgelegt:

$$l = L(y, z) \tag{2.3}$$

Je nach Aufgabe werden unterschiedliche Funktionen L in Betracht gezogen. Für die Klassifikation wird der Mean Squared Error (L2 loss) oder die Kreuzentropie (Crossentropy loss) genutzt. Für Regressionstasks wird häufig der absolute Fehler (L1 loss) bestimmt. Damit der Fehler im dritten Schritt des Algorithmus zurück propagiert werden kann, muss der Gradient der Netzparameter berechnet werden. Dies setzt voraus, dass das Netz, insbesondere die Aktivierungsfunktionen, differenzierbar sind. Zur Anpassung der Parametern wird der Gradient mit einer Lernrate lr multipliziert und zu den Parameter hinzu addiert. Eine Variante des Stochastic gradient descent (SGD)-Verfahrens ist das sogenannte Mini-Batch SGD. Anstatt die Fehler für jedes Trainingsbeispiel einzeln zu berechnen, wird der Gradient jeweils für eine kleine Teilmenge der Trainingsbeispiele bestimmt. Dies verringert zum einen die Trainingszeit, da weniger Iterationen zum Durchlaufen aller Beispiele benötigt werden, zum anderen wird der Gradient dadurch stabiler, wodurch das Netz besser generalisiert.

BP kann nicht nur für klassisiche Feedforward Netze, sondern auch für andere Netzarchitekturen wie Convolutional- und Rekurrente Netze angewandt werden. Ein Überblick über weitere Optimierungsmethoden gibt [83]. In dieser Arbeit wird Adaptive Moment Estimation (Adam)[49] verwendet, ein Algorithmus der sich vor allem durch adaptive Lernraten für einzelne Parameter auszeichnet.

2.1.3. Regularisierung

Die hohe Flexibilität beim Entwurf von neuronalen Architekturen erlaubt es, komplexe Modelle mit vielen Parametern zu entwickeln. Mit wachsender Kapazität der Netze steigt aber auch die Gefahr der Überanpassung (engl. *Overfitting*) des Modells auf die Trainingsdaten. Wird der Fehler auf den Trainingsdaten zu stark minimiert, so beginnt das Netz in kleinen Variationen des Eingabesignals (z.b. Rauschen) Muster zu erkennen, die nicht durch das darunterliegende Modell erklärt werden können. Die Trainingsdaten werden auswendig gelernt. Ein überangepasstes Modell generalisiert schlecht und macht bei der Prädiktion von unbekannten Eingabesignalen große Fehler.

Zur Vorbeugung von Overfitting gibt es zwei Herangehensweisen. Können zusätzliche gelabelte Daten leicht generiert werden und ist genügend Rechenleistung für das Training mit mehr Daten verfügbar, sind zusätzliche Trainingsdaten die beste Abhilfe. Anderenfalls muss die Kapazität des Modells reduziert werden. Das geschieht meist durch Kombination der folgenden Techniken:

1. Verändern der Netzarchitektur: Reduzierung der Schichten und der Anzahl der Neuronen pro Schicht.
2. Frühzeitiges Stoppen des Trainings
3. Gewichtsabfall (engl. *Weight Decay*)
4. Hinzufügen von Rauschen zu der Fehlerfunktion

Für den frühzeitigen Abbruch des Trainings, muss festgestellt werden, zu welchem Zeitpunkt die Overfitting-Effekte auftreten. Dazu müssen die Daten neben der Trainings- und Testmenge in eine weitere Validierungsmenge unterteilt werden. Während dem Training mit den Trainingsdaten wird kontinuierlich die Performance bei der Prädiktion der Validierungsdaten gemessen. Bleibt die gemessene Leistung über mehrere Epochen konstant oder nimmt ab, so ist dies ein Indiz für den Beginn einer Überanpassung des Modells und das Training wird gestoppt.

Beim Gewichtsabfall, auch unter dem Begriff $L1$- oder $L2$-Regularisierung bekannt, wird der Kostenfunktion ein sogenannter Regularisierungsterm hinzugefügt. Er ergibt sich aus der Summe der Absolutwerte aller Gewichte ($L1$-Regularisierung) oder der Summe der Quadrate aller Gewichte ($L2$-Regularisierung).

$$L1 = \sum_w |w|, \quad \text{und} \quad L2 = \sum_w w^2 \tag{2.4}$$

Dadurch ergibt sich mit Gleichung 2.3 die Kostenfunktion:

$$l = L(y, z) + \lambda_1 * L1 + \lambda_2 * L2 \tag{2.5}$$

Mit den Parametern λ_1 und λ_2 wird die Bedeutung der Regularisierungsterme festgelegt. Bei der Minimierung der Fehlerfunktion wird dadurch versucht, ein Kompromiss zwischen kleinen Gewichten und einem Minimum des Netzfehlers zu finden. Dies hat zur Folge, dass beim Training die Suche nach einfachen Modell forciert wird. Zwar garantieren kleine Gewichte noch keine höhere Performance, allerdings hat sich empirisch gezeigt, dass eine derartige Regularisierung der Gewichte vor allem bei kleinen Datensätzen zu einer besseren Performance führt.

Durch das Einfügen von Dropout-Schichten in die Netzarchitektur wird das Rauschen in den Trainingsdaten verstärkt[90]. Ein bestimmter Prozentsatz der Hiddenneuronen wird während des Trainings vor jeder Inferenz zufällig ausgeschaltet, mit der Folge, dass keine Ausgabe an die nachfolgenden Neuronen weitergegeben werden kann. Das Netz wird dadurch dazu gezwungen, beim Training Redundanzen für die ausgefallenen Neuronen zu erlernen, wodurch sich die Neuronen nicht so sehr auf die Trainingsdaten anpassen können. Der Dropout-Faktor ist ein weiterer Hyperparameter zum Steuern der Kapazität des Netzes und muss auf die Zahl der Trainingsdaten abgestimmt werden.

2.2. Maschineller Lerntransfer

Der Lerntransfer (engl. *Transfer learning*) beschreibt eine Technik des maschinellen Lernens, bei dem ein Model welches für eine Aufgabe *A* trainiert wurde, als Ausgangspunkt für das Modell einer zweiten Aufgabe *B* wiederverwendet wird. Das Ziel ist es, das beim Training der einen Aufgabenstellung gewonnene Wissen auszunutzen, um beim Erlernen der zweiten Aufgabe schneller und besser zu Generalisieren. Auch wenn der Lerntransfer sich wegen den vielen Trainingsparametern und den komplexen und großen Datensätzen in den letzten Jahren vor allem auf dem Gebiet des *Deep Learnings* großer Beliebtheit erfreut, so beschränkt sich der Anwendungsbereich nicht nur auf das Training von neuronalen Netzen sondern auch auf andere ML-Methoden.

Eine umfangreiche Analyse des gegnwärtigen Forschungsstand und verschiedene Anwendungsszenarien des Transferlernens findet sich in [97]. Es gibt keine allgemeingültige Kriterien, um vor der Entwicklung eines Systems beurteilen zu können, ob der Transfer von Wissen aus einer Domäne für ein Modell einer anderen Domäne einen positiven Nutzen hat. Es ist daher nur schwer Abzuschätzen, ob sich die Verwendung von Transfermethoden für einen Lerntask lohnt. Eine Beurteilung ist immer erst nach der Entwicklung und Evaluation des Modells möglich.

Um das Prinzip des Transferlernens besser verstehen zu können, werden Kenntnisse der dahinterliegenden Konzepte benötigt. Aus diesem Grund folgt nun eine formale Definition, welche weitläufig aus der Studie von S. Pan und Q. Yang[73] entnommen wurde:

2.2.1. Formale Definition

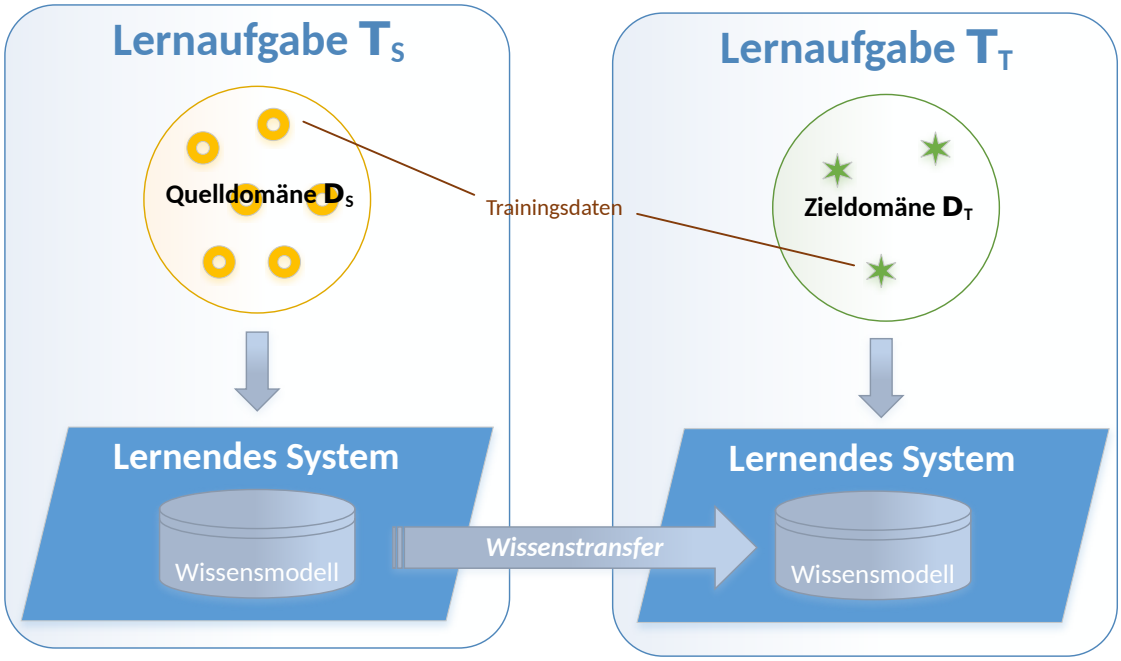


Abbildung 2.3.: Schematische Darstellung: Wissenstransfer

Für eine Beschreibung des Lerntransfers müssen zunächst einmal die beiden zentralen Konzepte *Domäne* und *Lernaufgabe* (engl. Task) eingeführt werden. Dies soll Beispielhaft anhand einer Klassifikationsaufgabe erklärt werden, bei der Texte entsprechend ihrer inhaltlichen Bedeutung einer festen Kategorie zugeordnet werden. Im Beispiel sollen Beschreibungstexte von Nachrichtenartikeln ihrem Ressort zugeteilt werden. Die Ressorts sind *Politik*, *Wirtschaft*, *Wissenschaft* und *Sport*. Der Text „*Außenminister Lawrow kündigte laut Nachrichtenagentur Ria an, britische Diplomaten müssten Russland verlassen.*" soll dem Ressort *Politik* zugeordnet werden. „*Barcelona steht im Viertelfinale der Champions League.*" fällt in die Kategorie *Sport*.

Als *Domäne* werde im weiteren Verlauf dieser Arbeit ein Tupel $\mathcal{D} = (\mathcal{X}, P(X))$ bezeichnen. Es bestehe aus dem Merkmalsraum $\mathcal{X}$ und $P(X)$ einer Randverteilung über $\mathcal{X}$ mit $X = (x_1, \ldots, x_n) \in \mathcal{X}$. Bei dem Beispiel zur Kategorisierung entspricht der Merkmalsraum der Menge aller darstellbaren Textrepräsentation. Werden Texte nach dem BoW-Modell kodiert, liegen in $\mathcal{X}$ alle Wortfrequenz-Vektoren. X ist die Menge der Trainingsbeispiele und x_i ist der Wortfrequenz-Vektor des *i-ten* Trainingsbeispiels.

Eine Lernaufgabe (engl. Task) $\mathcal{T} = (\mathcal{Y}, f(\cdot))$ bestehe aus einer $\mathcal{D} = (\mathcal{X}, P(\mathcal{X}))$, aus einem *Labelraum* $\mathcal{Y}$ und einer *prädiktiven Zielfunktion* $f(\cdot)$. Die Zielfunktion kann nicht direkt beobachtet werden, sondern wird üblicherweise aus den Trainingsdaten erlernt. Trainingsdaten werden als Menge von Paaren (x_i, y_i) mit $x_i \in X$ und $y_i \in \mathcal{Y}$ definiert. Mit der Funktion $f : \mathcal{X} \rightarrow \mathcal{Y}$ können die Label $f(x)$ neuer Instanzen x prädiziert werden. Aus wahrscheinlichkeitstheoretischer Sicht beschreibt sie die bedingte Wahrscheinlichkeitsfunktion $f(x) = P(y|x)$. Bei der Kategorisierung ist $\mathcal{Y}$ die Menge aller Kategorien, d.h. $\mathcal{Y} = \{\text{Politik, Wirtschaft, Wissenschaft, Sport}\}$

Nun kann Wissenstransfer in Hinblick auf das maschinelle Lernen folgendermaßen definiert werden:

Definition 1 *(Lerntransfer) Gegeben sei eine Quelldomäne $\mathcal{D}_S$ mit zugehöriger Lernaufgabe $\mathcal{T}_S$ sowie eine Zieldomäne $\mathcal{D}_T$ mit Lernaufgabe $\mathcal{T}_T$. Das Ziel beim Wissenstransfer ist es, das Erlernen der prädiktiven Zielfunktion $f_T(\cdot)$ in $\mathcal{D}_T$ unter Verwendung des Wissens aus $\mathcal{D}_S$ und $\mathcal{T}_S$ zu verbessern, wobei $\mathcal{D}_S \neq \mathcal{D}_T$ oder $\mathcal{T}_S \neq \mathcal{T}_T$ gelten muss.*

Für das Verständnis dieser Arbeit ist nur das Szenario von Relevanz, in dem das Wissen aus genau einer Quelldomäne $\mathcal{D}_S$ auf eine Zieldomäne $\mathcal{D}_T$ übertragen wird. Die Trainingsdaten der Quelldomäne werden mit $D_S = \{(x_{S_1}, y_{S_1}), \cdots, (x_{S_{n_S}}, y_{S_{n_S}})\}$ bezeichnet, wobei $x_{S_i} \in \mathcal{X}_S$ für die Trainingsinstanz der Quelldomäne und $y_{S_i} \in \mathcal{Y}$ für die dazugehörigen Labels steht. Analog bezeichne $D_T = \{(x_{T_1}, y_{T_1}), \cdots, (x_{T_{n_T}}, y_{T_{n_T}})\}$, $x_{T_i} \in \mathcal{X}_T$ und $y_{T_i} \in \mathcal{Y}$ die Trainingsdaten der Zieldomäne.

In den meisten Fällen soll durch den Transfer das Erlernen der Zielfunktion $f_T(\cdot)$ bei stark eingeschränkten Trainingsdaten in der Zieldomäne $\mathcal{D}_T$ verbessert werden. Häufig ist also die Menge an gelabelten Trainingsdaten im Vergleich zu der Zahl der vorhandenen Beispiele der Quelldomäne exponentiell kleiner. Es gilt also: $0 \leq n_T \ll n_S$.

Wenn eine explizite oder implizite Beziehung zwischen den Merkmalsräumen zweier Domänen besteht, wird davon gesprochen, dass die beiden Domänen miteinander verwandt (engl. related) sind.

Sind Quell- und Zieldomäne sowie Quell- und Zielaufgabe identisch, so wird das Problem zu einem klassischen ML-Problem ohne Wissenstransfer. Da $\mathcal{D} = (\mathcal{X}, P(\mathcal{X}))$ und $\mathcal{T} = (\mathcal{Y}, P(Y|X))$ jeweils als Tupel definiert sind, hat die Bedingung $\mathcal{D}_S \neq \mathcal{D}_T$ oder $\mathcal{T}_S \neq \mathcal{T}_T$ in Definition 1 zur

Folge, dass es vier unterschiedliche Szenarien für den Wissenstransfer gibt. Im Bezug auf das Kategorisierungsproblem können die Szenarien wie folgt interpretiert werden:

1. $\mathcal{X}_S \neq \mathcal{X}_T$: Der Merkmalsraum von Quell- und Zieldomäne ist unterschiedlich. Beispielsweise sind die Texte der beiden Domänen in unterschiedlicher Sprache (Crosslinguale Textklassifikation).

2. $P(X_S) \neq P(X_T)$: Die Randverteilung der Quell- und Zieldomänen sind unterschiedlich. Man spricht von einem Domänen Adaptionsproblem (engl. Domain Adaption Problem). Zum Beispiel wurden die Trainingsdaten aus verschiedenen Quellen zusammengestellt.

3. $\mathcal{Y}_S \neq \mathcal{Y}_T$: Die Labelräume der beiden Lernaufgaben sind unterschiedlich. Ist dies der Fall gilt für gewöhnlich ebenfalls $P(Y_S|X_S) \neq P(Y_T|X_T)$ (Fall 4). Dieses Szenario entspricht dem Thema dieser Arbeit, wenn das Wissen eines maschinellen Übersetzers (Quelltask) auf einen Textklassifikator (Zieltask) übertragen werden soll.

4. $P(Y_S|X_S) \neq P(Y_T|X_T)$: Die bedingten Wahrscheinlichkeiten (bzw. die prädiktiven Zielfunktionen) sind unterschiedlich. Ein häufig vorzufindendes Szenario in der Datenanalyse, bei dem durch Oversampling oder Undersampling eine inbalance zwischen den Klassen kompensiert werden soll.

2.3. Textklassifikation

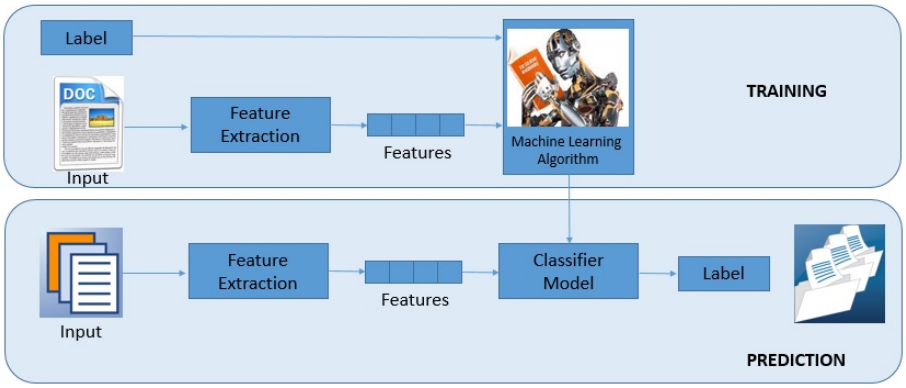


Abbildung 2.4.: Übersicht der Textklassifikation. Quelle: [92]

In diesem Abschnitt werden die Grundlagen der Textklassifikation (TK) vorgestellt. Die Aufgabe der TK besteht darin, Dokumente in Abhängigkeit ihres semantischen Inhaltes einer oder mehreren Kategorien oder Klassen zuzuordnen. Zur Automatisierung der TK werden Methoden des maschinellen Lernens angewandt. In dieser Arbeit wird die TK als überwachtes Lernproblem aufgefasst, für dessen Lösung gelabelte Trainingsbeispiele verfügbar sind. Ein häufig anzutrefendes Beispiel für ein überwachtes TK-Problem ist die Entwicklung eines Spamfilter, der eingehende E-Mails auf Grundlage deren Inhalts in fest vorgegebene Kategorien einsortiert. Abbildung 2.4 gibt einen Überblick über die Phasen in der Entwicklung eines TK-Systemen. Man unterscheidet zwischen der Lernphase und der Prädiktionsphase. Die

Wissenschaft fokussierte sich lange auf die Erforschung neuer Lernverfahren. Dem Training geht jedoch immer auch eine Vorverarbeitung voraus, die bis zu 80% der Entwurfszeit und der Rechenleistung in Anspruch nimmt[87]. Bevor ein Dokument Klassifiziert werden kann, muss zunächst durch *Feature-Engineering* eine geeignete numerische Repräsentation der Texte gefunden werden.

Die traditionellen Lernalgorithmen für die TK sind Entscheidungsbäume[57], (k-Nächste-Nachbarn (KNN)[95], naive Bayes (NB)[63] und Support Vektor Maschinen (SVMs)[45]. In einem anderen Anwendungsbereich des maschinellen Lernens, der Bildverarbeitung, wurden diese Verfahren in den letzten Jahren nahezu vollständig durch neuronale Klassifikatoren abgelöst. NNs bieten den Vorteil, dass sie beim Training selbständig die relevanten Merkmale aus dem Eingangssignal herleiten können und der Aufwand für die Vorverarbeitung der Daten drastisch sinkt. Neuronale TK-Architekturen, die im Fokus dieser Arbeit stehen, werden später in Abschnitt 3.3 eingeführt.

2.3.1. Formale Problembeschreibung

Sei $C = c1, \ldots, c_m$ eine Menge von Kategorien (Klassen) und $D = d_1, \ldots, d_n$ eine Menge von Texten (Dokumenten). Die Textklassifikation kann formal als die Aufgabe aufgefasst werden, eine unbekannte Zielfunktion $f : D \times C \rightarrow \{0, 1\}$ zu approximieren. Die Ausgabe von f entspricht der Klassifikation eines (menschlichen) Domänenexperten. Jedes Dokument wird durch eine Menge von Merkmalen repräsentiert. Dies sind für gewöhnlich die im Dokument auftretende Worte (siehe Unterabschnitt 2.3.2). Für die Approximation von f muss jedem Paar $(c_i, d_j) \in C \times D$ (mit $1 \leq i \leq m$ und $1 \leq j \leq n$) ein Wert 0 oder 1 zugeordnet werden, d.h. den Wert 0, wenn das Dokument d_j nicht zur Klasse c_i gehört, ansonsten 1.

2.3.2. Wortrepräsentation

Zur Verarbeitung von Sprache durch Computersysteme müssen Texte als numerische Werte dargestellt werden. Idealerweise repräsentiert die mathematische Darstellung eines Wortes auch die Bedeutung dieses Wortes im Kontext anderer Wörter.

One-Hot-Kodierung Der einfachste Ansatz zur Darstellung von Wortsequenzen ist die One-Hot-Kodierung: Der Textkorpus wird Bereinigt und ein Wörterbuch aller vorkommenden Wörter erstellt. Die Wörter werden dann absteigend nach deren Häufigkeit sortiert und jedem Wort wird eine Zahl zugeordnet. Die Zahl Null bleibt für Wörter vorbehalten, welche nicht im Wörterbuch vorhanden sind (Out-of-Vocabulary-Words, kurz: OOV-Words). Die Zahl 1 repräsentiert dann das Wort, welches am häufigsten im Trainingskorpus vorkommt, die Zahl 2 das zweithäufigste Wort, und so weiter. Der Nachteil dieser Darstellungsform liegt darin, dass die Zahl in keinem Zusammenhang zur Bedeutung der Worte steht. Als Eingabe für ein neuronales Netz kann daher pro Wort nur ein Vektor in der Länge des Wörterbuchs verwendet werden. Bei großen Korpussen erreichen diese One-Hot-Kodierten Vektoren schnell eine sehr große Länge, bei denen jeweils nur eine Eins und sonst nur Nullen enthalten sind. Solche sogenannten sprälichen Vektoren (engl. *Sparse Vectors*) sorgen für eine hohe Komplexität der Lernaufgabe und erschweren den Lernverfahren, die relevanten Muster in den Daten zu erkennen. Dies kann mit anderen Methoden vermieden werden.[5]

Word Embeddings Word Embeddings sind real-wertige Vektoren zur Repräsentation von Worten. Je nach Modell haben die Vektoren zwischen 100 und 300 Dimensionen und sind damit deutlich wenig komplex als die zuvor beschriebenen One-Hot-Kodierungen. Der entscheidende Vorteil der Embeddings ist jedoch, dass die Vektoren die semantische Bedeutung von Worten kodieren. Worte mit einer ähnlichen Bedeutung werden auf eine ähnliche Repräsentation abgebildet. Trainiert werden die Embeddings mit unüberwachten Lernverfahren, bei denen auf großen textuellen Datensätzen der Kontext der Worte analysiert wird. Die theoretische Grundlage dahinter ist die „Verteilten Hypothese" des Linguisten Z. Harris[33], wonach Worte die in einem ähnlich Kontext verwendet werden auch eine ähnliche Bedeutung haben. Namenhafte Beispiele für Algorithmen zur Berechnung von Embeddings sind *Word2Vec*[67] oder *GloVe*[76].

Abbildung 2.5 veranschaulicht beispielhaft die von Word2Vec erlernten Konzepte und die Beziehungen zwischen diesen. Durch die Beziehungen wird es möglich, einfache arithmetische Operationen auf dem Vektorraum der Embeddings zu definieren. Beispielsweise gilt:

$$emb(Koenig) - emb(Mann) + emb(Frau) \approx emb(Koenigin) \tag{2.6}$$

ebenso funktioniert:

$$emb(Berlin) - emb(Deutschland) + emb(Frankreich) \approx emb(Paris) \tag{2.7}$$

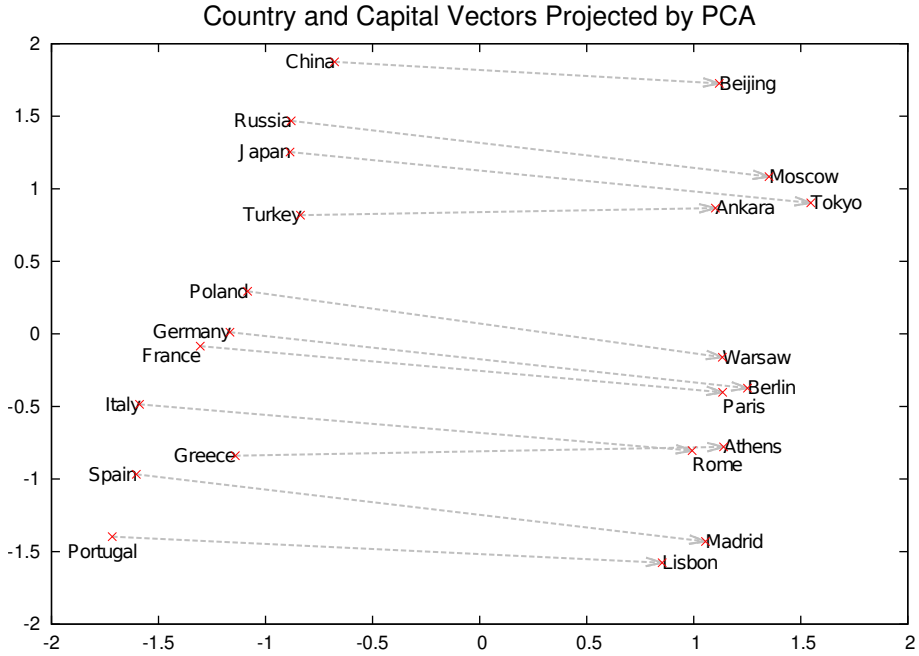


Abbildung 2.5.: Zweidimensionale PCA-Projektion der Word Embeddings von Ländern und deren Hauptstädte. Quelle: [67].

Überwachte Word Embeddings Die überwachten Word Embeddings sind eine Variante der Wortvektoren, wie sie Beispielsweise von den gängigen Deep Learning Frameworks wie Tensorflow oder Keras unterstützt werden. Sie werden gemeinsam mit dem neuronalen Modell eines

nachgelagerten NLP-Tasks (z.B. Klassifikation) trainiert. Eine am unteren Ende des neuronalen Modells eingefügtes *Embedding Layer* lernt, die interne Darstellung jedes Wort möglichst so zu wählen, dass er möglichst nah an den Klassen liegt, in denen das Wort häufig vorkommt und weniger nah an Klassen, in denen es wenig oder gar nicht vorkommt. Der so gelernte Wortrepräsentation enthält also Informationen darüber, wie wahrscheinlich es ist, dass dieses Wort im Rahmen einer Klasse verwendet wird. Als Eingabe der Schicht dienen die One-Hot-Kodierten Wortrepräsentationen der Eingabesequenz. Wie in Abschnitt 5.2 zu sehen sein wird, können unüberwacht trainierte Embeddings auch als Ausgangspunkt zur Initialisierung der Gewichte der *Embedding Layer* genutzt werden, um die GloVe oder Word2Vec Vektoren mit spezifischen Informationen der nachgelagerten Aufgabe anzureichern (Feintuning).

2.3.3. Metriken

Die Leistung eines Klassifikators wird daran gemessen, wie viele Fehler bei der Klassifikation von ungesehenen Testdaten gemacht werden. Die Testdaten müssen ebenso wie die Trainingsdaten gelabelt werden, um die Ausgabe mit einer gegebenen Referenz (Ground truth) vergleichen zu können. Für aussagekräftige Ergebnisse ist es notwendig, dass die Menge der Testdaten nicht für das Training verwendet wurden.

Bei der Bewertung der Fehler eines binären Klassifikators kann zwischen vier unterschiedlichen Fällen unterschieden werden. Diese Fehlertypen werden anhand des Geschäftssignals *Führungswechsel*, bei dem erkannt werden soll, ob in einem Text der Wechsel von Führungspersonal einer Firma thematisiert wird oder nicht, erläutert.

1. **Richtig positiv** (kurz: tp; engl. true positive): Der Klassifikator erkennt einen Führungswechsel. Laut Referenz ist dies korrekt und im Text wird ein Führungswechsel thematisiert.

2. **Richtig negativ** (kurz: tn; engl. true negative): Der Klassifikator erkennt keinen Führungswechsel. Laut Referenz ist dies korrekt und im Text wird kein Führungswechsel thematisiert.

3. **Falsch positiv** (kurz: fp; engl. false positive): Der Klassifikator erkennt einen Führungswechsel. Laut Referenz ist dies falsch und im Text wird kein Führungswechsel thematisiert.

4. **Falsch negativ** (kurz: fn; engl. false negative): Der Klassifikator erkennt keinen Führungswechsel. Laut Referenz ist dies falsch und im Text wird ein Führungswechsel thematisiert.

Ausgehend von den relativen Häufigkeiten der aufgetretenen Fehler können quantitative Maße für die Beurteilung des Klassifikators berechnet werden. Eines davon ist die Accuracy (deutsch: Treffergenauigkeit). Sie gibt den Anteil korrekt klassifizierter Objekte an der Gesamtheit aller Objekte an.

$$Acc = \frac{tp + tn}{tp + tn + fp + fn} \tag{2.8}$$

Die Accuracy ist ein gutes Fehlermaß, wenn in den Daten ein Gleichgewicht zwischen den positiven und negativen Fällen vorliegt. Bei extrem unbalancierten Klassifikationsproblemen ist sie jedoch keine adäquate Metrik, da die True postives und True negatives im Nenner gleichstark gewichtet werden. In der Realität handeln die meisten Texte nicht von Führungswechseln. Ein Klassifikator der Texte ausschließlich der negativen Klasse zuweist, kann eine Accuracy nahe 1 erzielen. Intuitiv hat in einem solchen Fall die korrekte Klassifikation positiver Beispiele (True positives) einen höheren Stellenwert für die Beurteilung des Klassifikators als die korrekte Klassifikation der negativen Beispiele (True negatives). Es gilt also den statistischen Recall (deutsch: Trefferquote) zu maximieren. Er gibt den Anteil der positiv klassifizierten Objekte an der Gesamtheit der tatsächlich positiven Objekte an.

$$Recall = \frac{tp}{tp + fn} \tag{2.9}$$

Würde der Führungswechselklassifikator alle Texte positiv klassifizieren, so erzielte er den bestmöglichen Recall von 1. Das Modell wäre aber vermutlich nicht sehr nützlich für die Lösung des Problems. Dies lässt sich statistisch daran erkennen, dass die Steigerung des Recalls auf Kosten eines anderes Maßes - der Precision - geschah. Die Precision (deutsch: Genauigkeit) gibt den Anteil der korrekt als positiv klassifizierten Ergebnisse an der Gesamtheit der als positiv klassifizierten Ergebnisse an.

$$Precision = \frac{tp}{tp + fp} \tag{2.10}$$

Das erste oben genannte Modell, das alle Texte negativ klassifiziert, hat eine Precision und einen Recall von 0, da keine True positives existieren. Werden alle Objekte positiv klassifiziert, liegt der Recall bei 1, die Precision ist dagegen aber sehr gering. Ein anderes Extrembeispiel wäre ein Modell, das genau ein positives Objekt richtig klassifiziert und alle anderen negativ. Es könnte eine Precision von 1 erzielen, wiederum wäre aber der Recall nahe 0. Für ein möglichst gutes Modell gilt es insgesamt, einen Trade-off zwischen Precision und Recall zu finden. Um die Güte an einer einzigen Kennzahl festzumachen, kann ein Maß genutzt werden, das Precision und Recall miteinander kombiniert.

$$F_1 = \frac{2 * (precision * recall)}{precision + recall} \tag{2.11}$$

2.4. Maschinelle Übersetzung

In diesem Abschnitt werden die Grundlagen der maschinellen Übersetzung (engl. machine translation, kurz: MT) vorgestellt. Die Aufgabenstellung der MT ist die automatische Suche nach der bestmöglichen Übersetzung eines Textes in einer Zielsprache, gegeben ein Text aus einer Quellsprache. Die Schwierigkeit bei der Übersetzung ist es, fließende und möglichst natürlich klingende Texte zu generieren. Für die Übersetzung ist es notwendig, den Text in seiner Gesamtheit zu interpretieren und zu analysieren. Dafür wird Expertenwissen über Grammatik, Syntax (Satzstruktur) und Semantik (Bedeutung) beider Sprachen benötigt.

Abbildung 2.6 zeigt einen Zeitstrahl der bedeutsamsten Arbeiten in der Historie der maschinellen Übersetzungssysteme. Als die Geburtsstunde der MT gilt das „Translation memorandum“

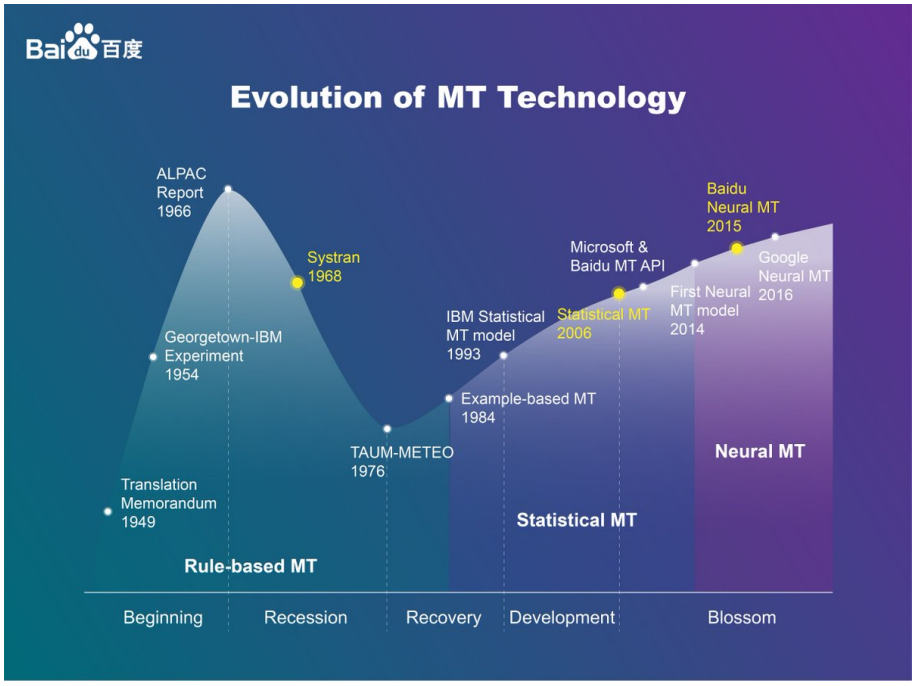


Abbildung 2.6.: Geschichtliche Entwicklung der maschinellen Übersetzung. Quelle: [42]

von Warren Weaver 1949, der in einem Brief an seine Arbeitskollegen der Rockefeller Foundation Möglichkeiten beschrieb, wie die neu erfundenen digitalen Computer zur Übersetzung von Texten genutzt werden können[102]. Die erste öffentliche Präsentation eines Übersetzungssystems fand 1954 im sogenannten Georgetown-IBM Experiment statt. Die Fähigkeiten des Systems beschränkten sich auf die Übersetzung von 49 vorausgewählten russischen Sätzen ins Englische, jedoch stieß das Experiment auf großes öffentliches Interesse und führte weltweit zu einem Forschungsboom. Die in den Folgejahren entwickelten regelbasierten Systeme hatten zumeist eine sehr schlechte Qualität. Der 1966 für das Verteidigungsministerium der Vereinigten Staaten erstellte ALPAC-Bericht[78] kritisierte schließlich den trotz intensiven Bemühungen ausbleibenden Fortschritt und brachte mit einem Schlag die Forschung für fast 20 Jahre praktisch ganz zum Erliegen. Erst in den 80ern nahm die Forschung mit neuen, auf der Statistik einer großen Anzahl von Übersetzungsbeispielen basierenden Methoden[70], wieder Fahrt auf. In den 90ern kam es, aufgrund des Aufkommens von billigen und leistungsfähigeren Computern, zu einem starken Anstieg von MT-Technologie. Im Jahre 2006 bot Google schließlich erstmals sein statisches Übersetzungssystem an, allerdings wurde 2010 die Qualität von MT-Systemen von vielen Menschen noch als unbefriedigend bewertet. Die Einführung von Übersetzungsmodellen auf Grundlage von neuronalen Netzen 2014 führte dann zu einem rasanten Fortschritt[15, 93]. Nahezu alle heutigen kommerziellen MT-Systeme basieren auf Methoden der NMT. Sie erzielen damit teilweise eine Qualität, die mit der von professionellen Übersetzern gleichzusetzen ist.[66]

2.4.1. Neuronale maschinelle Übersetzung

Die neuronale maschinelle Übersetzung (NMT) ist aktuell der de-facto Standard unter den MT-Technologien. Die wesentliche Stärke der NMT gegenüber herkömmlichen, auf statistischen Modellen basierenden Übersetzern liegt darin, dass alle Komponenten des Modells gemeinsam (Ende-zu-Ende) trainiert werden können. Viele der Entwurfsentscheidungen der traditionell phrasenbasierten MT können dadurch umgangen werden.[105]

In der NMT wird die Wahrscheinlichkeit, dass eine Sequenz $t = (t_1, \ldots, t_m)$ die Übersetzung einer Sequenz $s = (s_1, \ldots, s_m)$ ist, ausgedrückt durch die bedingte Wahrscheinlichkeit

$$\mathbb{P}(t|s) = \mathbb{P}(t_1, \ldots, t_m|s) = \prod_{i=1}^{m} \mathbb{P}(t_i|t_1, \ldots, t_{i-1}, s) \tag{2.12}$$

Zur Generierung der Übersetzung wird sequentiell die Wahrscheinlichkeit $\mathbb{P}(t_i|t_1, \ldots, t_{i-1}, s)$ maximiert. Der Grundbaustein der meisten NMT-Modelle ist das Sequence-to-Sequence (Seq2Seq) Framework nach Sutskever et al.[93], bestehend aus zwei hintereinandergeschalteten Recurrent Neural Network (RNN)s. Mit dem *Encoder*-Netz am Eingang wird der Eingabetext eingelesen und in einen Vektor fester Länge kodiert, das zweite RNN, das *Dekoder*-Netz, generiert aus diesem Vektor dann den Übersetzungstext. Für die Implementierungen in dieser Arbeit werden für die RNNs LSTM-Units nach Hochreiter et al.[40] verwenden. Um lange Eingabetexte besser verarbeiten zu können, kann das Modell um einen Attention-Mechanismus[2] ergänzt werden. Die finale Ausgabe wird durch eine Softmax-Schicht mit der Länge des Wörterbuchs der Zielsprache dargestellt. Die Ausgabeneuronen repräsentieren gemäß Gleichung 2.12 entsprechend die Wahrscheinlichkeiten für das nächste Wort in der Übersetzungssequenz.

2.4.2. Wortrepräsentation

In der NMT werden Ein- und Ausgabesequenzen als Zeitreihe modelliert. Wie schon bei der TK (siehe Unterabschnitt 2.3.2) werden entweder One-Hot-Kodierungen oder Wortvektoren zur mathematischen Repräsentation von Texten genutzt. Da sich das Vokabular von Quell- und Zielsprache unterscheidet, werden jeweils zwei separate Wörterbücher benötigt.

Das Kodieren von Sequenzen auf Wortebene bringt einige Nachteile mit sich. Die oftmals sehr großen Wörterbücher beeinflußen die Dimensionen von Ein- und Ausgabeschichten und steigern damit die Komplexität des Lernproblems. Weiter können Out-of-Vocabulary (OOV)-Wörter nicht übersetzt werden und müssen in der Ausgabe durch ein spezielles Symbol (*UNK-Token*) ersetzt werden. Als Alternative zu der Kodierung auf Wortebene kommt eine Kodierung von Texten auf Zeichenebene in Betracht, wodurch die Länge der Ein- und Ausgabeschicht auf die Anzahl der Buchstaben im Alphabet begrenzt werden. Bei einer Kodierung auf Zeichenebene kann die Übersetzung von unbekannten Wörter zudem aus kleineren Fragmenten abgeleitet werden. Ein Nachteil sind jedoch die dadurch entstehende langen Zeichensequenzen, welche beim Ausfalten der RNNs zu sehr tiefen Netzen führen, deren Training aufgrund von Langzeitabhängigkeiten erschwert ist. Ein Trade-off zwischen Wort- und Zeichenkodierung sind die seit 2015 für die NMT verwendeten Byte-Pair-Encodings (BPE, [86]), bei denen Worte aus kleinen Wortteilen zusammengesetzt werden, wodurch ebenfalls die Größe des Vokabulars und die Zahl der OOV-Worte verringert werden kann.

2.4.3. Metriken

Zur Evaluation von Übersetzungssystemen wird eine Methode zur Bewertung der Qualität der erstellten Übersetzungen benötigt. Ein professioneller Übersetzer beurteilt Übersetzungen an der Akkuratheit und am Lesefluss der Übersetzungstexte. Eine manuelle Evaluation von MT-Systemen ist jedoch zu zeit- und kostenintensiv, da für eine aussagekräftige Beurteilung sehr viele Beispiele ausgewertet werden müssen.

Bilingual Evaluation Understudy (BLEU) ist ein Algorithmus zur automatischen Evaluation von maschinellen Übersetzern. Die BLEU-Metrik weist eine hohe Korrelation zu den Ergebnissen einer von Menschen durchgeführten Evaluation auf und ist zudem effizient zu berechnen (polynomielle Komplexität), weshalb sie aktuell zu den Standardmetriken zur Bewertung von ML-Systemen gehört.[74]. Das grundlegende Prinzip des Verfahrens ist es, die Ausgabe von maschinellen Übersetzern mit den Übersetzungen eines professionellen, menschlichen Übersetzers zu vergleichen[23]. Der Wertebereich des Scores liegt im Intervall zwischen 0 und 1, wobei die Übersetzungsqualität umso besser ist, je näher der Wert an 1 liegt. Gegeben einen Übersetzungskandidat und eine manuell erstellte Referenzübersetzung, berechnet sich der BLEU-Score aus der Genauigkeit der sich überlappende n-Gramme. Die Genauigkeit wird für n-Gramme der Länge 1 bis 4 berechnet und deren geometrisches Mittel bestimmt. Es gilt:

$$\begin{aligned} Precision_i &= BP * \frac{\text{\#n-gram overlap}}{\text{\#n-grams}} \\ BLEU &= 100 * \sqrt[4]{\prod_{i=1}^{4} Precision_i} \end{aligned} \tag{2.13}$$

Der Faktor BP (engl. Brevity Penalty) wird eingefügt, um kurze Übersetzungen zu bestrafen. Es gilt zu beachten, dass BLEU zur Beurteilung der Performance auf Textkorpora entwickelt wurde. Für einzelne Sätze hat der Wert nur eine geringe Aussagekraft.

3. Verwandte Arbeiten

In diesem Kapitel werden die zu dieser Arbeit thematisch verwandten Publikationen vorgestellt. Abschnitt 3.1 beschreibt bekannte Modelle zur neuronalen Textklassifikation. Die präsentierten Modelle werden dann für die in dieser Arbeit angewandten Verfahren aus Kapitel 4 adaptiert. Abschnitt 3.2 beschäftigt sich mit den Veröffentlichungen zum Thema Lerntransfer. Der Schwerpunkt liegt dabei auf den neuronalen Verfahren der letzten Jahre. Danach geht Abschnitt 3.3 gesondert auf die Crosslinguale Textklassifikation (CTK) ein, die im weiteren Teil dieser Arbeit als Sonderfall des Lerntransfers aufgefasst werden soll. In Abschnitt 3.4 werden abschließend Arbeiten zur *Paraphrasierung* vorgestellt, welche parallelen zu dem TDE-Verfahren aus Unterabschnitt 4.3.1 aufweisen.

3.1. Neuronale Textklassifikation

Traditionelle Arbeiten zum Thema TK fokussieren sich auf das *Feature-Engineering* und die Optimierung der Lernalgorithmen. Das State-of-the-Art Verfahren waren lange Zeit Support Vektor Maschinen (SVMs), welche mit Merkmalen wie Bag-of-Words (BoW) oder N-Grammen trainiert werden. Begünstigt durch die bedeutenden Erfolge von Deep Learning in der Bild- und Videoverarbeitung[51], der akustischen Spracherkennung[31] und der maschinellen Übersetzung[93, 14], erfuhren NNs in jüngster Zeit auch auf dem Gebiet der TK einen großen Zuspruch aus der wissenschaftlichen Community[48, 111, 46, 96, 107, 72]. Methoden des *Deep Learnings* unterscheiden sich dabei grundlegend von dem traditionellen Vorgehen, da von NNs abstrakte Repräsentationen selbständig aus den Trainingsdaten erlernen, wodurch der aufwändige und kostenintensive Prozess des *Feature-Engineerings* vereinfacht wird.

In diesem Abschnitt werden die in der Literatur anzutreffenden Architekturmodelle für die neuronale Textklassifikation vorgestellt.

3.1.1. Convolutional Neural Networks

Die Convolutional Neural Networks (CNNs) sind eine Weiterentwicklung der Time Delay Neural Networks (TDNNs) für die Phonem Erkennung[99]. Time Delay Neural Network (TDNN)s besitzen die Fähigkeit, aus akustischen Signalen selbständig temporale, translationsinvariante Merkmale zu erlernen. Die Adaption des Konzepts auf die Bildverarbeitung führte schließlich zu den Convolutional Neural Network (CNN)s, eine vom visuellen Kortex inspirierte neuronale Netzarchitektur. Mit dem Aufkommen der Grafikprozessor-Programmierung (GPGPU-Computing) in den 2000er Jahren wurde es möglich, sehr große CNNs effizient zu trainieren. Aufgrund der hohen räumlichen Korrelation in natürlichen Bildern erzielen CNNs in der Computer Vision herausragende Ergebnisse[16, 51, 109]. In der Sprachverarbeitung konnten sie sich in den letzten Jahren für die Lösung verschiedener Aufgaben ebenfalls etablieren. CNNs

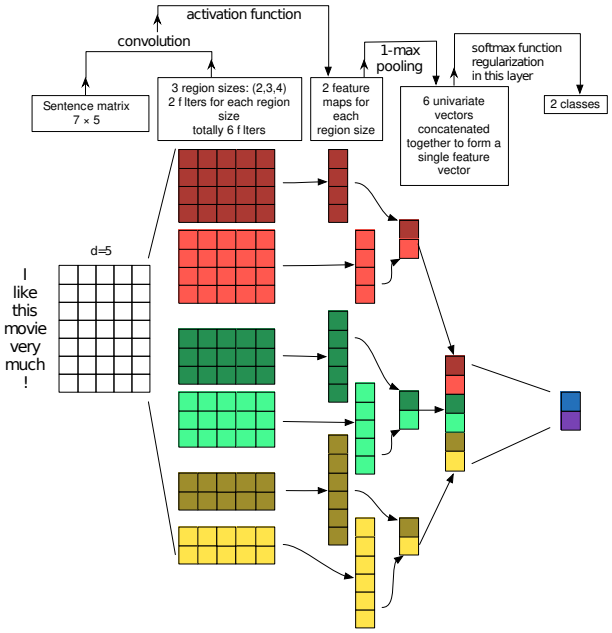


Abbildung 3.1.: CNNMC-Modell nach [48]. Quelle: [112].

wurden bereits erfolgreich für das semantische Parsen[32], Part-of-speech Tagging[18], die maschinelle Übersetzung[36] sowie für die hier besprochenen Textklassifikation[48, 47, 111] eingesetzt.

Die ersten Publikationen mit CNNs zur TK wurden 2014 von Kim et al.[48] und Kalchbrenner et al. [47] veröffentlicht. Die Netzarchitektur von Kim et al. wurde maßgeblich von der Architektur von Collobert et al.[18] beinflusst, die diese schon zuvor erfolgreich für andere NLP-Tasks wie *Part-of-Speech Tagging* und *Named Entity Recognition* einsetzten. Das Modell baut auf dem Modell unüberwacht trainierter Wortvektoren auf. Die Autoren verwenden dafür die von Mikolov et al. in [67] auf einem Google News Datensatz trainierten Word2Vec-Vektoren.[1] Die Netzstruktur ist flach und besteht aus einer einzigen Convolution-Schicht mit mehreren Filtern unterschiedlicher Größe. Im Anschluss werden von einer Max-Pooling-Schicht die wichtigsten Merkmale aus der Faltung ausgewählt, welche dann zur Prädiktion einer FC-Schicht mit Softmax-Ausgabe zugeführt werden. Der Zweck der Max-Pooling-Schicht ist es, das Netz unabhängig von der Länge des Eingabetexts werden zu lassen. Zudem soll das Netz dadurch gezwungen werden, die diskriminativen Phrasen des Texts zu erlernen. Trotz geringem Hyperparameter-Tuning konnten die Autoren bei ihrer Evaluation mit mehreren Benchmarks die Leistung bisheriger traditioneller ML-Methoden übertreffen. Ein Hinweis darauf, dass die vortrainierten Wordembeddings als universelle Feature-Extraktoren für personalisierte Klassifikationstasks aufgefasst werden kann. Eine Fortführung der Arbeit von Kim et al. stellt die Sensitivitätsanalyse von Zhang et al. in [112] dar. Sie analysieren das Netz bezüglich der bei der Entwicklung festzulegenden Hyperparameter und geben vor, von welchen Gegebenheiten deren Wahl abhängig gemacht werden sollte.

Das Modell von Kalchbrenner et al.[47] stellt eine Erweiterung der Architektur von Kim et al. dar. Sie beschreiben in ihrer Arbeit ein NN bestehend aus fünf hintereinander gereihten

[1] öffentlich verfügbar unter `https://code.google.com/p/word2vec/`.

Convolution-Schichten. Statt einem gewöhnlichen Max-Pooling wird nach jeder Faltung ein als *temporal K-Max-Pooling* Schritt durchgeführt. Dabei werden die k-größten Aktivierungen für die nachfolgende Schicht ausgewählt. Der Wert von k ist dabei variabel und hängt von der Länge des zu klassifizierenden Textes und von der Netzposition ab.

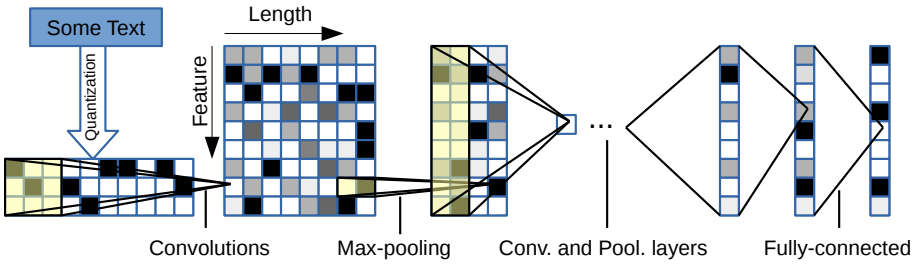


Abbildung 3.2.: Architektur eines CNNs auf Zeichenebene. Quelle: [111]

In [111, 110] stellen Zhang et al. eine weitere CNN-Architektur vor (siehe Abbildung 3.2). Das Modell besteht aus insgesamt sechs Convolution-Max-Pooling-Schichten gefolgt von drei FC-Schichten. Die Besonderheit ihres Ansatzes ist es, dass die Architektur nicht auf vortrainierte Wortrepräsentationen angewiesen ist. Stattdessen wird als Eingabesignale eine One-Hot-Kodierung der Zeichensequenz übergeben. Für die Evaluation erzeugten die Autoren aus Online Artikeln und Bewertungen mehrere große Datensätze mit Millionen von Trainingssampels. Außerdem experimentieren sie für eine zusätzliche Verbesserung der Leistung mit Datenerweiterungstechniken, bei denen sie Wörter zufällig durch deren Pendant aus einem englischsprachigen Thesaurus ersetzen. Im Vergleich zu anderen Verfahren zeigt sich, dass das zeichenbasierte Modell bei kleinen Trainingsdatensätzen (weniger als 1.000 Beispiele) keine ernsthafte Alternative zu traditionellen Methoden darstellt. Wegen des hochdimensionalen Hypothesenraums werden sie erst ab mehreren hunderttausend von Trainingsbeispielen zur leistungsfähigen Option.

Mit tiefen CNN-Architekturen für die TK beschäftigt sich die Publikation [19] von Conneau et al. Inspiriert von den Publikationenen aus der Bildverarbeitung, wo jüngst bei manchen Tasks eine klare Leistungssteigerung mit sehr tiefen Netzen erzielt werden konnte[88, 34], präsentieren die Autoren eine 29 Schichten tiefe *VD-CNN*-Architektur für die Textklassifikation. Das Netz arbeitet auf Zeichenebene und soll durch viele kleine lokale Operationen (Convolution und Max-Pooling) eine hierarchische Abstraktion von Sprache erlernen (Zeichen, Wörter, n-Gramme, Phrasen, Sätze). Anhand von acht Datensätzen mit 120.000 bis 3.600.000 Trainingsbeispielen gelingt es den Autoren, empirisch nachzuweisen, dass auch die TK von tiefen Architekturen profitiert.

3.1.2. Reccurent Neural Networks

Die Reccurent Neural Networks (RNNs) sind eine Netztopologie die für die Modellierung von Zeitreihen ausgelegt ist. Während CNNs die informativen n-Gramme aus einem Text extrahieren, berechnen RNNs eine gewichtete Kombination aller im Text vorhandenen Worte. Sie werden darauf trainiert, beliebig lange Wortsequenzen in einem Vektor fester Länge zu kodieren. Die so berechnete Textrepräsentation kann dann von nachfolgenden FC-Schichten als Merkmal für die Klassifikation genutzt werden. Der Artikel von Nowak et al.[72] evaluiert

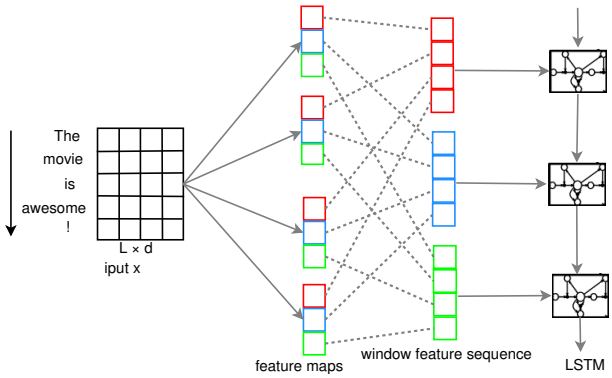


Abbildung 3.3.: Kombination von CNN und LSTM Architektur (C-LSTM). Quelle: [113]

die Performance einfacher Long Short-Term Memorys für die TK. Yin et al. kommen bei einem Vergleich von RNNs und CNNs zum Ergebniss, dass RNNs bei vielen NLP-Tasks gute und robuste Leistungen erzielen[108]. CNNs sind dann besser, wenn die Aufgabe die Erkennung einzelner Schlüsselphrasen erfordert. Aufgrund der Max-Pooling-Schicht fokussieren sich CNNs beim Training auf die lokalen, diskriminativen Merkmale in den Eingabedaten.

Ein weiterer Nachteil von RNNs ist, dass sie nicht nach dem Single „Instruction, Multiple Threads"-Paradigma parallel auf Grafikprozessoren trainiert werden können. Das Training benötigt mehr Zeit als das Training von CNNs. Die beiden Publikationen [53, 113] wollen die jeweiligen Vorteile von CNN und RNN miteinander kombinieren und verknüpfen in ihrer Architektur die beiden Netztypen miteinander. In Abbildung 3.3 ist der Aufbau der *C-LSTM* Architektur aus [113] abgebildet.

Für beide der vorgestellten Netzarchitekturen müssen bei der Implementierung Performance-kritische Hyperparameter festgelegt werden. Für Convolution-Schichten müssen konstante Fenstergrößen festgelegt werden. Ein zu kleines Fenster hat zur Folge, dass das Netz den zeitlichen Kontext unvollständig erfasst. Ein zu großes Fenster vergrößert den Hypothesenraum und führt bei zu wenigen Trainingsdaten zur Überanpassung. Bei rekurrenten Netzen muss modelliert werden, mit welchem Faktor Langzeitabhängigkeiten vergessen werden. Werden keine Informationen über die Zeit verworfen, treten beim Training mit langen Sequenzen Probleme mit verschwindenden Gradienten (engl. Vanishing Gradient Problem, siehe Kapitel 2) auf. Weitere essentielle Hyperparameter sind bei allen tiefen neuronalen Netze die Anzahl und Größe der Hidden-Layer und die Batch-Size[108].

Tang et al.[96] beschäftigen sich explizit mit der Klassifikation von langen Texten (Dokumente). Dafür wird zunächst in einer ersten Convolution- oder LSTM-Layer ein Satzvektor trainiert, der dann in der zweiten Hidden Layer von einem bidirektionalen GRU zu einem Dokumentenvektor zusammengefügt wird. Yang et al.[107] übertragen den Attention-Mechanismus aus der Maschinellen Übersetzung auf ein TK-System. Auch sie integrieren das Wissen über den hierarchischen Aufbau von Dokumenten in die Netzarchitektur. In der ersten Schicht werden mit dem Attention-Mechanismus die für die Domäne wichtigen Worte gewichtet. Die zweite Schicht fokussiert sich dann auf die als wichtig erkannten Sätze. Der Attention-Mechanismus eröffnet erstmals auch eine Möglichkeit, das im Netz gelernte Wissen zu visualisieren. Ein Nebeneffekt, dessen Bedeutung nicht zu vernachlässigen ist, da die so gewährten Einblicke genutzt werden

können, um der Ursache von Fehlklassifikationen nachzugehen und beispielsweise Biases in den Trainingsdaten zu erkennen.

3.1.3. Deep Averaging Network

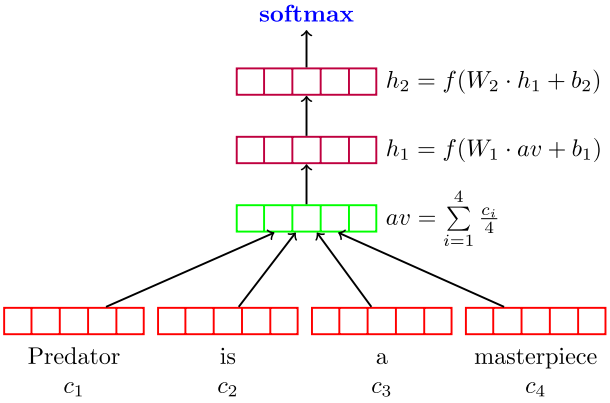


Abbildung 3.4.: Deep Averaging Network (DAN). Quelle: [43]

Durchaus überraschende Ergebnisse erzielen Iyyer et al. mit ihrem in [43] vorgestellten Deep Averaging Network (DAN). Das DAN berechnet in einem ersten Schritt den Mittelwert der als Eingabe übergebenen Wordembeddings. Anschließend wird mit mehreren übereinander gestackten FC-Schichten die Klassifikation durchgeführt (siehe Abbildung 3.4). Trotz seiner Einfachheit erreicht das DAN eine Genauigkeit die dem momentanen Stand der Technik entspricht. Aufgrund seiner Effizienz und den kurzen Trainingszeiten kann DAN in vielen Anwendungsszenarien als echte Alternative zu CNN- oder RNN-Architekturen aufgefasst werden. Die Autoren zeigen zudem, dass DAN bei syntaktisch und semantisch anspruchsvollen Texten die gleichen Fehler macht wie Modelle, bei denen dieses Wissen explizit in der Architektur modelliert wurde. Die Autoren kommen deshalb zur Schlussfolgerung, dass die Erforschung der TK noch nicht an ihrem Ende angekommen ist und die bisherigen Modelle noch nicht das volles Potential von NNs ausschöpfen können.

3.2. Lerntransfer

Methoden zum Lerntransfer blicken in der Wissenschaft auf eine lange Historie zurück. Das Forschungsgebiet fristete aber lange Zeit nur ein Nischendasein. Der 1995 abgehaltene NIPS-Workshop mit dem Titel „Learning to Learn“[2] formulierte erstmals den Wunsch, langlebige maschinelle Lernverfahren zu entwickeln, bei denen das Wissen aus zuvor gelernten Tasks und Domänen wiederverwendet wird[73]. Seither zieht das Thema unter verschiedenen Fachwörtern mehr und mehr Aufmerksamkeit aus der wissenschaftlichen Community auf sich. Neben dem hier verwenden Begriff des klassischen „Transfer Learnings“ beschäftigt sich das „Multitask Learning“ mit dem gleichzeitigen Trainieren unterschiedlicher Lernaufgaben, die dadurch von einer gemeinsamen Wissensbasis profitieren. Das „Incremental Learning“ ist eine

[2] http://socrates.acadiau.ca/courses/comp/dsilver/NIPS95_LTL/transfer.workshop.1995.html

Technik, bei dem Modelle mit überwachten und unüberwachten Lerntechniken kontinuierlich erweitert werden. Das „Meta Learning“ will flexiblere ML-Systeme entwickeln, die selbständig aus den Metadaten eines Lernproblems (z.B. Eigenschaften oder Performancemaße des Lernalgorithmus) lernen, sich auf andere Domänen zu adaptieren.

Die 2010 durchgeführte Studie[73] von Pan von Yang gibt einen detaillierten Überblick über Lerntransferverfahren aus der Ära vor dem Deep Learning. In diesem Abschnitt und im weiteren Verlauf der Arbeit wird näher auf Transfermethoden für neuronale Netze eingegangen. Neuronale Netze haben aufgrund ihres hierarchischen Aufbaus und der Fähigkeit, selbständig Repräsentationen der Eingabedaten zu erlernen, vielversprechende neue Möglichkeiten für den Wissenstransfer eröffnet. Das von einem Netz gelernte Wissen kann auf andere Lernaufgaben übertragen werden, indem gelernten Schichten und Komponenten als Merkmalsextraktor wiederverwendet werden.

3.2.1. Lerntransfer in der Bildverarbeitung

In der Bildverarbeitung konnten sich neuronale Netze zum State-of-the-Art Lernverfahren entwickeln. Ein bahnbrechender Erfolg waren die Ergebnisse von Krizhevsky et al.[51] auf dem ImageNet-Klassifikationstask[22]. Der ImageNet-Datensatz besteht aus 15 Millionen hochauflösenden Bildern, die manuell mithilfe von Amazon Crows-Sourcing Tool *Turk* manuell gelabelt wurden. Dank der vielen Trainingsdaten können auch tiefe Netzarchitekturen mit vielen Neuronen robuste Merkmale ausbilden. Zahlreiche Arbeiten haben sich seither mit dem Lerntransfer von ImageNet auf andere Computer Vision Tasks beschäftigt. Die Standardmethode für den Transfer bei NNs ist es, Gewichte nicht zufällig zu initialisieren, sondern *vortrainierte* Gewichte aus anderen Aufgaben zu übernehmen. Eine Leistungssteigerung durch vortrainierte Gewichte konnte für die Lernaufgaben Detektion[28], Tracking[79], Image Captioning[58, 89] und Visual Question Answering[106] dargelegt werden.

3.2.2. Lerntransfer in der Sprachverarbeitung

Das Labeling von textuellen Daten ist teuer und aufwendig. Textressourcen für unüberwachte Trainingsverfahren sind indes nahezu unbegrenzt verfügbar und können mit *Webcrawlern* automatisiert gesammelt werden. Ein Themenschwerpunkt der NLP der letzten Jahre war die Entwicklung von Methoden zur Übertragung des Wissens von ungelabelten Daten auf überwachte Trainingstasks. Zu diesem Zweck erfreuen sich *verteilte Repräsentatione*[39] großer Beliebtheit in der Wissenschaft[6, 17, 69, 98, 67, 76]. Die beiden etabliertesten Algorithmen zur Berechnung von Wortvektoren sind *Word2Vec*[67] und *GloVe*[76]. Sie übertreffen die Leistung herkömmlicher Modelle wie Bag-of-Words oder N-Gramme um ein Vielfaches und haben sich insbesondere bei neuronalen NLP-Verfahren als Standardmodell durchgesetzt. Durch das unüberwachte Training kann ein großes Vokabular für die Embeddings aufgebaut werden. Da semantisch ähnliche Wörter eine geringe Distanz zueinander haben, kann das TK-Verfahren besser auf die im gelabelten Datensatz nicht vorhandenen Wörter (engl. Out-of-Vocabulary-Words, kurz: OOV) generalisieren. Die Zahl der OOV-Wörter wird durch diesen Ansatz also deutlich gesenkt, da nur noch die Wörter für die keine Embeddings vorliegen als solche angesehen werden müssen.

Arbeiten, die sich über Word Embeddings hinausgehend mit dem Lerntransfer zwischen verschiedenen NLP-Tasks beschäftigen, sind rar. Le et al. stellen in [55] Paragraph-Vektoren für ganze Sätze und Dokumente vor. In [38] wird von Hill et al. evaluiert, wie gut sich das Wissen verschiedener NLP-Aufgaben auf den Semantic Textual Similarity-Task übertragen lässt. Aufgrund der vielen verfügbaren Daten für das Training maschineller Übersetzer wird der MT-Task als geeignete Quelle für den Transfer ausgemacht. Die selben Autoren kommen in [37] zum Ergebnis, dass die von MT-Systemen gelernten semantische Repräsentationen für manche Anwendungen den mit monolingualen Daten trainierten Sprachmodellen überlegen sind.

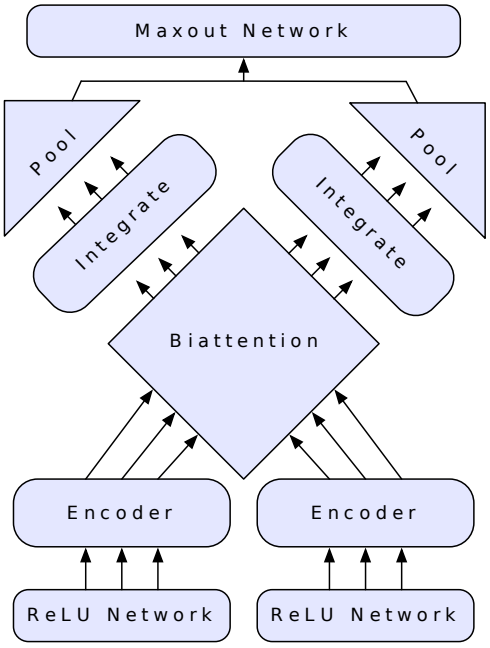


Abbildung 3.5.: Biattentive Classification Network (BCN). Quelle: [64]

Die Ende 2017 veröffentlichte Publikation von McCann et al.[64] ist die aktuellste Arbeit, die sich ausführlich mit dem MT-Tasks als Wissensquelle für andere Aufgaben der NLP beschäftigt. Zu den evaluierten Zieltasks gehören Sentimentanalyse, Question Classification und Entailment. Für alle drei Aufgaben kann durch den Lerntransfer eine gesteigerte Performance nachgewiesen werden. Für das Training des MT-Systems wird mit *OpenNMT*[50] ein Sequence-to-Sequence Modell mit Attentionmechanismus trainiert. Die erzeugte Vektorsequenz des Encoders bezeichnen die Autoren als Kontextvektor $CoVe$. Sie wird zusammen mit den GloVe-Vektoren als Sprachmodell für die Zieltasks verwendet. Für die TK verwenden die Autoren das *Biattentive Classification Network* (BCN) (siehe Abbildung 3.5). Es wurde für die Klassifikation von Texten aus ein bis zwei Sätzen entworfen. Die Eingabesequenz $\hat{w} = [GloVe; CoVe]$ wird einer mehrschichtigen Architektur aus FC-Schichten und bidirektionalen LSTMS zugeführt und abschließend von einem dreischichtigen Maxout-Netzwerk[29] klassifiziert.

3.3. Crosslinguale Textklassifikation

Der überwiegende Teil der Publikationen zum Thema TK beschäftigt sich mit der Klassifikation von englischsprachigen Texten. Die Arbeit von Dashtipour et al. aus 2016 gibt einen Überblick über die Performance von State-of-the-Art TK-Verfahren für nicht-englischsprachige Texte. Die Crosslingualen Textklassifikation (CTK) macht es sich zur Aufgabe, mithilfe der Trainingsdaten einer ressourcenreichen Sprache Klassifikatoren für eine Zielsprachen zu trainieren, in der es an Ressourcen mangelt. Das Thema wurde in der Literatur vorwiegend gegen Ende des letzten Jahrzehnts abgehandelt. Zu den wichtigen Beiträgen gehören die Arbeiten von Banea et al.[3], Wei et al.[103] und Wan et al.[100]. Die zum damaligen Zeitpunkt verbreiteten statistischen maschinellen Übersetzer waren mit ihrer Leistung weit von den heutigen System entfernt. Bei einer erneuten Evaluation der Verfahren mit modernen NMT-Systemen kann voraussichtlich eine Leistungssteigerung zu damals festgestelt werden. In [3] werden drei Ansätze vorgestellt, die ähnlich zu den beschriebenen Ansätzen aus der vorliegenden Arbeit sind (siehe Abschnitt 4.2). Die Evaluation wird mit SVMs und NB-Klassifikatoren durchgeführt. Dabei wird untersucht, wie Rumänische und Spanische Klassifikatoren aus englischen Trainingsdaten aufgebaut werden können. Die Arbeit von [103] versucht mit speziellen Trainingstechniken nur die „relevanten“ Teile der Übersetzung für das Training zu verwenden. Mit dem Verfahren soll der Rauschanteil durch die Übersetzung verringert werden. In [100] wird die CTK erstmalig als Domänenadaptionsproblem, eine Sonderform des Lerntransfers, aufgefasst. Die Autoren präsentieren ein Co-Learning-Verfahren, das aus gelabelten englischen und ungelabelten chinesischen Daten einen Klassifikator für chinesische Texte aufbaut.

In ihrem Leitartikel aus 2011 beschäftigten sich Duh et al.[26] mit der Fragestellung, ob die damaligen State-of-the-Art MT-Systeme „reif“ für die CTK seien. Sie machen die Beobachtung, dass selbst mit einem perfekten Übersetzungssystem ein Leistungsabfall in der Zielsprache auftreten kann, da die Übersetzer einen systematischen Bias haben, da sie im Regelfall nicht mit auf den Klassifikationstask zugeschnittenen Texten trainiert werden. Sie schlussfolgern, dass moderne MT-Systeme zwar für die CTK geeignet sind, aber dass das damit einhergehende Domänenadaptionsproblem noch nicht ausreichend verstanden wurde und weitere Untersuchungen notwendig sind.

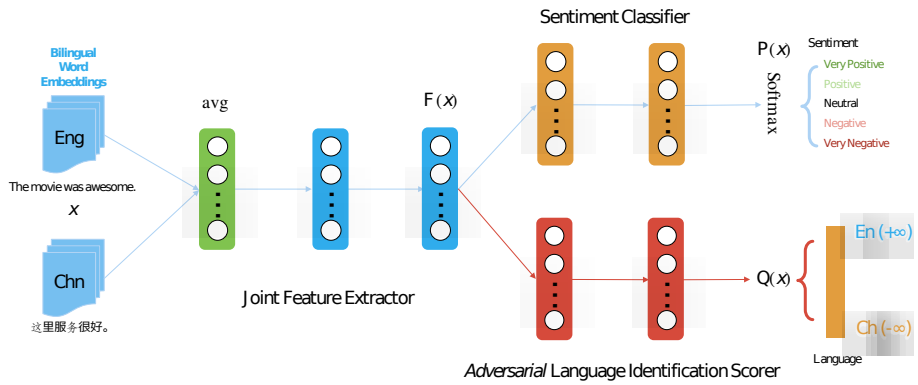


Abbildung 3.6.: Adverserial Deep Averaging Network (ADAN). Quelle: [13]

Ein modernes Verfahren zur CTK wird in [13] von Chen et al. eingeführt. Das präsentierte Adverserial Deep Averaging Network (ADAN) wird mit gelabelten englischen Daten und unge-

labelten chinesischen Daten für Sentimentanalyse in beiden Sprachen trainiert. Das Netz besitzt eine Y-Struktur und besteht aus zwei diskriminativen Zweigen (siehe Abbildung 3.6): Einem *Sentiment Classifier* und einem *Adversarial Language Identification Scorer*. Das grundlegende Prinzip beim Training ist es, das Erlernen einer sprachunabhängigen Zwischenrepräsentation $F(x)$ zu forcieren, indem der Language Scorer das Netz dafür „bestraft", wenn er die Sprache des Eingabetextes mit hoher Sicherheit prädizieren kann. Das Verfahren ist somit ein CTK-Verfahren, das ohne den Einsatz von Übersetzungstechnologien auskommt.

3.4. Paraphrasierung

Die *Paraphrasierung* (engl. Paraphrasing) ist ein noch junges Aufgabenfeld der NLP, das die Entwicklung von Systemen zur automatischen Generierung von Paraphrasen zum Ziel hat. Unter einer *Paraphrase* versteht man eine alternative Formulierung eines Ausdruckes in derselben Sprache und mit dem selben semantischen Inhalt wie dessen Ursprungsform[61]. Der Task ist eng verwandt mit dem *Semantic Textual Similarity Task* (STS-Task), bei dem versucht wird, ein quantitatives Maß für die semantische Ähnlichkeit von Phrasen und Texten zu finden. Der Anreiz für die Erforschung der Paraphrasierung kommt nicht zuletzt daher, dass Paraphrasen auf vielfältige Weise zur Lösung anderer Probleme der Sprachverarbeitung eingebracht werden können. In der Vergangenheit wurde die Paraphrasierung unter anderem schon für das Semantische Parsen[7] und die Verbesserung von maschinellen Übersetzern[12] eingesetzt. Im späteren Verlauf dieser Arbeit (siehe Abschnitt 4.3) wird ein Verfahren vorgestellt, bei dem mithilfe von Paraphrasen zusätzliche Trainingsbeispiele für die Textklassifikation erzeugt werden.

Paraphrasen können auf verschiedenen sprachlichen Ebenen auftreten. Es können einzelne Wörter, zusammenhängende Phrasen oder vollständige Sätze paraphrasiert werden (engl. *lexical*, *phrasal* und *sentential* paraphrases). Die in der Literatur anzutreffenden Lösungsansätze sind vielfältig und häufig auf einen speziellen Anwendungsfall zugeschnitten. Bei den regelbasierten Algorithmen werden Texte mit einfachen Termersetzungen umformuliert. Sie arbeiten zumeist auf lexikalischer Ebene und verwenden ausschließlich das Wissen aus Synonymwörterbüchern (sogenannte linguistische Thesauri) und formalen Grammatiken. Für die Paraphrasierung auf Satzebene haben sich in letzter Zeit, so wie auch in vielen anderen Bereichen der NLP, datengestützte ML-Verfahren durchgesetzt. Die Studie von N. Madnani und B. Dorr[61] aus 2010 gibt einen umfangreichen Überblick über korpusbasierte Paraphrasierungsverfahren.

Einer der erfolgreichsten Ansätze ist das *Bilingual Pivoting* nach Bannard et al.[4]. Dabei wird nach Textpaaren gesucht, die von einem phrasenbasierten statistischen Überersetzungssystem auf dieselbe Übersetzung abgebildet werden. Die beiden Texte sind somit bedeutungsgleich und können gegenseitig als Paraphrasen aufgefasst werden. Das Verfahren konnte in mehreren Anschlussarbeiten weiterentwickelt werden. In der Veröffentlichung von Mallinson et al. gelang es zuletzt, den Ansatz des *Bilingual Pivoting* auch mit Methoden der neuronalen maschinellen Übersetzung durchzuführen[62].

4. Neuronale Textklassifikation mittels Wissen aus der maschinellen Übersetzung

Viele der Problemstellungen in der NLP sind schwierig, da die natürliche Sprache komplex, mehrdeutig und ständigen Veränderungen unterworfen ist. Die Bedeutung von Wörtern und Sätzen hängt oftmals von deren Kontext und von der Domäne ab, aus der ein Text entnommen wurde. In Kapitel 1 wurde erläutert, wie insbesondere *personalisierte* TK-Tasks oft ein detailliertes Verständnis der sprachlichen Semantik voraussetzen. Gleichzeitig ist das manuelle Labeling von Texten nach semantischen Regeln ein teurer und aufwendiger Prozess, weshalb die Trainingsdatensätze häufig nur aus wenigen Beispielen bestehen. In diesem Kapitel werden mehrere Ansätze vorgestellt, mit denen der Datenknappheit bei TK-Tasks entgegengewirkt werden kann.

4.1. Gegenwärtige Situation

Die Leistungsfähigkeit heutiger TK-Systeme wird maßgeblich von der Anzahl der verfügbaren Trainingsbeispiele begrenzt. Liegen zu wenige Trainingsdaten vor, so kann der verwendete ML-Algorithmus kein ausreichend präzises Verständnis über die semantischen Eigenschaften der Domäne erlernen. Bei den in der Praxis überwiegend anzutreffenden Merkmalsrepräsentationen (Bag-of-Words und N-Gramme) werden Wörter als atomare Einheit aufgefasst. Klassische TK-Verfahren machen ihre Entscheidung über die Klassenzugehörigkeit einzig von den klassenbedingten Verteilungen der Wörter in den Trainingsdaten abhängig. Bei einer zu geringen Anzahl von Trainingsdaten unterliegen die Modelle dabei dem „Fluch der Dimensionalität“. Die Texte der Testphase unterscheiden sich von allen Wortsequenzen, mit denen der Klassifikator trainiert wurde. Je kleiner der Trainingsdatensatz, umso größer ist die Anzahl der Wörter, die nicht im Trainingsdatensatz vorkommen (engl. Out-of-Vocabulary Words, kurz: OOV-Wörter). Da dem Klassifikator kein Wissen über die Semantik von OOV-Wörtern und deren Wahrscheinlichkeitsverteilungen zur Verfügung steht, kann er sie folglich auch nicht in seine Entscheidung miteinbeziehen.

In der Vergangenheit wurden vornehmlich Techniken des *Feature-Engineerings* gegen das Problem der Datenknappheit entwickelt. Das Annotieren der Trainingsdaten mit zusätzlichen syntaktischen und semantischen Informationen muss jedoch von einem menschlichen Domänenexperten durchgeführt werden und verursacht weitere Kosten. Künstliche neuronale Netze ermöglichen es, den *Feature-Engineering* Prozess zu automatisieren, indem sie selbständig komplexe und nicht-lineare Zusammenhänge in den Merkmalen erlernen. Obwohl in den letzten Jahren zahlreiche Modelle zur neuronalen TK vorgestellt wurden (siehe Abschnitt 3.1) und trotz des breitflächigen Erfolgs von NNs in der Bildverarbeitung, insbesondere auch in der Bildklassifikation, konnten sie sich für die TK bislang jedoch noch nicht gegen die traditionellen

Verfahren durchsetzen. Aufgrund des vergleichsweise großen Hypothesenraums tendieren tiefe NNs beim Training mit wenigen Beispielen zur Überanpassung (engl. Overfitting). Andere Verfahren wie Support Vektor Maschinen, naive Bayes- oder Nächste-Nachbarn-Klassifikatoren sind leichter zu implementieren, erfordern weniger Parameter-Tuning und generalisieren bei wenigen Trainingsdaten vergleichsweise gut.

Für die Entwicklung von TK-Systemen der nächsten Generation werden Lösungen benötigt, um trotz der beschränkten Trainingsdaten besser auf ungesehene Wortsequenzen zu generalisieren. Zum einen sollen trainingsdatenintensive tiefe Neuronale Netze das teure und aufwendige *Feature-Engineering* obsolet machen. Zum anderen wird nach Methoden gesucht, mit denen semantisches Wissen über OOV-Wörter in den Klassifikator integriert werden kann.

Ein großer Fortschritt diesbezüglich sind die in den letzten Jahren entwickelten verteilten Wortrepräsentationen (siehe Abschnitt 3.2). Die aus ungelabelten Textressourcen gelernten Vektordarstellungen sind geeignete Merkmale, um Texte semantisch zu diskriminieren. Durch das unüberwachte Training kann ein großes Vokabular aufgebaut werden, wodruch sich auch die Zahl der OOV-Wörter bei der TK verringern lässt. Ein mit Word Embeddings noch nicht vollständig gelöstes Problem ist aber die Tatsache, dass sich Inhalt von Texten nicht immer aus der Summe der Wortsemantiken ergibt, sondern vielmehr aus der Semantik von ganzen Sätzen und Phrasen[71].

In diesem Kapitel wird untersuchen, wie das von maschinellen Übersetzern erlernte Wissen auf die Problemstellung der Textklassifikation übertragen werden kann. Für den Aufbau maschineller Übersetzer ist die Datenknappheit nicht so stark ausgeprägt wie bei anderen NLP-Aufgaben. Zwar gilt auch für MT-Systeme der Leitsatz „There's no data like more data", im Laufe der letzten Jahre konnten aber große zweisprachige Datensätze angesammelt werden. Die Conference on Machine Translation (Workshop on Machine Translation, kurz: WMT), einer jährlich abgehaltenen Konferenz zum Thema maschinelle Übersetzung, veröffentlicht beispielsweise jährlich große Korpora mit von professionellen Übersetzern angefertigten Übersetzungen.

Es werden zwei zwei grundsätzlich verschiedene Problemszenarien betrachtet:

In Abschnitt 4.2 erfolgt die Thematisierung der crosslingualen Textklassifikation. Es wird angenommen, dass für den zu erlernende Task keine Trainingsbeispiele vorhanden sind. In einer anderen Sprache liegen aber für eine äquivalent definierte Lernaufgabe ausreichend gelabelte Daten vor. Mithilfe des Wissens von maschinellen Übersetzern soll die Sprachbarriere überbrückt werden, um aus den Trainingsdaten der einen Sprache einen Klassifikator einer anderen Sprache zu trainieren.

Die Ausgangssituation in Abschnitt 4.3 ist eine andere. Hier sind gelabelte Trainingsdaten in der gewünschten Sprache verfügbar. Die Daten sind allerdings stark beschränkt und reichen nicht für das Training eines Klassifikators mit zufriedenstellender Akkuratheit aus. Es werden Verfahren untersucht, mit denen das Wissen von maschinellen Übersetzern zur Steigerung der Leistung von TK-Systemen genutzt werden kann.

Bei den beschriebenen Verfahren kommen verschiedene Methodiken zum Einsatz, mit denen das Wissen des MT-Systems übertragen wird. Die Verfahren in Abschnitt 4.2 und Unterabschnitt 4.3.1 fassen den maschinellen Übersetzer als Blackbox auf und generieren mithilfe des Übersetzers synthetische Trainings- oder Testdaten. Das erlernte Wissen des Übersetzers wird dabei auf die Daten des Klassifikator übertragen. Das Verfahren in Unterabschnitt 4.3.2

überträgt hingegen direkt das im Modell des MT-Systems kodierte Wissen auf das Modell des Klassifikators.

4.2. Crosslinguale Textklassifikation mit maschinellen Übersetzern

In diesem Abschnitt werden Verfahren zur crosslingualen Textklassifikation (CTK) besprochen. Die Problemstellung der CTK lässt sich wie folgt formalisieren:

Gegeben sei eine Quellsprache *SOURCE* und eine Zielsprache *TARGET*. Aufgabe sei es, einen Textklassifikator für Texte der Zielsprache *TARGET* zu trainieren. Für den Task sind jedoch keine gelabelten Trainingsbeispiele aus *TARGET* verfügbar. Es gilt $D_T = \emptyset$. Für einen semantisch äquivalenten Task in *SOURCE* sind jedoch ausreichend viele Daten vorhanden. Die gelabelten Daten aus *SOURCE* werden mit $D_S = \{(x_{S_1}, y_{S_1}), \cdots, (x_{S_{n_S}}, y_{S_{n_S}})\}$ bezeichnet.

Definition 2 *(Semantische Äquivalenz) Ein TK-Task $\mathcal{T}_S = (\mathcal{Y}, f_S(\cdot))$ einer Sprache SOURCE ist semantisch äquivalent zu einem TK-Task $\mathcal{T}_T = (\mathcal{Y}, f_T(\cdot))$ einer zweiten Sprache TARGET, wenn die Klassenzugehörigkeit in beiden Sprachen von denselben inhaltlichen Kriterien abhängt.*
Für die prädiktiven Zielfunktionen und Texte x_S in SOURCE und x_T in TARGET gilt:
x_S bedeutungsgleich zu $x_T \Rightarrow f_S(x_S) = f_T(x_T)$.

Die vorgestellten Ansätze zur CTK sehen vor, Texte unter Erhalt ihrer Semantik zwischen den Sprachen *SOURCE* und *TARGET* zu überführen. Hierfür drängt sich der Einsatz von MT-Systemen auf, da korrekte Übersetzungen invariant bezüglich der Semantik des Originaltexts sind (eine ausführliche Erläuterung des Zusammenhangs findet in Abschnitt 4.3 statt).

Es wird für einen beliebigen Text x_S aus *SOURCE* das MT-System zur Generierung einer Übersetzung x_T in *TARGET* definiert :

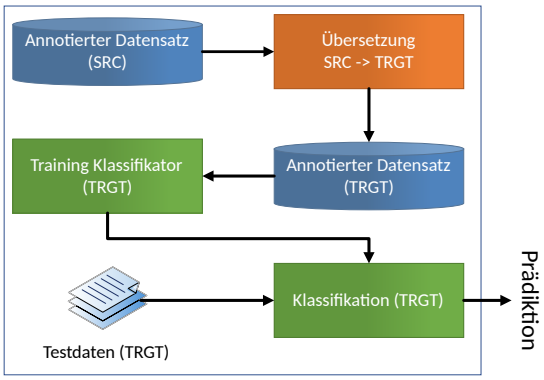
$$\mathrm{T}_{SRC \rightarrow TRGT}(x_S) \rightarrow x_T \tag{4.1}$$

Analog wird für einen beliebigen Text x_T aus *TARGET*

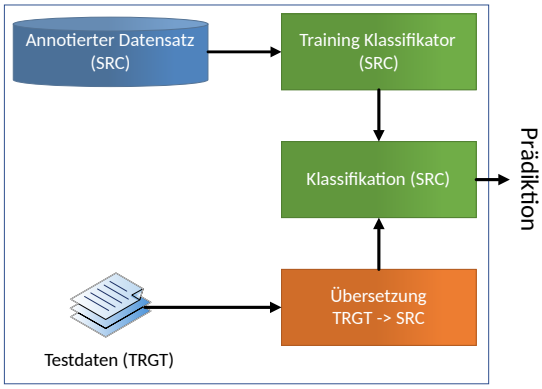
$$\mathrm{T}_{TRGT \rightarrow SRC}(x_T) \rightarrow x_S \tag{4.2}$$

für die umgekehrte Richtung zur Übersetzung von *TARGET* nach *SRC* definiert. Das konkrete Modell des Übersetzers ist dabei für die Betrachtung zunächst unerheblich. Generell kann gesagt werden, dass eine bessere Übersetzungsqualität auch zu einer leistungsstärkeren Textklassifikation in *TARGET* führt.

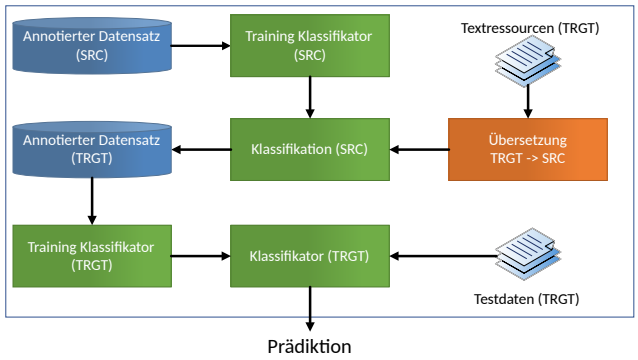
Es werden drei intuitive Ansätze zur Generierung von synthetische Daten für den Aufbau eines Klassifikators in *TARGET* (siehe Abbildung 4.1) präsentiert. Sie unterscheiden sich darin, ob Trainingsdaten oder Testdaten übersetzt werden. In Kapitel 5 soll dann die Leistung der Verfahren evaluiert werden.



(a) Methode 1: Syntethische Trainingsdaten



(b) Methode 2: Syntehthische Testdaten



(c) Methode 3: Synthethische Labels

Abbildung 4.1.: Methoden zur crosslingualen Textklassifikation mit MT-Systemen

4.2.1. Methode 1: Synthetische Trainingsdaten

Bei der ersten Methode (Abbildung 4.1a) werden in der Trainingsphase des Klassifikators die Trainingsbeispiele nach *TARGET* übersetzt. Aus dem Trainingsdatensatz

$$D_S = \{(x_{S_1}, y_{S_1}), \cdots, (x_{S_{n_S}}, y_{S_{n_S}})\}$$

wird mit $\mathrm{T}_{SRC \to TRGT}$ der synthetische Datensatz

$$D_T = \{(\mathrm{T}_{SRC \to TRGT}(x_{S_1}), y_{S_1}), \cdots, (\mathrm{T}_{SRC \to TARGET}(x_{S_{n_S}}), y_{S_{n_S}})\}$$

generiert. Mit den Daten wird dann mit herkömmlichen TK-Verfahren ein Klassifikator in *TARGET* trainiert.

4.2.2. Methode 2: Synthetische Testdaten

Bei der zweiten Methode (Abbildung 4.1b) wird mit D_S ein Modell für einen Klassifikator in *SOURCE* trainiert. Die Übersetzung findet dann erst später in der Testphase statt. Testdaten werden vor der Klassifikation mit $\mathrm{T}_{TRGT \to SRC}$ nach *SOURCE* übersetzt. Ein beliebiger Text x_T aus *TARGET* kann mit

$$y = f_S(\mathrm{T}_{TRGT \to SRC}(x_T))$$

klassifiziert werden. Das Verfahren zeichnet sich dadurch aus, dass es nicht auf die Trainingsdaten D_S angewiesen ist. Es können also auch bereits existierende Systeme (z.B. APIs zur Sentimentanalyse) für die Klassifikation verwendet werden. Das Übersetzen in der Testphase hat jedoch einen negativen Effekt auf die Gesamtlaufzeit des Systems. Ein Klassifikator muss einmalig trainiert werden und wird dann über seine restliche Lebenszeit für die Inferenz eingesetzt. Eine Anforderung an TK-Systeme ist oft, große Textmengen in einer möglichst kurzen Zeit zu klassifizieren. Das rechenaufwendige Übersetzen vor jeder Klassifikation steht gegen diese Anforderung.

4.2.3. Methode 3: Synthetische Labels

Das Ziel der dritten Methode (Abbildung 4.1c) ist es, anstelle von synthetischen Übersetzungen beim Training eines Klassifikators in *TARGET* natürlichsprachliche Texte zu verwenden. Vorausgesetzt wird das MT-System $\mathrm{T}_{TRGT \to SRC}$ sowie ungelabelte Daten $U_T = (u_1, \cdots, u_m)$ in *TARGET*. Das Beschaffen roher Textressourcen aus der Klassifikationsdomäne stellt bei Produktivsystemen kein Hindernis dar. Sie können aus der selben Quelle wie die Testdaten entnommen werden.

Zuerst wird mit den gelabelten Daten D_S ein Klassifikator für *SOURCE* trainiert. Danach werden die Rohdaten U_T aus *TARGET* mit $\mathrm{T}_{TRGT \to SRC}$ übersetzt und klassifiziert. Die prädizierten Klassenlabel werden auf die Ausgangstexte übertragen. Es entsteht ein neuer Datensatz

$$D_T = \{(u_1, f_S(\mathrm{T}_{TRGT \to SRC}(u_1)), \cdots, (u_m, f_S(\mathrm{T}_{TRGT \to SRC}(u_m))\}$$

der dann für das Training eines Klassifikators in *TARGET* verwendet wird.

Das Verfahren hat die folgenden Vorteile:

1. Die Übersetzung findet in der Trainingsphase statt. Die Laufzeit für die Klassifikation wird durch das Verfahren nicht beeinflusst.

2. Von MT-Systemen generierte Übersetzungen sind einem systematischem Bias unterworfen. Maschinelle Übersetzer werden zumeist mit allgemeinen Texten aus verschiedenen Quellen trainiert. Die Übersetzungen sind sprachlich nicht auf die Domäne des TK-Tasks zugeschnitten. Zudem ist der sprachliche Umfang (Grammatik und Vokabular) synthetisch generierter Texte gegenüber der natürlichen Sprache eingeschränkt. Der von dem hier angewandten Verfahren erzeugte Datensatz D_T besteht aus natürlichsprachigen Texten der Domäne und ist deshalb im Vergleich zu Unterabschnitt 4.2.1 in seiner Sprache besser auf den TK-Task in *TARGET* abgestimmt.

4.3. Textklassifikation bei begrenzten Trainingsdaten

In diesem Abschnitt soll mit den gelabelten Trainingsdaten

$$D = \{(x_1, y_1), \cdots, (x_n, y_n)\}$$

bestehend aus Texten x_i der Sprache *LANGUAGE*, ein TK-System in *LANGUAGE* trainiert werden. Die Trainingsdaten sind jedoch stark *beschränkt*, das heißt die Anzahl n der verfügbaren

Beispiele ist zu gering und ML-Verfahren können nicht ausreichend auf ungesehene Testdaten generalisieren.

Beschränkte Trainingsdaten sind bei TK-Tasks häufig anzutreffen. Das Labeling der Daten ist zeit- und kostenintensiv. Wird beispielsweise den Wortschatz der deutschen Sprache betrachtet, der auf rund 500.000 Wörter beziehungsweise Lexeme geschätzt wird[25], wird schnell ersichtlich, dass es in den meisten Fällen nicht möglich ist, alle für eine Lernaufgabe relevanten Wörter mit Trainingsbeispielen abzudecken.

Auf dem Gebiet des maschinellen Lernens gab es in den letzten Jahren die Tendenz, vorherrschende traditionellen Lernverfahren mit künstlich neuronalen Netzen zu ersetzen. Der Vorteil von NNs liegt darin, dass sie komplexe und nicht-lineare Zusammenhänge in den Merkmalen modellieren. Insbesondere profitieren neuronale TK-Systeme von der Fähigkeit von Long Short-Term Memory (LSTM)- oder CNN-Architekturen, komplexe zeitliche Zusammenhänge in Wortsequenzen zu erlernen. Das Problem ist jedoch, dass aufgrund der oben beschriebenen Datenknappheit dieser Vorteil oft nicht zum Tragen kommt. Für eine effektive Klassifikation mit neuronalen Netzen werden mehr Trainingsdaten benötigt.

Für die Lösung des Problems der beschränkten Trainingsdaten können die Erkenntnissen und Erfahrungen aus der Computer Vision zugrunde gelegt werden. In der Bildverarbeitung ist die Leistung von NNs anderen ML-Verfahren um Größenordnungen überlegen. Das Problem der Datenknappheit ist jedoch auch in der Bildverarbeitung bekannt und ausführlich erforscht. Um bei wenigen Trainingsdaten dennoch eine hohe Generalisierungsfähigkeit zu erzielen, kommen dort routinemäßig verschiedene leistungssteigernde Techniken zum Einsatz:

1. Trainingsdatenerweiterungsverfahren (TDE-Verfahren, engl. Data-Augmentation-Techniques) nutzen erwartete, invariante Eigenschaften des Modells, um automatisiert aus vorhandenen Trainingsdaten weitere Beispiele mit bekanntem Label abzuleiten und die vorliegenden Daten zu vervielfachen. In mehreren Arbeiten konnte bereits dargelegt werden, dass durch das zufällige Zuschneiden, Rotieren und Spiegeln von Bildern die Performance von neuronalen Architekturen verbessert werden kann[101].

2. Beim neuronalen Lerntransfer wird das interne Wissen eines Modells auf das Modell zur Lösung einer anderen, verwandten Aufgabenstellung übertragen. Ein ML-System zur Erkennung von Autos muss offensichtlich ähnliche visuelle Features wie ein System zur Erkennung von LKWs erlernen. Die hierarchische Struktur von tiefen neuronalen Netzen eignet sich hervorragend dafür, abstrakte Merkmale zwischen verschiedenen Tasks zu übertragen. Das Initialisieren von Netzschichten mit den anhand von großen Datensätzen (z.B. ImageNet[51]) erlernten Gewichten hat sich mittlerweile als Standardverfahren für viele Computer Vision Aufgaben etabliert.

In diesem Abschnitt werden Algorithmen vorgestellt, die angelehnt an die beiden beschriebenen Techniken (TDE-Verfahren und neuronaler Lerntransfer), die Leistung von Textklassifikatoren bei beschränkten Trainingsdaten verbessern. In Unterabschnitt 4.3.1 wird ein TDE-Verfahren beschrieben, dass mit den synthetischen Daten eines maschinellen Übersetzers zusätzliche Trainingsbeispiele generiert. In Unterabschnitt 4.3.2 wird erläutert, wie das Wissen eines NMT-Modells auf einen neuronalen Textklassifikator übertragen werden kann.

4.3.1. Trainingsdatenerweiterung mit MT-Systemen

In diesem Abschnitt wird ein Trainingsdatenerweiterungsverfahren (TDE-Verfahren) für gelabelte TK-Datensätze vorgestellt. Das Verfahren wird **vor dem Training** des Klassifikators durchgeführt und ist somit unabhängig vom eingesetzten ML-Algorithmus. Mithilfe von MT-Systemen sollen aus den vorhanden Daten zusätzliche synthetische Trainingsbeispiele mit bekanntem Label generiert werden.

Im folgenden wird ein TDE-Verfahren wie folgt definiert:

Definition 3 *(TDE-Verfahren) Ein TDE-Verfahren ist ein Verfahren zur Vergrößerung von (annotierten) Trainingsdatensätzen. Für die Generierung von Trainingsbeispielen werden vom Entwickler invariante Eigenschaften an das Modell festgelegt, unter deren Ausnutzung dann aus den vorhandenen Daten neue Beispiele mit bekanntem Label abgeleitet werden.*

Der Entwurf des erarbeitetenden Erweiterungsverfahrens für textuelle Daten wurde inspiriert durch die weiter oben bereits beschriebenen Techniken aus der Bildverarbeitung. Datensätze für die Bildklassifikation können durch einfache Transformationen auf ein beliebiges ihrer ursprünglichen Größe ausgedehnt werden. Zu den in der Praxis durchgeführten Transformationen gehören beispielsweise das Spiegeln, Zuschneiden, Verzerren und das Einfügen von Bildeffekten (z.B. Vinetten). Für verschiedene Trainingstasks konnte bereits dargelegt werden, dass durch die Erweiterung der Daten eine Überanpassung auf die Trainingsdaten gemindert werden kann[51, 41, 104, 35].

Zur grundsätzliche Funktionsweise von TDE-Verfahren kann folgende Beobachtung gemacht werden:

Beobachtung 1 *Ein TDE-Verfahren trägt dann zur besseren Generalisierung eines maschinellen Lernverfahrens bei, wenn die generierten Beispiele zusätzliches Wissen über die Entscheidungsgrenzen der Lernaufgabe enthalten.*

Bei der genaueren Betrachtung des Effekts von syntethischen Datenerweiterungen auf CNNs, stellen die Autoren [56, 75, 104] fest, dass die festgelegten invarianten Eigenschaften auch in den gelernten Merkmalen wiederzufinden sind. Bei der Bildklassifikation gehört ein gespiegeltes Bild zur selben Klasse wie dessen Ursprungsbild. Es lässt sich beobachten, dass beim Training mit erweiterten Daten das Netz diese Äquivalenz bereits nach wenigen Trainingsiterationen erkennt. In den tiefer gelegenen Schichten werden die synthetischen und originären Bilder dann mit den selben Repräsentationen dargestellt. Die Komplexität des Lernproblems sinkt, wodurch die Klassifikation akkurater wird.

Soll für die TK ein vergleichbares Verfahren entworfen werden, muss zunächst die Frage beantwortet werden, welche invarianten Eigenschaften von den Modellen erwartet werden. Um aus TK-Trainingsdaten neue Beispiele mit bekanntem Label zu generieren, müssen Texte unter Beibehaltung ihrer Semantik umformuliert werden. Die Leistung eines Klassifikators wird verbessert, wenn die generierten Texte zusätzliche Worte und Wortsequenzen enthalten, die zur Diskriminierung der Klassen geeignet sind. Das Erweitern von textuellen Datensätzen erfordert demnach ein komplexes Verständnis von Syntax und Semantik der menschlichen Sprache. Wird einmal von Wortersetzungssystemen mit linguistischen Theasuri abgesehen, so sind bisher keine Erweiterungstechniken für gelabelte TK-Datensätze in der Literatur anzutreffen. Es ist nicht zielführend, Texte so wie Bilder mit einfachen Signaltransformationen

zu vervielfachen. Die kontextunabhängige Ersetzung oder Vertauschung von Buchstaben und Worten verursacht sprachliche Fehler und würde einzig den Rauschanteil im Datenbestand verstärken, nicht aber zu einer verbesserten Generalisierung auf die Testdaten beitragen. Regelbasierte Ersetzungssysteme können nur einfache sprachliche Zusammenhänge (z.B. Synonyme) erfassen. Ein ausreichend komplexes Sprachverständnis kann nach heutigem Kenntnisstand nur mit ML-Methoden erlernt werden.

Die Problemstellung, bei der nach abgewandelten Formulierungen von Texten gesucht wird, ist in der NLP unter dem Begriff *Paraphrasierung* bekannt (siehe Abschnitt 3.4). Die Entwicklung eines von Grund auf neuen Verfahrens zur Paraphrasierung ist nicht Themenschwerpunkt dieser Arbeit und würde den Umfang übersteigen. Eine Anforderung an das hier beschriebene Verfahren ist es, es mit einem möglichst geringen Aufwand auf bestehende Daten anwenden zu können. Anstatt also ein Verfahren zur Paraphrasierung aufzubauen, wurde nach einem Weg gesucht, die Aufgabe mithilfe von maschinellen Übersetzern zu lösen. Eine von einem MT-System generierte Übersetzung ist nur dann korrekt, wenn sie den gleichen Inhalt wie der Ursprungstext hat. MT-Systeme erlernen folglich Wissen über die sprachliche Semantik. Es wird eine Methode beschrieben, mit der dieses Wissen ausgenutzt wird, um Texte zu paraphrasieren und den TK-Datensatz zu erweitern. Wie in Abschnitt 4.2 wird Wissen des MT-Systems in den Daten des Klassifikators kodiert.

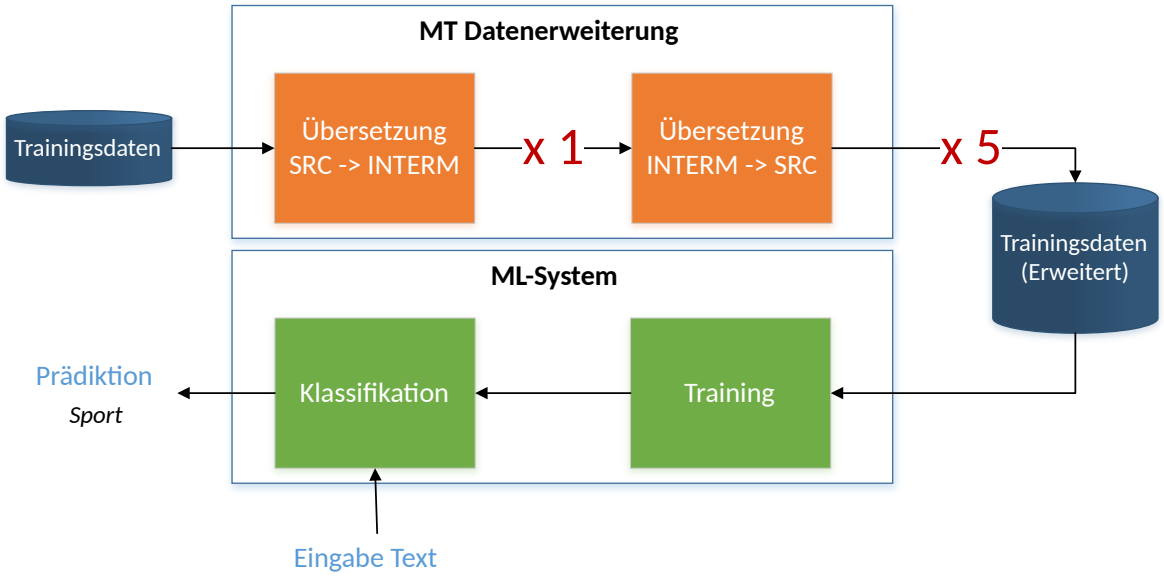


Abbildung 4.2.: Entwickeltes Verfahren zur Datenerweiterung.

Bezeichne *SRC* die Ausgangssprache des zu erweiternden Datensatzes und *INTERM* eine beliebig gewählte andere Sprache (Zwischensprache). Für das Verfahren werden die zwei Übersetzungssysteme

$$T_{SRC \rightarrow INTERM}(\cdot) \tag{4.3}$$

zur Übersetzung von *SRC* nach *INTERM* und

$$T_{INTERM \rightarrow SRC}(\cdot) \tag{4.4}$$

zur Übersetzung von *INTERM* nach *SRC* benötigt.

Algorithm 1 TDE-Verfahren mit maschinellen Übersetzern

```
1:  procedure MT-Augmentation(D)                          ▷ The training dataset
2:      D_aug ← D
3:      for all Samples (x, y) ∈ D do
4:          x_tmp ← T_{SRC→INTERM}(x)
5:          x_new ← T_{INTERM→SRC}(x_tmp)
6:          if x ≠ x_new then
7:              D_aug ← D_aug ∪ {(x_new, y)}
8:          end if
9:      end for
10:     return D_aug                                      ▷ The augmented dataset
11: end procedure
```

Algorithmus 1 zeigt den Ablauf des Verfahrens. Die Trainingsdaten werden zuerst mit $\mathrm{T}_{SRC \to INTERM}(\cdot)$ in die Zwischensprache überführt und anschließend mit $\mathrm{T}_{INTERM \to SRC}(\cdot)$ wieder rückübersetzt.

Aus dem Datensatz

$$D = \{(x_1, y_1), \cdots, (x_n, y_n)\}$$

werden neue Daten

$$D_{tmp} = \{(\mathrm{T}_{INTERM \to SRC}(\mathrm{T}_{SRC \to INTERM}(x_1)), y_1), \cdots, (\mathrm{T}_{INTERM \to SRC}(\mathrm{T}_{SRC \to INTERM}(x_n)), y_n)\}$$

generiert.

Für beide Übersetzungssysteme werden Modelle mit einer möglichst hohen Übersetzungsqualität verwendet. Die konkrete Wahl von *INTERM* ist für die grundlegende Funktionsweise des Verfahrens zunächst unerheblich. Die Effektivität des Verfahrens kann aber durch unvorhergesehene linguistischen Eigenschaften der Zwischensprache positiv oder negativ beeinflusst werden. Da in der Vergangenheit viel Arbeit in die Entwicklung von englischsprachigen Übersetzern investiert wurde, ist Englisch die bevorzugte Zwischensprache zur Erweiterung nicht-englischsprachiger Datensätze.

Bei korrekter Übersetzung trägt der Text nach zweimaliger Übersetzung wieder dieselbe Semantik wie sein Original. Im besten Fall kann der Algorithmus die Trainingsdaten verdoppeln. Da allerdings nicht garantiert werden kann, dass sich die Rückübersetzungen von ihrer ursprünglichen Formulierung unterscheiden, wird mit Zeile 6 sichergestellt, dass keine Duplikate in den Datensatz eingefügt werden. Die wirkliche Größe des erweiterten Datensatzes kann nicht beeinflusst werden und hängt maßgeblich von unvorhersehbaren Seiteneffekten (z.B. von der Domäne des TK-Tasks) ab.

Mit einer Variante kann die Zahl der erzeugten Samples nochmals gesteigert werden. State-of-the-Art MT-Modelle stellen intern ein Ranking der besten Übersetzungshypothesen auf. Ausgegeben wird die Übersetzung mit der besten Bewertung. Für das TDE-Verfahren können in Schritt 5 die n-besten Übersetzungskandidaten erstellt werden. Damit nimmt wird aber auch in kauf genommen, dass qualitativ schlechtere Übersetzungen in D_{aug} aufgenommen werden.

Häufig sind Übersetzungskandidaten zudem sehr ähnlich und unterscheiden sich nur in der Positionierung einzelner Worte. Bei der Wahl des Wertes von n muss abgewägt werden: Mit einem zu hohen n sinkt die Leistung des Klassifikators, da die Zahl der „Quasi-Duplikate“ im Datensatz steigt und es zu einer Überanpassung auf irrelevante Worte und Phrasen kommt. Auswirkungen der Wahl von n werden später in Abschnitt 5.4 evaluiert.

Da die beiden Übersetzer als Blackbox aufgefasst werden, kann auch auf existierende Lösungen wie beispielsweise kommerzielle Dienste (z.B. Google Translate oder Microsoft Translator) zurückgegriffen werden. Ebenso wie die TDE-Verfahren in der Bildverarbeitung kann das Verfahren somit schnell und mit wenig Aufwand implementiert werden. Einziger Nachteil bei der Verwendung von Blackbox-Systemen ist, dass oft keine Möglichkeit besteht, die n-besten Übersetzungen auszugeben. Wird für die Übersetzung nach *INTERM* ein mit anderen Daten trainiertes Modell als für die Übersetzung von *INTERM* nach *SRC* verwendet, steigt die Wahrscheinlichkeit, dass sich die Rückübersetzungen von den ursprünglichen Formulierungen unterscheiden.

4.3.2. Lerntransfer mit MT-Systemen

In Unterabschnitt 4.3.1 wurde ein leicht anzuwendendes Verfahren zur Generierung von synthetische Trainingsbeispielen vorgestellt. Es kann eingesetzt werden, um bei beschränkten Trainingsdaten mit minimalem Aufwand kleine Verbesserungen in der Leistungsfähigkeit eines TK-Systems herbeizuführen. Folgende Schwachpunkte konnten bei dem TDE-Verfahren festgestellt werden:

- Die Zahl der generierten Trainingsbeispiele hängt von Zufallseffekten ab und kann nicht aktiv beeinflusst werden.
- Die synthetischen Beispiele unterscheiden sich oft nur geringfügig von der ursprünglichen Formulierung. „Quasi-Duplikate“ können einen negativen Effekt auf die Generalisierungsfähigkeit des Klassifikators haben.

Ein weiterer Nachteil, der bislang noch keine Erwähnung fand ist die Tatsache, dass mit dem Verfahren nicht das volle Potential des Wissens des Übersetzers ausgeschöpft werden kann. Durch das Training mit großen Datensätzen sind moderne MT-Systeme dazu imstande, detaillierte Sprachmodelle zu erlernen. Bei dem TDE-Verfahren besteht keine direkte Korrelation zwischen der erreichbaren Performancesteigerung des Klassifikators und der Anzahl und Qualität der Trainingsdaten für die beiden Übersetzer.

In diesem Unterabschnitt soll nun eine zweite Methode zur Leistungssteigerung bei beschränkten Trainingsdaten vorgestellt werden, bei dem das interne Wissensmodell eines MT-Systems auf ein TK-System übertragen wird. Werden bei neuronalen Architekturen die Gewichte zwischen verschiedenen Tasks übertragen, so wird von einer Initialisierung des Netzes mit *vortrainierten Gewichten* (engl. pretrained weights) gesprochen. Das Verfahren wird während dem Training des Klassifikators durchgeführt und setzt ein neuronales Klassifikationsmodell voraus.

Wie in der Einleitung zu Abschnitt 4.3 bereits erläutert wurde, haben sich in der Bildverarbeitung Lerntransfer-Techniken zu einem beliebten Werkzeug zur Reduzierung der Trainingszeiten

und Verbesserung der Leistung von CNNs entwickelt. Häufig werden generische Architekturen wie das GoogLeNet[94] oder VGG-19[88] als Ausgangspunkt für die Entwicklung neuer Verfahren herangezogen. Das Netz wird zunächst mit zufälligen Gewichten initialisiert und mit den Daten eines ressourcenreichen Lerntasks trainiert. Ein Task, der sich aufgrund seiner immensen Datensatzgröße besondererer Beliebtheit beim Lerntransfer erfreut, ist der ImageNet-Datensatz[22, 51] zur Bildklassifikation. Er wurde manuell mithilfe des Amazon Crowd-Sourcing Tools *Turk* gelabelt und besteht aus 15 Millionen hochaufösenden Bildern die 22.000 Kategorien zugeteilt wurden. Nach dem Training werden die erlernten Gewichte dann zur Initialisierung des Modells eines zweiten Tasks wiederverwendet. Da tief gelegene Schichten taskunabhägige Merkmale erlernen, benötigt der zweite Tasks nun weniger Trainingsdaten und konvergiert beim Training schneller. Je ähnlicher die Domänen von Quell- und Zieltasks sind, desto stärker profitiert der Zieltask von den vortrainierten Gewichten. Für gängige Deep Learning Frameworks wie Caffe, Tensorflow oder Theano stehen für gängige Architekturen bereits vortrainierte Modelle zum freien Download bereit.[1] Wiederverwertbare Modelle machen den Einsatz von Lerntransfer besonders lohnenswert, da Entwickler ohne zusätzlichen Aufwand von dem Wissen früherer Tasks profitieren.

4.3.2.1. Wissensrepräsentation in NMT-Systemen

Die genaueren Voraussetzungen, die erfüllt sein müssen, damit beim Training neuronaler Netze durch den Lerntransfer eine Leistungssteigerung eintritt, sind nach jetzigem Stand noch nicht abschließend geklärt. Ob bei einem Modell mit vortrainierten Gewichten ein Vorteil gegenüber zufällig initialisierten Gewichten gemessen werden kann, muss individuell für jeden Lerntask evaluiert werden. Eine notwendige Bedingung für eine Leistungssteigerung ist eine *Verwandschaft* zwischen Quell- und Zieltasks. In der zeitgleich zu dieser Arbeit erstellten Publikation [64] stellen die Autoren McCann et al. die Hypothese auf, dass der MT-Task für die Sprachverarbeitung ein vergleichbares Potential in sich birgt, wie der ImageNet-Task für die Bildverarbeitung. Die starken Synergien zwischen maschineller Übersetzung und Textklassifikation können wie folgt begründet werden: Die Problemstellung beim Training von MT-Systemen ist es, einen Text aus der einen Sprache möglichst detailgetreu in eine andere Zielsprache zu überführen. Für die Korrektheit der Übersetzung ist es notwendig, dass die beiden Texte semantisch äquivalent sind. Das gelernte semantische Wissen von Sätzen und Phrasen kann von Klassifikatoren für die Trennung der Klassen genutzt werden.

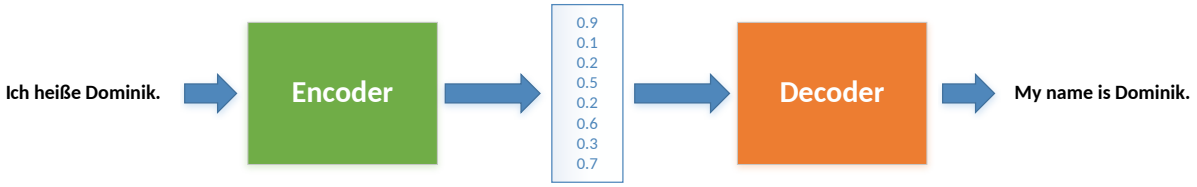


Abbildung 4.3.: Sequence-to-Sequence Modell (vereinfacht).

Als Modell für die neuronale maschinelle Übersetzung (NMT) kommen überwiegend Varianten der rekurrenten Sequence-to-Sequence (Seq2Seq)-Architektur nach Sutskever et al.[93]

[1] siehe z.B. Caffe Model-Zoo unter `https://github.com/BVLC/caffe/wiki/Model-Zoo`.

und Cho et al.[15] (siehe auch Kapitel 2) zum Einsatz. Seq2Seq-Architekturen lassen sich in die beiden Teilkomponenten *Encoder* und *Dekoder* unterteilen (siehe Abbildung 4.3). Der *Encoder* liest beliebig lange Eingabetexte ein und überführt sie in eine Vektorrepräsentation fester Länge. Der *Dekoder* wertet anschließend den vom *Encoder* berechneten Vektor aus, um aus ihm den Übersetzungstext in der Zielsprache zu generieren. Die vollständigen für die Übersetzung benötigten Informationen müssen folglich in der niedrigdimensionalen Zwischenrepräsentation zwischen *Encoder* und *Decoder* kodiert sein. Der Übergang stellt einen Flaschenhals in der Seq2Seq-Architektur dar.

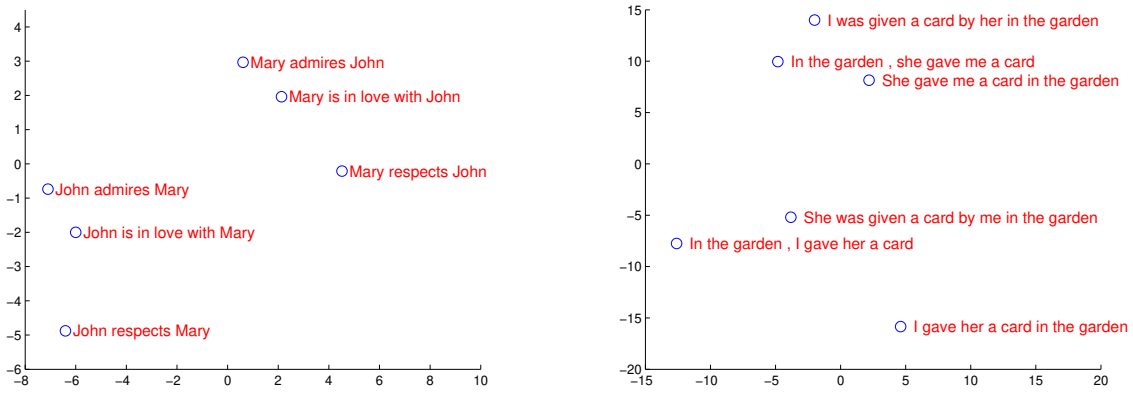


Abbildung 4.4.: Analyse der Zwischenrepräsentation zwischen *Encoder* und *Dekoder* (2-Dimensionale PCA Projektion). Quelle: [93].

An einem Flaschenhals muss ein NN beim Training eine möglichst kompakte Repräsentation der Eingabedaten finden und dabei dennoch alle für die Lösung des Tasks benötigten Informationen beibehalten. Beim Training maschineller Übersetzer hat dies zur Folge, dass die Ausgabe des *Encoder* alle für die Übersetzung benötigten syntaktischen und semantischen Eigenschaften erfassen muss[15]. Sutskever et al. machen bei ihrer Analyse in [93] die Beobachtung, dass von den gelernten Merkmalen semantische Cluster gebildet werden. Links in Abbildung 4.4 wird dies anhand verschiedener Varianten einer Phrase beispielhaft veranschaulicht. Auf der rechten Seite des Diagramms ist zudem zu sehen, wie die Vektoren unempfindlich gegenüber aktiven und passiven Satzkonstruktionen sind. Das abstrakte Sprachverständnis des Encoder-Modells kann mit dem Lerntransfer auf andere NLP-Modelle übertragen werden. Die Arbeit von Hill et al.[38] zeigt, dass die Repräsentationen des *Encoders* zur Lösung des Semantic-Textual-Similarity-Tasks verwendet werden können. Dasselbe Wissen wird auch von Textklassifikation benötigt. Es kann gefolgert werden, dass die Leistung von Textklassifikatoren durch das Modellwissen von NMT-Systemen gesteigert werden kann.

Für das Training von NMT-Systemen konnten in den letzten Jahren große öffentliche Datenbestände aufgebaut werden. In der Conference on Machine Translation (Workshop on Machine Translation, kurz: WMT), einer jährlich abgehalten Konferenz zum Thema maschinelle Übersetzung, werden regelmäßig von professionellen Übersetzern angefertigte parallele Datensätze veröffentlicht. Die Fülle an verfügbaren Trainingsdaten hat den Vorteil, dass es auch bei komplexen Modellen mit vielen freien Parametern nicht zur Überanpassung kommt. Der Wissenstransfer auf TK-Systeme ist dann besonders lohnenswert, wenn eine Korrelation

der Performance des Klassifikators mit Größe und Qualität des Trainingsdatensatzes für den Übersetzer besteht.

4.3.2.2. Modellbeschreibung

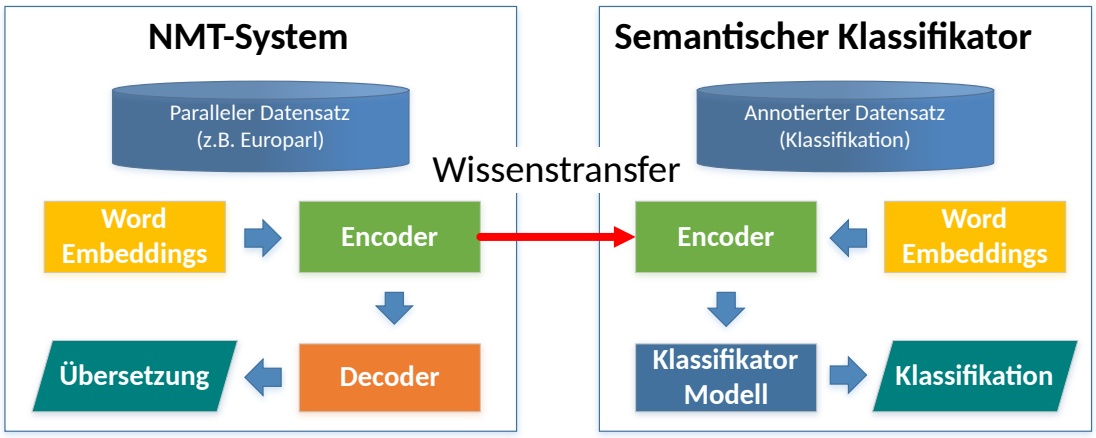


Abbildung 4.5.: Entwickeltes Verfahren zum Lerntransfer.

Nach der Definition 1 aus Abschnitt 2.2 soll beim Lerntransfer das Wissen eines Lerntasks $\mathcal{T}_S$ einer Quelldomäne $\mathcal{D}_S$ auf einen Lerntask $\mathcal{T}_T$ einer Zieldomäne $\mathcal{D}_T$ übertragen werden. Ziel ist es, die prädiktive Zielfunktion $f_T(\cdot)$ in $\mathcal{D}_T$ unter Verwendung des Wissens aus $\mathcal{D}_S$ und $\mathcal{T}_S$ zu verbessern.

In diesem Abschnitt soll in der Quelldomäne ein neuronaler maschineller Übersetzer $\mathrm{T}_{LANG \rightarrow INTERM}(\cdot)$ zur Übersetzung von Texten der Sprache *LANGUAGE* in eine beliebige andere Sprache *INTERM* trainiert werden. Für das Training des Übersetzers steht ein umfangreicher paralleler Datensatz D_S bereit. Das im Modell des Übersetzers kodierte Wissen soll zur Verbesserung des Textklassifikators $f_T(\cdot)$ der Sprache *LANGUAGE* verwendet werden. Die Trainingsdaten D_T zum Training von $f_T(\cdot)$ sind beschränkt.

Da Gewichte zwischen den neuronalen Modellen übertragen werden sollen, müssen die beiden Architekturen von Übersetzer und Klassifikator aufeinander abgestimmt sein. Für den Übersetzer wird die einfach gehaltene Seq2Seq-Architektur nach Cho et al.[15] (siehe Abschnitt 2.4) verwendet. Es wird bewusst auf einen Attentionmechanismus verzichtet, da dieser weitere Pfade zwischen Encoder und Decoder einfügt und somit die Bottleneckeigenschaft der gelernten Repräsentation abschwächt.

Das Seq2Seq-Modell $\mathrm{T}_{LANG \rightarrow INTERM}(\cdot)$ lässt sich Beschreiben durch die beiden Teilfunktionen $\mathrm{ENC}_{LANG \rightarrow VEC}$ und $\mathrm{DEC}_{VEC \rightarrow INTERM}$. Das heißt die Übersetzung w_{trans} einer Wortsequenz w ergibt sich aus:

$$\begin{aligned} w_{trans} &= \mathrm{T}_{LANG \rightarrow INTERM}(w) \\ &= \mathrm{DEC}_{VEC \rightarrow INTERM}(\mathrm{ENC}_{LANG \rightarrow VEC}(w)) \end{aligned}$$

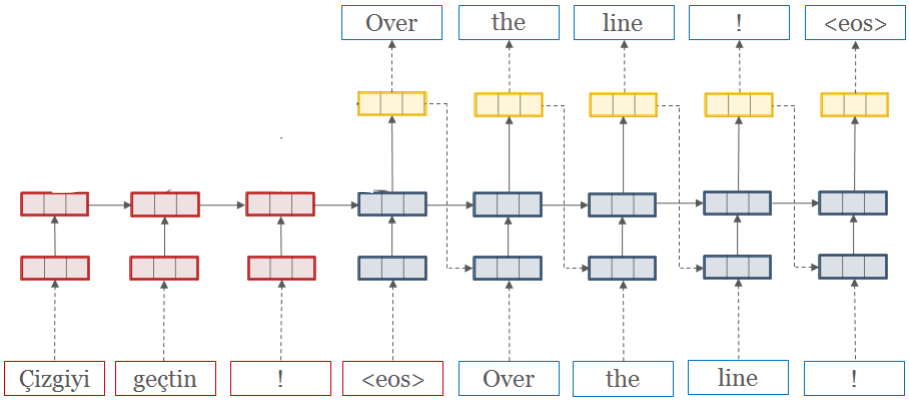


Abbildung 4.6.: Schematische Darstellung eines NMT-Systems. Quelle: [50].

Mit VEC wird dabei die vom *Encoder* ausgegebene Vektorrepräsentation für w bezeichnet. Beim Lerntransfer soll das Modell von $\text{ENC}_{LANG \to VEC}$ auf den Klassifikator $f_T(\cdot)$ übertragen werden. Die These ist, dass das Erlernen der Zielfunktion $f_T(\text{ENC}_{LANG \to VEC}(\cdot))$ leichter ist als das Training von $f_T(\cdot)$. Die Vermutung stützt sich insbesondere darauf, dass $dim(VEC) \ll dim(\mathcal{X})$ gilt. Das heißt die Dimensionalität von VEC ist wesentlich kleiner als die Dimensionalität des Merkmalsraums der Wortsequenzen. Wegen der geringeren Dimensionalität der Eingabe müssen für $f_T(\text{ENC}_{LANG \to VEC}(\cdot))$ weniger Parameter trainiert werden.

Da auch schon das MT-System mit Word Embeddings initialisiert wird, ergibt sich insgesamt ein dreistufiger Trainingsprozess, bei dem in jedem Schritt das zuvor gelernte Wissen wiederverwendet wird, um ein abstrakteres Sprachverständnis zu erlernen und die Komplexität des jeweiligen Lernproblems zu senken. Die Zahl der verfügbaren Trainingsdaten nimmt über die Stufen stetig ab. Begonnen wird mit dem unüberwachten Training von Wortvektoren. Hierfür stehen nahezu unbegrenzt Textressourcen bereit. Anschließend wird dieses Wissen für das Training eines NMT-Modells verwendet. Für MT-Systeme werden gelabelte Daten benötigt. Die Korpa sind aber im Vergleich zu anderen NLP-Tasks vergleichsweise groß. Im letzten Schritt wird das Wissen des Übersetzers auf den Textklassifikator übertragen. Bei vielen TK-Tasks sind nur wenige Beispiele vorhanden.

5. Evaluation

In diesem Kapitel sollen die in Kapitel 4 beschriebenen Verfahren experimentell evaluiert werden. Dafür werden in Abschnitt 5.1 zunächst die zur Evaluation verwendeten Datensätze beschrieben. In Abschnitt 5.2 werden neuronale Architekturen für die Textklassifikation trainiert, um ein Baselinemodell für die weiteren Experimente zu bestimmen. In Abschnitt 5.3 werden die drei vorgestellten CTK-Verfahren miteinander verglichen und in Abschnitt 5.4 die Verfahren zur Reduktion der Fehlerrate ausführlich untersucht.

5.1. Datensätze

Um die Effektivität der erarbeiteten Verfahren auswerten zu können, müssen Modelle für neuronale Textklassifikatoren trainiert werden. Im weiteren Verlauf dieses Abschnittes werden die Trainingstasks und die für das Training verwendeten Datensätze eingeführt. Insgesamt werden die Verfahren anhand von elf verschiedene Datensätze evaluiert. Tabelle 5.1 gibt einen Überblick über die jeweiligen Tasks und die Anzahl der Trainings- und Testbeispiele.

Datensatz	Aufgabe	Details	Training	Test
NEWS-DE	Topic Klassifikation	5 Klassen, einzelne Sätze	80.000	20.000
NEWS-EN	Topic Klassifikation	5 Klassen, einzelne Sätze	80.000	20.000
E-ERWT	Binäre Klassifikation	2 Klassen, einzelne Sätze	1.444	KV
E-FW	Binäre Klassifikation	2 Klassen, einzelne Sätze	1.714	KV
E-FSN	Binäre Klassifikation	2 Klassen, einzelne Sätze	712	KV
E-GS	Binäre Klassifikation	2 Klassen, einzelne Sätze	636	KV
E-MB	Binäre Klassifikation	2 Klassen, einzelne Sätze	751	KV
E-RA	Binäre Klassifikation	2 Klassen, einzelne Sätze	960	KV
E-SA	Binäre Klassifikation	2 Klassen, einzelne Sätze	734	KV
SEM17	Sentimentanalyse	3 Klassen, Tweets (max. 140 Zeichen)	50.333	12.284
IMDb	Sentimentanalyse	2 Klassen, mehrere Sätze	25.000	25.000

Tabelle 5.1.: Überblick der evaluierten TK-Datensätze.

5.1.1. Geschäftssignale

Sieben deutschsprachige TK-Datensätze wurden für die Arbeit von der Karlsruher Firma Echobot GmbH zur Verfügung gestellt. Die Echobot GmbH ist ein Software- und Dienstleistungsunternehmen, das Lösungen zur Online-Medienbeobachtung anbietet. Das Ziel der

Echobot GmbH ist, ihre Kunden darüber aufzuklären, wie im Internet momentan über sie, deren Produkte und Mitbewerber berichtet wird. Die gewährten Einblicke können dann dazu genutzt werden, um Geschäftsstrategien an die im Internet beobachteten Trends auszurichten. Von einem Webcrawler werden täglich mehrere hunderttausend Nachrichtenartikel und Social-Media Postings durchsucht. Die gefundenen Informationen müssen aufbereitet und mit Methoden des maschinellen Lernens strukturiert und analysiert werden.

Ein konkretes Einsatzgebiet von ML-Methoden bei Echobot sind die Geschäftssignale. Kunden können sich automatisiert darüber informieren lassen, wenn neu veröffentlichter Web-Content ein für sie relevantes Thema betrifft. Beispielsweise kann eine Firma sich über Erweiterungen oder Stellenangebote ihrer Konkurrenten informieren lassen, um frühzeitig die Entwicklung neuer Produkte zu erkennen. Ein Führungswechsel bei Mitbewerbern kann von Personalabteilungen genutzt werden, um potentielle Mitarbeiter anzuwerben.

Die Problemstellung der Erkennung von Geschäftssignalen wird als binäres TK-Problem aufgefasst. Für jedes der sieben betrachteten Signale steht ein separater Datensatz mit positiven und negativen Trainingsbeispielen zur Verfügung. Die Beispiele bestehen aus kurzen Textausschnitten, zumeist einzelne Sätze, mit einer durchschnittlichen Länge von rund 20 Wörtern. Die besondere Schwierigkeit ist die Größe der Datensätze, die aus jeweils 600 bis 1.700 Samples bestehen. Da zudem ein starkes Ungleichgewicht zwischen der Verteilung der Klassen besteht, wurden die negativen Beispiele von einem Domänenexperten so erstellt, dass sie schwer von den positiven Beispielen zu diskriminieren sind. Die Stichproben wurden so gewählt, dass in etwa gleich viele positive wie negative Daten vorliegen.

Folgende Geschäftssignale werden in dieser Arbeit evaluiert:

- **Erweiterung (E-ERWT)**: Textauschnitte die Firmenerweiterungen zum Thema haben.
  Beispiele:
  Erweiterung der Marktpräsenz: Eröffnung des Vertriebsbüros in Shanghai (China) → +
  Worauf muss man bei der Finanzierung eines Neubaus achten? → −

- **Führungswechsel (E-FW)**: Textausschnitte die sich mit dem Wechsel von Führungspersonal beschäftigen.
  Beispiele:
  Der Weltklimarat (IPCC) wählt im Oktober in Dubrovnik einen neuen Vorstand. → +
  Schulleiter Ewald Jüchems hieß die 160 neuen Schüler willkommen. → −

- **Fusion (E-FSN)**: Textausschnitte zum Thema Firmenfusion.
  Beispiele:
  Futterland und ZooRoyal.de fusionieren – DuMont Venture greift zooplus.de an. → +
  Diese Aufgabe hat für den europäischen Raum RIPE NCC in Amsterdam übernommen. → −

- **Gewinnspiel (E-GS)**: Textausschnitte die Gewinnspiele zum Thema haben.
  Beispiele:
  Nicht vergessen, bei unserem aktuellen Gewinnspiel mitzuspielen. → +
  Um das Finale dann zu gewinnen, brauchst du nicht nur Qualität, sondern auch Glück. → −

- **Messebesuch (E-MB)**: Textausschnitte in denen Messebesuche von Firmen angekündigt werden.

Beispiele:
Besuchen Sie uns auf der Messe in Veldhoven. → +
Die Ausstellung kann zu den Öffnungszeiten des Rathauses besucht werden. → −

- **Rückrufaktion (E-RA)**: Textausschnitte zu Produktrückrufaktionen.
  Beispiele:
  Die Firma Henkel hat zwei Bodenreiniger zurückgerufen. → +
  Orban pfeift auf Rücknahme von Flüchtlingen → −

- **Stellenangebot (E-SA)**: Textausschnitte mit Stellenausschreibungen.
  Beispiele:
  Zum weiteren personellen Ausbau der Firma suchen wir einen SPS-Programmierer (m/w) → +
  Wir suchen also nach Wertschriften, die auf lange Sicht attraktiv sind. → −

5.1.2. Nachrichtenressorts

Um bei der Evaluation der Verfahren aus Abschnitt 4.2 Aussagen über den Einfluss der Übersetzungsqualität auf die Leistung der Klassifikatoren machen zu können, werden gelabelte, natürlichsprachliche Texte in Quell- und Zielsprache benötigt. Der Vergleich und die Reproduzierbarkeit von Arbeiten zur crosslingualen Textklassifikation ist schwierig, da Autoren ihre Verfahren anhand von unterschiedlichen Datensätzen mit einer ungenau beschriebenen Vorverarbeitung testen[20]. Der einzige aus der Literatur bekannte TK-Datensatz mit deutschen und englischen Samples ist der *Reuters Corpus Volume 2* (RCV2). Da dieser für den Autor dieser Arbeit nicht zugänglich ist, wurde stattdessen entschieden, einen neuen Datensatz zu erstellen und diesen im Anschluss der Öffentlichkeit zugänglich zu machen.

Zur Durchführung einer Sensitivitätsanalyse der Klassifikatoren wurde ein Datensatz mit möglichst vielen gelabelten Beispielen benötigt, weswegen ein manuelles Labeling nicht in Betracht gezogen wurde. Die Aufgabenstellung wurde so definiert, dass gelabelte Daten aus dem Internet gecrawlt werden können. Aufgabe ist es, Textausschnitte aus Online-Nachrichtenartikeln gemäß ihres Ressorts zu kategorisieren. Die deutschsprachigen Daten werden mit *NEWS-DE* und die englischsprachigen Daten mit *NEWS-EN* bezeichnet. Die Klassen sind *Politik*, *Wirtschaft*, *Sport*, *Gesundheit* und *Wissenschaft*. Der Datensatz soll automatisiert ohne manuelles Labeling aufgebaut werden. Die Textauschnitte stammen aus den inhaltlichen Zusammenfassungen von Nachrichtenartikeln wie sie beispielsweise im HTML Meta Tag *Description* der meisten Nachrichtenseite zu finden sind. Das Ressort kann aus der Artikel-URL oder ebenfalls aus den HTML Meta Tags entnommen werden. Als Crawler wird das Open-Source Tool *Scrapy*[1] verwendet. Für die Trainingsbeispiele werden die Texte in Sätze zerlegt und aus der Gesamtmenge aller gecrawlten Artikel zufällige Sätze ausgewählt. Durch Unterabtastung (engl. *undersampling*) werden die Kategorien balanciert, so dass von allen Ressorts gleich viele Beispiele vorliegen. Für *NEWS-DE* wird das Archiv von Spiegel-Online und der Süddeutschen Zeitung durchsucht. Für *NEWS-EN* das Archiv der New York Times, der BBC und des Guardians. Die Artikel stammen aus dem Zeitraum zwischen dem 01. Januar 2000 und dem 01. September

[1] `https://scrapy.org/`

Datensatz	Text	Klasse
NEWS-DE	*Geburtstag von Staatsgründer Kim Il Sung - und stößt Drohungen gegen Donald Trump aus.*	Politik
	Der Dax bricht um mehr als fünf Prozent ein.	Wirtschaft
	Auch Lothar Matthäus schießt gegen den neuen Bayern-Coach.	Sport
	Neue Studien machen klar, wie schnell uns die Werbung überlisten kann.	Wissenschaft
	Dann übernimmt die Krankenkasse häufig die Kosten.	Gesundheit
NEWS-EN	*Prime minister woken at midnight to be given notice of raid.*	Politik
	It could allow the company to raise prices while limiting complaints from the public.	Wirtschaft
	A proposal to introduce sin-bins in football is to be considered by the FA.	Sport
	But many researchers still blame an earthquake.	Wissenschaft
	Luckily she turned out to be perfectly healthy.	Gesundheit

Tabelle 5.2.: Beispiele: Nachrichtenressort-Datensatz

2017. Die Samples werden gleichmäßig aus dieser Zeitspanne entnommen. Um die beiden Sprachdomänen möglichst gleich zu halten, werden lokal-politische Nachrichten nicht mit aufgenommen. Für beide Sprachen liegen 85.000 Trainingsbeispiel mit 17.000 Beispielen pro Ressort vor. Eine Grafik zur Verteilung der Beispiele über die Zeit ist im Anhang in Abschnitt A.1 zu finden.

5.1.3. Öffentliche Datensätze

Um eine Vergleichbarkeit der Ergebnisse mit anderen Arbeiten herzustellen, werden zusätzlich zu den obigen Datensätzen zwei weitere englischsprachige Datensätze evaluiert.

- Der **SEM17** Datensatz gehört zum *Sentiment Analysis in Twitter*-Task[82] des 2017 abgehaltenen Internationalen Workshop on Semantic Evaluation (SemEval). Die Aufgabe besteht darin, die in Tweets geäußerte Haltungen (Sentiments) als positiv, neutral oder negativ zu erkennen.
  Beispiele:
  Do you have a CD burner and cable? → neutral
  i cant go. i have a meeting on that day. → negativ
  Happy new year! ^_^ → positiv

- Der **IMDb** Datensatz ist der 2011 veröffentlichte *Large Movie Review Dataset*[60]. Er besteht aus einem Trainings-Set und einem Test-Set mit jeweils 25.000 Filmbewertungen. Die Bewertungen sollen einer positiven oder negativen Polarität zugeordnet werden. Die

zu klassifizierenden Texte bestehen aus mehreren Sätzen.
Beispiele:
Widow hires a psychopath as a handyman. Sloppy film noir thriller which doesn't make much of its tension promising set-up. (3/10) → negativ
This movie turned out to be better than I had expected it to be. Some parts were pretty funny. It was nice to have a movie with a new plot. → positiv

5.1.4. Vorverarbeitung

Eines der Ziel dieser Arbeit ist es, zu evaluieren, ob mit neuronalen Netzen das Feature-Engineering gegenüber traditionellen TK-Verfahren vereinfacht werden kann. Es wird deshalb auf eine aufwendige Präparierung der Datensätze verzichtet. Zur Vorverarbeitung der Daten wird die NLP-Bibliothek Spacy[2] eingesetzt. Es werden alle Zeichen entfernt, die nicht dem Unicodestandard entsprechen und Währungssymbole durch deren Bezeichner ersetzt (z.B. € → `EUR`). Zudem werden Telefonnummern, URLs, E-Mail-Adressen und Zahlen durch eine feste Zeichensequenz `*PHONE*`, `*URL*`, `*EMAIL*` und `*NUMBER*` ersetzt. Datensatzspezifische Vorverarbeitungen werden nicht durchgeführt.

5.1.5. Aufteilung in Training und Test-Set

Für die Aufteilung der Daten in eine Trainings- und Testmenge wird sich an die Angaben aus Tabelle 5.1 gehalten. Für die *NEWS*-Datensatz liegen für beide Sprachen jeweils 25.000 Testbeispiele vor. Wegen der geringen Anzahl von Beispielen bei den Geschäftssignalen wird bei ihnen eine 10-fach stratifizierte Kreuzvalidierung (KV) vorgenommen. Dafür wird die Datenmenge in zehn möglichst gleich große Teilmengen $T_1, \cdots, T_{10}$ aufgeteilt. Es werden zehn Testdurchläufe gestartet, bei denen die jeweils i-te Teilmenge T_i als Testmenge und die restlichen neun Teilmengen als Trainingsmenge verwendet werden. Bei der *stratifizierten* Kreuzvalidierung (KV) wird bei der Bildung der Teilmengen zudem darauf geachtet, dass die Verteilung der Klassen in allen Mengen annähernd gleich bleibt.

5.2. Neuronaler Baselinemodelle

Für die Klassifikation der Geschäftssignale werden in der Praxis bislang Support Vektor Maschinen (SVMs) eingesetzt. In einer ersten Versuchsreihe soll beurteilt werden, ob neuronale Klassifikatoren auf den Datensätzen dieselbe Leistung wie traditionelle Verfahren erzielen können. Die Schwierigkeit beim Entwurf neuronaler Modelle liegt darin, dass neben der Architektur noch eine Reihe weiterer Hyperparameter festzulegen sind (z.B. Netztiefe, Dimensionierung der Schichten, Aktivierungsfunktionen, Dropout, Regularisierung). Werden die Parameter nicht optimal gewählt, kommt es zum Over- oder Underfitting auf die Trainingsdaten. Die vielen Hyperparameter sind zugleich auch ein Vorteil von NNs, da die vielen Einstellungsmöglichkeiten auch viel Raum für Verbesserungen bieten.

[2] `https://spacy.io/`

In Abschnitt 3.1 wurden drei Architekturmodelle für neuronale Textklassifikatoren beschrieben: Convolutional Neural Networks (CNNs), Reccurent Neural Networks (RNNs) und die Deep Averaging Networks (DAN). Es werden Modelle für jede dieser Architekturen entworfen und auf den Geschäftssignalen trainiert. Die Hyperparameter werden dabei für alle Datensätze gleich gewählt. Eine Optimierung auf einzelne Datensätze kann zwar geringfügig bessere Ergebnisse erzielen, jedoch würde damit das Argument für NNs wegen ihres geringerem Aufwands für die Featureentwicklung ad absurdum geführt. Um die optimalen Werte für die jeweiligen Hyperparameter zu finden, werden Modelle mit verschiedenen Werte eines Hyperparameters trainiert, während die restlichen Parameter unverändert bleiben.

5.2.1. Modelle und Hyperparameter

Im Folgenden werden die drei Modelle nach Durchführung des Parametertunings beschrieben:

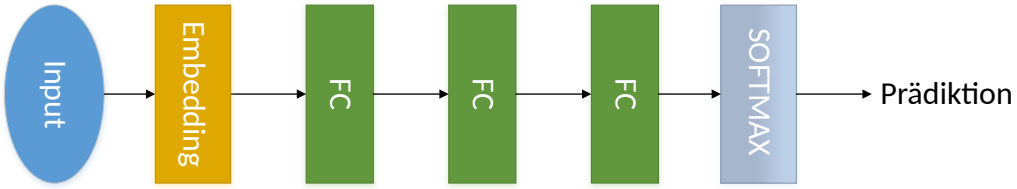


Abbildung 5.1.: Modell 1: Deep Averaging Network (DAN)

Das Deep Averaging Network (DAN, siehe Abbildung 5.1) besteht aus vier hidden Layer mit einer abschließenden Sigmoid- (binär) oder Softmax- (kategorisch) Ausgabeschicht. Die erste Schicht ist eine Embedding Layer und bildet Wortindizes auf deren Embedding ab. Auf die Embeddings wird anschließend mit einer Wahrscheinlichkeit von 0.3 ein Dropout durchgeführt. Danach wird der Durchschnitt entlang der Zeitachse über die Embeddings berechnet. Auf den 300-dimensionalen Ausgabevektor folgen drei FC-Schichten. Sie bestehen jeweils aus 16 Neuronen mit ReLu-Ausgabefunktion. Vor der Ausgabeschicht wird eine Dropout-Schicht mit dem Faktor 0.5 eingefügt. Zudem wird auf den FC-Schichten eine L2-Regularisierung von 0.003 angewandt, um zusätzliches Overfitting zu vermeiden.

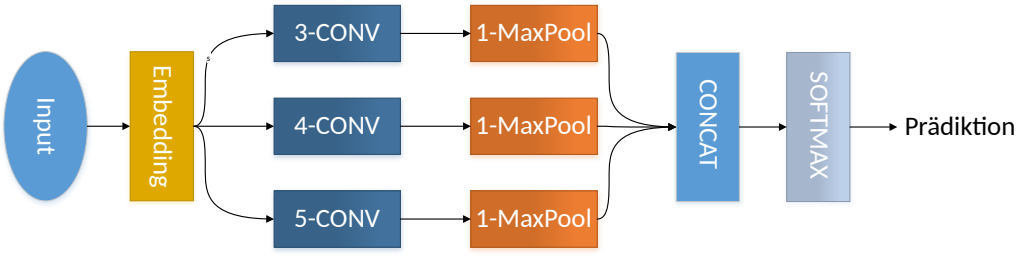


Abbildung 5.2.: Modell 2: Convolutional Network (CNN)

Das CNN (siehe Abbildung 5.2) wurde aufgebaut wie in [48] beschrieben. Für die Wahl der Hyperparameter wurde sich an der Analyse von [112] orientiert. Das Netz besteht aus nur zwei hidden Layers. Zu beginn steht wie auch schon beim `DAN` eine 300-dimensionale Embedding Layer. Im Anschluss folgen drei parallele Convolution-Schichten mit Filtergrößen 3, 4 und

5. Pro Schicht werden 10 Filter trainiert. Nach einem Max Pooling werden die Convolution-Schichten konkateniert. Es folgt ein Dropout von 0.5 und eine Sigmoid- (binär) oder Softmax- (kategorisch) Ausgabeschicht (L2-Regularisierung von 3).

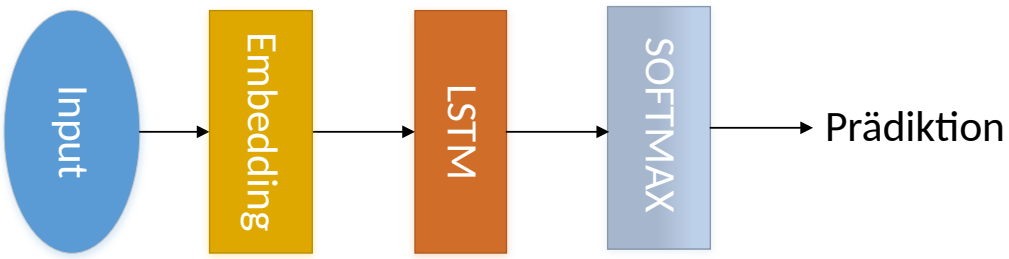


Abbildung 5.3.: Modell 3: LSTM

Es wurden Experimente mit mehrschichtigen und einschichtigen LSTM-Netzen durchgeführt. Die beste Performance wird mit einem einfachen, einschichtigem LSTM erzielt (siehe Abbildung 5.3). Am Netzanfang steht eine 300-dimensionale Embedding Layer. Die LSTM-Schicht besteht aus 32 Neuronen mit einem Dropout von 0.5. Die Ausgabe ist eine Sigmoid- (binär) oder Softmax (kategorisch) FC-Schicht. Experimente mit einem zusätzlichen Attention-Mechanismus brachten keine zusätzliche Leistungssteigerung hervor. Es wird vermutet, dass für einen Attention-Mechanismus nicht ausreichend viele Trainingsdaten für die Geschäftssignale vorliegen.

Alle die Modelle werden mit einer Lernrate von 0.001 trainiert. Für das Gradientenverfahren wird der Adam-Optimizer[49] der Keras-Bibliothek verwendet.

Die Parameter für den Gradientenabstieg sind:

- $\beta_1 = 0.9$
- $\beta_2 = 0.999$
- $\epsilon = 1e - 07$

Als Fehlerfunktion wird der Crossentropy-Loss verwendet. Die Lernrate wird über die Epochen nicht reduziert. Zudem wird, um einen explodierenden Gradienten zu vermeiden, bei zu großen Werten ein Clipping auf den Wert 5.0 durchgeführt. Es wird für 25 Epochen mit einer Batchgröße von 256 trainiert. Das Modell mit dem besten F1-Score auf einem Validierungssplit (stratifiziert 10% der Trainingsdaten) wird für die Evaluation auf der Testmenge verwendet.

5.2.2. Vortrainierte Wortvektoren

Vortrainierte Wortvektoren sind ein häufig verwendetes Werkzeug in der NLP, um bei Ermangelung eines großen gelabelten Trainingsdatensatzes geringere Fehlerraten zu erzielen. Die erste Schicht der drei neuronalen Arichtekturens ist eine Embedding Layer, mit der die Wortindizes der Eingabesequenz auf die entsprechenden Wortvektoren abgebildet werden. Es soll evaluiert werden, wie groß der Effekt von vortrainierten Vektoren auf die Modelle zur Klassifikation der Geschäftssignale ist.

Dafür werden mit dem *GloVe*-Algorithmus[76] trainierte, 300-dimensionale Word Embeddings verwendet. Für das Training der deutschen Vektoren wird ein 5GB großen Webcrawl Datensatz mit circa 600.000 verschiedenen Wörtern genutzt. Für die englischsprachigen Embeddings werden die vortrainierten *CommonCrawl-840B* Vektoren[3] der GloVe-Autoren verwendet.

Evaluiert werden alle drei Architekturen mit den in Unterabschnitt 5.2.1 beschriebenen Hyperparametern. Es werden drei verschiedene Versuchsaufbauten betrachtet:

1. **Zufällig**: Die *Embedding Layer* werden bei Generierung des Modells zufällig initialisiert. Die Zufallszahlen werden gleichverteilt aus dem Intervall $[-0.05; 0.05]$ gezogen. Während des Trainings des Klassifikators werden die Embeddings mittrainiert.

2. **Vortrainiert**: Die Wortvektoren werden, wie oben beschrieben, mit dem *GloVe*-Algorithmus trainiert. Die Gewichte der Embedding-Schicht werden anschließend mit den Wortvektoren initialisiert. Während dem Training des Klassifikators bleiben diese konstant.

3. **Feintuning**: Die Wortvektoren werden wie oben beschrieben mit dem *GloVe*-Algorithmus trainiert. Danach werden die Gewichte der Embedding-Schicht mit den Wortvektoren initialisiert. Während dem Training des Klassifikators werden für 2 Epochen (Wert empirisch bestimmt) mit geringer Lernrate die Embeddings auf die Klassifikationsaufgabe angepasst.

Datensatz	Zufällig			Vortrainiert			Feintuning		
	DAN	CNN	LSTM	DAN	CNN	LSTM	DAN	CNN	LSTM
E-ERWT	83, 8%	81, 2%	82, 3%	84, 0%	81, 7%	85, 9%	84, 6%	85, 9%	86, 5%
E-FW	87, 5%	88, 2%	89, 1%	87, 4%	90, 1%	91, 5%	87, 2%	90, 1%	91, 5%
E-FSN	85, 6%	83, 0%	82, 0%	84, 1%	85, 4%	86, 0%	84, 2%	85, 4%	86, 7%
E-GS	93, 2%	92, 8%	94, 4%	91, 6%	95, 6%	95, 8%	91, 6%	95, 8%	96, 0%
E-MB	90, 2%	87, 0%	89, 3%	89, 5%	91, 5%	92, 2%	89, 6%	92, 6%	93, 0%
E-RA	94, 0%	92, 4%	92, 1%	92, 5%	93, 0%	94, 0%	93, 0%	93, 9%	94, 2%
E-SA	95, 8%	96, 8%	98, 0%	97, 2%	95, 2%	98, 3%	97, 2%	98, 1%	98, 8%

Tabelle 5.3.: Vergleich verschiedener neuronaler Architekturen mit unterschiedlichen Initialisierungen der Wortvektoren (Accuracy).

In Tabelle Tabelle 5.3 sind die Ergebnisse der Evaluation aufgeführt. Es kann beobachtet werden, dass mit zufällig initialisierten Wortvektoren die einfache `DAN`-Architektur die geringste Fehlerrate aufweist. Sie schlägt die beiden anderen Architekturen in vier der sieben Datensätze. Das `CNN` liegt immer wenige Prozentpunkte hinter der Leistung des `LSTM`s. Mit den vortrainierten Wortvektoren verändert sich die Situation. Die *Accuracy* von `CNN` und `LSTM` steigt durch die zusätzlichen semantischen Informationen in den Embeddings, während beim `DAN` keine große Steigerung eintritt. Die beste Performance wird erreicht, indem die Embeddings auf die Klassifikationsaufgabe anpasst werden (Feintuning). Bei dem `LSTM` konnte durch vortrainierte Vektoren eine durchschnitte Steigerung von 2.8 Prozentpunkten erzielt werden.

[3] `https://nlp.stanford.edu/projects/glove/`

5.2.3. Vergleich mit traditionellen Verfahren

Nachfolgend wird die Güte des neuronalen Klassifikatoren mit den zuvor mit traditionellen Verfahren erzielten Baselinewerten vergleichen. Die bisherigen Bestwerte für die Klassifikation der Geschäftssignale werden der Arbeit „Lexical Substitution with Word Embeddings: A General Approach to Improve Short Text Classification"[91] von A. Di Stefano entnommen. Er beschäftigte sich 2017 mit der Verbesserung von traditionellen TK-Systemen bei begrenzten Trainingsdaten. Dafür entwickelte er ein Verfahren zur Berechnung von niedrigdimensionalen Features, um den Klassifikatoren zusätzliche semantische Informationen zuzuführen. Seine Methoden evaluiert er unter anderem mit State-of-the-Art Support Vektor Maschine (SVM)-Modellen. Hierbei verwendet er die Matlab-Implementierung einer $NBSVM$ mit dem Parameter $\beta = 0.25$.

Datensatz	DAN		CNN		LSTM		**Di Stefano[91]**		**Base**
	ACC	Δ	ACC	Δ	ACC	Δ	ACC	Δ	
E-ERWT	84, 6%	+2, 9	85, 9%	+4, 2	86, 5%	+4, 8	84, 1%	+2, 4	81, 7%
E-FW	87, 2%	−3, 2	90, 1%	−0, 3	91, 5%	+1, 1	92, 1%	+1, 7	90, 4%
E-FSN	84, 2%	±0, 0	85, 4%	+1, 2	86, 7%	+2, 5	86, 4%	+2, 2	84, 2%
E-GS	91, 6%	−1, 2	95, 8%	+3, 0	96, 0%	+3, 2	94, 7%	+1, 9	92, 8%
E-MB	89, 6%	+0, 2	92, 6%	+3, 2	93, 0%	+3, 6	91, 4%	+2, 0	89, 4%
E-RA	93, 0%	+0, 4	93, 9%	+1, 3	94, 2%	+1, 6	93, 8%	+1, 2	92, 6%
E-SA	97, 2%	−0, 2	98, 1%	+0, 7	98, 8%	+1, 4	98, 4%	+1, 0	97, 4%
AVG	89, 6%	−0, 2	91, 7%	+1, 9	92, 2%	+2, 6	91, 6%	+1, 8	89, 8%

Tabelle 5.4.: Vergleich der Ergebnisse der neuronalen Verfahren mit den Baselinewerten (Accuracy).

In Tabelle 5.4 werden die Bestwerte der neuronalen TK-Verfahren den in [91] genannten Ergebnissen gegenüber gestellt. Bis auf den Datensatz `E-FW` kann allen Datensätzen eine erhöhte Genauigkeit gemessen werden. Bei der kleinen Datensatzgröße unterliegen jedoch insbesondere neuronale Netze großen Schwankungen zwischen den KV-Splits. Verbesserungen von unter < 1 Prozentpunkt sind deshalb nicht sehr aussagekräftig. Dennoch kann gefolgert werden, dass neuronale Netze auch bei kleinen Trainingsdatensätzen an die Leistung anderer Verfahren anknüpfen können. Dafür ist kein Feature-Engineering und keine datensatzabhängige Optimierung der Architektur notwendig.

5.3. Crosslinguale Textklassifikation mit maschinellen Übersetzern

In diesem Abschnitt werden die CTK-Verfahren aus Abschnitt 4.2 bewertet. Als Voraussetzung für die Anwendung der Verfahren müssen zunächst einmal maschinelle Übersetzungssysteme trainiert werden. Da die Evaluation mithilfe des zweisprachigen `NEWS`-Task durchgeführt werden soll, müssen Systeme $\mathrm{T}_{DE \to ENG}(\cdot)$ und $\mathrm{T}_{ENG \to DE}(\cdot)$ aufgebaut werden.

5.3.1. Training der maschinellen Übersetzer

Für das Training eines MT-Systeme wird der Datensatz des *News Translation Shared Task* des Worshop on Machine Translation (WMT) 2017 verwendet. Er besteht aus insgesamt rund 7 Millionen parallelen Sätzen in deutscher und englischer Sprache. Die Texte stammen aus 4 unterschiedlichen Domänen:

- Mitschriften aus dem europäischen Parlament (*Europarl v7*)
- Diverse Webseiten (*Common Crawl*)
- Nachrichtenkommentare (*News Commentary v12*)
- Presseveröffentlichungen der EU (*Rapid corpus of EU press releases*)

Die Trainingsdaten werden mit dem Moses Toolkit präpariert (Tokenisierung und True-Casing). Das Modell und die Trainingsparameter werden aus [77] entnommen. Es wurde mit dem OpenNMT-Toolkit[50] erstellt und entspricht einer klassischen Encoder-Decoder-Architektur bestehend aus zwei LSTM-Schichten à 1.024 Neuronen. Optimierungen wurden wie [77] beschrieben durchgeführt („Context Gate for atten-tional model“). Zur Übersetzung seltener Worte werden die Daten vor der Übersetzung mit einem Byte-Paar-Encoding (BPE, [86]) vorverarbeitet. Bei der Übersetzung von Deutsch nach Englisch wird auf dem `newstest2016` Datensatz des WMT ein BLEU-Score von 38, 39% erzielt. Bei der Übersetzung von Englisch nach Deutsch beträgt der Wert 35, 23%.

5.3.2. Baseline: Klassifikation der Nachrichtenressorts

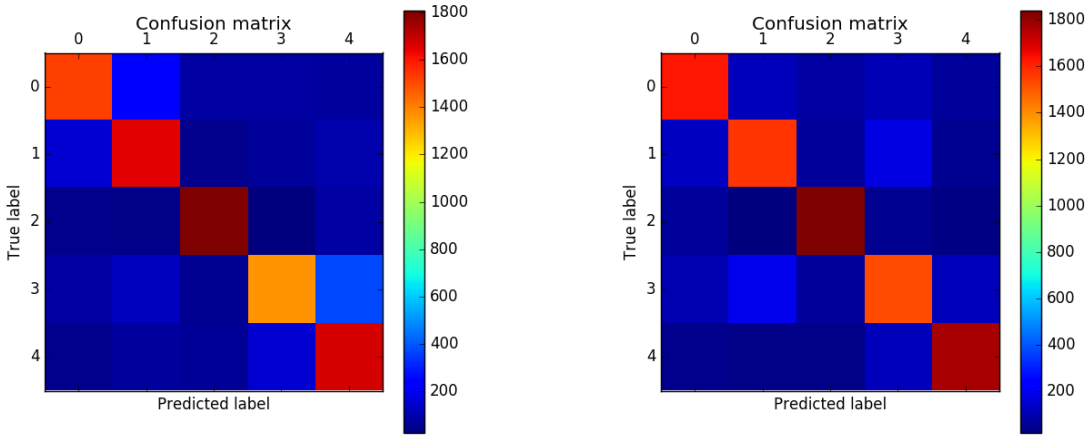
Zur Auswertung der Verfahren zur CTK werden Messwerte benötigt, welche Güte der Klassifikatoren beim Training auf natürlichen Texten, das heißt, Texte, die nicht mit einem Übersetzer erstellt wurden, erreichen kann. Für das Experiment wird die in Unterabschnitt 5.2.1 beschriebe CNN-Architektur verwendet. Bei dem Mehrklassenproblem wird die Sigmoid-Ausgabefunktion in der Ausgangsschicht durch eine Softmax-Ausgabe ersetzt. Als Metrik für die Beurteilung der Modelle wird der F1-Score genutzt.

Abbildung 5.4 zeigt die Konfusionsmatrix bei der die Klassifikation der Testdaten. Es ist zu erkennen, dass die beiden Klassen Politik und Wirtschaft sowie Technik und Gesundheit schwer voneinander zu trennen sind. Sport-Artikel können dagegen in beiden Sprachen leicht von den anderen Klassen diskriminiert werden. Zudem fällt auf, dass im englischsprachigen Raum oft nicht wie im deutschen zwischen Technik und Wirtschaftsnachrichten differenziert wird. Die Unterscheidung der beiden Klassen ist im englischen daher schwerer.

Tabelle 5.5 zeigt die quantitativen Ergebnisse der Evaluation. Auch hier kann beobachtet werden, dass die Ressorts in `NEWS-EN` und `NEWS-DE` teilweise nach anderen Kriterien zugeteilt wurden. Dies begründet sich dadurch, dass als Quelle jeweils unterschiedliche Nachrichtenportale gecrawlt wurden. Auf `NEWS-EN` wird ein 2, 9 höherer F1-Score als auf `NEWS-DE` erreicht.

5.3.3. Quantitative Analyse

Mit den Ergebnissen aus Unterabschnitt 5.3.2 kann nun ausgewertet werden, wie gut die Leistung der Klassifikatoren mit den CTK-Verfahren aus Abschnitt 4.2 ist. Ziel ist es, einen



(a) NEWS-DE

(b) NEWS-EN

Abbildung 5.4.: Konfusionsmatrizen der Klassifikation beider News-Datensätze

Vergleich der Verfahren vorzunehmen und den Performanceabfall bei der CTK mit State-of-the-Art Übersetzern zu messen.

Die gemessenen Ergebnisse sind in Tabelle 5.6 dargestellt. Zur Klassifikation wurde wieder die CNN-Architektur aus Unterabschnitt 5.2.1 eingesetzt. Festzustellen ist, dass bei beiden Übersetzungsrichtungen mit allen Verfahren zufriedenstellende Resultate erzielt werden können. Der Leistungsabfall liegt zwischen 5 und 10 Prozentpunkten. Auch bei der CTK liegen die Werte für `NEWS-EN` über denen von `NEWS-DE`. Die Methode 1 (synthetische Trainingsdaten) und die Methode 2 (synthetische Testdaten) weisen in etwa einen gleich hohen F1-Wert auf. Mit der Methode 3 (synthetische Labels) fällt der Wert noch einmal rund 2 Prozentpunkte höher aus. Als Erklärung hierfür kann angeführt werden, dass der Klassifikator in der Methode 3 auf natürlichsprachigen Texten der Zielsprache trainiert wird. Texte, die mit der in der Zielsprache gebräuchlichen, domänenspezifischen Sprache geschrieben wurden, scheinen zu einem besseren Trainingsprozess zu führen, als das direkte Training auf den Übersetzungen (Methode 1).

5.3.4. Qualitative Analyse

Unter idealen Voraussetzungen kann mit einem CTK-Verfahren in der Quell- und Zielsprache eine, von statistischen Messfehlern abgesehen, identische Leistungen erzielt werden. In der Realität hängt die Akkuratheit eines crosslingualen Klassifikators von mehreren Einfluss- und Störfaktoren ab.

Alle drei angewandten Verfahren benötigen für ihre Funktionalität leistungsstarke Übersetzungssysteme. Die Qualität der Übersetzungen beeinflussen maßgeblich, wie gut der Übergang zwischen den Sprachen gelingen kann. Wird der Klassifikator der ersten Methode mit fehlerhaften Übersetzungen trainiert, kann er die Komplexität der Trainingsaufgabe nicht vollständig erfassen. Darunter leidet die Generalisierungsfähigkeit auf ungesehen Texten der Zielsprache.

Klasse	Precision	Recall	F1-Score
Politik	82, 6%	76, 2%	79, 3%
Wirtschaft	77, 8%	82, 7%	80, 1%
Sport	88, 2%	90, 4%	89, 3%
Technik	81, 0%	68, 2%	74, 0%
Gesundheit	72, 9%	83, 6%	77, 9%
	80, 5%	80, 2%	80, 1%

(a) NEWS-DE

Klasse	Precision	Recall	F1-Score
Politik	82, 7%	81, 6%	82, 1%
Wirtschaft	81, 3%	78, 6%	80, 0%
Sport	88, 3%	92, 0%	90, 2%
Technik	77, 3%	76, 4%	76, 9%
Gesundheit	87, 3%	88, 7%	88, 0%
	83, 4%	83, 5%	83, 4%

(b) NEWS-EN

Tabelle 5.5.: Ergebnisse für die Klassifikation der beiden NEWS-Datensätze (Precision, Recall und F1-Score).

Quelle → Ziel	**Methode 1**			**Methode 2**			**Methode 3**		
	P	R	F1	P	R	F1	P	R	F1
EN → DE	73, 4%	72, 8%	73, 1%	73, 8%	73, 9%	73, 8%	75, 5%	75, 1%	75, 3%
DE → EN	76, 3%	77, 0%	76, 6%	75, 1%	75, 2%	75, 1%	78, 9%	78, 5%	78, 7%

Tabelle 5.6.: Ergebnisse der Evaluation der CTK-Verfahren.
Methode 1: Synthetische Trainingsdaten, Methode 2: Synthetische Testdaten, Methode 3: Synthetische Label.

Das gleiche Problem trifft auch auf die Klassifikatoren der zweiten und dritten Methode zu, wenn fehlerhafte Übersetzungen klassifiziert werden sollen.

Die daraus resultierende Frage ist, ob mit einem fehlerfreien Übersetzungssystem bei der CTK dieselben Fehlerraten wie beim Training mit natürlichsprachigen Texte der Zielsprache (siehe Unterabschnitt 5.3.3) erzielt werden kann oder aber, ob auch dann noch mit einem Leistungsabfall zu rechnen ist, wenn durch den Übersetzer keine semantischen Fehler gemacht werden. Mit dieser Fragestellung beschäftigten sich 2011 Duh, Fujino und Nagata in ihrem Leitartikel „Is Machine Translation Ripe for Cross-lingual Sentiment Classification?“[26]. Die Ergebnisse ihrer Analyse soll nun im Folgenden erläutert werden.

Die Klassifikationsfehler eines auf Übersetzer basierten CTK-Verfahrens lassen sich nach [44] in zwei Kategorien einteilen:

1. Fehlzuordnungen der Label,

2. Unterschiede der Domäne in Quell- und Zielsprache.

Bei einer Fehlzuordordnung der Labels hat sich durch eine falsche Übersetzung die Semantik und damit auch das Referenzlabel (Ground Truth) des Texts verändert. Dies lässt sich mit den Definitionen aus Abschnitt 2.2 folgendermaßen formalisieren: Für die prädiktiven Zielfunktionen f_S in der Quellsprache und f_T in der Zielsprache gilt für ein fehlerhaft übersetztes Beispiel x_s

$$f_S(x_S) = p(y_S|x_S) \neq p(y_T|\,\mathrm{T}_{SRC\rightarrow TRGT}(x_s)) = f_T(\mathrm{T}_{SRC\rightarrow TRGT}(x_s)) \tag{5.1}$$

für Labels y_S in der Quelldomäne und y_T in der Zieldomäne. Als Beispiel werde der Text

Die `CDU` verliert mehr als fünf prozent. → *Politik*

aus dem Datensatz `NEWS-DE` betrachtet. Wird durch ein Fehler im Übersetzungssystem der Text zu

The `DAX` loses more than five percent. → *Wirtschaft*

übersetzt, entsteht ein Trainingsbeispiel, bei dem `DAX` fälschlicherweise mit dem Label Politik assoziiert wird. In der Quell- und Zielsprache gilt demzufolge $p_S(y = politik|x = DAX) < p_T(y = politik|x = DAX)$. Abhilfe bei diesen Fehlern können nur bessere Übersetzungssysteme oder das manuelle Aussortieren falsch übersetzter Trainingsdaten schaffen. Nach den Beobachtungen von [26] machen CTK-Systeme bei der Qualität heutiger MT-Systeme nur selten Fehler in der Labelzuordnung.

Der andere Fehlertyp entsteht durch Unterschiede in den beiden Sprachdomänen und ist bei vielen CTK-Tasks der schwerwiegendere Störfaktor. Bei vielen Tasks stimmt die klassenbedingte Wahrscheinlichkeitsverteilung der Beispiele zwischen unterschiedlichen Sprachen nicht überein. Bei den beiden Nachrichtenressort-Datensätzen erfolgt die Einteilung nicht immer nach denselben Kriterien. Im Datensatz `NEWS-DE` findet sich das Trainingsbeispiel

Zwischen dem weltgrößtem Softwarekonzern Microsoft und Google, der Nummer eins bei den Internetsuchmaschinen, kracht es gewaltig. → Wirtschaft

während entsprechende Texte im `NEWS-EN` Datensatz in die Kategorie Technik einsortiert wurden:

The alliance between Google and Apple may make the prospect of outdueling Microsoft's empire better than ever. → Technik

Ein anderer Grund für abweichende Wortverteilungen sind sprachliche Eigenheiten in den beiden Sprachregionen. Eine englische Nachrichtenseite verwenden in ihren Nachrichten möglicherweise ein anderes Vokabular als eine deutsche Nachrichtenseiten. MT-Systeme können die Übersetzungen nicht an die fach- und domänenspezifische Sprache des Klassifikationstasks anpassen, da sie im Regelfall mit Trainingsdaten aus unterschiedlichen Domänen trainiert werden. Semantisch korrekte Übersetzungen können dennoch eine „unnatürliche" Wortwahl bezüglich der in der Zieldomäne gebräuchlichen Sprache darstellen. Zudem sind aktuelle MT-Systeme bei weitem noch nicht dazu in der Lage, die vollständige Vielfalt und Komplexität der menschlichen Sprache abzubilden. Wortschatz und Satzstrukturen von automatisch generierten Texten sind eingeschränkt und können bei genauerer Analyse von der natürlichen Sprache

unterschieden werden. Dies bestätigt sich auch darin, dass mit Methode 3 die besten Ergebnisse erzielen werden konnten. Zwar ist auch sie den Schwächen des Übersetzungssystems $T_{EN \to DE}$ unterworfen, der Klassifikator wird hier aber auf natürlichen Texten der Zieldomäne trainiert.

Die oben beschriebene Problemstellung hat zur Folge, dass die generierten Übersetzungen eines MT-Systems mit ihrer Wortverteilungen einen systematischen Bias auf einen crosslingualen Klassifikatoreninduzieren. Im anderen Kontext des maschinellen Lernens ist dieses Problem unter dem Begriff Domänenadaptionsproblem (DAP) bekannt. Domänenadaptionsproblem (DAP)s sind ein häufig widerkehrendes Problem in der NLP[44] und zur Lösung stehen etablierte ML-Algorithmen bereit, die auch auf die CTK angewandt werden können. Verbreitete Verfahren für Adaptionsprobleme sind die transduktiven SVMs[8]. Mit einem speziell für die CTK entwickeltem Co-Training Algorithmus beschäftigte sich 2009 bereits Wan et al.[100]. In dieser Arbeit werden keine zusätzlichen Anstrengungen zur Korrektur der Wortverteilungen unternommen. Es kann aber vermutet werden, dass sich das Ausmaß des Problems durch die Verbreitung vortrainierter Wordembeddings in den letzten Jahren gemindert hat. Aufgrund der Tatsache, dass Word Embeddings von semantisch ähnlichen Wörtern eine ähnliche Vektorrepräsentation haben, spielt es eine untergeordnete Rolle, welche Wörter im Einzelfall von den Übersetzern generiert werden und ob diese typisch für die Sprachdomäne sind.

5.4. Textklassifikation bei begrenzten Trainingsdaten

In diesem Abschnitt werden die Verfahren aus Abschnitt 4.3 evaluiert. Es folgt in Unterabschnitt 5.4.1 die Auswertung des Trainingsdatenerweiterungsverfahren (TDE-Verfahren) und in Unterabschnitt 5.4.1 die Auswertung des Verfahrens zum neuronalen Lerntransfer. Abschließend werden Unterabschnitt 5.4.3 die beiden Verfahren gemeinsam beurteilt und miteinander verglichen.

5.4.1. Trainingsdatenerweiterung mit MT-Systemen

In das TDE-Verfahren wird der Übersetzer als Blackbox eingebunden. Bis auf eine Option, die n-besten Übersetzungen ausgeben zu können, gibt es keine weiteren Anforderungen an das System. Als Zwischensprache wird für die deutschsprachigen Datensätze Englisch und für englischsprachigen Datensätze Deutsch verwendet. Es kann dasselbe NMT-System wie schon für die Evaluation der Abschnitt 5.3 Verfahren verwendet werden. Das System erzielt auf dem `newstest2016`-Datensatz einen BLEU-Score von 39, 39% bei der Übersetzung von Deutsch nach Englisch und einen BLEU-Score von 35, 23% bei der Übersetzung von Englisch nach Deutsch.

Als Klassifikator wird wieder das CNN-Modell aus Unterabschnitt 5.2.1 eingesetzt. Für jeden der elf Datensätze wird zuerst das Modell auf dem ursprünglichen Datensatz trainiert und die Performance gemessen. Das Ergebnis wird als wert für die nachfolgenden Tests festgelegt. Im Anschluss werden die Trainingsdaten mit den n-besten Übersetzungen erweitert und jeweils erneut die Performance auf den nicht erweiterten Testdaten ausgewertet. Es werden Experimente für $n \in \{1, 2, 4, 8\}$ durchgeführt.

In Tabelle Tabelle 5.7 sind die Ergebnisse der Evaluation quantitativ dargestellt. Auf neun der elf Datensätze wird ein Performanceanstieg gemessen. Dieser Beträgt durchschnittlich 0.3 Prozentpunkte. Auf den anderen neun Datensätzen stieg die Leistung bei kleinen n an. Für

Datensatz	*n*-best	Score
NEWS-DE	Base	80.1%
	1	80, 2%
	2	80, 4%
	4	80, 4%
	8	76, 0%
NEWS-EN	Base	83, 4%
	1	83, 4%
	2	83, 2%
	4	83, 5%
	8	79, 8%
E-ERWT	Base	85, 9%
	1	84, 7%
	2	85, 1%
	4	85, 2%
	8	84, 5%
E-FW	Base	90, 1%
	1	90, 7%
	2	91, 4%
	4	91, 5%
	8	91, 2%
E-FSN	Base	85, 4%
	1	85, 6%
	2	85, 2%
	4	84, 9%
	8	77, 3%
E-GS	Base	95, 8%
	1	94, 9%
	2	92, 3%
	4	94, 9%
	8	93, 1%

Datensatz	*n*-best	Score
E-MB	Base	92, 6%
	1	92, 7%
	2	92, 7%
	4	92, 9%
	8	91, 4%
E-RA	Base	93, 9%
	1	94, 1%
	2	94, 1%
	4	94, 4%
	8	78, 9%
E-SA	Base	98, 1%
	1	98, 2%
	2	99, 1%
	4	99, 2%
	8	90, 9%
SEM17	Base	62, 3%
	1	62, 0%
	2	62, 3%
	4	62, 6%
	8	62, 3%
IMDb	Base	87, 8%
	1	87, 9%
	2	87, 9%
	4	88, 0%
	8	85, 2%

Tabelle 5.7.: Ergebnisse der Evaluation des TDE-Verfahren.
Bei den Geschäftssignalen wurde die Accuracy gemessen, für die Datensätze `NEWS-DE`, `NEWS-EN`, `SEM17` und `IMDb` der F1-Score.

große n fällt die Leistung unter die Werte des Baselinemodells. Die bei NEWS-DE (+0, 3), NEWS-EN (+0, 1), SEM17 (+0, 3) und IMDb (0, 2) gemessen Ergebnisse sind aufgrund der vielen Trainings- und Testdaten stabil. Bei den Geschäftssignalen haben die Resultate keine ausreichende Signifikanz, da zwischen den KV-Iterationen Schwankungen von ±2 Prozentpunkte nachgewiesen werden. Bei den Datensätze E-ERWT (−0.7) und E-GS (−0.9) tritt durch die Erweiterung eine Verschlechterung ein. Die beste Leistung wird insgesamt bei $n = 4$ gemessen. E-FSN bildet hier eine Ausnahme, wo der Spitzenwert (+0.2 Prozentpunkte) bei $n = 1$ erreicht wird.

Beispiel 1

Doch er will seine umstrittenen Dekrete nicht zurücknehmen. → Politik
Seine umstrittenen Dekrete will er aber nicht zurücknehmen. → Politik
Doch seine umstrittenen Dekrete will er nicht zurücknehmen. → Politik
Aber er will seine umstrittenen Dekrete nicht zurücknehmen. → Politik
Er will seine umstrittenen Dekrete aber nicht zurücknehmen. → Politik
Er will aber seine umstrittenen Dekrete nicht zurücknehmen. → Politik
Aber seine umstrittenen Dekrete will er nicht zurücknehmen. → Politik
Er will aber nicht seine umstrittenen Dekrete zurücknehmen. → Politik

Beispiel 2

Die Staatsanwaltschaft hat Ermittlungen aufgenommen. → Wirtschaft
Die Staatsanwaltschaft hat Ermittlungen eingeleitet. → Wirtschaft
Die Staatsanwaltschaft hat eine Untersuchung eingeleitet. → Wirtschaft
Die Staatsanwaltschaft habe Ermittlungen aufgenommen. → Wirtschaft
Die Staatsanwaltschaft Hannover hat Ermittlungen aufgenommen. → Wirtschaft
Die Staatsanwaltschaft hat Ermittlungen eröffnet. → Wirtschaft
Die Staatsanwaltschaft hat eine Untersuchung eröffnet. → Wirtschaft
Die Staatsanwaltschaft hat Ermittlungen eingestellt. → Wirtschaft

Für die Erklärung des Leistungsabfall für große n wird das TDE-Verfahren qualitativ analysiert. Beispielhaft werden zwei repräsentative Trainingstexte aus dem erweiterten NEWS-DE Datensatz (mit $n = 8$) betrachtet. Die Beispiele zeigen, dass durch die Übersetzungen neue Trainingsbeispiele mit geändertem Wortschatz und Satzstruktur erzeugt werden. Der Klassifikator kann daraus zusätzliches semantisches Wissen extrahieren. Allerdings unterscheiden sich die n-besten Übersetzungen häufig nur in wenigen Worten (z.B. *aufgenommen*, *eingeleitet* und *eröffnet*) vom Ursprungstext. Durch eine Steigerung von n werden immer mehr Duplikate von Wörtern in den Trainingsdatensatz eingefügt, die nicht zur Diskriminierung der Klassen geeignet sind (z.B. *hat* oder *die*). Es kommt zur Überanpassung.

5.4.2. Lerntransfer mit MT-Systemen

Bei dem Verfahren zum neuronalen Lerntransfer werden die internen Gewichte eines neuronalen Übersetzers auf einen Klassifikator übertragen. Wie in Unterabschnitt 4.3.2 beschrieben, wird für das Übersetzungssystem ein einfaches Seq2Seq-Modell (siehe Unterunterabschnitt 4.3.2.2) verwendet, bei dem auf einen Attention-Mechanismus zugunsten der Bottleneckeigenschaft

der gelernten Repräsentation verzichtet wird. Das Modell wird mit Googles Deep Learning Framework Tensorflow implementiert. Dabei wird sich an die Beschreibung aus [59] gehalten und für 5000 Epochen mit 2 hidden LSTM-Layer mit jeweils 256 Neuronen und einem Dropout-Faktor von 0.2 trainiert.

Um herauszufinden, welchen Einfluss die Übersetzungsqualität auf die Leistung der Klassifikatoren hat, werden zwei unterschiedlich große parallele Datensätze für das Training der Übersetzer verwendet. Neben dem in Unterabschnitt 5.3.1 bereits beschriebenen Datensatz aus dem News Translation Shared Task des WMT 2017 (MT-Large) wird zusätzlich der Datensatz aus dem *Multimodal Machine Translation Shared Tasks* des WMT 2016 [10] verwendet (MT-Small). Er besteht aus 30.000 Flickr Bildunterschriften, die von professionellen Übersetzern von Englisch nach Deutsch übersetzt wurden. Die Sätze sind entsprechend kurz und in einfacher Sprache.

Für `MT-Small` erreicht das Seq2Seq-Modell einen BLEU-Score von 34% auf den Testdaten des Multimodal Machine Translation Shared Tasks des WMT 2016. Für `MT-Large` wird ein Score von 24.7% auf den Testdaten des News Translation Tasks des WMT 2017 erzielt. Die Zahlen dienen ausschließlich der Beurteilung der jeweiligen Qualität beider Übersetzer. Sie stehen in keinem direkten Zusammenhang zueinander, da für die Berechnung der Metriken unterschiedliche Datensätze verwendet wurden. Der geringere Score trotz einem vielfachen an Trainingsdaten bei `MT-Large` ist dadurch zu erklären, das die Testdaten für `MT-Large` schwieriger zu übersetzen sind als die relativ kurzen Phrasen beim Test von `MT-Small`.

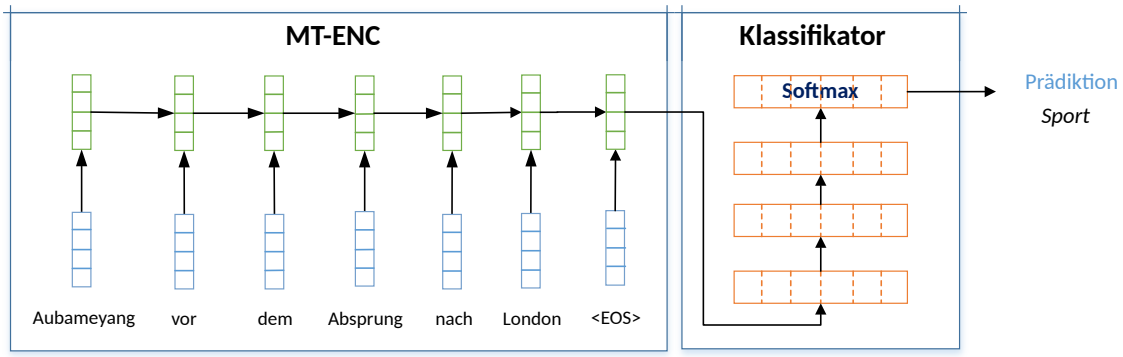


Abbildung 5.5.: Modellvariante 1 (MT-FC)
Die Ausgabe des vortrainierten Encoders wird bei der Klassifikation mehreren FC-Schichten zugeführt. **Blau**: Kodierung der Eingabesequenz, **Grün**: *Hidden States* des Encoders, **Orange**: *Hidden FC-Layer* des Klassifikators

Der Modell des Klassifikator muss am Netzeingang auf die Architektur des Übersetzers angepasst werden. Es werden zwei verschiedene Modellvarianten evaluiert:

- Die erste Variante (siehe Abbildung 5.5) ist ein Modell bestehend aus 3 FC-Schichten. Der Encoder wird als Merkmalsextrakter vorgeschaltet. Seine Gewichte bleiben beim Training des Klassifikators unverändert. Dem Modell steht kein explizites Wissen über die Wortsequenzen zur Verfügung. Die Klassifikation wird ausschließlich anhand des

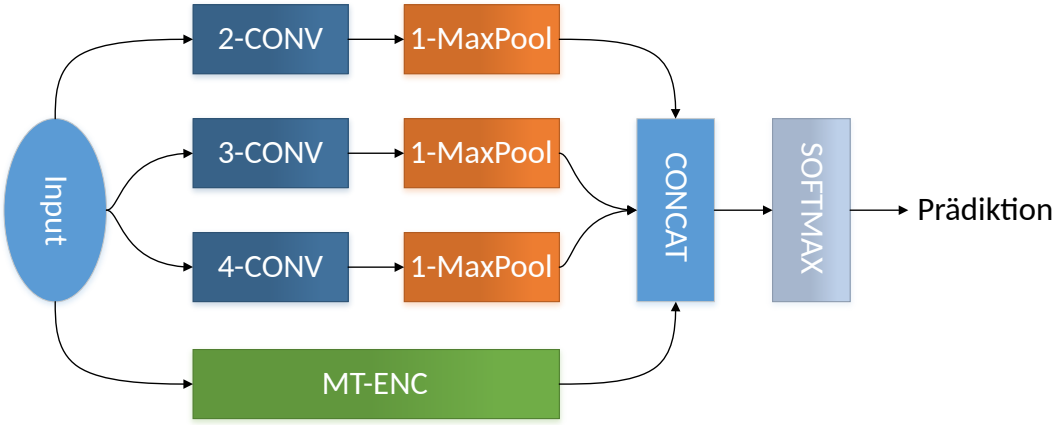


Abbildung 5.6.: Modellvariante 2 (MT-CNNMC)
CNN-Architektur wird mit *MT-ENC* Layer konkateniert.

Ausgabevektors des Encoders gefällt. Die hidden FC-Layer bestehen aus 16 Neuronen. Es werden *ReLu*-Aktivierungsfunktionen und eine L2-Reularisierung von 0.003 eingesetzt. Vor der Ausgabeschicht wird beim Training ein Dropout von 0.3 angewandt.

- Die zweite Variante (siehe Abbildung 5.6) ist eine Erweiterung der CNN-Architektur aus Unterabschnitt 5.2.1. Der gelernte Encoder *MT-ENC* wird mit der Ausgabe der MaxPooling-Schichten konkateniert. Der Klassifikator kann dadurch selbständig lernen, wie stark er die sequentiellen Informationen der CNNs in die Entscheidung mit einbezieht. Das CNN wird gleich dimensioniert wie die Architektur in Unterabschnitt 5.2.1.

Als Baseline werden wieder die in Unterabschnitt 5.3.2 mit der CNN-Architektur gemessenen Werte festgelegt. Es werden Übersetzer für `MT-Small` und `MT-Large` aufgebaut und für jeden der Datensätze die beiden Modelle `MT-FC` und `MT-CNN` mit den Trainingsparametern aus Abschnitt 5.2) trainiert.

In Tabelle Tabelle 5.8 sind die Ergebnisse aufgeführt, die mit den beiden Modellvarianten erzielt wurden. Mit den auf `MT-Small` trainierten Encodern kann keine Leistungssteigerung erzielt werden. Bei der Modellvariante `MT-FC` liegen die Messwerte hier deutlich unter denen des Baselinemodells. Mit den zusätzlichen CNN-Schichten erhält das Modell taskpezifische, sequentielles Wissen, wodurch die Leistung wieder steigt. Besser sind die für `MT-Large` gemessenen Werte. Beide Modellvarianten erreichen hier einen Leistungssprung zu den bisherigen Ergebnissen. Bis auf die Datensätzen `E-FSN`, `E-GS` und `IMDb` tritt auf allen Datensätze eine Verbesserung ein. Die höchste Steigerung konnte auf dem Datensatz `E-RA` mit einem Anstieg von 1, 5 Prozentpunkten erzielt werden. Das `MT-CNN` erreicht sowohl bei `MT-Small` als auch bei `MT-Large` bessere Werte als `MT-FC`.

5.4.3. Vergleich der beiden Verfahren

Tabelle 5.9 vergleicht die jeweils beste Leistung der Verfahren miteinander. Das TDE-Verfahren erzielt auf neun der elf Datensätze eine Verbesserung im Vergleich zum Baselinemodell. Durchschnittlich steigt die Leistung um 0, 3 Prozentpunkte. Bei dem Verfahren zum neuronalen

Datensatz	Übersetzer	MT-FC	MT-CNN	Baseline
NEWS-DE	MT-Small	49, 1%	79, 9%	80, 1%
	MT-Large	80, 0%	82, 2%	
NEWS-EN	MT-Small	51, 3%	82, 2%	83, 4%
	MT-Large	83, 8%	84, 9%	
E-ERWT	MT-Small	62, 9%	81, 3%	85, 9%
	MT-Large	86, 3%	86, 2%	
E-FW	MT-Small	57, 9%	85, 0%	90, 1%
	MT-Large	90, 1%	92, 5%	
E-FSN	MT-Small	56, 8%	85, 3%	85, 4%
	MT-Large	84, 1%	83, 5%	
E-GS	MT-Small	62, 9%	81, 2%	95, 8%
	MT-Large	93, 3%	95, 5%	
E-MB	MT-Small	59, 9%	90, 0%	92, 6%
	MT-Large	90, 8%	93, 4%	
E-RA	MT-Small	73, 2%	90, 8%	93, 9%
	MT-Large	94, 8%	95, 4%	
E-SA	MT-Small	83, 7%	93, 3%	98, 1%
	MT-Large	99, 3%	98, 4%	
SEM17	MT-Small	58, 5%	60, 2%	62, 3%
	MT-Large	60, 3%	63, 6%	
IMDb	MT-Small	66, 8%	77, 9%	87, 8%
	MT-Large	85, 3%	87, 7%	

Tabelle 5.8.: Ergebnisse der Evaluation des Transferlearning-Verfahrens.
Bei den Geschäftssignalen wurde die Accuracy gemessen, für die Datensätze `NEWS-DE`, `NEWS-EN`, `SEM17` und `IMDb` der F1-Score.

Datensatz	TDE-Verfahren F1-Score	Δ	Transferlernen F1-Score	Δ	Baseline
NEWS-DE	80, 4%	+0, 3	82, 2%	+1, 9	80, 1%
NEWS-EN	83, 5%	+0, 1	84, 9%	+1, 5	83, 4%
E-ERWT	85, 2%	−0, 7	86, 3%	+0, 4	85, 9%
E-FW	91, 5%	+1, 4	92, 5%	+2, 4	90, 1%
E-FSN	85, 6%	+0, 2	85, 3%	−0, 1	85, 4%
E-GS	94, 9%	−0.9	95, 5%	−0, 3	95, 8%
E-MB	92, 9%	+0, 3	93, 4%	+0, 8	92, 6%
E-RA	94, 4%	+0, 5	95, 4%	+1, 5	93, 9%
E-SA	99, 2%	+1, 1	99, 3%	+1, 2	98, 1%
SEM17	62, 6%	+0, 3	63, 6%	+1, 3	62, 3%
IMDb	88, 0%	+0, 2	87, 4%	−0, 1%	87, 7%

Tabelle 5.9.: Vergleich: TDE-Verfahren und Translearning-Verfahrenn.
Bei den Geschäftssignalen wurde die Accuracy gemessen, für die Datensätze `NEWS-DE`, `NEWS-EN`, `SEM17` und `IMDb` der F1-Score.

Lerntransfer wurde auf acht der elf Datensätze eine bessere Performance gemessen. Die durchschnittliche Steigerung liegt mit 1, 0 Prozentpunkten hier deutlich über der mit dem TDE-Verfahren erzielten Steigerung. Beide Verfahren erweisen sich als Wirksam. Das TDE-Verfahren zeichnet sich vor allem durch seine leichte Anwendbarkeit aus, da der maschinelle Übersetzer nur als Black-Box in das System eingebunden werden muss. Das Verfahren zum Lerntransfer liefert im Vergleich jedoch die besseren Resultate.

6. Fazit

Die beachtlichen Erfolge auf dem Gebiet des maschinellen Lernens der letzten Jahre sind zu weiten Teilen auf Fortschritte bei den überwachten Lernverfahren zurückzuführen. In der jüngeren Zeit gelang es, insbesondere aufgrund des gestiegenen wirtschaftlichen Interesses an intelligenten Systemen, für viele Lerntasks große gelabelte Trainingsdatensätze aufzubauen. Mit den zusätzlichen Trainingsbeispielen und dank der verbesserten Rechenleistung spezialisierter Mikroprozessoren können heute wesentlich komplexere Modelle trainiert werden.

Die Gesetzmäßigkeit der stetigen Leistungssteigerung durch zusätzliche Trainingsdaten stößt an ihre Grenzen, wenn es aus Kostengründen oder aufgrund von veränderlichen Umweltbedingungen nicht möglich ist, über einen langen Zeitraum viele gelabelte Trainingsbeispiele zu generieren. Die Wissenschaft beschäftigt sich daher intensiv mit der Entwicklung von Verfahren, um ML-Systeme auch mit wenigen oder keinen Trainingsdaten aufzubauen. Ein vielversprechender Lösungsansatz für dieses Problem ist der *Lerntransfer*. Das Konzept hat seine Wurzeln in der Psychologie und Pädagogik und bezeichnet die Fähigkeit des Menschen, das Wissen einer erlernten Problemlösung auch auf andere, vergleichbare Situationen zu übertragen. Diese Fähigkeit ist auch für ML-Systeme erstrebenswert.

Der Beitrag dieser Arbeit war es, die Problemstellung der Textklassifikation hinsichtlich des oben beschriebenen Sachverhaltes zu untersuchen. TK-Tasks sind oft kurzlebig und werden über ihre Lebensdauer mehrfach angepasst. In der Regel sind nur wenige Trainingsdaten vorhanden, da diese in kurzer Zeit teuer und aufwendig gelabelt werden müssen. Die ausgeprägte Trainingsdatenknappheit stellt eine große Herausforderung für das Training neuronaler Modelle dar. In der Arbeit wurden verschiedene Verfahren vorgestellt, mit denen das bei der maschinellen Übersetzung erlernte Wissen für die Entwicklung von Textklassifikatoren genutzt werden kann.

6.1. Zusammenfassung

In dieser Arbeit wurden zwei verschiedene Problemszenarien der neuronalen TK ausgiebig untersucht:

Das erste Szenario behandelte die Crosslinguale Textklassifikation (CTK). Es wurden drei Methoden präsentiert, um mit maschinellen Übersetzern Texte einer anderen Sprache zu klassifizieren, für die keine Trainingsdaten vorhanden sind. Sie unterscheiden sich in ihrer Funktionsweise darin, ob mit den Übersetzern synthetische Trainingsdaten (Methode 1), synthetische Testdaten (Methode 2) oder synthetische Labels (Methode 3) erstellt werden. Dabei muss entweder von der Quellsprache in die Zielsprache oder von der Zielsprache in die Quellsprache übersetzt werden. Übersetzt wird entweder zur Trainingszeit (Methode 1 und 3) oder vor jeder Inferenz (Methode 2).

Das zweite Szenario setzte sich mit dem Training von Klassifikatoren auseinander, wenn die Zahl der Trainingsbeispiele nicht ausreicht und deswegen keine zufriedenstellende Performance auf den Testdaten erzielt werden kann. Es wurden zwei Verfahren beschrieben, mit deren Hilfe das Wissen von maschinellen Übersetzern in TK-Systeme integriert werden kann, um deren Leistung zu verbessern. Das erste Verfahren ist ein Trainingsdatenerweiterungsverfahren zur Generierung zusätzlicher Trainingsdaten. Das andere Verfahren verwendet die Technik des neuronalen Lerntransfer und überträgt das Modell des Übersetzers direkt auf das Modell eines Klassifikators.

In Kapitel 5 wurden die vorgestellten Verfahren einer Evaluation unterzogen. Um für die weiteren Experimente eine neuronale TK-Architektur auszuwählen, wurden zunächst drei State-of-the-Art Architekturen aus der Literatur miteinander verglichen: Ein CNN, ein LSTM sowie ein Averaging-Netz. Als Benchmark zur Optimierung der Hyperparameter wurden Datensätze verwendeten, die in der Praxis zur Detektion von sogenannten *Geschäftssignale* eingesetzt werden. Bisher wurden hierfür SVM-basierte Verfahren verwendet. Es wurde festgestellt, dass NNs mit der Leistung traditioneller Verfahren in den Wettbewerb treten können. Bei der Untersuchung des Effekts von vortrainierten Word Embeddings ergab sich bei den Datensätzen eine Leistungssteigerung von 1 bis 2 Prozentpunkten. Das beste Ergebnis konnte erzielt werden, indem die GloVe-Vektoren beim Training des Klassifikators für wenige Epochen bei geringer Lernrate auf den Lerntask angepasst wurden (Feintuning). Bei der Evaluation der Architekturen konnten die Beobachtungen aus [43] bestätigt werden: Trotz des simplen Aufbaus lag die Performance des aus FC-Schichten bestehenden DANs nur leicht hinter der Leistung der getesteten CNN- und LSTM-Modelle. Es wurde geschlussfolgert, dass bei der Textklassifikation die nicht-lineare Transformation der Eingabe wichtiger ist als die Kodierung von Wortreihenfolgen und Semantik ist. Zudem benötigte das Training von DAN-Modellen deutlich weniger Rechenzeit als das Training der CNNs und LSTMs-Netze. Die Akkuratheit der LSTMs war leicht über der der CNNs. Da CNNs mithilfe von GPUs effizienter trainiert werden können, wurden für die weiteren Untersuchungen eine CNN-Architektur genutzt.

Zur Evaluierung der CTK-Verfahren wurde zunächst ein zweisprachiger Datensatz erstellt, bei dem Sätze aus Nachrichtenartikeln einem passenden Ressort zuzuordnen sind. Insgesamt stellte sich heraus, dass beim Übergang von der englischen in die deutsche Sprache ein stärkerer Leistungsabfall eintritt als bei der umgekehrten Übersetzung von Deutsch nach Englisch. Hierfür werden hauptsächlich Unterschiede in den Wortverteilungen zwischen den deutschen und englischen Sprachdomänen verantwortlich gemacht (engl. *Domain Mismatch*). Die Ressorteinteilung erfolgt bei englischen und deutschen Nachrichtenseiten nicht immer nach denselben Kriterien. Zudem gibt es regionale Unterschiede bei den Nachrichtenthemen und den verwendeten Vokabeln. Die Methode zur Generierung von synthetischen Labels konnte sich gegenüber den anderen beiden Methoden (syntethische Trainings- oder Testdaten) durchsetzten. Sie erreicht im Vergleich zu den anderen Methoden einen höhere Leistung und besitzt eine niedrigere Laufzeit bei der Klassifikation, da nur in der Trainingsphase Übersetzungen generiert werden.

Für die Evaluation der beiden Verfahren aus dem zweiten Problemszenario wurde die Leistungsverbesserung auf insgesamt elf verschiedenen Datensätzen bewertet, darunter acht deutschsprachige und drei englischsprachige Datensätze. Mit dem TDE-Verfahren konnte auf neun der elf Datensätze eine Verbesserung erzielt werden. Die Verbesserung betrug durchschnittlich 0.3% Prozentpunkte über die Datensätze. Bei hohen n kommt es beim Training der Klassifikatoren zur Überanpassung auf den Trainingsdatensatz. Das zweite Transferverfahren,

bei dem die Gewichte eines MT-Modells direkt auf das Modell eines Klassifikators übertragen wurden, führte bei acht von elf Datensätzen zu verbesserten Ergebnissen. Die Steigerung lag hier bei durchschnittlich 1, 0 Prozentpunkten.

6.2. Ausblick

Für viele Problemstellungen der NLP konnten sich in den letzten Jahren neuronale Verfahren gegen regelbasierte und statistische Verfahren durchsetzten. Dagegen werden nach Kenntnisstand des Autors bei der Textklassifikation in der Praxis heute noch überwiegend traditionelle Methoden verwendet. In dieser Arbeit konnte, wie auch schon in früheren Arbeiten, dargelegt werden, dass neuronale Verfahren die Leistung anderer State-of-the-Art TK-Verfahren erreichen und diese teilweise auch übertreffen können. Die NNs erfordern dafür nur geringfügiges *Feature-Engineering* und aufgrund von Standardarchitkturen kann der Aufwand für das Optimieren von Hyperparameter auf ein Minimum beschränkt werden. Mit der fortschreitenden Weiterentwicklung der Deep Learning Frameworks, darunter *Keras*, *Thensorflow* oder *Theano*, um nur die bekanntesten aufzuzählen, wird die Implementierung von neuronalen Systemen immer leichter. Es bleibt abzuwarten, ob die oben genannten Vorteil dazu führen werden, dass sie sich in naher Zukunft als TK-Verfahren durchsetzen.

Bei der Betrachtung des CTK-Verfahren wurde festgestellt, dass deren Leistung vor allem durch die Qualität der generierten Übersetzung und von den sprachlichen Unterschieden zwischen Quell- und Zieldomäne beschränkt sind. Jüngste Ansätze zur CTK versuchen auf den Einsatz von maschinellen Übersetzern zu verzichten. Bei der *multilingualen* Textklassifikation wird beispielsweise die Entwicklung von sprachunabhängige, TK-Modellen erforscht. Erwähnenswert in diesem Zusammenhang sind die bilingualen Word Embeddings[114], die als Merkmale für die Systeme verwendet werden können. Ein weiteres Beispiel für ein modernes CTK-Verfahren ist das Adverserial Deep Averaging Network (ADAN)[13], das in Abschnitt 3.3 vorgestellt wurde. Potentielle Themen für zukünftige Forschungsarbeiten sind die Entwicklung entsprechend neuer CTK-Architekturen und ein Vergleich der Performance von Adverserial Deep Averaging Network (ADAN) mit den hier beschriebenen Verfahren.

Der Einsatz von Lerntransfertechniken wird zurzeit in der Wissenschaft intensiv diskutiert. In der Bildverarbeitung wird schon heute exzessiv auf sie zurückgegriffen. Der Stanford Professor und leitende Wissenschaftler von Baidu, Andrew Ng, sagt, dass der Lerntransfer der Erfolgsgarant für den kommerziellen Erfolg von ML-Systemen der nächsten Genration sein wird[97]. Bis dahin muss aber noch weitere Energie in die Erforschung des Themenfeldes investiert werden. Es ist noch nicht abschließend geklärt, unter welchen Kriterien der Transfer zwischen verschiedenen neuronalen Modellen gelingt und welche Tasks miteinander „harmonieren“. In dieser Arbeit konnten gezeigt zeigen, dass das Wissen der maschinelle Übersetzung als Basis zur Lösung anderer NLP-Probleme beitragen kann. Zur Verbesserung der Technik zur Datenerweiterung sollte im Anschluss evaluiert werden, wie gut sich andere Paraphrasierungsmethoden für entsprechende Verfahren eignen. Es darf davon ausgegangen werden, dass mit einem System, das eigens für die Paraphrasierung trainiert wurde, bessere Ergebnisse erzielt werden können als bei der Paraphrasierung mit maschinellen Übersetzungssystemen.

Literatur

[1] Charu C. Aggarwal und ChengXiang Zhai. „A Survey of Text Classification Algorithms“. en. In: *Mining Text Data*. Springer, Boston, MA, 2012, S. 163–222. ISBN: 978-1-4614-3222-7 978-1-4614-3223-4. DOI: 10.1007/978-1-4614-3223-4_6. URL: https://link.springer.com/chapter/10.1007/978-1-4614-3223-4_6 (besucht am 04. 04. 2018).

[2] Dzmitry Bahdanau, Kyunghyun Cho und Yoshua Bengio. „Neural Machine Translation by Jointly Learning to Align and Translate“. In: *CoRR* abs/1409.0473 (2014). arXiv: 1409.0473. URL: http://arxiv.org/abs/1409.0473.

[3] Carmen Banea, Rada Mihalcea, Janyce Wiebe und Samer Hassan. „Multilingual Subjectivity Analysis Using Machine Translation“. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. EMNLP '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, S. 127–135. URL: http://dl.acm.org/citation.cfm?id=1613715.1613734.

[4] Colin Bannard und Chris Callison-Burch. „Paraphrasing with Bilingual Parallel Corpora“. In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. ACL '05. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, S. 597–604. DOI: 10.3115/1219840.1219914. URL: https://doi.org/10.3115/1219840.1219914 (besucht am 17. 05. 2018).

[5] Roland Becker. *Word Embeddings - Methoden zur Repräsentation von Wörtern in Algorithmen und neuronalen Netzen*. de-DE. Mai 2018. URL: https://jaai.de/word-embeddings-worteinbettung-word2vec-glove-bag-of-words-1872/ (besucht am 24. 07. 2018).

[6] Yoshua Bengio, Réjean Ducharme, Pascal Vincent und Christian Jauvin. „A Neural Probabilistic Language Model“. In: *Journal of Machine Learning Research* 3.Feb (2003), S. 1137–1155. ISSN: ISSN 1533-7928. URL: http://www.jmlr.org/papers/v3/bengio03a.html (besucht am 24. 11. 2017).

[7] Jonathan Berant und Percy Liang. „Semantic parsing via paraphrasing“. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Bd. 1. 2014, S. 1415–1425.

[8] Alessandro Bergamo und Lorenzo Torresani. „Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach“. In: *Advances in neural information processing systems*. 2010, S. 181–189.

[9] *Bing Übersetzer*. https://www.bing.com/translator. (Accessed on 09/27/2017).

[10] Ondřej Bojar u. a. „Findings of the 2017 Conference on Machine Translation (WMT17)“. In: *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, S. 169–214. URL: http://www.aclweb.org/anthology/W17-4717.

[11] Johan Bollen, Huina Mao und Xiaojun Zeng. „Twitter mood predicts the stock market“. In: *Journal of computational science* 2.1 (2011), S. 1–8.

[12] Chris Callison-Burch, Philipp Koehn und Miles Osborne. „Improved statistical machine translation using paraphrases“. In: *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, 2006, S. 17–24.

[13] Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie und Kilian Weinberger. „Adversarial Deep Averaging Networks for Cross-Lingual Sentiment Classification“. In: *ArXiv e-prints* (Juni 2016). arXiv: 1606.01614. URL: https://arxiv.org/abs/1606.01614.

[14] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau und Yoshua Bengio. „On the Properties of Neural Machine Translation: Encoder–Decoder Approaches“. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, 2014, S. 103–111. DOI: 10.3115/v1/W14-4012. URL: http://www.aclweb.org/anthology/W14-4012.

[15] Kyunghyun Cho u. a. „Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation“. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, S. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: http://www.aclweb.org/anthology/D14-1179.

[16] D. Ciregan, U. Meier und J. Schmidhuber. „Multi-column deep neural networks for image classification“. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. Juni 2012, S. 3642–3649. DOI: 10.1109/CVPR.2012.6248110.

[17] Ronan Collobert und Jason Weston. „A unified architecture for natural language processing: Deep neural networks with multitask learning“. In: *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, S. 160–167.

[18] Ronan Collobert u. a. „Natural language processing (almost) from scratch“. In: *Journal of Machine Learning Research* 12.Aug (2011), S. 2493–2537.

[19] Alexis Conneau, Holger Schwenk, Loïc Barrault und Yann Lecun. „Very Deep Convolutional Networks for Text Classification“. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, 2017, S. 1107–1116. URL: http://aclweb.org/anthology/E17-1104.

[20] Kia Dashtipour u. a. „Multilingual Sentiment Analysis: State of the Art and Independent Comparison of Techniques“. In: *Cognitive Computation* 8 (2016), S. 757–771. ISSN: 1866-9956. DOI: 10.1007/s12559-016-9415-7. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4981629/.

[21] *DeepL Übersetzer*. https://www.deepl.com/translate. (Accessed on 09/27/2017).

[22] Jia Deng u. a. „Imagenet: A large-scale hierarchical image database“. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, S. 248–255.

[23] George Doddington. „Automatic evaluation of machine translation quality using n-gram co-occurrence statistics“. In: *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 2002, S. 138–145.

[24] Richard O Duda, Peter E Hart und David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[25] Dudenredaktion Dudenredaktion. *Duden - Deutsches Universalwörterbuch - Das umfassende Bedeutungswörterbuch der deutschen Gegenwartssprache*. mannheim: Bibliographisches Institut GmbH, 2016. ISBN: 978-3-411-91171-4.

[26] Kevin Duh, Akinori Fujino und Masaaki Nagata. „Is Machine Translation Ripe for Cross-lingual Sentiment Classification?“ In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, S. 429–433. ISBN: 978-1-932432-88-6. URL: http://dl.acm.org/citation.cfm?id=2002736.2002823.

[27] Les Gasser, Kiran Lakkaraju, Sylvian Ray und Samarth Swarup. „DARPA BAA 05-29: Transfer Learning Issues and Potential Contributions“. In: (Mai 2005).

[28] Ross Girshick, Jeff Donahue, Trevor Darrell und Jitendra Malik. „Rich feature hierarchies for accurate object detection and semantic segmentation“. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, S. 580–587.

[29] Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville und Yoshua Bengio. „Maxout Networks“. In: *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*. ICML'13. Atlanta, GA, USA: JMLR.org, 2013, S. III-1319–III-1327. URL: http://dl.acm.org/citation.cfm?id=3042817.3043084.

[30] *Google Übersetzer*. https://translate.google.com/. (Accessed on 09/27/2017).

[31] A. Graves, A. Mohamed und G. Hinton. „Speech recognition with deep recurrent neural networks“. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Mai 2013, S. 6645–6649. DOI: 10.1109/ICASSP.2013.6638947.

[32] Edward Grefenstette, Phil Blunsom, Nando de Freitas und Karl Moritz Hermann. „A Deep Architecture for Semantic Parsing“. In: *Proceedings of the ACL 2014 Workshop on Semantic Parsing*. Baltimore, MD: Association for Computational Linguistics, 2014, S. 22–27. DOI: 10.3115/v1/W14-2405. URL: http://www.aclweb.org/anthology/W14-2405.

[33] Zellig S. Harris. „Distributional structure“. In: *Word* 10.2-3 (1954), S. 146–162.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren und Jian Sun. „Deep residual learning for image recognition“. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, S. 770–778.

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren und Jian Sun. „Spatial pyramid pooling in deep convolutional networks for visual recognition“. In: *european conference on computer vision*. Springer, 2014, S. 346–361.

[36] heise online heise. *Maschinelle Übersetzer: DeepL macht Google Translate Konkurrenz*. de. URL: https://www.heise.de/newsticker/meldung/Maschinelle-Uebersetzer-DeepL-macht-Google-Translate-Konkurrenz-3813882.html (besucht am 04. 04. 2018).

[37] Felix Hill, Kyunghyun Cho, Sébastien Jean und Yoshua Bengio. „The representational geometry of word meanings acquired by neural machine translation models“. In: *Machine Translation* 31.1-2 (2017), S. 3–18.

[38] Felix Hill, Kyunghyun Cho und Anna Korhonen. „Learning Distributed Representations of Sentences from Unlabelled Data“. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, 2016, S. 1367–1377. DOI: 10.18653/v1/N16-1162. URL: http://www.aclweb.org/anthology/N16-1162.

[39] Geoffrey E. Hinton, James L. McClelland und David E. Rumelhart. „Distributed representations“. In: *Parallel distributed processing: Explorations in the microstructure of cognition* 1.3 (1986), S. 77–109.

[40] Sepp Hochreiter und Jürgen Schmidhuber. „Long Short-Term Memory“. In: *Neural Computation* 9.8 (Nov. 1997), S. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[41] Andrew G. Howard. „Some Improvements on Deep Convolutional Neural Network Based Image Classification“. In: *CoRR* abs/1312.5402 (2013).

[42] Baidu Inc. *Ever wonder just how long #MachineTranslation has been evolving? It's been quite a journey to get to where we are today! For a deeper dive - here's how it was visualized by Baidu's Hua Wu at @techreview's #emtechdigital 2018. @BaiduResearch #AI #DeepLearning #MT #MachineLearningpic.twitter.com/wRuHGI4Q1g*. en. Tweet. Apr. 2018. URL: https://twitter.com/Baidu_Inc/status/984083800685731840 (besucht am 11. 04. 2018).

[43] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber und Hal Daumé III. „Deep unordered composition rivals syntactic methods for text classification“. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Bd. 1. 2015, S. 1681–1691.

[44] Jing Jiang und ChengXiang Zhai. „Instance weighting for domain adaptation in NLP“. In: *Proceedings of the 45th annual meeting of the association of computational linguistics*. 2007, S. 264–271.

[45] Thorsten Joachims. „Text categorization with Support Vector Machines: Learning with many relevant features“. en. In: *Machine Learning: ECML-98*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Apr. 1998, S. 137–142. ISBN: 978-3-540-64417-0 978-3-540-69781-7. DOI: 10.1007/BFb0026683. URL: https://link.springer.com/chapter/10.1007/BFb0026683 (besucht am 04. 04. 2018).

[46] Rie Johnson und Tong Zhang. „Effective Use of Word Order for Text Categorization with Convolutional Neural Networks“. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, 2015, S. 103–112. DOI: 10.3115/v1/N15-1011. URL: http://www.aclweb.org/anthology/N15-1011.

[47] Nal Kalchbrenner, Edward Grefenstette und Phil Blunsom. „A Convolutional Neural Network for Modelling Sentences“. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, 2014, S. 655–665. DOI: 10.3115/v1/P14-1062. URL: http://www.aclweb.org/anthology/P14-1062.

[48] Yoon Kim. „Convolutional Neural Networks for Sentence Classification“. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, S. 1746–1751. DOI: 10.3115/v1/D14-1181. URL: http://www.aclweb.org/anthology/D14-1181.

[49] Diederik P. Kingma und Jimmy Ba. „Adam: A Method for Stochastic Optimization“. In: *CoRR* abs/1412.6980 (2014). arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980.

[50] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart und Alexander Rush. „OpenNMT: Open-Source Toolkit for Neural Machine Translation“. In: *Proceedings of ACL 2017, System Demonstrations*. Vancouver, Canada: Association for Computational Linguistics, 2017, S. 67–72. URL: http://www.aclweb.org/anthology/P17-4012.

[51] Alex Krizhevsky, Ilya Sutskever und Geoffrey E. Hinton. „Imagenet classification with deep convolutional neural networks“. In: *Advances in neural information processing systems*. 2012, S. 1097–1105.

[52] Alison Kroulek. *Is Google Translate Really As Good As a Human Now?* en-GB. Okt. 2016. URL: http://www.k-international.com/blog/google_as_good_as_humans/ (besucht am 13. 02. 2018).

[53] Siwei Lai, Liheng Xu, Kang Liu und Jun Zhao. „Recurrent Convolutional Neural Networks for Text Classification.“ In: *AAAI*. Bd. 333. 2015, S. 2267–2273.

[54] Ken Lang. „NewsWeeder: Learning to Filter Netnews“. In: *Machine Learning Proceedings 1995*. Hrsg. von Armand Prieditis und Stuart Russell. San Francisco (CA): Morgan Kaufmann, 1995, S. 331–339. ISBN: 978-1-55860-377-6. DOI: 10.1016/B978-1-55860-377-6.50048-7. URL: https://www.sciencedirect.com/science/article/pii/B9781558603776500487 (besucht am 04. 04. 2018).

[55] Quoc Le und Tomas Mikolov. „Distributed representations of sentences and documents“. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014, S. 1188–1196.

[56] Karel Lenc und Andrea Vedaldi. „Understanding image representations by measuring their equivariance and equivalence“. In: *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, S. 991–999.

[57] David D. Lewis und Marc Ringuette. „A comparison of two learning algorithms for text categorization“. In: *Third annual symposium on document analysis and information retrieval*. Bd. 33. 1994, S. 81–93.

[58] Jiasen Lu, Caiming Xiong, Devi Parikh und Richard Socher. „Knowing When to Look: Adaptive Attention via a Visual Sentinel for Image Captioning“. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. 2017, S. 3242–3250. DOI: 10.1109/CVPR.2017.345. URL: https://doi.org/10.1109/CVPR.2017.345.

[59] Minh-Thang Luong, Eugene Brevdo und Rui Zhao. „Neural Machine Translation (seq2seq) Tutorial“. In: *https://github.com/tensorflow/nmt* (2017).

[60] Andrew L. Maas u. a. „Learning Word Vectors for Sentiment Analysis“. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, Juni 2011, S. 142–150. URL: http://www.aclweb.org/anthology/P11-1015.

[61] Nitin Madnani und Bonnie J. Dorr. „Generating Phrasal and Sentential Paraphrases: A Survey of Data-driven Methods“. In: *Comput. Linguist.* 36.3 (Sep. 2010), S. 341–387. ISSN: 0891-2017. DOI: 10.1162/coli_a_00002. URL: http://dx.doi.org/10.1162/coli_a_00002 (besucht am 11. 05. 2018).

[62] Jonathan Mallinson, Rico Sennrich und Mirella Lapata. „Paraphrasing Revisited with Neural Machine Translation“. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, S. 881–893. URL: http://www.aclweb.org/anthology/E17-1083 (besucht am 11. 05. 2018).

[63] Andrew McCallum, Kamal Nigam u. a. „A comparison of event models for naive bayes text classification“. In: *AAAI-98 workshop on learning for text categorization*. Bd. 752. Madison, WI, 1998, S. 41–48. URL: http://staff.icar.cnr.it/manco/Teaching/2005/datamining/articoli/multinomial-aaaiws98.pdf (besucht am 22. 09. 2017).

[64] Bryan McCann, James Bradbury, Caiming Xiong und Richard Socher. „Learned in Translation: Contextualized Word Vectors“. In: *arXiv:1708.00107 [cs]* (Juli 2017). arXiv: 1708.00107. URL: http://arxiv.org/abs/1708.00107.

[65] Warren S. McCulloch und Walter Pitts. „A logical calculus of the ideas immanent in nervous activity“. en. In: *The bulletin of mathematical biophysics* 5.4 (Dez. 1943), S. 115–133. ISSN: 0007-4985, 1522-9602. DOI: 10.1007/BF02478259. URL: https://link.springer.com/article/10.1007/BF02478259 (besucht am 19. 10. 2017).

[66] *Microsoft Research: KI übersetzt so gut wie ein Mensch - Golem.de*. de-DE. URL: https://www.golem.de/news/microsoft-research-computersystem-uebersetzt-praktisch-simultan-1803-133343.html (besucht am 23. 07. 2018).

[67] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado und Jeff Dean. „Distributed representations of words and phrases and their compositionality“. In: *Advances in neural information processing systems*. 2013, S. 3111–3119.

[68] Marvin Minsky, Seymour A. Papert und Léon Bottou. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[69] Andriy Mnih und Geoffrey E. Hinton. „A scalable hierarchical distributed language model“. In: *Advances in neural information processing systems*. 2009, S. 1081–1088.

[70] Makoto Nagao. „A framework of a mechanical translation between Japanese and English by analogy principle“. In: *Artificial and human intelligence* (1984), S. 351–354.

[71] Donald A. Norman und La Jolla. Center for Human Information Processing. California Univ. *Memory, Knowledge, and the Answering of Questions [microform] / Donald A. Norman*. English. Distributed by ERIC Clearinghouse [Washington, D.C.], 1972, 57 p. URL: http://www.eric.ed.gov/contentdelivery/servlet/ERICServlet?accno=ED097003.

[72] Jakub Nowak, Ahmet Taspinar und Rafał Scherer. „LSTM Recurrent Neural Networks for Short Text and Sentiment Classification“. en. In: *Artificial Intelligence and Soft Computing*. Lecture Notes in Computer Science. Springer, Cham, Juni 2017, S. 553–562. ISBN: 978-3-319-59059-2 978-3-319-59060-8. DOI: 10.1007/978-3-319-59060-8_50. URL: https://link.springer.com/chapter/10.1007/978-3-319-59060-8_50 (besucht am 25. 07. 2017).

[73] Sinno Jialin Pan und Qiang Yang. „A survey on transfer learning“. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2010), S. 1345–1359.

[74] Kishore Papineni, Salim Roukos, Todd Ward und Wei-Jing Zhu. „BLEU: a method for automatic evaluation of machine translation“. In: *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, S. 311–318.

[75] Xingchao Peng, Baochen Sun, Karim Ali und Kate Saenko. „Exploring invariances in deep convolutional neural networks using synthetic images“. In: *CoRR, abs/1412.7122* 2.4 (2014).

[76] Jeffrey Pennington, Richard Socher und Christopher D. Manning. „Glove: Global vectors for word representation.“ In: *EMNLP*. Bd. 14. 2014, S. 1532–1543. URL: http://llcao.net/cu-deeplearning15/presentation/nn-pres.pdf.

[77] Ngoc-Quan Pham u. a. „The Karlsruhe Institute of Technology Systems for the News Translation Task in WMT 2017“. In: Jan. 2017, S. 366–373. DOI: 10.18653/v1/W17-4736.

[78] John R. Pierce und John B. Carroll. *Language and Machines: Computers in Translation and Linguistics*. Washington, DC, USA: National Academy of Sciences/National Research Council, 1966.

[79] Yuankai Qi u. a. „Hedged deep tracking“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, S. 4303–4311.

[80] Beyond Regression. „New Tools for Prediction and Analysis in the Behavioral Sciences“. In: *Werbos, PJ https://books. google. se/books* (1974).

[81] Frank Rosenblatt. „The perceptron: A probabilistic model for information storage and organization in the brain.“ In: *Psychological review* 65.6 (1958), S. 386.

[82] Sara Rosenthal, Noura Farra und Preslav Nakov. „SemEval-2017 Task 4: Sentiment Analysis in Twitter“. In: *Proceedings of the 11th International Workshop on Semantic Evaluation*. SemEval '17. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017.

[83] Sebastian Ruder. „An overview of gradient descent optimization algorithms.“ In: *CoRR* abs/1609.04747 (2016). URL: http://dblp.uni-trier.de/db/journals/corr/corr1609.html#Ruder16.

[84] David E. Rumelhart, Geoffrey E. Hinton und Ronald J. Williams. „Learning representations by back-propagating errors“. In: *nature* 323.6088 (1986), S. 533.

[85] Mehran Sahami, Susan Dumais, David Heckerman und Eric Horvitz. „A Bayesian approach to filtering junk e-mail“. In: *Learning for Text Categorization: Papers from the 1998 workshop*. Bd. 62. 1998, S. 98–105.

[86] Rico Sennrich, Barry Haddow und Alexandra Birch. „Neural Machine Translation of Rare Words with Subword Units“. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, 2016, S. 1715–1725. DOI: 10.18653/v1/P16-1162. URL: http://www.aclweb.org/anthology/P16-1162.

[87] Catarina Silva und Bernardete Ribeiro. „Background on Text Classification“. In: *Inductive Inference for Large Scale Text Classification: Kernel Approaches and Techniques*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 3–29. ISBN: 978-3-642-04533-2. DOI: 10.1007/978-3-642-04533-2_1. URL: https://doi.org/10.1007/978-3-642-04533-2_1.

[88] Karen Simonyan und Andrew Zisserman. „Very Deep Convolutional Networks for Large-Scale Image Recognition“. In: *arXiv:1409.1556 [cs]* (Sep. 2014). arXiv: 1409.1556. URL: http://arxiv.org/abs/1409.1556 (besucht am 22. 03. 2018).

[89] Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning und Andrew Y. Ng. „Grounded compositional semantics for finding and describing images with sentences“. In: *Transactions of the Association for Computational Linguistics* 2 (2014), S. 207–218.

[90] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever und Ruslan Salakhutdinov. „Dropout: a simple way to prevent neural networks from overfitting.“ In: *Journal of Machine Learning Research* 15.1 (2014), S. 1929–1958. URL: http://www.jmlr.org/papers/volume15/srivastava14a.old/source/srivastava14a.pdf.

[91] Antonino Simone Di Stefano. *Lexical Substitution with Word Embeddings: A General Approach to Improve Short Text Classification*. Karlsruhe, [2017].

[92] *Supervised Learning for Text Classification | T/DG Blog - Digital Thoughts*. URL: http://blog.thedigitalgroup.com/supervised-learning-for-text-classification (besucht am 26. 07. 2018).

[93] Ilya Sutskever, Oriol Vinyals und Quoc V. Le. „Sequence to sequence learning with neural networks“. In: *Advances in neural information processing systems*. 2014, S. 3104–3112.

[94] C. Szegedy u. a. „Going deeper with convolutions“. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Bd. 00. Juni 2015, S. 1–9. DOI: 10.1109/CVPR.2015.7298594. URL: doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298594.

[95] Songbo Tan. „Neighbor-weighted k-nearest neighbor for unbalanced text corpus“. In: *Expert Systems with Applications* 28.4 (2005), S. 667–671.

[96] Duyu Tang, Bing Qin und Ting Liu. „Document Modeling with Gated Recurrent Neural Network for Sentiment Classification.“ In: *EMNLP*. 2015, S. 1422–1432.

[97] *Transfer Learning - Machine Learning's Next Frontier*. März 2017. URL: u=http://ruder.io/transfer-learning/ (besucht am 30. 03. 2018).

[98] Joseph Turian, Lev Ratinov und Yoshua Bengio. „Word representations: a simple and general method for semi-supervised learning“. In: *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, S. 384–394.

[99] Alexander H. Waibel, Toshiyuki Hanazawa, Geoffrey E. Hinton, Kiyohiro Shikano und Kevin J. Lang. „Phoneme recognition using time-delay neural networks“. In: *IEEE Trans. Acoustics, Speech, and Signal Processing* 37.3 (1989), S. 328–339. DOI: 10.1109/29.21701. URL: https://doi.org/10.1109/29.21701.

[100] Xiaojun Wan. „Co-training for Cross-lingual Sentiment Classification“. In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*. ACL '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, S. 235–243. ISBN: 978-1-932432-45-9. URL: http://dl.acm.org/citation.cfm?id=1687878.1687913.

[101] Jason Wang und Luis Perez. *The effectiveness of data augmentation in image classification using deep learning*. Techn. Ber. Technical report, 2017.

[102] *Warren Weaver memorandum, July 1949*. Okt. 2006. URL: https://web.archive.org/web/20061005232830/http://ourworld.compuserve.com/homepages/WJHutchins/Weaver49.htm (besucht am 20. 07. 2018).

[103] Bin Wei und Christopher Pal. „Cross Lingual Adaptation: An Experiment on Sentiment Classifications“. In: *Proceedings of the ACL 2010 Conference Short Papers*. ACLShort '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, S. 258–262. URL: http://dl.acm.org/citation.cfm?id=1858842.1858890.

[104] Ren Wu, Shengen Yan, Yi Shan, Qingqing Dang und Gang Sun. „Deep Image: Scaling up Image Recognition“. In: *CoRR* abs/1501.02876 (2015).

[105] Yonghui Wu u. a. „Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation“. In: *CoRR* abs/1609.08144 (2016). URL: http://arxiv.org/abs/1609.08144.

[106] Caiming Xiong, Stephen Merity und Richard Socher. „Dynamic memory networks for visual and textual question answering“. In: *International Conference on Machine Learning*. 2016, S. 2397–2406.

[107] Zichao Yang u. a. „Hierarchical Attention Networks for Document Classification.“ In: *HLT-NAACL*. 2016, S. 1480–1489. URL: http://www.aclweb.org/anthology/N16-1174 (besucht am 27. 07. 2017).

[108] Wenpeng Yin, Katharina Kann, Mo Yu und Hinrich Schütze. „Comparative Study of CNN and RNN for Natural Language Processing“. In: *CoRR* abs/1702.01923 (2017).

[109] C. Zhang und Z. Zhang. „Improving multiview face detection with multi-task deep convolutional neural networks“. In: *IEEE Winter Conference on Applications of Computer Vision*. März 2014, S. 1036–1041. DOI: 10.1109/WACV.2014.6835990.

[110] Xiang Zhang und Yann LeCun. „Text Understanding from Scratch“. In: *arXiv:1502.01710 [cs]* (Feb. 2015). arXiv: 1502.01710. URL: http://arxiv.org/abs/1502.01710 (besucht am 06. 12. 2017).

[111] Xiang Zhang, Junbo Zhao und Yann LeCun. „Character-level convolutional networks for text classification“. In: *Advances in neural information processing systems*. 2015, S. 649–657. URL: http://papers.nips.cc/paper/5782-character-level-convolutional-networks-fo (besucht am 27. 07. 2017).

[112] Ye Zhang und Byron C. Wallace. „A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification.“ In: *CoRR* abs/1510.03820 (2015). URL: http://dblp.uni-trier.de/db/journals/corr/corr1510.html#ZhangW15b.

[113] Chunting Zhou, Chonglin Sun, Zhiyuan Liu und Francis Lau. „A C-LSTM neural network for text classification“. In: *arXiv preprint arXiv:1511.08630* (2015).

[114] Will Y. Zou, Richard Socher, Daniel M. Cer und Christopher D. Manning. „Bilingual Word Embeddings for Phrase-Based Machine Translation.“ In: *EMNLP*. 2013, S. 1393–1398. URL: http://jan.stanford.edu/pubs/emnlp2013_ZouSocherCerManning.pdf.

Abkürzungsverzeichnis

CNN Convolutional Neural Network. 12, 25, 29, 45

CTK Crosslinguale Textklassifikation. v, 8, 14, 21–24, 45

DAP Domänenadaptionsproblem. 23, 45

GPGPU General Purpose Computation on Graphics Processing Unit. 45

KNN Künstliches Neuronales Netz. 45

LSTM Long Short-Term Memory. 32, 45

ML Maschinelles Lernen. 2, 23, 45

MT Maschinelle Übersetzung. v, 3, 4, 17, 21–23, 26–30, 32, 33, 45

NB Naiver Bayes-Klassifikator. 24, 45

NLP Natürliche Sprachverarbeitung. 1, 3, 4, 16, 17, 20, 23, 25, 32, 45

DAN Deep Averaging Network. 11, 12, 45

NMT Neuronale Maschinelle Übersetzung. v, 4, 17, 26, 30, 31, 45

NN Neuronales Netz. 4, 8, 9, 15, 45

OOV Out-of-Vocabulary. 45

RNN Recurrent Neural Network. 10, 12, 29, 31, 32, 45

SA Sentimentanlyse. 45

Seq2Seq Sequence-to-Sequence. 29–33, 45

SMT Statistische Maschinelle Übersetzung. 45

STK Neuronale Textklassifikation. 45

STS Semantic Textual Similarity. 25, 26, 30, 45

SVM Support Vektor Maschine. 24, 45

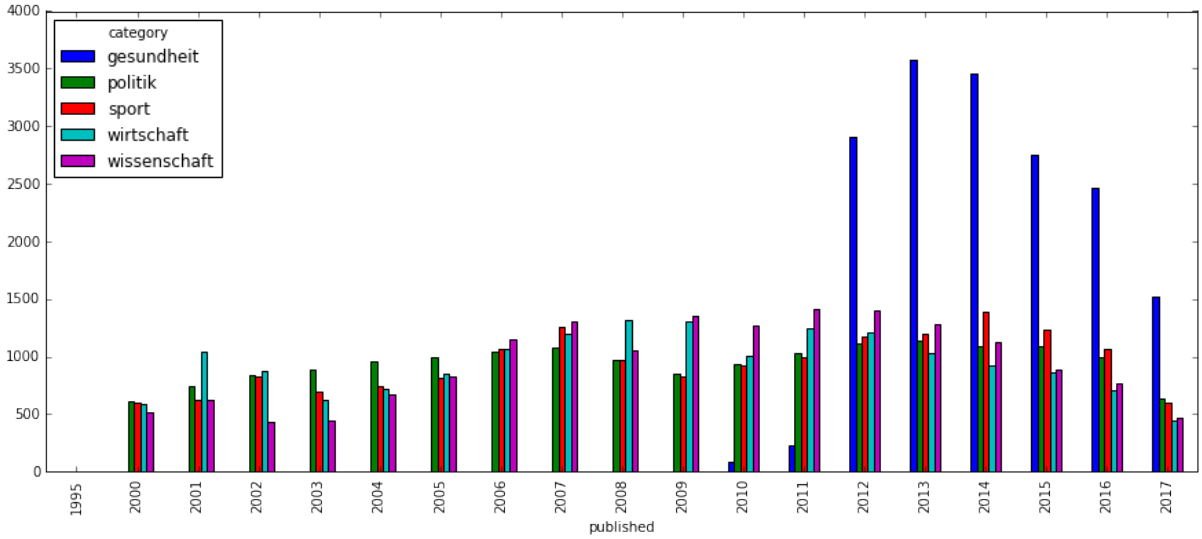
TDE-Verfahren Trainingsdatenerweiterungsverfahren. 24, 25, 27, 45

TK Textklassifikation. 2–4, 8, 10, 11, 15, 17, 20, 23–26, 28, 45

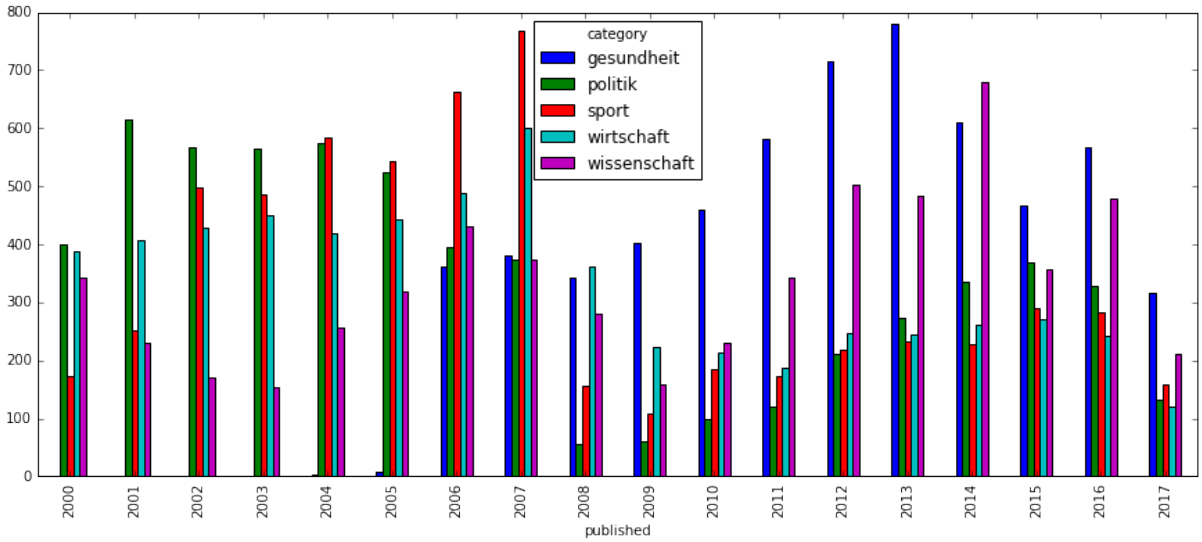
WMT Worshop on Machine Translation. 45

A. Anhang

A.1. Statistik zum Nachrichtenressort-Datensatz



(a) NEWS-DE



(b) NEWS-EN

Abbildung A.1.: Verteilung der Trainingsbeispiele über die Zeit in den jeweiligen Kategorien.