

Generierung ansprechender Arbeitszeugnisse

Wirtschaftsprojekt HS2019

Barmettler Reto / Gröger Fabian

Betreuer: Pfäffli Daniel

Hochschule Luzern - Departement Informatik
20. Dezember 2019

Wirtschaftsprojekt an der Hochschule Luzern – Informatik

Titel: Generierung ansprechender Arbeitszeugnisse

Studentin / Student: Barmettler Reto

Studentin / Student: Gröger Fabian

Studiengang: BSc Informatik

Jahr: 2019

Betreuungsperson: Pfäffli Daniel

Expertin / Experte: Dr. Tim vor der Brück

Auftraggeberin / Auftraggeber: confer! AG

Codierung / Klassifizierung der Arbeit: Innovationsprojekte (Projekte mit Erkenntnisgewinn, Forschungsprojekte)

- A: Einsicht (Normalfall)**
- B: Rücksprache** (Dauer: Jahr / Jahre)
- C: Sperre** (Dauer: Jahr / Jahre)

Eidesstattliche Erklärung

Ich erkläre hiermit, dass ich/wir die vorliegende Arbeit selbstständig und ohne unerlaubte fremde Hilfe angefertigt haben, alle verwendeten Quellen, Literatur und andere Hilfsmittel angegeben haben, wörtlich oder inhaltlich entnommene Stellen als solche kenntlich gemacht haben, das Vertraulichkeitsinteresse des Auftraggebers wahren und die Urheberrechtsbestimmungen der Fachhochschule Zentralschweiz (siehe Markblatt «Studentische Arbeiten» auf MyCampus) respektieren werden.

Ort / Datum, Unterschrift: _____

Ort / Datum, Unterschrift: _____

**Ausschliesslich bei Abgabe in gedruckter Form:
Eingangsvistum durch das Sekretariat auszufüllen**

Rotkreuz, den _____

Visum: _____

Hinweis: Das Wirtschaftsprojekt wurde von keinem Dozierenden nachbearbeitet. Veröffentlichungen (auch auszugsweise) sind ohne das Einverständnis der Studiengangleitung der Hochschule Luzern – Informatik nicht erlaubt.

Copyright © 2019 Hochschule Luzern - Informatik

Alle Rechte vorbehalten. Kein Teil dieser Arbeit darf ohne die schriftliche Genehmigung der Studiengangleitung der Hochschule Luzern - Informatik in irgendeiner Form reproduziert oder in eine von Maschinen verwendete Sprache übertragen werden.

Zusammenfassung

Das vorliegende Wirtschaftsprojekt überprüft den Einsatz von Neural Style Transfer für Sätze aus der deutschen Sprache. Anhand eines Datensatzes, bestehend aus Sätzen für Erwachsene und Kinder, wurde überprüft, ob ein weniger komplexer Satz in einen komplexeren Satz umgewandelt werden kann. Die Erkenntnisse sollen dem Auftraggeber als mögliche Grundlage für den Einsatz von Neural Style Transfer für die Verschönerung von Zwischen- und Arbeitszeugnissen dienen.

Inhaltsverzeichnis

1 Einleitung	1
1.1 Aufgabenstellung und Zielsetzung	1
1.1.1 Ausgangslage	1
1.1.2 Aufgabenstellung	1
1.1.3 Zielsetzung	1
2 Stand der Technik	3
2.1 Technologische Grundlagen	3
2.1.1 Natural Language Generation	3
2.1.2 Natural Language Processing	3
2.1.3 Neural Style Transfer	4
2.1.4 Künstliche neuronale Netze	5
2.2 Technische Konzepte	8
2.2.1 Markov Chains	8
2.2.2 Recurrent Neural Networks	9
2.2.3 Long Short-Term Memory	10
2.2.4 Autoencoder	13
2.2.5 Transformer	14
2.2.6 Generative Adversarial Networks	15
2.2.7 Aktivierungsfunktionen	16
2.2.8 N-Gramm	19
2.2.9 Metriken	19
2.3 Stand im Bezug auf eigenes Projekt	22
3 Ideen und Konzepte	23
3.1 Problemdefinition	23
3.2 Abschliessende Problemdefinition	24
3.3 Verwendeter Datensatz	24
3.4 Momentan verfügbare Forschung	25
3.4.1 Modelle	26
3.4.2 Classifier	27
3.4.3 BaseModel	28
3.4.4 ControlGen	29
3.4.5 CrossAlign	30

3.5	Training der Modelle	32
3.5.1	Verwendete Codebasis	32
3.5.2	Verwendete Trainingsumgebung	34
3.5.3	Trainingsplanung	34
4	Methode	35
4.1	Vorgehensmodell	35
4.1.1	Meilensteinplanung	36
4.2	Problemstellung	36
4.3	Lösungsansatz	38
4.4	Evaluierung	38
4.5	Testing	39
4.6	Aufbau Projekt	39
4.6.1	wipro-doc	40
4.6.2	wipro-data	40
4.6.3	wipro-source	40
4.6.4	wipro-logs	41
5	Realisierung	42
5.1	Aufbau Datensatz	42
5.1.1	Reinigung des Datensatz	42
5.1.2	Statistische Analyse des Datensatz	43
5.2	Training der Modelle	48
5.2.1	Standard Hyperparameter	48
5.2.2	Standard Hyperparameter mit gewichteter Verlustfunktion	52
5.2.3	ControlGen mit grösseren Dimensionen	55
5.3	Prototyp	57
6	Evaluation und Validation	59
6.1	Vorgehen der Evaluierung	59
6.2	Verteilung des Evaluierungs Datensatz	60
6.3	Evaluierung der Modelle	60
6.3.1	CrossAlign gewichtete Verlustfunktion	60
6.3.2	ControlGen mit grösseren Dimensionen	62
6.3.3	Evaluierung der Ausgabesätze	63
6.4	Vergleich mit Anforderungen	64
7	Fazit	65
7.1	Projekt Fazit	65
7.2	Empfehlung im Bezug auf Neural Style Transfer	66
7.3	Empfehlung im Bezug auf den Zeugnismanager	67
A	Meilensteinberichte	III
B	Recherche	VII

C Arbeitsjournal	X
D Anleitung Installation Prototyp	XII
E Aufgabenstellung Wirtschaftsprojekt	XIII
F Beispiel Arbeitszeugnis Zeugnismanager	XVIII

1. Einleitung

In diesem Kapitel wird das Projekt eingeführt, so dass klar wird, was mit der Arbeit erreicht werden soll und aus welchem Grund diese durchgeführt wird.

1.1. Aufgabenstellung und Zielsetzung

In dieser Sektion wird die Aufgabenstellung sowie die Zielsetzung des Projektes genauer beschrieben. Ziel ist es, aufzuzeigen, was genau mit dem Projekt erreicht werden möchte und aus welcher Situation dieses Projekt entstanden ist. Die ausführliche Aufgabenstellung, welche bei der Transferstelle der Hochschule Luzern eingereicht wurde, ist im Anhang Aufgabenstellung Wirtschaftsprojekt (E) zu finden.

1.1.1. Ausgangslage

Mit dem Zeugnismanager der confer! AG können Arbeits- und Zwischenzeugnisse in vier Sprachen erstellt werden. Die Zeugnisse werden durch einfaches anwählen von Bausteinen zusammengestellt. Die so entstehenden Zeugnisse sind «holprig» zu lesen. Ein Beispiel ist der Einsatz der Anrede (Frau Meier) und des Personalpronomens (Sie) an geeigneter Stelle. Solche und ähnliche Problemstellungen werden im Moment mit einfachen Heuristiken gelöst. Weitere Probleme ergeben sich z. B. durch die unterschiedlichen Grammatiken in den jeweiligen Sprachen. So müssen viele Spezialfälle behandelt werden.

1.1.2. Aufgabenstellung

Das Ziel ist es, den Zeugnismanager dabei zu unterstützen, sprachlich korrekt ausformulierte Zeugnisse mit einem guten Lesefluss zu erstellen. Es soll eine Komponente mit Prototypcharakter erarbeitet werden. Der zu erarbeitende Prototyp kann sich auf eine Sprache, Zeitform und ggf. sogar Geschlecht einschränken, sollte aber entsprechend erweiterbar sein.

1.1.3. Zielsetzung

Wie in der Aufgabenstellung im Anhang Aufgabenstellung Wirtschaftsprojekt (E) beschrieben wird, ist das das Projekt ausserhalb des Tagesgeschäft angesiedelt und Teil von Forschung und Entwicklung. Daher wird das vorliegende Projekt nach einem explorativen Ansatz durchgeführt. Aus verschiedenen Lösungsansätzen soll ein Ansatz überprüft werden und eine Empfehlung für den Einsatz beim Arbeitgeber abgegeben werden. Dadurch wird

die Zielsetzung und Formulierung der genauen Problemdefinition im Verlauf der Arbeit vorgenommen. Da es sich um ein exploratives Innovations- / Forschungsprojekt ist die Erreichung der Aufgabenstellung und Zielsetzung zu Beginn schwer einschätzbar.

2. Stand der Technik

In diesem Kapitel werden Begriffe und Konzepte eingeführt und erklärt, welche durch die Arbeit verwendet werden. Somit kann diese Sektion als ein Nachschlagewerk genutzt werden um allfällige Lücken zu schliessen und Zusammenhänge zwischen den einzelnen Konzepten besser zu verstehen.

2.1. Technologische Grundlagen

In diesem Abschnitt werden die wichtigsten technologischen Grundlagen für das Projekt behandelt. Diese Grundlagen beinhalten vor allem Begriffdefinitionen, welche in den weiteren Kapiteln immer wieder verwendet werden. Die meisten dieser Grundlagen sind Überbegriffe, welche vorausgesetzt werden um die detaillierten Konzepte in Technische Konzepte (2.2) zu verstehen.

2.1.1. Natural Language Generation

Natural Language Generation (NLG) („Natural-language generation“, 2019) (*dt. natürliche Textgenerierung*), ist ein Unterbereich von Artificial Intelligence, welcher sich auf die automatisierte Generierung von geschriebenem oder gesprochenem Text fokussiert.

Das Ziel von Natural Language Generation (NLG) ist es sequenzielle Daten zu generieren, welche nicht von menschlichen Sequenzen unterschieden werden können. Diese Sätze sollen aussagekräftig und ausserdem grammatisch korrekt sein. Dies geschieht in einem Bruchteil der Zeit welche ein Mensch dafür bräuchte.

Etabliert hat sich die Textgenerierung vor allem im Bereich des Journalismus, in denen Nachrichten auf Daten basieren. So werden heutzutage zum Beispiel Nachrichtentexte für Wetter-, Sport und Finanzberichte zu einem grossen Teil von NLG-Systemen generiert.

2.1.2. Natural Language Processing

Natural Language Processing (NLP) („Natural language processing“, 2019) (*dt. natürliche Textverarbeitung*) ist ein Unterbereich von Artificial Intelligence, welcher sich auf die Verarbeitung von natürlicher Sprache fokussiert. Ziel ist es, eine direkte Kommunikation auf Grund natürlicher Sprache, zwischen dem Menschen und dem Computer herzustellen. NLP wird z.B. in Sprachassistenten wie Siri oder Alexa verwendet.

2.1.3. Neural Style Transfer

Der Bereich von Neural Style Transfer (NST) ist eine junge, erst 2015, von Gatys et al. (Gatys, Ecker und Bethge, 2015) eingeführte Technik, welche sich auf die Manipulation von Bildern bezieht um das Aussehen oder den visuellen Stil eines anderen Bildes zu übernehmen. NST verwenden sehr tiefe Netze, um die Bildtransformation durchzuführen. Häufige Anwendungen für NST sind die Erstellung von künstlichen Kunstwerken aus Fotos, zum Beispiel durch die Übertragung des Aussehens berühmter Gemälde auf Fotos. NST findet die Anwendung vor allem in verschiedenen mobilen Apps.



Abbildung 2.1.: Neural Style Transfer für artistische Gemälde

Infolge des Erfolgs der Technik für artistische Gemälde wurde diese über die Jahre weiter verfolgt. Dabei wird ebenfalls versucht die Technik auf andere Bereiche anzuwenden. So gibt es eine Bewegung (Fuzhenxin, 2019), die versucht NST auch für Texte zu nutzen.

Dabei werden in der Forschung momentan verschiedene NST untersucht. Beliebt sind dabei beispielsweise der Transfer von «positiv» zu «negativ», «informell» zu «formell» oder auch «aggressiv» zu «passiv». Dabei wurden in der untersuchten Literatur vor allem die Transfer «positiv» zu «negativ» und «informell» zu «formell» verwendet. In der Tabelle 2.1 sind Beispiele solcher Transfer mit dem Domain Adaptive Style Transfer (DAST) Modell, Li et al., 2019, aufgelistet.

Positiv → Negativ	
Eingabe	Ausgabe
	the service was great , food delicious , and the value impeccable . the service was horrible , food bland , and the value lousy .
Negativ → Positiv	
Eingabe	Ausgabe
	the service was horrible , food bland , and the value lousy . and the pizza was tasty , juicy , and definitely quite amazing .

Tabelle 2.1.: Beispiele eines NST mit Stil «positiv» und «negativ» mit dem DAST Modell

Bei einem NST wird versucht für einen Input $x = x_{content} + x_{style}$ eine Repräsentation $\tilde{x} = \tilde{x}_{content} + \tilde{x}_{style}$ für den Zielstil Y_{style} zu finden. Die Transformation soll dabei Werte für

$\tilde{x}_{content}$ und \tilde{x}_{style} finden so, dass

$$\tilde{x}_{content} = x_{content} \quad \text{und} \quad \tilde{x}_{style} = Y_{style} \quad (2.1)$$

Damit ist das Finden von \tilde{x} ein Minimierungsproblem der Distanzen.

$$\arg \min \tilde{x} = |\tilde{x}_{content} - x_{content}| + |\tilde{x}_{style} - Y_{style}| \quad (2.2)$$

Um nun diese Technik anzuwenden, muss entsprechend einen Messung für Inhalt (*content*) und Stil (*style*) definiert werden.

2.1.4. Künstliche neuronale Netze

Künstliche neuronale Netze sind spezielle Typen von Machine Learning Modellen, die von der Funktionsweise des menschlichen Gehirns inspiriert sind. Künstliche neuronale Netze sind von biologischen neuronalen Netzen inspiriert, unterscheiden sich jedoch in vielerlei Hinsicht. Der Vergleich wird im Bild Unterschied zwischen einem ANN und einem NN (source) (2.2) anschaulich dargestellt. Ein Künstliches neuronales Netz (KNN) besteht aus Berechnungseinheiten, den Neuronen, sowie Verbindungselementen, den Synapsen, welche Neuronen miteinander verbinden und ein Gewicht enthalten.

Wenn ein Neuron eine Zahl als Eingabe erhält, wird eine Art von Berechnung durchgeführt (z.B. eine Sigmoid Aktivierungsfunktion) und anschliessend wird das Ergebnis mit dem Gewicht der Synapse zum nächsten Neuron multipliziert, wenn es sich «aktiviert».

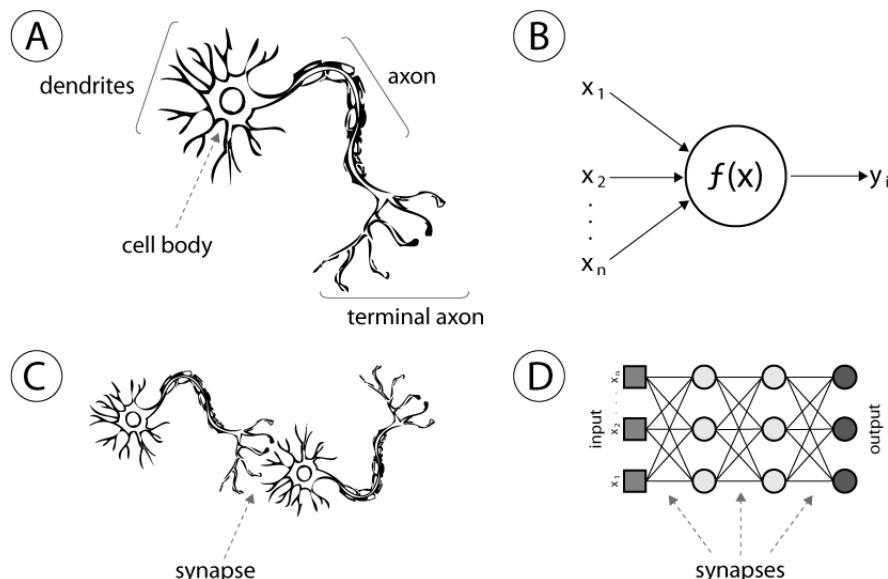


Abbildung 2.2.: Unterschied zwischen einem ANN und einem NN (source)

Sie bieten eine alternative Möglichkeit zu den linearen Modellen der Datenverarbeitung. Die Neuronalen Netze modellieren eine nicht lineare Beziehung zwischen Ein- und Ausgängen.

Somit kann sich das Modell den Daten entsprechend anpassen, egal wie die Daten verteilt sind. Es kann jede versteckte, mathematische Funktion zwischen den Daten und dem Ergebnis erkennen und erlernen. Ein Problem der neuronalen Netze ist die enorme Menge an Rechenressourcen und Daten, die benötigt werden, um gut zu funktionieren.

Deep Neural Networks ist eine weitere Form von neuronalen Netzen. Der einzige Unterschied besteht darin, dass *deep neural network* mehrere hidden Layers haben und somit komplexere mathematische Zusammenhänge lernen können. Diese Netzwerke kommen vor allem in der Bilderkennung zum Einsatz. Ein gutes Beispiel des Einsatzes eines *deep neural network* ist, die Erkennung von verschiedenen Gesichtern mittels einer Verkehrskamera.

Einer der grössten Nachteile eines Neuronalen Netzes ist, dass es nicht weiss, welche Ereignisse vorher passiert sind. Das heisst, es hat keine Erinnerungen an vergangene Handlungen. Im Bereich Natural Language Processing (NLP) und NLG ist dieser Nachteil sehr bedeutsam, denn ohne sich an das vorher Geschehene zu erinnern, kann man keine Voraussagen über zukünftige Ereignisse treffen. Aus diesem Grund wurden spezielle neuronale Netze entwickelt, die eine Art von Gedächtnis haben, wie zum Beispiel Recurrent neural network (RNN) 2.2.2 oder LSTMs 2.2.3, welche im nächsten Abschnitt genauer beschrieben werden.

Prezeptron

Künstliche neuronale Netze sind kein neues Konzept, anfänglich hießen diese «Prezeptronen» und wurden bereits um 1960 entwickelt. Diese Prezeptronen bestanden aus sogenannten McCulloch-Pitts-Neuronen. Diese wurden stetig weiterentwickelt und es entstanden gewichtete, sowie mehrschichtige Prezeptronen, aus denen die heutigen künstlichen neuronalen Netze entstanden.

Aufbau

Neuronale Netze haben immer eine ähnliche Grundstruktur, sie bestehen aus sogenannten Schichten (eng. Layers). Diese Schichten sind einfache Gruppen von Neuronen, welche mittels Synapsen mit einer anderen Gruppe von Neuronen verbunden sind. Die grundsätzliche Struktur eines Artificial Neuronal Net (ANN) beinhaltet eine Eingabeschicht (eng. input layer), mehrere versteckte Schichten (eng. hidden layers) und eine Ausgabeschicht (eng. output layer). In der Eingabeschicht werden die Daten dem neuronalen Netz zur Verarbeitung gegeben. Die versteckten Schichten bestehen aus mehreren Schichten von Neuronengruppen, welche die zu grundlegende mathematische Funktion der Daten lernen. Am Schluss hat jedes Netz noch eine Ausgabeschicht, wo die Resultate des Netzes ausgegeben werden. Dieser output layer hat immer die Grösse des gewünschten Ergebnisses, d.h. wenn man ein Neuronales Netz (NN) trainieren möchte um zwischen gesunden und kranken Zellen zu unterscheiden, wird die Ausgabeschicht zwei Neuronen beinhalten.

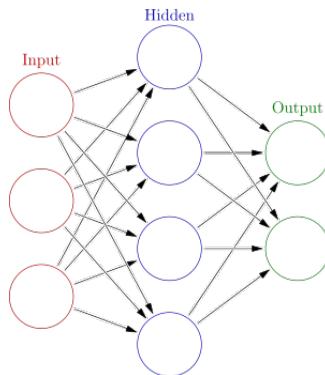


Abbildung 2.3.: Struktur eines neuronalen Netzes (source)

Training

Ein KNN muss trainiert werden um die richtigen Berechnungen und die richtigen Gewichte in den einzelnen Neuronen und Synapsen zu setzen. Dieses Training funktioniert im *Supervised Learning* so, dass dem KNN eine Vielzahl an Beispielen gegeben wird. Das neuronale Netz versucht dann, die gleichen Antworten wie im Beispiel zu erhalten. Wenn es eine falsche Antwort voraussagt, wird ein Fehler berechnet und die Werte an jedem Neuron und jeder Synapse werden für das nächste Mal rückwärts durch das NN propagiert. Diesen Prozess nennt man **Backpropagation** und wird in jedem KNN verwendet, um die einzelnen Parameter anzupassen.

Loss Funktion

Das in 2.1.4 beschriebene Training behilft sich einer sogenannten Loss Funktion um die Gewichte des Netzes anzupassen. Dies ist eine Funktion um zu evaluieren, wie gut die aktuelle nichtlineare Funktion des KNN die Daten repräsentiert. Wenn die Vorhersagen des NN weit von den echten Daten entfernt sind, gibt die Loss Funktion einen hohen Wert aus. Wenn diese allerdings einigermaßen ähnlich sind, ist der Wert der Funktion gering.

Es wird zwischen Regression Loss und Classification Loss unterschieden. Regression Loss Funktionen werden gebraucht bei Vorhersagen für kontinuierliche Werte. Classification Loss Funktionen werden gebraucht bei Vorhersagen von kategorischen Labels, zum Beispiel ob es sich auf einem Bild um eine Katze oder einen Hund handelt.

Diese Funktion wird minimiert um ein möglichst gutes Ergebnis zu erhalten, dies geschieht mittels einem Optimizer welcher in der Sektion 2.1.4 genauer beschrieben wird.

Optimizer

Ein Optimizer im KNN macht im Grunde nichts anderes als die Loss Funktion (2.1.4) zu Optimieren indem diese minimiert wird. Durch das kontinuierliche Minimieren der Loss Funktion performt das Modell nach jeder Iteration besser und sucht die optimale Funktion um die Daten zu repräsentieren. Die Gewichte des NN müssen nach jeder Iteration so angepasst werden, dass die Loss Funktion, welche mittels dem Output berechnet wird, minimiert wird. Diese Anpassung kann zum Beispiel mittels der Partiellen Ableitung der einzelnen Gewichte mit Respekt zum Loss berechnet werden. Dieser spezielle Optimizer wird auch Gradient Descent genannt.

2.2. Technische Konzepte

In diesem Abschnitt werden technische Konzepte genauer erklärt, welche in der Arbeit verwendet wurden. Diese Konzepte sind meist sehr umfangreich und werden daher nur angeschnitten, so dass die weiteren Schlüsse dieser Arbeit nachvollziehbar verstanden werden können.

2.2.1. Markov Chains

Markov-Ketten gehören zu den frühesten Algorithmen im Bereich der Natural Language Generation. Sie sagen das nächste Wort in einem Satz voraus, indem sie einfach das aktuelle Wort verwenden. Eine Markov-Kette berücksichtigt die Beziehung zwischen jedem einzelnen Wort, um die Wahrscheinlichkeit des nächsten Wortes zu berechnen. Sie wurden in früheren Versionen von Smartphone-Tastaturen verwendet, um Vorschläge für das nächste Wort im Satz zu generieren.

In Abbildung 2.4 wird eine solche Kette anhand eines Beispiels von „Andrew’s Adventures in Technology“, 2019 dargestellt. Die Datensätze für die Markov Kette sind: «I like turtles», «I like rabbits» und «I don’t like snails». Die Abbildung 2.4 bedeutet:

1. die Sätze zu 100% mit einem *I* starten
2. gefolgt von einem *like* zu 66% und 33% einem *don’t*
3. das Wort *don’t* wird zu 100% gefolgt von einem *like*
4. zum Schluss des Satzes gibt es zu 33% entweder ein *rabbits*, *turtles* oder *snails*

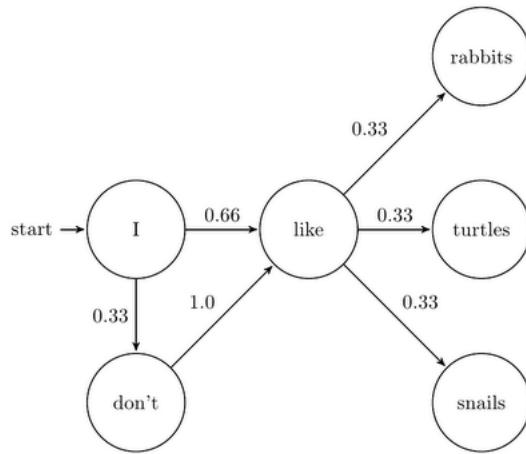


Abbildung 2.4.: Visualisierung einer Markov Chain für Sequenzen

2.2.2. Recurrent Neural Networks

RNN gehören zu den neusten Errungenschaften im Bereich der Neuronalen Netzen. Sie erlauben es, lange Sätze zu verarbeiten mit einer deutlichen Verbesserung der Genauigkeit. RNN ist eine spezielle Art der neuronalen Netzwerke, welche sich die sequenzielle Natur der Eingabe zu nutzen macht. Jeder einzelne Teil der Sequenz (z.B. ein Satz oder ein Vers) durchläuft ein Feedforward-Netzwerk und gibt die Ausgabe des Modells als Input für den nächsten Teil der Sequenz, wie in Grafik 2.5 gut ersichtlich ist. Dies ermöglicht die Speicherung von Informationen aus vorherigen Schritten, wie eine Art von Gedächtnis.

Durch dieses Gedächtnis über vorherige Schritte sind RNN besonders beliebt bei Sprachgenerierung, da sie sich an den Kontext des Satzes oder des Gespräches im Laufe der Zeit erinnern können. Aber nicht nur bei der Sprachgenerierung sind RNN beliebt, sondern auch bei: Spracherkennungen, Sprachmodellierungen und Übersetzungen.

RNN haben einen grossen Nachteil, nämlich das Problem des verschwindenden Gradienten (eng. «vanishing gradient problem»). Mit zunehmender Länge der Sequenz verliert das Netzwerk die Erinnerung an Wörter die weit zurück liegen und somit werden die Vorhersagen nur auf Grund von neuen Wörtern getroffen. Um dieses Problem zu lösen wurden Long short term memory (LSTM) 2.2.3 entwickelt, welche dieses Problem geschickt lösen.

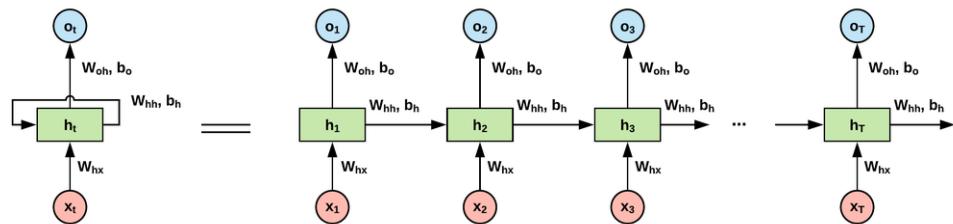


Abbildung 2.5.: Visualisierung eines RNN

2.2.3. Long Short-Term Memory

LSTM sind eine spezielle Art von Recurrent Neural Networks und somit auch eine Art von neuronalen Netzen. Die LSTM sind darauf ausgelegt, dass sie mit langen Sequenzen von Wörtern genauer arbeiten können. LSTM unterscheiden sich von herkömmlichen RNN in einem wichtigen Punkt, nämlich haben die LSTM ein vierstufiges NN. Normale RNN haben nur ein einschichtiges Netzwerk. Die kettenartige Struktur wurde auch in den LSTM gebraucht.

Das LSTM hat die Fähigkeit, Informationen vom Zellzustand zu entfernen oder hinzuzufügen, welche sorgfältig durch Strukturen reguliert werden, die als Gates bezeichnet werden. Gates sind eine Möglichkeit, Informationen optional durchzulassen. Sie bestehen aus einer sigmoiden neuronalen Netzsicht und einer punktweisen Multiplikationsoperation, wie in der Abbildung 2.6 zu sehen ist. Die sigmoidale Schicht des Netzwerks gibt eine Zahl zwischen Null und Eins aus, welche beschreibt wie viel von dem Vektor durchgelassen werden soll. Ein Wert von Null bedeutet, dass nichts durchgelassen werden soll und ein Wert von Eins bedeutet, dass alles durchgelassen werden soll. Ein LSTM hat drei solcher Gates um den Zellzustand zu kontrollieren.

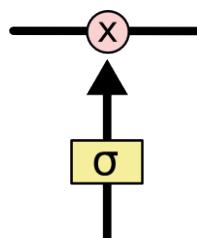


Abbildung 2.6.: Visualisierung eines LSTM Gates (Colah's Blog)

Ein Long Short-Term Memory besteht aus vier Komponenten, wie man der Grafik 2.7 entnehmen kann:

1. einer Zelle (cell), beinhaltet den Zellzustand
2. einem Eingangsgate (input gate)

3. einem Ausgangsgate (output gate)
4. einem Vergessensgate (forget gate)

Diese Komponenten ermöglichen es dem Netzwerk, Wörter über beliebige Zeitintervalle zu merken oder diese auch wieder zu vergessen, indem sie selber den Informationsfluss in und aus der Zelle regeln können. In der Zelle werden die Informationen über vorgängige wichtige Wörter gespeichert, so dass auf diese zu einem späteren Zeitpunkt zugegriffen werden können. Außerdem kann das Netzwerk erkennen, wann ein Satzende eingetroffen ist und sich der Kontext ändert. Durch diese Feststellung können durch das Vergessensgate Informationen übersprungen werden, welche in diesem Kontext für nicht mehr relevant angeschaut werden. Dies ermöglicht es dem Netzwerk, selektiv nur relevante Informationen zu verfolgen und gleichzeitig Informationen über einen längeren Zeitraum zu speichern.

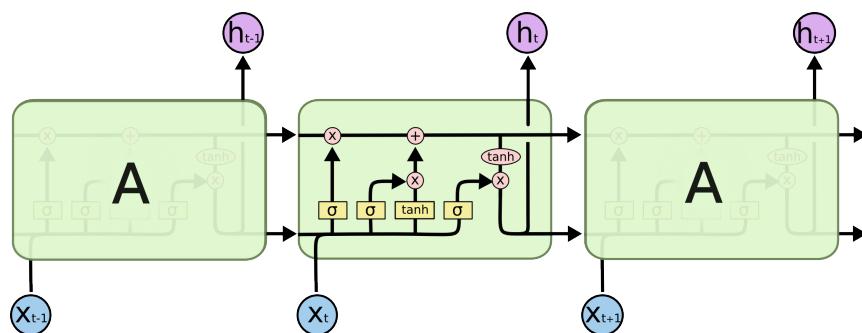


Abbildung 2.7.: Visualisierung eines LSTM (Colah's Blog)

LSTM und ihre Variationen schienen die Antwort auf das «vanishing gradient problem» zu sein, jedoch gibt es auch bei diesen Netzwerken gewisse Einschränkungen wie viele Informationen effektiv gespeichert werden können. Denn es muss immer noch ein komplexer sequentieller Pfad der letzten Zelle zur nächsten Zelle übergeben werden. Dadurch beschränkt sich die Länge der Sequenz, an die sich ein LSTM noch erinnern kann, auf wenige hundert Wörter.

Ein weiterer Nachteil ist, dass LSTM aufgrund ihrer hohen Rechenanforderungen sehr schwer zu trainieren sind. Aufgrund ihrer sequentiellen Natur sind sie schwer zu parallelisieren und können somit den Vorteil der heutigen Rechengeräten wie Graphics processing unit (GPU) und Tensor processing unit (TPU) nicht ausgiebig nutzen.

Ablauf eines Schrittes im LSTM

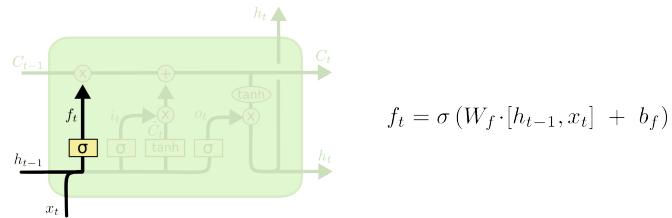


Abbildung 2.8.: Visualisierung eines LSTM Vergessensgates (Colah's Blog)

Der erste Schritt in einem LSTM ist es zu entscheiden, welche Informationen aus der Zelle verworfen werden. Diese Entscheidung wird durch die Sigmoid Funktion des Vergessensgates getroffen. Es schaut sich den Input x_t und den Output des vorherigen Wortes h_{t-1} an und gibt eine Zahl zwischen 0 bis 1 aus für jeden einzelne Komponente des Zellzustandvektor C_{t-1} . Eine 1 bedeutet, dass der Wert komplett behalten werden soll. Eine 0 hingegen bedeutet, dass der Wert komplett verworfen werden soll. Dadurch entsteht ein «Vergessensvektor» f_t .

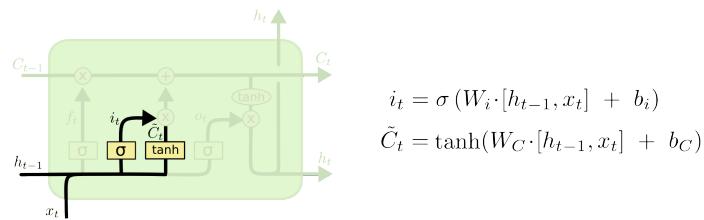


Abbildung 2.9.: Visualisierung eines LSTM Eingangsgates (Colah's Blog)

In einem nächsten Schritt wird darüber entschieden, welche neuen Informationen im Zellzustand gespeichert werden. Diese Entscheidung wird in zwei Schritten getroffen. Als Erstes entscheidet die Sigmoid Funktion des Eingangsgates, welche Werte aktualisiert werden und erstellt einen «Aktualisierungsvektor» i_t , welcher für jeden Wert des Zellzustands eine Zahl zwischen 0 und 1 beinhaltet. Als nächstes erstellt eine Tanh Funktion einen neuen Zellzustandvektor \tilde{C}_t mit den aktualisierten Werten des Zustands. In einem nächsten Schritt werden die beiden Vektoren i_t und \tilde{C}_t kombiniert und der Zellzustand wird aktualisiert.

Anschliessend wird der alte Zellzustand C_{t-1} zum neuen Zustand C_t aktualisiert. Der alte Zustand wird mit dem berechneten «Vergessensvektor» f_t multipliziert, um die Dinge zu vergessen, die irrelevant geworden sind. Dann multiplizieren wir das Ergebnis des vorherigen Schrittes mit dem «Aktualisierungsvektor» i_t kombiniert mit dem neuen Zellzustandvektor \tilde{C}_t . Aus dem ergibt sich der neue Zellzustand C_t .

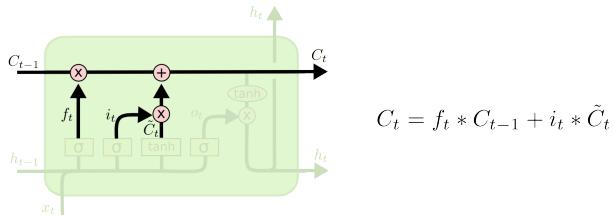


Abbildung 2.10.: Visualisierung der Addition der Vektoren eines LSTMs (Colah's Blog)

Schliesslich muss entschieden werden, was ausgegeben werden soll, diese Entscheidung wird im Ausgangsgate getroffen. Der Output basiert auf dem neuen Zellzustandvektor C_t , wird aber vorgängig noch gefiltert. Zuerst wird eine Sigmoid Funktion auf dem Input x_t und dem Output des vorherigen Wortes h_{t-1} ausgeführt, um zu entscheiden welche Teile des Zustandes ausgegeben werden. Dann wird der aktuelle Zellzustand C_t mittels einer Tanh Funktion auf die Werte zwischen -1 und 1 gebracht und anschliessend mit dem Resultat der Sigmoid Funktion multipliziert. Somit werden nur die Teile ausgegeben, die wirklich relevant sind.

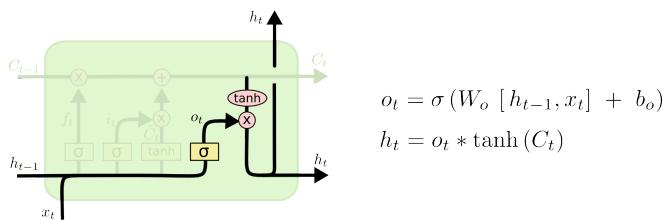


Abbildung 2.11.: Visualisierung eines LSTM Ausgangsgate (Colah's Blog)

2.2.4. Autoencoder

Autoencoder sind eine spezielle Art von KNN, welche oft verwendet werden, um effiziente Datencodierungen auf unbeaufsichtigte (unsupervised) Weise zu erlernen. Ein Autoencoder besteht auf der einen Seite aus einem Encoder und auf der anderen aus einem Decoder.

Das Ziel des Encoders ist es, eine Darstellung für einen Datensatz zu erlernen, typischerweise mit dem Zweck der Reduzierung der Dimensionalität. Indem das Netzwerk trainiert wird, lernt es das Rauschen oder Unreinheiten zu ignorieren und sich auf die echten Daten zu konzentrieren.

Der Decoder hat das Ziel, aus der reduzierten Codierung eine Darstellung zu rekonstruieren, die seiner ursprünglichen Form so nahe wie möglich kommt.

Durch diese Eigenschaft der Komprimierung und der Rekonstruktion werden Autoencoder effektiv zur Lösung vieler Probleme eingesetzt, von der Gesichtserkennung bis zum Erlan-

gen der semantischen Bedeutung für die Worte.

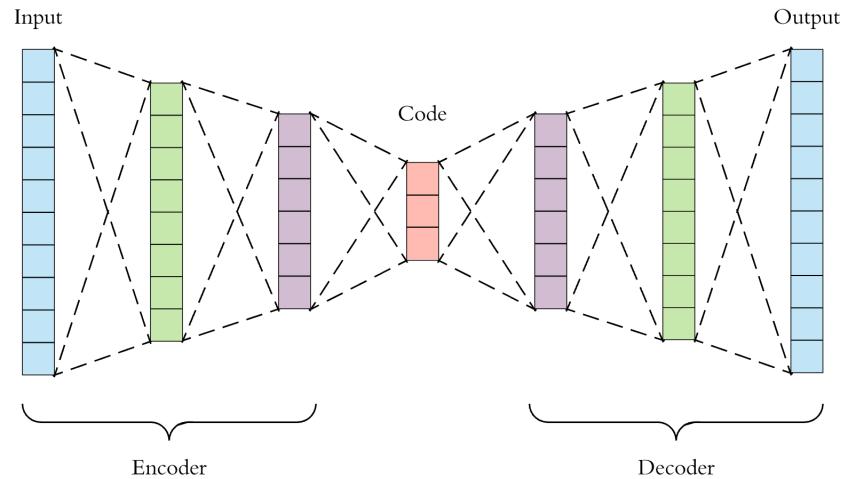


Abbildung 2.12.: Visualisierung eines Autoencoders

2.2.5. Transformer

Transformer sind eine weitere Art von KNN, welche sehr intuitiv verstanden werden können. Die Hauptaufgabe besteht darin, einen Input (z.B. einen deutschen Satz) in einen Output (z.B. in einen englischen Satz) zu transformieren. Das Transformermodell besteht aus mehreren Encoder- und Decoder Komponenten, welche zusammen verbunden sind. Die Encoder Komponenten sind nichts weiteres als mehrere Encoders, welche aufeinander gestapelt werden und eine Verbindung zum oberen Encoder haben. Dasselbe wird auch bei der Decoder Komponente gemacht, dort werden mehrere Decoders aufeinander gestapelt und sind mit dem oberen Layer verbunden. Der oberste Encoder der Encoder Komponente ist mit jedem Decoder der Decoder Komponente verbunden. Dem ersten Encoder wird der Input übergeben und der letzte Decoder liefert den gewünschten Output.

Transformatoren wurden entwickelt, um das Problem der Sequenztransduktion (Transformation einer Eingangssequenz zu einer Ausgangssequenz) oder der neuronalen maschinellen Übersetzung zu lösen. Dazu gehören Spracherkennung, Text-zu-Sprache Transformation, Übersetzungen und noch viele weitere.

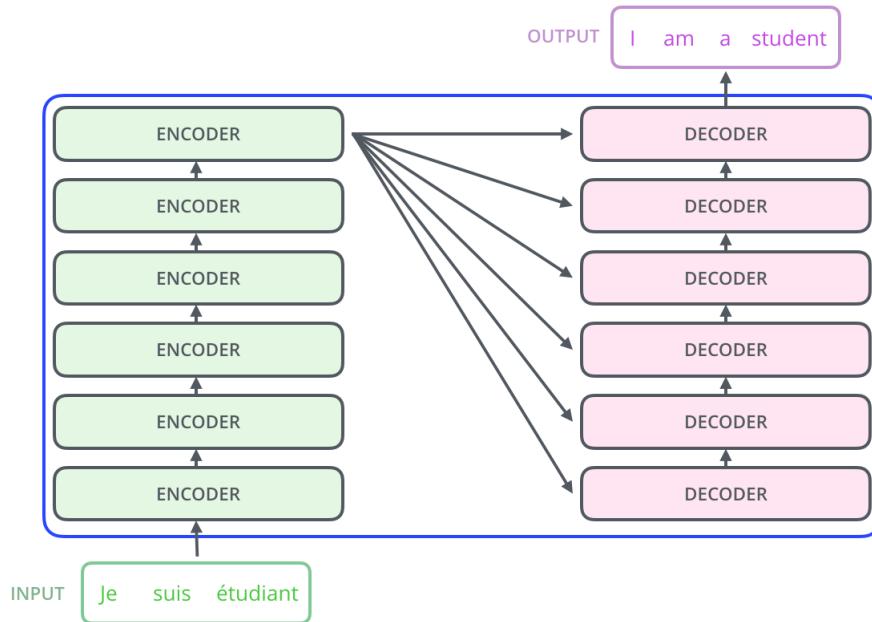


Abbildung 2.13.: Visualisierung eines Transformers

2.2.6. Generative Adversarial Networks

Generative Adversarial Networks (GAN) verfolgen einen spieltheoretischen Ansatz, im Gegensatz zu einem herkömmlichen neuronalen Netzwerk. Das Netzwerk lernt die Generierung aus einer Trainingsverteilung durch ein 2-Spieler-Spiel. Die beiden Spieler sind der Generator und der Diskriminator. Diese beiden Gegner befinden sich während des gesamten Trainingsprozesses im ständigen Kampf miteinander. Da eine gegensätzliche (adversielle) Lernmethode angewandt wird, beinhaltet das Netzwerk einige Eigenheiten.

Der Generator wird verwendet um aus einem zufälligen Rauschen möglichst echte Daten zu generieren. Und der Diskriminator hingegen soll unterscheiden können, ob es sich bei den Daten um echte oder generierte Daten handelt. Die beiden Instanzen befinden sich im konstanten Kampf gegeneinander, wobei der Generator, versucht möglichst echte Daten zu erzeugen und der Diskriminator versucht sich nicht täuschen zu lassen. Um möglichst echte Daten zu generieren, benötigt es einen sehr guten Generator, so wie einen sehr guten Diskriminator. Dies liegt daran, dass wenn der Generator nicht gut genug ist, er nie in der Lage sein wird, den Diskriminator zu täuschen und das Modell nie konvergieren wird. Wenn aber der Diskriminator schlecht ist, dann werden auch Bilder, welche keinen Sinn geben, als echt eingestuft. Dadurch kann das Modell nicht korrekt trainiert werden, denn es gibt keinen genug guten «Validator» der Daten.

Der Ablauf eines Schrittes in einem GAN fängt mit der Erzeugung eines Zufallsrauschen an. Dieses Rauschen wird danach dem Generator übergegeben, welcher daraus einen mög-

lichst echten Datensatz generiert. Dieser generierte Datensatz wird anschliessend dem Diskriminatator übergeben, welcher entscheidet ob die Daten generiert oder echt sind und gibt demnach ein entsprechendes Label aus.

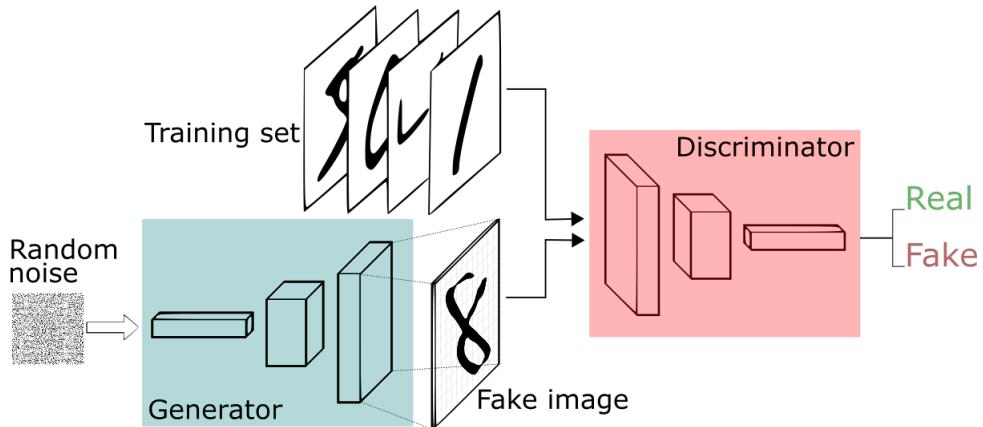


Abbildung 2.14.: Visualisierung eines GANs

2.2.7. Aktivierungsfunktionen

Aktivierungsfunktionen werden in NN gebraucht, um die Ausgaben der einzelnen Layers zu bestimmen. Die Ausgaben der Layers können mit Ja oder Nein verglichen werden, nur sind es bei den Aktivierungsfunktionen entweder eine 0 bis 1 oder -1 bis 1, je nach Funktion. Aktivierungsfunktionen lassen sich in zwei Typen unterteilen: lineare Aktivierungsfunktionen und nicht lineare Aktivierungsfunktionen.

Lineare Aktivierungsfunktionen

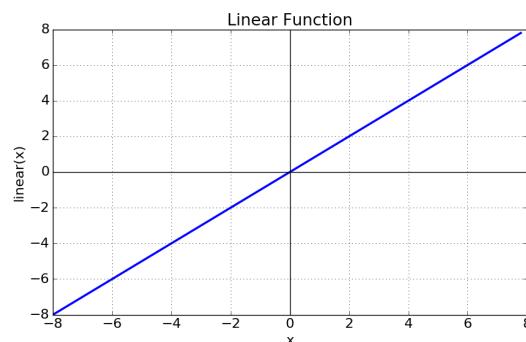


Abbildung 2.15.: Lineare Aktivierungsfunktion

Bei der linearen Aktivierungsfunktion ist die Funktion eine Linie, die stetig steigt. Daher ist die Ausgabe der Funktion nicht auf einen Bereich beschränkt. Eine solche Funktion kann

zum Beispiel $f(x) = x$ sein, diese befindet sich im Bereich von $-\infty$ bis ∞ . Das Problem bei linearen Funktionen ist, dass die Daten meist nicht linear verteilt sind und somit kann sich das Modell nicht genügend den Daten anpassen.

Nicht lineare Aktivierungsfunktionen

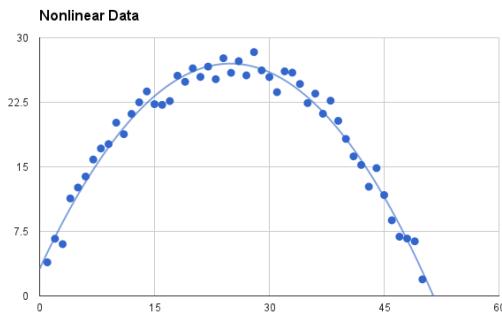


Abbildung 2.16.: Nicht lineare Aktivierungsfunktion

Nicht lineare Aktivierungsfunktionen werden für die Ausgabe von Layern der neuronalen Netzen am häufigsten verwendet. Durch die nicht Linearität kann sich das Modell leicht, verschiedensten verteilten Daten anpassen und sehr viele verschiedene «Formen»annehmen. Eine solche Funktion kann zum Beispiel $f(x) = x^2$ sein.

Sigmoid

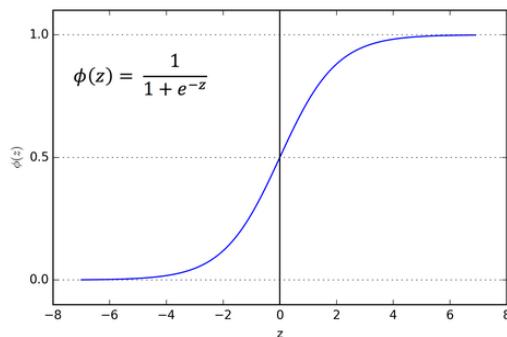


Abbildung 2.17.: Sigmoid Funktion

Eine Sigmoidfunktion ist eine mathematische Funktion mit einem S-förmigen Graphen, sie ist eine der weitverbreitesten Funktionen im Umgang mit neuronalen Netzen. Der Grund dafür ist, dass sie sich zwischen 0 und 1 befindet. Dadurch ist die Sigmoidfunktion sehr nützlich an Orten, wo man eine Vorhersage über Wahrscheinlichkeiten treffen muss, da sich Wahrscheinlichkeiten auch immer zwischen 0 und 1 befinden.

Ein Problem der Sigmoidfunktion ist, dass Parameter «stecken bleiben» können, weil sich der Bereich der Funktion nur zwischen 0 und 1 befindet. Somit ist es möglich, dass Parameter die sich sehr nahe bei 0 befinden nur wenig angepasst werden und somit gleich bleiben. Dies kann bei Parametern passieren, welche erst nach einer gewissen Zeit zum Vorschein kommen.

Tanh

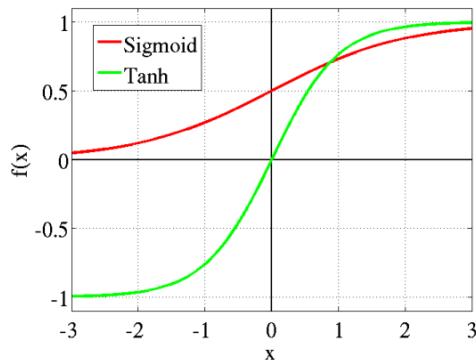


Abbildung 2.18.: Tanh Funktion

Wie die Sigmoidfunktion, ist auch die Tanhfunktion eine mathematische Funktion mit einem S-förmigen Graphen, gibt aber stattdessen Werte im Bereich von -1 und 1 aus. Somit können stark negative Einträge in der Funktion negativ abgebildet werden, was in der Sigmoidfunktion nicht möglich ist. Zusätzlich werden nur 0 ähnliche Einträge auf nahezu 0 abgebildet, somit wird das «stecken bleiben» beim Training eines Parameters unwahrscheinlicher. Die Tanhfunktion wird vor allem bei der Klassifikation von zwei verschiedenen Klassen verwendet.

ReLU (Rectified Linear Unit)

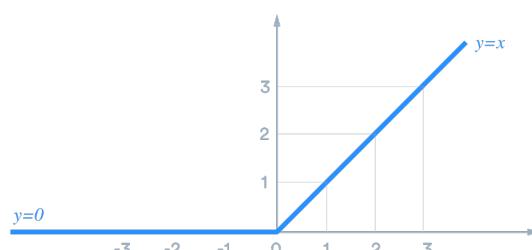


Abbildung 2.19.: ReLU Funktion

ReLU ist die am weitesten verbreitete und die am meisten benutzte Aktivierungsfunktion. Sie wird vor allem in Convolutional Neural Networks sowie in den meisten Deep Learning Modellen gebraucht. Der Bereich der Funktion ist von 0 bis ∞ . Außerdem ist die Funktion, sowie ihre Ableitung monoton.

Das Problem der ReLU Aktivierungsfunktion ist das Abbilden der negativen Werte, denn jede negative Eingabe der Funktion wird zu 0 werden. Dies verringert die Fähigkeit des Modells sich den Daten richtig anzupassen und zu trainieren, denn die negativen Werte können nicht entsprechend abgebildet werden. Aus diesem Grund wurde die **Leaky ReLU Aktivierungsfunktion** entwickelt, welche dieses Problem löst.

2.2.8. N-Gramm

N-Gramme sind das Ergebnis der Zerlegung eines Textes oder Satzes in einzelne Fragmente. Der Text wird dabei zerlegt und jeweils N aufeinanderfolgende Fragmente werden als N-Gramme zusammengefasst. Die Fragmente können Buchstaben, Wörter oder ähnliches sein. Die Zerlegung eines Satzes in N-Gramme wird vor allem zur Hilfe von Analysen oder statistischen Auswertungen gebraucht.

Es gibt verschiedene Arten von N-Grammen. Wichtige Arten sind zum Beispiel Monogramme, bei dem $N = 1$, oder Bigramme, bei dem $N = 2$ ist. Wobei N für die Anzahl Zeichen steht die betrachtet werden.

Beispiel:

$$\begin{aligned}s(\text{Zeichenkette}) &= \text{„WIPRO“} \\ N(\text{Anzahl Zeichen}) &= 2 \text{ (Bigramm)} \\ f(\text{Frequenzvektor}) &= \{_W : 1, WI : 1, IP : 1, PR : 1, RO : 1, O_ : 1\}\end{aligned}$$

2.2.9. Metriken

Im Bereich von NLG sind Metriken ein zentrales Thema, da dort ein Vergleich von Labels nicht reicht. Daher mussten neue Metriken entwickelt werden, die auf den Inhalt der Eingabe und Ausgabe Daten schauen und auf Grund dieses einen Wertes berechnen wie gut die beiden Daten zusammenpassen. Diese Problematik ist komplex, denn es gibt keinen Blueprint, der für jeden Fall passt. Eher gibt es einen ganzen Stapel an verschiedenen Methoden zur Auswertung, aus denen eine entsprechende Metrik gewählt werden muss.

Die Intuition für die Bewertung von generiertem Text ist die gleiche wie für die Bewertung von Labels. Wenn der Text A näher an einem der Referenztexte liegt als der Text B , dann soll A höher bewertet werden als B . Wie bei anderen Verfahren basiert diese Übereinstimmung auf Präzision (Spezifität, eng. specificity) und Erinnerung (Sensitivität, eng. sensitivity). Einfach ausgedrückt, A ist genauer als B , falls der Prozentsatz von A , der mit einem Referenztext übereinstimmt, höher ist als B . Die Erinnerung von A ist höher, wenn er mehr übereinstimmenden Text aus einer Referenz enthält als B .

Bilingual evaluation understudy (BLEU)

BLEU ist die weitverbreiteste Metrik für die Bewertung von maschinellen Übersetzungssystemen. Sie wurde im Juli 2002 von einer Forschungsgruppe von IBM entwickelt, da bis dort hin die Metriken zur Auswertung von Machine Translation zu ineffizient waren. Die Grundidee von BLEU ist: BLEU Scores sollen den Unterschied zwischen Referenzübersetzungen und maschinellen Übersetzungen messen. Dafür werden Präzision und Erinnerung durch modifizierte n-Gramm-Präzision bzw. beste Abgleichlänge approximiert.

Zunächst werden einzelne Segmente (meist Sätze) verglichen, später wird ein Durchschnittswert für den gesamten Text ermittelt. Je näher die maschinelle Übersetzung der Referenzübersetzungen kommt, umso besser ist ihr Score. Generell wird dabei eine Skala von 0 bis 1 verwendet, auf welcher der Wert 1 identisch mit der qualitativ hochwertigen Referenzübersetzung ist, während ein Score von 0 signalisiert, dass die maschinelle Übersetzung keine Übereinstimmungen mit der Referenz hat.

Ein BLEU Score von 1 ist nicht erstrebenswert, denn dies würde bedeuten, dass die maschinelle Übersetzung identisch mit der Referenz ist. Dies sollte jedoch in keinem Fall das Ziel sein. Es sollte versucht werden, eine möglichst korrekte Übersetzung zu liefern und nicht die Referenzübersetzungen zu imitieren.

Interpretation Es wird dringend davon abgeraten, BLEU-Werte über verschiedene Korpusse und Sprachen hinweg zu vergleichen. Auch der Vergleich der BLEU-Werte für denselben Korpus, aber mit einer abweichenden Anzahl von Referenzübersetzungen, kann sehr irreführend sein. Die genauere Interpretation des Wertes ist in Tabelle 2.2 ersichtlich.

BLEU-Wert	Interpretation
< 0.1	Fast unbrauchbar
0.1 – 0.2	Schwierig, das Wesentliche zu verstehen
0.2 – 0.3	Das Wesentliche ist verständlich, aber es gibt erhebliche Grammatikfehler
0.3 – 0.4	Verständliche bis gute Übersetzungen
0.4 – 0.5	Hochwertige Übersetzungen
0.5 – 0.6	Sehr hochwertige, adäquate und flüssige Übersetzungen
> 0.6	Qualität oft besser als menschliche Übersetzungen

Tabelle 2.2.: Interpretation des BLEU Wertes

Mathematischen Details Mathematisch wird der BLEU-Score definiert mit („Modelle bewerten AutoML“, 2019):

$$\text{BLEU} = \underbrace{\min \left(1, \exp \left(1 - \frac{\text{reference-length}}{\text{output-length}} \right) \right)}_{\text{brevity penalty}} \underbrace{\left(\prod_{n=1}^4 \text{precision}_n \right)^{1/4}}_{\text{n-gram overlap}} \quad (2.3)$$

mit

$$\text{precision}_n = \frac{\sum_{\text{sentence} \in \text{Candidates-Corpus}} \sum_{n \in \text{sentence}} \min(m_{\text{cand}}^n, m_{\text{ref}}^n)}{w_t^n = \sum_{\text{sentence}' \in \text{Candidates-Corpus}} \sum_{n' \in \text{sentence}'} m_{\text{cand}}^{n'}} \quad (2.4)$$

wobei n die Anzahl an n -grams ist.

Wobei Folgendes gilt:

- m_{cand}^n entspricht der Anzahl an n-Gramme für den Kandidaten, die mit der Referenzübersetzung übereinstimmen
- m_{ref}^n entspricht der Anzahl an n-Gramme in der Referenzübersetzung
- w_t^n entspricht der Gesamtzahl der n-Gramme in der Kandidatenübersetzung

Die Formel besteht aus zwei Teilen: dem Abzug für die Kürze (eng. brevity penalty) und der N-Gramm Übereinstimmung (eng. n-gram overlap).

- Der Abzug für die Kürze bestraft generierte Übersetzungen, die verglichen mit der ähnlichsten Referenzlänge exponentiell abnehmend zu kurz sind. Dies kompensiert den fehlenden Trefferquoten (eng. recall) Term.
- Die N-Gramm-Übereinstimmung zählt, wie viele N-Gramme mit ihrem N-Gramm-Gegenstück in den Referenzübersetzungen übereinstimmen. Über die N-Gramm-Übereinstimmung wird die Genauigkeit (eng. precision) der Übersetzung gemessen. Kürzere N-Gramme ermitteln die Ädequatheit, längere N-Gramme die Flüssigkeit der Übersetzung. Zur Vermeidung einer unnötigen Zählung wird die N-Gramm-Zählung auf die maximale N-Gramm-Anzahl begrenzt, die in der Referenz auftritt (m_{ref}^n).

Translation error rate (TER)

Während es bei BLEU vor allem darum geht, zu bestimmen, wie nah eine maschinelle Übersetzung einer Referenz kommt, sagt TER aus, wie viele Schritte im Post-Editing benötigt werden, um von der erstellten maschinellen Übersetzung zur korrekten Übersetzung zu gelangen. BLEU und TER sind sich in vielerlei Hinsicht sehr ähnlich und werden daher gerne zusammen verwendet.

TER wie auch BLEU, interessieren sich nicht für die zu grunde liegende Sprache des Satzes, sondern der Algorithmus möchte nur herausfinden, wie viele Unterschiede es zwischen dem Referenzsatz und dem generierten Satz gibt.

Auch TER benötigt einen Referenzkorpus an dem gemessen werden soll, wie nahe der generierte Output der Referenz kommt.

Diese Metrik wird dazu verwendet um zu prüfen, wie viele verändernde Schritte nötig sind um ein gutes Ergebnis zu erhalten.

$$\text{TER} = \frac{\# \text{ of edits}}{\text{average } \# \text{ of references}} \quad (2.5)$$

2.3. Stand im Bezug auf eigenes Projekt

Alle erwähnten technischen Konzepte sind längst keine eigenständigen, geschlossenen Einheiten mehr. Sonder werden in den aktuellen Forschungen miteinander verknüpft und es werden vielversprechende Ansätze von einem Konzept ins Andere übernommen, um neue Architekturen für verschiedene Probleme zu entwickeln. Auf diese Art und Weise sind die meisten Durchbrüche der aktuellen Forschung gelungen.

Eines der aktuellsten Themen im Bereich von Natural Language Generation (2.1.1) und Natural Language Processing (2.1.2) ist Neural Style Transfer (2.1.3), welches in momentanen Forschungen auf Textbasierte Daten angewendet wird, mit moderaten Ergebnissen, um den Stil eines Satzes auf einen anderen zu übertragen. Bei diesem Ansatz werden meist Recurrent Neural Networks (2.2.2), Long Short-Term Memory (2.2.3), Autoencoder (2.2.4) und viele weiter der oben erwähnten Konzepte verwendet.

3. Ideen und Konzepte

In diesem Kapitel werden auf die verschiedenen Ideen und Konzepte genauer eingegangen, welche in der Arbeit verwendet wurden. Es beinhaltet Ideen zu dem verwendeten Datensatz, den verwendeten Modellen und dem Training der Modelle.

3.1. Problemdefinition

Der Aufbau eines Satzes S kann linguistisch als Semantik und Syntax beschrieben werden.

$$S = S_{semantic} + S_{syntax} \quad (3.1)$$

Syntax und Semantik in der natürlichen Sprache zu trennen, ist eine Herausforderung. Da vor allem das Erkennen der Semantik schwierig ist, denn die selben Wörter in unterschiedlichen Kontexten können zur Syntax oder auch zur Semantik gehören. Zu beachten ist, dass die Syntax einen Einfluss auf die Semantik haben kann. So zum Beispiel das Pronomen «nichts» und das gross geschriebene Substantiv «Nichts». So kann der Transfer, welcher «positive» in «negative» Sätze wandelt, die Semantik des Ausgabesatzes beeinflussen. Je- doch möglichst den Bezug zum Kontext, dem positiven oder negativen Subjekt, zu wahren.

Der Input des zu erarbeitenden Prototypen des Zeugnismanagers, ist die Bewertung c_P einer Person P sowie eine Eingabesequenz x . Diese Bewertung c_P ist dem Kontext eines Satzes gleich zu stellen, und soll beim Transfer nicht geändert werden. Ausserdem wird jeder Satz einem stylistischen Label y zugeordnet. Aus der Bewertung, Stylelabel und dem Eingabesatz folgt x welcher aus c_P und y zusammengesetzt ist, ähnlich wie 3.1.

$$\begin{aligned} y_k &= holprig = klexikon \\ &\text{oder} \\ y_w &= flssig = wikipedia \end{aligned} \quad (3.2)$$

Diese Labels sind wie in Formel 3.2 ersichtlich, in *holprig* und *flssig* unterteilt. Dabei soll *holprig* bedeuten, dass der Satz aus dem Zeugnismanager generiert wurde und noch gewisse Verbesserungen vorgenommen werden müssen bevor dieser in ein Arbeitszeugnis einfließen kann. *flssig* hingegen bedeutet, dass die Sequenz qualitativ hochwertig ist und nicht mehr weiter bearbeitet werden muss. Diesen definierten Labels wurden, aufgrund der Abschliessende Problemdefinition (3.2) ein synonymes Label zugeordnet, welches den Datensatz 3.3 wiederspiegelt. Aus der Anforderung folgt, dass der angestrebte Zielstil y_w ist.

Wie aus der Aufgabenstellung (1.1.2) zu entnehmen ist, entspricht der Stil der Eingabesatzes x dem Stillabel y_k und soll in den Zielstil y_w transferiert werden, ohne dass dessen Bewertung c_P verloren geht. Somit entsteht ein neuer transferierter Satz \hat{x} .

$$\hat{x} = (x - y_k) + y_w \quad \text{mit} \quad y = y_w \quad (3.3)$$

3.2. Abschliessende Problemdefinition

Ein Wissenstext für Kinder aus dem Klexikon, unterscheidet sich im Schreibstil zu einem Text für Erwachsene aus Wikipeida. Durch das Ersetzen von *holprig* und *fllsig*, durch *klexikon* und *wikipedia* kann ein ähnliches Problem, wie in Problemdefinition (3.1) beschrieben, modelliert werden. Wie unter Statistische Analyse des Datensatz (5.1.2) aufgezeigt, beinhalten die Sätze von Wikipedia, Stillabel *wikipedia*, mehr Wörter als die Sätze aus dem Klexikon, mit Stillabel *klexikon*. Daher kann das Problem abschliessend so beschreiben werden, dass der Style Transfer aus einem kurzen Satz einen längeren Satz mit gleichem, erweiterten Inhalt generiert. Entsprechendes gilt für den umgekehrten Fall. So sollte die Länge der generierten Sätze aus einem Stillabel der Verteilung der Länge aus dem Zielstil folgen.

3.3. Verwendeter Datensatz

Um die mögliche Modelle evaluieren zu können, wird ein Datensatz aufbereitet. Für die Gebiete von Arbeitszeugnisse sind leider zu wenig Daten vorhanden. Eine naheliegende Möglichkeit ist, die Daten eines Lexikon oder andere, weitverbreitete und vielfach übersetzte Informationen. Die Texte weisen dabei einen zielgruppengerechten Schreibstil auf.

Themengebiete aus welchem entsprechende Texte stammen könnten sein,

1. Lexikon / Wikiseite
2. Bibel
3. Zeitungen

Bei der Recherche für eine geeignete Datenquelle, wurden Zeitungen früh ausgeschlossen. Es finden sich keine praktikablen und einfach zusammentragbaren Kinderzeitungen online. Weiter wären die Themen der Zeitungsartikel sehr breit aufgestellt. Auch fokussieren sich Kinderzeitungen nicht auf das gleiche Weltgeschehen wie Zeitungen.

Aus dem Themengebiet der Religion, namentlich «die Bibel», finden sich zahlreiche Quellen. Dabei unterscheidet sich jedoch der Inhalt teilweise enorm. Hierzu werden in Kinderbibeln oft eine Vielzahl an Illustrationen verwendet. Auch sind sie oft als Erzählbücher gestaltet. So unterscheiden sie sich nicht zwingend im Schreibstil, jedoch mehr in der Wortwahl. Weiter fanden sich wenige Quellen, bei welchen die Daten in einem handhabbaren Format aufbereitet sind. Eine Extraktion der Texte aus PDFs oder gar gebundenen Büchern hätte einen zu grossen Aufwand mit sich gezogen. Daher wurde die Verwendung der Bibel als

Quelle bis auf weiteres zurückgestellt.

An Lexikas gibt es eine Vielzahl von Ausgaben. Sowohl für Erwachsene als auch für Kinder. Die naheliegenste Quelle für Texte für Fortgeschrittene ist, die Online Enzyklopädie «Wikipedia». Bei der Recherche liess sich ebenfalls ein Online Lexikon für Kinder finden, das «Klexikon». Diese beiden Quellen basieren auf dem Open-Source Framework «WikiMedia». Entsprechend kann einfach ein Crawler für diese Seiten erstellt werden. Das Klexikon enthält ca. 2'700 Artikel, in einem für Kinder verständlichen Schreibstil. Für nahezu alle Artikel besteht ein korrespondierender Artikel in Wikipedia. Nach Rücksprache mit der Betreuungsperson hat man sich für die Verwendung dieser Artikel entschieden. Die Aufbereitung des Datensatzes wird im Kapitel Realisierung (5) beschrieben.

3.4. Momentan verfügbare Forschung

In einem ersten Teil wurde nach bestehenden Forschungen, im Bereich der Problemstellung, gesucht. Es konnten schnell viele verschiedene Ansätze gefunden werden, um das Problem lösen zu können. Dabei gab es viele unterschiedlichen Lösungsansätzen, Neural Style Transfer (NST), Recurrent neural network (RNN), Transformer, Fuzzy Logic und noch viele weitere. Es wurde schnell klar, dass das Gebiet eingeschränkt werden muss, um die Ressourcen des Projektes optimal auszuschöpfen. Das spannendste Teilgebiet war der NST, auf welches sich diese Forschung hauptsächlich konzentrierte.

Bei der Suche nach den vielversprechendsten Ansätzen von NST, wurden eine Menge an aktuellen Forschungen gefunden, da das Gebiet als sehr jung gilt. Die meisten dieser Arbeiten fokussieren sich auf die Übertragung von der Stimmungslage in eine andere, zum Beispiel von einem positiven in einen negativen Satz. Diese Problemdefinition wird im NST immer wieder verwendet um neue Modelle zu testen. Meistens werden dafür Sätze von Bewertungen (z.B. Tripadvisor oder IMDB) als Datensatz verwendet. Diese Problemstellung dient in den meisten Fällen nur zur Veranschaulichung der Einsatzmöglichkeiten der Modelle.

Bei der Recherche wurde ein sehr gut gepflegtes und aktuelles GitHub Projekt gefunden, welches sich darauf fokussiert, die neusten wissenschaftlichen Arbeiten, im Gebiet des Style Transfer, zu verlinken. Außerdem wird immer versucht Code zu den entsprechenden Werken zu finden und zu verlinken. Das Repository findet sich auf GitHub unter GitHub Style Transfer in Text (Fu, o.D.).

Im Projekt werden die Arbeiten in verschiedene Abschnitte unterteilt, je nach dem, um welches Untergebiet es sich handelt: Dataset, Supervised (Parallel Data), Unsupervised (Non-parallel Data) und Semi-Supervised. Es wurde sehr schnell klar, dass die Unterlagen im Bereich Supervised und Semi-Supervised nicht erforscht werden müssen, da es sich bei unserem Datensatz 3.3 um nicht parallele Daten handelt.

Im Bereich des Unsupervised Style Transfer, wurden verschiedene Arbeiten, im Bezug auf die Anwendbarkeit auf unsere Problemdefinition, untersucht. Dabei wurden vor allem die

Unterlagen mit entsprechendem Code untersucht, da die Umsetzung solcher Arbeiten sehr aufwendig ist. Die meisten dieser Papers beschreiben ein spezielles Modell, welches einen Style Transfer für bestimmte Probleme vornehmen kann. Diese Modelle sind meistens sehr ähnlich aufgebaut und haben viele Gemeinsamkeiten. Notizen zu den einzelnen Arbeiten kann im Anhang unter Recherche (B) gefunden werden.

Es wurde schnell bemerkt, dass viele der untersuchten Arbeiten immer wieder die Modelle **ControlGen** und **CrossAlign** brauchen um neue exotische Architekturen zu überprüfen, meistens mit der Frage: „Ist die neue Architektur besser als die allgegenwärtige?“ Daher wurde sehr schnell klar, das in diesem Projekt diese Baseline Modelle verwendet werden sollen, da es sich bei dem Problem um erste Versuche mit NST in deutscher Sprache handelt. Weitere Architekturen können in einem weiterführenden Projekt untersucht werden, übersteigen jedoch den Rahmen dieses Projektes.

Anschliessend wurden die Unterlagen zum ControlGen (Hu, Yang, Liang, Salakhutdinov und Xing, 2017) und CrossAlign (Shen, Lei, Barzilay und Jaakkola, 2017) Modell durchgearbeitet. Beide Arbeiten der Modelle haben auch eine Implementation in Tensorflow, diese wurden auch genau untersucht, um die Architekturen besser zu verstehen. Sehr schnell wurde bemerkt, dass die Standardimplementierungen der Modelle eher dürftig sind und es wurde weiter gesucht, nach besseren Implementierungen.

Im Repository wurde dann das Paper (Li et al., 2019) gefunden, welches die beiden Architekturen als Baseline brauchen um ihre domain adaptiven Modelle damit zu vergleichen. Die Arbeit möchte vor allem zeigen, dass es mit ihren Modellen möglich ist, durch parallele Daten nicht nur den Stil zu erlernen, sondern auch deren Domäne und ihre Eigenschaften. Die Idee des Projektes ist sehr interessant, jedoch können diese speziellen Modelle nicht auf unseren Datenkorpus angewendet werden. Das Projekt bietet aber einen hochwertigen Implementation in Python und Tensorflow zu den beiden Modellen. Daher wurde entschieden dieses Projekt für die Experimente und das weitere Projekt zu verwenden. Das Original Repository findet sich auf GitHub unter GitHub DAST (Li et al., o.D.).

3.4.1. Modelle

Das Repository (Li et al., o.D.) welches in diesem Projekt verwendet wurde, beinhaltet zwei Style Transfer Modelle, ControlGen (3.4.4) und CrossAlign (3.4.5). Diese beiden sind im Grundsatz sehr ähnlich und erben daher beide vom BaseModel (3.4.3), daher können die Modelle gleich trainiert, getestet und verwendet werden. Ausserdem sind die Hyperparameter der beiden Architekturen identisch und können somit gegenüber gestellt werden. Um die beiden Modelle zu evaluieren, befindet sich in dem Projekt ein Classifier (3.4.2). Dieser ist dafür zuständig, zu überprüfen ob die Sätze zu dem korrekten Style transferiert wurden. Mittels dem Classifier kann daher eine *transfer accuracy* berechnet werden.

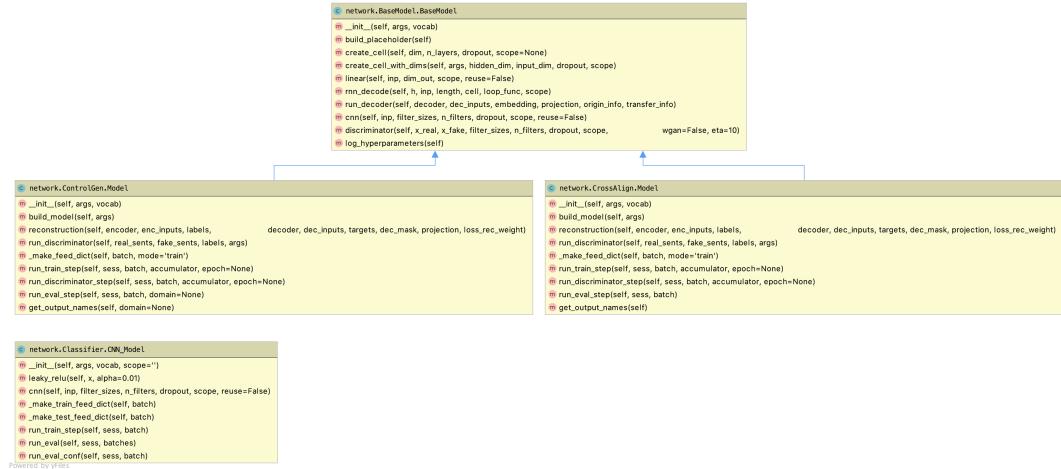


Abbildung 3.1.: Klassendiagramm der Modelle

Die Modelle wurden mit dem Datensatz, welcher in Verwendeter Datensatz (3.3) beschrieben wurde, trainiert und evaluiert. Um diesen Datensatz für das Modell lesbar zu machen, müssen die Sätze in einen Vektor aus IDs transformiert werden. Jede ID repräsentiert dabei ein Wort, und kann damit eindeutig zugeordnet werden.

Um diese Vektoren zu erstellen, muss als erstes ein Vokabular von den Daten erstellt werden. Dieses beinhaltet alle Vorkommnisse der einzelnen Wörter im Datenkorpus. Um die Grösse dieses Vokabulars einzuschränken, muss jedes Wort eine gewisse Anzahl darin vorkommen um aufgenommen zu werden, diese Anzahl an Vorkommnissen ist als Hyperparameter verfügbar.

Jedem dieser Wörter im Vokabular wird eine eindeutige ID zugeordnet um ein «Wort zu ID» und «ID zu Wort» Array zu erstellen. Diese beiden Arrays werden danach genutzt um die einzelnen Sätze in ID Vektoren zu transformieren und wieder zurück. Wenn nun ein Wort, welches das minimale Vorkommen im Datenkorpus nicht erreicht hat, transferiert werden möchte, wird diesem Wort ein Platzhalter zugeordnet, wie zum Beispiel `<unk>`. Dadurch kann es sein, dass bei der Rekonstruktion der Sätze aus den ID Vektoren solche Platzhalter vorkommen, was in Evaluierung der Ausgabesätze (6.3.3) ersichtlich ist.

3.4.2. Classifier

Der Classifier ist ein Convolutional Neural Network (CNN), welches aus mehreren Schichten mit dem gleichen Aufbau bestehen. Zuerst eine convolution Schicht gefolgt von der Leaky Relu Aktivierungsfunktion 2.2.7 und zum Schluss noch ein Max-Pooling Layer. Dieser Aufbau wird mehrmals hintereinander geschaltet. Als Verlustfunktion wird eine Softmax Funktion verwendet, da es sich bei den Daten um gelabelte Daten, welche entweder 0 oder 1 sind, handelt. Als Optimizer wird hier und im ganzem Projekt Adam verwendet. Das Ziel des Classifiers ist es ein CNN zu trainieren um zu unterscheiden ob es sich bei dem vorliegenden

Satz um einen Klexikon oder Wikipedia Satz handelt.

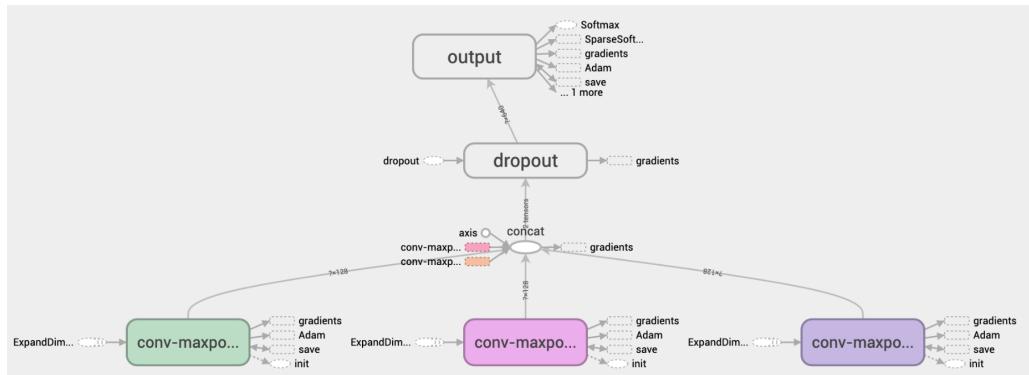


Abbildung 3.2.: Visualisierung der Classifier Architektur

Beim Classifier können mit verschiedenen Hyperparametern unterschiedliche Ergebnisse erzielt werden, je nachdem wie der Datenkorpus aufgebaut ist und von welcher Grösse dieser ist.

Hyperparameter	Interpretation
learning rate	Wie stark die Funktion sich in die Richtung des Minimums bewegen soll
dimension embedding	Dimension des verwendeten Embedding Layers
filter sizes	Wie gross die einzelnen Filterkernels der Layers sind
number of filters	Anzahl der Filters pro Convolution Schicht

Tabelle 3.1.: Hyperparameter des Classifiers

3.4.3. BaseModel

Das BaseModel wird in beiden Architektur ControlGen (3.4.4) und CrossAlign (3.4.5) verwendet, um ein objektorientierter Ansatz zu verfolgen. Dieses Modell ist nicht vollständig und kann nicht ohne weitere Implementationen trainiert werden, es dient nur der Abstraktion. In diesem Modell werden alle Hyperparameter für die geerbten Modelle zur Verfügung gestellt, sowie die wichtigsten Variablen definiert, Dropout, Batch Länge, Encoder Inputs, Decoder Inputs und noch viele weitere. Ausserdem beinhaltet das Base Modell Implementationen zu den wichtigsten Bestandteile der Netwerke. Wie zum Beispiel die Implementation einer einzelnen RNN Zelle, einer linearen Funktion, einem CNN und dem Diskriminatoren des Generative Adversarial Networks (GAN). Durch das Modell ist es möglich, die verwendeten Hyperparameter der Modelle in einem lesbaren Format zu loggen.

3.4.4. ControlGen

Das ControlGen Paper (Hu et al., 2017) schlägt ein neues generatives Modell vor, welches Variational Auto-Encoder (VAE) und Ansätze eines GAN kombiniert um die effektiven semantischen Strukturen von Daten zu erlernen. Mit dieser Kombination soll das Modell im Stande sein Sätze zu interpretieren und diese mit den gewünschten Eigenschaften zu transformieren. Dabei soll ein besonderes Augenmerk darauf gelegt werden, dass die Erzeugung der Sätze mittels Merkmale kontrolliert werden kann.

Das Modell zielt darauf ab, plausible Sätze zu erzeugen, die auf Repräsentationsvektoren basieren, die aus bestimmten semantischen Strukturen aufgebaut sind. Für die Kontrolle des Satzstiles, wird das Modell eine Dimension des Latentcodes, für den *klexikon* und *wikipedia* Stil, zugewiesen. Das Erzeugen des entsprechenden Stils, geschieht aufgrund der Angabe des entsprechenden Latentcodes. Eine solche Dimension, erfasst immer ein markantes Attribut der Daten, welches von anderen Merkmalen unabhängig ist.

Die ControlGen Architektur baut auf der Struktur eines VAE auf, welches aus einem traditionellem Auto-Encoder besteht, wobei der Latentspace regularisiert wird, um kontrolliert Daten zu erzeugen. Der Encoder des Modelles bekommt als Input eine Textsequenz x , diese wird auf den Latentspace z projiziert, welche die einzelnen Latentcodes c als Dimensionen enthalten. Die Abbildungen des Inputs auf den Latentspace z sind unstrukturierte Vektoren, welche reduzierte wichtige Merkmale eines Satzes beinhalten, um diesen wiederherzustellen. Um diese Attribute besser abzubilden und zu kontrollieren, wird der Latentspace z um eine Reihe von strukturierten Variablen y , den Latentcodes, ergänzt, von denen jede auf ein markantes und unabhängiges semantisches Merkmal abzielt, den Labels der Daten. Diese Latentcodes y werden im Netzwerk manipuliert um Daten im gewünschten Label zu erzeugen. Der Generator soll von dem kombinierten Vektor z und y abhängig sein, und generiert Daten welche die semantischen Merkmale des strukturierten Codes y erfüllen. Dieser grobe Ablauf des Modelles ist in Abbildung 3.3 anschaulich dargestellt.

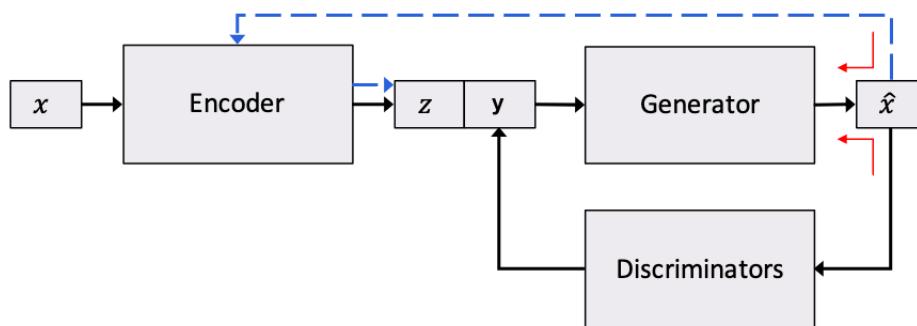


Abbildung 3.3.: Abbildung der Idee von ControlGen

Für jedes Merkmal (Label) in y gibt es einen individuellen Diskriminator, um zu messen,

wie gut die erzeugten Daten mit den gewünschten Attributen übereinstimmen. Außerdem werden diese Diskriminatoren gebraucht um die semantische Struktur der einzelnen Codes zu entschlüsseln. Dies soll wiederum den Generator dazu antreiben, bessere Ergebnisse zu erzeugen, sowie den Latentcode anzupassen.

Wenn es einen interpretierbaren Code y gibt, welcher den Output des Generators kontrolliert, muss sichergestellt werden, dass die verschiedenen Codes nicht voneinander abhängen und sich gegenseitig beeinflussen. Dies ist vor allem das Problem bei Attributen, welche nicht explizit abgebildet werden möchten. Diese Unabhängigkeit wird erreicht durch Erzwingen, dass die irrelevanten Attribute vollständig im unstrukturierten Code z abgebildet werden müssen.

Durch die Aufteilung des Latentspaces in unstrukturierten Code z und strukturierten Code y kann die Generierung der Daten vollständig kontrolliert werden. Wenn der Code y gleich bleibt und z zufällig generiert wird, können zum Beispiel Sequenzen erzeugt werden, welche die gleichen Attribute haben (z.B. *klexikon*, *wikipedia*) jedoch vom Inhalt komplett verschieden sind. Außerdem ist es durch diese Aufteilung möglich, den Style Transfer zu kontrollieren. Wenn ein Satz durch den VAE Encoder auf den unstrukturierten Code z abgebildet wird und nur der strukturierte Code (Label) y geändert wird, ist es möglich diese beiden Inputs dem Generator zu übergeben. Dieser generiert daraus ein Satz, welcher das gleiche Aussagt, jedoch sein Stil (strukturiertes Attribut) geändert wurde.

Der Latentcode (z, y) wird aus den Daten erlernt. Der unstrukturierte Code z wird durch ein unsupervised Training (d.h. die Daten müssen keine Labels enthalten) erlernt durch das konstante Feedback der Diskriminatoren. Wobei der strukturierte Code y gelabelte Daten (Supervised) braucht um trainiert zu werden, denn y soll die semantischen Attribute der Sequenzen so optimal wie möglich abbilden.

Hyperparameter

In diesem Abschnitt werden die einzelnen Hyperparameter und deren Interpretation aufgelistet. Diese können verwendet werden, um die Architektur besser an gewisse Eigenheiten der Problemstellung anzupassen, dies wird Hyperparameter Optimierung genannt.

3.4.5. CrossAlign

In der wissenschaftlichen Arbeit (Shen et al., 2017) in welcher die CrossAlign Architektur vorgestellt wird, ist der Fokus auf dem Übertragen von Stilen aufgrund von nicht-parallelen Daten. Darin wird das Trennen von Stil und Inhalt als grösste Herausforderung gesehen, welches mittels dem neu entwickelten Modell gelöst werden soll.

Um ein Attribut des Satzes, wie z.B. den Stil, unabhängig vom Inhalt zu kontrollieren, muss es möglich sein den Stil von anderen Attributen vollständig zu lösen. Das Problem ist jedoch, dass viele verschiedene Attribute miteinander interagieren, vor allem in Sprachen, wo

Hyperparameter	Interpretation
max epochs	Maximale Anzahl an Epochen die das Modell trainiert wird
batch size	Anzahl an Batches in einer Epoche
pretrain epochs	Wieviele Epochen die Reconstruction und die Diskriminatoren trainiert werden soll
learning rate	Wie stark die Funktion sich in die Richtung des Minimums bewegen soll
max length sentences	Maximale Länge der einzelnen Sätze
dropout rate	Wie gross der Dropout des Modelles ist
number of layers	Wie viele Schichten der Encoder und Decoder haben
loss rec weight	Wie stark der Reconstruction Loss gewichtet wird
trim padding	Ob die generierten und echten Sätze von der selben Länge sein sollen
word embedding	Welches Word Embedding verwendet werden soll, ob es gelernt wird als Embedding Layer oder Fasttext verwendet wird
dimension embedding	Dimension des verwendeten Embedding Layers
dimension y	Dimension des strukturierten Latentcode y
dimension z	Dimension des unstrukturierten Latentspace z
ρ (rho)	Wie stark der Generator Loss gewichtet werden soll
γ (gamma)	Wie grosser Anteil der Softmax Funktion im Decoder ist
filter sizes	Wie gross die einzelnen Filterkernels der Layers sind
number of filters	Anzahl der Filters pro Convolution Schicht

Tabelle 3.2.: Hyperparameter des ControlGen Modell

die Grammatik eine wichtige Rolle spielt.

Das Ziel dieses Modells ist es, die Verteilungsäquivalenz von Inhalten zu nutzen, um einen Satz in einem Stil auf einen stilunabhängigen Inhaltsvektor abzubilden und diesen dann in einen Satz mit dem gleichen Inhalt, aber einem anderen Stil zu dekodieren. Diese Verteilungsäquivalenz nimmt nur an, dass die Daten in den beiden unabhängigen Domänen die selbe Verteilung des Inhaltes haben.

Das Netzwerk besteht aus einem Encoder, welcher einen Satz und seinen ursprünglichen Stil als Eingabe nimmt und diesen auf eine stilunabhängige Inhaltsrepräsentation abbildet. Sowie stilabhängige Generatoren, welche die Inhaltsrepräsentation ohne Stil zu einer Repräsentation mit Stil umwandeln. Für diesen Prozess werden keine typischen VAE verwendet, da es zwingend notwendig ist, die Darstellung der latenten Inhalte reichhaltig und störungsfrei abzubilden.

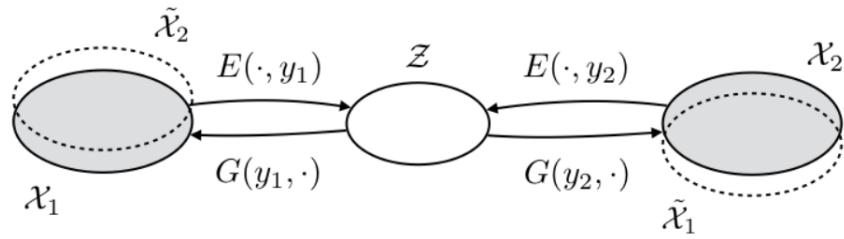


Abbildung 3.4.: Abbildung der Idee von CrossAlign

In Abbildung Abbildung der Idee von CrossAlign (3.4) ist die allgemeine Idee von der CrossAlign Arbeit ersichtlich. X_1 und X_2 sind zwei Datensätze mit unterschiedlichen Stilen y_1 und y_2 , z.B. $y_1 = \text{klexikon}$ und $y_2 = \text{wikipedia}$. Z ist der gemeinsame Latentspace. Der Encoder E projiziert eine Sequenz auf den gemeinsamen Latentspace Z und Generator G generiert den Satz zurück wenn dieser mit dem ursprünglichen Stil kombiniert wird. Wenn eine Inhaltsrepräsentation mit einem anderen Stil dem Generator übergeben wird, transferiert dieser die Sequenz in einen anderen Stil \tilde{X}_1 .

Im Datensatz X gibt es verschiedene Sequenzen $X = \{x^{(1)}, \dots, x^{(n)}\}$ welche von der gleichen bedingten Verteilung $p(x|y, z)$ stammen. Diese Verteilung hängt von der Latent Style Variable y und der Latent Content Variable z ab, wobei beide Variablen unbekannt sind. Wichtig ist, dass X , der aus verschiedenen Stilen generiert wird, unterschiedlich genug sein sollte, da sonst die Transferaufgabe zwischen den Stilen nicht gut definiert ist. Dies erscheint zwar trivial, kann aber auch bei vereinfachten Datenverteilungen nicht immer gelten.

Hyperparameter

In diesem Abschnitt werden die einzelnen Hyperparameter und deren Interpretation aufgelistet. Diese können verwendet werden, um die Architektur besser an gewisse Eigenheiten der Problemstellung anzupassen, dies wird Hyperparameter Optimierung genannt.

3.5. Training der Modelle

In diesem Kapitel, Training der Modelle (3.5), wird der Aufbau des Trainings der Modelle beschrieben. Es geht darum die Modelle aus Modelle (3.4.1) auf den in Aufbau Datensatz (5.1) aufgebauten Datensätzen zu trainieren. Die Resultate und Weiterführungen der einzelnen Trainings werden in Training der Modelle (5.2) vorgestellt.

Hyperparameter	Interpretation
max epochs	Maximale Anzahl an Epochen die das Modell trainiert wird
batch size	Anzahl an Batch in einer Epoche
learning rate	Wie stark die Funktion sich in die Richtung des Minimums bewegen soll
max length sentences	Maximale Länge der einzelnen Sätze
dropout rate	Wie gross der Dropout des Modelles ist
number of layers	Wie viele Schichten der Encoder und Decoder haben
loss rec weight	Wie stark der Reconstruction Loss gewichtet wird
trim padding	Ob die generierten und echten Sätze von der selben Länge sein sollen
word embedding	Welches Word Embedding verwendet werden soll, ob es gelernt wird als Embedding Layer oder Fasttext verwendet wird
dimension embedding	Dimension des verwendeten Embedding Layers
dimension y	Dimension des strukturierten Latentcode y
dimension z	Dimension des unstrukturierten Latentspace z
ρ (rho)	Wie stark der Generator Loss gewichtet werden soll
γ (gamma)	Wie grosser Anteil der Softmax Funktion im Decoder ist
filter sizes	Wie gross die einzelnen Filterkernels der Layers sind
number of filters	Anzahl der Filters pro Convolution Schicht

Tabelle 3.3.: Hyperparameter des CrossAlign Modell

3.5.1. Verwendete Codebasis

Für das Training der Modelle wurde eine bestehende Codebasis verwendet. Dies aufgrund fehlender Ressourcen, sowie Erfahrung, eine neue Implementation für die Modelle zu entwickeln. Als Codebasis dient das Github-Repository der Forschungsgruppe (Li et al., 2019) verwendet. Das Original Repository findet sich auf GitHub unter <https://github.com/cookielee77/DAST> (Li et al., o.D.).

Die Codebasis wurde zwecksmässig angepasst. Dabei wurden nicht verwendete Codeteile entfernt. Weiter wurden die Basis mit für das Projekt nötige Implementation angepasst. Die ursprüngliche Base beinhaltet dabei die Implementation für das Modell aus ControlGen (3.4.4) und CrossAlign (3.4.5). Die Implementationen sind mit Python (Python Software Foundation, o.D.) und Tensorflow (Google Brain Team, o.D.) umgesetzt.

Dabei wurde für die Umsetzung des Wirtschaftsprojekt folgende Erweiterungen für die Codebasis implementiert.

- Erweiterung des Projektes mit einem Prototyp um die Modelle manuell zu testen
- Erweiterung der Trainings und Evaluations Metriken mit Metric for Evaluation for Translation with Explicit Ordering (METEOR) und Word error rate (WER)

- Erweiterung der Funktionalität vom Embedding Layer der einzelnen Modellen, um Fasttext zu verwenden
- Kontrolle über die Gewichtung des Reconstruction Loss
- Kontrolle über die Anzahl der minimalen Vorkommnisse der Wörter im Vokabular
- Umfangreiche Unitests der einzelnen Modelle
- Logging der Hyperparameter
- Visualisierung der einzelnen Modelle auf dem Tensorboard

3.5.2. Verwendete Trainingsumgebung

Um ein effizientes Training der Modelle zu ermöglichen, sind entsprechend Rechenressourcen notwendig. Privat standen keine aussreichende Rechner zur Verfügung um das Training durchzuführen. Um eine möglichst flexible Gestaltung der Zeiten für das Training zu ermöglichen, wurde entschieden auf <http://vast.ai> (nachfolgend vast.ai) zurückzugreifen. Bei vast.ai können Grafikkarten für Berechnungen angemietet werden. Diese werden über ein Docker Container bereitgestellt. Der benötigte Code, sowie der Datensatz, wurden in den Container eingespielt. Anschliessend konnte für die Mietdauer die Grafikkarte verwendet werden. Dabei wurden Grafikkarten vom Typ Nvidia GeForce („GeForce“, 2019) verwendet, hauptsächlich die Modelle Titan X und 1080Ti.

3.5.3. Trainingsplanung

Damit die Trainings der Modelle einem geregeltem Ablauf folgen, wird eine Trainingsplanung durchgeführt. Wie in 5.2 eingesehen werden kann, geht es darum, die Hyperparameter der beiden Modelle anzupassen. Damit sollen die Modelle den NST auf den beiden Datensatz möglichst genau durchführen. Weiter werden zu Beginn, wie in 5.2.1 beschrieben, die Modelle auf beiden Datensätzen, «Ausgeglichen» und «Gekürzt», trainiert. Auch sollen zuerst die standard Hyperparameter der Autoren der beiden Ansätze verwendet werden. So können die Modelle ein erstes Mal auf ihre Leistung geprüft werden. Anschliessend können aufgrund der Ergebnisse der ersten Trainings die Hyperparameter angepasst werden. Um die Resultate gegeneinander vergleichbar zu halten, werden alle Trainings mit 200 Epoche trainiert.

4. Methode

Im Kapitel Methode (4) werden die verwendeten Methoden vorgestellt. Dabei geht es um die Definition der Problemstellung, sowie der Projektabgrenzung.

4.1. Vorgehensmodell

Dieses Projekt wurde mittels einem explorativen Vorgehensmodell umgesetzt, da sich dieses Modell optimal für ein Innovation- / Forschungsprojekt anwenden lässt. Außerdem wird in diesem Modell die kreative Problemlösung mittels mehreren Lernzyklen gefördert, was in einem Innovationsprojekt ein Muss ist. Dieses Modell wird in Grafik 4.1 dargestellt.

In einer ersten Phase, der Initialisierungsphase, wird das Projekt genauer untersucht sowie allfällige Fragen und Unwissenheiten geklärt. Nach der Initialisierungsphase sollte klar sein, was genau mit der Arbeit erreicht werden soll und welches die grössten Schwierigkeiten sind.

Anschliessend gibt es eine divergierende Ideenfindungsphase, in der der aktuelle Stand der Technik untersucht wird und verschiedene Ansätze erforscht werden. In dieser Phase sollten auch verschiedene Experimente durchgeführt werden, um die Machbarkeit besser zu beurteilen. Nach dieser Phase sollte klar sein, welche Ansätze in Frage kommen, um das Projekt umzusetzen.

Nach der divergierenden Phase kommt die konvergierende Ideenfindungsphase, in der das Ziel ist, aus den vorher erforschten Methoden wenige auszuwählen und genauer zu untersuchen. So kann die Anwendbarkeit auf das Projekt beurteilt werden. Nach dieser Phase sollten nur noch etwa 1-2 Ideen bestehen, welche die Problemstellung optimal angehen und einen vielversprechenden Ansatz zur Lösung dieser bieten.

Die letzte Phase in diesem Vorgehensmodell ist die Umsetzungsphase, diese beinhaltet die komplette Umsetzung der Idee mittels den erforschten Ideen aus den vorherigen Phasen. Dabei geht es vor allem darum die Problemstellung so weit wie möglich zu lösen und auch Vorschläge für die Umsetzung anderer Ideen in der gleichen Domäne zu geben. Nach der Umsetzungsphase sollte das Projekt abgeschlossen sein und ein Prototyp sowie eine klare Forschung zum jeweiligen Themengebiet ersichtlich sein.

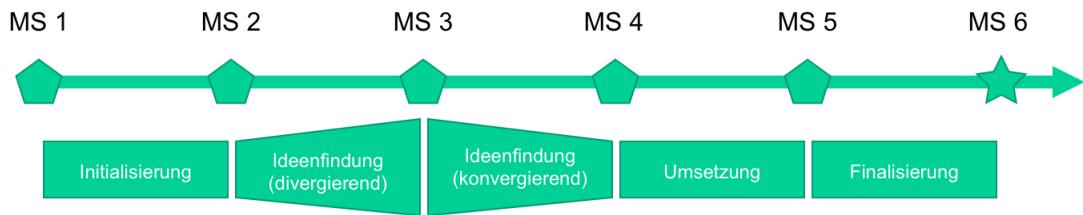


Abbildung 4.1.: Meilensteinplanung

4.1.1. Meilensteinplanung

Die Meilensteinplanung ist im explorativen Vorgehensmodell naheliegend, nach jeder abgeschlossenen Phase gibt es einen neuen Meilenstein, sowie beim Start und bei der Abgabe. Demnach gibt es insgesamt 6 Meilensteine, die erreicht werden sollen. Die Termine der Meilensteine ergeben sich aus der Zeitplanung des Herbstsemesters an der HSLU und den Rahmenbedingungen des Moduls «Wirtschaftsprojekt». Die detaillierten Berichte zu den einzelnen Meilensteinen können im Anhang eingesehen werden unter Meilensteinberichte (A).

- **M1: 26. September 2019**, Start des Projektes
- **M2: 10. Oktober 2019**, Ende Initialisierungsphase
- **M3: 21. Oktober 2019**, Ende Ideenfindungsphase divergierend
- **M4: 11. November 2019**, Ende Ideenfindungsphase konvergierend
- **M5: 9. Dezember 2019**, Ende Umsetzungsphase
- **M6: 20. Dezember 2019**, Ende Finalisierung und Abgabe Projekt

4.2. Problemstellung

Die Aufgabenstellung wird bereits in Kapitel Aufgabenstellung und Zielsetzung (1.1) ausgeführt. Für die Problemstellung soll anhand der Bausteine des Zeugnismanagers und entsprechender Sätze das Problem aufgezeigt werden.

Im Zeugnismanager können Arbeits-, sowie Zwischenzeugnisse, für Mitarbeitende erstellt werden. Der Baukasten umfasst dabei die Sprachen Deutsch, Französisch, Italienisch und Englisch. Weiter sind die Bausteine für Mitarbeiter beider Geschlechter vorhanden. Arbeits- und Zwischenzeugnis unterscheidet sich anhand der Zeitform, wobei das Arbeitszeugnis in Präteritum, das Zwischenzeugnis in Präsens verfasst wird.

Der Benutzer des Zeugnismanagers kann mithilfe von verschiedenen Bewertungskategorien, die Leistungen des zu bewertenden Mitarbeiters beurteilen. Für jede Kategorie kann

eine Bewertung zwischen *A* und *D* abgegeben werden, wobei *A* den oberen Teil der Skala markiert. Für die entsprechenden Bewertung stehen anschliessend verschiedene Textbausteine zur Auswahl.

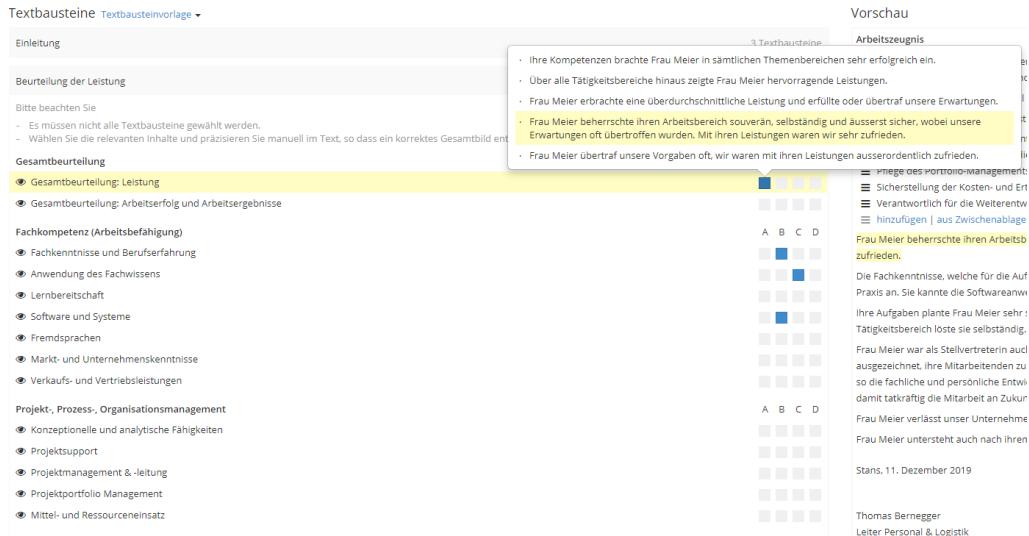


Abbildung 4.2.: Screenshot Zeugnismanager

Dabei sind die unten aufgeführten Sätze Beispiele für eine Bewertungskategorie mit der Bewertung *A*.

1. *Herr Foo schaffte die Voraussetzungen für ein leistungsförderndes Arbeitsklima und unterstützte aktiv den konstruktiven Austausch im Team.*
2. *Herr Foo schaffte die Voraussetzungen für ein konstruktives und leistungsförderndes Arbeitsklima.*
3. *Herr Foo schaffte die nötigen Voraussetzungen für ein leistungsförderndes Arbeitsklima und eine konstruktive Zusammenarbeit.*

Für tiefere Bewertungen stehen oft weniger Textbausteine zur Verfügung.

1. *Die Zusammenarbeit im Team förderte Herr Foo nicht mit dem nötigen Engagement.*

Ein beispielhaftes Arbeitszeugnis, welches mit dem Zeugnismanager erstellt wurde, ist im Anhang zu finden Beispiel Arbeitszeugnis Zeugnismanager (F). Die meisten Kunden des Auftraggebers editieren nach dem Erstellen des Zeugnisses den Text, um das holprige Lesen zu verhindern. Die Unternehmen können die Textbausteine frei editieren, um bereits beim Erstellen der Zeugnisse den Lesefluss zu fördern. Ein qualitativ hochwertiges Arbeitszeugnis, kann aufgrund des Datenschutzes nicht angehängt werden. Jedoch soll anhand einiger Beispielsätze gezeigt werden, welche Qualität angestrebt werden soll.

1. Bei der Ausübung seiner Aufgaben zeichnet er sich aus durch eine äusserst selbständige und effiziente Arbeitsweise, Beharrlichkeit sowie hohes Qualitäts- und Kostenbewusstsein.
2. Herr Foo ist Neuerungen gegenüber **offen, erkennt** Innovationsmöglichkeiten und bringt konkret umsetzbare Ideen ein.
3. Er pflegt eine offene Kommunikationskultur und informiert **zeit- und stufengerecht**.

Dabei ist auf die Verschachtlung und Referenzen in den einzeln Sätzen zu achten. In den Textbausteinen des Zeugnismanagers beschränkt sich die Verschachtlung auf Aufzählungen.

4.3. Lösungsansatz

Für die Lösung der Aufgabenstellung könnten verschiedene Ansätze verwendet werden, siehe Empfehlung im Bezug auf den Zeugnismanager (7.3). Die Arbeitszeugnis des Zeugnismanagers zeugen bereits von guter Qualität. Sie vermögen es jedoch nicht mit der Qualität von individuell geschriebener Zeugnisse mitzuhalten.

Da der Zeugnismanager wie erwähnt bereits ein auf dem Markt konkurrenzfähiges Produkt ist, käme die Umgestaltung des Manager oder der Textbausteine einer Weiterentwicklung gleich. Dies ist nach Absprache mit dem Auftraggeber und des Betreuer nicht das Ziel dieser Arbeit. Es soll daher ein neuer Ansatz gesucht werden, welcher ein Alleinstellungsmerkmal für die Marktlösung des Auftraggebers sein kann. Weiter soll der Lösungsansatz im Bereich des maschinellen Lernen angesiedelt sein, siehe Aufgabenstellung Wirtschaftsprojekt (E).

Für das Errechnen der Modelle ist ein grosser Datensatz nötig. Die Daten sollten dabei aus dem entsprechenden Anwendungsgebiet stammen. Für die Umsetzung solcher Lösungen sind im Falle für Arbeitszeugnisse bedauerlicherweise nicht genügend Daten vorhanden. Für die Arbeit soll daher ein Datensatz aufgebaut werden, welches das Problem abbildet. Anhand des Datensatzes sollen Modelle für die Problemlösung evaluiert werden. Die Modelle sollen dabei den Style Transfer vornehmen. Daher wurde entschieden einen Datensatz aufzubauen, welcher das Problem abbildet. Wie in Problemdefinition (3.1) beschrieben geht es darum den Stil eines Satzes zu ändern. Es wird nun nach Daten gesucht, die aus dem selben Themengebiete stammen und verschiedene Schreibstile aufweisen. Dabei werden die Stilllabels *holprig* und *flssig* durch Stilllabels des verwendeten Datensatzes ersetzt. Der Datensatz sowie die neuen Stilllabel werden in Verwendeter Datensatz (3.3) beschrieben.

4.4. Evaluierung

Die Evaluierung der Arbeit ist einer der wichtigsten Punkte des Projektes, denn dadurch soll festgestellt werden ob die verfolgten Ansätze zum Lösen des Problemstellungen vielversprechend sind und diese weiter verfolgt werden sollen.

Eine weiter Methode zur Überprüfung, ist die Messung der Verteilung des Datensatzes vor und nach dem Transfer. Damit soll überprüft werden ob sich die Verteilung nach dem Transfer deutlich von der Vorherigen unterscheidet. Die neue Verteilung sollte aufzeigen, dass die Sätze vor allem länger werden. Jedoch kann die Komplexität der einzelnen Sätze nur schlecht statistisch gemessen werden und muss daher manuell gemacht werden.

Weiter sollen die Sätze manuell überprüft werden. Dabei werden diese vor und nach dem Transfer genauer betrachtet. So kann abgeschätzt werden ob der Satz korrekt in die neue Domäne gebracht wurde. Ausserdem soll auf die Grammatik der generierten Sätze geachtet werden um zu beurteilen ob diese korrekt erlernt wurde.

4.5. Testing

Um die Funktionsfähigkeit der Modelle sicherzustellen müssen diese automatisiert und nachvollziehbar getestet werden können. Um diese Art von Testing zu gewährleisten, werden Unit Tests („Unit Test Wikipedia“, o.D.) verwendet. Diese können automatisiert ausgeführt werden und testen immer den gleichen Ablauf mit vordefinierten Parametern. Unit Tests sollten immer nur einen Fall testen, was in diesem Projekt auch so umgesetzt wurde.

Jedes verwendete Modell wird separat mit den gleichen Testfällen auf Richtigkeit überprüft. Die Modelle werden auf vier solcher Testcases überprüft:

1. Testen ob die trainierbaren Variablen sich nach einem Trainingsschritt verändern
2. Testen ob der Loss nach einem Trainingsschritt nicht Null ist
3. Testen ob der Diskriminator separat von dem Generator trainiert werden kann
4. Testen ob das Modell einen Output liefert

Durch diese vier Testfälle kann sichergestellt werden, dass das Modell trainiert werden kann und keinen Fehler in der Architektur vorhanden sind und die einzelnen Schichten korrekt miteinander verbunden sind. Ausserdem kann durch Testcase 3 vergewissert werden, dass bei einem Generative Adversarial Networks (GAN) die beiden Komponenten separat voneinander trainiert werden können. Jeder einzelne dieser Testfälle muss korrekt durchlaufen nach einer Änderung des Modells und am Schluss des Projektes.

4.6. Aufbau Projekt

Das Projekt wurde in vier verschiedene Git Repositories unterteilt, um die einzelnen Schritte der Umsetzung sinnvoll zu trennen. Diese werden im folgenden Abschnitt genauer beschrieben.

4.6.1. wipro-doc

In diesem Repository befindet sich die ganze Dokumentation des Projektes, sowie das Verzeichnis welches für die Recherche der Arbeit gebraucht wird. Dieses Repository wird immer wieder bearbeitet und beinhaltet schlussendlich auch die finale Dokumentation, welche abgegeben wird.

Link zum Git Repository: [wipro-doc](#)

```
wipro-doc
└── assets ... (assets for the project)
└── documentation ... (latex file for final documentation)
└── research ... (markdown file for research)
└── study-doc ... (origin of study-doc)
```

4.6.2. wipro-data

Wird verwendet, um die ganzen Daten, welche im Projekt genutzt werden, an einem zentralen Ort zu speichern. Darin sind mehrere Datensätze vorhanden, welche im Repository aufbereitet und statistisch analysiert werden. Dieses Git Projekt ist ein Git LFS Repository, da es sich um grosse Files handelt.

Link zum Git Repository: [wipro-data](#)

```
wipro-data
└── utils ... (util classes for cleaning, processing the data)
└── cleaned ... (cleaned data)
└── crawled ... (crawled data)
└── notebooks ... (jupyter notebooks for prepare, process and analyze data)
└── processed ... (processed data)
└── statistics ... (statistic data)
```

4.6.3. wipro-source

Dieses Projekt beinhaltet die ganze Codebasis dieser Projektarbeit. Es ist dafür da, um verschiedene Modelle zu testen und zu evaluieren. Darin befinden sich auch Datensätze von *wipro-data*, um die entsprechenden Modelle damit zu trainieren. In diesem Repository befindet sich ausserdem der Prototyp der Arbeit.

Link zum Git Repository: [wipro-source](#)

```
wipro-source
└── data ... (data to train the models)
└── dataloader ... (loads data for the models to process)
└── network ... (architecture of the models)
└── notebooks ... (jupyter notebooks to analyze models)
└── save-model ... (saved models)
```

```
wipro-logs
└── test ... (unit tests for the networks)
└── prototype.py ... (prototype of the project)
```

4.6.4. wipro-logs

In diesem Git Projekt befinden sich Log Files, Tensorboard Daten und Testsätze aller trainierten Modelle. Dies wird zum Hyperparametertuning sowie zur Evaluierung gebraucht. In diesem Repository kann Tensorboard gestartet werden und somit die entsprechenden Graphen eingesehen werden. Zu jedem trainierten Modell gibt es im *logs* Ordner ein entsprechende Verzeichnis mit den Daten.

Link zum Git Repository: [wipro-logs](#)

```
wipro-logs
├── log-archive ... (archive of the first models trained logs)
├── logs ... (logs of the trained models)
│   ├── Classifier ... (logs of the classifiers)
│   ├── ControlGen ... (logs of the ControlGen models)
│   └── CrossAlign ... (logs of the CrossAlign models)
└── Tensorboard.ipynb ... (jupyter notebook to start tensorboard)
```

5. Realisierung

In diesem Kapitel wird die ganze Realisierung des Projektes beschrieben, einschliesslich der aufgetretenen Schwierigkeiten und Einschränkungen. Dabei geht es darum, die Umsetzung der definierten Ziele aufzuzeigen.

5.1. Aufbau Datensatz

Wie in Kaptiel Abschliessende Problemdefinition (3.2) beschrieben, wird ein Datensatz mit Texten aufgebaut, welchem dem Label *erwachsen*, sowie dem Label *kindlich* zugewiesen werden können. Die Quellen der Texte sind Online Lexikas. Wikipedia dient dabei als Quelle für Sätze des Labels *erwachsen*, analog dient das Klexikon als Quelle für Sätze des Labels *kindlich*. Für das Zusammentragen der Texte wurde ein Crawler realisiert, welcher die Artikel aus dem Klexikon und Wikipedia zusammenträgt.

5.1.1. Reinigung des Datensatz

Die Artikel wurden entsprechend gereinigt und die einzelnen Sätze aus den Artikeln extrahiert. Dabei wurde auf das Framework Natural Language Toolkit (NLTK) zurückgegriffen. Die Pipeline für die Reinigung und der Extrahierung wurde dabei wie folgt aufgebaut.

1. Reinigung der Strings durch ersetzen von Buchstaben mit Akzent zeichen wie z.B. è, à, â, ê.
2. Einheitliche Verwendung von Anführungszeichen.
3. Einfügen von Leerschläge bei Kommas (,), Gedankenstrichen (-), Schrägstrichen (/), Doppelpunkt (:), Strichpunkt (;) sowie an Satzenden.
4. Tokenizierung der Artikel zu Sätzen mithilfe von NLTK. Dabei wurden nur Sätze erlaubt, welche gewisse Charaktere enthalten.
5. Anpassung der Grossschreibung der Satzanfänge mithilfe von TreeTagger. Dabei wurde die Grossschreibung nur beibehalten, falls das Wort ein Nomen oder Eigennamen ist.

Weiter wurden Daten für die statistische Analyse der Datenquelle sowie der Sätze mitaufgezeichnet. Dies, um in einem nächsten Schritt, den Datensatz weiter zu bereinigen. Für die Datenquellen wurden folgende Daten aufgezeichnet.

1. Anzahl an Artikel aus der Datenquelle

2. Anzahl Sätze aus der Datenquelle
3. Anzahl Wörter aus der Datenquelle

Für die Analyse der einzelnen Sätze wurde während der Reinigung der Sätze folgendes mitabgespeichert.

1. Datenquelle des Satzes
2. Artikel des Satzes
3. Anzahl Wörter im Satz

So entstand aus den Artikeln der einzelnen Datenquelle ein gemeinsamer Datensatz, abgespeichert als CSV, welcher folgende Struktur aufweist.

```
source article length sentence
klexikon Aal 9 Aale sind Fische , die wie Schlangen aussehen .
klexikon Aal 10 ihre Körper sind sehr lang , schlank und beweglich .
klexikon Aal 13 sie haben eher kleine Flossen , die wie Bänder am Körper anliegen .
wikipedia Reh 7 die Fluchtdistanz von Feldrehen ist hoch .
wikipedia Reh 8 diese hohe Fluchtdistanz kompensiert die fehlende Deckung .
```

5.1.2. Statistische Analyse des Datensatz

In einem weiteren Schritt wurde der Datensatz statistisch analysiert. Dies, um den Datensatz weiter zu reinigen und zu normalisieren. Dabei soll zuerst die Zählung der Artikel, Sätze, sowie der Wörter aufgezeigt werden.

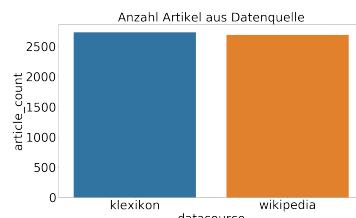


Abbildung 5.1.: Anz. Artikel im Datensatz

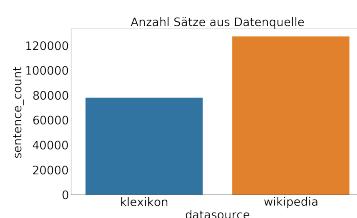


Abbildung 5.2.: Anz. Sätze im Datensatz

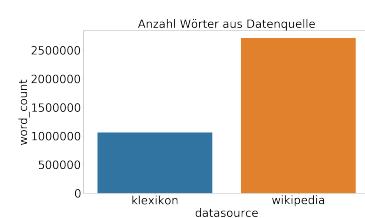


Abbildung 5.3.: Anz. Wörter im Datensatz

Datenquelle	Anz. Artikel	Anz. Sätze	Anz. Wörter
Klexikon	2'733	77'924	1'058'623
Wikipedia	2'689	127'177	2'710'452

Tabelle 5.1.: Anzahl der Artikel, Sätze und Wörter aus Wikipedia und Klexikon.

Wie angenommen sind die Artikel von Wikipedia umfangreicher als die vom Klexikon. Die Artikel beinhalten mehr Sätze als aus dem Lexikon. Entsprechend auch mehr Wörter. Dies kann auch anhand der Verteilung für Sätze pro Artikel aufgezeigt werden, siehe 5.4 und 5.5.

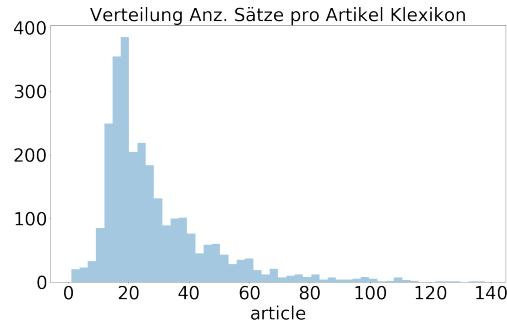


Abbildung 5.4.: Verteilung Sätze pro Artikel Klexikon

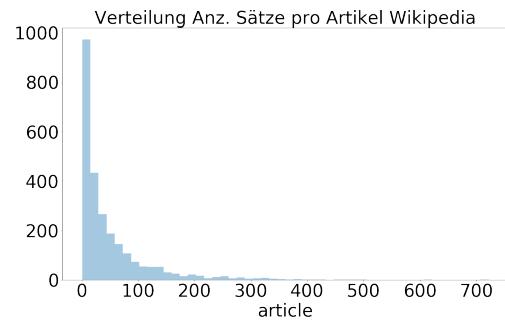


Abbildung 5.5.: Verteilung Sätze pro Artikel Wikipedia

Datenquelle	25% Quartil	Median	75% Quartil	Mean	Standardabweichung
Klexikon	17	23	35	28.6	18.3
Wikipedia	8	24	63	49.9	70.0

Tabelle 5.2.: Verteilung der Anzahl Sätze pro Artikel für Klexikon und Wikipedia

Für die Problemstellung ist jedoch die Verteilung der Anzahl Wörter in einem Satz relevant. Daher wurden Sätze mit weniger als 4 Wörter aus dem Datensatz entfernt. Dies, da Sätze mit 3 oder weniger Wörter eine zu geringe Relevanz für diverse Stile aufweisen. Anhand der Grafiken 5.6 und 5.7 wird gezeigt, dass die Sätze aus Wikipedia länger gestaltet sind als die Sätze aus dem Klexikon. Hier wird auch wie in Abschliessende Problemdefinition (3.2) beschrieben, das Ziel des Style Transfer legitimiert.

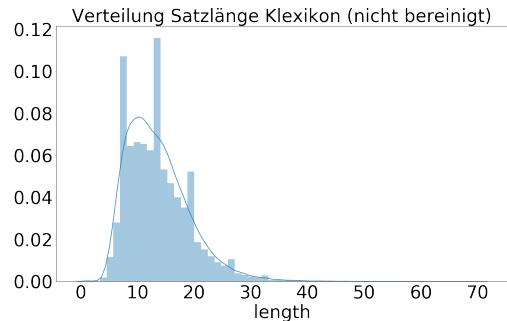


Abbildung 5.6.: Verteilung Satzlänge Klexikon nicht bereinigt

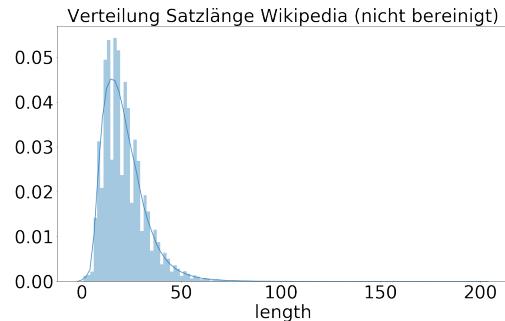


Abbildung 5.7.: Verteilung Satzlänge Wikipedia nicht bereinigt

Datenquelle	25% Quartil	Median	75% Quartil	Mean	Standardabweichung
Klexikon	9	13	17	13.6	5.5
Wikipedia	14	19	27	21.4	10.8

Tabelle 5.3.: Verteilung Anzahl Wörter Klexikon und Wikipedia

Mit Hilfe des Interquartilabstand (eng. Inter-quantil range) (IQR) wurden die Ausreisser aus dem Datensatz entfernt. Dabei gibt es nur Aussreisser im oberen Bereich, da die untere Grenze unter 0 liegt. Mit der Entfernung der zu kurzen Sätze, unter 4 Wörter, und Sätze über dem oberen IQR repräsentiert der Datenstanz weiterhin das Ziel der Problemdefinition (3.1). Dieser Datenbestand bildet die Basis für weitere Datensätze, siehe Aufbau Datensatz «Ausgeglichen» (5.1.2) und Aufbau Datensatz «Gekürzt» (5.1.2).

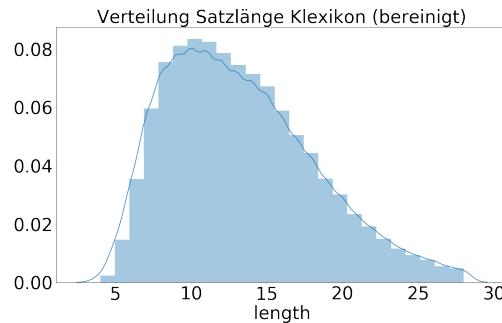


Abbildung 5.8.: Verteilung Satzlänge Klexikon bereinigt

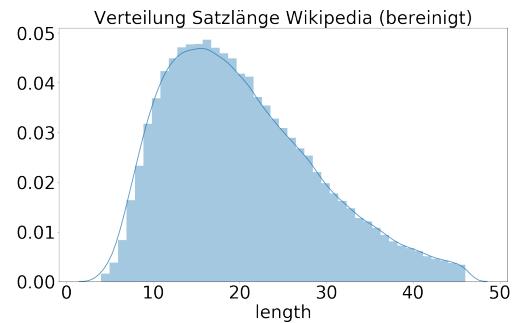


Abbildung 5.9.: Verteilung Satzlänge Wikipedia bereinigt

Datenquelle	25% Quartil	Median	75% Quartil	Mean	Standardabweichung
Klexikon	9	13	16	13.3	5.0
Wikipedia	13	19	26	20.3	8.8

Tabelle 5.4.: Verteilung Anzahl Wörter Klexikon und Wikipedia ohne Ausreisser

Aufbau Datensatz «Ausgeglichen»

Als nächstes wurden die Grösse, die Anzahl an Sätzen, der beide Datenquelle aneinander angepasst. Der Wikipedia Datensatz beinhaltet ca. die doppelte Anzahl an Sätzen. Daher wurde jeder zweite Satz aus dem Wikipedia Datensatz entfernt. Der so neu entstandene Datensatz wird wikishort genannt. In den Abbildungen 5.10, 5.11 und 5.12 wird die neue Grösse der Datensätze abgebildet.

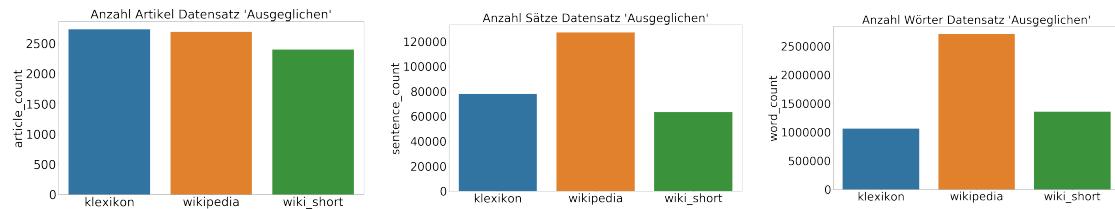


Abbildung 5.10.: Anz. Artikel im bereinigten Datensatz

Abbildung 5.11.: Anz. Sätze im bereinigten Datensatz

Abbildung 5.12.: Anz. Wörter im bereinigten Datensatz

Datenquelle	Anz. Artikel	Anz. Sätze	Anz. Wörter
Klexikon	2'733	77'924	1'058'623
Wikipedia	2'398	61'543	1'249'414

Tabelle 5.5.: Anzahl Sätze, Sätze und Wörter Klexikon, Wikipedia und Wikipedia «Ausgeglichen»

Da die Anzahl an Sätzen für beide Datenquellen ausgeglichen wurden, erhält dieser Datensatz die Bezeichnung «Ausgeglichen». Dies, da die Anzahl der Sätze für beide Datenquellen ausgeglichen wurden. Dabei bleibt die Verteilung der Satzlänge auf beiden Datenquellen gleich, welches wie im Kapitel 3.2 beschrieben die Charakteristik der beiden Stile, *erwachsen* und *kindlich*, beschreibt. In Abbildung 5.13 sind die beiden Verteilungen der Datenquellen noch einmal übereinander aufgezeigt.

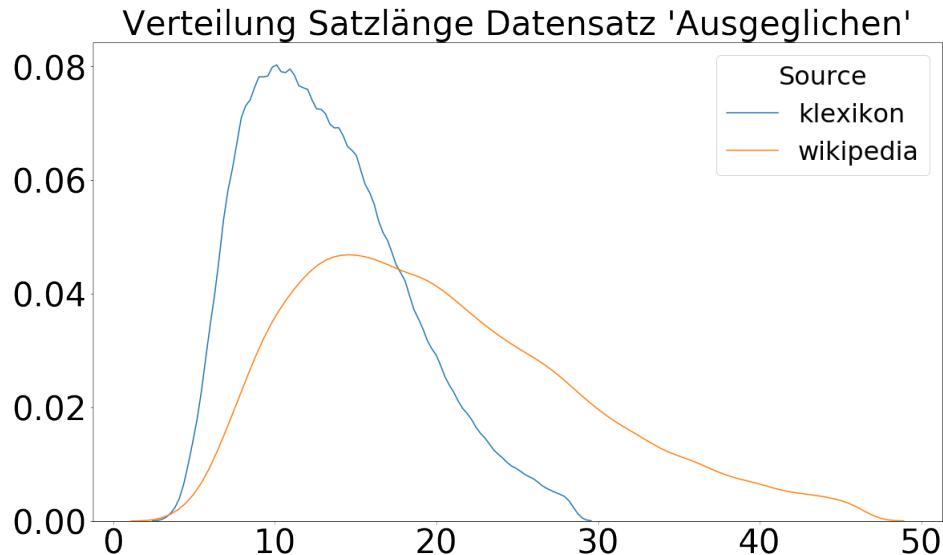


Abbildung 5.13.: Verteilung Satzlänge Datensatz «Ausgeglichen»

Aufbau Datensatz «Gekürzt»

Auf der Datenbasis des Datensatz aus Statistische Analyse des Datensatz (5.1.2) wurde neben dem Datensatz «Ausgeglichen», siehe Aufbau Datensatz «Ausgeglichen» (5.1.2), ein zweiter Datensatz erstellt. In diesem Datensatz sollen die beiden Stile klarer differenziert werden. Im Datensatz «Ausgeglichen» gibt es Sätze, welche sich in der Satzlänge überschneiden. Dies soll in diesem Datensatz nicht der Fall sein. Dies bedeutet für beide Datenquellen werden nur die Sätze verwendet die in einen gewissen Bereich fallen. Dieser Bereich soll die Stile besser definieren. Es wurde entschieden aus der Datenquelle Klexikon nur Sätze zu verwenden, welche eine Satzlänge zwischen 10 und 20 Wörter aufweisen. Für die Datenquelle Wikipedia wurden nur Sätze, welche eine Länge zwischen 20 und 30 Wörter enthalten. Dadurch soll ein Datensatz entstehen, in welchem die beiden Stile, welche anhand der Länge definiert werden, klarer differenziert werden. Dieser Datensatz kann hilfreich sein, falls die Modelle die Stile aus dem Datensatz «Ausgeglichen» nicht extrahieren kann, da die Satzlängen sich aufgrund Überschneidung zuwenig unterscheiden. Die Verteilung der beiden Datenquelle im Datensatz «Gekürzt» sind in den Abbildungen 5.14 und 5.15 zu sehen.

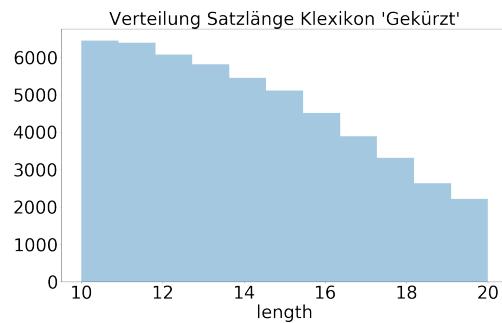


Abbildung 5.14.: Verteilung Satzlänge Klexikon «Gekürzt»

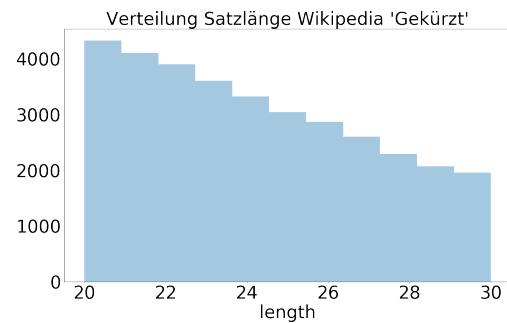


Abbildung 5.15.: Verteilung Satzlänge Wikipedia «Gekürzt»

5.2. Training der Modelle

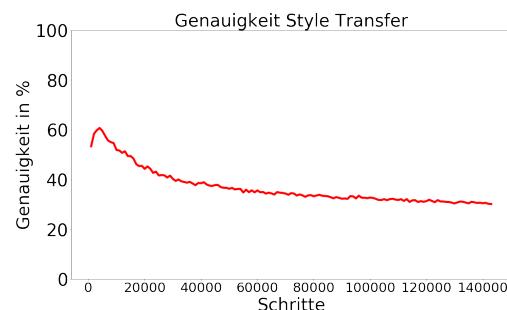
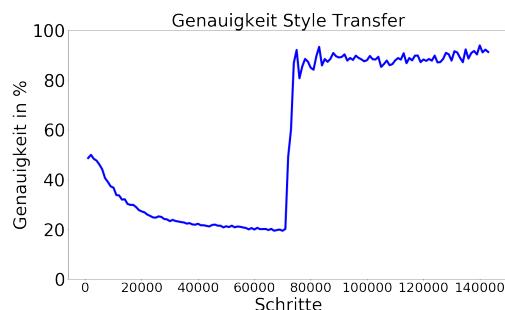
Aufbauend auf den Erwähnungen in Training der Modelle (3.5) werden hier die Resultate der Modelle nach dem Training vorgestellt. Auch sollen die Entscheidungen für die Änderungen der Hyperparameter ausgeführt werden.

In der linken Spalte (blaue Farbe), finden sich die jeweils Resultate des Models ControlGen. In der rechten Spalte, in Rot, sind die Resultate der CrossAlign Modelle abgebildet. Es werden drei Graphen dargestellt. Alle Graphen zeigen die Resultate aus dem Validations Schritt des Trainings. Die erste Grafik zeigt die Genauigkeit des durchgeföhrten Style Transfer. Die zweite zeigt den Verlauf der Entwicklung des Bilingual evaluation understudy (BLEU) Scores für die generierten Sätze. In der dritten Grafik wird die Verlustfunktion über die Trainingszeit dargestellt. Zum Schluss jedes Trainings wird ein Fazit aus den Hyperparameter Einstellungen gezogen. Die Gestaltung des nächsten Trainingsdurchlauf wurde aufgrund der Resultate der Trainings durchgeführt.

5.2.1. Standard Hyperparameter

Zuerst wurden die beiden Modelle ControlGen und CrossAlign mit den vorgeschlagenen Hyperparameter trainiert. Auch wurden beide Modelle auf beiden Datensätzen «Ausgeglichen» und «Gekürzt» trainiert. Die Hyperparameter für die entsprechenden Modelle und Graphen sind in Tabelle Training der Modelle mit standard Hyperparameter (5.6) aufgelistet und sind im Abschnitt 3.4.1 beschrieben.

Datensatz «Ausgeglichen»



Hyperparameter	Werte			
network	CrossAlign	CrossAlign	ControlGen	ControlGen
dataset	trimmed	equalized	trimmed	equalized
max epochs	200	200	200	200
pretrain epochs	100	100	100	100
batch size	64	64	64	64
learning rate	0.0005	0.0005	0.0005	0.0005
max len. sentences	50	50	50	50
dropout rate	0.5	0.5	0.5	0.5
number of layers	1	1	1	1
loss rec weight	1	1	1	1
trim padding	false	false	false	false
word min. occur	3	3	3	3
word embedding	embed. layer	embed.layer	embed. layer	embed. layer
dimension embedding	100	100	100	100
dimension y	200	200	200	200
dimension z	500	500	500	500
ρ (rho)	1	1	1	1
γ (gamma)	0.1	0.1	0.1	0.1
filter sizes	1,2,3,4,5	1,2,3,4,5	1,2,3,4,5	1,2,3,4,5
number of filters	128	128	128	128

Tabelle 5.6.: Training der Modelle mit standard Hyperparameter

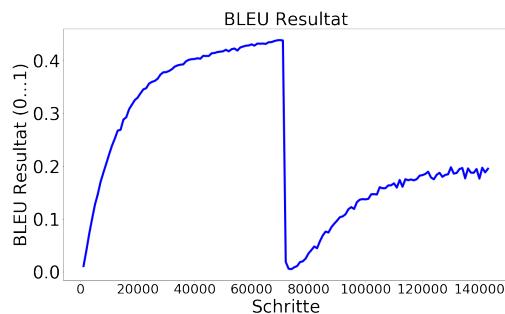


Abbildung 5.18.: BLEU Score Control-Gen Default «Ausgeglichen»

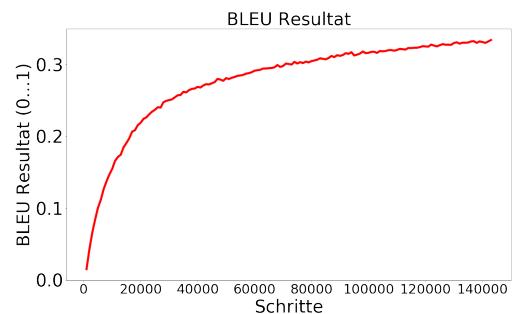


Abbildung 5.19.: BLEU Score Transfer CrossAlign Default «Ausgeglichen»

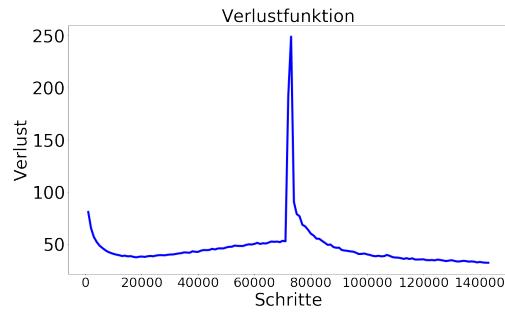


Abbildung 5.20.: Verlustfunktion ControlGen Default «Ausgeglichen»

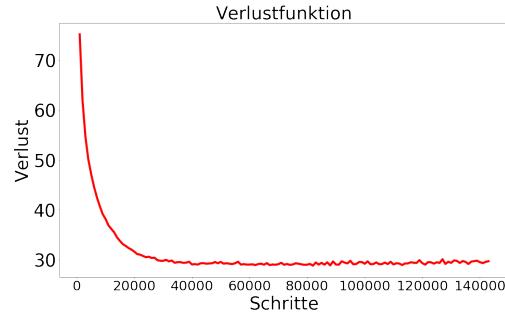


Abbildung 5.21.: Verlustfunktion CrossAlign Default «Ausgeglichen»

Datensatz «Gekürzt»

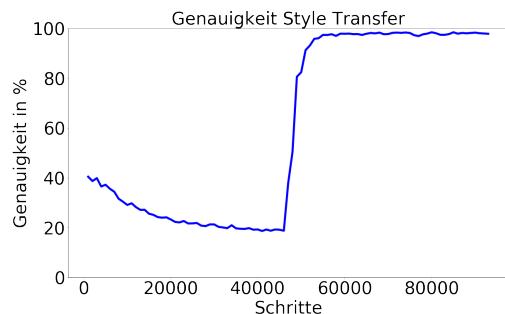


Abbildung 5.22.: Genauigkeit ControlGen Transfer Default «Gekürzt»

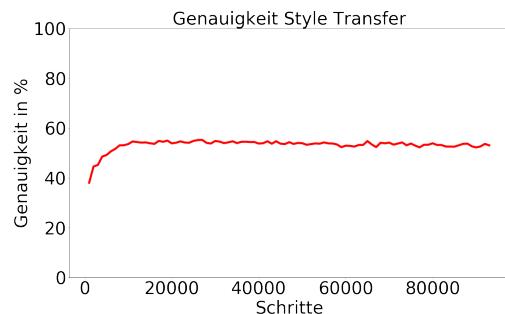


Abbildung 5.23.: Genauigkeit CrossAlign Transfer Default «Gekürzt»

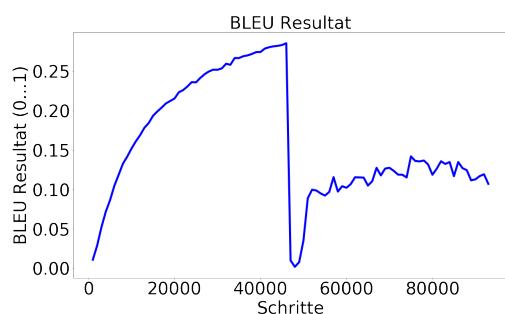


Abbildung 5.24.: BLEU Score ControlGen Default «Gekürzt»

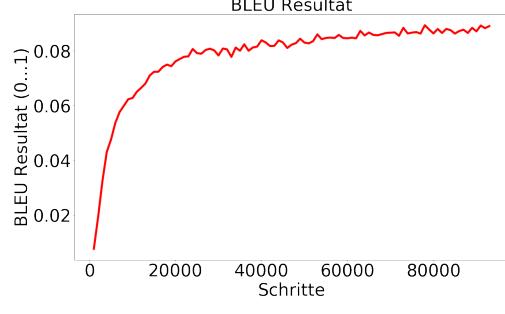


Abbildung 5.25.: BLEU Score Transfer CrossAlign Default «Gekürzt»

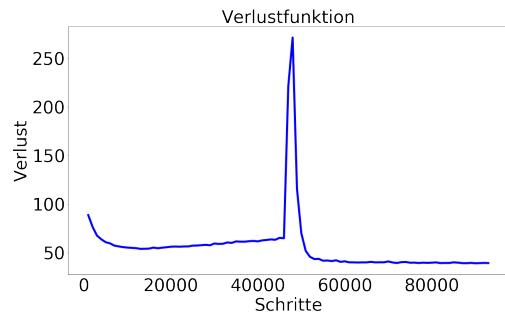


Abbildung 5.26.: Verlustfunktion Control-Gen Default «Gekürzt»

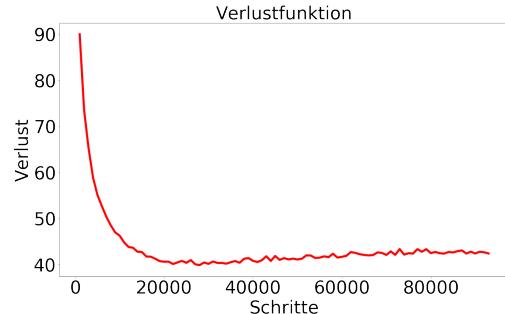


Abbildung 5.27.: Verlustfunktion CrossAlign Default «Gekürzt»

Fazit der Hyperparameter Einstellungen

Wie in Abbildung 5.17 und 5.23 ersichtlich wird kann der Style Transfer mit dem Modell CrossAlign nicht richtig durchgeführt werden. Dies ist aufgrund der sich nicht ändernden Genauigkeit des Transfers zu erklären. Obwohl der BLEU Score für den Datensatz «Ausgeglichen» höher ausfällt als für den Datensatz «Gekürzt», siehe 5.18, 5.19, 5.24, 5.25. Aufgrund der transferierten Sätze, siehe Repository wipro-logs (4.6.4), ist ersichtlich, dass die Länge der Sätze dabei jedoch nicht ändert. Da die Satzlängen sich nicht ändern kann auch der hohe BLEU Score erklärt werden, siehe BLEU (2.2.9). Dies bestätigt Teilweise die Vermutung, der sich zu stark überschneidenden Sätze, siehe Aufbau Datensatz (5.1). Daher wurde nach der Sichtung der transferierten Sätze entschieden, die weiteren Trainings nur noch auf dem Datensatz «Gekürzt» durchzuführen.

Die Verlustfunktion, 5.20 und 5.21, setzt sich aus der Verlustfunktion für die Diskriminatoren und Generatoren zusammen, siehe Repository Generative Adversarial Networks (2.2.6). In Absprache mit der Betreuungsperson wurden daher für den nächsten Trainingsdurchlauf entschieden, eine gewichtete Funktion auszuprobieren. Dies, da der Verlust des Transfers einen grossen Einfluss auf den gesamt Verlust hat.

5.2.2. Standard Hyperparameter mit gewichteter Verlustfunktion

In diesem Trainingsdurchlauf wurde die gewichtete Verlustfunktion für den Transfer Verlust eingefügt. Die Hyperparameter wurde auf den Standardwerten belassen, dies um einen Vergleich zum Training ohne gewichtete Verlustfunktionen zu erhalten. Diese Einstellungen wurden nur noch auf dem Datensatz «Gekürzt» durchgeführt. Die Hyperparameter für die entsprechenden Modelle und Graphen sind in Tabelle Training der Modelle mit gewichteter Transfer Lossfunktion Hyperparameter (5.7) aufgelistet und sind im Abschnitt 3.4.1 beschrieben.

Hyperparameter	Werte	
network	CrossAlign	ControlGen
dataset	trimmed	trimmed
max epochs	200	200
pretrain epochs	100	100
batch size	64	64
learning rate	0.0005	0.0005
max len. sentences	50	50
dropout rate	0.5	0.5
number of layers	1	1
loss rec weight	0.5	0.5
trim padding	false	false
word min. occur	3	3
word embedding	embed. layer	embed. layer
dimension embedding	100	100
dimension y	200	200
dimension z	500	500
ρ (rho)	1	1
γ (gamma)	0.1	0.1
filter sizes	1,2,3,4,5	1,2,3,4,5
number of filters	128	128

Tabelle 5.7.: Training der Modelle mit gewichteter Transfer Lossfunktion Hyperparameter

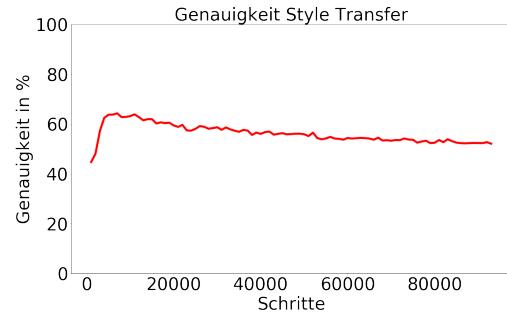
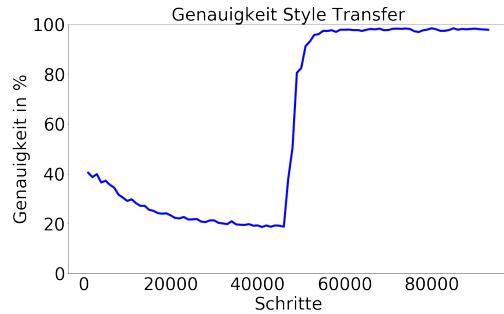


Abbildung 5.28.: Genauigkeit Transfer ControlGen Gewichtet «Gekürzt»

Abbildung 5.29.: Genauigkeit Transfer CrossAlign Gewichtet «Gekürzt»

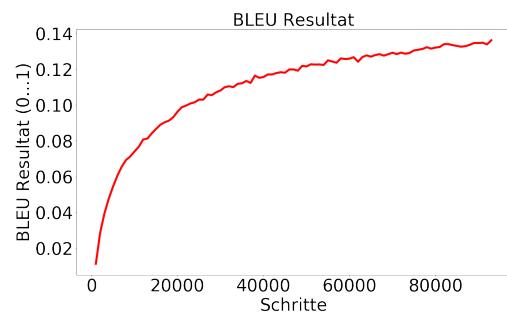
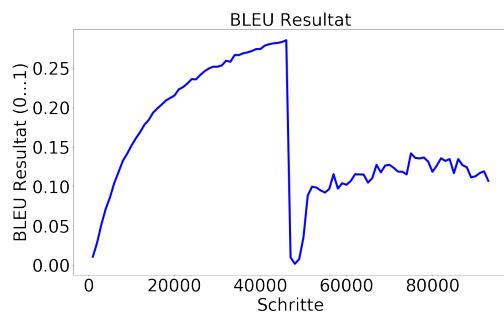


Abbildung 5.30.: BLEU Score ControlGen Gewichtet «Gekürzt»

Abbildung 5.31.: BLEU Score Transfer CrossAlign Gewichtet «Gekürzt»

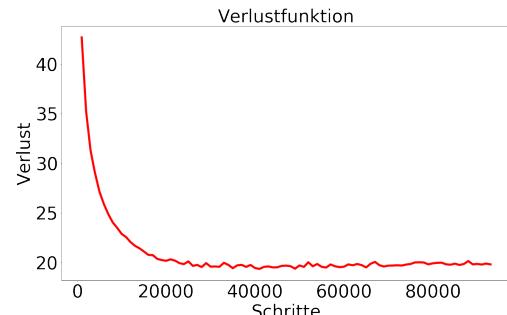
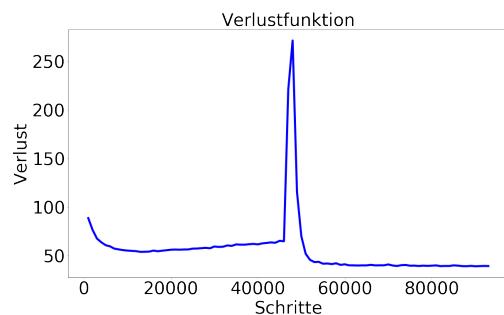


Abbildung 5.32.: Verlustfunktion ControlGen Gewichtet «Gekürzt»

Abbildung 5.33.: Verlustfunktion CrossAlign Gewichtet «Gekürzt»

Fazit der Hyperparameter Einstellungen

Die Resultate des Trainings Standard Hyperparameter mit gewichteter Verlustfunktion (5.2.2) sind nahezu identisch mit Standard Hyperparameter (5.2.1). Daraus kann die Verbesserung der Modelle mit einer Gewichtung des Transfer Verlust nicht erreicht werden. Weiter wurde nach diesem Trainingsdurchlauf entschieden, weitere Trainings nur noch auf dem Control-Gen Modell durchzuführen. Dies ist aufgrund der sich nicht ändernden Genauigkeit des Transfers. In 5.28 und 5.29 ist ersichtlich dass, das Modell CrossAlign den Style Transfer nicht wie gewünscht durchführt.

5.2.3. ControlGen mit grösseren Dimensionen

Da der BLEU Score für die ControlGen Modelle niedrig ausfällt, wurde entschieden, die Dimension für das Word Embedding, sowie den Kontext und Style Raum zu vergrössern. Dadurch hat das Modell mehr Variablen für das Training zur Verfügung. Damit das Modell trainiert werden kann ist mehr Random Access Memory (RAM) auf der Grafikkarte nötig. Daher wurde für dieses Training eine Grafikkarte mit 32GB RAM, NVIDIA Tesla 100V, verwendet. Die Hyperparameter für die entsprechenden Modelle und Graphen sind in Tabelle Training der Modelle mit grösseren Embedding Hyperparametern (5.8) aufgelistet und sind im Abschnitt 3.4.1 beschrieben.

Hyperparameter	Werte
network	ControlGen
dataset	trimmed
max epochs	200
pretrain epochs	100
batch size	64
learning rate	0.0005
max len. sentences	50
dropout rate	0.5
number of layers	1
loss rec weight	1
trim padding	false
word min. occur	3
word embedding	embed. layer
dimension embedding	300
dimension y	300
dimension z	900
ρ (rho)	1
γ (gamma)	0.1
filter sizes	1,2,3,4,5
number of filters	128

Tabelle 5.8.: Training der Modelle mit grösseren Embedding Hyperparametern

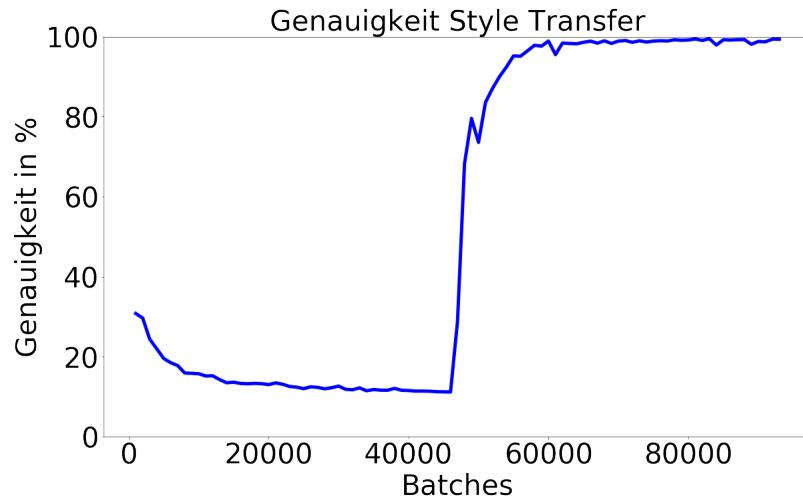


Abbildung 5.34.: Genauigkeit Transfer ControlGen Dimensionen «Gekürzt»

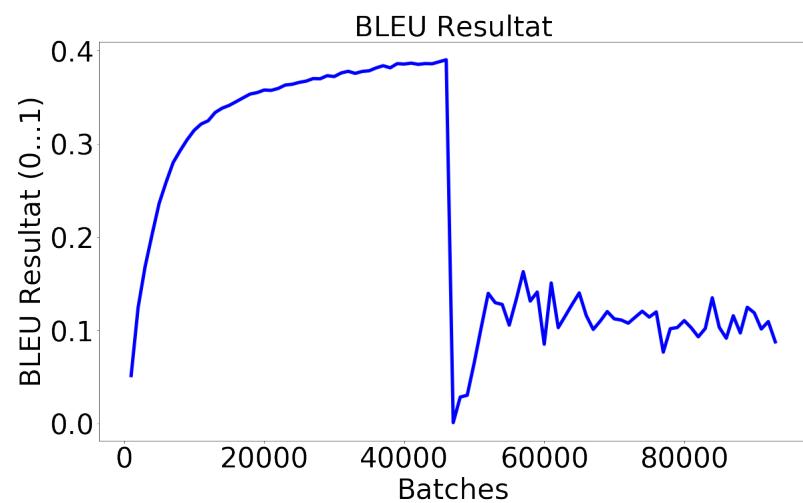


Abbildung 5.35.: BLEU Score ControlGen Dimensionen «Gekürzt»

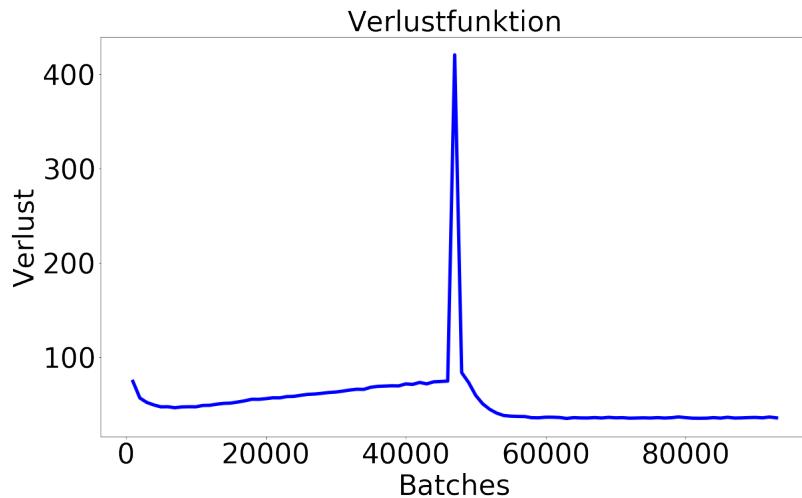


Abbildung 5.36.: Verlustfunktion ControlGen Dimensionen «Gekürzt»

Fazit der Hyperparameter Einstellungen

Wie erwartet, konnte der BLEU Score mit einem grösseren Raum verbessert werden. Dies jedoch nur in der Trainingsphase des Vortrainings, siehe ControlGen (3.4.4). Die transferierten Sätze sind jedoch auch mit diesen Trainingseinstellungen nicht zufriedenstellend. Auf eine weitere Erhöhung der Dimensionen wurde aufgrund von Zeit- und Ressourcenmangel verzichtet. Zu diesem Zeitpunkt im Projekt wurde entschieden auf weitere Trainings zu verzichten. Es scheint so, dass mit dem verwendeten Modellen kein zufriedenstellendes Resultat erreichen werden kann. Begründungen für diese Annahme werden in Evaluation und Validation (6) aufgeführt.

5.3. Prototyp

Um die einzelnen Modelle auch mit einem beliebigen Eingabesatz zu testen wurde ein Prototyp in Python (Python Software Foundation, o.D.) entwickelt, welcher ein gewisses Modell laden kann und anschliessend einen eingegebenen Satz in einen anderen transferiert.

Der Prototyp wurde aus zeitlichen Gründen ohne User Interface entwickelt und ist daher ein Python Skript mit einem Command Line Interface. Die Umsetzung wurde bewusst sehr minimal gehalten, da der Fokus dieser Arbeit mehrheitlich auf dem Erarbeiten vielversprechender Modelle war anstatt auf der Entwicklung eines erweiterten Prototypen. Dies wurde aufgrund der dürftigen Ergebnisse der trainierten Modelle entschieden.

Um die Benutzung und Installation des Prototypen zu erleichtern wurde ein Docker Image („Enterprise Container Platform“, o.D.) geschrieben, welches als erstes Git herunterlädt, um das wipro-source (4.6.3) Git klonen zu können. Anschliessend werden die PIP Requirements

ments des Projektes installiert so dass alles korrekt funktioniert. Bei jedem ausführen des Containers wird der Prototyp direkt ausgeführt, so dass dieser benutzt werden kann ohne weitere Schritte. Das Docker Image wurde auf Docker Hub („Docker Hub“, o.D.) geladen, so dass dieses wie ein Git Repository, heruntergeladen werden kann. Dieses Image ist unter der URL Docker Image Prototype erreichbar und ist öffentlich. Das Dockerfile um das Docker Image zu erstellen befindet sich im wipro-source (4.6.3) Projekt.

Wenn das Docker Image ausgeführt wird, startet sogleich der Prototyp. Als Erstes wird nach dem Modell, welches für den Style Transfer gebraucht wird, gefragt. Dabei stehen drei verschiedene zur Verfügung:

1. **cross-align-trimmed-new-loss-full**, ein CrossAlign (3.4.5) Netzwerk trainiert auf dem «wipro-trimmed» Datensatz über 200 Epochen mit den default Hyperparametern, Standard Hyperparameter (5.2.1)
 2. **control-gen-trimmed-bigger-embedding-full**, ein ControlGen (3.4.4) Netzwerk trainiert auf dem «wipro-trimmed» Datensatz über 200 Epochen mit den Hyperparametern für ein grösseres Embedding, ControlGen mit grösseren Dimensionen (5.2.3)
 3. **control-gen-equalized-default-full**, ein ControlGen (3.4.4) trainiert auf dem «wipro-trimmed» Datensatz über 200 Epochen mit den default Hyperparametern, Standard Hyperparameter (5.2.1)

Nach dem Eingeben der Nummer des gewünschten Modells wird das Modell initialisiert und anschliessend die gespeicherten Gewichte geladen, dieser Prozess kann je nach Computer etwas länger dauern. Als Nächstes kann ein Satz zum Transfer eingegeben werden. Dieser wird anschliessend zu einem Batch hinzugefügt und dem Modell übergeben. Der eine Output ist danach der transferierte Satz, welcher noch aus einem encodierten Vektor der Wörter besteht. Diesen kann man anhand des Vokabulars des Modells in Wörter zurückmappen. Als zweiter Output ist der rekonstruierte Satz, welcher auch mittels dem Vokabular zurück transferiert wird. Diese beiden Sätze werden anschliessend ausgegeben und dargestellt.

```
docker run -it --name wipro-prototype -e LANG=C.UTF-8 fabiangroeger96/wipro-prototype:1.0.2
WIPRO Prototype
Welcome to our beautiful WIPRO Prototype
Enter "exit" to stop the prototype

Which styler would you like to use
 1) cross_align_trimmed_new_loss_full
 2) contro_align_trimmed_bigger_embedding_full
 3) contro_align_trimmed_default_full
Enter the number of the styler: 2
2019-12-28 00:28:19.827111: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
Please enter a sentence: Das ist ein Test für die Dokumentation .
Transferred sentence: dieses ist bekannteste Definition die global Definition Infrastruktur .
Reconstructed sentence: Das ist ein Gefährt für die Decke .
Please enter a sentence: 
```

Abbildung 5.37.: Command Line Interface des Prototypen

6. Evaluation und Validation

In diesem Kapitel werden die Resultate der Modelle wie in 4.4 beschrieben evaluiert. Zuerst wird darauf eingegangen wie die Resultate der Modelle evaluiert werden. Weiter wird aufgeführt welche Anforderungen erreicht werden sollten. Anschliessend wird die eigentliche Evaluation durchgeführt.

6.1. Vorgehen der Evaluierung

Wie bereits in 4.4 beschrieben wird soll die Verteilung der Satzlänge der Ausgangssätze sowie den Zielsätzen verglichen werden. Wie auch in 5.1.2 aufgeführt werden die Stilllabel *wikipedia* und *klexikon* durch ihre Satzlänge unterschieden. Für die Evaluierung wird auf den Validationsteil des Datensatzes ein Transfer durchgeführt. Dabei werden die Sätze aus dem Stilllabel *wikipedia* in einen Satz aus dem Stilllabel *klexikon* transferiert. Entsprechendes gilt für die Sätze aus dem Label *klexikon*, welche in einen Satz aus dem Stilllabel *wikipedia* gewandelt werden. Für das erreichen der Anforderungen ist vor allem die Wandlung von kürzeren Sätzen in das Labels *klexikon* interessant. Dies da es sich bei den Zielen der Aufgabenstellung, 1.1, ebenfalls um einen Transfer handelt welche eine längere Satzlänge fordert.

Um die Verteilung er Satzlängen aus den Stilllabels zu vergleichen werden ca. 15'000 Sätze aus jedem Stilllabel einem Transfer unterzogen. Anschliessend werden die Anzahl Wörter des Eingabesatzes sowie des entsprechenden Ausgabesatz verglichen. Anschliessend wird die Differenz der Länge der beiden Sätze gebildet. Es wird erwartet das der Durchschnitt der Differenz für die Ausgabesätze grösser 0 entspricht.

Aufgrund der Resultate in 5.2 wurde entschieden das letzte Modell ControlGen mit grösseren Dimensionen (5.2.3) zu evaluieren. Um ebenfalls einen Vergleich zwischen den beiden Modelle ControlGen und CrossAlign zu erhalten, wird ebenfalls das CrossAlign Modell aus 5.2.2 mit der beschriebenen Methode evaluiert. Beide evaluierten Modelle wurden auf dem Datensatz «Gekürzt»trainiert.

Als Evaluierungs Datensatz wird ein Teil des Datensatz «Ausgeglichen» verwendet. Dieser beinhaltet beinhaltet die Verteilung welche in Statistische Analyse des Datensatz (5.1.2), Abbildung Verteilung Satzlänge Datensatz «Ausgeglichen» (5.13) aufgezeigt wird.

6.2. Verteilung des Evaluierungs Datensatz

Die Verteilung der Sätzlängen auf dem Evaluierungs Datensatz folgt der gleichen Verteilung wie der gesamte Datensatz «Ausgeglichen». Die Verteilung der Sätzlängen der beiden Stile ist in Verteilung Satzlänge Klexikon bereinigt (5.8) und Verteilung Satzlänge Wikipedia bereinigt (5.9) aufgezeigt. Zum Vergleich sind in den Abbildungen 6.1 und 6.2 die Verteilungen der Sätzlängen aus dem Validationsteil des Datensatz «Ausgeglichen» abgebildet.

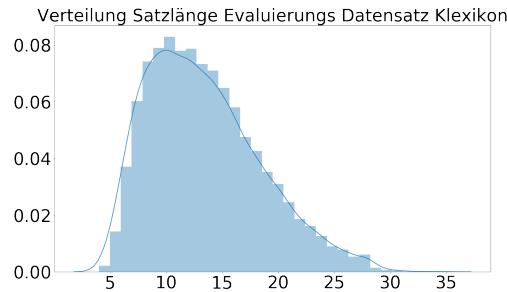


Abbildung 6.1.: Verteilung Satzlänge Klexikon Evaluierungs Datensatz

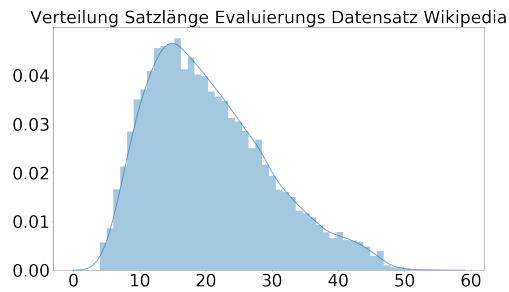


Abbildung 6.2.: Verteilung Satzlänge Wikipedia Evaluierungs Datensatz

Datenquelle	25% Quartil	Median	75% Quartil	Mean	Standardabweichung
Klexikon	9	13	17	13.4	5.1
Wikipedia	14	19	26	20.6	9.1

Tabelle 6.1.: Verteilung Evaluierungs Datensatz

Wie in Abbildung 6.1 und 6.2 zu sehen ist folgt der Evaluierungs Datensatz wie erwartet einer nahezu identischen Verteilung wie der Datensatz «Ausgeglichen».

6.3. Evaluierung der Modelle

6.3.1. CrossAlign gewichtete Verlustfunktion

Als erstes wurde eine Evaluierung auf dem CrossAlign Modell aus 5.2.2 durchgeführt. Nachfolgend wird gezeigt wie die Verteilung der Ausgabesätze für die beiden Transfers ausfällt.

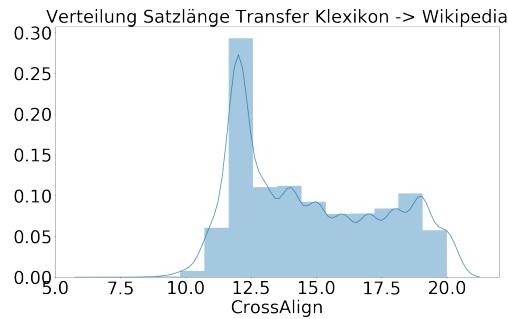


Abbildung 6.3.: Verteilung Satzlänge Transfer Klexikon zu Wikipedia CrossAlign

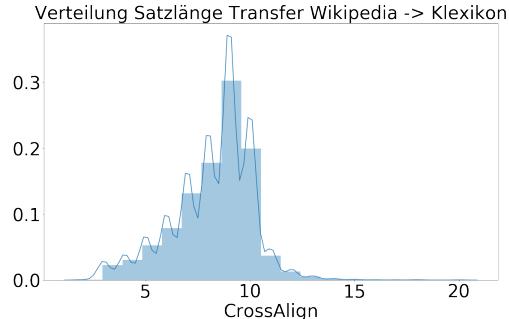


Abbildung 6.4.: Verteilung Satzlänge Transfer Wikipedia zu Klexikon CrossAlign

Transfer	25% Quartil	Median	75% Quartil	Mean	Std. Abw.
Klexikon -> Wikipedia	12	14	17	14.7	2.8
Wikipedia -> Klexikon	11	13	15	12.7	2.5

Tabelle 6.2.: Verteilung Satzlänge Transfer CrossAlign

Wie aus den Grafiken 6.3 und 6.4 und der Tabelle 6.2 entnommen werden kann findet durchaus für beide Transfer eine Verschiebung der Verteilungen statt. Dabei scheint der Transfer vom Stil *wikipedia* in den Stil *klexikon* besser zu funktionieren. Der Transfer von *klexikon* zu *wikipedia* ändert die Verteilung ebenfalls, diese ist jedoch weniger nahe bei einer Gaußverteilung.

Als nächstes soll auf die Differenz der Länge des Eingabe- und des Ausgabesatzes eingegangen werden.

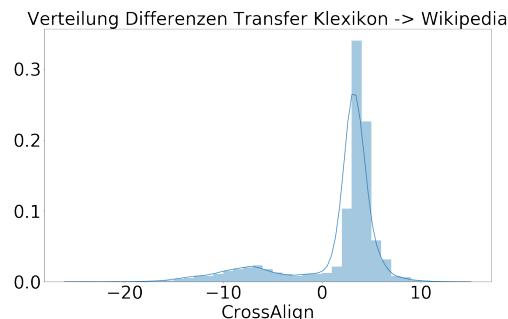


Abbildung 6.5.: Verteilung Differenz Klexikon -> Wikipedia CrossAlign

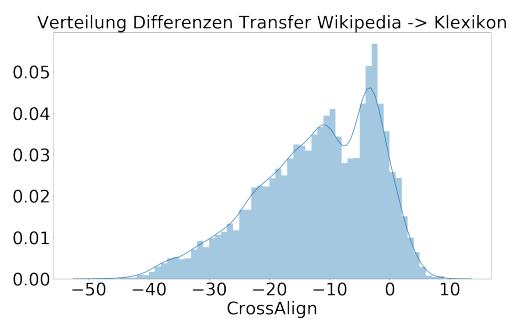


Abbildung 6.6.: Verteilung Differenz Wikipedia -> Klexikon CrossAlign

Transfer	25% Quartil	Median	75% Quartil	Mean	Std. Abw.
Klexikon -> Wikipedia	2	3	4	1.35	4.8
Wikipedia -> Klexikon	-19	-11	-4	-12.39	9.9

Tabelle 6.3.: Verteilung Differenzen Transfer CrossAlign

In den Differenzen zeigen sich die Erkenntnisse welche auch aus den Verteilungen in Abbildungen 6.3 und ?? ersichtlich sind. Beide Transfers verlängern, beziehungsweise verkürzen, den Satz. Der Transfer von *wikipedia* zu *klexikon* weist im Durchschnitt, siehe Mean in Tabelle 6.3, eine höhere Differenz auf als der Transfer von *klexikon* zu *wikipedia*. Diese Aussage wird jedoch durch die ebenfalls höhere Standardabweichung relativiert.

6.3.2. ControlGen mit grösseren Dimensionen

Es wird nun das Modell ControlGen evaluiert welches für den Style Transfer vielversprechende Werte aufzeigt, siehe ControlGen mit grösseren Dimensionen (5.2.3). Ebenfalls wird hier zuerst auf die Verteilung der Ausgabesätze eingegangen und anschliessend die Differenzen untersucht.

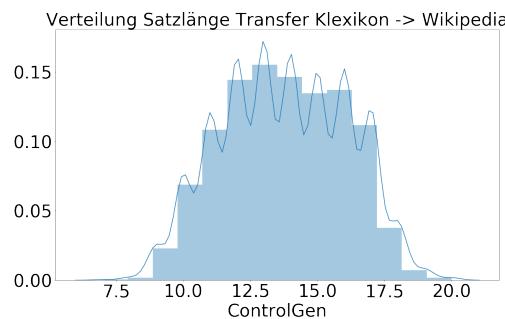


Abbildung 6.7.: Verteilung Satzlänge Transfer Klexikon zu Wikipedia ControlGen

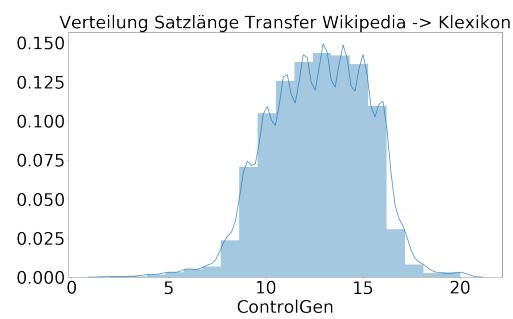


Abbildung 6.8.: Verteilung Satzlänge Transfer Wikipedia zu Klexikon ControlGen

Transfer	25% Quartil	Median	75% Quartil	Mean	Std. Abw.
Klexikon -> Wikipedia	12	14	16	13.8	2.3
Wikipedia -> Klexikon	11	13	15	12.7	2.5

Tabelle 6.4.: Verteilung Satzlänge Transfer ControlGen

Auch für das ControlGen kann eine Verschiebung der Verteilung beobachtet werden. Diese fällt jedoch weniger stark aus als beim Modell CrossAlign. Das ControlGen Modell scheint dabei die Ausgabesätze in einen kleinen Bereich zu zwängen. Dieser befindet sich für den

Transfer von *klexikon* zu *wikipedia* bei 12 bis 16. Für den Transfer *klexikon* zu *wikipedia* bei 11 bis 15. Dies entspricht den 25%- und 75% Quartilen der Verteilung. Dies kann ebenfalls aus der Tabelle 6.4 entnommen werden.

Interessant ist dabei der Rückschluss auf Resultate des Trainings, siehe ControlGen mit grösseren Dimensionen (5.2.3). Dort weis das Modell eine höhe Genauigkeit für den Style Transfer aus. Obwohl die Längen der Ausgabesätze sich minimal unterscheidet.

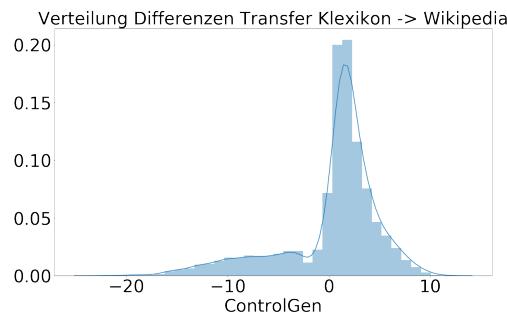


Abbildung 6.9.: Verteilung Differenz Kle-
xikon -> Wikipedia ControlGen

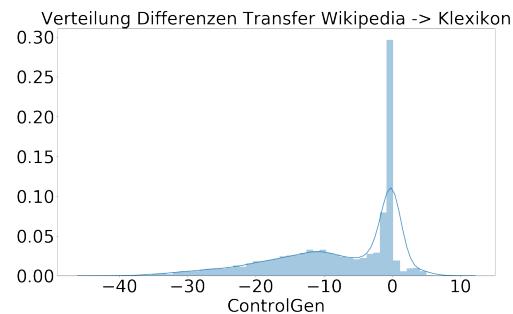


Abbildung 6.10.: Verteilung Differenz
Wikipedia -> Klexikon ControlGen

Transfer	25% Quartil	Median	75% Quartil	Mean	Std. Abw.
Klexikon -> Wikipedia	0	2	3	0.45	4.9
Wikipedia -> Klexikon	-14	-5	0	-7.88	9.14

Tabelle 6.5.: Verteilung Differenzen Transfer ControlGen

Auch hier zeigen die Differenzen ein ähnliches Bild wie beim CrossAlign gewichtete Verlustfunktion (6.3.1). Der Durschnitt, siehe Tabelle 6.5 für das Modell ControlGen fällt jedoch kleiner aus als für das CrossAlign. Jedoch weist auch für dieses Modell der Transfer von *wikipedia* zu *klexikon* eine höhere Standardabweichung auf.

6.3.3. Evaluierung der Ausgabesätze

Wie bereits in Abschliessende Problemdefinition (3.2) beschrieben bestand der Fokus dar- auf die Länge der Sätze zu ändern, anstelle der Verständlichkeit der Sätze. Auch der Bi- lingual evaluation understudy (BLEU) Scores der Resultate der Trainings, siehe 5.2, kann erkennt werden, dass die Verständlichkeit der Sätze gering ist. Dennoch soll zur Vervoll- ständigung an einem Beispiel aufgezeigt werden, wie der Transfer der Modelle ausgeführt wurde. Das Wort *<unk>* markiert ein Wort welches sich nicht im Vokabular des Models be- findet, siehe BaseModel (3.4.3).

Eingabesatz Stilllabel *klexikon*

dort sollte Varus die Grenze gegen die Germanen bewachen .

Ausgabesatz Stilllabel *wikipedia*

dort sollte Ricken die Nationalversammlung gegen die Lage Populationen Leistungen auf Lage , <unk> .

Wie am Beispiel gesehen werden kann, ist der transferierten Sätze unverständlich und bezieht sich kaum bis gar nicht auf den Kontext des Eingabesatzes. Daher wird auf eine genauere Evaluation der Qualität der Ausgabesätze verzichtet. Dies auch wie bereits erwähnt aufgrund des Fokus die Satzlänge zu verändern.

6.4. Vergleich mit Anforderungen

Im Kapitel Evaluation und Validation (6) wurde gezeigt, dass durchaus eine Veränderung der Verteilung statt gefunden hat. Die Verteilungen ändern sich jedoch zu gering um sich der Verteilung des Zielstils anzupassen. Damit konnte mit dem Neural Style Transfer (NST) das Ziel, die Verteilung des Zielstils zu erhalten nicht erreicht werden.

Da die Modelle kein Vokabular aus dem Bereich von Zwischen- und Arbeitzeugnissen kennen, sowie der Transfer, wie in Evaluierung der Ausgabesätze (6.3.3) gezeigt, keine zufriedenstellende Resultate liefert, wurde auf ein Transfer der Textbausteine des Auftraggebers verzichtet.

7. Fazit

Zum Abschluss der Arbeit soll für den Auftraggeber ein Fazit gezogen, sowie ein Ausblick gemacht werden. Diese können als Empfehlungen für die weitere Verwendung der Erkenntnisse der Arbeit verwendet werden.

7.1. Projekt Fazit

Bereits zu Beginn des Projekts wurde eine Problematik aufgezeigt. Für das erfolgreiche Nutzen von Ansätzen aus maschinellem Lernen und künstlicher Intelligenz, sind umfangreiche Datensätze vorausgesetzt. Diese sind im expliziten Bereich von Zwischen- und Arbeitszeugnissen nicht vorhanden. Auch fanden sich keine vergleichbaren Daten, welche eine Anwendung in der produktiven Umgebung des Arbeitgebers verwenden liessen. Zu Beginn des Projektes wurden die möglichen Ansätzen zur Verbesserung der generierten Arbeitszeugnissen durchgespielt, siehe auch Empfehlung im Bezug auf den Zeugnismanager (7.3). Dabei muss aufgeführt werden, dass der Verwendung von Neural Style Transfer (NST) mit neuronalen Netzwerken bereits zu diesem Zeitpunkt, in dem kurzen Projektzeitraum, keine grosse Erfolgsschance zugesprochen wurde. Weiterentwicklungen des Zeugnismanager im Bereich des User Interfaces, des Bewertungsprozesses oder der manuellen Verbesserung der Textbausteine, wurden höhere Erfolgsschancen eingeräumt.

Im Rahmen des Projektantrags für das Wirtschaftsprojekt an der HSLU, sowie in Absprache mit der Betreuungsperson, sollte das Projekt jedoch klar im Bereich von maschinellem Lernen sowie künstlicher Intelligenz angesiedelt sein. Daher wurde entschieden den Style Transfer für die Verwendung im Zeugnismanager zu untersuchen. Dies auch da es sich bei diesem Wirtschaftsprojekt um ein Innovationsprojekt handelt. Auch hat der Auftraggeber bisher keine Berührungspunkte mit maschinellem Lernen oder künstlicher Intelligenz. Daher soll die Arbeit auch einen ersten Einblick in die Disziplin gewähren.

Durch das junge Alter von NST und der geringen Anzahl an kommerziell eingesetzten Lösungen, gestaltete es sich schwierig die entsprechenden Ansätze für die Aufgabenstellung einzuschätzen. Anhand des Sammelsuriums an Literatur aus der Forschung, siehe Fuzhenxin, 2019, lässt sich sehen, dass NST in Kombination mit Natural Language Processing (NLP) gerade eine starke Entwicklung durchläuft. Ein genaues Sichten dieser Arbeiten, sowie das Einschätzen der Resultate stellt bereits ein hoher Aufwand dar.

Weiter war zum Start der Arbeit nicht klar, welcher Datensatz verwendet werden soll. Durch die Entscheidung einen eigenen Datensatz, welcher das Problem annähern soll, zu anzule-

gen, wurde ein Mehraufwand generiert. Dies war sicherlich nötig um eine Untersuchung zu ermöglichen, beanspruchte jedoch einen grossen Teil der Zeitressourcen.

Die Resultate lassen durchaus darauf schliessen, dass NST eine Möglichkeit zur Verbesserung von Sätzen wäre. Auch da, dass Ziel Sätze zu verlängern und zu verkürzen kann teilweise als erreicht gewertet werden kann. Dennoch sind die Resultate in keiner Weise mit den Resultaten aus der Literatur vergleichbar. Die Gründe warum nicht ähnliche Resultate erreicht werden konnte können schwer erörtert werden. Das Projekt behandelte eine grosse Anzahl verschiedener Aspekte aus maschinellem Lernen und künstlicher Intelligenz. Einerseits wurde der Datensatz eigenhändig aufgebaut, sowie der Transfer auf Problem angewandt welches sich von den Problemstellung in der Literatur unterscheidet.

Weiter soll nochmals erwähnt werden, dass es sich beim Projekt um ein exploratives Projekt handelt. Es ging darum Versuche und Möglichkeiten die Aufgabenstellung mit künstlicher Intelligenz zu lösen. Dabei ist ein Scheitern leider nicht auszuschliessen.

7.2. Empfehlung im Bezug auf Neural Style Transfer

NST kann im Bereich von Natural Language Generation (NLG) und NLP durchaus Erfolge verzeichnen und wird zukünftig eine Rolle einnehmen. Die Forschung in diesem Bereich verspricht ein grosses Potenzial und wird sich in den nächsten Jahren stark wandeln. Es ist durchaus möglich, dass Ansätze und Modelle auftauchen werden die den gewünschten Transfer durchführen können. Dabei darf jedoch nicht vernachlässigt werden, dass ein umfangreicher Datensatz dafür notwendig ist. Auch wenn es Ansätze gibt Themenbereich übergreifend Transfers durchzuführen, bleibt das Problem der Dimensionalität in NLP und NLG wohl weiterhin bestehen. Um eine deutliche Verbesserung der Sätze mit dem Transfer zu erhalten, muss ein detaillierter Raum für die Bedeutung einzelner Wörter geschaffen werden. Gerade in einem Bereich, wie in persönlich-adressierten Zeugnissen, wird eine hohe Detailtreue gefordert. Daher ist ein grosser und geprüfter Datensatz aus Arbeitszeugnissen unumgänglich.

Abschliessend werden für den Auftraggeber im Bezug auf Neural Style Transfer und dem Zeugnismanager folgende Empfehlung abgegeben.

- Weiterführen der Arbeit um Ergebnisse für eine abschliessende Beurteilung zu erhalten
- Untersuchung von alternativen Modellen aus dem Bereich von NST
- Wiederaufnahme der Thematik in drei bis fünf Jahren
- Aufbau eines Datensatzes welcher den Style Transfer im Themenbereich von persönlich-adressierten Zwischen- und Arbeitszeugnissen abbildet

7.3. Empfehlung im Bezug auf den Zeugnismanager

Dem Auftraggeber sollen ebenfalls Ansätze zur Verbesserung des Zeugnismanager ausserhalb von NST vorgeschlagen werden. Auch wenn der Zeugnismanager an sich nicht untersucht wurde, können gewisse Vorschläge eingebracht werden.

Verbesserung des User Interface durch Abstimmung Anstelle die Zeugnisse mit Textbausteinen für einzelne Kategorien zu erstellen, könnte für die einzelnen Kategorien über das Wählen von Adjektiven ein Zeugnis erstellt werden. Dabei könnte zum Beispiel eine Art Abstimmung zwischen zwei Adjektiven durchgeführt werden. Dabei wäre es möglich für eine Kategorie zwei bis drei Abstimmungen durchzuführen. Anschliessend kann Anhand dessen ein Textbaustein oder auch ein vollständiges Zeugnis vorgeschlagen werden. Dadurch könnte der Verfasser des Zeugnis sich auf die genaue Bewertung fokussieren.

Einsatz von Fuzzy Logic Mit Fuzzy Logic („Fuzzy Logic Wikipedia“, o.D.) wird es möglich Wörter für eine Kategorie zu bewerten. Anstelle Wörter Binär in beispielsweise «positiv» und «negativ» einzuteilen, wird bei der Fuzzy Logic ein Wort im Interval $[0, 1]$ eingeteilt. Dadurch wird es möglich Wörter wie beispielsweise «gut», «hervorragend» und «herausragend» genauer zu gewichten.

Im Zeugnismanager könnte nun ein Score über das mit den Textbausteinen erzeugte Zeugnis erstellt werden. Dieser Score kann über die verschiedenen Abschnitte erstellt werden. Anschliessend könnte die Kohärenz der Scores überprüft werden und so Verbesserungsvorschläge erstellt werden. Auch wäre es möglich mit einem Slider dem Verfasser ein Feineinstellung der relevanten Wörter in einem Textbaustein zu ermöglichen.

Abbildungsverzeichnis

2.1	Neural Style Transfer für artistische Gemälde	4
2.2	Unterschied zwischen einem ANN und einem NN (source)	5
2.3	Struktur eines neuronalen Netzes (source)	7
2.4	Visualisierung einer Markov Chain für Sequenzen	9
2.5	Visualisierung eines RNN	10
2.6	Visualisierung eines LSTM Gates (Colah's Blog)	10
2.7	Visualisierung eines LSTM (Colah's Blog)	11
2.8	Visualisierung eines LSTM Vergessensgates (Colah's Blog)	12
2.9	Visualisierung eines LSTM Eingangsgates (Colah's Blog)	12
2.10	Visualisierung der Addition der Vektoren eines LSTMs (Colah's Blog)	13
2.11	Visualisierung eines LSTM Ausgangsgate (Colah's Blog)	13
2.12	Visualisierung eines Autoencoders	14
2.13	Visualisierung eines Transformers	15
2.14	Visualisierung eines GANs	16
2.15	Lineare Aktivierungsfunktion	16
2.16	Nicht lineare Aktivierungsfunktion	17
2.17	Sigmoid Funktion	17
2.18	Tanh Funktion	18
2.19	ReLU Funktion	18
3.1	Klassendiagramm der Modelle	27
3.2	Visualisierung der Classifier Architektur	28
3.3	Abbildung der Idee von ControlGen	29
3.4	Abbildung der Idee von CrossAlign	32
4.1	Meilensteinplanung	36
4.2	Screenshot Zeugnismanager	37
5.1	Anz. Artikel im Datensatz	43
5.2	Anz. Sätze im Datensatz	43
5.3	Anz. Wörter im Datensatz	43
5.4	Verteilung Sätze pro Artikel Klexikon	44
5.5	Verteilung Sätze pro Artikel Wikipedia	44
5.6	Verteilung Satzlänge Klexikon nicht bereinigt	44
5.7	Verteilung Satzlänge Wikipedia nicht bereinigt	44
5.8	Verteilung Satzlänge Klexikon bereinigt	45

5.9	Verteilung Satzlänge Wikipedia bereinigt	45
5.10	Anz. Artikel im bereinigten Datensatz	45
5.11	Anz. Sätze im bereinigten Datensatz	45
5.12	Anz. Wörter im bereinigten Datensatz	45
5.13	Verteilung Satzlänge Datensatz «Ausgeglichen»	46
5.14	Verteilung Satzlänge Klexikon «Gekürzt»	47
5.15	Verteilung Satzlänge Wikipedia «Gekürzt»	47
5.16	Genauigkeit Transfer ControlGen Default «Ausgeglichen»	48
5.17	Genauigkeit Transfer CrossAlign Default «Ausgeglichen»	48
5.18	BLEU Score ControlGen Default «Ausgeglichen»	49
5.19	BLEU Score Transfer CrossAlign Default «Ausgeglichen»	49
5.20	Verlustfunktion ControlGen Default «Ausgeglichen»	50
5.21	Verlustfunktion CrossAlign Default «Ausgeglichen»	50
5.22	Genauigkeit Transfer ControlGen Default «Gekürzt»	50
5.23	Genauigkeit Transfer CrossAlign Default «Gekürzt»	50
5.24	BLEU Score ControlGen Default «Gekürzt»	50
5.25	BLEU Score Transfer CrossAlign Default «Gekürzt»	50
5.26	Verlustfunktion ControlGen Default «Gekürzt»	51
5.27	Verlustfunktion CrossAlign Default «Gekürzt»	51
5.28	Genauigkeit Transfer ControlGen Gewichtet «Gekürzt»	53
5.29	Genauigkeit Transfer CrossAlign Gewichtet «Gekürzt»	53
5.30	BLEU Score ControlGen Gewichtet «Gekürzt»	53
5.31	BLEU Score Transfer CrossAlign Gewichtet «Gekürzt»	53
5.32	Verlustfunktion ControlGen Gewichtet «Gekürzt»	53
5.33	Verlustfunktion CrossAlign Gewichtet «Gekürzt»	53
5.34	Genauigkeit Transfer ControlGen Dimensionen «Gekürzt»	56
5.35	BLEU Score ControlGen Dimensionen «Gekürzt»	56
5.36	Verlustfunktion ControlGen Dimensionen «Gekürzt»	57
5.37	Command Line Interface des Prototypen	58
6.1	Verteilung Satzlänge Klexikon Evaluierungs Datensatz	60
6.2	Verteilung Satzlänge Wikipedia Evaluierungs Datensatz	60
6.3	Verteilung Satzlänge Transfer Klexikon zu Wikipedia CrossAlign	61
6.4	Verteilung Satzlänge Transfer Wikipedia zu Klexikon CrossAlign	61
6.5	Verteilung Differenz Klexikon -> Wikipedia CrossAlign	61
6.6	Verteilung Differenz Wikipedia -> Klexikon CrossAlign	61
6.7	Verteilung Satzlänge Transfer Klexikon zu Wikipedia ControlGen	62
6.8	Verteilung Satzlänge Transfer Wikipedia zu Klexikon ControlGen	62
6.9	Verteilung Differenz Klexikon -> Wikipedia ControlGen	63
6.10	Verteilung Differenz Wikipedia -> Klexikon ControlGen	63

Tabellenverzeichnis

2.1	Beispiele eines NST mit Stil «positiv » und «negativ » mit dem Domain Adaptive Style Transfer (DAST) Modell	4
2.2	Interpretation des BLEU Wertes	20
3.1	Hyperparameter des Classifiers	28
3.2	Hyperparameter des ControlGen Modell	31
3.3	Hyperparameter des CrossAlign Modell	33
5.1	Anzahl der Artikel, Sätze und Wörter aus Wikipedia und Klexikon.	43
5.2	Verteilung der Anzahl Sätze pro Artikel für Klexikon und Wikipedia	44
5.3	Verteilung Anzahl Wörter Klexikon und Wikipedia	44
5.4	Verteilung Anzahl Wörter Klexikon und Wikipedia ohne Ausreisser	45
5.5	Anzahl Sätze, Sätze und Wörter Klexikon, Wikipedia und Wikipedia «Ausgeglichen»	46
5.6	Training der Modelle mit standard Hyperparameter	49
5.7	Training der Modelle mit gewichteter Transfer Lossfunktion Hyperparameter .	52
5.8	Training der Modelle mit grösseren Embedding Hyperparametern	55
6.1	Verteilung Evaluierungs Datensatz	60
6.2	Verteilung Satzlänge Transfer CrossAlign	61
6.3	Verteilung Differenzen Transfer CrossAlign	62
6.4	Verteilung Satzlänge Transfer ControlGen	62
6.5	Verteilung Differenzen Transfer ControlGen	63
C.1	Arbeitsjournal	XI

Formelverzeichnis

2.1	Style Transfer Variablen Definition	5
2.2	Style Transfer Loss Funktion	5

2.3	BLEU Score	21
2.4	Precision im BLEU Score	21
2.5	TER Score	21
3.0	Struktur eines Satzes	23
3.1	Möglichkeiten der Werte der Stillabels	23
3.2	Aufgabenstellung mathematisch formuliert	24

Abkürzungsverzeichnis

A

ANN Artificial Neuronal Net 6

B

BLEU Bilingual evaluation understudy VIII, 20, 21, 48, 51, 55, 57, 63

C

CNN Convolutional Neural Network 27, 28

D

DAST Domain Adaptive Style Transfer 4

G

GAN Generative Adversarial Networks V, 15, 28, 29, 39

GPU Graphics processing unit 11

I

IQR Interquartilabstand (eng. Inter-quantil range) 45

K

KNN Künstliches neuronales Netz 5, 7, 8, 13, 14

L

LSTM Long short term memory 9–12

M

METEOR Metric for Evaluation for Translation with Explicit Ordering 33

N

NLG Natural Language Generation 3, 6, 19, 66

NLP Natural Language Processing 6, 65, 66

NN Neuronales Netz 6–8, 10, 16

NST Neural Style Transfer XI, 4, 25, 26, 34, 64–67

R

RAM Random Access Memory 55

RNN Recurrent neural network 6, 9, 10, 25, 28

T

TER Translation error rate 21

TPU Tensor processing unit 11

V

VAE Variational Auto-Encoder 29–31

W

WER Word error rate 33

Literaturverzeichnis

- Andrew's Adventures in Technology. (2019). Andrews Adventures in Technology. Zugriff unter <http://awalsh128.blogspot.com/2013/01/text-generation-using-markov-chains.html>
- Docker Hub. (o.D.). Zugriff 19. Dezember 2019 unter <https://hub.docker.com/>
- Enterprise Container Platform. (o.D.). Zugriff 19. Dezember 2019 unter <https://www.docker.com/>
- Fu, Z. (o.D.). fuzhenxin/Style-Transfer-in-Text. Zugriff unter <https://github.com/fuzhenxin/Style-Transfer-in-Text>
- Fuzhenxin. (2019). fuzhenxin/Style-Transfer-in-Text. Zugriff unter <https://github.com/fuzhenxin/Style-Transfer-in-Text>
- Fuzzy Logic Wikipedia. (o.D.). Zugriff 19. Dezember 2019 unter <https://de.wikipedia.org/wiki/Fuzzylogik>
- Gatys, L. A., Ecker, A. S. & Bethge, M. (2015). A Neural Algorithm of Artistic Style. arXiv: 1508.06576 [cs.CV]
- GeForce. (2019). Wikimedia Foundation. Zugriff unter <https://en.wikipedia.org/wiki/GeForce>
- Google Brain Team. (o.D.). TensorFlow. Zugriff unter <https://www.tensorflow.org/>
- Hu, Z., Yang, Z., Liang, X., Salakhutdinov, R. & Xing, E. P. (2017). Toward Controlled Generation of Text. arXiv: 1703.00955 [cs.LG]
- Li, D., Zhang, Y., Gan, Z., Cheng, Y., Brockett, C., Sun, M.-T. & Dolan, W. B. (o.D.). cookielee77/DAST. Zugriff unter <https://github.com/cookielee77/DAST>
- Li, D., Zhang, Y., Gan, Z., Cheng, Y., Brockett, C., Sun, M.-T. & Dolan, W. B. (2019). Domain Adaptive Text Style Transfer. ArXiv, *abs/1908.09395*.
- Modelle bewerten AutoML. (2019). Google. Zugriff unter <https://cloud.google.com/translate/automl/docs/evaluate?hl=de>
- Natural language processing. (2019). Wikimedia Foundation. Zugriff unter https://en.wikipedia.org/wiki/Natural_language_processing
- Natural-language generation. (2019). Wikimedia Foundation. Zugriff unter https://en.wikipedia.org/wiki/Natural-language_generation
- Python Software Foundation. (o.D.). Welcome to Python.org. Zugriff unter <https://www.python.org/>
- Shen, T., Lei, T., Barzilay, R. & Jaakkola, T. (2017). Style Transfer from Non-Parallel Text by Cross-Alignment. arXiv: 1705.09655 [cs.CL]
- Unit Test Wikipedia. (o.D.). Zugriff 19. Dezember 2019 unter <https://de.wikipedia.org/wiki/Modultest>

A. Meilensteinberichte

In diesem Kapitel werden die einzelnen Berichte der Meilensteine aufgeführt. Diese sollen Aufschluss über den Stand des Projektes während der Umsetzung geben.

A.1. Meilensteinbericht M1 vom 26. September 2019

In der Phase bis zum ersten Meilenstein ging es darum das Projekt zu starten sowie die Initialisierungsphase einzuläuten. Ziel dieses ersten Eckpunktes war, sich nochmals genau mit der Aufgabenstellung zu befassen und allenfalls Unklarheiten mit dem Projektbetreuer zu klären. Ausserdem sollte zum Ende dieser Phase die Aufgaben der Initialisierungsphase aufgeteilt sein.

A.1.1. Aktueller Stand bei Meilenstein

Alle vorgenommenen Aufgaben, welche in diesem Meilenstein zu erledigen waren, konnten abgeschlossen werden. Die Aufgabenstellung war klar und es war deutlich was in der Initialisierungsphase des Projektes zu tun war.

A.1.2. Ausblick zum nächsten Meilenstein

Da alle Aufgaben des aktuellen Meilensteins erledigt werden konnten, müssen keine Aufgaben in den nächsten Meilenstein übernommen werden.

A.2. Meilensteinbericht M2 vom 10. Oktoper 2019

In der Phase bis zum Meilenstein M2 soll die Initialisierungsphase abgeschlossen werden. Diese beinhaltet, dass die Fragestellung des Projektes nochmals genau durchgearbeitet wird, allfällige Fragen geklärt werden und vor allem Eckpunkte des Projektes definiert werden.

A.2.1. Aktueller Stand bei Meilenstein

In diesem Meilenstein konnte alle Aufgaben der Initialisierungsphase abgeschlossen werden, die Aufgabenstellung wurde nochmals gründlich untersucht und verschiedene Fragen die währenddessen aufgetaucht sind, geklärt. Ausserdem wurde sich darauf geeinigt, dass die Sprache des Projektes Deutsch ist und auch der Prototyp für die Deutsche Sprache entwickelt werden soll.

A.2.2. Ausblick zum nächsten Meilenstein

Da in dieser Phase alle Aufgaben erledigt werden konnten, müssen keine Tasks in die nächste Phase übernommen werden und der Meilenstein kann abgeschlossen werden. Ausserdem wird mit dem Abschluss dieses Meilensteins die Ideenfindungsphase gestartet.

A.3. Meilensteinbericht M3 vom 21. Oktober 2019

In der Phase bis zum Meilenstein M3 soll die divergierende Ideenfindungsphase abgeschlossen werden. Diese Phase beinhaltet, dass der aktuelle Stand der Technik untersucht wird und verschiedene aktuelle Ansätze erforscht werden. Diese Phase soll auch dazu dienen, erste Experimente bezüglich der Aufgabenstellung durchzuführen um die Machbarkeit der verschiedenen Ansätze zu beurteilen.

A.3.1. Aktueller Stand bei Meilenstein

In diesem Meilenstein wurden viele verschiedene Ansätze für die Umsetzung des Projektes untersucht und auch einige wenige Experimente durchgeführt. Während dieser Phase wurde sehr schnell klar, dass es viele verschiedene Möglichkeiten gibt die Problemstellung zu lösen. Daher wurde entschieden, dass der Fokus der Ideenfindung auf dem Bereich des Neuronal Style Transfer liegt und sich die Ideenfindung eher auf die Untersuchung verschiedener Style Transfer Modelle beschränken sollte. Dadurch konnten in der Ideenfindungsphase, viele verschiedene Modelle anhand deren wissenschaftlichen Arbeiten beurteilt werden.

A.3.2. Ausblick zum nächsten Meilenstein

Der Meilenstein wurde erreicht, durch das erfolgreiche Finden verschiedener Ansätze die eine vielversprechende Lösung für das Projekt bieten. In der nächsten Phase wird es deutlich ob die verschiedenen Modelle Erfolgsschancen bieten.

A.4. Meilensteinbericht M4 vom 11. November 2019

In der Phase bis zum Meilenstein M4 soll die konvergierende Ideenfindungsphase abgeschlossen werden. Diese Phase beinhaltet, dass die gefundenen Modelle aus der divergierenden Ideenfindungsphase, bezüglich der Umsetzbarkeit und Anwendbarkeit auf das Projekt, beurteilt werden sollen. Ausserdem sollte ein Datenkorpus gewählt werden für das Projekt, um die jeweiligen Modelle damit zu trainieren.

A.4.1. Aktueller Stand bei Meilenstein

In diesem Meilenstein konnten die in der vorherigen Phase gefundenen Modelle beurteilt werden. Dabei wurde vor allem bemerkt, dass die Auswahl des Datenkorpus eine der wichtigsten Entscheidungen ist, die in unserem Projekt getroffen werden muss. Denn durch die

gewählten Daten wird auch entschieden welche Modelle überhaupt in Frage kommen, z.B. ist eine der Entscheidung welche getroffen werden muss ob es sich bei um parallele Daten oder nicht handelt. Die Sprache des Korpus wurde bereits in einer vorherigen Phase definiert und somit wurde die Auswahl auch eingeschränkt. Geeinigt wurde sich auf einen deutschen nicht parallelen Korpus bestehend aus Sätzen eines Kinderlexikons und Wikipedia, um einen genügend grossen Unterschied zu erhalten. Anschliessend konnten auch verschiedene Modelle ausgeschlossen werden, welche einen parallelen Datenkorpus benötigen. Bis zum Schluss hatten wir noch 2 Modelle (ControlGen und CrossAlign), welche unseren Anforderungen entsprachen und auch mit den Daten von uns arbeiten können.

A.4.2. Ausblick zum nächsten Meilenstein

Der Meilenstein M4 konnte erreicht werden, vor allem durch die Entscheidung sich zuerst auf die Auswahl der Daten zu konzentrieren und erst anschliessend auf die Auswahl der Modelle. In dieser Phase konnten sehr viele wichtige Erkenntnisse über das Trainieren der Modelle und der breite der Modelle gewonnen werden.

A.5. Meilensteinbericht M5 vom 9. Dezember 2019

Bis zum Meilenstein M5 soll die Umsetzungsphase abgeschlossen sein, es soll also ein funktionsfähiger Prototyp zur Verfügung stehen, welcher die Problemstellung ansatzweise löst. Dieser Prototyp soll mit den erforschten Modellen entwickelt werden und die gelernten Erkenntnisse aus vorherigen Phasen angewandt werden. Ausserdem soll die Dokumentation der ganzen Arbeit in einer ersten Fassung vorliegen, so dass mit dem Gegenlesen und Korrigieren angefangen werden kann.

A.5.1. Aktueller Stand bei Meilenstein

Die Aufgaben in diesem Meilenstein konnten nicht vollständig erreicht werden, da sehr viel Zeit verloren ging beim Testen der beiden Modelle, um die optimalen Hyperparameter zu finden. Der Grund dafür war, dass das nötige Wissen über das Trainieren von Generative Adversarial Networks (GAN) fehlt und dadurch die Graphen der einzelnen Modelle schlecht interpretiert werden konnten. Daher wurde im Gespräch mit dem Projektbetreuer versucht von diesem Wissen zu profitieren, so dass die nächsten Modelle besser interpretiert werden können. Durch das viele probieren der Hyperparameter konnten noch keine Fortschritte bei der Entwicklung des Prototypen gemacht werden.

A.5.2. Ausblick zum nächsten Meilenstein

Der Meilenstein konnte nicht erreicht werden, da Artefakte wie der Prototyp oder das vollständige Testen der Modelle nicht vorliegen. Dadurch müssen diese Aufgaben in den nächsten Meilenstein einfließen und erledigt werden. Obwohl der Meilenstein nicht erreicht wurde, ist unser Wissen über GAN und ihr Training extrem gestiegen, so dass wir zuversicht-

lich sind diese in der nächsten Phase zu meistern. Die Dokumentation wurde durch diese Probleme auch ein wenig zurückgestellt und muss in dem letzten Meilenstein nochmals aufgenommen werden.

A.6. Meilensteinbericht M6 vom 20. Dezember 2019

Der Meilenstein M6 ist der letzte im Projekt und bis dorthin muss die Dokumentation fertig sein, sowie ein funktionsfähiger Prototyp vorliegen. In dieser Phase geht es vor allem um die Finalisierung des Dokumentes und Prototypes.

A.6.1. Aktueller Stand bei Meilenstein

Im letzten Meilenstein mussten die Artefakte, welche in der vorherigen Phase nicht zu Stande kamen, als erstes erarbeitet werden. Zunächst konnten nochmals verschiedene Versionen der Modelle erstellt, interpretiert und verglichen werden. Dies konnte nur mithilfe der vorher gewonnenen Erkenntnisse geschehen. Danach konnte relativ schnell der Prototyp geschrieben werden, weil dieser auf das Minimum an Funktionalität reduziert wurde. Und zum Schluss wurde noch die Dokumentation über das Projekt fertiggestellt und der Meilenstein somit erreicht werden.

B. Recherche

B.1. Links

- List of various useful links: <<https://github.com/fuzhenxin/Style-Transfer-in-Text>>

B.2. Arbeiten

All paper listed. Every paper will assigned a number to make communication easier. Papers are tried to be hold in categories.

B.2.1. Overview - Neural Style Transfer

Papers which describe neural style transfer in general and show a general overview over the existing technologies and ideas.

01 - What is wrong with style transfer for texts?

Link: <<https://arxiv.org/pdf/1808.04365.pdf>>

Das Paper bietet eine gute Übersicht über die aktuelle Lage mit dem Thema Neural Style Transfer. Die Übersicht wird unter der Frage nach den schwächen des Bereichs.

Dabei wird eine Übersicht über die vorhanden vorgeschlagenen Modelle gegeben.

- **Ad-hoc defined:** Der Style Transfer erfolgt über ein Datenset eines Autoren oder Plattform. Dabei wird versucht der Stil des Datenset wiederzugeben. Dabei wird, z.B. mit einem LSTM, ein neuer Text aus dem Set generiert. Es handelt sich somit um die generierung von neuem Text aufgrund eines Datensatz.
- **Neural Machine Translation (NMT):** Die Stile werden als Sprachen angesehen. Dabei wird ein Übersetzer für den einen in den anderen Stil (Sprache) verwendet. So mit kann mithilfe eines parallelens Datensatzes stilisiert werden.
- **Post NMT:** Allgemein wurde sich geeinigt das der Style Transfer nicht aufgrund von Fragmenten getätigigt werden soll. Eher soll der Transfer aufgrund der Daten parametrisiert werden. Dieser Ansatz wird momentan stark weiterverfolgt und erweitert.

Weiter wird im Paper die Problemstellung der Definition für Text-Semantik und Text-Stil aufgeführt.

03 - A Neural Algorithm of Artistic Style

Link: <<https://arxiv.org/abs/1508.06576>>

Das Paper bietet die Grundlagen für Neural Style Transfer. Dabei wird die Technik anhand von Transfer von artistischem Stil anhand von Gemälden und Bildern eingeführt.

Das Paper ist relevant da oft zitiert und als eigentlicher Ursprung der Technik gehandelt.

B.2.2. Domain Adaptive Approach**02 - Domain Adaptive Style Transfer**

Link: <<https://arxiv.org/abs/1908.09395>>

Dieses Modell versucht aufgrund von eines nicht parallelen domain-übergreifenden Datensets den Stil daraus zu parametrisieren.

Der Ansatz ist interessant, da der zu Datensatz geringe Mengen an Domain spezifischen Daten benötigt.

B.2.3. Latent Representation Approach**04 - Controllable Unsupervised Text Attribute Transfer via Editing Entangled Latent Representation**

Link: <<https://arxiv.org/abs/1905.12926>>

In diesem Ansatz wird versucht der Latent Space (Feature Space) mit Gewichtung zu kontrollieren. Dadurch kann ein Input auf verschiedene Weisen interpretiert werden. Dabei repräsentiert das Gewicht, wie stark der Text vom Source-Style zum Target-Style stilisiert werden soll.

Der Ansatz ist interessant, da eine Abstufung des Stils möglich ist. So könnte eventuell die Bewertung beibehalten werden.

05 - Evaluating prose style transfer with the Bible

Link: <<https://arxiv.org/abs/1711.04731>>

In dieser Forschungsarbeit geht es darum, dass ein System, welches mit einem Input Text und einem Zielprosa-Stil versorgt wird, eine Ausgabe liefert, die die Bedeutung des Eingabertextes bewahrt, aber den Stil ändert. Um das Modell zu trainieren wird ein paralleler Datenkorpus benötigt, in diesem Beispiel wird die Bibel in verschiedenen Ausführungen als qualitativ hochwertiger Korpus genutzt. Als Metriken werden Bilingual evaluation understudy

(BLEU) und PINC verwendet.

Da der Datenkorpus qualitativ sehr hochwertig ist, kann dieser Korpus auch für weitere Modelle verwendet werden. Jedoch wird in unserer Arbeit ein deutscher Datenkorpus angestrebt. Dennoch ist die Idee die Bibel als parallele Datensammlung zu verwenden interessant und wird mit deutschen Editionen weiter untersucht.

Ausserdem werden in dieser Schriftstück wertvolle Referenzen auf andere Arbeiten genannt, welche sich mit einigen Themen des Style Transfers noch tiefer auseinander gesetzt haben. Diese Referenzen sollten weiter untersucht werden.

Das grösste Problem von Style Transfer wird hier immer wieder angesprochen, nämlich die mangelhafte Anzahl parallelen Datenkorpusse.

B.3. Metriken

B.3.1. BLEU Score

Links:

- <https://towardsdatascience.com/evaluating-text-output-in-nlp-bleu-at-your-own-risk-e8609665a213>
- <https://www.sdl.com/blog/understanding-mt-quality-bleu-scores.html>

C. Arbeitsjournal

In diesem Kapitel wird das Arbeitsjournal, welches im Rahmen des Projektes geführt wurde, dargestellt. Es soll aufzeigen wie die Arbeit des Projektes auf die zwei Studenten verteilt wurde. Das Projekt wurde in mehrere grössere Teile zusammengefasst und im Journal aufgelistet, so dass das Arbeitsjournal möglichst kompakt und simpel gehalten werden konnte. Die Auflistung wurde an den Aufbau des Projektes und die vier Git Repositories angelegt. Um detailliertere Angaben zur Aufgabenverteilung und deren Umsetzung zu erhalten können in den Git Projekten die Commits verfolgt werden unter:

1. wipro-doc
2. wipro-data
3. wipro-source
4. wipro-logs

In der Tabelle C.1 ist das Arbeitsjournal des Projektes aufgelistet, wobei in der Spalte *Student* das Kürzel des jeweiligen Studenten verwendet wurde. RB für Reto Barmettler und FG für Fabian Gröger.

Arbeit	Details	Projekt	Student	Stunden
Git Repositories	Pflegen der Git Repositories	gitlab	RB, FG	10h
Dokumentation	Dokumentieren der Arbeit	wipro-doc	RB, FG	110h
Projektmanagement	Managen des ganzen Projektes	wipro-doc	RB	5h
Recherche	Recherche zur Problemstellung	wipro-doc	RB, FG	10h
Recherche	Recherche zum Neural Style Transfer	wipro-doc	RB, FG	20h
Recherche	Recherche zu Modellen von Neural Style Transfer (NST)	wipro-doc	RB, FG	30h
Datensatz	Idee welchen Datensatz verwenden	wipro-data	RB, FG	10h
Datensatz	Daten von Internet crawlen	wipro-data	FG	5h
Datensatz	Daten aufbereiten, säubern	wipro-data	RB	15h
Datensatz	Daten statistisch analysieren	wipro-data	RB	10h
Datensatz	Daten aufbreiten für Weiterverarbeitung	wipro-data	RB	5h
Modelle	Bestehendes GitHub Projekt integriert	wipro-source	RB, FG	5h
Modelle	Anpassungen an den Modellen	wipro-source	FG	15h
Modelle	Training der Modelle	wipro-source	FG	15h
Prototyp	Erstellen des Prototypen	wipro-source	RB, FG	10h
Evaluation	Bewerten der Resultate der Modelle	wipro-logs	RB, FG	20h
Evaluation	Evaluieren der Resultate der Modelle	wipro-logs	RB, FG	20h
Docker	Dockerfile für Prototypen	wipro-source	FG	5h

Tabelle C.1.: Arbeitsjournal

D. Anleitung Installation Prototyp

In diesem Abschnitt wird die Installation des Prototypen genau beschrieben. Der Installationsprozess gestaltet sich relativ simpel, weil dieser in ein Docker Image gepackt wurde.

1. Als erstes muss gewährleistet werden, dass Docker korrekt installiert ist und eine konstante Internetverbindung besteht. Wenn diese beiden Punkte sichergestellt wurden, kann mit dem zweiten Schritt weitergemacht werden.
2. Danach muss nur noch der Docker Container von Docker Hub gepullt werden, dies geschieht mittels diesem Command:

```
docker run \
-it --name wipro-prototype \
-e LANG=C.UTF-8 \
fabiangroeger96/wipro-prototype:1.0.2
```

Dieser holt sich die Version 1.0.2 von Docker Hub und führt diese gleich aus, mit einer interaktiven Shell gekoppelt.

3. Nach dem Starten des Docker Containers wird sogleich der Prototyp gestartet und dieser kann verwendet werden.
4. Um den Prototypen zu beenden, kann das Keyword `exit` verwendet werden
5. Nach dem Benutzen des Docker Images kann dieses mittels dem folgenden Befehl gelöscht werden

```
docker rm -f wipro-prototype
```

E. Aufgabenstellung Wirtschaftsprojekt

1. Starttermin:

spätmöglichster Starttermin: HS KW 38; FS KW 8

16.09.2019

16.09.2019, KW38

2. Abgabetermin:

Abgabetermin Wirtschaftsprojekt: Starttermin + 14 Arbeitswochen*

Abgabetermin Bachelorarbeit: Starttermin + 16 Arbeitswochen*

*Montag bis Freitag gilt als eine Arbeitswoche

22.12.2019

22.12.2019, KW51

Abschluss Präsentation zwischen 13.01.2019 – 24.01.2019

3. Studierende:

	Student/in 1:	Student/in 2:
Name, Vorname:	Barmettler Reto	Gröger Fabian
Studiengang:	Informatik	Informatik
Mobile:	+41 79 549 31 00	+41 79 554 93 01
E-Mail:	reto.barmettler@stud.hslu.ch	Fabian.groeger@stud.hslu.ch
Projekt mit Arbeitgeber (bb-Studierende)	<input type="checkbox"/> Ja <input checked="" type="checkbox"/> Nein	<input type="checkbox"/> Ja <input checked="" type="checkbox"/> Nein

4. Auftraggeber/in:

Firma:	confer! AG
Ansprechperson:	Mario Jacomet
Funktion:	Leiter Softwareentwicklung
Strasse:	Im Eichli 9
PLZ / Ort:	6370 Stans
Telefon:	041 610 83 11
Email:	mario.jacomet@confer.ch
Website:	http://www.confer.ch

5. Betreuende/r Dozent/in:

Daniel Pfäffli

6. Zusätzliche/r Dozent/in (beim Wirtschaftsprojekt) oder externe/r Expert/in (bei der Bachelorarbeit):

Tim vor der Brück

FH Zentralschweiz



7. Aufgabenstellung

Titel:	Generierung ansprechender Arbeitszeugnisse
Ausgangslage und Problemstellung:	Mit dem Zeugnismanager der confer! AG können Arbeits- und Zwischenzeugnisse in vier Sprachen erstellt werden. Die Zeugnisse werden durch einfaches anwählen von Bausteinen zusammengestellt. Die so entstehenden Zeugnisse sind «holprig» zu lesen. Ein Beispiel ist der Einsatz der Anrede (Frau Meier) und des Personalpronomens (Sie) an geeigneter Stelle. Solche und ähnliche Problemstellungen werden im Moment mit einfachen Heuristiken gelöst. Weitere Probleme ergeben sich z. B. durch die unterschiedlichen Grammatiken in den jeweiligen Sprachen. So müssen viele Spezialfälle behandelt werden.
Ziel der Arbeit und erwartete Resultate:	<p>Das Ziel ist es, den Zeugnismanager dabei zu unterstützen, sprachlich korrekt ausformulierte Zeugnisse mit einem guten Lesefluss zu erstellen. Es soll eine Komponente mit Prototypcharakter erarbeitet werden. Der zu erarbeitende Prototyp kann sich auf eine Sprache, Zeitform und ggf. sogar Geschlecht einschränken, sollte aber entsprechend erweiterbar sein.</p> <p>Erwartete Resultate:</p> <ul style="list-style-type: none"> • Ausführliches Study Doc • Schlussbericht gemäss Vorgaben des Bachelor-Studiengangs Informatik (siehe «WIPRO Methodisches Arbeiten») • Source Repository • Daten-Korpus • Lauffähiger Prototype <p>Ausführungen zu den einzelnen Teilaufgaben:</p> <p>Daten-Korpus</p> <ol style="list-style-type: none"> 1. Identifikation mind. eines geeigneten Daten-Korpus <ul style="list-style-type: none"> ◦ Umfang eignet sich für Deep Learning Algorithmen ◦ Enthält für die Aufgabe geeignete Labels ◦ Sprache ist bevorzugt Deutsch oder Englisch <p>Identifikation und Begründung von geeigneten Modellen:</p> <ol style="list-style-type: none"> 1. Zusammenfassung und Erklärung pro evaluiertes Verfahren 2. Vergleich der Modelle auf Funktionsweise und publizierten Resultaten 3. Identifikation von mind. 2 vielversprechenden Modellen <p>Modell Evaluation:</p> <ol style="list-style-type: none"> 1. Identifizieren von geeigneten Metriken 2. Evaluations Workflow 3. Vergleich der identifizierten Modelle anhand der Metriken auf den Daten-Korpus. 4. Entscheid des besten Modelles <p>Kritische Reflexion und Empfehlung:</p> <ol style="list-style-type: none"> 1. Interpretation der Resultate, Schlussfolgerungen und Massnahmen 2. Reflexion der gesamten Arbeit/ Fazit 3. Empfehlung für confer! AG bezüglich Verfahren, Datenumfang, Implementation <p>Source Code allgemein:</p> <ul style="list-style-type: none"> • Nachvollziehbarkeit <ul style="list-style-type: none"> ◦ Abhängigkeiten werden durch ein Skript installiert oder es steht ein (Windows-)Docker Image oder vorinstallierte virtuelle Maschine im Enterpriselab zur Verfügung. ◦ Skripts für automatisierte Tests (Code coverage Reports sind erwünscht) und Build-Prozesse

	<ul style="list-style-type: none"> ○ Resultate von Experimenten sind per Skript-Aufruf oder Anleitung nachvollziehbar. ● Software Qualität <ul style="list-style-type: none"> ○ Teststrategie ○ Software Architektur ist begründet ○ Kommentare für Klassen/ Funktionen/ etc. ● Repository auf Gitlab von Enterpriselab <p>Prototype Zeugnismanager:</p> <ul style="list-style-type: none"> ● Infrastruktur <ul style="list-style-type: none"> ○ Ausführbar auf Windows ● Funktionalität <ul style="list-style-type: none"> ○ Auswahl aus mind. 5 Themen mit den Bewertungen A (hervorragend) bis D (ungenügend) und den dazugehörigen Textbausteinen. ○ Generierung von Zeugnissen ● Modelle <ul style="list-style-type: none"> ○ Alle trainierten Modelle des Demonstrators stehen in persistierter Form zur Verfügung
Gewünschte Methoden, Vorgehen:	<p>Vorgehen</p> <ul style="list-style-type: none"> ● Planung, Organisation, Risiko- und Ressourcenmanagement sind Teil der Aufgabe und werden von den Studierenden wahrgenommen ● Das Study Doc soll zu Beginn der Arbeit initialisiert und laufend nachgeführt werden. ● Während der Arbeit ist ein persönliches Arbeitsjournal zu führen. Ein Arbeitsjurnaleintrag umfasst Datum, Anzahl Stunden und Arbeitsschritt/Thema. ● Für folgende Teile muss eine Abnahme durch den Betreuer in die Planung aufgenommen werden: <ul style="list-style-type: none"> ○ Inhalt und Umfang der Daten-Korpus ○ Evaluationsmethodik (Metriken, Vorgehen) ○ Entscheid, welche Verfahren evaluiert werden ○ Minimales User Interface Design für Zeugnisgenerator <p>Study Doc: Das Führen eines Study Docs soll ein systematisches, reproduzierbares Vorgehen, insb. für die Phase der Modellbildung, des Trainings und des Hyperparametertunings, erleichtern. Es enthält eine Problembeschreibung, einen Vorgehensplan, Ideen, Experimente sowie deren Resultaten und Interpretationen. Für den Schlussbericht sollen idealerweise grosse Teile direkt aus dem Study Doc übernommen werden können.</p> <p>Zu verwendende Technologien: Evaluation: <ul style="list-style-type: none"> • Muss: Git • Vorschlag: Python, Keras mit Tensorflow/ Tensorflow, Numpy/Pandas, Jupyter Notebooks, NLTK Demonstrator: <ul style="list-style-type: none"> • Muss: Git </p>
Kreativität, Varianten, Innovation*	Das Verfahren wie auch die Anforderungen an eine mögliche Integration im Zeugnismanager sind den Studierenden überlassen. Am Ende des Projekts soll eine Empfehlung zur Verbesserung des aktuellen Zeugnismanager abgegeben werden. Mögliche Varianten wären die Generierung aufgrund grammatischer Regeln (vgl. SimpleNLG), die Umwandlung von Daten in Text (wie es z. B. für Reports oder Wetterberichte eingesetzt wird) oder das verbessern von Text mittels KI. Weitere Ideen sind erwünscht.

	Die Idee kommt aus der F&E und ist ganz klar ausserhalb des Tagesgeschäfts angesiedelt.
Schlagwörter:	Natural Language Generation (NLG), Natural Language Processing (NLP)
Wirtschaftsprojekt oder Bachelorarbeit:	<input checked="" type="checkbox"/> Wirtschaftsprojekt: 180 Stunden pro Studierender <input type="checkbox"/> Bachelorarbeit: 360 Stunden

* Bitte heben Sie in diesem Punkt hervor, inwiefern Ihre Projektidee **über kreativen Spielraum** verfügt. Dabei sind folgende Kriterien relevant: Die Idee erlaubt den Studierenden eigene Ideen zu entwickeln und Varianten zu erarbeiten, ist ausserhalb vom Tagesgeschäft angesiedelt, beinhaltet Neuland/Innovation und ist nicht durch Produkte & Tools getrieben.

Bitte kreuzen Sie eine Projektart und die zutreffenden Schwerpunkte an.

Projektarten:

- Einsatz von Standardsoftware und Services
- Software- und Produkt-Entwicklung
- Innovationsprojekte (Projekte mit Erkenntnisgewinn, Forschungsprojekte)
- IT-Infrastrukturentwicklung
- Strukturierte Analyse und Konzeption von Systemen und Abläufen

Schwerpunkte:

- Artificial Intelligence & Machine Learning
- Business Process Modelling
- Data Engineering
- Hardwarenahe Software-Erstellung
- Human Computer Interaction Design
- ICT Business Solutions
- ICT Infrastrukturen
- Internet of Things
- Mobile Systems
- Security/Privacy
- Software-Erstellung
- Visual Computing (Grafik, Bildverarbeitung, Vision, VR, AR)
- Anderes: _____

8. Rechtliche Grundlagen und Reglemente

Folgende Rechtsgrundlagen und Reglemente sind für die Wirtschaftsprojekte und Bachelorarbeiten an der Hochschule Luzern – Informatik massgebend:

- Studienordnung für die Ausbildung an der Hochschule Luzern, FH Zentralschweiz ([Link](#))
- Studienreglement für die Bachelor-Ausbildung an der Hochschule Luzern - Informatik ([Link](#))

9. Bestätigung

Mit der Kenntnisnahme der Aufgabenstellung bestätigen Student/in und Auftraggeber/in, dass

- Sie mit der Aufgabenstellung einverstanden sind.
- Auftraggebende damit einverstanden sind, dass die Hochschule Luzern – Informatik für die Organisation einer Bachelorarbeit von ihm/ihr einen Kostenbeitrag von CHF 1'000.00 (zuzgl. MWST) pro Studierende/Studierender erhebt. Dies gilt nicht für Arbeiten, welche berufsbegleitende Studierende in Verbindung mit Ihrem Arbeitsgeber machen und für HSLU interne Auftraggeber. Für die Wirtschaftsprojekte wird kein Kostenbeitrag verrechnet.

- betreuende Dozierende und Experten uneingeschränkten Einblick in die Arbeit erhalten. Auch anlässlich von Präsentationen und Marketingaktivitäten kann die Arbeit der Öffentlichkeit gezeigt werden. Eine Zusammenfassung der Arbeit wird in jedem Fall veröffentlicht. Falls das Thema vertraulich behandelt werden soll, muss der Aufgabenstellung eine entsprechende Vertraulichkeitserklärung beiliegen.

Datum: 18. September 2019

**Die definitive Aufgabenstellung (pdf-Format) bitte per E-Mail an die Transferstelle senden,
zwingend in Kopie an alle involvierten Parteien.**

Anlaufstelle für alle Informationen im Zusammenhang mit studentischen Arbeiten sowie für Entgegennahme von Projektideen & Aufgabenstellungen:

Hochschule Luzern - Informatik
Transfer Services
Suurstoffi 41b
6343 Rotkreuz

T: 041 228 24 66
E: transfer.informatik@hslu.ch

F. Beispiel Arbeitszeugnis Zeugnismanager

Arbeitszeugnis ENTWURF

Die Prima Produkte AG fertigt hochwertige Solarpanel für den industriellen Gebrauch. Unsere innovativen Lösungen helfen Unternehmen, ihre Energiebilanz zu optimieren und gleichzeitig effizienter und umweltschonender zu produzieren.

Frau Lara Meier, geboren am 13. April 1991, von Stans, war vom 28. Oktober 2011 bis 31. Oktober 2019 bei uns angestellt.

Sie arbeitete Vollzeit als Leiterin Logistik in der Abteilung Logistik und Zentrale Dienste im Bereich Corporate Center. Zu ihren Hauptaufgaben zählen:

- Führung und Organisation der unterstellten Einheiten adsfasdfasdf
- Pflege des Portfolio-Managements der eigenen Immobilien und Grundstücke im Rahmen der Immobilienstrategie
- Sicherstellung der Kosten- und Ertragsoptimierung im Zusammenhang mit der Substanzerhaltung der Immobilien
- Lehrlingsbetreuung
- neue Aufgabe auf Stelle

Lara Meier beherrschte ihren Arbeitsbereich souverän, selbständig und äusserst sicher, wobei unsere Erwartungen oft übertroffen wurden. Mit ihren Leistungen waren wir sehr zufrieden. Auch bei anspruchsvollen Problemstellungen erzielte sie gute Arbeitsergebnisse.

Ihr Arbeitsgebiet im Bereich Logistik & Supply Chain Management beherrschte Lara Meier und überzeugte zudem mit einem soliden Wissen in angrenzenden Fachgebieten. Das Fachwissen setzte sie zielgerichtet und ergebnisorientiert in der Praxis ein.

Dank ihrer schnellen Auffassungsgabe, arbeitete sich Lara Meier zügig in ihre Aufgabengebiete ein. Sie war gut organisiert und verfolgte ihre Aufgaben mit grosser Selbstdisziplin. Darüber hinaus informierte sie sich jeweils proaktiv über allfällige Änderungen der Rahmenbedingungen und passte die eigene Arbeitsweise an. Die Aufträge erledigte Lara Meier zeitnah und mit hoher Qualität. Bei Problemstellungen in ihrem Tätigkeitsgebiet trug sie wesentlich zur Lösungsfindung bei. Ihre Flexibilität wurde ausserordentlich geschätzt. Sie war Veränderungen gegenüber offen und fand sich in neuen Situationen problemlos zurecht. Lara Meier verfügte über das erforderliche Durchsetzungsvermögen, um ihre Meinung in sachlicher Art und Weise auszudrücken und bezog dabei auch Anregungen anderer mit ein. Bei wichtigen Entscheidungen formulierte sie passende Massnahmen, um die gesetzten Ziele zu erreichen.

Als Leiterin Logistik war Lara Meier zuletzt verantwortlich für 13 Mitarbeitende. Sie führte ihre Mitarbeitenden fach- und anforderungsgerecht und wurde als Vorgesetzte sehr geschätzt und respektiert. Als Führungsperson verstand sie es, ihre Mitarbeitenden auch in herausfordernden Situationen zu motivieren und ihnen Handlungssicherheit zu vermitteln. Lara Meier unterstützte die Zusammenarbeit unter den Mitarbeitenden und förderte ein konstruktives Arbeitsklima. Ihren Mitarbeitenden liess sie grossen Freiraum für deren eigenen Problemlösungen und weckte so Inspiration und Initiative für Verbesserungen und Zukunftsthemen. Der Wissenstransfer zu ihren Mitarbeitenden funktionierte dank ihrem pädagogischen Geschick und ihrer Geduld gut.

Gegenüber Vorgesetzten und Mitarbeitenden verhielt sich Lara Meier absolut vorbildlich und zeigte eine hohe Dienstleistungsbereitschaft. Sie genoss das Vertrauen unserer Kunden und wurde als kompetente und dienstleistungsorientierte Ansprechpartnerin anerkannt und geschätzt. Sie informierte zeitgerecht und wählte eine adresatengerechte Sprache. Ihre Inhalte formulierte Lara Meier prägnant und verständlich. Sie war im Team integriert und trug zur Zusammenarbeit bei. Sie akzeptierte Kritik und übte diese auch selbst aus, wenn sie der Sache diente.

Lara Meier schätzte die Folgen ihrer Handlungen überzeugend ein und berücksichtigte die damit verbundenen Chancen und Risiken ausgewogen. Sie stellte die Interessen des Gesamtunternehmens in den Vordergrund und vertrat diese gegenüber Dritten konsequent.

Lara Meier verlässt unser Unternehmen auf eigenen Wunsch per 31. Oktober 2019.

Stans, 23. Oktober 2019

Thomas Bernegger
Leiter Personal & Logistik