Stiftung Universität Hildesheim
Institut für Informationswissenschaft
und Sprachtechnologie
Lübecker Straße 3
31141 Hildesheim

Bachelorarbeit zum Thema: Multimodale Transformer für die Erkennung hasserfüllter Memes

Milan Kalkenings
kalkenin@uni-hildesheim.de
Matrikelnummer: 289248
Studiengang: Angewandte Informatik
Erstgutachter: Apl. Prof. Dr. Thomas Mandl
Zweitgutachter: Sebastian Diem

Zusammenfassung

Memes werden zumeist für konstruktive oder unterhaltende Zwecke eingesetzt, doch Trolle und menschenverachtende Gruppierungen verwenden sie zur Verbreitung ihrer hasserfüllten Weltanschauungen. Die vorliegende Arbeit greift auf spezielle neuronale Netzwerke, sogenannte Transformer, zurück, um die automatische Erkennung hasserfüllter Memes auf zwei unterschiedlichen Datensätzen zu untersuchen. Transformer erhalten Eingaben in der Form einer Vektorsequenz, dessen optimale Bildung für die Erkennung hasserfüllter Memes in dieser Arbeit angenähert werden soll. Die Versuchsergebnisse zeigen, dass die besten Transformer-Eingabesequenzen isolierte Bildregionen der Meme-Bilder als auch Textbausteine der Meme-Texte und damit assoziierte Begriffe repräsentieren. Letztere können mittels eines in dieser Arbeit definierten Verfahrens aus einem Wissensgraphen in das Netzwerk injiziert werden. Die Versuchsergebnisse weisen darauf hin, dass eine Aufgliederung des Hassspektrums und die gezielte Erkennung hasserfüllter Memes der einzelnen Hasskategorien zu einer Verbesserung der Netzwerkvorhersagen führen kann. Der verwendete Code wurde in `https://github.com/MilanKalkenings/modular_transformer` *hinterlegt.*

Inhaltsverzeichnis

1 Abkürzungsverzeichnis

AUC-ROC	Area under a Receiver Operating Characteristic Curve
BDVT	bilddublettenverwirrte Testdatenpunkte
HM	Hateful Memes
MA	Multimodal Alignment
MAMI	Multimedia Automatic Misogyny Identification
MM	Multimodal Matching
MMLM	Multimodal Masked Language Modeling
MMSM	Multimodal Masked Scene Modeling
MQA	Multimodal Question Answering
PK	Positionskodierung
SBE	Sequential Backward Elimination
SDVT	semidublettenverwirrte Testdatenpunkte
SFS	Sequential Forward Elimination
TDVT	textdublettenverwirrte Testdatenpunkte
[CLS]	Classification-Token
[PAD]	Padding-Token
[SEP]	Separation-Token
[UNK]	Unknown-Token

2 Einleitung

Das Internet bietet viele Möglichkeiten zur Verbreitung des eigenen Gedankenguts. Veröffentlichte Inhalte können zu einem beliebigen späteren Zeitpunkt aufgegriffen und für eine Vielzahl unterschiedlicher Zwecke genutzt werden. Eine häufig verwendete Form zur Kommunikation der eigenen Gedankenwelt im Internet stellt die Verwendung sogenannter Memes dar. Laut (Osterroth, 2015) ist ein Meme eine Kombination aus einer Bildkomponente und einer Textkomponente, welche politische, popkulturelle oder alltägliche Themen behandelt und rekontextualisiert. Der Meme-Text und das Meme-Bild werden dabei von (Kiela et al., 2020a) als Meme-Modalitäten interpretiert, deren Semantiken miteinander verwoben werden müssen, um ein vollständiges Verständnis des Memes zu gewährleisten. Die überwiegende Mehrheit der Meme-Ersteller fokussiert sich auf die Verbreitung informativer oder unterhaltender Inhalte, doch einige Akteure setzen diffamierende Memes ein, um ihre rassistischen, sexistischen, demokratiefeindlichen oder andersartig zersetzenden Weltanschauungen zu propagieren (Kiela et al., 2020a; Fersini et al., 2022). Beispiele für derartige diffamierende Memes stellen die *Free Helicopter Rides*-Memes dar, welche die Ermordung politischer Gegner durch das Pinochet-Regime glorifizieren (KnowYourMeme, 2021). Somit können Memes ein Medium zur Verbreitung von Hassrede darstellen, welche laut (Bilewicz und Soral, 2020) als Motivation für die Terroranschläge 2018 in Pittsbugh und 2019 in El Paso und Christchurch diente.

Die vorliegende Arbeit befasst sich mit der Klassifikation von Memes in die beiden Kategorien *hasserfüllt* und *harmlos*. Einerseits wird überprüft, welche Meme-Merkmale von speziellen künstlichen neuronalen Netzwerken, sogenannten multimodalen Transformern (Lu et al., 2019), erlernt werden sollten, um eine möglichst genaue Klassifikation zu gewährleisten. Andererseits wird untersucht, welche Unterschiede zwischen zwei Datensätzen für die Durchführung einer derartigen Klassifikation bestehen, inwiefern eine gegenseitige Nutzbarmachung der Datensätze erzielt werden kann und welche Anforderungen an zukünftige Meme-Datensätze aus den durchgeführten Experimenten abgeleitet werden können.

2.1 Künstliche neuronale Netzwerke

Zunächst werden einige theoretische Grundlagen erläutert, welche für das Verständnis dieser Arbeit benötigt werden. Bei einem künstlichen neuronalen Netzwerk handelt es sich um eine Verkettung mehrdimensionaler Abbildungen, wobei einzelne oder aufeinanderfolgende Kettenglieder auch als Module bezeichnet werden (Torch Contributers, 2019). Die lineare Abbildung $f(x) = Ax + b$, welche Vektoren $x \in \mathbb{R}^n$ mittels der beiden Parameter $A \in \mathbb{R}^{m,n}$ und $b \in \mathbb{R}^m$ auf einen Ausgabevektor projiziert, sowie das eindimensionale Embedding stellen die beiden im Rahmen dieser Arbeit am häufigsten verwendeten Netzwerkmodule dar. Das eindimensionale Embedding kann als eine Variante der linearen Abbildung verstanden werden, welches jedoch natürliche Zahlen als Eingabe erhält. Im Verlauf dieser Arbeit werden weitere Module und ihre Funktionsweisen erläutert. Um ein neuronales Netzwerk für die Klassifikation von Memes einsetzen zu können, müssen die Parameter seiner Netzwerkmodule mittels Training angepasst werden, wofür annotierte Memes benötigt werden (Goodfellow et al., 2016, S. 98). Im vorliegenden Fall gibt die Annotation eines Memes an, ob es hasserfüllt oder harmlos ist. Bevor das Training beginnen kann, müssen die annotierten Memes auf einen Trainings-, einen Validierungs- und einen Test-Datensatz aufgeteilt werden (Baheti, 2021). Während des Trainings wird jedes Meme des Trainingsdatensatzes in das Netzwerk eingegeben und zu einer Netzwerkvorhersage verarbeitet (Goodfellow et al., 2016, S. 165). Sie gibt an, ob das Netzwerk das eingegebene Meme als hasserfüllt oder als harmlos klassifiziert. Weicht die Netzwerkvorhersage von der Meme-Annotation ab, so wird mittels des gradientenbasierten Backpropagation-Algorithmus (Werbos, 1974) ermittelt, inwiefern die Parameter des neuronalen Netzwerkes angepasst werden müssen, um die Vorhersage des Netzwerkes an die Meme-Annotation anzupassen. Durch dieses Verfahren erlernt das Netzwerk die Merkmale hasserfüllter sowie die Merkmale harmloser Memes des Trainingsdatensatzes. In der Regel muss dieses Verfahren mehrfach durchgeführt werden, um ein ausreichendes Merkmalsverständnis zu gewährleisten. Jede Wiederholung des Trainingsprozesses wird dabei als Trainingsepoche bezeichnet (Baheti, 2021).

Mit jeder Trainingsepoche wird das neuronale Netzwerk stärker an die Trainingsdaten angepasst, was nach einer zuvor unbekannten Epochenanzahl dazu führen kann, dass nicht-generalisierende Merkmale der Trainingsdaten, wie exakte Pixelintensitäten, auswendig gelernt werden. Ein solches Trainingsergebnis wird als Overfitting bezeichnet und führt zu einer Verschlechterung der Netzwerkvorhersagen außerhalb der Trainingsdaten. Um zu erkennen, ab welchem Zeitpunkt das Netzwerk zu stark an die Trainingsdaten angepasst ist, werden zwischen den Trainingsepochen Netzwerkvorhersagen auf den Validierungsdaten durchgeführt. Verschlechtern sich diese, so wird das Training vorzeitig abgebrochen. (Goodfellow et al., 2016, S. 241ff)

Die Anzahl der Trainingsepochen ist somit ein sogenannter Hyperparameter, dessen optimaler Wert nicht auf den Trainingsdaten erlernt werden kann, sondern auf den Validierungsdaten bestimmt wird (Baheti, 2021). Weitere Hyperparameter stellen beispielsweise die Anzahl verwendeter Netzwerkmodule oder Modulparameter dar. Um sicherzustellen, dass die identifizierten Hyperparameter auch auf bisher ungesehenen Daten zu einem zufriedenstellenden Ergebnis führen, wird das neuronale Netzwerk zuletzt auf dem bisher ungenutzten Testdatensatz evaluiert (Baheti, 2021). Somit gilt für den Trainingsprozess:

Netzwerkparameter werden auf dem Trainingsdatensatz erlernt, optimale Hyperparameter auf den Validierungsdaten ermittelt und das finale neuronale Netzwerk auf dem Testdatensatz evaluiert (Baheti, 2021).

Vorangegangene Forschung zeigt, dass die Verkettung mehrerer Trainingsprozesse zu besseren Resultaten führen und einst erlernte Zusammenhänge für die Lösung weiterer Aufgaben genutzt werden können. Dafür werden zuvor erlernte Parameter als Netzwerkinitialisierung für aufbauende Trainingsprozesse eingesetzt. Der Aufwand, welcher zur Erlernung sinnhafter Initialisierungsparameter betrieben wird, wird dabei als Vortraining bezeichnet. (Radford et al., 2018, Devlin et al., 2019)

2.2 Zielsetzung und Beiträge dieser Arbeit

In der vorliegenden Arbeit werden multimodale Transformer-Netzwerke für die Klassifikation von Memes verwendet. Transformer sind spezielle neuronale Netzwerke, welche eine Eingabesequenz aus $N \in \mathbb{N}$ reellwertigen Vektoren der Größe $P \in \mathbb{N}$ erhalten und die semantischen Beziehungen zwischen den einzelnen Sequenzpositionen berechnen (Vaswani et al., 2017). Multimodale Transformer können sowohl textuelle als auch visuelle Eingabedaten miteinander in Beziehung setzen (Lin et al., 2021). Das vorrangige Ziel dieser Arbeit ist die Ermittlung der optimalen Verfahren zur Bildung einer multimodalen Transformer-Eingabesequenz für die Unterscheidung von hasserfüllten und harmlosen Memes. Um dieses Ziel zu erreichen, wurden folgende Arbeitsschritte unternommen:

- Die Identifikation der vier verbreitetsten Formen zur Berechnung der Eingabesequenz multimodaler Transformer. Dafür wurde eine intensive Recherche durchgeführt, dessen Nebenerzeugnis der in Anhang 1 zur Verfügung gestellte Vergleich von Transformern, Datensätzen und Vortrainingsmethoden darstellt.
- Die Implementierung der vier häufigsten Verfahren zur Berechnung der Eingabesequenzen multimodaler Transformer.
- Die Implementierung einer Methode, welche die Stimmungslage textueller Daten in den Positionen einer Transformer-Eingabesequenz repräsentiert.
- Die Implementierung eines Verfahrens, welches domänenspezifisches Kontextwissen aus einem speziellen Wissensgraphen entnimmt und als Positionen einer Transformer-Eingabesequenz darstellt.
- Die Erstellung eines multimodalen Transformer-Netzwerkes, welches separate Module zur Durchführung der sechs soeben genannten Verfahren zur Bildung der eigenen Eingabesequenz enthält.

- Eine umfassende Untersuchung des Zusammenspiels und der Effektivität der mittels der sechs Verfahren erhobenen Positionen der Transformer-Eingabesequenz.

Ein weiteres Ziel dieser Arbeit ist es, zwei Datensätze, welche sowohl harmlose als auch hasserfüllte Memes enthalten, miteinander in Beziehung zu setzen und Anforderungen an zukünftige Datensätze dieser Forschungsrichtung zu formulieren. Die dafür durchgeführten Methoden beinhalten:

- Eine einheitliche Vorverarbeitung und eine Bestimmung der Ähnlichkeit der beiden Datensätze.
- Eine Überprüfung, inwiefern die Netzwerkvorhersagen auf beiden Datensätzen durch datensatzübergreifende Trainingsprozesse verbessert werden können.
- Die Identifikation schwer zu klassifizierender Memes, basierend auf einer Analyse der Vorhersagefehler der verwendeten Netzwerke.

Um die umgesetzten Methoden und die daraus gewonnenen Erkenntnisse in einer rahmengebenden Struktur zu präsentieren, gliedert sich die vorliegende Arbeit wie folgt:

Im nächsten Kapitel werden verwandte Forschungsartikel herangezogen, um die vorliegende Arbeit in dem wissenschaftlichen Kontext zu verorten, zuvor gewonnene Erkenntnisse zusammenzufassen und offene Forschungsfragen zu ermitteln. Im Anschluss daran wird die Implementierung des verwendeten Transformers beschrieben, bevor der Versuchsaufbau und dessen Ergebnisse in den beiden darauffolgenden Kapiteln behandelt werden. Im vorletzten Kapitel werden die Ergebnisse kritisch hinterfragt und interpretiert, Einschränkungen dieser Arbeit beleuchtet und zukünftige Forschungsrichtungen vorgeschlagen, bevor die gewonnen Erkenntnisse im letzten Kapitel zusammengefasst werden.

3 Aktueller Forschungsstand

3.1 Multimodale Transformer

Die Verwendung multimodaler Transformer für die gemeinsame Verarbeitung textueller und visueller Daten basiert auf zuvor gewonnenen Erkenntnissen. Die klassische Herangehensweise an die Verarbeitung visueller Daten beruht auf der Verwendung konvolutionaler neuronaler Netzwerke, welche Konvolutionen für die Extraktion visueller Merkmale nutzen (Krizhevsky et al., 2012). Die Verarbeitung textueller Daten hingegen stützte sich zunächst auf die Verwendung rekurrenter neuronaler Netzwerke, welche ihre Eingabe als eine Sequenz von Textbausteinen verstehen und die darin enthaltenen Informationen iterativ zu einem gemeinsamen Kontext zusammenfügen (Elman, 1990; Hochreiter und Schmidhuber, 1997). Die Effektivität rekurrenter Netzwerke für die Verarbeitung textueller Daten wurde unlängst in einer Vielzahl von Anwendungsfällen durch die der Text-Transformer (Vaswani et al., 2017) übertroffen.

Analog zu ihren Vorgängern repräsentieren Text-Transformer ihre Eingabe als eine Sequenz von Textbausteinen, welche mittels eines Vektors repräsentiert werden. Die Überlegenheit der Transformer fußt auf ihrer parallelen Verarbeitung der Eingabesequenz, welche eine feingliedrige In-Beziehung-Setzung der Sequenzpositionen ermöglicht und mittels des sogenannten Attention-Mechanismus durchgeführt wird. (Vaswani et al., 2017; Devlin et al., 2019)

Die Leistung der Text-Transformer hat zu einem wachsenden wissenschaftlichen Interesse an ihrer Nutzbarmachung für die Verarbeitung visueller Daten geführt (Dosovitskiy et al., 2020). Die größte Hürde stellte dabei die Repräsentation visueller Daten als transformer-typische Eingabesequenz dar, welche jedoch unter anderem durch (Dosovitskiy et al., 2020) umgesetzt und zur Erzielung überzeugender Ergebnisse genutzt werden konnte. Somit ergab sich die Möglichkeit, sowohl Text-Daten als auch Bild-Daten in der gleichen Eingabeform darzustellen, mittels eines Transformers zu verarbeiten und überzeugende Ergebnisse zu erzielen. Diese Ausgangslage führte zu der Konzipierung einer Reihe multimodaler Transformer, welche Text-

Eingabesequenzen und Bild-Eingabesequenzen gemeinsam verarbeiten können. Sie unterscheiden sich vorrangig anhand der Bildung ihrer Eingabesequenzen, ihrer Vortrainingsmethoden und Vortrainingsdatensätze sowie der Fusion ihrer Modalitäten. Letztere gliedert sich in *early fusion*, die Kombination der Modalitäten während der Bildung der Eingabesequenz (Su et al., 2020), und *late fusion*, der Verwendung einer separaten Eingabesequenz für jede Modalität (Lu et al., 2019).

In Anhang 1 werden 16 verschiedene multimodale Transformer-Netzwerke miteinander verglichen. Es zeigt sich, dass die analyiserten Transformer auf lediglich vier verschiedene Verfahren zur Bildung ihrer Eingabesequenzen zurückgreifen. Eingabetexte werden dabei von allen untersuchten Transformern entsprechend der ursprünglichen Text-Transformer als Vektorrepräsentationen aufeinanderfolgender Textbausteine dargestellt (Qi et al., 2020). Für die Repräsentation von Bildeingaben hingegen wird auf drei unterschiedliche grundlegende Verfahren zurückgegriffen. Sie werden mittels der visuellen Merkmale automatisch erkannter Objekte (Wang et al., 2020a; Zhang et al., 2020b; Zhou et al., 2020), isoliert verarbeiteter gleichgroßer Bildkacheln (Lin et al., 2021; Gao et al., 2020) oder des gesamten Eingabebildes (Huang et al., 2020) repräsentiert. In Anhang 1 werden die unterschiedlichen Ausprägungen dieser Verfahren zur Bildung der Positionen der Transformer-Eingabesequenzen genauer beleuchtet. Die Analyse der multimodalen Transformer ergab außerdem, dass ihre zahlreichen Vortrainingsverfahren in fünf übergeordnete Kategorien unterteilt werden können. In Anhang 1 werden die Eigenschaften dieser Vortrainingskategorien beschrieben, ihre Eignung zur Anwendung auf Meme-Datensätzen kritisch beurteilt und für die 16 Transformer festgehalten, auf welche Vortrainingskategorien sie zurückgreifen, welche Datensätze dafür verwendet wurden und welche grundlegenden Eigenschaften diese Datensätze besitzen.

3.2 Erkenntnisse aus der Hateful Memes Challenge

Teilnehmer der sogenannten Hateful Memes Challenge (Kiela et al., 2020a) erhielten einen Datensatz, welcher zur Erkennung hasserfüllter Memes mittels multimodaler Netzwerke genutzt werden sollte. Die Verfahren der Challenge-Sieger basieren auf einigen wiederholt gewonnenen Erkenntnissen. (Lippe et al., 2020) identifizieren, dass UNITER (Chen et al., 2020) bessere Ergebnisse als LXMERT (Tan und Bansal, 2019) und Oscar (Li et al., 2020b) hervorbringt und vermuten, dass dies auf dessen intensivem Vortraining beruht. (Velioglu und Rose, 2020) nutzen VisualBERT (Li et al., 2019) und stellen fest, dass ein Vortraining auf Conceptional Captions (CC) (Sharma et al., 2018) zu besseren Netzwerkvorhersagen führt als das ursprüngliche VisualBERT-Vortraining auf COCO (Lin et al., 2014). Darüber hinaus werden die Netzwerkvorhersagen in (Sandulescu, 2020) durch ein Ensemble mehrerer Transformer gebildet, um von unterschiedlichen Vortrainingsergebnissen zu profitieren. Zusammenfassend zeigt sich, dass die Netzwerkauswahl häufig in Abhängigkeit von den damit durchgeführten Vortrainingsmethoden und nicht auf Basis der Verfahren zur Bildung ihrer Eingabesequenzen geschah.

Für die Erzielung guter Netzwerkvorhersagen fokussierten sich die Challenge-Sieger zudem auf die Anreicherung der Memes durch domänenspezifisches Kontextwisssen. (Zhu, 2020) gibt seinem Netzwerk Informationen über automatisch erkannte Entitäten in Meme-Texten sowie die Ethnie und das Geschlecht von erkannten Personen in Meme-Bildern und beschreibt diese zusätzlichen Kontextinformationen als Weltwissen, welches laut ihm eine dritte Meme-Modalität darstellt (Zhu, 2020). Laut (Zhu, 2020) bezieht sich etwa die Hälfte der hasserfüllten Memes des Challenge-Datensatzes auf die Ethnie oder das Geschlecht von Personen, wodurch das Hinzufügen derartiger Kontextinformationen einen Vorteil bringen kann. Analog dazu verwenden (Lippe et al., 2020) eine Software zur Erkennung weiterer Aspekte der Objekte im Eingabebild, darunter ebenfalls die Ethnie der dargestellten Personen. Sowohl (Zhu, 2020), als auch (Lippe et al., 2020) repräsentieren diese zusätzlichen Informationen als Positionen der Transformer-Eingabesequenz.

Ungeklärt bleibt, welche Verfahren zur Bildung der Eingabesequenz am besten geeignet sind, um hasserfüllte Memes zu erkennen. Eine Entscheidung anhand der Challenge-Sieger ist nicht möglich, weil die variablen Vortrainingsintensitäten einen direkten Vergleich der Verfahren zur Bildung der Eingabesequenzen verhindern. Die besten Verfahren können nur bei vergleichbarer Vortrainingsintensität ermittelt werden und ein Transformer mit einer überlegenen Eingabesequenz kann die erzielten Ergebnisse mit ausreichendem Vortraining gegebenenfalls übertreffen. Auch bleibt offen, welche zusätzlichen Methoden zur Einspeisung von domänenspezifischem Kontextwissen eine Verbesserung der Netzwerkvorhersagen nach sich ziehen. Darüber hinaus stellt sich die Frage, ob die Bearbeitung der Challenge durch das Hinzuziehen großer Mengen zusätzlicher Meme-Daten, in der Form eines zweiten Datensatzes, erleichtert werden kann. Es zeigt sich, dass die Zielsetzung dieser Arbeit im Einklang mit der Beantwortung der offenen Fragestellungen steht.

4 Ein Transformer zur Beantwortung der Forschungsfragen

Dieses Kapitel beschreibt einen zur Beantwortung der Forschungsfragen implementierten Early-fusion-Transformer. Die Verarbeitung eines Memes innerhalb dieses Transformers kann in vier grundlegende Schritte unterteilt werden:

Im ersten Verarbeitungsschritt werden die linguistischen und visuellen Merkmale des Memes mittels unterschiedlicher Netzwerkmodule extrahiert und in der Form von Positionen einer Transformer-Eingabesequenz repräsentiert. Um eine individuelle Gütebeurteilung der unterschiedlichen Verfahren zu ermöglichen, lassen sich die Netzwerkmodule von dem Transformer abkoppeln, wodurch sie keine Berechnungen durchführen. Im zweiten Verarbeitungsschritt werden die Ausgaben der angekoppelten Module mittels einiger Angleichungsverfahren zu einer gemeinsamen Transformer-Eingabesequenz fusioniert. Darauffolgend werden die Positionen der Transformer-Eingabesequenz

durch einen Transformer-Kodierer (Vaswani et al., 2017) miteinander in Beziehung gesetzt und zu einer Transformer-Ausgabesequenz verwoben. Im letzten Verarbeitungsschritt wird die Transformer-Ausgabesequenz zur Bildung einer Netzwerkvorhersage genutzt. In den folgenden Abschnitten dieser Arbeit werden die zuvor genannten Verarbeitungsschritte näher erläutert. Eine Übersicht über das Netzwerk kann Abbildung 9 entnommen werden.

4.1 Die Berechnung von Positionen der Transformer-Eingabesequenz

Die sechs, in diesem Abschnitt beschriebenen Netzwerkmodule können dafür verwendet werden um Merkmale multimodaler Eingabedaten in der Form von Positionen der Transformer-Eingabesequenz darzustellen.

4.1.1 Das Token-Modul

Dieses Modul erstellt Positionen der Transformer-Eingabesequenz, indem es Textbausteine des Eingabetextes auf Vektorrepräsentationen abbildet und diese um syntaktische Informationen ergänzt. Eine solche Textrepräsentation kann dazu genutzt werden, um die semantischen Beziehungen innerhalb des Eingabetextes zu erlernen (Devlin et al., 2019). Nach dessen erstmaliger Verwendung in (Devlin et al., 2019) wurde dieses Modul in unzähligen, darunter auch allen in Anhang 1 zusammengefassten, Transformern verwendet. Im Rahmen dieser Arbeit wird auf die in (Hyugen AI, 2021) zur Verfügung gestellte Implementierung eines solchen Moduls zurückgegriffen.

Die Textbausteine des Eingabetextes sind sogenannte Tokens, welche Wortuntereinheiten oder einzelne Wörter repräsentieren. Der Eingabetext wird mittels eines Tokenizers (Hugging Face, 2021a) auf eine Tokensequenz abgebildet und durch spezielle syntaktische Tokens ergänzt. Die ersten beiden syntaktischen Tokens sind das Classification- ([CLS]) und das Separation-Token ([SEP]), welche vor den Anfang beziehungsweise hinter das Ende des Eingabetextes platziert werden. Die Verwendung dieser beiden Tokens ermöglicht es Transformern zu erkennen, wo der Eingabetext beginnt und wo er aufhört. Darüber hinaus können diese Tokens für spezielle Trainingsver-

fahren eingesetzt werden, auf welche in dieser Arbeit nicht näher eingegangen wird. Zeichenketten, welche exotische Begriffe oder Rechtschreibfehler beinhalten, werden gegebenenfalls nicht vom Tokenizer erkannt und somit auf das Unknown-Token ([UNK]) abgebildet. Die Länge der Tokensequenz wird über den Hyperparameter $N_{Token} \in \mathbb{N}$ festgesetzt, um eine parallele Verarbeitung mehrerer Datenpunkte zu ermöglichen. Werden mehr als N_{Token} Tokens identifiziert, so werden die überzähligen Tokens vom Satzende ausgehend entfernt. Falls weniger Tokens erkannt werden, ergänzt der Tokenizer die Tokensequenz durch Padding-Tokens ([PAD]), welche hinter [SEP] angehängt werden. Nach ihrer Erkennung durch den Tokenizer werden die Tokens auf ihren eindeutigen Identifizierer abgebildet, woraus sich eine Identifiziererse-quenz der Form $T_0, T_1, ..., T_{N_{Token}-1}$ mit $T_n \in \mathbb{N}$ ergibt. (Devlin et al., 2019)

Um eine Vektorrepräsentation für jedes Token erlernbar zu machen wird, wie in (Devlin et al., 2019), ein eindimensionales Embedding auf die Identifiziersequenz angewendet. Dadurch werden die umgebenden Tokens, anders als in Verfahren wie Word2Vec (Mikolov et al., 2013), nicht direkt in die Erlernung der Vektorrepräsentationen involviert. Eine Berücksichtigung anderer Tokens geschieht stattdessen implizit durch die Anwendung des Backpropagation-Algorithmus auf die Netzwerkvorhersage, welche auf Basis einer In-Beziehung-Setzung der einzelnen Tokens geschieht.

(Devlin et al., 2019) folgend wird eine eindeutige Positionskodierung (PK) auf jedes Tokenembedding addiert, um dem Transformer die Möglichkeit zu geben, räumliche Distanzen innerhalb der Satzstruktur zu erlernen. Dieser Schritt ist notwendig, weil Transformer nicht implizit dazu in der Lage sind, die Reihenfolge ihrer Eingabesequenz zu erkennen (Vaswani et al., 2017). Vergleichbare Verfahren für die Positionskodierung finden sich auch in späteren Abschnitten dieses Kapitels. Sie werden stets auf die mit ihnen korrespondierenden Vektorrepräsentationen addiert und stellen somit keine eigenen Positionen der Transformer-Eingabesequenz dar. Eine beispielhafte Anwendung des Token-Moduls mit $N_{Token} = 10$ ist in Abbildung 1 dargestellt.

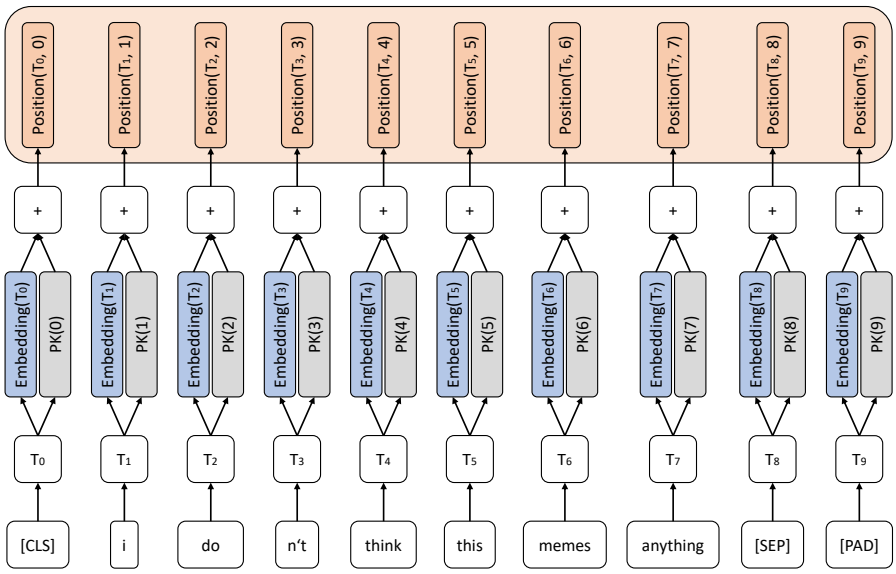

Abbildung 1: Die Anwendung des Token-Moduls

4.1.2 Das Assoziations-Modul

Das Assoziations-Modul erstellt eine Position der Transfromer-Eingabesequenz für jede hinterlegte Assoziation zu den Wörtern im Eingabetext, wodurch eine Anreicherung des Textkontextes und eine Unterstützung des Meme-Verständnisses erzielt werden soll. Dabei greift das Modul auf einen hinterlegten Wissensgraphen (Ontotext, 2022) zurück. Er beinhaltet unterschiedliche Begriffe und ungerichtete unbeschriftete Assoziationsrelationen. Werden zwei Begriffe innerhalb des Wissensgraphen miteinander assoziiert, so werden sie mittels einer Assoziationsrelation miteinander verknüpft. Um den Wissensgraphen zu bilden, wurden einige Memes durch einen Annotator untersucht und Assoziationen zweier unterschiedlicher Kategorien festgehalten:

Assoziationen der ersten Kategorie stellen diskriminierende Vorurteile dar, welche durch hasserfüllte Memes vermittelt werden. Bei den Assoziationen der zweiten Kategorie handelt es sich um triviales Weltwissen, welches das Verständnis der Memes vereinfacht.

In Abschnitt 5.1 werden zwei Trainingsdatensätze basierend auf den in (Kiela et al., 2020a) und (Fersini et al., 2022) vorgestellten Datensätzen gebildet. Der resultierende Wissensgraph enthält 157 Assoziationen, welche aus

Memes des ersten, und 15 weitere Assoziationen, welche aus Memes des zweiten Trainingsdatensatzes extrahiert wurden. Dadurch ist der Wissensgraph stärker an die Memes des in (Kiela et al., 2020a) beschriebenen Datensatzes angepasst, wodurch im Anschluss an die durchgeführten Experimente überprüft werden kann, ob eine stärkere Datensatzanpassung eine Verbesserung dieses Moduls hervorbringt. Ein beispielhafter Ausschnitt des verwendeten Wissensgraphen ist in Abbildung 2 dargestellt. Die darin enthaltene Assoziation zwischen den Begriffen *obama* und *democrat* stellt triviales Weltwissen dar, wohingegen die Assoziation zwischen *obama* und *illegal* ein rassistisches Vorurteil widerspiegelt, welches durch einige der Memes vermittelt werden soll.

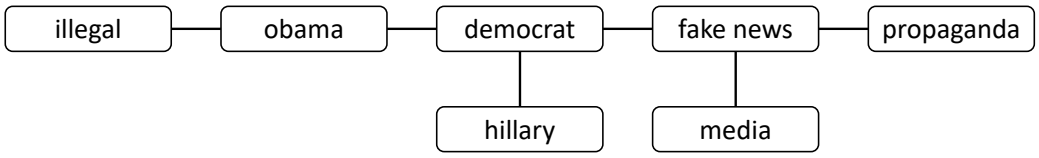

Abbildung 2: Ein beispielhafter Ausschnitt des verwendeten Wissensgraphen

Das Assoziations-Modul funktioniert ähnlich wie das Token-Modul. Anstatt jedoch eine Repräsentation für jedes Token im Eingabetext zu erstellen, bildet es eine Repräsentation für jedes Token, welches eine hinterlegte Assoziation zu einem der Tokens im Eingabetext enthält. Wird ein Wort mit dem siebten Token des Eingabetextes assoziiert, so wird es durch die Summe des eindimensionalen Embeddings des eigenen Tokenidentifizierers und der Positionskodierung des siebten Tokens des Eingabetextes repräsentiert. Durch dieses Verfahren soll erzielt werden, dass der Transformer erkennt, zwischen welchen Tokens die Assoziation besteht.

Das Assoziations-Modul erstellt $N_{Assoziation} \in \mathbb{N}$ Positionen der Transformer-Eingabesequenz. Werden mehr als $N_{Assoziation}$ Assoziationen gefunden, so werden die überzähligen Assoziationen verworfen. Werden weniger Assoziationen gefunden, so werden die restlichen Positionen durch Padding-Tokens gefüllt.

Bisher wurde noch kein Modul für die Erstellung assoziationsbasierter Transformer-Eingabesequenzpositionen erstellt, jedoch liefern (Liu et al., 2019) ein ähnliches Verfahren. (Liu et al., 2019) reichern einzelne Positionen einer Variante des Token-Moduls durch die Semantik von Begriffen aus einem weitläufigen Wissensgraphen an, wodurch die restliche Eingabesequenz nur indirekt von der Wissensinjektion profitiert. Die vom Assoziations-Modul erstellten Positionen der Transformer-Eingabesequenz hingegen können direkt mit allen anderen Positionen in Beziehung gesetzt werden. Dazu wurde sich bewusst entschieden, weil der in dieser Arbeit verwendete Wissensgraph lediglich zielgerichtete Informationen für die Interpretation von Memes beinhaltet und eine Erweiterung der Transformer-Eingabesequenz durch derartiges Wissen zu keiner unnötigen Aufblähung und daraus resultierender Verzerrung des Textkontextes führt. Die Anwendung des Assoziations-Moduls auf einen Eingabetext mit $N_{Assoziation} = 3$ wird in Abbildung 3 dargestellt. Die roten Zahlen verdeutlichen die Anwendung der Positionskodierung.

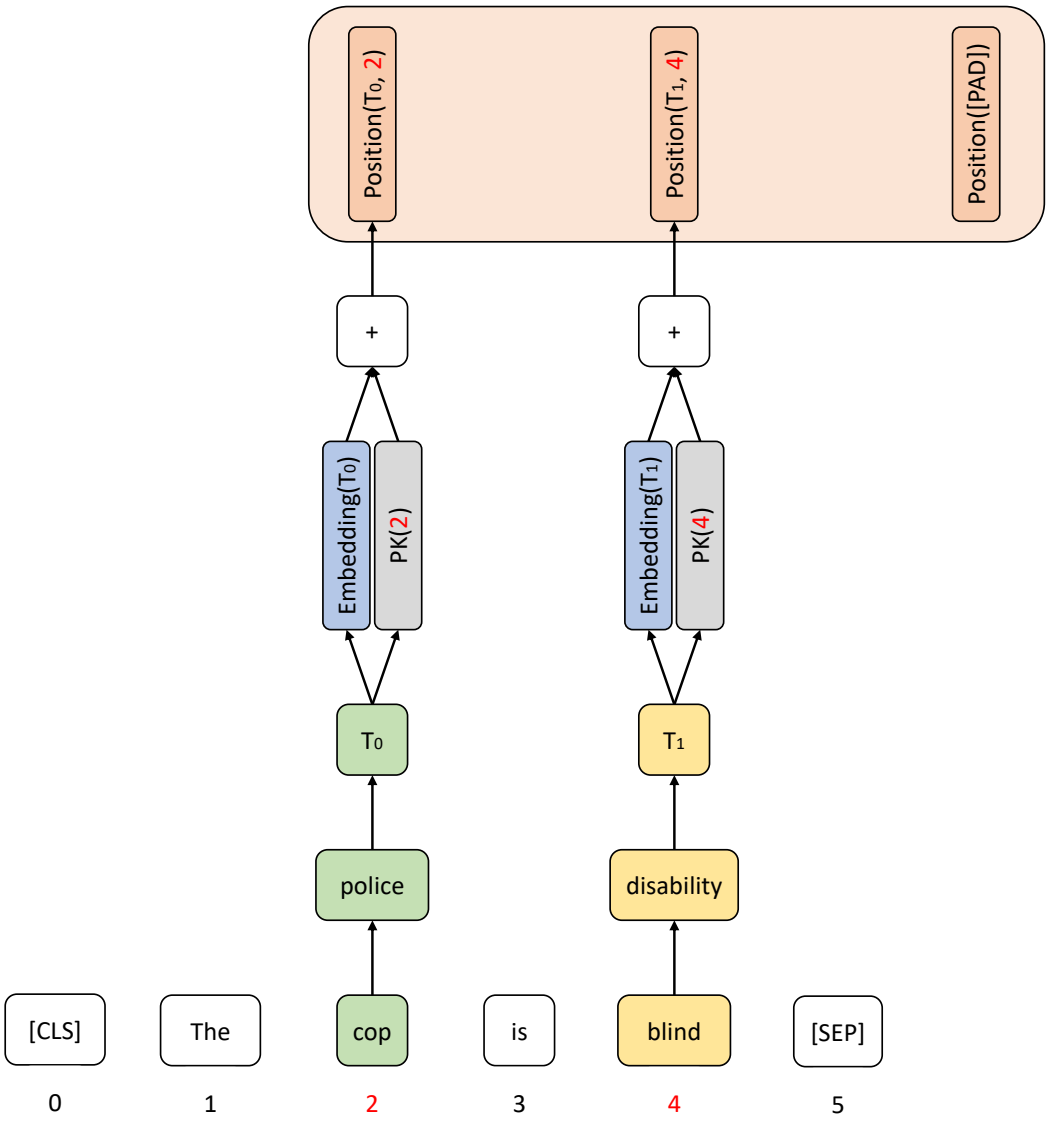

Abbildung 3: Die Anwendung des Assoziations-Moduls

4.1.3 Das Stimmungs-Modul

Dieses Modul kodiert die Stimmungslage eines Eingabetextes in der Form einer Position der Transformer-Eingabesequenz. Eine derartige Textrepräsentation wurde bisher von keinem Transformer verwendet, jedoch wurde im Rahmen dieser Arbeit angenommen, dass die Texte hasserfüllter Memes zu einer negativeren Stimmungslage tendieren könnten als die Texte ihrer harmlosen Gegenstücke.

Dieses Modul ermittelt die Stimmungslage durch die Berechnung von sechs unterschiedlichen Merkmalen. Die ersten beiden Merkmale werden durch die in (Hutto, 2021; Loria, 2020) implementierten Verfahren gewonnen und berechnen die gemittelte Gesamtstimmung des Eingabetextes. Um emotionale Spitzen aufzuzeichnen, werden zusätzlich sowohl die maximal positive als auch die maximal negative Wortstimmung mittels weiterer durch (Loria, 2020) zur Verfügung gestellter Methoden ermittelt. Das fünfte Stimmungsmerkmal ist die, ebenfalls mittels (Loria, 2020) berechnete, Subjektivität des Eingabetextes. Bei dem letzten Merkmal handelt es sich um den Anteil an Reizwörtern im Eingabetext, wobei alle möglicherweise diskriminierenden Wörter in der durch (von Ahn, o. D.) zur Verfügung gestellten Liste als Reizwörter interpretiert werden. Wie in Abbildung 4 dargestellt, werden alle sechs Stimmungsmerkmale in einem Vektor zusammengefasst und mittels einer linearen Abbildung zu genau einer Position einer Transformer-Eingabesequenz verarbeitet.

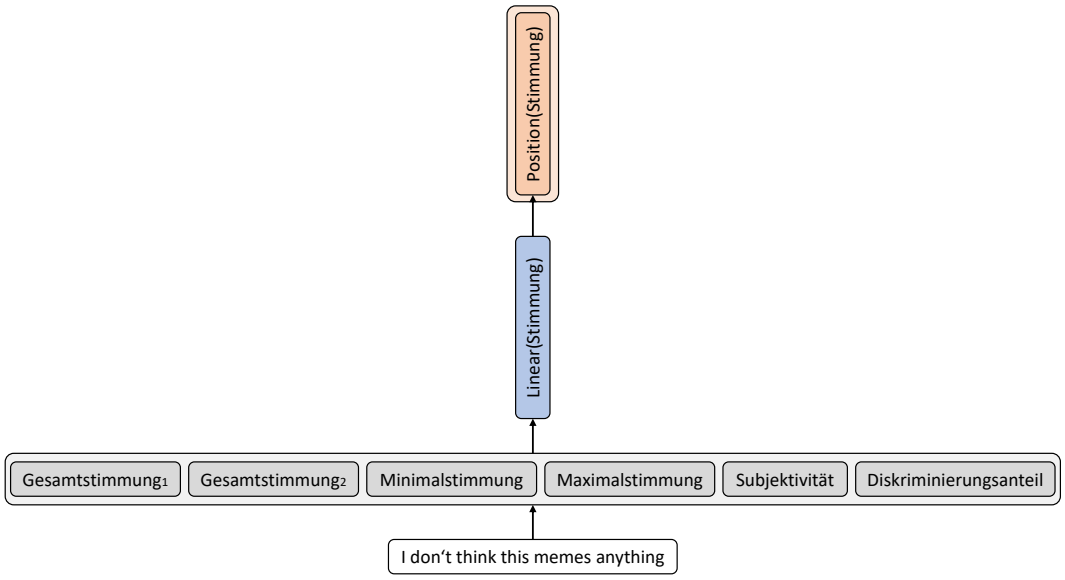

Abbildung 4: Die Anwendung des Stimmungs-Moduls

4.1.4 Das Objekt-Modul

Das Objekt-Modul berechnet die visuellen Merkmale automatisch erkannter Objekte in Eingabebildern und stellt diese als Positionen der Transformer-Eingabesequenz dar. Im Rahmen dieser Arbeit können die so ermittelten Merkmale dafür genutzt werden, den Fokus des Transformers von unwichtigen Bildregionen abzuwenden und zusätzliche Informationen über die Entitäten im Eingabebild zur Verfügung zu stellen.

Für die Objekterkennung wurde das in (Torch Contributers, 2017) zur Verfügung gestellte konvolutionale Faster R-CNN-Netzwerk (Ren et al., 2016) und die in (Rosebrock, 2021) implementierten Algorithmen genutzt. Faster R-CNN identifiziert rechteckige informationsschwangere Bildregionen, sogenannte Regions of Interest (RoI), errechnet deren visuelle Merkmale und bestimmt auf Grundlage dessen die Vorkommenswahrscheinlichkeiten der ihm bekannten Objektklassen in den RoI. Eine beliebige RoI_n wird im Rahmen dieser Arbeit, neben ihren Pixelintensitäten, durch mehrere Werte repräsentiert:

$(x_{1,n}, y_{1,n})$ sind die Koordinaten ihrer oberen linken und $(x_{2,n}, y_{2,n})$ ihrer

unteren rechten Ecke. w_n und h_n sind die Breite und die Höhe des Eingabebildes, welches RoI_n enthält. $i_n \in \mathbb{N}$ ist der Identifizierer der in RoI_n identifizierten Objektklasse und $p_{i_n} \in \mathbb{R}$ ist die dafür vergebene Vorkommenswahrscheinlichkeit. Im Rahmen dieser Arbeit gelten Objekte als automatisch erkannt, wenn ihre errechneten Vorkommenswahrscheinlichkeiten bei mehr als 90 % liegen. In Abbildung 5 wird eine Objekterkennung durch das verwendete Netzwerk auf einem durch (Fersini et al., 2022) zur Verfügung gestellten Meme dargestellt. Die roten Rechtecke stellen die RoI dar, während ihre Beschriftungen die automatisch erkannten Objektklassen wiedergeben.

Aus Anhang 1 geht hervor, dass die meisten multimodalen Transformer auf eine solche Objekterkennung zurückgreifen und die vom Faster R-CNN errechneten visuellen Merkmale der RoI als Repräsentationen erkannter Objekte nutzen, wobei die Parameter von Faster R-CNN während des Transformer-Trainings nicht angepasst werden. Faster R-CNN lernt jedoch lediglich Objekte wie Menschen oder Autos voneinander zu unterscheiden, wobei spezielle Eigenschaften der Objekte, wie zum Beispiel der Gemütszustand von Menschen, nicht erfasst werden. Bei der Lösung speziellerer Aufgaben, wie der Erkennung hasserfüllter Memes, können derartige Objekteigenschaften jedoch eine tragende Rolle spielen, weshalb das hier definierte Objekt-Modul sich von den bestehenden Verfahren abgrenzt, indem es eigene Repräsentationen der RoI erlernt. Um eine Erlernung aufgabenspezifischer visueller Merkmale zu ermöglichen, werden alle durch Faster R-CNN identifizierten RoI auf eine einheitliche Größe skaliert und anschließend in das in (Torch contributers, 2017) implementierte konvolutionale ResNet (He et al., 2015) eingegeben. Die ResNet-Parameter werden während des Transformer-Trainings angepasst und dafür genutzt, um eine Repräsentation der visuellen Merkmale in der Form eines Vektors aus $\mathbb{R}^{1000}$ zu errechnen.

Um dem Transformer zusätzliche Informationen über die RoI zu liefern, werden die visuellen Merkmale, in Anlehnung an viele der multimodalen Transformer in Anhang 1, durch zusätzliche Informationen aus einem Anreicherungsvektor ergänzt. Der Anreicherungsvektor $a(RoI_n)$ für die visuellen Merkmale von RoI_n errechnet sich anhand der in Gleichung 1 dargestellten Formel. Seine Einträge beinhalten die relativen Koordinaten der

RoI-Ecken, den relativen Bildanteil, welcher von der RoI eingenommen wird, und Informationen über das in RoI_n erkannte Objekt. Um den Vektor, welcher die visuellen Merkmale von RoI_n enthält, durch $a(RoI_n)$ anzureichern, werden die beiden Vektoren addiert. Dafür müssen sie zunächst aneinander angeglichen werden. Als erstes werden beide Vektoren mittels einer linearen Abbildung auf dieselbe Größe projiziert. Bereits jetzt wäre eine Addition der beiden Vektoren möglich, jedoch nicht zielführend, weil die beiden Vektoren aufgrund ihrer unterschiedlichen Berechnungsweisen mitunter stark voneinander abweichende Werte beinhalten können. Eine Addition derartig unterschiedlicher Vektoren kann, wie in Abschnitt 4.2 genauer beschrieben, zu einem wenig aussagekräftigem Ergebnisvektor führen. Zur Lösung dieses Problems werden die beiden Vektoren, analog zu Abschnitt 4.2, mittels Layer Normalization (Ba et al., 2016) normalisiert und erst im Anschluss daran mittels einer Vektoraddition kombiniert. Der aus dieser Berechnung hervorgehende Vektor repräsentiert ein erkanntes Objekt und kann als eine Positionen der Transformer-Eingabesequenz verwendet werden. Die Anzahl der Eingabesequenz-Positionen, welche durch das Objekt-Modul berechnet wird, ist durch den Hyperparameter $N_{Objekt} \in \mathbb{N}$ begrenzt. Werden mehr als N_{Objekt} Objekte in einem Bild identifiziert, so werden nur die Objekte mit den höchsten Vorkommenswahrscheinlichkeiten übernommen. Werden weniger Objekte erkannt, so werden die überzähligen Positionen mit dem Nullvektor belegt. Die Anwendung dieses Moduls auf ein Bild mit drei erkannten Objekten und $N_{Objekt} = 4$ wird in Abbildung 6 ersichtlich.

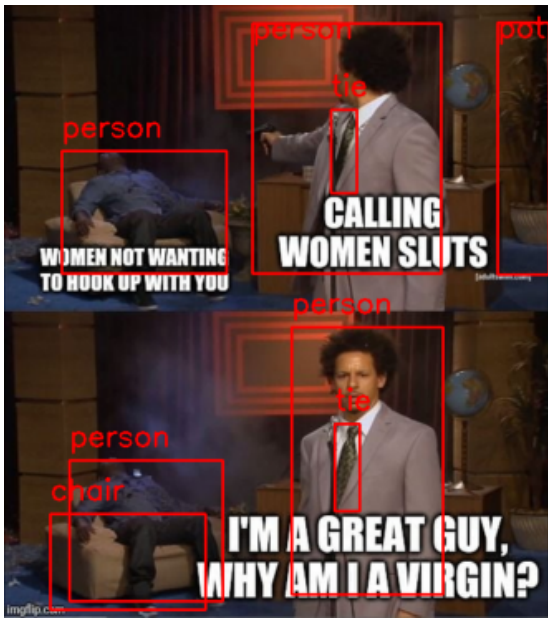

Abbildung 5: Anwendung der Objekterkennung auf einem Meme aus dem in (Fersini et al., 2022) vorgestellten Datensatz

$$a(RoI_n) = (\frac{x_{1,n}}{w_n}, \frac{y_{1,n}}{h_n}, \frac{x_{2,n}}{w_n}, \frac{y_{2,n}}{h_n}, \frac{(x_{2,n} - x_{1,n})(y_{2,n} - y_{1,n})}{w_n h_n}, i_n, p_{l_n}) \quad (1)$$

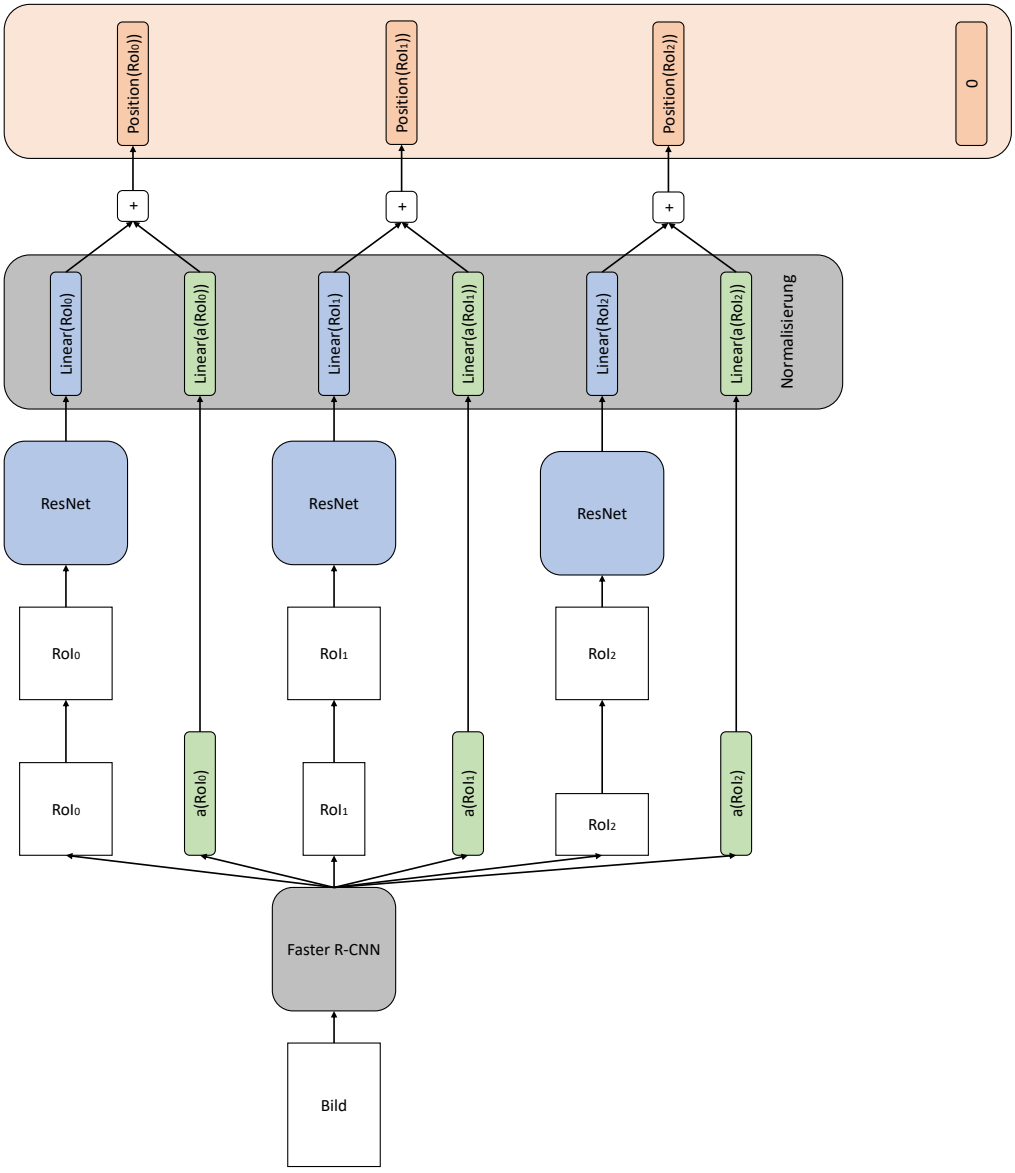

Abbildung 6: Die Anwendung des Objekt-Moduls

4.1.5 Das Kachel-Modul

Dieses Modul unterteilt das Eingabebild in gleichgroße Bildkacheln und repräsentiert deren visuelle Merkmale in der Form jeweils einer Position der Transformer-Eingabesequenz. Die Verwendung dieses Moduls verspricht eine Verarbeitung des Eingabebildes, unabhängig von einem vortrainierten Objekterkennungsverfahren, wodurch unerkannte Objekte und Bildhintergründe ebenfalls berücksichtigt werden.

Aufbauend auf der in (Dosovitskiy et al., 2020) definierten, durch (Gao et al. 2020) und (Lin et al., 2021) erweiterten und in (Hugging Face, 2021b) implementierten Methodik, werden alle Eingabebilder auf die gleiche Größe projiziert, gleichgroße disjunkte Bildkacheln aus ihnen gelöst und die visuellen Merkmale der Bildkacheln mittels eines konvolutionalen neuronalen Netzwerkes berechnet. Die visuellen Merkmale jeder Kachel werden als Vektor dargestellt und eine Positionskodierung, analog zum Token-Modul, auf sie addiert. Die Grundlage für die Positionskodierung stellt dabei die Zuweisung einer eindeutigen natürlichen Zahl für jede Bildkachel dar. Die so erzeugten Repräsentationen der Kacheln können als Positionen der Transformer-Eingabesequenz genutzt werden. Eine Anwendung des Kachel-Moduls wird in Abbildung 7 dargestellt. Zur Vereinfachung der Lesbarkeit wird das Eingabebild in der Abbildung in lediglich vier Kacheln unterteilt, wohingegen die Implementierung auf 196 Kacheln zurückgreift.

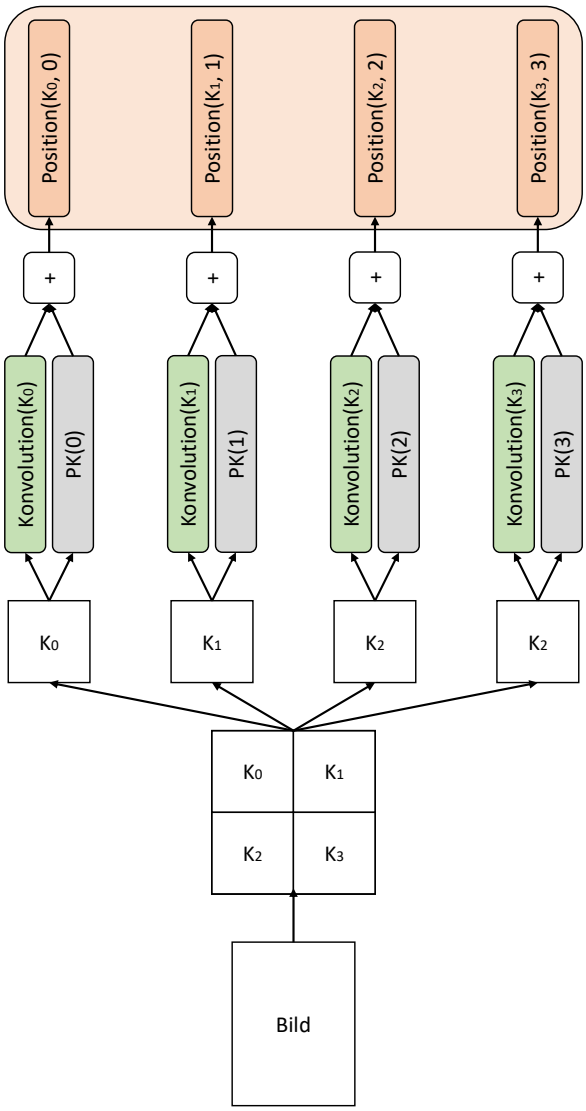

Abbildung 7: Die Anwendung des Kachel-Moduls

4.1.6 Das Bild-Modul

Das Bild-Modul wendet das in (Torch Contributers, 2017) implementierte ResNet auf das vollständige Eingabebild an und bringt das Ergebnis in eine Form, welche in die Transformer-Eingabesequenz integriert werden kann. Eine Anwendung fand ein ähnliches Verfahren in (Huang et al., 2020), wo es als visueller Kodierer bezeichnet wird. Ähnlich wie das Kachel-Modul verspricht auch das Bild-Modul eine Erfassung visueller Merkmale des gesamten Eingabebildes. Der Unterschied besteht darin, dass dieses Modul nicht auf eine Kachelunterteilung angewiesen ist, welche zusammengehörige Bildelemente in einzelne Abschnitte unterteilt. Dadurch ist dieses Modul dazu in der Lage, die wichtigsten Bildelemente zu ermitteln und diese mit dem Bildhintergrund in Beziehung zu setzen. Somit verarbeitet das Bild-Modul das gesamte Eingabebild in einem, wodurch potenziell eine größere Menge visueller Merkmale erkannt werden kann als bei der isolierten Betrachtung einzelner Bildabschnitte. Um die durch das Bild-Modul erkannten, potenziell zahlreichen visuellen Merkmale innerhalb der Transformer-Eingabesequenz zu vertreten, wird die Ausgabe des ResNets in mehrere Positionen der Transformer-Eingabesequenz überführt. Dafür wird der Ausgabevektor des ResNets in zwei gleichgroße Vektoren unterteilt und jeder Vektor mittels einer linearen Abbildung zu einer Position der Transformer-Eingabesequenz verformt. Die Anwendung dieses Moduls auf ein Eingabebild wird in Abbildung 8 dargestellt, wobei die Zahlen 0 und 1 die beiden gleichgroßen Abschnitte des ResNet-Ausgabevektors darstellen.

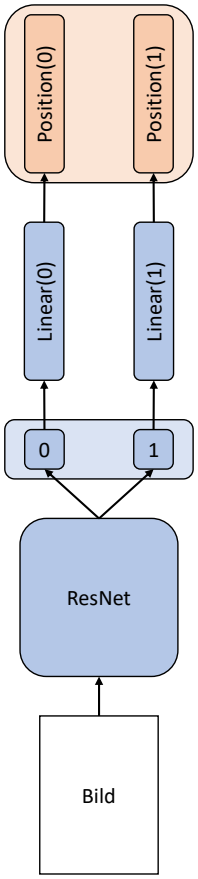

Abbildung 8: Die Anwendung des Bild-Moduls

4.2 Die Fusion der Eingabesequenz-Positionen

Greift der Transformer auf mehr als ein Modul zur Bildung von Positionen seiner Eingabesequenz zurück, so müssen die Ausgaben der Module zu einer gemeinsamen Eingabesequenz fusioniert werden. Die einzelnen Module extrahieren Merkmale unterschiedlicher Komplexität, verfügen dabei über größtenteils disjunkte Parameter und führen mitunter drastisch unterschiedliche Berechnungen durch. Dadurch kann es in erster Linie dazu kommen, dass die Module zu unterschiedlichen Zeitpunkten beginnen, nichtgeneralisierende Merkmale zu erlernen und somit die Trainingsdaten zu overfitten. Um die Overfitting-Zeitpunkte der einzelnen Module aneinander anzugleichen, wird jeweils eine individuelle Dropout-Funktion (Hinton et al., 2012) auf die Ausgabe jedes einzelnen Moduls angewendet. Dropout-Funktionen ersetzen die Einträge ihrer Eingabe während des Netzwerktrainings mit einer zuvor bestimmten Wahrscheinlichkeit durch 0, wodurch das Netzwerk kaum noch dazu in der Lage ist, seine Eingaben auswendig zu lernen und somit in letzter Konsequenz generalisierende Merkmale der Eingabedaten erlernen muss (Hinton et al., 2012). Durch die Wahl einer individuellen Dropout-Wahrscheinlichkeit für jedes der sechs Module zur Bildung von Positionen der Transformer-Eingabesequenz können ihre Overfitting-Zeitpunkte aneinander angeglichen werden.

Die oben beschriebenen Modulunterschiede können zudem dazu führen, dass die von ihnen berechneten Positionen der Transformer-Eingabesequenz stark voneinander abweichende Werte beinhalten. Nehmen beispielsweise die ausgegebenen Positionen des Token-Moduls deutlich größere Werte an als die des Objekt-Moduls, so können die Reize aus den Positionen des Objekt-Moduls gedämpft und deren Wichtigkeit vom Netzwerk falsch eingeschätzt werden. Um diese potentielle Problemquelle zu umgehen, werden zwei unterschiedliche Verfahren angewendet: Zunächst wird eine Modulkodierung auf die Positionen jeder Modulausgabe addiert. Dafür werden die Module zur Berechnung der Positionen der Eingabesequenz durchnummeriert und ihre Nummerierung mittels eines eindimensionalen Embeddings auf einen Vektor der Größe der Transformer-Eingabesequenzpositionen abgebildet. Daraufhin

wird der Vektor auf alle Positionen der Ausgabe des jeweiligen Moduls addiert. Formen einer solchen Kodierung finden sich auch in anderen Transformern (Li et al., 2019; Su et al., 2020; Liu et al., 2019; Huang et. al, 2020). Um eine weiterführende Angleichung der Modulausgaben zu erreichen, werden die Positionen ihrer Ausgaben mittels Layer Normalization (Ba et al., 2016) normalisiert. Ein ähnliches Verfahren findet sich unter anderem auch in dem in (Li et al., 2020a) vorgestellten Transformer. Nach der Anwendung dieser Verfahren werden die Ausgaben der verwendeten Module entlang ihrer Positionen zu einer gemeinsamen Transformer-Eingabesequenz konkateniert.

4.3 Die Kontextualisierung der Transformer-Eingabesequenz

Nach der Bildung der vollständigen Transformer-Eingabesequenz werden ihre Positionen miteinander in Beziehung gesetzt. Dafür werden die in (Vaswani et al., 2017) vorgestellten Transformer-Kodierer verwendet, dessen Implementierung für die in dieser Arbeit durchgeführten Experimente aus (Hyugen AI, 2021) übernommen wurde. Vorangegangene Forschung zeigt, dass derartige Kodierer in der Lage sind, selbst filigrane Verwebungen zwischen den Eingabepositionen zu modellieren (Devlin et al., 2019; Dosovitskiy et al., 2020). Dies gilt insbesondere auch für multimodale Beziehungen, deren Erkennung unter anderem durch die Visualisierungen in (Huang et al., 2020; Wang et al., 2020a; Li et al., 2019) belegt wird. Die beiden Hauptverarbeitungsschritte innerhalb eines Transformer-Kodierers sind die Beziehungsermittlung über den in (Vaswani et al., 2017) vorgestellten Attention-Mechanismus und die darauffolgende Anreicherung des Ergebnisses mittels mehrerer linearen Abbildungen. Dieses Bauteil erlaubt, bestimmte Positionen der Transformer-Eingabesequenz vor den anderen Positionen mittels eines Maskierungsverfahrens zu verbergen, wodurch diese nicht mit den anderen Positionen in Beziehung gesetzt werden. In dem hier verwendeten Netzwerk wurde dieses Verfahren eingesetzt, um die mit Padding-Tokens besetzten Einträge des Token- und des Assoziations-Moduls, sowie die mit dem Nullvektor belegten Positionen des Objekt-Moduls zu maskieren. Wird eine Transformer-Eingabesequenz in einen Transformer-Kodierer eingegeben, so besitzt dessen Ausgabe die

gleiche Dimensionalität wie die Transformer-Eingabesequenz. Diese Eigenschaft erlaubt es, dass die Ausgabe eines Transformer-Kodierers entweder als die Transformer-Ausgabesequenz oder erneut in einen Transformer-Kodierer eingegeben werden kann, wodurch eine Reihenschaltung der Transformer-Kodierer ermöglicht wird und komplexere Beziehungen zwischen den Positionen der Transformer-Eingabesequenz erlernt werden können. Im Rahmen dieser Arbeit wird die Transformer-Ausgabesequenz durch die Anwendung zweier in Reihe geschalteter Transformer-Kodierer auf die Transformer-Eingabesequenz gebildet.

4.4 Die Bildung der Netzwerkvorhersage

Nach der Bildung der Transformer-Ausgabesequenz durch die Anwendung von Transformer-Kodierern kann eine Netzwerkvorhersage gebildet werden. Um die Transformer-Ausgabesequenz zur Klassifikation eines Memes nutzen zu können, müssen die darin enthaltenen Informationen auf die Wahrscheinlichkeit abgebildet werden, mit welcher das eingegebene Meme als hasserfüllt erkannt wird. Dafür wird die Transformer-Ausgabesequenz auf einen einzigen Vektor gestaucht, durch Anwendung einer linearen Abbildung auf einen Wert reduziert und mittels einer Sigmoidalfunktion auf eine Wahrscheinlichkeit $p \in [0, 1]$ abgebildet. Liegt die ausgegebene Wahrscheinlichkeit bei mehr als 50 %, so besagt die Netzwerkvorhersage, dass es sich bei dem eingegebenen Meme um ein hasserfülltes Meme handelt.

(Ma et al., 2019) haben unterschiedliche Verfahren zur Stauchung der Transformer-Ausgabesequenz verglichen und festgestellt, dass das sogenannte Mittelwert-Pooling über alle getesteten Aufgabenstellungen hinweg zu den besten Ergebnissen führt. Für die Durchführung des Mittelwert-Poolings müssen alle Positionen der Sequenz eintragsweise aufaddiert und das Ergebnis durch die Positionsanzahl N geteilt werden. Der Empfehlung von (Ma et al., 2019) folgend wird auch in dieser Arbeit auf jenes Verfahren zurückgegriffen. Das vollständige Netzwerk wird in Abbildung 9 dargestellt und weiterführende Informationen über das Netzwerk und dessen Training können Anhang 2 entnommen werden.

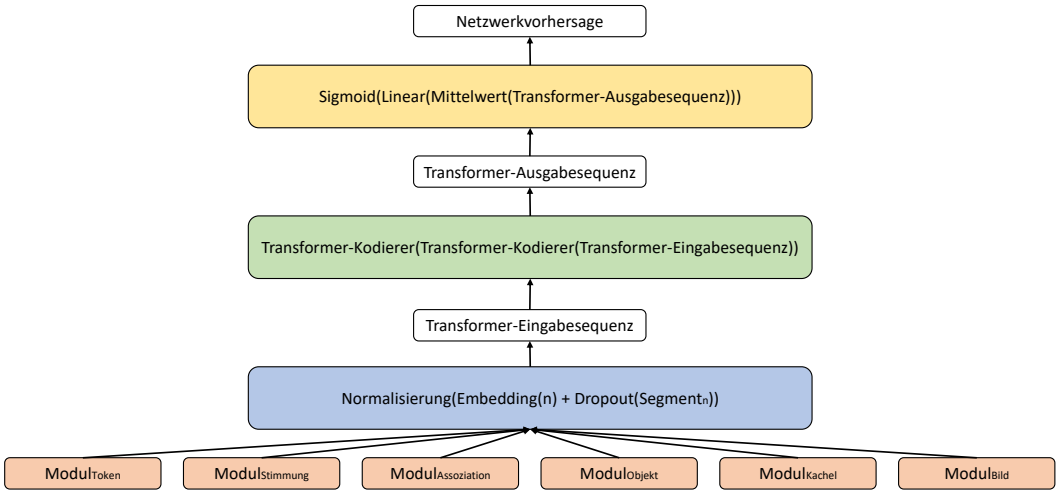

Abbildung 9: Das vollständige Transformer-Netzwerk zur Beantwortung der Forschungsfragen

5 Versuchsaufbau

Der Versuchsaufbau gliedert sich in mehrere Phasen, welche in Abbildung 10 dargestellt und in den Abschnitten dieses Kapitels im Detail beschrieben werden. In der ersten Versuchsphase wird überprüft, welche Module zur Bildung der Transformer-Eingabesequenz zu den besten Netzwerkvorhersagen auf den Validierungsdatensätzen führen. In der zweiten Versuchsphase wird ein Transformer, welcher auf die zuvor identifizierten besten Module zurückgreift, mittels mehrerer datensatzübergreifender Trainingsverfahren trainiert und mit dem herkömmlichen Training verglichen. Nach Abschluss dieser beiden Phasen stehen sowohl die besten Module zur Bildung der Transformer-Eingabesequenz als auch das beste Trainingsverfahren fest. Anschließend werden sowohl eine Early-fusion-Variante als auch eine Late-fusion-Variante des Transformers entsprechend der zuvor gewonnenen Erkenntnisse trainiert und auf den Testdaten evaluiert. Dieser Vorgang wird auf den beiden in dieser Arbeit betrachteten Meme-Datensätzen durchgeführt. Bevor die einzelnen Phasen näher beleuchtet werden, werden zunächst die beiden verwendeten Datensätze und die verwendeten Evaluierungsmetriken beschrieben.

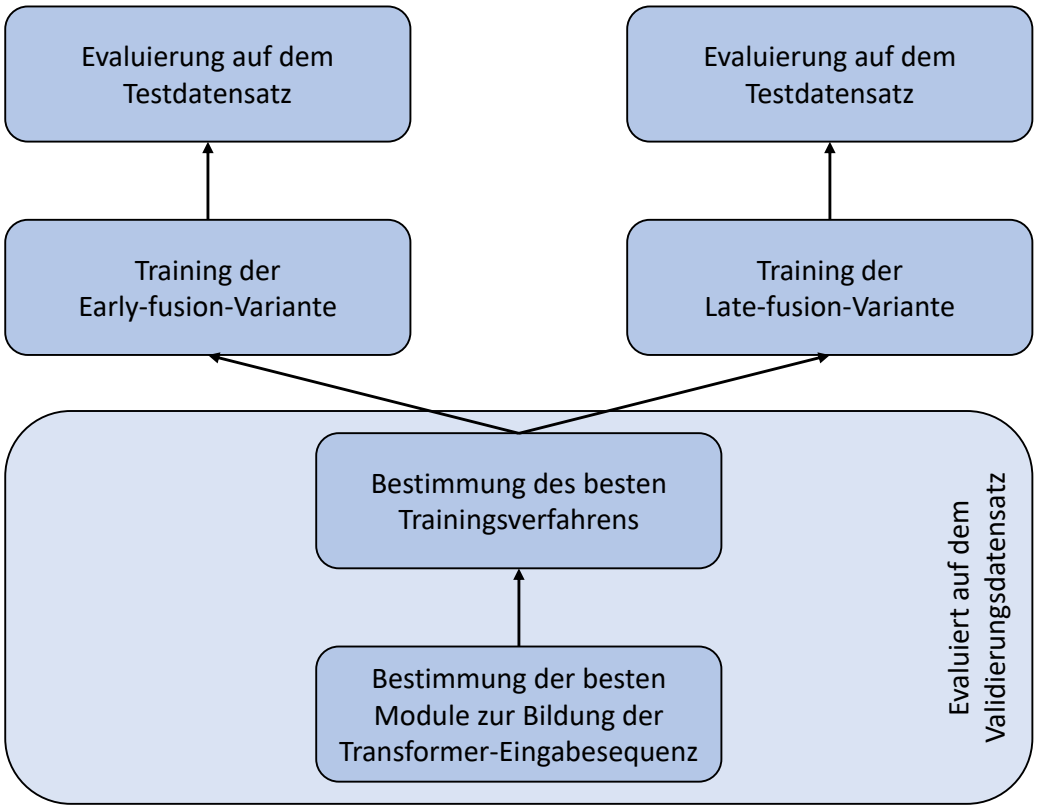

Abbildung 10: Die Versuchsaufbauphasen

5.1 Verwendete Datensätze

Zur Klärung der Forschungsfragen wurden zwei verschiedene Datensätze verwendet. Bei dem Ersten handelt es sich um den in (Kiela et al., 2020a) beschriebenen und in der Hateful Memes (HM) Challenge verwendeten Datensatz, welcher in den folgenden Kapiteln als HM-Datensatz bezeichnet wird. Er beinhaltet 9.000 Memes, welche entweder als hasserfüllt oder als harmlos gekennzeichnet sind. Eine Besonderheit dieses Datensatzes ist, dass er eine Reihe von Semidubletten enthält, welche den gleichen Inhalt in einer ihrer beiden Modalitäten teilen (Kiela et al., 2020a). Im Rahmen dieser Arbeit werden die Semidubletten in Bilddubletten, Memes, welche ihre Bild-Komponente mit anderen Memes teilen, und Textdubletten, Memes, welche ihre Text-Komponente mit anderen Memes teilen, kategorisiert. Abbildung

11 enthält sowohl ein Bilddublettenpaar als auch ein Textdublettenpaar aus dem HM-Datensatz.

Der zweite Datensatz wird stückweise im Rahmen der aktuell abgehaltenen Multimedia Automatic Misogyny Identification (MAMI) Challenge (Fersini et al., 2022) veröffentlicht und enthält zum aktuellen Zeitpunkt 10.000 annotierte Memes, welche als entweder *frauenfeindlich* oder *nicht frauenfeindlich* gekennzeichnet sind. Da frauenfeindliche Äußerungen als eine Unterkategorie der Hassrede verstanden werden können, werden die als frauenfeindlich identifizierten Memes im Rahmen dieser Arbeit als hasserfüllt und die verbleibenden Memes als harmlos gekennzeichnet. Als frauenfeindlich gekennzeichneten Memes enthalten eine zusätzliche Beschreibung der Frauenfeindlichkeitsform (Fersini et al., 2022), welche jedoch im Rahmen dieser Arbeit keine Verwendung findet. In den folgenden Kapiteln wird dieser Datensatz als MAMI-Datensatz bezeichnet.

Neben ihren oben genannten Besonderheiten unterscheiden sich die Datensätze in ihrem Veröffentlichungsdatum, woraus sich eine gewisse semantische Diskrepanz ergibt. So beinhaltet der HM-Datensatz beispielsweise keinerlei Memes über die aktuell grassierende Corona-Pandemie, wohingegen diese im MAMI-Datensatz thematisiert wird.

Tabelle 1 verdeutlicht die Ergebnisse eines quantitativen Vergleichs beider Datensätze, wobei fett gedruckte Zahlen erreichte Mittelwerte, eingeklammerte Zahlen Standardabweichungen darstellen und sämtliche Ergebnisse auf die nächste natürliche Zahl gerundet sind. Es zeigt sich, dass die Memes des MAMI-Datensatzes im Durchschnitt wesentlich längere Texte beinhalten. Dass der Anteil an Memes mit erkannten Assoziationen im MAMI-Datensatz größer ist als im HM-Datensatz kann eine natürliche Folge dieser Eigenschaft sein.

Beide Datensätze werden in drei disjunkte Untermengen, einen Trainingsdatensatz, einen Validierungsdatensatz und einen Testdatensatz, aufgeteilt. Der Trainingsdatensatz enthält 80 % der Memes pro Datensatz und die restlichen Memes sind jeweils in gleichen Teilen auf den Validierungsdatensatz und den Testdatensatz aufgeteilt. Pro Datensatz enthalten der Validierungsdatensatz und der Testdatensatz jeweils 10% der Ausgangsmenge.

Tabelle 1: Ein quantitativer Vergleich der beiden Meme-Datensätze

Eigenschaft	**HM**	**MAMI**
Anzahl Memes	9.000	10.000
Hasserfüllte Memes	3.300	5.000
Zeichenanzahl pro Meme-Text	**62** (37)	**101** (83)
Bildbreite pro Meme	**599** (179)	**569** (100)
Bildhöhe pro Meme	**524** (166)	**583** (86)
Erkannte Objekte pro Meme	**2** (2)	**2** (2)
Memes mit erkannten Objekten	7856	7777
Erkannte Assoziationen pro Meme	**2** (4)	**2** (4)
Memes mit erkannten Assoziationen	3968	5419

Abbildung 11: Zwei Semidubletten aus dem HM-Datensatz

5.2 Evaluierungsmetriken

Um die Qualität der Netzwerkvorhersagen bestimmen zu können, muss eine einheitliche Metrik verwendet werden. In dieser Arbeit wird dafür auf den Area under a Receiver Operating Characteristic Curve (AUC-ROC) Score zurückgegriffen. Er berechnet die Wahrscheinlichkeit, mit welcher ein Klassifizierer ein zufällig ausgewähltes Paar von Datenpunkten unterschied-

licher Klasse korrekt klassifiziert (Hanley und McNeil, 1982). Somit ist er unabhängig von der Klassenverteilung im Evaluierungsdatensatz, was insbesondere für die Bestimmung der Modellgüte auf dem ungleich verteilten HM-Datensatz von Vorteil ist. Enthalten beispielsweise 95% der Datenpunkte im Evaluierungsdatensatz hasserfüllte Memes und liegt ein triviales Netzwerk vor, welches für jede Eingabe mit einer Überzeugung von 100% voraussagt, dass ein eingegebenes Meme hasserfüllt ist, so haben die Netzwerkvorhersagen eine Genauigkeit von 95%. Unter denselben Umständen erhalten die Netzwerkvorhersagen einen deutlich geringeren AUC-ROC Score von 50%, welcher die Trivialität des Netzwerks enttarnt.

Zwei weitere Metriken, welche weniger ausschlaggebend für die Beurteilung der Güte der Netzwerkvorhersagen sind, aber einen tieferen Einblick in das Netzwerkverhalten liefern, sind der Precision Score und der Recall Score. Für ein Netzwerk zur Klassifikation von Memes in die beiden Klassen hasserfüllt und harmlos gilt:

Der Precision Score gibt an, wie hoch die Wahrscheinlichkeit ist, dass ein als hasserfüllt vorhergesagtes Meme tatsächlich hasserfüllt ist. Der Recall Score hingegen gibt an, wie hoch die Wahrscheinlichkeit ist, dass ein hasserfülltes Meme die Netzwerkvorhersage hasserfüllt erhält. (Powers, 2011)

Um die Robustheit der Evaluationsmetriken zu erhöhen werden alle angewendeten Vortrainings-, und Trainings-Methoden mit drei unterschiedlichen Netzwerkinstanzen durchgeführt, evaluiert und die Ergebnisse epochenweise über alle Netzwerkinstanzen hinweg gemittelt. Bei den Netzwerkinstanzen handelt es sich jeweils um dasselbe Transformer-Netzwerk, welches jedoch mittels drei unterschiedlicher pseudozufälliger Sampling-Methoden trainiert wird. Durch die manuelle Unterbindung nicht-deterministischen Verhaltens, sowohl auf Software- als auch auf Hardware-Ebene, und die Dokumentation sämtlicher technischer Daten der verwendeten Versuchsumgebung liegt eine vollständige Reproduzierbarkeit der Versuchsergebnisse vor.

5.3 Evaluierung der Bildung der Transformer-Eingabesequenz

Zur Identifikation der besten Module zur Bildung der Transformer-Eingabesequenz werden die Algorithmen Sequential Forward Selection (SFS) und Sequential Backward Elimination (SBE) genutzt und an die Anforderungen innerhalb dieser Arbeit angepasst.

Beide Verfahren haben zum Ziel, diejenige Eingaberepräsentation zu identifizieren, durch dessen Nutzung die besten Netzwerkvorhersagen erzielt werden können. Im Rahmen der vorliegenden Arbeit werden entsprechend die besten Module zur Bildung der Positionen der Transformer-Eingabesequenz mittels SFS und SBE identifiziert. SFS beginnt mit einer leeren Modulmenge und erweitert diese iterativ. In jeder Iteration wird der Modulmenge jeweils ein Modul zur Bildung der Transformer-Eingabesequenz angehängt und das Netzwerk auf der Transformer-Eingabesequenz, welche mittels aller Module aus der aktuellen Modulmenge gebildet wird, trainiert und evaluiert, bevor das Modul wieder aus der Modulmenge herausgenommen wird. Die Modulhinzufügung, welche zu der größten Verbesserung der Netzwerkvorhersagen führt, wird anschließend für die folgenden Iterationen übernommen und der Algorithmus konvergiert, sobald keine Verbesserung mehr durch das Hinzufügen eines weiteren Moduls erzielt werden kann. SBE verfolgt einen ähnlichen Ansatz, beginnt jedoch mit der vollständigen Modulmenge und entfernt jeweils ein Modul pro Iteration. (Mao, 2004)

5.4 Datensatzübergreifende Trainingsverfahren

Im Anschluss an die Ermittlung der optimalen Menge an Modulen zur Bildung der Transformer-Eingabesequenz wird überprüft, ob eine gegenseitige Nutzbarmachung der beiden Meme-Datensätze mittels datensatzübergreifenden Trainingsverfahren erzielt werden kann. Die sechs Trainingsverfahren bestehen jeweils aus einer Vortrainings- und einer Trainings-Phase und werden als HM-MAMI, MAMI-HM, CROSS-MAMI, CROSS-HM, SEP-MAMI und SEP-HM bezeichnet und epochenweise evaluiert. In der Vortrainigsphase von HM-MAMI wird das Netzwerk zunächst auf den HM-Trainingsdaten dazu trainiert hasserfüllte Memes zu erkennen. Daraufhin wird evaluiert, inwiefern

es die hasserfüllten Memes des MAMI-Validierungsdatensatzes korrekt klassifiziert. Im Anschluss daran beginnt die Trainingsphase, in welcher das Netzwerk auf den MAMI-Trainingsdaten darauf spezialisiert wird die hasserfüllten Memes des MAMI-Datensatzes zu erkennen. Die Trainingsphase wird, analog zu der Vortrainingsphase, auf dem MAMI-Validierungsdatensatz evaluiert. Die Intention hinter diesem Verfahren ist, dass das Netzwerk während der Vortrainingsphase sowohl harmlose als auch hasserfüllte Memes eines fremden Datensatzes erlernt und somit ein grobes Verständnis der Semantik der beiden Klassen erhält. CROSS-MAMI basiert auf demselben Prinzip, jedoch wird während dessen Vortrainingsphase auf der Vereinigung des HM-Trainingsdatensatzes und des MAMI-Trainingsdatensatzes trainiert hasserfüllte Memes zu erkennen. SEP-MAMI verfolgt einen etwas anderen Ansatz als HM-MAMI und CROSS-MAMI. Zunächst erhalten alle verwendeten Memes eine zusätzliche Annotation, welche preisgibt, zu welchem der beiden Datensätze sie gehören. In der Vortrainingsphase von SEP-MAMI erhält das Netzwerk die Vereinigung der beiden Trainingsdatensätze und wird dazu trainiert, die Memes den richtigen Datensätzen zuzuweisen. Evaluiert wird das Netzwerk in der Vortrainigsphase, indem überprüft wird, wie gut es die Memes der Vereinigung der beiden Validierungsdatensätze ihren richtigen Datensätzen zuweisen kann. Diese Vortrainingsmethode soll dazu dienen, dass das Netzwerk ein grundlegendes Verständnis der Meme-Strukturen erlernt. Anschließend daran wird das vortrainierte Netzwerk dazu trainiert die hasserfüllten Memes des MAMI-Trainingsdatensatzes zu erkennen. Die Güte des Netzwerks wird evaluiert, indem überprüft wird, wie gut das Netzwerk die hasserfüllten Memes des MAMI-Validierungsdatensatzes erkennt. MAMI-HM, CROSS-HM und SEP-HM werden analog zu den oben beschriebenen Verfahren durchgeführt, wobei jedoch die Rolle der MAMI-Daten und die der HM-Daten vertauscht werden.

5.5 Evaluierung auf den Testdatensätzen

Nach der Bestimmung der optimalen Module zur Bildung der Transformer-Eingabesequenzen und der besten Trainingsverfahren werden die gebildeten Netzwerke und ihre Late-fusion-Varianten auf den Testdatensätzen evaluiert und ihre Fehler analysiert. Neben dem AUC-ROC Score werden dafür auch der Precision Score und der Recall Score berechnet sowie weitere später beschriebene Untersuchungen der Netzwerkvorhersagen auf den Testdaten durchgeführt.

In Anlehnung an die Darstellung in (Kiela et al., 2020b) wird die Late-fusion-Variante eines, wie in dieser Arbeit definierten, Transformers dabei wie folgt gebildet:

Greift ein Transformer auf die beiden Module A und B zur Bildung seiner Eingabesequenz zurück und werden seine Parameter mittels des Trainingsverfahrens C trainiert, so besteht seine Late-fusion-Variante aus den beiden Transformern, welche die Module A und B in isolierter Form nutzen und mittels C trainiert werden. Wird die Late-fusion-Variante dieses beispielhaften Transformers auf den Testdaten evaluiert, so werden die Vorhersagen all ihrer Teiltransformer gemittelt.

6 Versuchsergebnisse

In diesem Kapitel werden die Ergebnisse der Versuche zusammengefasst.

6.1 Die optimalen Transformer-Eingabesequenzen

Die Ergebnisse der Anwendung von SFS und SBE zur Ermittlung der optimalen Module zur Bildung der Transformer-Eingabesequenzen werden in Tabelle 2 und Tabelle 3 festgehalten. Dabei wird jedes Modul mit seinem Anfangsbuchstaben abgekürzt, die Modulmengen durch Mengenklammern dargestellt und unter Anwendung der jeweiligen Module erzielten AUC-ROC Scores dahinter dargestellt. Versuche, welche eine Verschlechterung im Vergleich zur vorherigen Iteration erzielen, sind in der Abbildung ausgegraut. Die Anwendung beider Algorithmen führt auf dem HM-Datensatz zu der Bildung der gleichen Modulmenge, bestehend aus dem Token-Modul, dem Assoziatins-Modul, dem Stimmungs-Modul und dem Kachel-Modul. Analog dazu konvergieren beide Algorithmen auf dem MAMI-Datensatz in der Kombination des Token-, des Objekt- und des Assoziations-Moduls. Im Detail zeigt sich:

Das Token-Modul wird auf beiden Datensätzen in der ersten SFS-Iteration hinzugefügt und erzielt somit die besten Ergebnisse unter allen isolierten Modulen. Darüber hinaus führt die Eliminierung dieses Moduls in allen SBE-Iterationen zu einer Verschlechterung des AUC-ROC Scores.

Das Assoziations-Modul kommt ebenfalls in beiden resultierenden Ergebnismengen vor. Auf dem HM-Datensatz führt das Hinzufügen dieses Moduls zu dem zweitbesten Ergebnis in der ersten SFS-Iteration und wird dort bereits in der zweiten SFS-Iteration hinzugefügt, wohingegen es auf dem MAMI-Datensatz erst in der dritten SFS-Iteration von der Ergebnismenge übernommen wird. Auf den HM-Daten führt die Entfernung des Assoziations-Moduls zu einer Verschlechterung in allen SBE-Iterationen.

Das Kachel-Modul ist eines der final ausgewählten Module auf dem HM-Datensatz und es mittels SBE zu entfernen führt dort jeweils zu einer Verschlechterung des Ergebnisses. Auf dem MAMI-Datensatz hingegen ist

dieses Modul das letzte, welches mittels SBE entfernt wird. Darüber hinaus führte dessen Hinzufügung auf den MAMI-Daten in allen SFS-Iterationen, mit Ausnahme der Letzten stets zu einer Modell-Verbesserung.

Das Stimmungs-Modul ist das letzte Modul, welches in die finale Modulmenge auf den HM-Daten aufgenommen wird. Es bleibt jedoch festzuhalten, dass es mit Abstand das schlechteste Ergebnis in der ersten SFS-Iteration auf dem MAMI-Datensatz darstellt.

Das Objekt-Modul ist das zweite Modul, welches mittels SFS in die Modulmenge auf dem MAMI-Datensatz hinzugefügt wird und es auf dem MAMI-Datensatz mittels SBE zu entfernen führt stets zu einer Verschlechterung des AUC-ROC Scores. Auf den HM-Daten hingegen wird dieses Modul nicht in die finale Modulmenge aufgenommen.

Tabelle 2: Ergebnisse der Sequential Forward Selection

Iteration	**HM**	**MAMI**
1	{O}: 56,07 % {K}: 56,33 % {S}: 57,06 % {B}: 59,58 % {A}: 65,88 % {T}: 66,17 % ⇒ füge T hinzu	{S}: 48,33 % {K}: 68,44 % {O}: 68,77 % {A}: 71,03 % {B}: 71,34 % {T}: 84,01 % ⇒ füge T hinzu
2	{T, O}: 67,08 % {T, S}: 67,35 % {T, B}: 67,39 % {T, K}: 68,27 % {T, A}: 68,95 % ⇒ füge A hinzu	{T, B}: 82,81 % {T, A}: 83,29 % {T, S}: 83,56 % {T, K}: 84,27 % {T, O}: 84,4 % ⇒ füge O hinzu
3	{T, A, B}: 67,76 % {T, A, O}: 69,05 % {T, A, S}: 69,59 % {T, A, K}: 69,79 % ⇒ füge K hinzu	{T, O, B}: 82,54 % {T, O, K}: 85,18 % {T, O, S}: 85,22 % {T, O, A}: 85,56 % ⇒ füge A hinzu
4	{T, A, K, O}: 70,23 % {T, A, K, B}: 71,05 % {T, A, K, S}: 71,36 % ⇒ füge S hinzu	{T, O, A, B}: 84,42 % {T, O, A, S}: 84,64 % {T, O, A, K}: 85,09 % ⇒ Konvergenz
5	{T, A, K, S, B}: 69,62 % {T, A, K, S, O}: 70,66 % ⇒ Konvergenz	

Tabelle 3: Ergebnisse der Sequential Backward Elimination

Iteration	HM	MAMI
1	{T, O, S, A, K, B}: 69,67 %	{T, O, S, A, K, B}: 82,99 %
2	{O, S, A, K, B}: 65,77 % {T, O, S, K, B}: 68,84 % {T, O, S, A, B}: 68,96 % {T, S, A, K, B}: 69,62 % {T, O, A, K, B}: 70,3 % {T, O, S, A, K}: 70,66 % ⇒ entferne B	{O, S, A, K, B}: 79,02 % {T, S, A, K, B}: 82,32 % {T, O, S, A, B}: 83,01 % {T, O, A, K, B}: 83,65 % {T, O, S, K, B}: 84,62 % {T, O, S, A, K}: 85,13 % ⇒ entferne B
3	{O, S, A, K}: 66,49 % {T, O, S, K}: 69,24 % {T, O, S, A}: 70,19 % {T, O, A, K}: 70,23 % {T, S, A, K}: 71,36 % ⇒ entferne O	{O, S, A, K}: 79,71 % {T, S, A, K}: 84,96 % {T, O, S, A}: 84,97 % {T, O, S, K}: 85,2 % {T, O, A, K}: 85,37 % ⇒ entferne S
4	{S, A, K}: 67,1 % {T, S, K}: 69,29 % {T, S, A}: 69,59 % {T, A, K}: 69,79 % ⇒ Konvergenz	{O, A, K}: 78,84 % {T, A, K}: 84,92 % {T, O, K}: 85,18 % {T, O, A}: 85,56 % ⇒ entferne K
5		{O, A}: 77,06 % {T, A}: 83,29 % {T, O}: 84,4 % ⇒ Konvergenz

6.2 Ergebnisse des datensatzübergreifenden Trainings

Die Ergebnisse des datensatzübergreifenden Trainings werden in Tabelle 4 dargestellt. Sowohl die Vortrainingsphase als auch die Trainingsphase wurden mittels des AUC-ROC Scores auf den jeweiligen Validierungsdaten evaluiert und die besten Ergebnisse pro Datensatz wurden hervorgehoben. Es zeigt sich, dass keine signifikanten Verbesserungen der Netzwerkvorhersagen durch das Hinzufügen der Vortrainingsphasen erzielt werden können. Einzig die Anwendung von CROSS-MAMI mündet in einer geringen Verbesserung des erzielten AUC-ROC Scores auf den Validierungsdaten.

Tabelle 4: Evaluierung aller Trainingsprozesse auf den Validierungsdaten

Verfahren	Vortraining	Training
HM	/	**71,36 %**
MAMI-HM	54,53 %	70,38 %
CROSS-HM	66,71 %	67,13 %
SEP-HM	98,98 %	68,11 %
MAMI	/	85,56 %
HM-MAMI	63,46 %	84,52 %
CROSS-MAMI	84,23 %	**85,6 %**
SEP-MAMI	94,96 %	80,62 %

6.3 Ergebnisse auf den Testdatensätzen

Nachdem die optimalen Module zur Bildung der Transformer-Eingabesequenz und das beste der getesteten Trainingsverfahren für jeden der beiden Datensätze bestimmt wurde, wurden die optimalen Netzwerkkonfigurationen sowie ihre Late-fusion-Varianten auf den Testdatensätzen evaluiert. Die Ergebnisse werden in Tabelle 5 festgehalten und zeigen, dass alle vier Netzwerke ihr erlerntes Wissen auf bisher ungesehene Daten übertragen können, wobei die besten Ergebnisse pro Datensatz hervorgehoben wurden.

Während die Fragmentierung des MAMI-Netzwerks zur Bildung seiner Late-fusion-Variante MAMI_{late} eine Verbesserung aller Scores nach sich zieht, führt selbiges zu einer einem verringerten AUC-ROC Score und einer erheblichen Verschlechterung des Recall Scores zugunsten einer Precision-Verbesserung auf den HM-Daten.

Die in dieser Arbeit erreichten AUC-ROC Scores können nicht direkt mit den in der HM-Challenge erzielten Ergebnissen verglichen werden, weil der Testdatensatz der Challenge nicht veröffentlicht wurde, jedoch scheint das hier verwendete HM-Netzwerk schlechtere Ergebnisse zu erzielen als die intensiv vortrainierten Netzwerke der Challengesieger (Kiela et al., 2020a; Fitzpatrick, 2021). Analog dazu können die Ergebnisse auf den MAMI-Testdaten nicht mit denen der Challengeteilnehmer der MAMI-Challenge verglichen werden, weil die finale Netzwerkkonfiguration zum Zeitpunkt der MAMI-Evaluation noch nicht ermittelt wurde.

Tabelle 5: Evaluierung der Netzwerke mittels verschiedener Scores auf den Testdatensätzen

Verfahren	AUC-ROC	Recall	Precision
HM	**71,46 %**	**61,34 %**	53,34 %
HM_{late}	70,69 %	31,31 %	**67,12 %**
MAMI	84,21 %	82,17 %	72,61 %
$MAMI_{late}$	**84,6 %**	**85,29 %**	**73,21 %**

7 Diskussion

In diesem Kapitel werden die Ergebnisse der durchgeführten Experimente interpretiert und miteinander verknüpft, bevor die Netzwerkvorhersagefehler auf Regelmäßigkeiten untersucht werden, um weitere Erkenntnisse über die inneren Zustände der Netzwerke und die Eigenheiten schwer zu klassifizierender Memes zu gewinnen. Im letzten Abschnitt dieses Kapitels werden Einschränkungen dieser Arbeit erläutert und eine Reihe zukünftiger Forschungsrichtungen abgeleitet.

7.1 Interpretation der Versuchsergebnisse

In den folgenden Abschnitten werden die Versuchsergebnisse interpretiert und daraus gewonnene Erkenntnisse festgehalten.

7.1.1 Interpretation der Güte der Module zur Bildung der Transformer-Eingabesequenz

Zunächst zeigt sich, dass auf beiden Datensätzen unterschiedliche Mengen an Modulen zur Bildung der Transformer-Eingabesequenzen ermittelt wurden. Somit wird die Klassifikation der Memes der beiden Datensätze durch unterschiedliche Verarbeitungsverfahren optimiert, woraus abgeleitet werden kann, dass ein signifikanter Unterschied zwischen den beiden Datensätzen besteht. Des Weiteren bleibt festzuhalten, dass sowohl SFS als auch SBE pro Datensatz zu derselben finalen Modulmenge konvergieren, was die Robustheit der Ergebnisse unterstreicht. Im Folgenden werden die finalen Mengen an Modulen zur Bildung der Transformer-Eingabesequenzen interpretiert und

Vermutungen angestellt, worauf die Unterschiede zwischen den beiden Mengen zurückzuführen sind:

Das Token-Modul bringt die stärkste Verbesserung des AUC-ROC Scores mit sich, woraus abgeleitet werden kann, dass ein semantisches Verständnis der Meme-Texte den größten Beitrag zu der Erkennung hasserfüllter Memes liefert. Das Assoziations-Modul spielt eine weitere wichtige Rolle bei der Klassifizierung der Memes, wobei dessen Einsatz auf den HM-Daten wesentlich erfolgreicher ist als auf dem MAMI-Datensatz. Dieser Zusammenhang resultiert aller Wahrscheinlichkeit nach daraus, dass der verwendete Assoziationsgraph vorrangig auf Basis der HM-Trainingsdaten erstellt wurde und lediglich durch eine verhältnismäßig geringe Anzahl an Assoziationen aus den MAMI-Trainingsdaten ergänzt wurde. Somit zeigt sich, dass die Stärke des Assoziations-Moduls auf der Einbringung möglichst domänenspezifischen Kontextwissens beruht. Dass das Stimmungs-Modul lediglich in der HM-Modulmenge zu finden ist, ist vermutlich darauf zurückzuführen, dass die HM-Memes ein breiteres Spektrum des Hasses, darunter auch weniger subtile und vulgärere Hassformen, abbilden. Die Verwendung des Objekt-Moduls führt lediglich auf den MAMI-Daten zu einer Verbesserung des AUC-ROC Scores, obwohl der Anteil an Memes mit erkannten Objekten in dem MAMI-Datensatz kleiner ist als in dem HM-Datensatz. Auf den HM-Daten werden die Meme-Bilder stattdessen durch das Kachel-Modul repräsentiert, was eventuell auf die Semidubletten des HM-Datensatzes zurückzuführen ist. Wo das Objekt-Modul lediglich die im Bild enthaltenen Objekte verarbeitet, besitzt das Kachel-Modul die Fähigkeit, alle Abschnitte des Eingabebildes zu berücksichtigen. Dadurch kann es theoretisch dazu kommen, dass das Kachel-Modul die visuelle Darstellung des Meme-Textes interpretiert, wodurch es Semidubletten implizit enttarnen kann. Das Bild-Modul ist in keiner der beiden finalen Modulmengen enthalten, was die Vermutung nahelegt, dass die gemeinsame Verarbeitung der gesamten Eingabebilder zu viel semantisches Rauschen, etwa durch unwichtige Elemente der Bildhintergründe, mit sich bringt.

Um Erkenntnisse über die Verarbeitung der durch die Module erstellten Positionen der Transformer-Eingabesequenz gewinnen zu können, wur-

de eine Methode zur Visualisierung der Attention-Scores implementiert. Die Attention-Scores werden innerhalb der Transformer-Kodierer genutzt, um die Positionen der Transformer-Eingabesequenz miteinander in Beziehung zu setzen und zu einer Transformer-Ausgabesequenz zu verarbeiten. Die gewählte Darstellungsform der Attention-Scores beruht auf einer Matrix, dessen Zeilen und Spalten mit den Namen der verwendeten Module zur Erstellung der Transformer-Eingabesequenz beschriftet sind. Die Matrix gibt an, wie sehr die Ausgaben der in den Zeilen festgehaltenen Module mit den Ausgaben der in den Spalten festgehalten Module in Beziehung gesetzt werden, um die Transformer-Ausgabesequenz zu bilden. Dabei ist zu bedenken, dass Positionen der Transformer-Eingabesequenz auch mit sich selbst in Beziehung gesetzt werden können. In Abbildung 12 sind zwei Darstellungen von Memes aus dem MAMI-Datensatz sowie die dafür berechnet Attention-Scores dargestellt. Zu dem Meme auf der linken Seite wurden dabei keine Assoziationen im Wissensgraphen des Assoziations-Moduls hinterlegt, wodurch die Transformer-Kodierer die Ausgabe des Assoziations-Moduls bei der Berechnung der Transformer-Ausgabesequenz nicht berücksichtigen. Dasselbe Verhalten kann bezüglich des Memes auf der rechten Seite in Abbildung 12 beobachtet werden, wobei die Memes in der entsprechenden Abbildung aus dem HM-Datensatz stammen. Die Ausgabe des Assoziationsmoduls wird, in Anlehnung an (Vaswani et al., 2017), dennoch durch die Informationen der anderen Modulausgaben angereichert, wodurch die Attention-Scores ausgehend vom Assoziations-Modul zu allen anderen Modulen positiv ist.

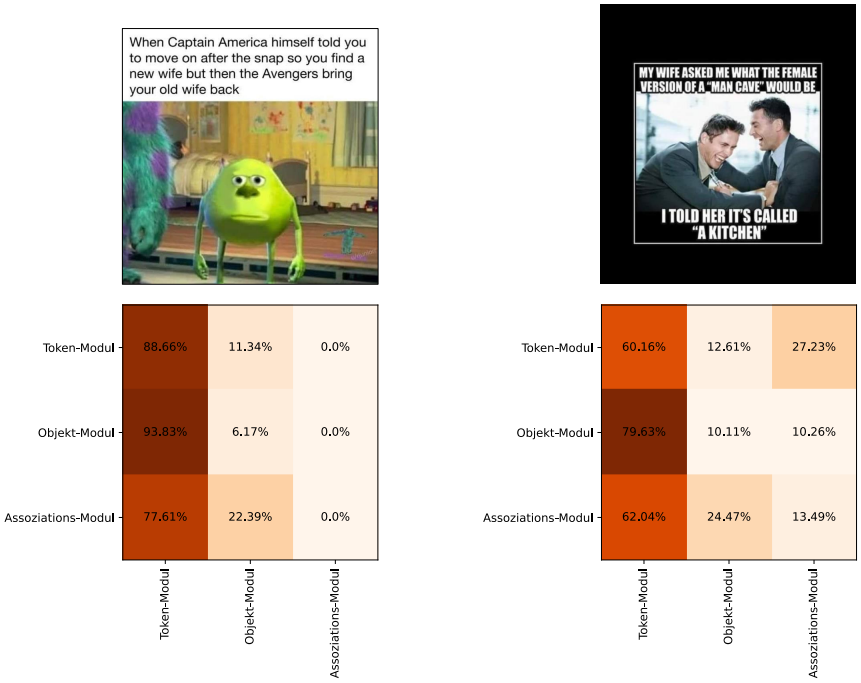

Abbildung 12: Die gemittelten Attention-Scores des MAMI-Netzwerkes für zwei Memes aus dem MAMI-Datensatz

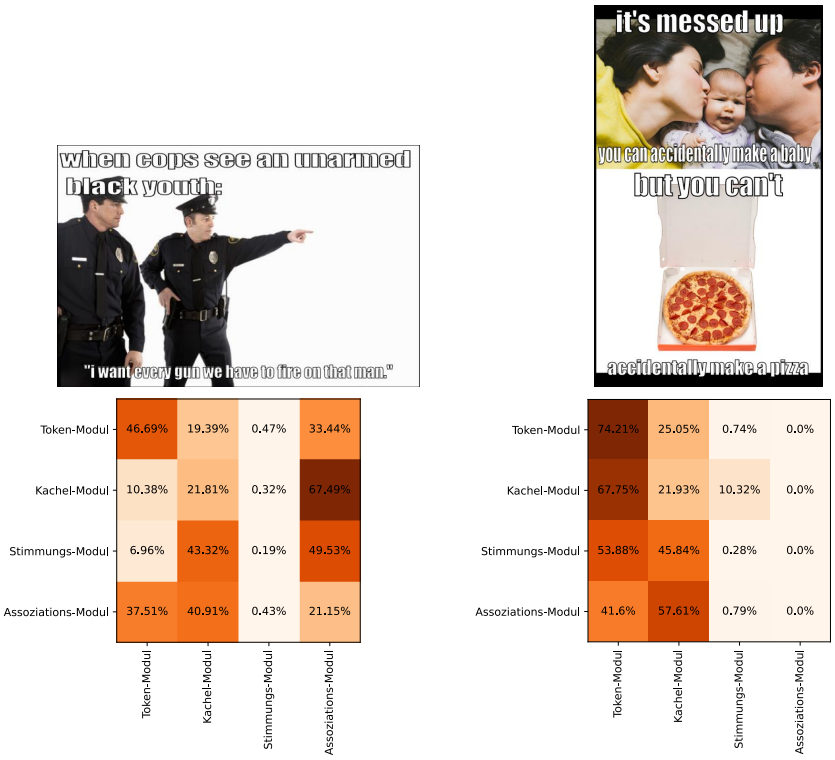

Abbildung 13: Die gemittelten Attention-Scores des HM-Netzwerkes für zwei Memes aus dem HM-Datensatz

7.1.2 Interpretation des datensatzübergreifenden Trainings

Auch die Ergebnisse des datensatzübergreifenden Trainings zeigen, dass die beiden Datensätze sich stark voneinander unterschieden und eine gegenseitige Nutzbarmachung für die Verbesserung der Netzwerkvorhersagen auf den beiden Datensätzen nicht zustande kommen kann. Die insignifikante Verbesserung, welche aus der Anwendung von CROSS-MAMI resultiert, stellt einen vernachlässigbaren Einzelfall dar.

Die AUC-ROC Scores in den Vortrainingsphasen von CROSS-HM und CROSS-MAMI liegen nah an den AUC-ROC Scores der jeweiligen Trainingsphasen. Daraus kann abgeleitet werden, dass die generalisierenden Merkmale der jeweiligen Trainingsdaten bereits größtenteils in den Vortrainingsphasen erkannt werden. Aus den relativ geringen AUC-ROC Scores in den Vortrainingsphasen von MAMI-HM und HM-MAMI geht hervor, dass sich die generalisierenden Merkmale der beiden Datensätze stark voneinander unterschei-

den. Am auffälligsten wird der Unterschied zwischen den beiden Datensätzen in den Vortrainingsphasen von SEP-HM und SEP-MAMI, in welchen eine Unterscheidung zwischen den Memes beider Datensätze beinahe vollständig erlernt wird. Ein semantischer Unterschied zwischen den beiden Datensätzen lässt sich auch aus den im folgenden Abschnitt analysierten Ergebnissen auf den Testdaten erkennen.

7.1.3 Interpretation der Ergebnisse auf den Testdatensätzen

Aus den Ergebnissen in Tabelle 5 geht hervor, dass auf den MAMI-Daten ein besserer AUC-ROC Score mittels des Late-fusion-Transformers gewonnen werden kann, während auf den HM-Daten ein besserer AUC-ROC Score durch die Anwendung des Early-fusion-Transformers erzielt wird.

Die Late-fusion-Varianten verarbeiten die Ausgaben der Module zur Bildung der Transformer-Eingabesequenzen getrennt voneinander. Folglich kann aus den Testergebnissen abgeleitet werden, dass die Erlernung der HM-Daten eine intensive Berücksichtigung modulübergreifender, und somit auch modalitätsübergreifender, Zusammenhänge zwischen den Positionen erfordert, während dies auf den MAMI-Daten nicht der Fall ist. Zurückzuführen ist dies vermutlich auf die HM-Semidubletten, welche ein Hindernis für die unimodale Verarbeitung des Datensatzes darstellen sollen und somit eine Erlernung eben jener Zusammenhänge erfordern (Kiela et al., 2020a). Auch, wenn der in dieser Arbeit ausschlaggebende AUC-ROC Score der auf den HM-Daten trainierten Late-fusion-Variante geringer ist als der seines Early-fusion-Gegenstücks, so kann sie dennoch sinnvoll eingesetzt werden. Sie erkennt zwar lediglich 31,31 % der hasserfüllten Memes, doch falls ein Meme von ihr als hasserfüllt vorhergesagt wird, ist dies in 67,12 % der Fälle richtig. Dadurch kann diese Variante in Einsatzfeldern eingesetzt werden, in welchen möglichst wenige Memes fälschlicherweise als hasserfüllt erkannt werden sollen und somit eine Überregulierung vermieden werden soll.

Wie es sich bereits auf den Validierungsdaten abzeichnete, sind die erreichten Scores auf den MAMI-Daten erheblich größer als die auf den HM-Daten. Daraus ist abzuleiten, dass die Erlernung der generalisierenden Merk-

male der MAMI-Daten leichter ist, wofür eine Reihe von Ursachen zuständig sein kann:

Wo der HM-Datensatz einen großen Teil des Hassspektrums abbildet, fokussiert sich der MAMI-Datensatz lediglich auf frauenfeindliche Memes und somit auf einen geringeren Teil des Hassspektrums. Somit sind die hasserfüllten Memes im MAMI-Datensatz verhältnismäßig ähnlich zueinander, woraus resultiert, dass die generalisierenden Muster ausgeprägter und leichter zu erlernen sind. Unterstützt wird dies zusätzlich dadurch, dass der MAMI-Trainingsdatensatz etwas größer ist als der HM-Trainingsdatensatz. Unter Berücksichtigung der Wichtigkeit des Token-Moduls kann zudem die, wie in Tabelle1 festgehaltene, deutlich größere durchschnittliche Länge der Meme-Texte im MAMI-Datensatz dazu führen, dass die darauf erreichten Scores größer sind als die auf den HM-Daten. Zuletzt besteht die Möglichkeit, dass die Semidubletten des HM-Datensatzes einen negativen Einfluss auf die Netzwerkvorhersagen haben. Um diesen und weitere Zusammenhänge zu prüfen, wird im nächsten Kapitel eine intensive Analyse der Netzwerkvorhersagefehler durchgeführt.

7.2 Analyse der Netzwerkvorhersagefehler auf den Testdatensätzen

Zunächst wird der Einfluss der Semidubletten des HM-Datensatzes betrachtet, bevor überprüft wird, inwiefern die Fehler des Netzwerkes auf die Verwendung des Assoziations-Moduls zurückzuführen sind.

7.2.1 Der Einfluss der Semidubletten

Wie zuvor beschrieben beinhaltet der HM-Datensatz eine Reihe von Semidubletten. In diesem Abschnitt wird der Einfluss klassenübergreifender Semidubletten auf die Netzwerkvorhersagen überprüft. Dafür wurde eine Liste aller Trainingsdatenpunkte erstellt, welche eine Semidublette zu einem Datenpunkt im HM-Testdatensatz darstellen. Ausgehend von dieser Liste wurden alle semidublettenverwirrten Testdatenpunkte identifiziert. Ist ein Testdatenpunkt als harmlos gekennzeichnet, so wird er der Menge der text-

dublettenverwirrten Testdatenpunkte (TDVT) hinzugefügt, falls mindestens eine seiner Textdubletten im Trainingsdatensatz hasserfüllt ist. Ist ein Testdatenpunkt hingegen hasserfüllt, so wird der Trainingsdatensatz nach harmlosen Textdubletten durchsucht. Analog dazu wurden die bilddublettenverwirrten Testdatenpunkte (BDVT) identifiziert. Die Vereinigung von TDVT und BDVT wird im Rahmen dieser Arbeit als semidublettenverwirrte Testdatenpunkte (SDVT) bezeichnet. SDVT enthält weniger als $|\text{TDVT}| + |\text{BDVT}|$ Elemente, weil einige Memes sowohl Bilddubletten als auch Textdubletten besitzen. Um den Arbeitsaufwand zur Identifikation der Semidubletten drastisch zu verringern, wurden die in Anhang 3 beschriebenen Verfahren zur Durchsuchung des Datensatzes angewendet, wodurch eine geringe Anzahl an Semidubletten unter Umständen nicht entdeckt wurde.

Im Anschluss an die zuvor beschriebene Datenanalyse wurde der fertig trainierte HM-Transformer auf vier Untermengen des Testdatensatzes evaluiert. Bei den Untermengen handelt es sich um BDVT, TDVT, SDVT und die Menge aller Testdatenpunkte, welche nicht in SDVT enthalten, also nicht semidublettenverwirrt, sind. Die Ergebnisse dieser Evaluierung wurden, analog zu dem Verfahren auf dem Validierungsdatensatz, über drei Netzwerkinstanzen hinweg gemittelt und sind Tabelle 6 zu entnehmen.

Im direkten Vergleich zeigt sich, dass der AUC-ROC Score auf allen Testdatenpunkten erheblich geringer ist als auf den Datenpunkten, welche keinerlei Semidubletten-Verwirrung erleiden. Am niedrigsten, und insbesondere niedriger als auf den BDVT, ist der erreichte AUC-ROC Score auf den TDVT. Daraus lässt sich ableiten, dass das Netzwerk dazu tendiert, Memes mit gleichen Meme-Texten dieselbe Klasse zuzuweisen, ohne dabei große Rücksicht auf die Meme-Bilder zu nehmen.

Tabelle 6: Analyse der semidublettenbedingten Netzwerkvorhersagefehler auf dem HM-Testdatensatz

Test-Datensatz	**Memes**	**AUC-ROC**
Alle	900	71.46 %
Alle ohne SDVT	560	81.28 %
SDVT	340	55.13 %
TDVT	199	41.68 %
BDVT	209	66.55 %

7.2.2 Der Einfluss gefundener Assoziationen

Bei der Einbringung eines Assoziations-Moduls, wie es in dieser Arbeit definiert wurde, besteht die Gefahr, dass das Netzwerk eine gefundene Assoziation, unabhängig von ihrer Semantik, als eindeutigen Indikator für das Vorhandensein hasserfüllter Inhalte nutzt. Ist dies der Fall, so werden alle Memes mit einer hinterlegten Assoziation als hasserfüllt klassifiziert, wohingegen alle Memes ohne eine hinterlegte Assoziation als harmlos klassifiziert werden. Um zu überprüfen, ob ein dementsprechendes Netzwerkverhalten vorliegt, wurden die in Tabelle 7 dargestellten Informationen erhoben. Dafür wurden die beiden Testdatensätze aufgeteilt in Memes, für welche Assoziationen im Wissensgraphen hinterlegt sind, und Memes, für welche dies nicht der Fall ist. Anschließend wurden die resultierenden Untermengen anhand ihrer Annotationen erneut aufgeteilt. Zuletzt wurde überprüft, wie groß der Anteil der durch die Netzwerke korrekt klassifizierten Memes pro Untermenge ist. Die Ergebnisse dieser Analyse zeigen, dass die Netzwerke tatsächlich dazu tendieren, Memes mit hinterlegten Assoziationen als hasserfüllt, und Memes ohne Assoziationen als harmlos zu klassifizieren. Besonders deutlich lässt sich dies auf den HM-Daten feststellen, auf welchen das Assoziations-Modul, wie Abschnitt 7.1.1 zu entnehmen, einen größeren Einfluss auf die Netzwerkvorhersagen hat. Dennoch lässt sich feststellen, dass das Vorhandensein von Assoziationen keine binäre Entscheidungsgrundlage für die Netzwerke darstellt. Stattdessen beurteilen die Netzwerke die Semantik der gefunden Assoziationen im Kontext der restlichen Transformer-Eingabesequenz.

Tabelle 7: Der Einfluss gefundener Assoziationen auf die Netzwerkvorhersagen

Datensatz	Assoziationen	Annotation	Memes	Anteil richtig klassifizierter Memes
HM	> 0	Hasserfüllt	189	80.07 %
	0	Harmlos	374	84.76 %
	0	Hasserfüllt	124	32.8 %
	> 0	Harmlos	213	47.1 %
MAMI	> 0	Hasserfüllt	301	87.71 %
	0	Harmlos	255	74.38 %
	0	Hasserfüllt	202	73.93 %
	> 0	Harmlos	242	61.57 %

7.3 Einschränkungen und zukünftige Forschungsrichtungen

Eine maßgebliche Einschränkung dieser Arbeit ist, dass der zeitliche Rahmen keine Durchführung intensiven Vortrainings mittels der in Anhang 1 aufgelisteten Verfahren und Datensätze erlaubte. Stattdessen wurde lediglich auf die in Abschnitt 5.4 beschriebenen datensatzübergreifenden Trainingsverfahren zurückgegriffen. Darüber hinaus wurde keine intensive Evaluierung zweitrangiger Hyperparameter, insbesondere der Modulanzahl des verwendeten ResNets, durchgeführt, was zu einer Veränderung der Ergebnisse hätte führen können.

Die Versuchsergebnisse zeigen, dass spürbare Unterschiede zwischen den beiden verwendeten Datensätzen bestehen, jedoch wurden sie lediglich in ihrer Gesamtheit betrachtet. In einem größeren zeitlichen Rahmen hätte getestet werden können, inwiefern die Hinzugabe heuristisch ermittelter Untermengen des jeweils anderen Datensatzes zu einer Verbesserung der Ergebnisse führen kann.

Eine weitere Einschränkung dieser Arbeit betrifft die automatische Erkennung von Objekten, welche mittels eines auf COCO (Lin et al., 2014) vortrainierten Netzwerkes basiert, wohingegen die meisten in Anhang 1 erwähnten Netzwerke auf ein Netzwerk zurückgreifen, welches auf Visual Genome (Krishna et al., 2017) vortrainiert wurde. Ein Vergleich der beiden Objekterkennungsverfahren konnte aus zeitlichen Gründen nicht durchgeführt werden,

hätte allerdings gegebenenfalls einen positiven Einfluss auf die Stärke des Objekt-Moduls.

Neben der Auflösung der oben genannten Einschränkungen lassen sich aus den in dieser Arbeit gewonnen Erkenntnissen mehrere vielversprechende zukünftige Forschungsrichtungen ableiten:

Zunächst kann ein direkter Vergleich des Assoziations-Moduls mit der in (Liu et al., 2019; Wang et al., 2020b) vorgestellten Methodiken zur Einbringung von Wissensgraphen weitere Einblicke in dessen Güte hervorbringen. Auch bleibt offen, inwiefern das einfache, lediglich durch einen Annotator durchgeführte, Verfahren zur Bildung des Wissensgraphen verbessert werden kann und inwiefern Varianten des Assozaitions-Moduls in weiteren Domänen eingesetzt werden können.

Die dynamische Entwicklung von Memes stellt eine langfristige Herausforderung dar, für dessen Lösung noch kein Konzept entwickelt wurde. Neben dem Aufkommen neuartiger, wie den in dieser Arbeit angesprochenen Corona-Memes, verändert sich auch die Interpretation bereits bestehender Memes. Ein prominentes Beispiel stellt dabei das *Pepe the Frog*-Meme dar, welches zunächst für harmlose Zwecke eingesetzt, jedoch seit 2015 von zersetzenden Kräften instrumentalisiert wurde (Sierpinski, 2016). Die genannten Beispiele geben einen Hinweis darauf, dass die Verarbeitung von Memes mit Hinblick auf den aktuellen zeitlichen Kontext geschehen muss. Um dieser Anforderung gerecht zu werden, müssen bestehende Meme-Datensätze stets um aktuelle Memes ergänzt und veraltete Meme-Interpretationen gegebenenfalls entfernt werden. Auch zeitsensitive Netzwerke könnten eine Rolle in der Lösung dieser Aufgabe darstellen.

Die erzielten Scores auf dem MAMI-Datensatz geben, mit Hinblick auf die verhältnismäßig schlechten Ergebnisse auf dem HM-Datensatz, einen Hinweis darauf, dass der Fokus auf kleinere Ausschnitte des Hassspektrums zu vielversprechenden Ergebnissen führt. Daraus lässt sich ableiten, dass sich zukünftige Datensätze auf die gezielte Behandlung einzelner Hassunterkategorien fokussieren sollten. Das gesamte Hassspektrum kann daraufhin durch ein Ensemble mehrerer Netzwerke erkannt werden, welche sich auf jeweils eine Unterkategorie der Hassrede fokussieren.

Auch zeigen die Ergebnisse, dass die Einbringung von Semidubletten das multimodale Bewusstsein von Transformer-Netzwerken stärken kann, weshalb zukünftige Datensätze diese berücksichtigen sollten, um subtile Meme-Unterschiede erlernbar zu machen.

8 Fazit

Im Rahmen dieser Arbeit wurden zwei übergeordneter Forschungsfragen behandelt. Zum einen wurde überprüft, welche Module zur Bildung der Transformer-Eingabesequenz am besten für die Erkennung hasserfüllter Memes mittels multimodaler Transformer-Netzwerke geeignet sind. Zum anderen wurde untersucht, welche Erkenntnisse aus der In-Beziehung-Setzung zweier Datensätze, welche sowohl hasserfüllte als auch harmlose Memes enthalten, gewonnen werden können. In diesem Kapitel werden die maßgeblichen Erkenntnisse der vorliegenden Arbeit mit Hinblick auf die formulierten Forschungsfragen zusammengefasst.

Die Identifikation der besten Module zur Bildung der Transformer-Eingabesequenz hat ergeben, dass die Erfassung der textuellen Semantik der Memes die größte Rolle bei der Erkennung hasserfüllter Memes spielt. Darüber hinaus zeigt sich, dass die Infusion domänenspezifischen Kontextwissens mittels eines speziellen Wissensgraphen ebenfalls eine erhebliche Unterstützung bei der Identifikation hasserfüllter Memes spielen kann. Die Ergebnisse geben einen Hinweis darauf, dass der Grad der Wirksamkeit eines solchen Mechanismus maßgeblich durch die Domänenspezifizität des Wissensgraphen beeinflusst wird. Auch zeigte sich, dass eine Approximation der Stimmungslage der Meme-Texte unter Umständen eine unterstützende Funktion bei der Erkennung hasserfüllter Memes haben kann. Bezogen auf die Repräsentation der Meme-Bilder ergab sich, dass eine einfache Verarbeitung mittels eines konvolutionalen neuronalen Netzwerkes nicht zu einer Verbesserung der Netzwerkvorhersagen führt, wohingegen die Verarbeitung gleichgroßer Bildabschnitte oder informationsschwangerer Bildregionen sinnvolle Informationen für den Transformer bereitstellt. Es zeigte sich, dass sich

die Methoden zur Bildung der Transformer-Eingabesequenzen auf den beiden untersuchten Meme-Datensätzen unterscheiden. Zukünftige Forschungsarbeiten können auf den gewonnenen Ergebnissen aufbauen und prüfen, inwiefern die genutzte Technik zur Einbringung eines Wissensgraphen auf weitere Felder angewendet werden kann und welche Methodik zur Erstellung des Wissensgraphen dafür am besten geeignet ist. Auch bleibt offen, inwiefern die ermittelten Transformer durch die Anwendung zeitintensivem Vortrainings und Hyperparameter-Optimierung verbessert werden können.

Die In-Beziehung-Setzung der beiden Datensätze hat eine Reihe von Erkenntnissen zur Folge. Mehrere Versuchsergebnisse zeigen, dass ein signifikanter Unterschied zwischen den beiden Datensätzen vorliegt, wodurch eine gegenseitige Nutzbarmachung der beiden Datensätze zur Erzielung besserer Netzwerkvorhersagen nicht zustande kommen konnte. Aus den gewonnenen Einblicken konnten allerdings mehrere Anforderungen an zukünftige domänenspezifische Datensätze abgeleitet werden. Zunächst wurden Hinweise darauf deutlich, dass die automatische Erkennung hasserfüllter Memes deutlich besser erlernt werden kann, wenn der zu Grunde liegende Datensatz sich auf eine Unterkategorie der Hassrede fokussiert. Der Inhalt zukünftiger Datensätze sollte sich somit beispielsweise auf LGBTQ-feindliche oder islamophobe Memes fokussieren, anstatt Memes beider Kategorien zu umfassen. Darüber hinaus wurden Hinweise darauf identifiziert, dass die Verwendung von Semidubletten, Memes welche sich lediglich in einer ihrer Modalitäten unterscheiden, das multimodale Verständnis von Transformern erweitern können, weshalb diese in zukünftigen Datensätzen eingebracht werden sollten, um nuancierte Unterschiede zwischen den Memes erlernbar zu machen.

Neben den bisher erwähnten Möglichkeiten zur Einbringung der gewonnenen Erkenntnisse wurde eine weitere Herausforderung für zukünftige Forschungsarbeiten identifiziert:

Die Verwendung von Memes verändert sich im Laufe der Zeit. Neue

Memes entstehen, andere werden verdrängt und einige Memes werden von menschenverachtenden Bewegungen adaptiert. Zukünftige Forschung sollte dementsprechend Rücksicht auf den zeitlichen Kontext von Memes legen und sowohl zeitbewusste multimodale Netzwerke als auch zeiterfassende Datensätze für die Bekämpfung von Hass-Memes einsetzen.

9 Quellen

(Antol et al., 2015):
Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, Lawrence C. Zitnick und Devi Parikh (2015):
'VQA: Visual Question Answering',
Proceedings of the IEEE International Conference on Computer Vision,
Santiago, Chile, 7-13 Dezember, S. 2425-2433.

(Ba et al., 2016):
Jimmy Ba, Jamie Ryan Kiros und Geoffrey E. Hinton (2016):
'Layer Normalization',
arXiv: 1607.06450.

(Baheti, 2021):
Pragati Baheti (2021):
'The Train, Validation, and Test Sets: How to Split Your Machine Learning Data',
https://www.v7labs.com/blog/train-validation-test-set,
abgerufen am 23.02.2022.

(Bilewicz und Soral, 2020):
Michal Bilewicz und Wiktor Soral (2020):
'Hate Speech Epidemic. The Dynamic Effects of Derogatory Language on Intergroup Relations and Political Radicalization',
Political Psychology, Vol. 41, S. 3-33,
John Wiley & Sons.

(Buchner, 2016):
Johannes Buchner (2016):
'ImageHash',
https://github.com/jgraving/imagehash,
abgerufen am 23.02.2021.

(Chen et al., 2020):
Yen-Chun chen, Linjie Li, Ahmed Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, Jingjing Liu (2020):
'UNITER: UNiversal Image-TExt Representation Learning',
Eurpoean conference on computer vision,
online, 23-28 August, S. 104-120.

(Das et al., 2017):
Abishek Das, Satwik Kottur, Khushi Gupta, Avi singh, Deshraj Yadav, Jose M. F. Moura, Devi Parikh und Dhruv Batra (2017):
'Visual Dialog',
Proceeedings of the IEEE conference on Computer Vision and Pattern Recognition,
Honolulu, USA, 21-26 Juli, S. 326-335.

(Devlin et al., 2019): Jacob Devlin, Ming-Wei Chang, Kenton Lee und Kristina Toutanova (2019):
'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding',
Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,
Minneapolis, USA, S. 4171-4186.

(Dosovitskiy et al., 2020):
Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit und Neil Houlsby (2020):
'An Image is Worth 16x16 Words: Transformers for Image recognition at Scale',
arXiv: 2010.11929.

(Elman, 1990):
Jeffrey L. Elman (1990):
'Finding Structure in Time',
Cognitive Science, Vol. 14, S. 179-211,
John Wiley & Sons.

(Fersini et al., 2022):
Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees und Jeffrey Sorensen (2022):
'SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification'
Proceedings of the 16th International Workshop on Semantic Evaluation,
Association for Computational Linguistics.

(Fitzpatrick, 2021):
Casey Fitzpatrick (2021):
'MEET THE WINNERS OF THE HATEFUL MEMES CHALLENGE',
https://www.drivendata.co/blog/hateful-memes-winners,
abgerufen am 23.02.2022.

(Gao et al., 2020):
Dehong Gao, Linbo Jin, Ben Chen, Minghui Qiu, Peng Li, Yi Wei, Yi Hu und Hao Wang (2020):
'FashionBERT: Text and Image Matching with Adaptive Loss for Cross-modal Retreival',
Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, USA, 25-30 Juli, S. 2251-2260.

(Goodfellow et al., 2016):
Ian Goodfellow, Yoshua Bengio und Aaron Courville (2016):
'Deep Learning',
MIT press, Cambridge, USA.

(Hanley und Mcneil, 1982):
James A. Hanley und Barbara J. Mcneil (1982):
'The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve',
Radiology, Vol. 143, S. 29-36,
RSNA.

(He et al., 2015):
Kaiming He, Xiangyu Zhang, Shaoqing Ren und Jian sun (2015):
'Deep Residual Learning for Image Recognition',
2016 IEEE Conference on Computer Vision and Pattern Recognition,
Las Vegas, USA, 27-30 Juni, S. 770-778.

(Hinton et al., 2012):
Geoffrey E. hinton, Nitish Srivastava, Alex Krizhevsky, Ilja Sutskever, Ruslan Slakhutdinov (2012):
'Improving neural networks by preventing co-adaption of feature detectors',
arXiv: 1207.0580.

(Hochreiter und Schmidhuber, 1997):
Sepp Hochreiter und Jürgen Schmidhuber (1997):
'Long short-term memory',
Neural Computation, Vol. 9, S. 1735-1780,
MIT Press.

(Hu und Singh, 2021):
Ronghang Hu und Amanpreet Singh (2021):
'Transformer is All You Need: Multimodal Multitask Learning with a Unified Transformer',
arXiv: 2102.10772.

(Huang et al., 2020):
Zhicheng Huang, Zhaoyang Zen, Bei Liu, Dongmei Fu und Jianlong Fu (2020):
'Pixel-BERT: Aligning Image Pixels with Text by Deep Multi-Modal Transformers',
arXiv: 2004.00849.

(Hudson und Manning, 2019):
Drew A. Hudson und Christopher D. Manning (2019):
'GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering',
2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 15-20 Juni, S. 6693-6702.

(Hugging Face, 2021a):
Hugging Face (2021):
'bert-base-uncased',
https://huggingface.co/bert-base-uncased,
abgerufen am 23.02.2022.

(Hugging Face, 2021b):
Hugging Face (2021):
'vit-base-patch16-224-in21k',
https://huggingface.co/google/vit-base-patch16-224-in21k,
abgerufen am 23.02.2022.

(Hutto, 2021):
Clayton J. Hutto (2021):
'vaderSentiment',
https://github.com/cjhutto/vaderSentiment,
abgerufen am 23.02.2022.

(Hyugen AI, 2021):
Hyugen AI (2021):
'Transformers in Pytorch from scratch for NLP Beginners',
https://hyugen-ai.medium.com/transformers-in-pytorch-from-scratch-for-nlp-beginners-ff3b3d922ef7,
abgerufen am 23.02.2022.

(Kiela et al., 2020a):
Duowe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswani, Amanpreet Singh, Pratik Ringshia und Davide Testuggine (2020):
'The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes',
Advances in Neural Information Processing Systems, Vol. 33, S. 2611-2624, Curran Associates, Inc..

(Kiela et al., 2020b):
Douwe Kiela, Hamed Firooz und Aravind Mohan (2020):
'Hateful Memes Challende and dataset for research on harmful multimodal content',
https://ai.facebook.com/blog/hateful-memes-challenge-and-data-set,
abgerufen am 24.02.2022.

(Kingma und Ba, 2014):
Diederik P. Kingma und Jimmy Ba (2014):
'Adam: A method for stochastic optimization',
arXiv: 1412.6980.

(KnowYourMeme, 2021):
KnoqYourMeme (2021):
'Free Helicopter Rides',
https://knowyourmeme.com/memes/free-helicopter-rides,
abgerufen am 24.02.2022.

(Krishna et al., 2017):
Ranjay Krishna, Zuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, david A. Shamma, Michal S. Bernstein und Li Fei-Fei (2017):
'Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations',
'International Journal of Computer Vision', Vol. 123, S. 32-73,
Springer.

(Krizhevsky et al., 2012):
Alex Krizhevsky, Ilya Sutskever und Geoffrey E. Hinton (2012):
'Imagenet classification with deep convolutional neural networks',
Advances in neural information processing systems, Vol. 25, S. 1097-1105,
Curran Associates, Inc..

(Li et al., 2019):
Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh und Kai-Wei Chang (2019):
'VisualBERT: A Simple and Performant Baseline for Vision and Language',
arXiv: 1908.03557.

(Li et al., 2020a):
Gen Li, Nan Duan, Yuejian Fang, Ming Gong und Daxin Jiang (2020):
'Unicoder-VL: A Universal Encoder for Vision and Language by Cross-Modal Pre-Training',
Proceedings of the AAAI Conference on Artificial Intelligence 34, New York, USA, 7-12 Feburar, S. 11336-11344.

(Li et al., 2020b):
Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi und Jianfeng Gao (2020):
'Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks',
arXiv: 2004.06165.

(Lin et al., 2014):
Tsung-Yi Lin, Michael Maire, Serge BelongieJ, ames Hays, Pietro Perona, Deva Ramanan, Piotr Dollár und C. Lawrence Zitnick (2014):
'Microsoft coco: Common objects in context',
European conference on computer vision 2014,
Zürich, Schweiz, 6-12 September, S. 740-755.

(Lin et al., 2020):
Jungyang Lin, An Yang, yichang Zhang, Jie Liu, Jingren Zhou und Hongxia Yang (2020):
'InterBERT: Vision-and-Language Interaction for Multi-modal Pretraining',
arXiv: 2003.13198.

(Lin et al., 2021):
Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, Jie Zhang, Jianwei Zhang, Xu Zou, Zhikang Li, Xiaodong Deng, Jie Liu, Jinbao Xue, Huiling Zhou, Jianxin Ma, Jin Yu, Yong Li, Wei Lin, Jingren Zhou, Jie Tang und Hongxia Yang (2021):
'M6: A Chinese Multimodal Pretrainer',
arXiv: 2103.00823.

(Lippe et al., 2020):
Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova und Helen Yannakoudakis (2020):
'A Multimodal Framework for the Detection of Hateful Memes',
arXiv: 2012.12871.

(Liu et al., 2019):
Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng und Ping Wang (2019):
'K-BERT: Enabling Language Representation with Knowledge Graph',
Proceedings of the AAAI Conference on Artificial Intelligence 34,
New York, USA, 7-12 Februar, S. 2901-2908.

(Loria, 2020):
Steven Loria (2020):
'TextBlob: Simplified Text Processing',
https://textblob.readthedocs.io/,
abgerufen am 24.02.2022.

(Loschilov und Hutter, 2017):
Ilja Lohchilov und Frank Hutter (2017):
'Decoupled weight decay regularization',
arXiv: 1711.05101.

(Lu et al., 2019):
Jiasen Lu, Dhruv Batra, Devi Parikh und Stefan Lee (2019):
'ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks',
Advances in Neural Information Processing Systems, Vol. 32, S. 13-23,
Curran Associates, Inc..

(Ma et al., 2019):
Xiaofei Ma, Zhiguo Wang, Patrick Ng, Ramesh Nallapati und Bing Xiang (2019):
'Universal Text Representation from BERT: An Empirical Study',
arXiv: 1910.07973.

(Mao, 2014): Kudo Mao (2014):
'Orthogonal forward selection and backward elimination algorithms for feature subset selection',
IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), Vol. 34, S. 629-634,
IEEE.

(Mikolov et al., 2013):
Tomas Mikolov, Kai Chen, Greg Corrado und Jeffrey Dean (2013):
'Efficient Estimation of Word Representations in Vector Space',
arXiv: 1301.3781.

(Murahari et al., 2020):
Vishvak Murahari, Dhruv Batra, Devi Parikh und Abhishek Das (2020):
'Large-scale Pretraining for Visual Dialog: A Simple State-of-the-Art Baseline',
ECCV 2020,
Glasgow, UK, 23-28 August, S. 336-352.

(Ontotext, 2022):
Ontotext (2022):
'What is a Knowledge Graph?',
https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph,
abgerufen am 24.02.2022.

(Ordonez et al., 2011):
Vicento Ordonez, Girish Kulkarni und Tamara Berg (2011):
'Im2Text: Describing Images Using 1 Million Captioned Photographs',
Advances in Neural Information Processing Systems, Vol. 24, S. 1143-1151,
Curran Associates Inc..

(Osterroth, 2015):
Andreas Osterroth (2015):
'Das Internet-Meme als Sprache-Bild-Text',
IMAGE. Zeitschrift für interdisziplinäre Bildwissenschaft Vol. 11, S. 26-46,
Herbert von Halem.

(Pascanu et al., 2013):
Razvan Pascanu, Thomas Mikolov und Yoshua Bengio (2013):
'On the difficulty of training recurrent neural networks',
Proceedings of the 30th International Conference on Machine Learning, Atlanta, USA, 16-21 Juni, S. 1310-1318.

(Powers, 2011):
David M. W. Powers (2011):
'Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation',
Journal of Machine Learning Technologies, Vol. 2, S. 37-63,
Bioinfo Publications.

(Qi et al., 2020):
Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti und Arun Sacheti (2020):
'ImageBERT: Cross-modal Pre-training with Large-scale Weak-supervised Image-Text Data',
arXiv: 2001.07966.

(Radford et al., 2018):
Alec Radford, Karthik Narasimhan, Tim Salimans und Ilja Sutskever (2018):
'Improving Language Understanding by Generative Pre-Training',
https://openai.com/blog/language-unsupervised.

(Ren et al., 2016):
Shaoqing Ren, Kaiming He, Ross Girshick und Jian Sun (2016):
'Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks',
Advances in Neural Information Processing Systems 28,
Montreal, Canada, 7-12 Dezember, S.91-99.

(Rosebrock, 2021):
Adrian Rosebrock (2021):
'PyTorch object detection with pre-trained networks',
https://www.pyimagesearch.com/2021/08/02/pytorch-object-detection-with-pre-trained-networks,
abgerufen am 24.02.2022.

(Rostamzadeh et al., 2018):
Negar Rostamzadeh, Seyedarian Hosseini, Thomas Boquet, Wojciech Stokowiec, Ying Zhang, Christian Jauvin und Chris Pal (2018):
'Fashion-Gen: The Generative Fashion Dataset and Challenge',
arXiv: 1806.08317.

(Sandulescu, 2020):
Vlad Sandulescu (2020):
'Detecting hateful memes using a multimodal deep ensemble',
arXiv: 2012.13235.

(Sharma et al., 2018):
Piyush Sharma, Nan Ding, Sebastian Goodman und Radu Soricut (2018):
'Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning',
Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),
Melbourne, Australien, Juli, S. 2556-2565.

(Sierpinski, 2016):
Diana Sierpinski (2016):
'Wie Pepe der Frosch zum Nazi wurde',
https://www.n-tv.de/panorama/Wie-Pepe-der-Frosch-zum-Nazi-wurde-article18747616.html,
abgerufen am 25.02.2022.

(Smith et al., 2017):
Samuel L. Smith, Pieter-Jan Kindermans und Quoc V. Le (2017):
, 'Don't Decay the Learning Rate, Increase the Batch Size',
arXiv: 1711.00489.

(Sun et al., 2019a):
Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, und Qun Liu (2019):
'ERNIE: Enhanced Representation through Knowledge Integration',
Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,
Florenz, Italien, 28. Juli bis 2. August, S. 1441-1451.

(Sun et al., 2019b):
Chen Sun, Fabien Baradel, Kevin Murphy und Cordelia Schmid (2019):
'Learning Video Representations using Contrastive Bidirectional Transformer',
arXiv: 1906.05743.

(Su et al., 2020):
Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei und Jifeng Dai (2020):
'VL-BERT: Pre-training of Generic Visual-Linguistic Representations',
arXiv: 1908.08530.

(Tan und Bansal, 2019):
Hao Tan und Mohit Bansal (2019):
'LXMERT: Learning Cross-Modality Encoder Representations from Transformers',
arXiv: 1908.07490.

(Torch Contributers, 2017):
Torch Contributers (2017):
'TORCHVISION.MODELS',
https://pytorch.org/vision/stable/models.html,
abgerufen am 25.02.2022.

(Torch Contributers, 2019):
Torch Contributers (2019):
'MODULE',
https://pytorch.org/docs/stable/generated/torch.nn.Module.html,
abgerufen am 25.02.2022.

(Vaswani et al., 2017):
Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser und Illia Polosukhin (2017):
'Attention is all you need',
Proceedings of the 31st International Conference on Neural Information Processing Systems,
Red Hook, USA, 4-7 Dezember, S. 6000-6010.

(Velioglu und Rose, 2020):
Riza Velioglu und Jewgeni Rose (2020):
'Detecting Hate Speech in Memes Using Multimodal Deep Learning Approaches: Prize-winning solution to Hateful Memes Challenge',
arXiv: 2012.12975.

(von Ahn, o. D.): Luis von Ahn:
'Useful Resources',
https://www.cs.cmu.edu/ biglou/resources,
abgerufen am 25.02.2022.

(Wang et al., 2020a):
Yue Wang, Shafiq Joty, Michael Lyu, Irwin King, Caiming Xiong, und Steven C.H. Hoi (2020):
'VD-BERT: A Unified Vision and Dialog Transformer with BERT',
Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),
Punta Cana, Dominikanische Republik, 8-12 November, S. 3325-3338.

(Wang et al., 2020b):
Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu ji, Guihong Cao, Daxin Jiang und Ming Zhou (2020):
'K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters',
arXiv: 2002.01808.

(Weijie et al., 2019):
Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei und Jifeng Dai (2019):
'VL-BERT: Pre-training of Generic Visual-Linguistic Representations',
arXiv: 1908.08530.

(Werbos, 1974):
Paul Werbos (1974):
'Beyond regression: new tools for prediction and analysis in the behavioral sciences.',
Ph. D. dissertation, Harvard University.

(Yu et al., 2020):
Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu und Haifeng Wang (2020):
'ERNIE-ViL: Knowledge Enhanced Vision-Language Representations Through Scene Graphs',
arXiv: 2006.16934.

(Young et al., 2014):
Peter Young, Alice Lai, Micah Hodosh, und Julia Hockenmaier (2014):
'From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions',
Transactions of the Association for Computational Linguistics, Vol. 2, S. 67-78,
MIT Press.

(Zhang et al., 2020a):
Jingzhao Zhang, Tianxing He, Suvrit Sra und Ali Jadbabaie (2020):
'Why gradient clipping accelerates training: A theoretical justification for adaptivity',
arXiv: 1905.11881.

(Zhang et al., 2020b):
Shengyu Zhang, Tan Jiang, Tan Wang, Kun Kuang, Zhou Zhao, Jianke Zhu, Jin Yu, Hongxia Yang und Fei Wu (2020):
'DeVLBert: Learning Deconfounded Visio-Linguistic Representations',
MM '20: Proceedings of the 28th ACM International Conference on Multimedia,
Seattle, USA, 12-16 Oktober, S. 4373-4382.

(Zhu et al., 2015):
Yuke Zhu, Oliver Groth, Michael Bernstein und Li Fei-Fei (2015):
'Visual7W: Grounded Question Answering in Images',
arXiv: 1511.03416.

(Zhu, 2020):
Ron Zhu (2020):
'Enhance Multimodal Transformer With External Label And In-Domain Pretrain: Hateful Meme Challenge Winning Solution',
arXiv: 2012.08290.

(Zhou et al., 2020):
Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J. Corso und Jianfeng Gao:
'Unified Vision-Language Pre-Training for Image Captioning and VQA',
arXiv: 1909.11059.

Anhang

Anhang 1: Zusammenfassung bestehender multimodaler Transformer

Tabelle 8 beschreibt 16 multimodale Transformer für die gemeinsame Verarbeitung von Bildern und Texten. Zur Einbringung von Eingabetexten in ihre Eingabesequenzen greifen die Transformer ausschließlich auf Varianten der in (Devlin et al., 2019) definierten Methodik beziehungsweise des in dieser Arbeit beschriebenen Token-Moduls zurück. Für die Repräsentation von Eingabebildern in den Transformer-Eingabesequenzen hingegen werden mitunter stark unterschiedliche Verfahren eingesetzt. Neben den Methoden zur Repräsentation ihrer Eingabebilder in ihren Eingabesequenzen werden die Transformer anhand ihrer Vortrainingsmethoden und ihren Vortrainingsdatensätzen beschreiben. Late-fusion-Transformer werden in Tabelle 8 durch ein * hinter ihrem Namen gekennzeichnet. In Tabelle 10 werden die Vortrainingsmethoden in fünf übergeordnete Kategorien unterteilt und ihre Eigenschaften beschrieben sowie ihre Eignung zur Anwendung auf Meme-Datensätzen eingeschätzt. Die Eigenschaften der verwendeten Vortrainingsdatensätze hingegen werden in Tabelle 9 zusammengefasst. Für das Verständnis dieses Anhangs empfiehlt es sich, Kapitel 4 der vorliegenden Arbeit vollständig gelesen zu haben.

Tabelle 8: Eine Zusammenfassung 16 verschiedener multimodaler Transformer.

Transformer	**Einbringung der Bildeingaben in die Transformer-Eingabesequenz**	**Multimodales Vortraining**	**Vortrainings-datensätze**
VL-BERT (Su et al., 2020):	Die durch Faster R-CNN ermittelten RoI-Merkmale werden errechnet und die Positionen der RoI werden mittels ihrer Eckkoordinaten kodiert. Um die Positionskodierung auf die gleiche Dimension zu bringen wie die RoI-Merkmalsvektoren, werden sie mittels trigonometrischer Funktionen auf die benötigte Größe projiziert. Die Summe eines RoI-Merkmalsvektoren und dessen Positionskodierung bildet jeweils eine Position der Transformer-Eingabesequenz von VL-BERT. Darüber hinaus addiert VL-BERT eine derartige Repräsentation des gesamten Eingabebildes auf alle Eingabesequenz-Positionen, welche Tokens des Eingabetextes repräsentieren.	MMLM, MMSM	CC
VD-BERT (Wang et al., 2020a)	Faster R-CNN ermittelt RoI-Merkmale und stellt diese als Vektoren dar. Auf die so ermittelten Vektoren werden Anreicherungsvektoren addiert, welche sowohl die Eckkoordinaten der RoI als auch deren relativen Bildanteil und die identifizierte Objektklasse der RoI sowie deren Vorkommenswahrscheinlichkeiten beinhalten. Jeder der dadurch entstandenen Vektoren bildet eine Position der Transformer-Eingabesequenz.	MMLM	VisDial
ViLBERT* (Lu et al., 2019)	Zunächst werden die visuellen Merkmale von RoI mittels Faster R-CNN ermittelt und als Vektoren dargestellt, bevor diese Vektoren durch die Addition von jeweils einem Anreicherungsvektor um zusätzliche Informationen erweitert werden. Die Anreicherungsvektoren beinhalten die relativen Eckkoordinaten und die relativen Bildanteile der RoI und werden mittels einer linearen Abbildung auf die Größe der Merkmalsvektoren projiziert. Die so erzeugten Vektoren bilden jeweils eine Position der Transformer-Eingabesequenz. Darüber hinaus wird eine weitere Position erzeugt, welche durch die Summe des Mittelwertes aller Merkmalsvektoren und einer speziellen Positionskodierung gebildet wird, welche das gesamte Eingabebild als RoI behandelt und dessen Position darstellt.	MMLM, MMSM, MM	CC
LXMERT* (Tan und Bansal, 2019)	Die durch Faster R-CNN ermittelten RoI-Merkmale werden errechnet, als Vektoren dargestellt und mittels Anchreicherungsvektoren um zusätzliche Informationen erweitert. Die Anreicherungsvektoren beinhalten dabei Informationen über die Eckkoordinaten der RoI und werden mittels einer linearen Abbildung auf die Größe der Merkmalsvektoren gebracht. Sowohl die Merkmalsvektoren als auch die Anreicherungsvektoren werden mittels Layer Normalization aneinander angeglichen und anschließend addiert.	MMLM, MMSM, MM	COCO, VG, VQA, GQA, VG-QA
VisualBERT (Li et al., 2019)	Zunächst werden RoI erkannt und mittels eines konvolutionalen Netzwerkes auf einen Merkmalsvektor abgebildet. Aus (Li et al., 2019) wird nicht ersichtlich, welche Verfahren dabei verwendet werden, allerdings ist naheliegend, dass für beide Verfahren Faster R-CNN verwendet wird. Falls der verwendete Datensatz Informationen über die Verflechtung von Text-Tokens zu Bildregionen enthält, addiert VisualBERT die Positionskodierungen der entsprechenden Text-Tokens auf die mit ihnen korrespondierenden Merkmalsvektoren der Bildregionen.	MMLM, MM	COCO

UNITER (Chen et al., 2020)	UNITER greift auf Faster R-CNN zurück, um RoI-Merkmale zu berechnen, welche mittels jeweils eines Anreicherungsvektors um eine Positionskodierung erweitert werden. Der Anreicherungsvektor enthält dabei die relativen Eckkoordinaten der RoI, die Breite und die Höhe der RoI sowie den relativen Bildanteil, welcher von den RoI eingenommen wird. Sowohl die Merkmalsvektoren als auch die Anreicherungsvektoren werden mittels einer linearen Abbildung auf die gleiche Größe projiziert und anschließend addiert. Die Ergebnisvektoren bilden jeweils eine Position der Transformer-Eingabesequenz.	MMLM, MMSM, MA, MM	COCO, VG, CC, SBUC
Unicoder-VL (Li et al., 2020a)	Faster R-CNN ermittelt RoI-Merkmale und reichert sie mittels eines Anreicherungsvektors an. Der Anreicherungsvektor besteht aus den relativen Eckkoordinaten der RoI und dem relativen Bildanteil, welchen die RoI einnehmen. Die Anreicherung geschieht mittels einer Addition der Merkmalsvektoren und der Anreicherungsvektoren, nachdem beide mittels einer linearen Schicht auf dieselbe Größe projiziert werden. Die Ergebnisvektoren bilden jeweils eine Position der Transformer-Eingabesequenz.	MMLM, MMSM, MM	CC, SBUC
Pixel-BERT (Huang et. al., 2020)	Pixel-BERT verarbeitet das gesamte Eingabebild mittels eines konvolutionalen Netzwerkes. Das konvolutionale Netzwerk stellt die visuellen Merkmale der Eingabebilder innerhalb mehrerer zweidimensionaler Kanäle dar, welche auf einen eindimensionalen Vektor abgeflacht und als Positionen der Transformer-Eingabesequenz genutzt werden.	MMLM, MM	COCO, VG
Oscar (Li et al., 2020b)	Analog zu den meisten anderen Architekturen in dieser Tabelle greift OSCAR auf die visuellen Merkmale zurück, welche mittels Faster R-CNN aus den RoI gewonnen werden. Anstatt die RoI-Positionen auf einen Vektor zu projizieren und auf die RoI-Merkmale zu addieren, konkateniert Oscar die beiden Informationsquellen zu einem gemeinsamen Vektor, welcher anschließend mittels einer linearen Abbildung auf die gewünschte Größe abgebildet wird. Zusätzlich zu dieser Repräsentation der Eingabebilder wird ein zusätzlicher Vektor pro erkanntem Objekt erstellt. Dafür werden die Bezeichnungen der erkannten Objekte als Tokens dargestellt auf einen Vektor abgebildet. Diese zusätzliche Repräsentation der Objekte soll die semantische Kluft zwischen den beiden Modalitäten schmälern.	MMLM, MMSM, MM	COCO, CC, SBUC, flickr30k, GQA, VQA, VG-QA
ERNIE-ViL* (Yu et al., 2020)	ERNIE-ViL errechnet RoI-Merkmale mittels Faster R-CNN erkannter Objekte und reichert diese um jeweils einen Vektor an, welcher die Eckkoordinaten der RoI und den relativen Bildanteil, welcher von den RoI eingenommen wird, beinhaltet. Die finalen Positionen der Transformer-Eingabesequenz werden durch die Addition der Merkmalsvektoren und ihrer korrespondierenden Anreicherungsvektoren gebildet. Zusätzlich zu diesen Positionen erstellt ERNIE-ViL eine weitere Position der Transformer-Eingabesequenz, welche die visuellen Merkmale des gesamten Eingabebildes enthält und diese mit dem Anreicherungsvektor addiert, welcher das gesamte Eingabebild als eine große RoI begreift.	MMLM, MMSM, MM	CC, SBUC

VLP (Zhou et al., 2020)	Die in VLP eingesetzte Variante des **Objekt-Segments** addiert, analog zu dessen Gegenstück in dieser Arbeit, die Regions-Merkmale auf dessen Anreicherungs-Vektor, verwendet dabei allerdings individuelle lineare Projektionen für die Objekt-Wahrscheinlichkeit und die RoI-Position. Erst nach der Anwendung der Projektionen werden diese normalisiert und zu einem Anreicherungs-Vektor konkateniert.	MMLM	CC
M6 (Lin et al., 2021)	Die Eingabebilder werden auf eine einheitliche Größe skaliert, in gelichgroße Bildkacheln unterteilt und die visuellen Merkmale jeder Bildkachel mittels eines konvolutionalen Netzwerkes extrahiert und auf einen Vektor abgebildet. Daraufhin werden die visuellen Merkmale der Bildkacheln durch eine Positionskodierung erweitert und als Positionen der Transformer-Eingabesequenz verwendet.	MMLM	M6-Corpus
FashionBERT (Gao et al., 2020)	Analog zu M6.	MMLM, MM	Fashion-Gen
ImageBERT (Qi et al., 2020)	ImageBERT greift auf Faster R-CNN zurück, um RoI-Merkmale zu berechnen und erweitert diese mittels Anreicherungsvektoren. Die Anreicherungsvektoren beinhalten Informationen über die Eckkoordinaten und die relativen Bildanteile der RoI. Die Merkmalsvektoren und die Anreicherungsvektoren werden mittels linearer Abbildungen auf die gleiche Größe projiziert und addiert, um finale Positionen der Transformer-Eingabesequenz zu bilden.	MMLM, MMSM, MM	LAIT, CC, SBUC
DeVLBert* (Zhang et al., 2020b)	Analog zu ViLBERT.	MMLM, MMSM	CC
UniT* (Hu und Singh, 2021)	Analog zu Pixel-BERT.	/	/
VisDial ViLBERT* (Murahari et al., 2020)	Analog zu ViLBERT.	MMLM, MMSM	CC, VQA
InterBERT (Lin et al., 2020)	Analog zu ViLBERT.	MMLM, MMSM, MM	CC, SBUC, COCO

Tabelle 9: Eine Zusammenfassung von Datensätzen für multimodales Vortraining.

Datensatz	**Beschreibung**
Conceptional Captions (CC) (Sharma et al., 2018)	CC enthält mehr als 3.300.000 Paare aus Bildern und Bildbeschreibungen, welche einer Vielzahl unterschiedlicher Internetseiten entnommen wurden.
COCO (Lin et al., 2014)	Ein Datensatz, bestehend aus mehr als 300.000 Bildern, in welchen 91 verschiedene Objektinstanzen in ihren natürlichen Umgebungen enthalten sind. Für jedes Bild wurden fünf Bildbeschreibungen festgehalten.
Visual Genome (VG) (Krishna et al., 2017)	Dieser Datensatz enthält mehr als 100.000 Bilder, zu welchen sowohl die in ihnen vorkommenden Objekte als auch deren Eigenschaften und Beziehungen zu anderen Objekten erfasst wurden. Zusätzlich wurden Bildbeschreibungen einzelner Bildapsekte und eine Graphstruktur zur Modellierung der Bildstruktur sowie Frage-Antwort Paare bezüglich der Bildinhalte festgehalten.
VQA (Antol et al., 2015)	VQA enthält etwa 250.000 Bilder, 760.000 Fragen bezüglich der Bildinhalte sowie 10.000.000 mögliche Antworten auf diese Fragen, welche entsprechend des Multiple-Choice Formats entweder richtig oder falsch sein können.
GQA (Hudson und Manning, 2019)	Dieser Datensatz enthält 22.000.000 Frage-Antwort Paare bezüglich einer Vielzahl von Bildern. Aus den enthaltenen Fragen lassen sich Eigenschaften über die im Bild dargestellten Objekte herleiten.
VG-QA (Zhu et al., 2015)	VG-QA enthält 74.300 Bilder aus dem COCO-Datensatz und stellt für jedes Bild Frage-Antwort Paare bezüglich der Bildinhalte, basierend auf den Grundbausteinen *what*, *where*, *who*,*why*, *when*, *how* und *which*, bereit.
VisDial (Das et al., 2017)	VisDial enthält 120.000 Bilder aus dem COCO-Datensatz und ergänzt jedes dieser Bilder um 10 Frage-Antwort Paare über den Bildinhalt.
SBU Captions (SBUC) (Ordonez et al., 2011)	SBU Captions enthält 1.000.000 mit Inhaltsbeschreibungen versehene Fotos.
flickr30k (Young et al., 2014)	Dieser Datensatz besteht aus 30.000 mit insgesamt 150.000 Inhaltsbeschreibungen versehenen Bildern.
M6-Corpus (Lin et al., 2021)	Dieser Datensatz enthält sowohl Bild-Text Paare als auch rein textuelle Datenpunkte. Bei der Erstellung des Datensatzes wurde darauf geachtet, dass die enthaltenen Textdaten in Chinesisch verfasst wurden, und weder hasserfüllte noch pornografische oder politische Themen behandeln. Das M6-Corpus enthält etwa 60.500.000 Bilder und 124.900.000 Textpassagen, womit es mit Abstand den größten Datensatz in dieser Aufzählung darstellt.
Fashion-Gen (Rostamzadeh et al., 2018)	Fashion-Gen enthält etwas weniger als 300.000 hochauflösende und mit Bildbeschreibungen versehene Bilder von Kleidungsstücken.
LAIT (Qi et al., 2020)	LAIT besteht aus 10.000.000 mit Beschreibungen versehenen Bildern.

Tabelle 10: Eine Zusammenfassung der multimodalen Vortrainingskategorien.

Methode	**Beschreibung**	**Anwendung auf Meme-Daten**
Multimodal Masked Language Modeling (MMLM)	Ein geringer Token-Anteil wird durch ein falsches Token ersetzt und das Netzwerk muss die entstehenden Lücken mit den korrekten Tokens füllen. Die falschen Tokens sind entweder spezielle[MASK]-Tokens oder zufällig ausgewählte andere Tokens. Zur Lösung dieser Aufgabe hat das Netzwerk Zugriff auf die restliche Transformer-Eingabesequenz. (Su et al., 2020)	Da sich die Meme-Texte nur über nuancierte und oft über zusätzliches Weltwissen gebildete semantische Verflechtungen auf die Meme-Bilder beziehen, ist ein Vortraining mittels MMLM auf Meme-Datensätzen vermutlich nicht sonderlich zielführend. Darüber hinaus besteht das Risiko, dass sich der Transformer darauf spezialisiert, die visuelle Repräsentation der Meme-Texte innerhalb der Eingabebilder zu erlernen, um die entstehenden Textlücken zu füllen. Somit würden die Eigenschaften des restlichen Bildes vermutlich kaum erlernt werden.
Multimodal Masked Scene Modeling (MMSM)	Dieses Verfahren stellt den visuellen Zwilling des MMLM dar. Anstelle einzelner Tokens werden Repräsentationen erkannter Objekte maskiert und mit Hilfe der restlichen Transformer-Eingabesequenz wiederhergestellt. Die Implementierung dieses Verfahrens hängt dabei von der Objektrepräsentation ab und kann beispielsweise durch die Bestimmung der Objektklasse oder die Regression der dazugehörigen visuellen Merkmale geschehen. Umsetzungen dieses Verfahrens werden somit auch als Masked Region Classification (Chen et al., 2020) oder Masked Region Classification with KL-Divergence (Chen et al., 2020) beziehungsweise RoI-Feature Regression (Tan und Bansal, 2019) bezeichnet.	Wie oben beschrieben bestehen lediglich nuancierte Beziehungen zwischen den beiden Meme-Modalitäten, wodurch auch die Mitglieder dieser Vortrainingskategorie beeinträchtigt werden.
Multimodal Matching (MM)	Das Netzwerk bekommt entweder ein zusammengehöriges Bild-Text Paar oder ein Tupel bestehend aus dem Bild eines Datenpunktes n und dem Text eines anderen Datenpunktes m als Eingabe. Das Netzwerk muss entscheiden, ob die beiden Eingabeelemente zu dem selben Datenpunkt gehören. (Tan und Bansal, 2019)	Die Durchführung von MM auf Meme-Datensätzen würde vermutlich dazu führen, dass das Netzwerk lernt, die visuelle Repräsenation der Meme-Texte in den Meme-Bildern lesen und diese mit der linguistischen Darstellung der Meme-Texte abzugleichen, wodurch eine weitestgehend triviale Lösung erzielt werden würde.
Multimodal Question Answering (MQA)	Das Netzwerk bekommt ein Bild sowie eine Frage zu dem Inhalt des Bildes gestellt und muss sie passend beantworten. (Tan und Bansal, 2019)	Methoden dieser Vortrainingskategorie sind vermutlich sehr gut geeignet um selbst die feingliedrigsten Beziehungen zwischen den beiden Meme-Modalitäten zu lernen und können, falls passende Frage-Antwort Paare zur Verfügung stehen, sogar zur Einspeisung weiterführenden Weltwissens genutzt werden. Aktuell existiert kein Datensatz zur Durchführung eines solchen Vortrainings.

Multimodal Alignment (MA)	Das Netzwerk wird dazu trainiert, die Transformer-Eingabesequenzpositionen, welche aus visuellen und linguistischen Eingabedaten erzeugt werden, mittels möglichst ähnlicher Vektoren zu repräsentierten. (Chen et al., 2020)	Dieses Verfahren wird durch die besonderen Eigenschaften von Memes nicht beeinträchtigt.

Anhang 2: Trainingsdetails

Während der Durchführung des Netzwerktrainings kann es zu einer Reihe von Komplikationen kommen. In diesem Abschnitt werden die Methoden erläutert, welche ergriffen wurden, um diese einzudämmen.

Neben dem in Abschnitt 4.2 beschriebenen Dropout-Verfahren wurden weitere Techniken zur Bekämpfung von Overfitting eingesetzt. So wurde der Optimierungsalgorithmus AdamW (Loschilov und Hutter, 2017) für die Anpassung der Netzwerkparameter verwendet, welcher unter Umständen zu besseren Netzwerkvorhersagen außerhalb der Trainingsdaten führen kann als der häufig verwendete Optimierungsalgorithmus Adam (Kingma und Ba, 2014).

Initiale Experimente haben ergeben, dass das in dieser Arbeit verwendete Netzwerktraining zu zunehmend größer werdenden Gradienten und somit zu wachsenden Parameterupdates führen kann. Extremfälle dieses Trainingsphänomens werden als das exploding gradients problem beschrieben, welches das Training negativ beeinflusst (Pascanu et al., 2013). In Anlehnung an (Pascanu et al., 2013) wurde bei der Durchführung der Trainingsprozesse auf gradient clipping, eine Methode zur Rückprojizierung großer Gradienten, zurückgegriffen. Ein positiver Nebeneffekt der Verwendung von gradient clipping ist, dass es die Trainingszeit verringern kann (Zhang et al., 2020a).

Neben den oben genannten Methoden wurde die schrittweise Verkleinerung der Parameterupdates mittels learning rate scheduling getestet, welches die Netzwerkvorhersagen außerhalb der Trainingsdaten positiv beeinflussen kann (Smith et al., 2017), jedoch hat dessen Verwendung nicht zu einer Verbesserung geführt, weshalb dessen Einsatz verworfen wurde.

Anhang 3: Identifikation der semidublettenverwirrten Testdatenpunkte

Die semidublettenverwirrten Testdatenpunkte können entweder durch Bilddubletten oder durch Textdubletten beeinträchtigt worden sein. Um die textdublettenverwirrten Testdatenpunkte zu identifizieren, musste lediglich für jeden Testdatenpunkt geprüft werden, ob Trainingsdatenpunkte der jeweils anderen Klasse mit exakt demselben Text existieren. Die Identifikation der bilddublettenverwirrten Testdatenpunkte gestaltete sich schwieriger, weil die Meme-Bilder eine visuelle Repräsentation der Meme-Texte beinhalten. Eine Prüfung der exakten Gleichheit aller Pixelintensitäten ist somit nicht zielführend. Um den Suchaufwand drastisch zu verringern, wurde auf den in (Buchner, 2016) zur Verfügung gestellten Average-hashing-Algorithmus zurückgegriffen, welcher zueinander ähnlich Bilder auf nah aneinander liegende Werte abbildet. Der Algorithmus wurde auf jeden Testdatenpunkt angewendet, um gleichartige Bilder im Trainingsdatensatz zu identifizieren. Die Güte dieses Verfahrens hängt von dem eingestellten Schwellenwert zur Identifikation ähnlicher Bilder ab, wodurch es dazu gekommen sein kann, dass eine geringe Anzahl an bilddublettenverwirrten Testdatenpunkte nicht identifiziert wurde.

Eidesstattliche Erklärung

Erklärung über das selbstständige Verfassen von Milan Kalkenings
Ich versichere hiermit, dass ich die vorstehende Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der obigen Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, habe ich in jedem einzelnen Fall durch die Angabe der Quelle bzw. der Herkunft, auch der benutzten Sekundärliteratur, als Entlehnung kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet und anderen elektronischen Text- und Datensammlungen und dergleichen. Die eingereichte Arbeit ist nicht anderweitig als Prüfungsleistung verwendet worden oder in deutscher oder in einer anderen Sprache als Veröffentlichung erschienen. Mir ist bewusst, dass wahrheitswidrige Angaben als Täuschung behandelt werden.

18.3.2022, Hildesheim — Milan Kalkenings

Datum, Ort — Unterschrift

Eidesstattliche Erklärung

Erklärung über das selbstständige Verfassen von Milan Kalkenings
Ich versichere hiermit, dass ich die vorstehende Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der obigen Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, habe ich in jedem einzelnen Fall durch die Angabe der Quelle bzw. der Herkunft, auch der benutzten Sekundärliteratur, als Entlehnung kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet und anderen elektronischen Text- und Datensammlungen und dergleichen. Die eingereichte Arbeit ist nicht anderweitig als Prüfungsleistung verwendet worden oder in deutscher oder in einer anderen Sprache als Veröffentlichung erschienen. Mir ist bewusst, dass wahrheitswidrige Angaben als Täuschung behandelt werden.

18.3.2022, Hildesheim — Milan Kalkenings

Datum, Ort — Unterschrift