Karlsruher Institut für Technologie

Einfluss dynamischer Kontexterweiterungen auf die Schlussfolgerungsfähigkeiten neuronaler Sprachmodelle

Bachelorarbeit von

Sina Schmitt

an der Fakultät für Informatik
Institut für Programmstrukturen und Datenorganisation (IPD)

Erstgutachterin:	Prof. Dr. Anne Koziolek
Zweitgutachter:	Prof. Dr. Ralf Reussner
Betreuungspersonen:	M.Sc. Jan Keim,
	Prof. Dr. Gregor Betz,
	M.Sc. Sophie Schulz

24. März 2021 – 23. August 2021

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

Abstract

Die meta-kognitive Strategie „laut nachzudenken“ kann auf neuronale Sprachmodelle übertragen werden, wie Betz et al. zeigen: Ein vortrainiertes Sprachmodell ist besser in der Lage, deduktive Schlussfolgerungsprobleme zu lösen, wenn es zuvor dynamische Problemelaborationen generiert [3]. Das Sprachmodell verwendet auf dem Datensatz von Betz et al. eine einfache Heuristik für seine Antwortvorhersage, die es mithilfe der selbst generierten Kontexterweiterungen effektiver einsetzen kann. In dieser Arbeit untersuche ich, wie dynamische Kontexterweiterungen die Performanz eines neuronalen Sprachmodells beeinflussen, wenn es *nicht* auf eine solche Heuristik zurückgreifen kann. Ich überprüfe (i) die Schlussfolgerungsfähigkeiten eines vortrainierten neuronalen Sprachmodells im Zero-Shot-Setting, (ii) den Einfluss verschiedener vorgegebener Kontexterweiterungen auf die Zero-Shot-Performanz und (iii) die Fähigkeiten des Sprachmodells, selbst effektive Kontexterweiterungen zu generieren und zu nutzen. Dazu erstelle ich einen synthetischen Datensatz mit deduktiven Schlussfolgerungsaufgaben unterschiedlicher Komplexität. Das verwendete Sprachmodell zeigt im Zero-Shot-Setting leichte bis moderate Schlussfolgerungsfähigkeiten. Anhand eines zweiten Datensatzes zeige ich, dass das Sprachmodell sogar eingeschränkt in der Lage ist, Schlussfolgerungen über erfundene Prädikate zu ziehen. Wie verschiedene Komponenten der Schlussfolgerungsaufgaben die Performanz des Sprachmodells beeinflussen, untersuche ich mithilfe synthetischer Kontexterweiterungen. Ich zeige unter Anderem, dass Aufgaben, deren Inhalte stärker verknüpft sind, dem Modell größere Schwierigkeiten bereiten. Außerdem demonstriere ich, dass Kontexterweiterungen ein wirkungsvolles Mittel sein können, um die Zero-Shot-Performanz von Sprachmodellen zu verbessern. Besonders hilfreich sind synthetische Kontexterweiterungen, die das Prinzip deduktiver Ableitungen veranschaulichen. Dynamische Kontexterweiterungen, die das Sprachmodell auf Grundlage der Problembeschreibung selbst erzeugt, beeinflussen die Vorhersage hingegen kaum. Few-Shot-Learning mit zwei Beispielen kann die Qualität der generierten Elaborationen leicht erhöhen.

Inhaltsverzeichnis

1. Einführung

In den letzten Jahren haben vortrainierte neuronale Sprachmodelle beeindruckende Resultate in verschiedenen Bereichen der Textverarbeitung erzielt. Sie können auf natürlichsprachliche Anweisungen reagieren und dynamisch zwischen Aufgaben wie Übersetzungen, Zusammenfassungen und arithmetischen Operationen wechseln. Außerdem sind sie in der Lage, hochqualitative freie Texte zu generieren, die für Menschen kaum von menschengeschriebenen Texten unterscheidbar sind [30][5]. Mit Aufgaben, in denen Schlussfolgerungsfähigkeiten gefordert sind, haben vortrainierte Sprachmodelle jedoch größere Schwierigkeiten [4]. In vielen Fällen scheitern sie schon daran, logische Relationen wie Negationen und Vergleiche richtig zu interpretieren [33][20]. Ihre Fähigkeit, vorhandenes Wissen wiederzugeben, hängt außerdem stark von der Formulierung der Eingabe ab. Sie sind anfällig dafür, sich durch gezielte falsche Hinweise wie „`Talk? Birds can...`“ dazu verleiten zu lassen, unpassende Vervollständigungen (beispielsweise „`talk`“ anstatt „`fly`“) zu generieren [20].

Menschen lösen komplexe Schlussfolgerungsprobleme häufig mithilfe aufgabenübergreifender Problemlösestrategien. Sie denken beispielsweise laut über das Problem nach, formulieren ihre Wissenslücken, erstellen Diagramme oder probieren verschiedene Antwortmöglichkeiten aus. Solche meta-kognitiven Strategien werden auch im Schulunterricht eingesetzt. Eine Mathematiklehrerin würde ihren Schüler beispielsweise fragen: „Was wissen wir über x?“ oder „Was können wir aus den gegebenen Voraussetzungen ableiten?“. Das Erlernen meta-kognitiver Strategien fördert nachweislich die akademische Leistungsfähigkeit von Schülern [10][11]. Es liegt daher nahe, zu untersuchen, ob Abwandlungen dieser etablierten Strategien im maschinellen Lernen eingesetzt werden können.

Diesen Ansatz verfolgt unter Anderem die Arbeit von Betz et al. [3]: Die Autoren benutzen die meta-kognitive Strategie, „laut nachzudenken“, um die Schlussfolgerungsfähigkeiten eines vortrainierten neuronalen Sprachmodells zu verbessern. Sie verwenden dafür einen Datensatz mit deduktiven Schlussfolgerungsproblemen. Die Aufgaben bestehen aus einem Fakt und einer Menge an Regeln, welche die Ableitung einer korrekten Schlussfolgerung aus dem Fakt ermöglichen. Das Sprachmodell generiert auf Grundlage der Aufgabenstellung zunächst dynamisch eine Problemelaboration, die zum Kontext hinzugefügt wird. Anschließend verwendet es den dynamisch erweiterten Kontext als Grundlage für seine Antwortvorhersage. Betz et al. stellen fest, dass das Sprachmodell eine einfache Heuristik nutzt, um die Antworten vorherzusagen. Der Datensatz ist daher nur eingeschränkt dazu geeignet, die Schlussfolgerungsfähigkeit des neuronalen Sprachmodells zu messen. Dynamische Kontexterweiterungen verbessern die Akkuratheit der Vorhersage um bis zu 9%. Wie die Autoren nachweisen, korrelieren positive Effekte der

Kontexterweiterungen damit, dass die entsprechenden Kontexterweiterungen dem Modell ermöglichen, seine Heuristik erfolgreicher einzusetzen. Die Resultate von Betz et al. werfen unter Anderem die folgenden Forschungsfragen auf: Ist das vortrainierte Sprachmodell in der Lage, mehrschrittige, deduktive Schlussfolgerungen zu vollziehen, wenn es keine einfache Heuristik ausnutzen kann? Führen dynamische Kontexterweiterungen in diesem Fall zu einer Verbesserung der Schlussfolgerungsfähigkeiten?

Um diese Fragen zu beantworten, generiere ich einen Datensatz mit komplexen, deduktiven Schlussfolgerungsproblemen (siehe Kapitel 5). Mithilfe dieses Datensatzes evaluiere ich die Fähigkeiten, die das neuronale Sprachmodell im unüberwachten Vortraining erlernt hat (Kapitel 6). Neben der Fähigkeit, deduktive Ableitungen zu vollziehen, überprüfe ich mit einem zweiten Datensatz, ob das Sprachmodell semantisches Wissen nutzt, um seine Vorhersagen zu treffen. Die Resultate gehören zum Themenkomplex „Explainable AI“ und dienen einem besseren Verständnis vortrainierter neuronaler Sprachmodelle und ihrer Einschränkungen.

Darüber hinaus untersuche ich, welche Anteile die einzelnen Komponenten der metakognitiven Strategie an den beobachteten Effekten haben. Um seine Vorhersage mithilfe dynamischer Kontexterweiterungen zu verbessern, muss ein Sprachmodell zwei voneinander unabhängige Schritte vollziehen können:

Zum Einen muss es in der Lage sein, den zusätzlichen Kontext auszunutzen, um seine Vorhersage zu verbessern. Um diese Fähigkeit zu untersuchen, betrachte ich in Kapitel 7 den Effekt unterschiedlicher Arten vorgegebener Problemelaborationen auf die Performanz des Sprachmodells. Dabei gehe ich unter Anderem auf die folgenden Fragen ein: Wie gut kann das Sprachmodell unterschiedliche Arten hochqualitativer Problemelaborationen für seine Vorhersagen ausnutzen? Welchen Einfluss hat das Vorkommen der richtigen Antwort in den Elaborationen auf die Vorhersage? Gibt es Hinweise darauf, dass das Modell die Elaborationen ausnutzt, um eine einfache Heuristik anzuwenden? Welche Rückschlüsse auf die Schlussfolgerungsfähigkeiten des Sprachmodells lassen sich aus den Beobachtungen ziehen?

Zum Anderen benötigt das Sprachmodell die Fähigkeit, auf Grundlage einer Aufgabenstellung selbst eine geeignete Problemelaboration zu generieren. In Kapitel 8 untersuche ich unterschiedliche Methoden, dynamische Kontexterweiterungen zu erzeugen. Neben drei Methoden zur Erstellung von Problemelaborationen im Zero-Shot-Setting (Abschnitt 8.1) betrachte ich auch Elaborationen, die mithilfe von Few-Shot-Learning generiert werden (Abschnitt 8.3). Schafft das Sprachmodell beim Few-Shot-Learning, die Struktur der gesehenen Beispiele korrekt zu generalisieren?

Eine Zusammenfassung der Ergebnisse findet sich in Kapitel 9. In diesem Kapitel gehe ich auch genauer auf Einschränkungen des Ansatzes ein und beleuchte Fragen für zukünftige Forschungsarbeiten.

2. Grundlagen

In diesem Kapitel führe ich zunächst einige grundlegende Begriffe im Zusammenhang mit neuronalen Sprachmodellen ein und gehe anschließend auf Textgenerierungsstrategien und Training ein.

2.1. Neuronale Sprachmodelle

Ein neuronales Sprachmodell (*Neural Language Model, NLM*) ist ein probabilistischer Klassifikator, der in Abhängigkeit von einem gegebenen Kontext eine Wahrscheinlichkeitsverteilung über sein Vokabular erstellt. Das Vokabular eines Sprachmodells kann aus Token verschiedener Art, etwa Wörtern, Wortbestandteilen und Phrasen, bestehen. Intern werden diese Token üblicherweise durch dichtbesetzte Fließkommazahlvektoren, sogenannte Einbettungen, repräsentiert. Man unterscheidet dabei zwischen statischen und dynamischen (auch: kontextualisierten) Einbettungen [43][28].

Statische Einbettungen zeichnen sich durch eine kontextunabhängige, bijektive Zuordnung zwischen Token und Einbettungsvektoren aus. Eine hohe Wahrscheinlichkeit, dass Token in ähnlichen Kontexten verwendet werden, soll dabei idealerweise durch eine hohe Ähnlichkeit ihrer Einbettungsvektoren abgebildet werden. Als Maß für den geometrischen Abstand zwischen zwei Vektoren dient beispielsweise ihre Cosinus-Ähnlichkeit (*cosine similarity*). Im Idealfall spiegeln sich semantische Beziehungen zwischen den Token in mathematischen Beziehungen zwischen den Einbettungsvektoren wider. Seien beispielsweise v_{Paris}, v_{France}, $v_{Germany}$ und v_{Berlin} die Einbettungen der Worte „Paris“, „France“, „Germany“ und „Berlin“. Die Beziehung zwischen „Paris“ und „France“ ähnelt der zwischen „Berlin“ und „Germany“. Deshalb soll $v_{Paris} - v_{France} + v_{Germany}$ einen Vektor ergeben, der einen möglichst geringen Abstand zu v_{Berlin} hat [31]. Transformer-Modelle wie GPT [29] erlernen im unüberwachten Vortraining unter Anderem eine Einbettungsmatrix, die statische Einbettungsvektoren für alle Token im Vokabular des Sprachmodells enthält [41][29].

Das Vokabular des unidirektionalen Sprachmodells GPT-2, das ich in dieser Arbeit verwende, wird beispielsweise mit einem Byte-Pair-Encoding-Algorithmus (BPE) [36] auf Byte-Ebene (BBPE) [42] erzeugt. Würde man jedes Unicode-Zeichen durch ein eigenes Token im Vokabular repräsentieren, bräuchte man ein Basisvokabular von über 138.000 Tokens. Um das zu vermeiden, verwendet GPT-2 stattdessen die UTF-8-Kodierung der Unicode-Zeichen. Da jeder Text durch UTF-8 als Sequenz von Bytes dargestellt werden kann, ist ein Basisvokabular von nur 256 Tokens nötig, um jede beliebige Zeichensequenz zu verarbeiten. Ausgehend von diesem Basisvokabular lernt GPT-2 mithilfe eines

BPE-Algorithmus 50.000 zusätzliche Worte und Wortbestandteile, die häufig in den Trainingsdaten vorkommen. Mit einem weiteren speziellen Token, das das Ende einer Sequenz markiert („<|endoftext|>“), ergibt sich so ein Vokabular von 50.257 Tokens [30].

Im Gegensatz zu statischen Einbettungen enthalten dynamische Einbettungen Informationen über den Kontext, in dem sich ein Token befindet. In unterschiedlichen Kontexten wird das gleiche Token durch unterschiedliche dynamische Einbettungsvektoren repräsentiert, die seinen jeweiligen Zusammenhang zu den anderen Token in diesem Kontext abbilden. Dadurch können auch kontextabhängige Wortbedeutungen und Mehrdeutigkeiten abgebildet werden [28]. Transformer-Modelle wie GPT [29] und BERT [9] ermitteln unter Nutzung der statischen Einbettungen aufgabenspezifisch dynamische Einbettungen der Input-Tokens. Wie Rogers et al. zeigen, enthalten von BERT berechnete dynamische Einbettungen syntaktisches, semantisches und konzeptuelles Wissen des Modells [34].

Unidirektionale neuronale Sprachmodelle werden dazu trainiert, abhängig von einer gegebenen Wortsequenz $s_1, ..., s_{n-1}$ eine Wahrscheinlichkeitsverteilung $\mathcal{P}(x_i|s_1, ..., s_{n-1})$ über alle Token x_i im Vokabular zu berechnen, um das nächste Wort s_n in der Sequenz vorherzusagen [28]. Sie können nur den Kontext vor (oder nach) dem gesuchten Wort für ihre Berechnung nutzen. Aktuelle Beispiele für solche Modelle sind GPT [29], GPT-2 [30], GPT-3 [5] und Transformer-XL [8].

Bidirektionale neuronale Sprachmodelle wie BERT [9] und RoBERTa [23] sind dagegen in der Lage, maskierte Worte an einer beliebigen Stelle in der Textsequenz vorherzusagen. Sie verarbeiten sowohl den Kontext, der dem gesuchten Wort vorhergeht, als auch den Kontext, der darauf folgt.

Die Wahrscheinlichkeitsverteilung, die ein Sprachmodell auf Grundlage des gegebenen Kontexts berechnet, kann zur Textgenerierung genutzt werden. Ein Dekodierer selektiert dabei Token anhand ihrer bedingten Wahrscheinlichkeiten aus dem Vokabular und gibt diese aus. Generative, unidirektionale Sprachmodelle können Textsequenzen von variabler Länge erzeugen, indem sie ihre Ausgabe in jedem Zeitschritt zur Sequenz hinzufügen und im nächsten Zeitschritt als zusätzlichen Kontext für ihre Vorhersagen nutzen. Sie werden dann auch als *autoregressiv* bezeichnet [41].

Durch natürlichsprachliche Aufgabenbeschreibungen, so genannte *Prompts*, können neuronale Sprachmodelle für verschiedene Arten konditionaler Textgenerierungsaufgaben eingesetzt werden, beispielsweise Übersetzungen und das Beantworten von Wissensfragen (*Question Answering, QA*). Auch Multiple-Choice-Aufgaben können als Sprachmodellierungsproblem behandelt werden. Dazu werden die Wahrscheinlichkeitsvorhersagen, die das Sprachmodell für die natürlichsprachlichen Antwortmöglichkeiten berechnet, normalisiert und verglichen [5]. Dabei ist allerdings zu beachten, dass Sprachmodelle sensibel auf die genaue Formulierung der Prompts reagieren [26] [19] und anfällig dafür sind, sich durch „falsche Hinweise“ (*mispriming*) fehlleiten zu lassen [20].

2.2. Textgenerierung

Die Wahrscheinlichkeit, dass eine Wortsequenz $x_1, ..., x_n$ auf den Kontext Φ folgt, ergibt sich aus dem Produkt der bedingten Wahrscheinlichkeiten

$$\mathcal{P}(x_1, ..., x_n|\Phi) = \prod_{i=1}^{n} \mathcal{P}(x_i|\Phi, x_1, ..., x_{i-1}).$$

Eine Sequenz mit maximaler Gesamtwahrscheinlichkeit auszuwählen, entspricht daher dem Problem, einen optimalen Pfad durch einen exponentiell in der Größe des Vokabulars wachsenden Baum zu finden. Da dies nicht effizient möglich ist, werden verschiedene Dekodier-Strategien eingesetzt, um eine hinreichend gute Approximation dieser Lösung zu finden [46].

Beim *greedy decoding* wird zu jedem Zeitschritt dasjenige Wort aus dem Vokabular ausgewählt, dem das Sprachmodell die höchste bedingte Wahrscheinlichkeit prognostiziert. Diese Strategie hat den Nachteil, dass sie zu Worten tendiert, die generell häufig in natürlicher Sprache vorkommen, beispielsweise Artikeln und Präpositionen. Außerdem werden potentielle Pfade mit einer hohen Gesamtwahrscheinlichkeit gegebenenfalls verworfen, weil sie einzelne Worte mit schlechten Wahrscheinlichkeitsvorhersagen enthalten [15].

Um dieses Problem zu umgehen, verfolgt *beam search* nicht nur einen, sondern β mögliche Pfade gleichzeitig. Der Parameter β wird dabei als *beam width* bezeichnet. In jeder Schicht des Suchbaums werden die β Worte ausgewählt, deren bisherige Pfade die höchsten Gesamtwahrscheinlichkeiten haben. Die bedingten Wahrscheinlichkeiten aller möglichen Sukzessoren für diese Worte bilden die nächste Schicht, aus der wiederum die β vielversprechendsten Worte selektiert und die restlichen verworfen werden. Im Vergleich zu *greedy decoding* findet *beam search* immer eine mindestens gleich gute Lösung, da es einen Pfad, den *greedy decoding* finden würde, nur verwirft, wenn dessen Gesamtwahrscheinlichkeit zu gering ist. Allerdings ist zu beachten, dass die Gesamtwahrscheinlichkeit eines Pfades sich aus den Produkten der Einzelwahrscheinlichkeiten ergibt, wodurch sie mit zunehmender Länge der Sequenz sinkt. Dadurch tendiert *beam search* bei Aufgaben, in denen die Länge des generierten Textes nicht festgelegt ist, zu möglichst kurzen Sequenzen [46] [15].

Greedy decoding und *beam search* sind mit dem Ziel konstruiert, die Wahrscheinlichkeit des gewählten Pfades zu maximieren. Diese Strategien eignen sich gut für Aufgaben, in denen vom Modell erwartet wird, einen spezifischen, strikt vom Input abhängigen Text zu generieren. Zu solchen Problemen zählen beispielsweise maschinelle Übersetzungen, Textzusammenfassungen, Transkriptionen und Bildbeschreibungen. Bei Aufgaben aus dem Bereich *open ended text generation* wie beispielsweise dem Erstellen längerer Texte oder Dialoge, die dem Modell viele Freiheiten lassen, erweisen sie sich allerdings als defizitär. Wie Holtzman et al. zeigen, zeichnen sich von Menschen geschriebene Texte dadurch aus, dass die Wahrscheinlichkeiten der einzelnen Worte innerhalb eines Textes stark variieren. Dadurch sind sie interessant, informativ und abwechslungsreich. Die maschinell erstellten Texte dagegen haben eine geringe Diversität und wirken vorhersehbar und

repetitiv. Außerdem passiert es häufig, dass Modelle, die eine wahrscheinlichkeitsmaximierende Dekodier-Strategie verfolgen, Schleifen von immer gleichen Sätzen erzeugen. Die Einführung von Zufall soll diesen Problemen entgegenwirken [15].

Beim *sampling* wählt der Dekodierer mithilfe einer Zufallsfunktion ein Wort aus dem Vokabular aus, um die Diversität der Ausgabe zu erhöhen. In der einfachsten Form entspricht die Wahrscheinlichkeit, dass der Dekodierer ein bestimmtes Wort selektiert, der bedingten Wahrscheinlichkeit, die das Sprachmodell für dieses Wort berechnet hat. Da der Dekodierer in einigen Fällen auch ungeeignete Kandidaten auswählt, wird der erzeugte Text zwar weniger repetitiv, aber auch inkohärenter. Beim *temperature sampling* wird die Wahrscheinlichkeitsverteilung vor dem *sampling* durch Variation des Temperatur-Parameters verzerrt, sodass Kandidaten mit schlechten Vorhersagen mit geringerer Wahrscheinlichkeit gewählt werden. Das hat einen gemischten Effekt: Der Dekodierer selektiert einerseits seltener unpassende Kandidaten, andererseits agiert er tendenziell *greedy*, wodurch die positiven Effekte des Zufalls verringert werden [15].

Diesem Problem soll durch *top-k-sampling* abgeholfen werden. Um die Selektion schlechter Kandidaten zu vermeiden und gleichzeitig die positiven Effekte des Zufalls beizubehalten, werden dabei nur die k Wörter mit den höchsten Wahrscheinlichkeiten in die Auswahl einbezogen. Durch eine Abbildung der ursprünglichen Wahrscheinlichkeitsfunktion auf diese k Wörter wird eine neue Wahrscheinlichkeitsverteilung erstellt, die anschließend zum *sampling* verwendet werden kann. Die Schärfe der Wahrscheinlichkeitsverteilungen kann innerhalb einer Aufgabe stark variieren, da zu manchen Zeitpunkten nur sehr wenige Wörter als nächstes Wort infrage kommen, wohingegen der Verlauf des Textes in anderen Situationen offen ist. Da der Parameter k aber statisch festgelegt ist, ist eine Abwägung notwendig: Wählt man k klein, so werden bei flachen Verteilungen gute Kandidaten verworfen und die Kreativität des Modells wird stark eingeschränkt. Wählt man k dagegen groß, fallen bei scharfen Verteilungen unpassende Wörter in die Auswahl und die Wahrscheinlichkeit, dass diese gewählt werden, wird durch die Redistribution der Wahrscheinlichkeitsverteilung erhöht [15].

Um die Menge an Wörtern, die zum *sampling* verwendet werden, dynamisch an die jeweiligen Wahrscheinlichkeitsverteilungen anpassen zu können, kann *top-p-sampling/ nucleus sampling* verwendet werden. Die Idee dabei ist, schlechte Kandidaten, die den Text inkohärent werden lassen, aus der Auswahl auszuschließen und gleichzeitig möglichst alle guten Kandidaten zu behalten. Um dies zu erreichen, wird die kleinste Menge an Wörtern aus dem Vokabular ausgewählt, deren kumulierte Wahrscheinlichkeiten einen festgelegten Anteil p übersteigen. Zum *sampling* wird wieder die auf diese Menge redistribuierte Wahrscheinlichkeitsverteilung verwendet. Dadurch zieht das Modell bei scharfen Verteilungen, also in Situationen, in denen das nächste Wort stark durch die bisherige Textsequenz festgelegt ist, nur die wenigen gute Kandidaten in Betracht. Bei flachen Verteilungen, wenn also der Fortgang des Textes durch den bisherigen Kontext wenig eingeschränkt ist, stehen dem Modell hingegen viele Wörter zur Auswahl [15].

2.3. Vortraining, Finetuning, Few-Shot-Learning

Neuronale Sprachmodelle werden üblicherweise auf großen Datenmengen vortrainiert, um allgemeine linguistische Merkmale der Sprache zu erlernen [28]. Ihre Architektur ist aufgabenagnostisch, sodass sie mithilfe von anwendungsspezifischem Finetuning für verschiedene Aufgaben im NLP eingesetzt werden können [29]. Das Vortraining findet unüberwacht statt – das Sprachmodell verarbeitet die Trainingsdaten eigenständig, um daraus konzeptuelles, syntaktisches und semantisches Wissen zu erlangen. Um zu analysieren, was die Sprachmodelle im Vortraining lernen, kann man beispielsweise ihre Performanz in Aufgaben testen, die spezielle Fähigkeiten erfordern, etwa Wissensfragen oder Schlussfolgerungsaufgaben [40][34][1][13][12]. Im Gegensatz dazu findet beim Finetuning überwachtes Lernen statt. Die Trainingsdaten werden dabei mit „Etiketten" versehen, die den Eingaben gewünschte Ausgaben zuordnen. Anhand dieser Paare von Ein- und Ausgaben erlernt das Sprachmodell, wie es den neuen Aufgabentyp lösen muss. Dazu werden überwachte Datensätze mit typischerweise tausenden bis hunderttausenden Instanzen benötigt. Die Parameter, die das Sprachmodell im unüberwachten Vortraining erlernt hat, werden durch Finetuning angepasst. Bei großen Modellen mit vielen Parametern besteht die Gefahr, dass irrelevante Merkmale aus dem Datensatz erlernt werden und keine gute Generalisierbarkeit der erworbenen Fähigkeiten besteht [25] [21].

Brown et al. schlagen die Methoden *Zero-Shot-Learning*, *One-Shot-Learning* und *Few-Shot-Learning* als Alternative zum Finetuning vor [5]. Beim Zero-Shot-Learning bekommt das vortrainierte Modell eine natürlichsprachliche Aufgabenbeschreibung gegeben und generiert auf Grundlage dieser Beschreibung eine Antwort. Beim One-Shot-Learning bzw. Few-Shot-Learning soll das Sprachmodell durch Beispiele konditioniert werden – analog zu Menschen, welche einen neuen Aufgabentyp häufig schon nach Betrachtung weniger Musterlösungen beherrschen. Das Modell bekommt als Eingabe eine oder mehrere Demonstrationen von Kontexten und deren gewünschten Vervollständigungen, sowie einen finalen Kontext gegeben, den es anschließend vervollständigen soll. In allen drei Fällen werden keine Anpassungen der Gewichte vorgenommen. Das Modell lernt ausschließlich durch *forward propagation* zur Inferenzzeit, sogenanntes *priming*. Brown et al. berichten beeindruckende Resultate in einer großen Bandbreite an NLP-Aufgaben mit dem autoregressiven Sprachmodell GPT-3. Die Verbesserungen, die durch Few-Shot-Learning im Vergleich zu Zero-Shot-Learning erzielt werden, variieren dabei abhängig von Modellgröße und Aufgabe stark. Auf dem NLI-Datensatz RTE, der im populären Benchmark SuperGLUE [42] enthalten ist, erreicht das größte Modell (175 Milliarden Parameter) beispielsweise eine Zero-Shot-Korrektklassifizierungsrate von 63,5 %, die durch Few-Shot-Learning auf 72,9 % gesteigert wird. Bei kleineren Modellen (bis zu 13 Milliarden Parameter) liegt die Korrektklassifizierungsrate auf dieser Aufgabe hingegen nicht signifikant über der eines Zufallsklassifikators. Auch durch Few-Shot-Learning ist in diesem Fall keine klare Verbesserung zu erkennen [5].

3. Verwandte Arbeiten

In diesem Kapitel fasse ich zunächst einige Resultate aus dem Bereich *Natural Language Inference* zusammen. Anschließend stelle ich vier Arbeiten vor, die mit moderatem Erfolg versuchen, die Performanz vortrainierter Sprachmodelle zu verbessern, indem sie Beobachtungen über menschliche Problemlösestrategien auf diese übertragen. Zu diesen Arbeiten gehört insbesondere die Studie von Betz et al. [3], auf deren Resultate meine Arbeit aufbaut.

3.1. Schlussfolgerungsfähigkeiten vortrainierter Sprachmodelle

Aufgaben aus dem Bereich *Natural Language Inference (NLI)* betreffen die Fähigkeit, Relationen zwischen Sätzen zu verstehen. Sie sind typischerweise als Klassifikationsaufgaben konstruiert, in denen entschieden werden soll, ob ein Satz (Hypothese) aus einem anderen (Prämisse) folgt, diesem widerspricht oder neutral zu ihm steht. Vortrainierte Sprachmodelle zeigen im Allgemeinen eine mangelhafte Zero-Shot-Performanz bei NLI-Aufgaben [3].

Wie Kassner und Schütze zeigen, haben vortrainierte Sprachmodelle große Schwierigkeiten, zwischen negierten („`Birds can [MASK]`“) und nicht-negierten Sätzen („`Birds cannot [MASK]`“) zu unterscheiden [20]. Die Autor:innen erweitern die von Petroni et al. entwickelte LAMA-Aufgabe, die dazu dient, im Vortraining erlerntes Faktenwissen bidirektionaler Sprachmodelle abzufragen [27]. Wissensfragen werden dabei zu Lückentexten umformuliert, die das Modell vervollständigen soll. Um beispielsweise die Frage „`Where was Dante born?`“ zu beantworten, bekommt das Modell den Kontext „`Dante was born in [MASK]`“ gegeben. Die Leistungsfähigkeit der Sprachmodelle hängt sensibel von der Formulierung der Eingabe ab [26][19], sodass mit dieser Methode nur eine untere Schranke für das Faktenwissen des Modells gemessen werden kann. Kassner und Schütze erweitern den Datensatz um Aussagen, denen fehlleitende Informationen, sogenannte *misprimes*, vorangestellt werden, beispielsweise „`Rome? Dante was born in [MASK]`“. Die untersuchten Sprachmodelle sind anfällig dafür, sich von *misprimes* verwirren zu lassen. Kassner und Schütze ergänzen den Datensatz außerdem mit negierten Varianten aller Aussagen, etwa „`Dante was *not* born in [MASK]`“. Dabei zeigt sich, dass das Sprachmodell seine Vorhersagen größtenteils nach Kookkurenz des Subjekts („`bird`“) und der Vervollständigung („`fly`“) ausrichtet und Negationen ignoriert. Durch Finetuning mit einem

synthetischen überwachten Datensatz ist das verwendete Sprachmodell problemlos in der Lage, die Bedeutung von negierten Aussagen zu erlernen und korrekt zu generalisieren.

Richardson et al. beschreiben ebenfalls die Notwendigkeit, bestehende NLI-Datensätze zu erweitern. Sie demonstrieren, dass vortrainierte Sprachmodelle nur schlecht dazu in der Lage sind, Schlussregeln anzuwenden, die logische Relationen wie Negationen und Vergleiche beinhalten. Selbst Sprachmodelle mit Finetuning auf NLI-Datensätzen scheitern an dieser Aufgabe, was auf eine schlechte Generalisierbarkeit der erlernten Fähigkeiten hinweist. Beispielsweise erzielt BERT nach Finetuning auf den Datensätzen SNLI und MNLI [44] eine Korrektklassifizierungsrate von nur 47,3 % beim Test mit logischen Fragmenten. Durch zusätzliches aufgabenspezifisches Training des Modells erreichen Richardson et al. eine Korrektklassifizierungsrate von 98,0 % auf diesem Datensatz [32].

Auch bei komplexen Schlussfolgerungsproblemen erzielen vortrainierte Sprachmodelle nach entsprechendem Finetuning gute Ergebnisse [3]. Clark et al. stellen RuleTaker vor, ein Sprachmodell, das mithilfe eines Klassifikator-Moduls vorhersagt, ob eine Aussage aus einer Menge an explizit gegebenen, natürlichsprachlichen Fakten und Regeln deduktiv abgeleitet werden kann [7]. Der zu diesem Zweck generierte Datensatz enthält Problemstellungen, die neben den erforderlichen Fakten und generalisierten konditionalen Regeln auch irrelevante Aussagen enthalten. Außerdem sind gegebenenfalls mehrere Schlussfolgerungen nötig, um zur Konklusion zu gelangen. RuleTaker beherrscht diese Aufgabe nahezu perfekt und ist sogar in der Lage zu identifizieren, welche Aussagen für die Ableitung der Konklusion gebraucht werden. Die erlernten Fähigkeiten sind zudem stabil gegenüber manuellen Umformulierungen der Problemstellungen. Betz et al. gehen noch einen Schritt weiter und trainieren das unidirektionale Sprachmodell GPT-2 mit einem synthetisch erstellten Argumentkorpus, der natürlichsprachliche Instanzen deduktiver Argumentationsschemata enthält. Wie die Autoren zeigen, ist das Modell nach Finetuning auf nur drei Kern-Schemata in der Lage, die erlernten Schemata korrekt zu generalisieren und sogar zu komplexeren Argumentationsmustern zu verknüpfen [4]. Wie Talmor et al. nachweisen, sind Sprachmodelle außerdem in der Lage, explizites Wissen mit implizitem, im Training erlangtem Wissen zu kombinieren, um sogenannte enthymemische Schlüsse zu vollziehen [39].

3.2. Meta-kognitive Strategien im Natural Language Processing

Meta-Kognition ist ein Konzept aus der kognitiven Psychologie, das die Tätigkeit des Nachdenkens über das eigene Denken bezeichnet. Meta-kognitive Strategien sind Verfahren, die dazu dienen, aufgabenübergreifende Problemlösefähigkeiten zu fördern. Dazu gehören Lernstrategien wie beispielsweise den kognitiven Prozess zu verbalisieren („Think Aloud“), über die Problemstellung zu reflektieren, Wissenslücken zu identifizieren und zu schließen, Zwischenschritte auszuformulieren, Schaubilder zu erstellen und verschiedene Antwortmöglichkeiten zu überprüfen [11]. Meta-kognitive Strategien zu erlernen, fördert

nachweislich die akademische Leistungsfähigkeit von Schüler:innen [11]. Es liegt daher nahe, diese Strategien aufs maschinelle Lernen zu übertragen. Im Folgenden werden vier Studien vorgestellt, die diesen Ansatz verfolgen.

Um komplexe Schlussfolgerungsprobleme zu lösen, kann es hilfreich sein, sich die Zusammenhänge mithilfe eines Graphen zu veranschaulichen. Saha et al. setzen diese meta-kognitive Strategie bei dem von Clark et al. vorgestellten deduktiven Schlussfolgerungsproblem (siehe Abschnitt 3.1) ein [35]. Sie erweitern RuleTaker dafür um zwei weitere neuronale Klassifikatoren. Das zugrundeliegende Sprachmodell berechnet kontextualisierte Einbettungen der Eingabe-Tokens (siehe Abschnitt 2.1), die anschließend von den Klassifikator-Modulen für ihre Vorhersage verwendet werden. Neben dem Modul, das die Antwortvorhersage erstellt, hat PRover zwei Klassifikator-Module, um die Knoten und Kanten eines Beweisgraphen vorherzusagen. Das erweiterte Modell wird beim Training nicht nur darauf optimiert, richtige Antworten zu geben, sondern auch korrekte Beweisketten zu erstellen. PRover demonstriert mit einer Korrektklassifizierungsrate von durchschnittlich 99,3 % eine vergleichbar gute Performanz wie RuleTaker. Es konstruiert zudem in durchschnittlich 87,3 % der Aufgaben korrekte Beweisgraphen. Die richtige Antwort vorherzusagen, stellt für das Modell also eine leichtere Aufgabe dar als korrekte Beweisgraphen zu generieren. Um zu überprüfen, wie gut PRover die erlernten Fähigkeiten verallgemeinern kann, testen die Autoren das Modell im Zero-Shot-Setting auf einem Datensatz mit anders formulierten Schlussfolgerungsproblemen. Hierbei zeigt sich, dass die Konstruktion von Beweisketten auch die Schlussfolgerungsfähigkeiten des Modells verbessert: PRover zeigt insbesondere bei schwierigen Aufgaben eine um bis zu 15 % verbesserte Korrektklassifizierungsrate gegenüber RuleTaker. Dieses Ergebnis weist darauf hin, dass die meta-kognitive Strategie, Behauptungen mithilfe von Graphen zu beweisen, auch bei neuronalen Sprachmodellen dazu geeignet ist, aufgabenübergreifende Fähigkeiten zu verbessern.

Eine andere Herangehensweise an solche Probleme ist, sich die Zwischenschritte nacheinander aufzuschreiben. Diese Idee verfolgen Tafjord et al.: Sie trainieren ein Sprachmodell darauf, durch einschrittige Ableitungen aus der Problembeschreibung einen natürlichsprachlichen Beweis zu generieren [38]. Im Unterschied zu Saha et al. brauchen sie dafür keine zusätzlichen neuronalen Klassifikatoren. Dadurch diese Konstruktion stellen sie sicher, dass das Sprachmodell den Beweis bei seiner Antwortvorhersage berücksichtigt. Sie trainieren ihr Modell ProofWriter zusätzlich darauf, alle Aussagen aufzulisten, die aus einem gegebenen Kontext logisch folgen, sowie eine einzelne Aussage zu identifizieren, die dem Kontext hinzugefügt werden müsste, um die Antwort beweisen zu können. Tafjord et al. zeigen, dass ProofWriter auf dem gleichen Datensatz 9% mehr korrekte Beweise generiert als PRover und die erlernten Fähigkeiten zudem besser auf einen anderen Datensatz übertragen kann. Bei Problemen, die mehr Beweisschritte erfordern als die Probleme im Trainingsdatensatz, funktioniert ein Modell, das iterativ einschrittige Beweise erzeugt, besser als ein Modell, das alle Ableitungsschritte auf einmal generiert.

Wenn man versucht, die richtige Antwort auf eine Wissensfrage zu finden, kann es nützlich sein, sich zunächst die eigenen Wissenslücken bewusst zu machen und diese zu schließen. Shwartz et al. zeigen, dass eine Adaption dieser meta-kognitiven Strategie

neuronalen Sprachmodellen helfen kann, Common-Sense-Fragen besser zu beantworten [37]. Sie geben dem vortrainierten Sprachmodell die Möglichkeit, zunächst aufgabenspezifische, stark strukturierte Kontexterweiterungen zu generieren, bevor es sich für eine Antwortoption entscheidet. Indem das Modell sich selbst klärende Fragen stellt und diese beantwortet, soll es aufgabenrelevantes implizites Hintergrundwissen explizieren und somit Wissenslücken schließen. Das Sprachmodell bekommt zunächst den Kontext und einen Frage-Präfix, beispielsweise „`What's the definition of`“, als Prompt gegeben und vervollständigt diesen mit *nucleus sampling* zu fünf klärenden Fragen. Jede dieser Fragen wird anschließend an den Kontext angefügt und mit dem zugehörigen Antwort-Präfix (etwa „`The definition of`“) konkateniert, um jeweils zehn sogenannte *clarifications* (Klarstellungen) zu generieren. Ein (möglicherweise anderes) Sprachmodell sagt für jede mögliche Konkatenation aus ursprünglichem Kontext, Klarstellung und Antwortmöglichkeit eine Gesamtwahrscheinlichkeit vorher. Aus der Kontexterweiterung mit der höchsten Gesamtwahrscheinlichkeit ergeben sich die vorhergesagte Lösung sowie die dafür verwendete Klarstellung. Shwartz et al. erreichen auf vier von sechs Benchmarks aus dem Bereich *common sense QA* eine deutliche Verbesserung der Zero-Shot-Performanz durch solche strukturierten Kontexterweiterungen. Sie zeigen durch manuelle Analyse, dass 60 % der Klarstellungen, welche die Vorhersage verbessern und 12 % der Klarstellungen, welche die Vorhersage verschlechtern, von Menschen als faktisch korrekt bewertet werden.

Eine Kombination beider Fähigkeiten ist zur Lösung enthymemischer Schlussfolgerungsprobleme gefordert: Das Modell muss in der Lage sein, ein Schlussfolgerungsproblem zu lösen und dafür zusätzlich auf implizites Wissen zurückgreifen. Gontier et al. untersuchen anhand solcher Probleme die Fähigkeit von Sprachmodellen, Beweisstrategien zu erlernen und systematisch anzuwenden [14]. Sie trainieren Sprachmodelle mit einem Korpus, der auf CLUTTR beruht, einem Datensatz mit relationalen Inferenzproblemen, die nur unter Verwendung impliziter genereller Prämissen gelöst werden können. Jedes Beispiel enthält neben den Fakten, die in Form einer Geschichte gegeben sind, und der Antwort auch einen Beweis für die Lösung. Gontier et al. trainieren Sprachmodelle darauf, zunächst eine Kontexterweiterung in Form eines natürlichsprachlichen Beweises und anschließend eine Lösung für das enthymemische Schlussfolgerungsproblem zu generieren. Die Autoren unterscheiden zwischen zwei Arten von Beweisen, jeweils in unterschiedlichen Längen: Bei rückwärtsgerichteten Beweisen wird die Antwort erklärt, indem die Regeln von der Konklusion ausgehend rückwärts zu den Prämissen hin verkettet werden. Vorwärtsgerichtete Beweise entsprechen dagegen eher dem Ausformulieren von Zwischenschlüssen. Dabei werden die Prämissen verknüpft, um neue Fakten abzuleiten und so iterativ zur Konklusion zu gelangen. Die Autoren zeigen, dass Sprachmodelle mit gewissen Einschränkungen lernen können, valide Beweise zu generieren. Vorwärtsgerichtete Beweise sind dabei für die Modelle zwar einfacher zu generieren, aber schwieriger zu verstehen als rückwärtsgerichtete Beweise. Ebenso wie bei Saha et al. wird auch hier deutlich, dass das Generieren der Antwort ein einfacheres Problem darstellt als das Erstellen eines Beweises und einer anschließenden Antwort: Die Akkuratheit der Vorhersage bleibt unter der von Modellen zurück, die darauf trainiert wurden, keine Kontexterweiterung, sondern direkt eine Antwort zu generieren.

Betz et al. nutzen die Textgenerierungsfähigkeiten vortrainierter Sprachmodelle aus, um dynamische Problemelaborationen für deduktive Schlussfolgerungsprobleme zu generieren [3]. Sie beziehen sich dabei auf die Beobachtung, dass Menschen komplexe Schlussfolgerungsprobleme mithilfe von Problemelaborationen lösen: Sie durchdenken die Fragestellung, formulieren sie gegebenenfalls um, ziehen Zwischenschlüsse und entwickeln und überprüfen verschiedene Lösungsmöglichkeiten, bevor sie eine Antwort geben. Im Unterschied zu Saha et al., Gontier et al. und Tafjord et al. trainieren Betz et al. das verwendete Sprachmodell GPT-2 aber nicht aufgabenspezifisch. Stattdessen untersuchen sie den Einfluss selbst generierter Problemelaboration auf die Zero-Shot-Performanz des vortrainierten Modells.

Dazu generieren Betz et al. *ChainRuler*, einen Datensatz mit deduktiven Schlussfolgerungsproblemen (siehe auch 5.1). Eine Problembeschreibung besteht aus einem Fakt und einer Menge an allgemeinen Regeln. Manche der Regeln können genutzt werden, um die Konklusion aus dem Fakt abzuleiten, andere dienen zur Ablenkung. In manchen Fällen enthält die Regelkette eine Kontraposition an letzter Stelle. Das vortrainierte Sprachmodell sagt auf Grundlage der Problembeschreibung Wahrscheinlichkeiten für die richtige Lösung und zwei falsche Alternativen vorher. Betz et al. überprüfen, wie die Vorhersage sich verändert, wenn das Modell die Möglichkeit bekommt, zuvor eine aufgabenspezifische Problemelaboration zu generieren, die zum Kontext hinzugefügt wird. Sie zeigen, dass das Sprachmodell eine einfache Heuristik anwendet, um die ChainRuler-Aufgaben zu lösen. Diese Heuristik funktioniert bei einfachen Aufgaben gut, führt bei Aufgaben mit höherer Kompexität oder Kontraposition aber zu systematischem Versagen. Dynamisch generierte Kontexterweiterungen können die Korrektklassifizierungsrate um bis zu 9 % verbessern.

4. Vorgehen

In der vorliegenden Bachelorarbeit untersuche ich aufbauend auf den Ergebnissen von Betz et al. (i) die Schlussfolgerungsfähigkeiten eines vortrainierten neuronalen Sprachmodells im Zero-Shot-Setting, (ii) den Einfluss verschiedener hochqualitativer Kontexterweiterungen auf die Schlussfolgerungsfähigkeiten, sowie (iii) die Fähigkeiten des Sprachmodells, selbst aufgabenspezifische Kontexterweiterungen zu generieren und zu nutzen [3].

Um die Schlussfolgerungsfähigkeiten des Sprachmodells GPT-2 zu messen, verwenden Betz et al. *ChainRuler*, einen synthetisch generierten Datensatz mit deduktiven Schlussfolgerungsproblemen (siehe dazu auch Abschnitt 5.1). Eine Problembeschreibung aus diesem Datensatz lautet beispielsweise:

> „`If someone is empty, then they are innocent. If someone is green, then they are loud. If someone is loud, then they are guilty. Jill is green.`“ [3, Tabelle 2]

Das neuronale Sprachmodell soll ausgehend vom Fakt „`Jill is green.`“ über die Regelkette „`If someone is green, then they are loud. If someone is loud, then they are guilty.`“ durch mehrmaliges deduktives Ableiten zu der gewünschten Konklusion „`Jill is guilty.`“ gelangen. Die Vorhersagen, die das Sprachmodells für die richtige und die beiden falschen Antwortmöglichkeiten, „`Jill is not guilty.`“ und „`Jill is innocent.`“, berechnet, geben einen Hinweis darauf, ob es aus dem gegebenen Kontext deduktiv schlussfolgert. Die Aussage „`If someone is empty, then they are innocent.`“ dient dabei als ablenkende Regel (Distraktor). Die Anzahl an Regeln und Distraktoren wird systematisch variiert. Der *ChainRuler*-Datensatz ist so konstruiert, dass Distraktoren immer mit dem Prädikat der Konklusion (hier „`guilty`“), dessen logischem Komplement („`not guilty`“), seinem konzeptuellen Komplement („`innocent`“) oder dessen Negation („`not innocent`“) enden. Wie die Autoren feststellen, nutzt GPT-2 diese Eigenschaft des Datensatzes für seine Antwortvorhersagen aus (siehe dazu auch Abschnitt 5.3). Die Schwierigkeiten, die vortrainierte Sprachmodelle damit haben, negierte Aussagen zu interpretieren [20][12], führen ebenfalls zu Problemen (siehe dazu Abschnitt 5.2). Die Performanz von GPT-2 auf dem *ChainRuler*-Datensatz wird also durch Zufallsgrößen wie der Anzahl negierter Aussagen und der Häufigkeit des Lösungs-Prädikats in der Problembeschreibung beeinflusst. Mit diesem Datensatz lässt sich also nicht untersuchen, ob das Sprachmodell in der Lage wäre, deduktive Schlussfolgerungen aus komplexen Problembeschreibungen zu ziehen, wenn es nicht die Möglichkeit hätte, eine einfache Heuristik anzuwenden.

Zu diesem Zweck generiere ich einen balancierten Datensatz namens *ModifiedChainRuler*, der ebenfalls aus mehrschrittigen, deduktiven Schlussfolgerungsproblemen besteht (Abschnitt 5.4). Eine Aufgabenbeschreibung lautet beispielsweise:

> „`If someone is happy, then they are awake. If someone is young, then they are sad. Maria is charming. If someone is careless, then they are happy. If someone is charming, then they are careless. If someone is serious, then they are sleeping.`“

Analog zum *ChainRuler*-Datensatz enthält sie einen Fakt („`Maria is charming.`“) und eine Regelkette („`If someone is charming, then they are careless. If someone is careless, then they are happy. If someone is happy, then they are awake.`“), aus der die Lösung („`Maria is awake.`“) abgeleitet werden kann. Die Distraktoren („`If someone is young, then they are sad. If someone is serious, then they are sleeping.`“) enthalten – im Gegensatz zu Distraktoren im *ChainRuler*-Datensatz – genau einmal das konzeptuelle Komplement („`sleeping`“) des Lösungsprädikats. Dadurch wird der statistischen Effekt des Lösungsprädikats („`awake`“) in der Aufgabenbeschreibung ausgeglichen. Das macht es GPT-2 unmöglich, die Heuristik auszunutzen, die es beim *ChainRuler*-Datensatz anwendet.

Probleme aus dem *ModifiedChainRuler*-Datensatz enthalten außerdem keine negierten Aussagen. Die Antwortmöglichkeiten reduzieren sich damit auf die richtige (hier „`Maria is awake.`“) und eine falsche Alternative („`Maria is sleeping.`“). In Abschnitt 5.2 diskutiere ich die Konsequenzen dieser Entscheidung.

Des Weiteren stellt sich die Frage, wie die Bedeutungen der verwendeten Prädikate (wie „`careless`“, „`young`“ und „`sleeping`“) die Schlussfolgerungsfähigkeiten des Sprachmodells beeinflussen. Neuronale Sprachmodelle erlernen im Vortraining semantisches Wissen [34][28], das sie bei der Lösung von Schlussfolgerungsaufgaben einsetzen können [39]. Es ist denkbar, dass Regeln wie „`If someone is not happy, then they are not upset`“ [3, Bild 1] schlecht mit dem internen Wissen des Sprachmodells vereinbar sind und sich negativ auf seine Performanz auswirken. Regeln im *ModifiedChainRuler*-Datensatz enthalten daher nur Konzepte, die einander möglichst wenig widersprechen. Darüber hinaus untersuche ich, welchen Einfluss semantisches Wissen auf die Vorhersagen des Sprachmodells hat: Ist GPT-2 in der Lage, Schlussfolgerungen über ihm unbekannte Konzepte zu ziehen? Um diese Frage zu beantworten, erstelle ich einen weiteren balancierten Datensatz, bei dem alle Prädikate mit Fantasiewörtern ersetzt werden (Abschnitt 5.6). Eine Aufgabenstellung aus diesem Datensatz lautet beispielsweise:

> „`If someone is laphiful, then they are not canimery. Lisa is foaby. If someone is benoosy, then they are potaugh. If someone is foaby, then they are laphiful. If someone is pawounig, then they are canimery.`“

Dass die *ModifiedChainRuler*-Datensätze als Maß für die Schlussfolgerungsfähigkeiten neuronaler Sprachmodelle geeignet sind, zeige ich in Abschnitt 5.5. In Kapitel 6 evaluiere ich die Performanz von GPT-2 auf den beiden Datensätzen.

Betz et al. testen auf dem *ChainRuler*-Datensatz neben den Schlussfolgerungsfähigkeiten von GPT-2 vor Allem die meta-kognitive Strategie, „laut nachzudenken“. Das neuronale

Sprachmodell bekommt dabei die Möglichkeit, auf Grundlage der Aufgabenbeschreibung zunächst einen Text zu generieren, der an die Problembeschreibung angefügt wird. Anhand des erweiterten Kontexts berechnet es anschließend seine Antwortvorhersage. Betz et al. testen sechs verschiedene Arten, solche dynamischen Problemelaborationen zu generieren. Sie erzielen damit Verbesserungen von bis zu 9 Prozentpunkten in der Akkuratheit. Die Verbesserungen lassen sich allerdings darauf zurückführen, dass die Kontexterweiterungen dem Modell ermöglichen, seine Heuristik erfolgreicher anzuwenden: Je häufiger das Lösungsprädikat in einem erweiterten Kontext vorkommt, desto wahrscheinlicher sagt GPT-2 die richtige Antwort vorher. Aufbauend auf dieser Beobachtung untersuche ich, welchen Einfluss dynamische Kontexterweiterungen auf die Vorhersagen von GPT-2 haben, wenn es auf keine einfache Heuristik zurückgreifen kann.

Dynamische Problemelaborationen für die Antwortvorhersage einzusetzen, erfordert vom neuronalen Sprachmodell zwei Fähigkeiten: Erstens muss es in der Lage sein, auf Grundlage des gegebenen Kontexts eine geeignete Problemelaboration zu generieren. Zweitens muss es diese ausnutzen, um seine Vorhersage zu verbessern. Um den Einfluss beider Komponenten getrennt zu untersuchen, gehe ich folgendermaßen vor:

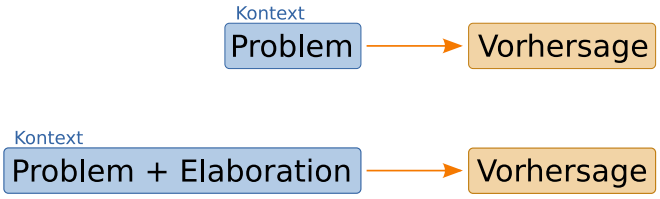


Abbildung 4.1.: Vorhersage mit und ohne vorgegebene Problemelaborationen

Zunächst untersuche ich, ob vorgegebene Problemelaborationen die Schlussfolgerungsfähigkeiten des Modells verbessern (Kapitel 7). Als Maß für die Schlussfolgerungsfähigkeit dient die Akkuratheit der Vorhersage auf dem *ModifiedChainRuler*-Datensatz. Zu allen Einträgen des Datensatzes stelle ich synthetische, hochqualitative Problemelaborationen her, die an die Problembeschreibung angefügt werden. In Abschnitt 7.1 überprüfe ich zunächst die Plausibilität des Ansatzes. Anschließend untersuche ich, wie die Performanz des Sprachmodells sich verändert, wenn es die Zwischenergebnisse zu den jeweiligen Aufgaben gegeben bekommt (Abschnitt 7.2). Da bei Problemen mit gegebenen Zwischenschlüssen nur noch ein Ableitungsschritt nötig ist, um zur richtigen Konklusion zu gelangen, vergleiche ich die Performanz auf diesen Problemen zusätzlich mit der Performanz auf Problemen der Tiefe 1, welche die gleiche Menge an Regeln enthalten (siehe Abschnitt 7.2.3 und Abschnitt 7.2.4).

Inspiriert von der Arbeit von Gontier et al. [14] betrachte ich dann den Effekt vorwärts- und rückwärtsgerichteter Elaborationen. Vorwärtsgerichtete Problemelaborationen gehen vom Fakt aus und verketten die Regeln und alle daraus ableitbaren Zwischenschlüsse

in der richtigen Reihenfolge, um zur Konklusion zu gelangen. Die vorwärtsgerichtete Elaboration zur oben genannten Aufgabe lautet beispielsweise:

> „Maria is charming. If someone is charming, then they are careless. Therefore, Maria is careless. If someone is careless, then they are happy. Therefore, Maria is happy. If someone is happy, then they are awake. Therefore, Maria is awake.“

Bei rückwärtsgerichteten Elaborationen wird die Konklusion durch Verkettung der Regeln und Zwischenschlüsse begründet:

> „Maria is awake. That's because if someone is happy, then they are awake. Maria is happy. That's because if someone is careless, then they are happy. Maria is careless. That's because if someone is charming, then they are careless. Maria is charming.“

Ich untersuche dabei nicht nur, *wie stark* vorgegebene Problemelaborationen die Schlussfolgerungsfähigkeiten des Sprachmodells beeinflussen, sondern auch, *wodurch* sie diesen Effekt haben.

Beide Arten von idealen Problemelaborationen enthalten die richtige Lösung – vorwärtsgerichtete Elaborationen an letzter und rückwärtsgerichtete Elaborationen an erster Stelle. Um zu überprüfen, ob das Modell die gegebenen Schlussfolgerungen für seine Antwortvorhersage nutzt oder nur die richtige Lösung memorisiert, werden beide Arten von Elaborationen auch in einer maskierten Form getestet. Die Konklusion „`Maria is awake.`“ wird darin durch „`Maria is (...).`“ ersetzt. Außerdem wird überprüft, welchen Anteil die Heuristik, die beim *ChainRuler*-Datensatz zum Einsatz kommt, am Effekt vorgegebener Elaborationen hat: Der erweiterte Kontext enthält sowohl in der Aufgabenbeschreibung als auch in der Elaboration die Regel „`If someone is happy, then they are awake.`“ aber nur einmal den ausgleichenden Distraktor „`If someone is serious, then they are sleeping.`“. Das Lösungsprädikat kommt daher zweimal im Kontext vor, sein konzeptuelles Komplement aber nur einmal. Um zu überprüfen, welchen Anteil die Heuristik am Nutzen der vorgegebenen Elaborationen hat, teste ich zusätzliche Varianten aller Elaborationen, die einen weiteren ausgleichenden Distraktor enthalten. Aus den Ergebnissen der Experimente in Kapitel 6 und Kapitel 7 leite ich Informationen darüber ab, (i) wie das Sprachmodell die *ModifiedChainRuler*-Aufgaben löst, (ii) welche Komponenten der Aufgaben ihm Schwierigkeiten bereiten und (iii) welche Kontexterweiterungen ihm helfen, diese Hürden zu überwinden.

In Kapitel 8 untersuche ich die Fähigkeiten des Sprachmodells, selbst Problemelaborationen zu generieren und für seine Vorhersage zu nutzen (siehe Schaubild 4.2). Zunächst evaluiere ich drei verschiedene Methoden, Kontexterweiterungen im Zero-Shot-Setting zu erstellen (Abschnitt 8.1). Dabei wird das Sprachmodell mit der Aufgabenstellung und einem natürlichsprachlichen Prompt konfrontiert, woraufhin es vier Sätze generiert. Die selbst erstellte Problemelaboration wird anschließend zum Kontext hinzugefügt und als zusätzliche Grundlage für die Vorhersage der Antwort verwendet. Eine Problemelaboration ist in dem Maße funktional gut, wie sie dem Sprachmodell hilft, seine Antwortvorhersa-

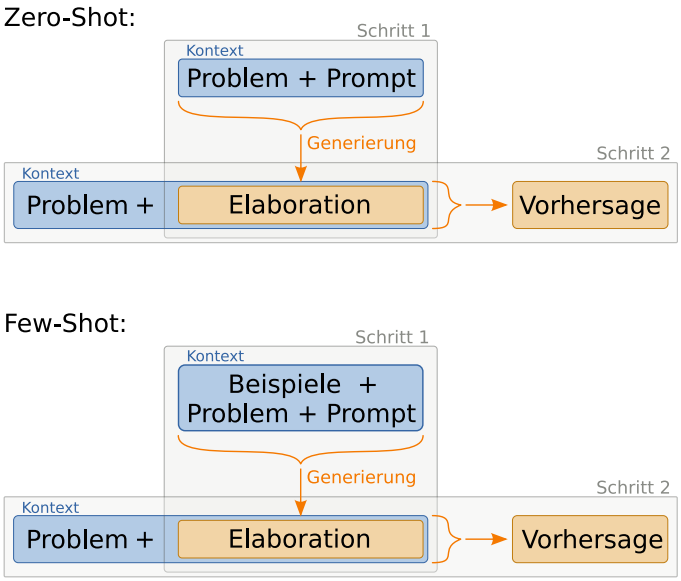


Abbildung 4.2.: Generieren und Nutzen von Problemelaborationen

ge zu verbessern. Ich messe die funktionale Güte der Problemelaborationen, indem ich die Akkuratheit der Vorhersagen auf dem *ModifiedChainRuler*-Datensatz mit und ohne Problemelaborationen vergleiche (siehe Abbildung 4.1).

In Abschnitt 8.3 untersuche ich, ob die Qualität der generierten Elaborationen sich durch Few-Shot-Learning verbessert. Dabei werden der Problembeschreibung zwei Aufgaben der gleichen Komplexität mit den dazugehörigen hochqualitativen Elaborationen vorangestellt (siehe Schaubild 4.2). Von diesen Musterlösungen soll GPT-2 sich die erwünschte Struktur der Elaborationen „abschauen“. Im Unterschied zu Betz et al. erstelle ich keine statischen Beispiele, sondern wähle zu jedem Eintrag zufällig zwei andere Aufgaben der gleichen Komplexität aus dem Datensatz aus. Ich teste auf beiden *ModifiedChainRuler*-Datensätzen die funktionale Güte von Elaborationen, die durch Few-Shot-Learning mit Vorwärtsbeweisen und von Elaborationen, die durch Few-Shot-Learning mit Rückwärtsbeweisen generiert werden. Anhand einer Stichprobe von je 125 Elaborationen untersuche ich manuell, was das Sprachmodell aus den Beispielen im Few-Shot-Learning übernimmt (Abschnitt 8.4).

5. Generierung der Datensätze

Um die Schlussfolgerungsfähigkeiten neuronaler Sprachmodelle und den Einfluss dynamischer Kontexterweiterungen zu bewerten, generiere ich einen synthetischen Datensatz auf Grundlage des *ChainRuler*-Datensatzes von Betz et al. [3, Kapitel 3.1]. Im Folgenden stelle ich den *ChainRuler*-Datensatz vor und diskutiere anhand seiner Einschränkungen die Entscheidungen, die ich beim Erstellen des *ModifiedChainRuler*-Datensatzes getroffen habe.

5.1. *ChainRuler*-Datensatz

Der *ChainRuler*-Datensatz besteht aus deduktiven Schlussfolgerungsproblemen, die jeweils die folgenden natürlichsprachlichen Komponenten enthalten [3, Kapitel 3.1]:

- ein Fakt der Form „a is F“ mit einem Subjektnamen a und einem Prädikat F (möglicherweise negiert, „a is not F“)
- eine Lösung der Form „a is G“ mit Lösungsprädikat G, möglicherweise negiert
- zwei falsche Alternativen, die im Widerspruch zur Konklusion stehen: „a is not G“ und „a is $\overline{G}$“, wobei das Prädikat $\overline{G}$ konzeptuell komplementär zu G ist
- Regelkette der Länge d (Tiefe des Problems): Verallgemeinerte Konditionalsätze, die die Ableitung der Konklusion aus dem Fakt ermöglichen. Die Struktur der Regelkette ist $F \subset I_1, I_1 \subset I_2, ..., I_{d-1} \subset G$. Bei Kontrapositionsaufgaben hat sie die Form $F \subset I_1, I_1 \subset I_2, ..., \neg G \subset \neg I_{d-1}$.
- Menge von b (Breite des Problems) Distraktoren: Verwirrende Regeln $H_1 \subset X_1, H_2 \subset X_2, ..., H_b \subset X_b$, mit Antezedens-Prädikaten $H_1, ..., H_b$, welche ansonsten nicht in der Aufgabe vorkommen. Die Konsequenz-Prädikate $X_1, ..., X_b$ entsprechen jeweils dem Lösungsprädikat, seinem logischen oder konzeptuellen Komplement oder dem logischen Komplement des konzeptuellen Komplements: $X_i \in \{G, \neg G, \overline{G}, \neg\overline{G}\}$ für alle $i = 1, ..., b$.

Die Probleme des Datensatzes werden synthetisch erzeugt. Dazu werden Subjektnamen und Prädikate zufällig aus einer Datenbank an Vornamen, natürlichsprachlichen Prädikaten und deren konzeptuellen Komplementen ausgewählt und in die Vorlagen für Regeln („`If someone is [Antezendens-Prädikat] then they are [Konsequenz-Prädikat]`“) und Aussagen („`[Name] is [Prädikat]`“) eingesetzt. Die Prädikate, die in der Konsequenz (dem „`then`“-Teil) der Distraktoren auftauchen, werden randomisiert aus den jeweiligen vier

Optionen gewählt. Es kann also passieren, dass zufälligerweise alle Distraktoren einer Aufgabe das Lösungsprädikat beinhalten. Betz et al. messen daher die Anzahl an Distaktoren, deren Konsequenz-Prädikat nicht dem Lösungsprädikat entspricht, die *effektive Ablenkung*. Ob der Fakt in negierter Form vorkommt, wird ebenfalls zufällig entschieden. Die Tiefe d (Anzahl an Regeln) und Breite b (Anzahl an Distraktoren) der Aufgaben werden systematisch variiert, wobei $1 \leq d \leq 6$ und $0 \leq b \leq 5$ gilt. Die Hälfte der Aufgaben sind Kontrapositionsaufgaben. Der Datensatz ist außerdem balanciert: Zu einem Problem P_1 mit Lösungsprädikat G existieren Probleme P_2, P_3 und P_4 mit den Lösungsprädikaten $\neg G$, $\overline{G}$ und $\neg\overline{G}$, die ansonsten zu P_1 identisch sind. Betrachten wir dazu ein Beispiel aus [3, Tabelle 1]:

fact	„Lily is blue.“
rulechain	„If someone is blue, then they are careful.“, „If someone is careful, then they are loud.“, „If someone is not generous, then they are not loud.“
distractors	„If someone is in need of money, then they are not generous.“, „If someone is guilty, then they are not generous.“
conclusion	„Lily is generous“
alternatives	„Lily is not generous“, „Lily is stingy“
depth	3
breadth	2
contraposition	True
eff. distraction	2

Eine Problembeschreibung (Kontext) besteht aus einer zufälligen Permutation der relevanten Regeln, der Distraktoren und des Fakts. Beim obigen Beispiel [3, Tabelle 2] etwa: „If someone is not generous, then they are not loud. If someone is blue, then they are careful. Lily is blue. If someone is in need of money, then they are not generous. If someone is guilty, then they are not generous. If someone is careful, then they are loud.“

Das Problem kann gelöst werden, indem man aus dem Kontext die relevanten Regeln identifiziert und in der richtigen Reihenfolge d logische Ableitungen vollzieht:

1. „Lily is blue. If someone is blue, then they are careful.“ $\xrightarrow{\text{Modus Ponens}}$ „Lily is careful.“
2. „Lily is careful. If someone is careful, then they are loud.“ $\xrightarrow{\text{Modus Ponens}}$ „Lily is loud.“
3. „Lily is loud. If someone is not generous, then they are not loud.“ $\xrightarrow{\text{Modus Tollens}}$ „Lily is generous.“

5.2. Negationen

Betz et al. testen die Zero-Shot-Performanz des vortainierten Sprachmodells GPT-2 auf dem *ChainRuler*-Datensatz. Dabei zeigt sich unter Anderem, dass die Antwortvorhersagen des Sprachmodells davon abhängen, ob zufällig Negationen in der Aufgabenbeschreibung vorkommen. Betz et al. schreiben:

> „The ability of GPT-2 to generate and exploit ideal elaborations – in particular: proof chains – is strongly influenced by the presence of negations in the problem description. Once more, „not“ turns out to be a trouble-maker.“ [3, S.12]

Diese Beobachtung ist konsistent mit Forschungsergebnissen, die zeigen, dass unidirektionale Sprachmodelle die Bedeutung negierter Sätze im unüberwachten Vortraining nicht implizit erfassen [22], dass Negationen die Qualität maschineller Übersetzungen stark beeinträchtigen [17] und dass vortrainierte Sprachmodelle nicht zwischen negierten und nicht-negierten Aussagen unterscheiden können [20]. Um diese Schwächen im Verständnis negierter Aussagen zu beheben, sind grundlegende Änderungen beim Training oder in der Architektur neuronaler Sprachmodelle nötig [20], die den Umfang dieser Bachelorarbeit übersteigen. Sie erschweren jedoch die Interpretation der Ergebnisse auf dem *ChainRuler*-Datensatz: Angenommen, ein Sprachmodell sagt das logische Komplement „Tim is loud“ anstatt der richtigen Lösung „Tim is not loud“ vorher. In diesem Szenario ist nicht klar, ob das Sprachmodell daran gescheitert ist, auf Grundlage der Regelkette logisch abzuleiten, oder ob es die Aussagen „Tim is loud“ und „Tim is not loud“ konzeptuell nicht unterscheiden kann. Die Performanz des Sprachmodells variiert zusätzlich dadurch, dass auch die Fakten in manchen Problembeschreibungen zufällig negiert vorkommen.

Für die Zwecke dieser Bachelorarbeit ist es sinnvoll, von diesem bekannten Problem zu abstrahieren. Der *ModifiedChainRuler*-Datensatz enthält aus diesem Grund keine Negationen und folglich auch keine Kontrapositionsaufgaben. Das ist auch bei der Interpretation der Ergebnisse zu beachten: Der *ModifiedChainRuler*-Datensatz kann keine Aussagen über die *allgemeinen* Fähigkeiten der getesteten Sprachmodelle liefern, komplexe Argumentationen zu vollziehen. Er ist nur dazu geeignet, zu überprüfen, ob Sprachmodelle in der Lage sind, Schlussfolgerungen aus einer nicht-negierten Aussage und einfachen Regeln ohne Negationen zu ziehen. Der Begriff „Schlussfolgerungsfähigkeit“ wird hier in diesem Sinn verwendet.

Die Entscheidung, keine Negationen zu verwenden, hat zur Konsequenz, dass keine Aufgaben erzeugt werden können, deren Konklusion negiert ist (beispielsweise „`Lily is not generous`“). Das bedeutet, dass es nicht möglich ist, einen balancierten Datensatz zu erzeugen, der als Antwortoptionen die Konklusion („`Lily is generous`“), das logische Komplement („`Lily is not generous`“) und das konzeptuelle Komplement („`Lily is stingy`“) verwendet. Im *ModifiedChainRuler*-Datensatz gibt es daher nur zwei Antwortmöglichkeiten – die Konklusion und ihr konzeptuelles Komplement.

Welche Rolle die beiden falschen Alternativen für die Aufgabe spielen, soll anhand eines anderen Beispiels aus dem *ChainRuler*-Datensatz [3, Tabelle 2] illustriert werden:

> „`If someone is empty, then they are innocent. If someone is green, then they are loud. If someone is loud, then they are guilty. Jill is green.`“

Um die richtige Antwort vorherzusagen, muss das Sprachmodell auf Grundlage des gegebenen Kontexts die höchste Wahrscheinlichkeit für die Aussage „`Jill is guilty`“ im Vergleich zu den beiden Alternativen „`Jill is not guilty`“ und „`Jill is innocent`“ berechnen.

Betrachten wir zunächst das logische Komplement: Durch Ableitungen aus dem Kontext sollte das Sprachmodell zu dem Ergebnis kommen, dass „`Jill is guilty`“ gilt. Mit dem Satz des ausgeschlossenen Dritten folgt daraus, dass „`Jill is not guilty`“ falsch ist. Die Ergebnisse von Kassner und Schütze weisen allerdings darauf hin, dass Sprachmodelle ohne Finetuning nicht in der Lage sind, diese Schlussfolgerung zu ziehen [20]. Vortrainierte Sprachmodelle, denen Lückentext-Aufgaben (beispielsweise „`A beagle is a type of [MASK]`“) und ihre logischen Komplemente (hier „`A beagle is not a type of [MASK]`“) präsentiert werden, sagen in den meisten Fällen die gleichen Vervollständigungen (hier „`dog`“) für beide Varianten vorher. Die Autor:innen vermuten, dass die Schwächen bei der Interpretation von Negationen sich darauf zurückführen lassen, dass negierte Aussagen in den Trainingsdaten unterrepräsentiert sind. Diese Überlegungen sind konsistent mit anderen Arbeiten, die zeigen, dass negierte Aussagen im Allgemeinen seltener in Texten vorkommen als nicht-negierte Aussagen (siehe [18] für eine Zusammenstellung). Da die Vorhersagen eines Sprachmodells auf den Verteilungen basieren, die es im Training gesehen hat (siehe Abschnitt 2.1), liegt die Vermutung nahe, dass vortrainierte Sprachmodelle generell dazu tendieren, eine höhere Wahrscheinlichkeit für nicht-negierte Aussagen als für deren negierte Komplemente zu berechnen. Bei der Verwendung logischer Komplemente sollte man daher beachten, dass ein Sprachmodell (i) vermutlich nicht in der Lage ist, die Konklusion und ihr logisches Komplement konzeptuell zu unterscheiden und (ii) möglicherweise dazu tendiert, die nicht-negierte Variante der Aussage zu bevorzugen.

Betrachten wir nun das konzeptuelle Komplement „`Jill is innocent.`“. Dem Kontext zufolge gilt die allgemeine Regel „`If someone is empty, then they are innocent.`“. Unter der Annahme zur Weltoffenheit (*open-world assumption* [38]) gehen wir davon aus, dass Aussagen, die sich mit den explizit gegebenen Fakten und Regeln nicht beweisen lassen, nicht zwangsläufig falsch sind. Eine Aussage wie „`Jill is friendly.`“ ist aus dem Kontext zwar nicht beweisbar, könnte aber trotzdem wahr sein. Wieso ist es dagegen nicht zulässig, anzunehmen, dass „`Jill is empty.`“ wahr ist und mithilfe der Regel „`If someone is empty,`

`then they are innocent.“` auf „`Jill is innocent.“` zu schließen? Das ergibt sich nur, wenn man semantisches Wissen über die Bedeutung der Prädikate einbezieht. Wir gehen davon aus, dass die Aussagen „`Jill is guilty.“` und „`Jill is innocent.“` sich konzeptuell widersprechen und folglich nicht gleichzeitig wahr sein können. Dabei ist zu beachten, dass es einzelne Situationen geben kann, in denen diese Annahme ungültig ist. Beispielsweise könnten die Aussagen „`Jill is guilty.“` und „`Jill is innocent.“` gleichzeitig wahr sein, wenn man annehmen würde, dass Jill für zwei verschiedene Verbrechen angeklagt ist. Solche Ausnahmen lassen sich auch für andere konzeptuelle Komplemente finden. Die Aussagen „`Anne is still.“` und „`Anne is moving.“` könnten zum Beispiel gleichzeitig wahr sein, wenn Anne sich in einem Zug befände. Wir treffen im Folgenden trotzdem die Annahme, dass konzeptuelle Komplemente sich gegenseitig ausschließen, denn: Es ist im obigen Beispiel zwar prinzipiell *möglich*, dass Jill in einer weiteren Angelegenheit angeklagt und unschuldig ist, aber es ist nicht *wahrscheinlich*. Da Sprachmodelle nicht mit absoluten Wahrheiten, sondern mit Wahrscheinlichkeiten arbeiten, können wir sowieso nur *interpretieren*, was das Modell „für wahr hält“, indem wir messen, wofür es eine hohe Wahrscheinlichkeit vorhersagt. Das heißt, für die Zwecke dieser Aufgabe können wir „unwahrscheinlich“ und „falsch“ als äquivalent betrachten.

Diese Überlegungen zeigen, dass die Konklusion und ihr konzeptuelles Komplement als Antwortmöglichkeiten ausreichen, um zu überprüfen, ob ein Sprachmodell in der Lage ist, logische Schlussfolgerungen aus einem gegebenen Kontext zu ziehen.

5.3. Distraktoren

Neben dem Fakt und den notwendigen Regeln enthalten die *ChainRuler*-Aufgaben auch bis zu fünf verwirrende Regeln, sogenannte Distraktoren. Dabei handelt es sich um Konditionalsätze der Form „`If someone is [Antezedens-Prädikat], then they are [Konsequenz-Prädikat].“`. Distraktoren machen es zum Einen schwieriger, die relevanten Regeln im Kontext zu identifizieren. Zum Anderen lässt sich mit ihnen überprüfen, ob das Sprachmodell die Regelkette korrekt nachvollzieht. Betrachten wir wieder die Problembeschreibung „`If someone is empty, then they are innocent. If someone is green, then they are loud. If someone is loud, then they are guilty. Jill is green.“` [3, Tabelle 2]. Angenommen, das Sprachmodell gelangt nicht zu dem Zwischenergebnis „`Jill is loud.“`. In diesem Fall hat es keinen Anhaltspunkt, welche der Regeln „`If someone is empty, then they are innocent.“` und „`If someone is loud, then they are guilty.“` es auf das Subjekt anwenden kann. Beim *ChainRuler*-Datensatz werden die Konsequenz-Prädikate der Distraktoren jeweils zufällig aus den folgenden vier Optionen gewählt:

- Lösungsprädikat G, hier „`guilty`“
- logisches Komplement $\neg G$, hier „`not guilty`“
- konzeptuelles Komplement $\overline{G}$, hier „`innocent`“
- logisches Komplement des konzeptuellen Komplements $\neg\overline{G}$, hier „`not innocent`“

Die ersten drei Optionen sind die Prädikate der Antwortmöglichkeiten. Es kann also vorkommen, dass mehrere Distraktoren einer Aufgabe auf die richtige Lösung verweisen. Um diese Eigenschaft der Aufgaben zu messen, führen Betz et al. die „effektive Ablenkung“ als zusätzliches Maß ein. Die effektive Ablenkung eines Problems ist die Anzahl an Distraktoren, deren Konsequenz-Prädikat nicht dem Lösungsprädikat G entspricht.

Betz et al. stellen fest, dass das verwendete Sprachmodell seine Antwortvorhersagen danach ausrichtet, wie häufig die Antwortprädikate G, $\neg G$ und $\overline{G}$ in der Aufgabenstellung vorkommen. Sie schreiben dazu:

> „First of all, we find that GPT-2 follows a **simple heuristic** for solving the ChainRuler task: It's predictions are seemingly just based on how frequently the predicate of an answer-option appears in the consequent of the problem's rules. Whenever a problem description contains, by chance, many distractors whose „then“-part corresponds to the correct answer, GPT-2 achieves very high accuracy. [...] If the model is, however, not lucky and many distractors coincidentally point towards the wrong answers (i.e., high effective distraction), then the model typically gets the answer wrong and performs substantially worse than naïve random guessing (accuracy=.33).“ [3, S.7]

Das Sprachmodell basiert seine Vorhersagen also darauf, welche Prädikate häufig im Kontext vorkommen, anstatt deduktive Schlussfolgerungen durchzuführen. Da die Prädikate der Distraktoren zufällig gewählt sind, lässt sich ihr Effekt nur schwer kontrollieren. Mit der *effektiven Ablenkung* wird nur erfasst, wie viele der Distraktoren nicht auf die richtige Lösung hinweisen. Dabei muss allerdings beachtet werden, dass nicht alle dieser Distraktoren auf *falsche Antwortmöglichkeiten* hinweisen. Beispielsweise könnten zufällig alle b Distraktoren einer Aufgabe das Prädikat $\neg\overline{G}$ enthalten, welches in keiner Antwortoption vorkommt. Die effektive Ablenkung dieses Problems wäre dann ebenfalls b. Selbst wenn alle Distraktoren auf falsche Antworten verweisen, bleibt offen, wie viele davon auf das konzeptuelle und wie viele auf das logische Komplement entfallen.

Die Beobachtungen von Betz et al. legen nahe, dass GPT-2 eine höhere Wahrscheinlichkeit für Prädikate berechnet, die es im Kontext „gesehen“ hat. Wie stark dieser Bias ist, wird allerdings nicht systematisch untersucht. Hinzu kommt, dass die Vorhersagen vermutlich zusätzlich in Richtung nicht-negierter Aussagen verschoben sind (siehe Abschnitt 5.2). Das macht es besonders schwierig, den Einfluss der Heuristik auf dem *ChainRuler*-Datensatz zu untersuchen.

Beim *ModifiedChainRuler*-Datensatz enthält jede Aufgabe daher genau einen Distraktor, der auf die falsche Alternative hinweist. Die restlichen verwirrenden Regeln verweisen auf Komplemente von Konzepten, die in der Regelkette auftauchen (siehe Schaubild 5.1). Dadurch kommen das Lösungsprädikat und sein Komplement jeweils genau einmal in der Aufgabenbeschreibung vor und das Sprachmodell kann seine Heuristik nicht anwenden. Die ablenkenden Eigenschaften der Distraktoren bleiben dabei erhalten.

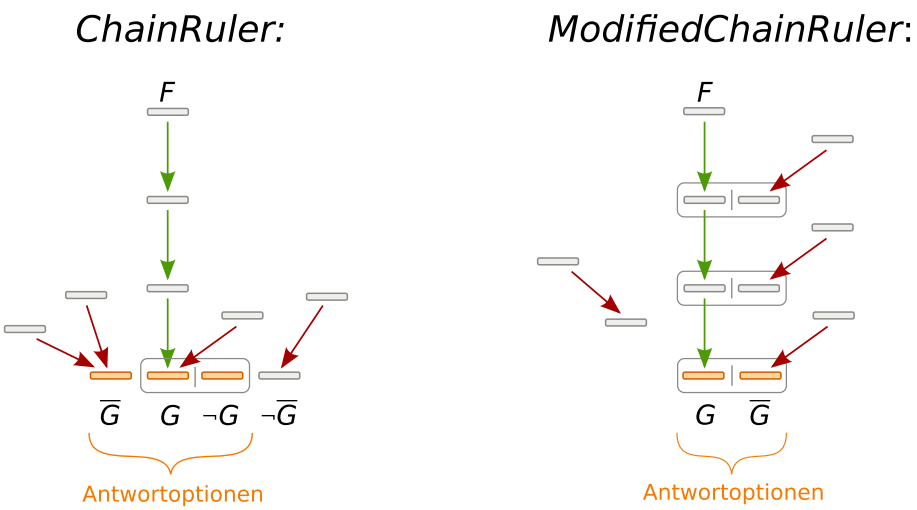


Abbildung 5.1.: Konstruktion der Distraktoren bei *ChainRuler*-Aufgaben und *ModifiedChainRuler*-Aufgaben

5.4. *ModifiedChainRuler*-Datensatz

Der *ChainRuler*-Datensatz eignet sich wegen der in Abschnitt 5.2 und 5.3 diskutierten Probleme nur eingeschränkt dazu, die Schlussfolgerungsfähigkeiten neuronaler Sprachmodelle zu messen. Ob das Modell in der Lage wäre, mehrschrittige deduktive Ableitungen durchzuführen, wenn es nicht auf eine einfache Heuristik zurückgreifen könnte, ist mit diesem Datensatz nicht erfassbar. Ich erstelle daher einen balancierten Datensatz, der die Verwendung der beschriebenen Heuristik unterbindet und keine Negationen enthält.

Der *ModifiedChainRuler*-Datensatz wird mithilfe einer Datenbank an Vornamen und konzeptuell komplementären Prädikaten (siehe Anhang, Tabelle A.1) synthetisch erstellt. Jede Aufgabe hat die folgenden Bestandteile:

- ein **Fakt** der Form „a `is` F“ mit einem Subjektnamen a und einem Prädikat F
- eine **Konklusion** der Form „a `is` G“ mit Lösungsprädikat G
- eine **falsche Alternative** der Form „a `is` $\overline{G}$“ mit einem Prädikat $\overline{G}$, das konzeptuell komplementär zu G ist
- **Regelkette** der Länge d (Tiefe des Problems): Verallgemeinerte Konditionalsätze, die die Ableitung der Konklusion aus dem Fakt ermöglichen. Die Struktur der Regelkette ist

  $$F \subset I_1, I_1 \subset I_2, ..., I_{d-1} \subset G$$

  mit paarweise verschiedenen Prädikaten ($I_j \neq I_k$ für alle $j \neq k$)
- Menge von b (Breite des Problems) **Distraktoren**: Verwirrende Regeln

  $$H_1 \subset X_1, ..., H_b \subset X_b$$

mit paarweise verschiedenen Antezedens-Prädikaten $H_1, ..., H_b$, welche von der Regelkette unabhängig sind. Der ausgleichende Distraktor $H_1 \subset X_1$ verweist auf die falsche Alternative ($X_1 = \overline{G}$). Falls $b \leq d$ gilt, verweisen die restlichen Distraktoren auf Komplemente der Zwischenergebnisse. Jedes Komplement kommt dabei höchstens einmal vor: $\{X_2, ..., X_b\} \subseteq \{\overline{I_1}, ..., \overline{I_{d-1}}\}$.
Falls $b > d$ gilt, kommt jedes Komplement genau einmal vor, $\{X_2, ..., X_d\} = \{\overline{I_1}, ..., \overline{I_{d-1}}\}$ und die restlichen Antezedens-Prädikate $X_{d+1}, ..., X_b$ sind von der Regelkette unabhängig.

Zusätzlich zur Tiefe und Breite erfasse ich auch die *Gruppe* eines Problems. Die Namen und Prädikate werden manuell in disjunkte Gruppen geteilt, bevor der Datensatz synthetisiert wird. Regeln werden nur aus Prädikaten gebildet, die sich in der gleichen Gruppe befinden. Dafür gibt es zwei Gründe: Erstens sollen die Prädikate innerhalb einer Regel wenig semantischen Bezug zueinander haben, um potentiell verwirrende Aussagen wie „`If someone is sleeping, then they are attentive.`“ zu vermeiden. Daher werden nur möglichst unabhängige Prädikate miteinander gruppiert. In Abschnitt 5.6 diskutiere ich diesen Ansatz und seine Einschränkungen ausführlicher. Zweitens teilen Probleme, die unterschiedlichen Gruppen angehören, weder Subjektnamen noch Prädikate (abgesehen vom Quell-Prädikat F). Sie können dadurch unkompliziert für Anwendungen wie beispielsweise Few-Shot-Learning (siehe 8.3) verwendet werden, in denen es notwendig ist, dass sich aus den Regeln eines Problems keine Ableitungen über die Antwortoptionen eines anderen Problems treffen lassen.

Bei der Generierung des Datensatzes werden Tiefe und Breite der Probleme systematisch variiert, wobei für die Tiefe $1 \leq d \leq 5$ und für die Breite $1 \leq b \leq 5$ gilt. Insbesondere existieren im Gegensatz zum *ChainRuler*-Datensatz keine Probleme der Breite 0, da jedes Problem zumindest einen ausgleichenden Distraktor enthält. Der *ModifiedChainRuler*-Datensatz ist balanciert: Zu jedem Problem A existiert ein Problem B, das sich von A nur dadurch unterscheidet, dass das Lösungsprädikat und sein konzeptuelles Komplement darin vertauscht vorkommen.

Betrachten wir als Beispiel das folgende Problem 5.2:

Fakt	„`Maria is charming.`“
Regelkette	„`If someone is charming, then they are careless.`“, „`If someone is careless, then they are happy.`“, „`If someone is happy, then they are awake.`“
Zwischenergebnisse	„`Maria is careless.`“, „`Maria is happy.`“
Distraktoren	„`If someone is young, then they are sad.`“, „`If someone is serious, then they are sleeping.`“
Konklusion	„`Maria is awake.`“
Falsche Alternative	„`Maria is sleeping.`“
Tiefe	3
Breite	2

Gruppe	1

Eine Problembeschreibung besteht aus der Regelkette, dem Fakt und den Distraktoren in zufälliger Reihenfolge. Beispielsweise „`If someone is happy, then they are awake. If someone is young, then they are sad. Maria is charming. If someone is careless, then they are happy. If someone is charming, then they are careless. If someone is serious, then they are sleeping.`“

Wie wir sehen, kommen das Lösungsprädikat „`awake`“ und sein konzeptuelles Komplement „`sleeping`“ jeweils einmal im Kontext vor, wodurch das Modell seine Heuristik nicht anwenden kann. Gleichzeitig bleiben die ablenkenden Eigenschaften der Distraktoren erhalten: Wenn das Modell nicht bis zum Zwischenergebnis „`Maria is happy.`“ gelangt, hat es keinen Hinweis darauf, ob es die Regel „`If someone is happy, then they are awake.`“ oder den ausgleichenden Distraktor „`If someone is serious, then they are sleeping.`“ benutzen muss, um zur richtigen Lösung zu finden. Der zweite Distraktor „`If someone is young, then they are sad.`“ erschwert es dem Modell zusätzlich, das Zwischenergebnis „`Maria is happy.`“ zu ziehen.

Ein Problem, dessen Breite seine Tiefe übersteigt, hat zusätzlich Distraktoren, die auf unabhängige Konzepte verweisen. Diese Regeln dienen vor Allem dazu, das Problem komplexer zu machen. Das macht es schwieriger, die relevanten Regeln zu identifizieren. Beispiel 5.3 zeigt ein Problem aus dem *ModifiedChainRuler*-Datensatz, für das $b > d$ gilt:

Fakt	„`Patricia is beautiful.`“
Regelkette	„`If someone is beautiful, then they are small.`“, „`If someone is small, then they are wise.`“
Zwischenergebnisse	„`Patricia is small.`“
Distraktoren	„`If someone is sitting, then they are tall.`“, „`If someone is angry, then they are fresh.`“, „`If someone is attentive, then they are stingy.`“, „`If someone is popular, then they are cold.`“, „`If someone is blunt, then they are foolish.`“
Konklusion	„`Patricia is wise.`“
Falsche Alternative	„`Patricia is foolish.`“
Tiefe	2
Breite	5
Gruppe	0

Dieses Problem enthält alle drei Typen von Distraktoren: „`If someone is blunt, then they are foolish.`“ verweist auf die falsche Alternative, „`If someone is sitting, then they are tall.`“ deutet auf das Komplement des Zwischenergebnisses hin und „`If someone is angry, then they are fresh.`“, „`If someone is attentive, then they are stingy.`“ und „`If someone is popular, then they are cold.`“ führen zu unabhängigen Konzepten. Durch die Konstruktion der Distraktoren kommt jedes Prädikat höchstens einmal in der Aufgabenbeschreibung vor. Das Sprachmodell hat folglich keine quantitativen Anhaltspunkte, an denen es sich bei der Vorhersage orientieren kann.

5.5. Eignung zum Messen der Schlussfolgerungsfähigkeiten

Im Folgenden lege ich dar, wieso der *ModifiedChainRuler*-Datensatz dazu geeignet ist, die Schlussfolgerungsfähigkeiten neuronaler Sprachmodelle zu bewerten. Betrachten wir dazu, welche Faktoren neben der Schlussfolgerungsfähigkeit eine Rolle bei der Vorhersage spielen könnten:

Wie in Abschnitt 5.2 besprochen, ist zu erwarten, dass die Vorhersagen des Sprachmodells generell in Richtung nicht-negierter Aussagen verschoben sind. Da beide Antwortoptionen beim *ModifiedChainRuler*-Datensatz affirmativ (nicht-negiert) sind, kann dieser Bias die Vorhersage nicht verfälschen.

Des Weiteren zeigen die Ergebnisse von Betz et al., dass die Vorhersagen stark dadurch beeinflusst werden, ob und wie häufig die jeweiligen Antwortprädikate im Kontext vorkommen (siehe Abschnitt 5.3). Im Gegensatz zum *ChainRuler*-Datensatz wird beim *ModifiedChainRuler*-Datensatz durch die Konstruktion der Distraktoren sichergestellt, dass die Prädikate der beiden Antwortmöglichkeiten (im Beispiel „`awake`“ und „`sleeping`“) jeweils genau einmal im Kontext auftauchen.

Denkbar ist auch, dass das Sprachmodell eine allgemeine Tendenz zu einer der beiden Aussagen hat. Beispielsweise ist es möglich, dass der Ausdruck „`sleeping`“ häufiger in den Trainingsdaten vorkommt als der Begriff „`awake`“. Das könnte dazu führen, dass das Sprachmodell es im Allgemeinen für wahrscheinlicher hält, dass ein Satz das Prädikat „`sleeping`“ beinhaltet. Wie Kassner und Schütze zeigen, spielt die Wortverteilung im gegebenen Kontext eine deutlich größere Rolle für die Vorhersage als im Vortraining erlangtes Wissen [20]. Da beide Prädikate im Kontext in syntaktisch ähnlichen Sätzen auftauchen, können wir also davon ausgehen, dass dieser Effekt höchstens einen kleinen Einfluss auf die Vorhersage hat. Unabhängig von diesen Überlegungen ist der *ModifiedChainRuler*-Datensatz aber auch dazu geeignet, solche Einflüsse auszugleichen: Zu jedem Problem existiert ein symmetrisches Problem, in dem nur die Antwortprädikate vertauscht sind.

Das zu Beispiel 5.2 symmetrische Problem lautet etwa:

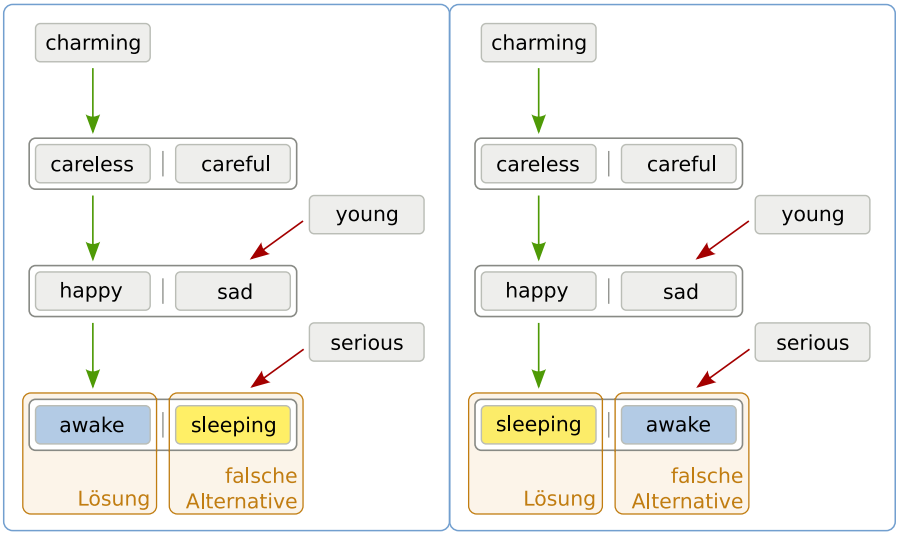


Abbildung 5.2.: Beispiel für symmetrische *ModifiedChainRuler*-Aufgaben

Fakt	„`Maria is charming.`“
Regelkette	„`If someone is charming, then they are careless.`“, „`If someone is careless, then they are happy.`“, „`If someone is happy, then they are sleeping.`“
Zwischenergebnisse	„`Maria is careless.`“, „`Maria is happy.`“
Distraktoren	„`If someone is young, then they are sad.`“, „`If someone is serious, then they are awake.`“
Konklusion	„`Maria is sleeping.`“
Falsche Alternative	„`Maria is awake.`“
Tiefe	3
Breite	2
Gruppe	1

Wenn das Sprachmodell also keine Schlussfolgerungen vollzieht, sondern Wissen aus dem Vortraining verwendet, um sich für eine der Optionen zu entscheiden, trifft es in einem der symmetrischen Probleme die richtige und im anderen die falsche Wahl (siehe Schaubild 5.2). In diesem Fall entspricht seine Vorhersage auf dem Datensatz also einem Zufallsklassifikator.

Des Weiteren könnte die Reihenfolge der Prädikate im Kontext beeinflussen, welche der beiden Antwortmöglichkeiten vorhergesagt wird. Aus diesem Grund enthält der Kontext die Sätze der Problembeschreibung in zufällig permutierter Reihenfolge. Das Lösungs-

prädikat taucht also mit 50-prozentiger Wahrscheinlichkeit vor seinem Komplement im Kontext auf.

Aus diesen Überlegungen können wir folgern, dass ein Sprachmodell, das nicht in der Lage ist, Schlussfolgerungen zu vollziehen, auf dem *ModifiedChainRuler*-Datensatz mit einer Wahrscheinlichkeit von 50 Prozent die richtige Lösung vorhersagt. Signifikante Abweichungen von dieser Basiswahrscheinlichkeit sind also darauf zurückzuführen, dass das Sprachmodell korrekte deduktive Ableitungen aus dem Kontext durchführt – oder im negativen Fall systematisch falsch ableitet.

5.6. *ModifiedChainRuler*-Fantasie-Datensatz

Mit dem *ModifiedChainRuler*-Datensatz kann getestet werden, ob ein Sprachmodell in der Lage ist, aus natürlichsprachlichen Aussagen logische Schlüsse zu ziehen. Die formalen Problembeschreibungen werden dafür mit natürlichsprachlichen Prädikaten instanziiert. Dafür existieren mehrere Gruppen von Prädikaten, wobei Prädikate innerhalb einer Gruppe im Idealfall möglichst wenig semantische Verbindung zueinander haben. Regeln werden gebildet, indem zwei zufällige Prädikate der gleichen Gruppe ausgewählt und in die Vorlage „`If someone is [Antezedens-Prädikat], then they are [Konsequenz-Prädikat].`“ eingesetzt werden. Durch dieses Vorgehen sollen Aussagen vermieden werden, die im Widerspruch zu semantischem Wissen stehen, welches das Sprachmodell im Vortaining erlernt hat. Eine Aussage wie „`If someone is sleeping, then they are attentive.`“ könnte das Modell verwirren und seine Schlussfolgerungsfähigkeiten untergraben. Die Konzepte [„`awake`“, „`sleeping`“] stehen semantisch zu den Konzepten [„`attentive`“, „`inattentive`“] in Zusammenhang und gehören daher beim *ModifiedChainRuler*-Datensatz unterschiedlichen Gruppen an. Damit wird verhindert, dass sie zusammen in einer Regel auftauchen. Dieses Vorgehen ist allerdings immer fehlerbehaftet, da die Entscheidung, welche Prädikate „unabhängig genug“ zueinander sind, auf individueller Einschätzung beruht. Semantische Beziehungen zwischen Konzepten lassen sich nicht eindeutig messen, weil Wortbedeutungen im Allgemeinen unscharf sind. Bei der Nutzung der Prädikate in verallgemeinerten Regeln kommt hinzu, dass sie mithilfe von übersprachlichem Wissen interpretiert werden. Eine Person, die den Satz „`If someone is young, then they are happy.`“ liest, würde vermutlich zunächst überlegen, ob sie junge Personen kennt, welche nicht fröhlich sind, bevor sie die Regel akzeptiert. Es hängt also zusätzlich vom persönlichen Hintergrund ab, welche Regeln eine menschliche Sprecherin verständlich findet. Das semantische Wissen, über das ein Sprachmodell verfügt, ist dagegen von den Kontexten abhängig, in denen dieses Wort im Trainingsdatensatz vorkommt [2].

Wie viel semantisches Wissen verwendet das Sprachmodell beim Lösen der Aufgabe? Dienen die Prädikate nur als „Platzhalter“, mit denen logische Ableitungen durchgeführt werden? „Verwirren“ die Bedeutungen der Prädikate das Modell oder helfen sie ihm, sich die Zusammenhänge zu „merken“? Ist es in der Lage, Schlussfolgerungen über ihm unbekannte Prädikate zu ziehen? Um diese Fragen zu beantworten, generiere ich einen weiteren Datensatz, bei dem alle Prädikate durch Neologismen ersetzt werden.

Die ausgedachten Wörter (beispielsweise „`gloosy`“, „`rakouming`“, „`lirful`“) haben keine Bedeutung. Mit ihnen sollen untersucht werden, ob das Sprachmodell in der Lage ist, Ableitungen über beliebige Platzhalter durchzuführen. Dass das verwendete Sprachmodell GPT-2 diese Neologismen verarbeiten kann, liegt an seinem Vokabular (siehe auch Abschnitt 2.1): Da mit seinem Basisvokabular auf Byte-Ebene alle Unicode-Zeichen repräsentiert werden können, ist GPT-2 in der Lage, jede beliebige Zeichensequenz zu verarbeiten. Die ausgedachten Worte enden auf typische Prädikatendungen wie „`-ly`“ und „`-ing`“, um ihre grammatische Funktion im Satz zu verdeutlichen. GPT-2 erlernt im Vortraining durch einen *byte pair encoding (BPE)*-Algorithmus [42][36] 50.000 Worte und typische Wortbestandteile. Es ist daher möglich, dass das Sprachmodell die verwendeten Suffixe interpretieren kann.

Strukturell ist der *ModifiedChainRuler*-Fantasie-Datensatz so weit wie möglich an den *ModifiedChainRuler*-Datensatz angelehnt. Er ist ebenfalls balanciert mit systematisch variierten Tiefen und Breiten. Durch die Verwendung von Neologismen ergeben sich allerdings kleine Unterschiede: Zu den erfundenen Prädikaten existieren keine konzeptuellen Komplemente. Daher gibt es zum Lösungsprädikat (beispielsweise „`steaby`“) auch kein konzeptuelles Komplement, das zur Konstruktion der falschen Alternative verwendet werden könnte. Eine Möglichkeit, dieses Problem zu lösen, wäre, ein anderes erfundenes Prädikat (beispielsweise „`soaful`“) zu benutzen. Das Sprachmodell hat aber keine Möglichkeit zu wissen, dass die Konzepte „`steaby`“ und „`soaful`“ sich widersprechen und deswegen vermutlich nicht gleichzeitig wahr sind. Nehmen wir als Beispiel an, der ausgleichende Distraktor in diesem Problem würde folgendermaßen lauten: „`If someone is funely, then they are soaful.`“. Unter der Annahme der Weltoffenheit (siehe Abschnitt 5.3) ist es möglich, dass das Prädikat „`funely`“ auf das Subjekt (nennen wir es „`Barbara`“) zutrifft. Das Sprachmodell hat zwar keinen besonderen Grund, anzunehmen, dass „`Barbara is funely.`“ gilt, aber auch keinen Grund, es für unwahrscheinlich zu halten. Die Antwortmöglichkeit „`Barbara is soaful.`“ wäre damit nicht *falsch*, sondern nur aus dem Kontext *nicht beweisbar*. Das würde die Aussagekraft der Ergebnisse drastisch reduzieren. Der *ModifiedChainRuler*-Fantasie-Datensatz verwendet als falsche Alternativen daher die logischen Komplemente („`Barbara is not steaby`“) der Konklusion („`Barbara is steaby`“), obwohl es die in Abschnitt 5.2 besprochenen Nachteile mit sich bringt.

Der spezielle ausgleichende Distraktor einer Aufgabe verweist wie im *ModifiedChainRuler*-Datensatz auf die falsche Alternative. Dadurch kommt eine Negation in der Aufgabenbeschreibung vor – entweder in der letzten Regel oder im ausgleichenden Distraktor. Um zusätzliche negierte Aussagen zu vermeiden, enden alle weiteren Distraktoren auf unabhängigen Konzepten. Betrachten wir dazu ein Beispiel:

Fakt	„`Lisa is foaby.`“
Regelkette	„`If someone is foaby, then they are laphiful.`“, „`If someone is laphiful, then they are not canimery.`“
Zwischenergebnisse	„`Lisa is laphiful.`“
Distraktoren	„`If someone is benoosy, then they are potaugh.`“, „`If someone is pawounig, then they are canimery.`“

Konklusion	„`Lisa is not canimery.`“
Falsche Alternative	„`Lisa is canimery.`“
Tiefe	2
Breite	2
Gruppe	1

Ein möglicher Kontext zu dieser Aufgabe lautet „`If someone is laphiful, then they are not canimery. Lisa is foaby. If someone is benoosy, then they are potaugh. If someone is foaby, then they are laphiful. If someone is pawounig, then they are canimery.`“. Um die Konklusion aus der Problembeschreibung abzuleiten, muss das Sprachmodell die relevanten von den verwirrenden Regeln unterscheiden und in der richtigen Reihenfolge über die unbekannten Prädikate schlussfolgern.

6. Evaluation grundlegender Schlussfolgerungsfähigkeiten

In diesem Kapitel evaluiere ich die Performanz eines vortrainierten Sprachmodells auf den *ModifiedChainRuler*- und *ModifiedChainRuler*-Fantasie-Datensätzen (für die Beschreibung der Datensätze siehe Abschnitt 5.4 und 5.6). Als Sprachmodell verwende ich (analog zu Betz et al. [3]) GPT2-XL in der Implementierung von HuggingFace [45] ohne Finetuning. Die Ergebnisse aus diesem Kapitel helfen dabei, zu verstehen, welche Kenntnisse und Fähigkeiten das Sprachmodell im Vortraining erlangt hat. Ich nutze sie außerdem als Basiswerte für die Schlussfolgerungsfähigkeiten des Sprachmodells, um den Effekt vorgegebener und dynamischer Kontexterweiterungen (Kapitel 7 und 8) zu messen.

6.1. Berechnung der Antwortvorhersage

Neuronale Sprachmodelle sind dazu geeignet, Wahrscheinlichkeitsverteilungen über Textsequenzen zu berechnen. Wie in Abschnitt 2.1 erklärt, werden unidirektionale Sprachmodelle wie GPT-2 darauf trainiert, abhängig von einer gegebenen Textsequenz Φ das nächste Token x_1 vorherzusagen. Diese Fähigkeit des Sprachmodells wird üblicherweise zur Textgenerierung genutzt (siehe auch Abschnitt 2.2). Sie kann aber auch dazu verwendet werden, zu vergleichen, für wie wahrscheinlich das Sprachmodell mehrere *vorgegebene* Textverläufe hält. Die Wahrscheinlichkeit, dass eine Sequenz $x_1, ..., x_n$ auf den Kontext Φ folgt, ergibt sich aus dem Produkt der bedingten Wahrscheinlichkeiten der einzelnen Tokens: $\mathcal{P}(x_1, ..., x_n|\Phi) = \prod_{i=1}^{n} \mathcal{P}(x_i|\Phi, x_1, ..., x_{i-1})$. Das Sprachmodell kann also bedingte Wahrscheinlichkeiten $\mathcal{P}(x_1, ..., x_n|\Phi)$ und $\mathcal{P}(y_1, ..., y_n|\Phi)$ dafür vorhersagen, dass die Sequenzen $x_1, ..., x_n$ beziehungsweise $y_1, ..., y_n$ auf den Kontext Φ folgen.

Übertragen wir diese Überlegungen nun auf den *ModifiedChainRuler*-Datensatz. Gegeben einem Kontext c, der die Problembeschreibung (und gegebenenfalls eine Problemelaboration, siehe Kapitel 7 und 8) enthält, können wir die Vorhersage des Sprachmodells folgendermaßen ermitteln (vgl. [3, Kapitel 3.3]):

Seien a_1 und a_2 die möglichen Antworten, also natürlichsprachliche Instanzen von „`[Subjekt] is [`G`].`“ und „`[Subjekt] is [`$\overline{G}$`].`“. Die bedingte Wahrscheinlichkeit, die das Sprachmodell für die Antwort a_i abhängig vom gegebenen Kontext c berechnet, ist $\mathcal{P}(a_i|c)$. Die Vorhersage des Modells ist diejenige Antwort, für die es die höchste bedingte

Wahrscheinlichkeit berechnet:

$$\text{argmax}_{i=1,2}(\mathcal{P}(a_i|c))$$

Der prozentuale Anteil richtiger Vorhersagen auf dem Datensatz ergibt die Korrektklassifizierungsrate (auch „Akkuratheit“ genannt).

6.2. Schlussfolgerungsfähigkeiten ohne Problemelaborationen

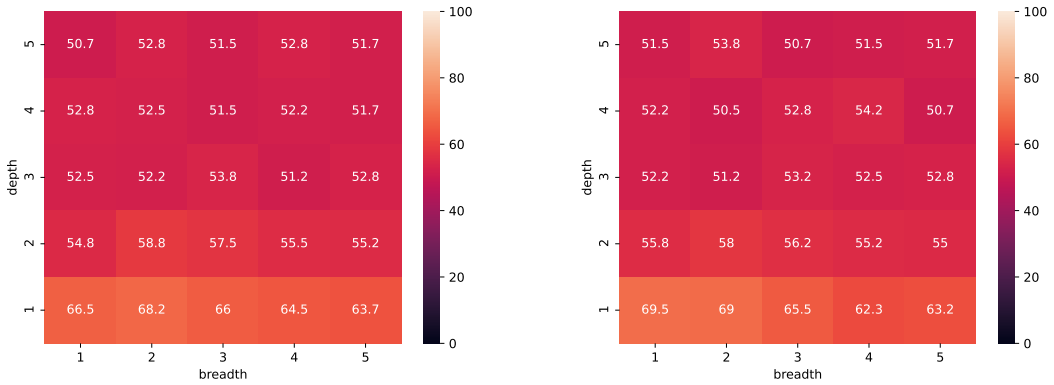
Um die Schlussfolgerungsfähigkeiten des vortrainierten Sprachmodells GPT-2 zu evaluieren, messe ich die Akkuratheit seiner Vorhersagen auf dem *ModifiedChainRuler*-Datensatz und dem *ModifiedChainRuler*-Fantasie-Datensatz mit jeweils 10.000 Problemen. Als Kontext verwende ich die Problembeschreibung (bestehend aus den relevanten Regeln, dem Fakt und den Distraktoren, siehe Abschnitt 5.4) sowie eine natürlichsprachliche Überleitung zur Antwort, die „finale Frage“. Als finale Frage wird analog zu Betz et al. „`Therefore,`“ verwendet [3, Abbildung 1]. Betrachten wir dieses Vorgehen am Beispiel eines Problems aus dem *ModifiedChainRuler*-Datensatz:

Kontext c (Problembeschreibung + finale Frage)	„`If someone is happy, then they are awake. If someone is young, then they are sad. Maria is charming. If someone is careless, then they are happy. If someone is charming, then they are careless. If someone is serious, then they are sleeping. Therefore,`“
Konklusion a_1	„`Maria is awake.`“
Falsche Alternative a_2	„`Maria is sleeping.`“

Die Vorhersage des Sprachmodells bei diesem Problem ist korrekt, wenn es für die Konklusion eine mindestens so hohe Wahrscheinlichkeit berechnet wie für die falsche Alternative, $\mathcal{P}(a_1|c) \geq \mathcal{P}(a_2|c)$.

6.2.1. Auswertung auf dem *ModifiedChainRuler*-Datensatz

Um die Stabilität der Ergebnisse gegenüber Veränderungen der finalen Frage zu kontrollieren, werden die Versuche auf dem *ModifiedChainRuler*-Datensatz einmal mit der finalen Frage „`Therefore,`“ (FQ1) durchgeführt und anschließend der finalen Frage „`Therefore, we know that`“ (FQ2) wiederholt. Abbildung 6.1 zeigt die Korrektklassifizierungsraten des vortrainierten Sprachmodells GPT-2 in beiden Varianten, aufgeschlüsselt nach der



(a) Akkuratheit | FQ1

(b) Akkuratheit | FQ2

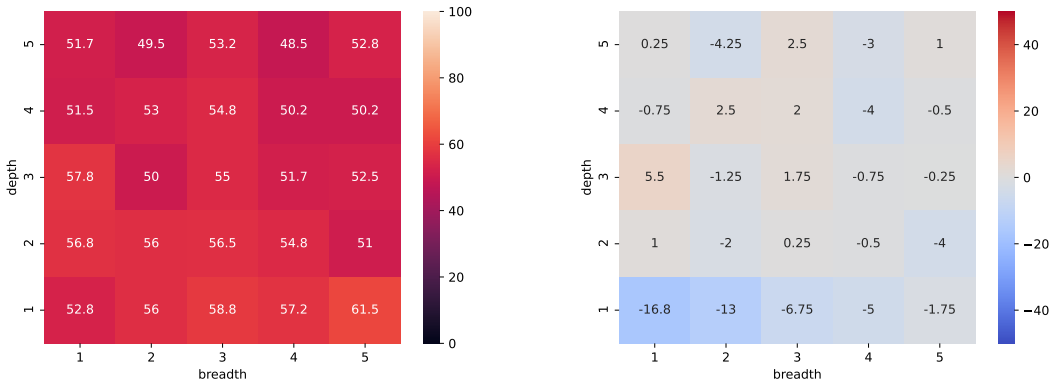
Abbildung 6.1.: Akkuratheit auf *ModifiedChainRuler*-Datensatz nach Tiefe und Breite mit finaler Frage „Therefore," (FQ1) oder „Therefore, we know that" (FQ2)

Komplexität (Tiefe und Breite) der Probleme. Die Korrektklassifizierungsraten in diesen und folgenden Grafiken sind in Prozentpunkten angegeben. Eine Akkuratheit von 66.5 in 6.1a, $depth = breadth = 1$, bedeutet beispielsweise, dass GPT-2 für 66,5% der Aufgaben mit Tiefe und Breite 1 – für 266 von 400 Aufgaben – die richtige Antwort vorhersagt.

Wie die Ausführungen in Abschnitt 5.5 verdeutlichen, ist eine Korrektklassifizierungsrate von 50 % zu erwarten, wenn das Sprachmodell nicht in der Lage ist, deduktive Schlussfolgerungen aus dem gegebenen Kontext zu ziehen. GPT-2 übertrifft diesen Wert selbst in Problemen mit hoher Komplexität geringfügig. Insgesamt zeigt GPT-2 ohne Finetuning leichte bis moderate Schlussfolgerungsfähigkeiten auf dem *ModifiedChainRuler*-Datensatz.

Bei Problemen der Tiefe 1 sagt GPT-2 in bis zu 69,5% der Probleme die richtige Antwort vorher. Diese Aufgaben sind so angelegt, dass man zur richtigen Lösung gelangt, indem man eine einzige Regel anwendet. Innerhalb der Probleme der Tiefe 1 verschlechtert sich die Akkuratheit (abgesehen von einer Ausnahme, siehe Abbildung 6.1a) mit zunehmender Breite. Die zusätzlichen Distraktoren beinhalten bei diesen Problemen ausschließlich Prädikate, die von der Regelkette unabhängig sind. Ab einer Tiefe von 2 ist es notwendig, mehrere Regeln zu verketten oder Zwischenschlüsse zu ziehen, um die Konklusion aus dem Fakt abzuleiten. Daher ist es nicht verwunderlich, dass die Performanz zwischen Tiefe 1 und Tiefe 2 sprunghaft abnimmt, mit einer Verschlechterung von 7,1 bis 13,7 absoluten Prozentpunkten zwischen Problemen der gleichen Breite. Ab Tiefe 2 sinkt die Akkuratheit mit zunehmender Tiefe der Probleme nur noch leicht. Die Anzahl an Distraktoren hat dann keinen eindeutigen Einfluss mehr auf die Vorhersagen.

Insgesamt geben die Ergebnisse einen Hinweis darauf, dass GPT-2 Probleme damit hat, mehrschrittige Ableitungen korrekt durchzuführen. Diese Vermutung wird in Kapitel 7 anhand vorgegebener Kontexterweiterungen näher untersucht.



(a) Akkuratheit | FQ2 Fantasie-Datensatz

(b) Differenz von 6.1b zu 6.2a

Abbildung 6.2.: Akkuratheit auf *ModifiedChainRuler*-Fantasie-Datensatz nach Tiefe und Breite (a) und Veränderung gegenüber *ModifiedChainRuler*-Datensatz (b)

6.2.2. Auswertung auf dem *ModifiedChainRuler*-Fantasie-Datensatz

Um zu untersuchen, ob das Sprachmodell in der Lage ist, Schlussfolgerungen über unbekannte Prädikate zu ziehen, wird seine Vorhersage auf 10.000 Problemen des *ModifiedChainRuler*-Fantasie-Datensatzes (im Folgenden auch kurz „Fantasie-Datensatz“) evaluiert. Details zur Konstruktion dieses Datensatzes und Beispiele finden sich in Abschnitt 5.6. Als finale Frage wird „`Therefore, we know that`“ verwendet. Abbildung 6.2 zeigt die Ergebnisse der Evaluation sowie die absoluten Veränderungen, die sich gegenüber dem normalen *ModifiedChainRuler*-Datensatz ergeben. Wie man an diesen Grafiken erkennen kann, unterliegen die Vorhersagen auf dem Fantasie-Datensatz stärkeren Schwankungen. Eine mögliche Begründung für diese Beobachtung könnten die kontextualisierten Einbettungen bieten, die GPT-2 für die Neologismen berechnet. Da die erfundenen Worte nicht im Trainingsdatensatz vorkommen, werden sie vom Sprachmodell in mehrere Tokens zerlegt, zu denen jeweils kontextualisierte Einbettungen berechnet werden. Es ist zu erwarten, dass die Repräsentationen zweier Neologismen dadurch stärker voneinander abweichen als die Repräsentationen zweier Prädikate, die das Sprachmodell aus dem Vortraining kennt. GPT-2 verwendet die Repräsentationen der einzelnen Tokens, um sich Informationen über die Textsequenz (wie beispielsweise logische Zusammenhänge zwischen Begriffen) zu erschließen. Die Einbettungen werden mit Informationen angereicherten und schließlich als Grundlage für die Vorhersagen des Sprachmodells genutzt [16][6][41]. Das könnte erklären, wieso die Vorhersagen auf dem Fantasie-Datensatz stärker variieren.

Unterschiede zur Akkuratheit auf dem *ModifiedChainRuler*-Datensatz zeigen sich vor Allem bei Problemen der Tiefe 1, wie sich aus Abbildung 6.2b ablesen lässt. Bei der Interpretation der Korrektklassifizierungsraten ist allerdings zu beachten, dass die Verschlechterungen teilweise mit den notwendigen Unterschieden in der Konstruktion der Datensätze zusammenhängen könnten. Probleme im Fantasie-Datensatz enthalten im

Gegensatz zu *ModifiedChainRuler*-Problemen jeweils eine negierte Aussage (siehe Abschnitt 5.6), wodurch sich die in Abschnitt 5.2 besprochenen Schwierigkeiten ergeben.

Des Weiteren lässt sich anhand der Abbildung 6.2a erkennen, dass die Akkuratheit auf Problemen der Tiefe 1 mit höherer Breite zunimmt. Betrachten wir dieses Phänomen anhand zweier Problembeschreibungen aus dem *ModifiedChainRuler*-Fantasie-Datensatz:

> Problem A ($d = 1, b = 1$):
>
> „`If someone is foaby, then they are not canimery. If someone is trouby, then they are canimery. Maria is foaby.`“
>
> Problem B ($d = 1, b = 5$):
>
> „`If someone is toumy, then they are lanig. Catherine is foaby. If someone is foaby, then they are funely. If someone is maming, then they are soaful. If someone is conill, then they are patouny. If someone is parouly, then they are not funely. If someone is wouring, then they are gleered.`“

Beide Probleme haben die Tiefe 1, es muss also nur eine Ableitung aus dem Fakt vollzogen werden. Ein ausgleichender Distraktor verweist jeweils auf die falsche Alternative („`canimery`“ beziehungsweise „`not funely`“). Problem B enthält zusätzlich vier Distraktoren, deren Prädikate von der Regelkette unabhängig sind. Das Sprachmodell scheint besser in der Lage zu sein, Probleme von Typ B zu lösen als Probleme von Typ A. Eine mögliche Erklärung für dieses Phänomen ist die folgende: Die Distraktoren in Problem B demonstrieren dem Sprachmodell, wie die Neologismen verwendet werden. Anhand des zusätzlichen Kontexts kann das Sprachmodell lernen, die syntaktische Funktion der Neologismen in den Sätzen richtig zu interpretieren. Dadurch ist es besser in der Lage, die ausgedachten Prädikate wie „echte“ Prädikate zu benutzen.

Insgesamt zeigt sich, dass die Korrektklassifizierungsrate von GPT-2 auf dem Fantasie-Datensatz bei geringer Tiefe leicht besser ist als die eines Zufallsklassifikators. Das lässt darauf schließen, dass GPT-2 wenig semantisches Wissen nutzt, um die Aufgaben zu lösen. Es ist also eingeschränkt dazu in der Lage, die unbekannten Prädikate als „Platzhalter“ zu verwenden und logische Schlüsse über sie zu ziehen.

7. Einfluss vorgegebener Problemelaborationen

Die Ergebnisse aus Kapitel 6 zeigen, dass GPT-2 auf den *ModifiedChainRuler*-Datensätzen ohne Finetuning nicht über ausgeprägte Schlussfolgerungsfähigkeiten verfügt. Diese Beobachtung ist konsistent mit anderen Studien, die eine mangelhafte Zero-Shot-Performanz vortrainierter Sprachmodelle bei NLI-Aufgaben berichten [47][7][32]. In diesem Kapitel untersuche ich, welchen Einfluss verschiedene vorgegebene Kontexterweiterungen auf die Performanz haben. Die Aufgabenstellung wird dabei um eine sogenannte *Problemelaboration* erweitert. Dabei handelt es sich um eine problemspezifische Ausführung, die dem Sprachmodell helfen soll, die Aufgabe zu lösen. Die verschiedenen Arten von Elaborationen reduzieren die Komplexität der Probleme in unterschiedlichen Weisen. Die Evaluation dieser Kontexterweiterungen ermöglicht also Rückschlüsse darauf, welche Komponenten der Probleme dem Sprachmodell Schwierigkeiten bereiten.

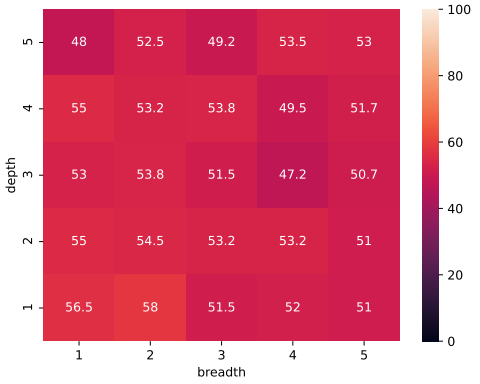
Dynamisch generierte Kontexterweiterungen stellen eine mögliche meta-kognitive Strategie dar, um die Performanz vortrainierter Sprachmodelle in Schlussfolgerungsaufgaben zu verbessern (siehe Abschnitt 3.2 für verwandte Arbeiten). Vorgegebene Problemelaborationen sind in diesem Zusammenhang nützlich, um zu ermitteln, welche Arten von Kontexterweiterungen die Vorhersagen des Modells verbessern. Sie bieten damit Zielvorgaben für die Elaborationen, die das Sprachmodell selbst erzeugen soll, wenn es die meta-kognitive Strategie erfolgreich anwendet.

Das allgemeine Vorgehen in diesem Kapitel ist wie folgt: Für alle Probleme im Datensatz werden zunächst synthetisch aufgabenspezifische Kontexterweiterungen erzeugt. Anschließend werden die Kontexterweiterungen an die Problembeschreibungen angefügt. Zusammen mit der finalen Frage „`Therefore,`“ (FQ1) oder „`Therefore, we know that`“ (FQ2) bilden sie den Kontext, der dem Modell übergeben wird. Anschließend wird seine Vorhersage für die beiden Antwortoptionen überprüft. Der einzige Unterschied zum Vorgehen aus Abschnitt 6.2 liegt also darin, dass zwischen Problembeschreibung und finaler Frage eine Problemelaboration in den Kontext eingefügt wird. Alle Kontexterweiterungen werden auf dem *ModifiedChainRuler*-Datensatz sowie dem *ModifiedChainRuler*-Fantasie-Datensatz mit jeweils 10.000 Problemen evaluiert.

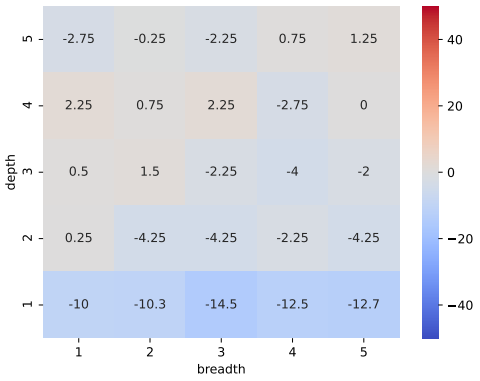
7.1. Plausibilitätsprüfung

Zunächst untersuche ich, ob Kontexterweiterungen überhaupt geeignet sind, die Performanz des Sprachmodells auf dem *ModifiedChainRuler*-Datensatz zu beeinflussen. Dadurch stelle ich sicher, dass das Sprachmodell die Kontexterweiterungen für seine Vorhersage nutzt und der Ansatz dementsprechend sinnvoll ist.

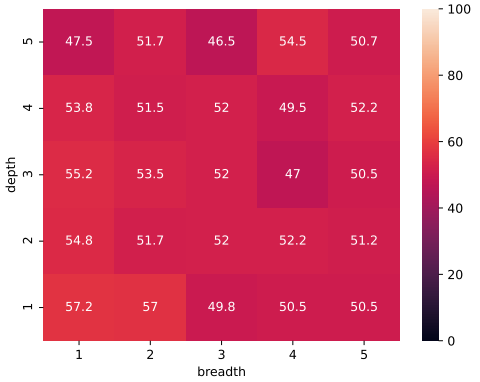
7.1.1. Zufallsantwort



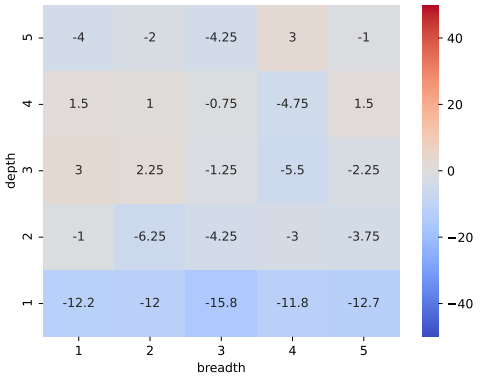
(a) Akkuratheit(Zufalls-Antwort) | FQ1



(b) Δ(keine, Zufalls-Antwort) | FQ1



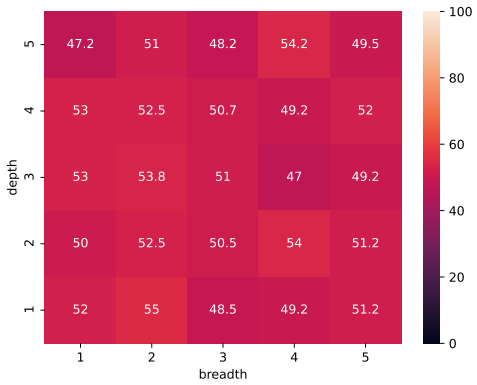
(c) Akkuratheit(Zufalls-Antwort) | FQ2



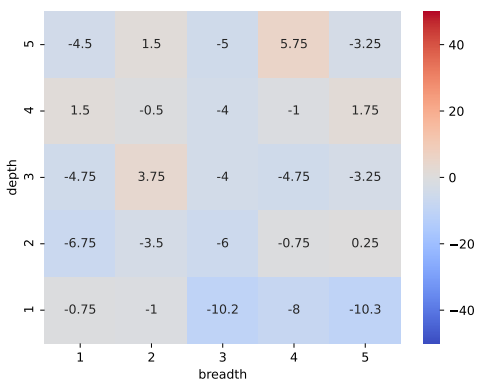
(d) Δ(keine, Zufalls-Antwort) | FQ2

Abbildung 7.1.: Akkuratheit mit Zufalls-Antwort-Elaboration und Differenz zur Akkuratheit ohne Elaboration, jeweils für finale Frage FQ1 und FQ2

Um die erste Kontexterweiterung zu synthetisieren, wird zufällig eine der beiden Antwortmöglichkeiten des jeweiligen Problems ausgewählt und viermal wiederholt. Abbildung 7.1 zeigt die Ergebnisse der Evaluation auf dem *ModifiedChainRuler*-Datensatz für die finalen Fragen „`Therefore,`“ (FQ1) und „`Therefore, we know that`“ (FQ2). Abbildung 7.2 zeigt die Ergebnisse auf dem *ModifiedChainRuler*-Fantasie-Datensatz. Wie sich den Grafiken



(a) Akkuratheit(Zufalls-Antwort) | FQ2 Fantasie-Datensatz



(b) Δ(keine, Zufalls-Antwort) | FQ2 Fantasie-Datensatz

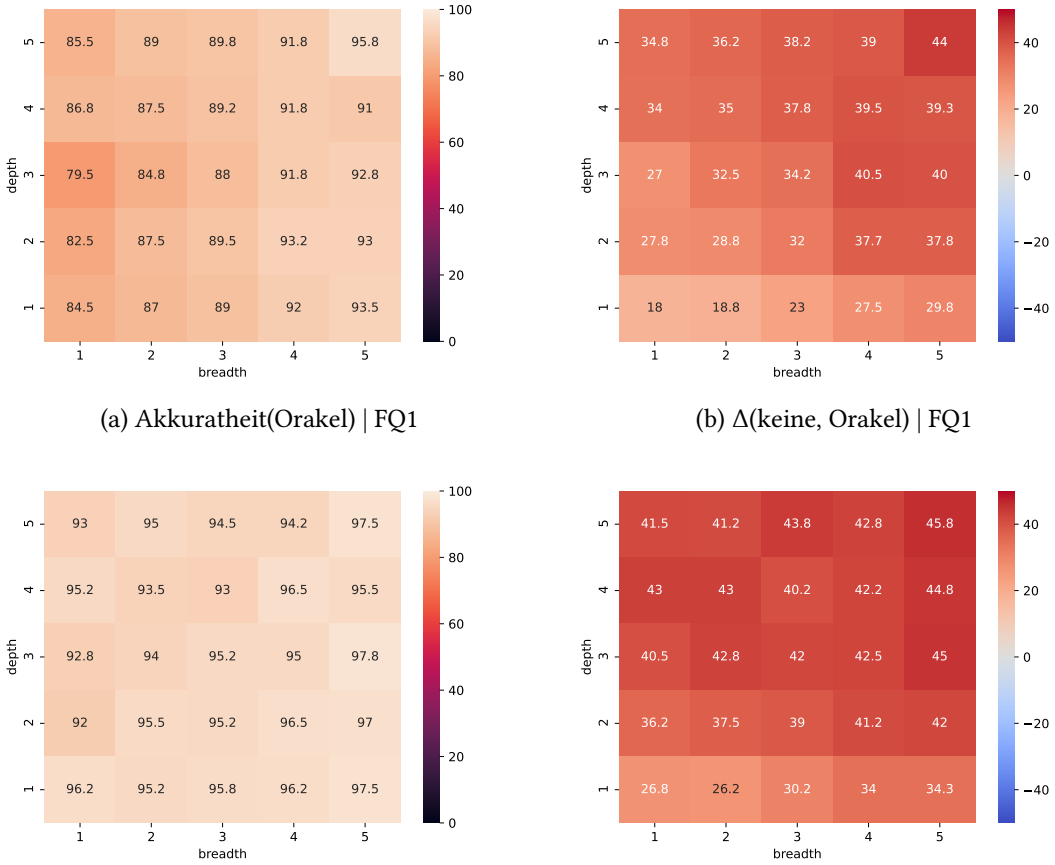
Abbildung 7.2.: Akkuratheit mit Zufalls-Antwort-Elaboration und Differenz zur Akkuratheit ohne Elaboration auf dem *ModifiedChainRuler*-Fantasie-Datensatz

entnehmen lässt, hat das Experiment den erwünschten Effekt: Die Akkuratheit verschiebt sich deutlich in Richtung des Basiswerts von 50%. Das zeigt, dass das Sprachmodell sich stark an der Kontexterweiterung orientiert, wenn es seine Antwortvorhersage trifft.

7.1.2. Orakel

„Orakel-Elaboration“ besteht aus vier Wiederholungen der richtigen Antwort. Erwartungsgemäß verbessert diese Kontexterweiterung die Akkuratheit deutlich (Abbildungen 7.3 für *ModifiedChainRuler*-Datensatz und 7.4 für *ModifiedChainRuler*-Fantasie-Datensatz). Das bestätigt die Beobachtung, dass das Sprachmodell sich bei seiner Vorhersage stark auf die Informationen in der gegebenen Kontexterweiterung stützt. Beim Fantasie-Datensatz ist der Effekt besonders stark ausgeprägt: Die Korrektklassifizierungsrate beträgt hier mindestens 97,5% auf allen Komplexitäten. Wie in Abschnitt 6.2.2 ausgeführt, ist das Sprachmodell auf dem Fantasie-Datensatz weniger gut in der Lage, korrekte Ableitungen aus dem Kontext zu ziehen (siehe Abbildung 6.2b). Mit gegebenen Orakel-Elaborationen übertrifft seine Vorhersage auf dem Fantasie-Datensatz jedoch die Vorhersage auf dem *ModifiedChainRuler*-Datensatz. Aus diesen Beobachtungen lässt sich die Hypothese ableiten, dass das Sprachmodell bei Fantasie-Problemen weniger gefestigtes Wissen über die Zusammenhänge erlangt, die in den Problembeschreibungen erläutert werden. Das würde erklären, wieso es stärker dazu bereit ist, seine Vorhersagen zu ändern, wenn es durch die Kontexterweiterungen zusätzliche Informationen bekommt.

Die Korrektklassifizierungsraten bieten Referenzwerte dafür, wie stark die Verbesserungen sein können, die durch vorgegebene hochqalitative Problemelaborationen erzielt werden können. Interessant sind die Unterschiede zwischen den beiden finalen Fragen auf dem *ModifiedChainRuler*-Datensatz. Wenn die möglichen Antworten mit „`Therefore, we know that`“ (FQ2) eingeleitet werden, kann das Sprachmodell die Kontexterweiterung



(a) Akkuratheit(Orakel) | FQ1

(b) Δ(keine, Orakel) | FQ1

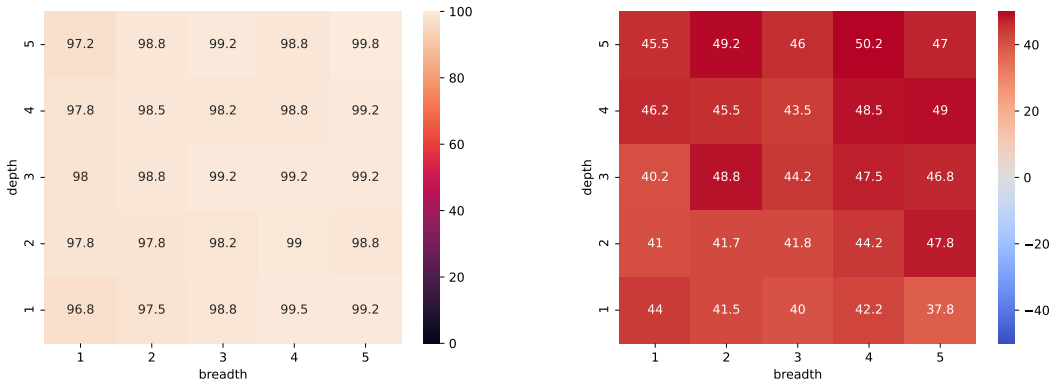
(c) Akkuratheit(Orakel) | FQ2

(d) Δ(keine, Orakel) | FQ2

Abbildung 7.3.: Akkuratheit mit Orakel-Elaboration und Differenz zur Akkuratheit ohne Elaboration

deutlich besser ausnutzen als wenn die Antwortmöglichkeiten mit „`Therefore,`“ (FQ1) beginnen. Der Unterschied in der Korrektklassifizierungsrate beträgt bis zu 13,3 absoluten Prozentpunkten (vgl. Abbildungen 7.3a und 7.3c, Tiefe 3, Breite 1).

Dieser Unterschied erscheint einleuchtend, wenn man sich vergegenwärtigt, dass das Sprachmodell darauf trainiert ist, vorherzusagen, wie eine gegebene Textsequenz weiter verläuft. Es ist plausibel anzunehmen, dass der Trainingsdatensatz Beispiele enthält, in denen nach der Phrase „`we know that`“ eine Information wiederholt wird, die vorher schon im Text aufgetaucht ist. In diesem Fall hätte das Sprachmodell im Vortraining folgende Heuristik erlernt: Um den Verlauf eines Textes nach „`we know that`“ vorherzusagen, ist es hilfreich, den Text vor „`we know that`“ zu betrachten und Aussagen aus diesem Teil zu wiederholen. Wenn man als Mensch versucht, eine Schlussfolgerungsaufgabe zu lösen, kann es eine sinnvolle meta-kognitive Strategie sein, „erst einmal zu schauen, was man schon weiß“. Es ist daher wünschenswert, GPT-2 dazu anzuregen, eine solche Herangehensweise für *ModifiedChainRuler*-Aufgaben nutzen. Fassen wir die bisherigen Beobachtungen über die beiden finalen Fragen zusammen: Im Experiment ohne Kontexterweiterungen (Ab-



(a) Akkuratheit(Orakel) | FQ2 Fantasie-Datensatz

(b) Δ(keine, Orakel) | FQ2 Fantasie-Datensatz

Abbildung 7.4.: Akkuratheit mit Orakel-Elaboration und Differenz zur Akkuratheit ohne Elaboration auf dem *ModifiedChainRuler*-Fantasie-Datensatz

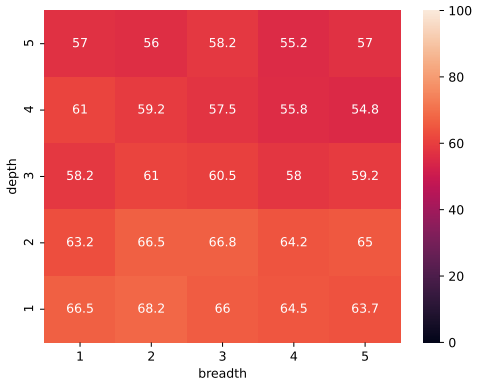
schnitt 6.2) ergeben sich keine signifikanten Unterschiede zwischen den Vorhersagen mit FQ1 und FQ2. Wenn der Kontext hingegen so erweitert wird, dass er die richtige Antwort enthält, ist das Modell mit FQ2 deutlich besser in der Lage, diese Information auszunutzen.

Die Beobachtung könnte einen Hinweis darauf liefern, dass die finale Frage „`Therefore, we know that`“ (FQ2) dem Modell besser signalisiert, dass im Folgenden etwas abgefragt wird, was sich aus den Informationen im Kontext erschließen lässt. In diesem Fall wäre sie besser geeignet, das Sprachmodell dazu anzuregen, die Aufgabe so zu bearbeiten wie durch die Problemstellung beabsichtigt. Die bisherigen Beobachtungen reichen aber noch nicht aus, um die Eignung der beiden finalen Fragen abschließend zu beurteilen. Eine gute Elaboration beinhaltet nämlich nicht zwangsläufig die Konklusion. Es gibt gute Gründe, auch Elaborationen als sinnvoll anzusehen, die beispielsweise nur die relevanten Regeln wiederholen oder Zwischenschlüsse explizieren. Eine finale Frage, die dem Modell *ausschließlich* signalisiert, dass eine Aussage abgefragt wird, die es bereits gesehen hat, wäre daher für die Aufgabenstellung ungeeignet. In Abschnitt 7.3 schließe ich diese Wissenslücke mithilfe geeigneter Experimente.

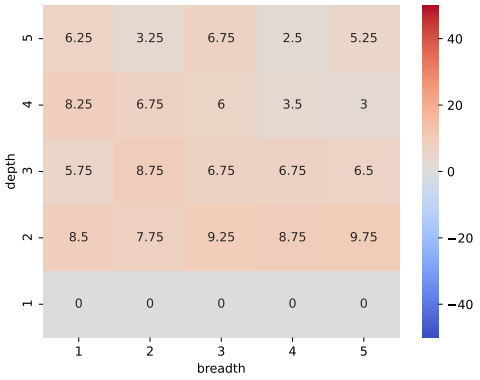
7.2. Kontexterweiterungen mit Zwischenschlüssen

Die Korrektlassifizierungsrate von GPT-2 auf dem *ModifiedChainRuler*-Datensatz fällt von Problemen der Tiefe 1 zu Problemen der Tiefe 2 drastisch ab (siehe Abschnitt 6.2). Um Probleme der Tiefe 1 korrekt zu lösen, müssen keine Zwischenschlüsse gezogen werden. In diesem Abschnitt untersuche ich, welchen Einfluss diese Komponente der *ModifiedChainRuler*-Probleme auf die Performanz des Sprachmodells hat. Dazu synthetisiere ich zu jedem Problem eine Elaboration, die alle möglichen Zwischenschlüsse dieses Problems in der richtigen Reihenfolge enthält. Mit dieser Kontexterweiterung muss bei Proble-

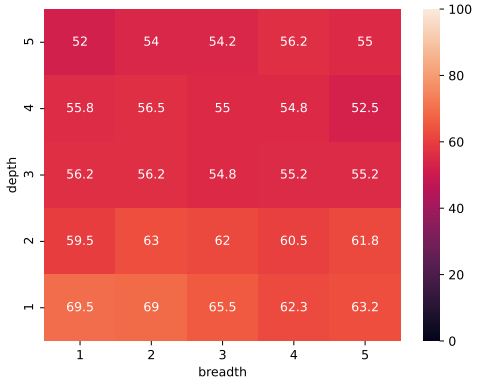
men beliebiger Tiefen nur noch eine Ableitung durchgeführt werden, um zur richtigen Lösung zu gelangen. Probleme mit Zwischenschluss-Elaborationen entsprechen in dieser Komponente also Problemen der Tiefe 1. Diese Überlegung wird in Abschnitt 7.2.1 formalisiert. Abbildung 7.5 und Abbildung 7.6 zeigen die Ergebnisse des Versuchs auf



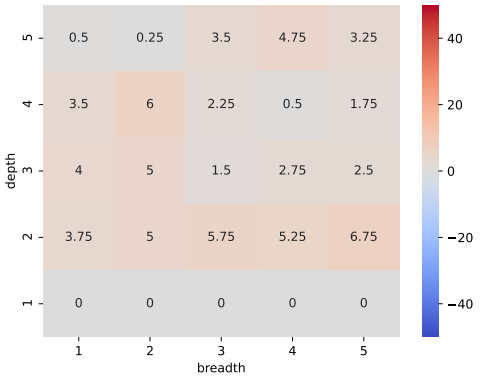
(a) Akkuratheit(Zwischenschlüsse) | FQ1



(b) Δ(keine, Zwischenschlüsse) | FQ1



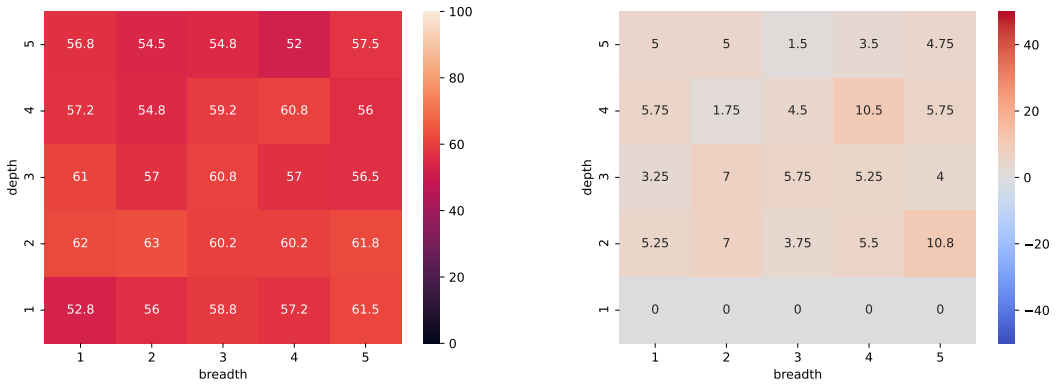
(c) Akkuratheit(Zwischenschlüsse) | FQ2



(d) Δ(keine, Zwischenschlüsse) | FQ2

Abbildung 7.5.: Akkuratheit mit Zwischenschlüsse-Elaboration und Differenz zur Akkuratheit ohne Elaboration

dem *ModifiedChainRuler*- und dem Fantasie-Datensatz. Die Korrektklassifizierungsrate liegt auf dem *ModifiedChainRuler*-Datensatz insgesamt höher aus als auf dem Fantasie-Datensatz. Betrachtet man aber die Veränderungen der Akkuratheit im Vergleich zum Experiment ohne Problemelaboration (Abbildungen 7.5b, 7.5d und 7.6b), so zeigt sich, dass die Kontexterweiterungen auf dem Fantasie-Datensatz einen ähnlich guten Effekt haben wie auf dem *ModifiedChainRuler*-Datensatz. Insgesamt verbessert sich die Performanz des Sprachmodells durch gegebene Zwischenschlüsse deutlich. Da bei Tiefe 1 keine Zwischenschlüsse gezogen werden können, sind die Vorhersagen in dieser Zeile identisch. Ab Tiefe 2 ergeben sich Verbesserungen von bis zu 9,75 absoluten Prozentpunkten auf dem *ModifiedChainRuler*-Datensatz. Das zeigt, dass GPT-2 die gegebenen Zwischenschlüsse nutzt, um die Aufgaben zu lösen.



(a) Akkuratheit(Zwischenschlüsse) | FQ2 Fantasie-Datensatz

(b) Δ(keine, Zwischenschlüsse) | FQ2 Fantasie-Datensatz

Abbildung 7.6.: Akkuratheit mit Zwischenschlüsse-Elaboration und Differenz zur Akkuratheit ohne Elaboration auf dem *ModifiedChainRuler*-Fantasie-Datensatz

Die Unterschiede zwischen Abbildung 7.5b und 7.5d scheinen die in Abschnitt 7.1 formulierte Überlegung zu bestätigen: Die finale Frage „`Therefore, we know that`“ (FQ2) scheint dem Modell zwar besser zu helfen, die Elaboration auszunutzen, wenn die Lösung schon im Kontext enthalten ist – wenn die Lösung aber noch abgeleitet werden muss, eignet sich „`Therefore,`“ (FQ1) besser.

7.2.1. Projektion auf Pseudo-Probleme

Betrachten wir das folgende Problem 7.1, um den Einfluss von Zwischenschluss-Elaborationen zu illustrieren:

Fakt	„`Frances is beautiful.`“
Regelkette	„`If someone is beautiful, then they are early.`“, „`If someone is early, then they are joking.`“, „`If someone is joking, then they are careful.`“, „`If someone is careful, then they are rude.`“, „`If someone is rude, then they are young.`“
Zwischenergebnisse	„`Frances is early.`“, „`Frances is joking.`“, „`Frances is careful.`“, „`Frances is rude.`“
Distraktoren	„`If someone is happy, then they are old.`“
Konklusion	„`Frances is young.`“
Falsche Alternative	„`Frances is old.`“
Tiefe	5
Breite	1

Gruppe	1

Die dazugehörige Problembeschreibung enthält den Fakt, die relevanten Regeln und den Distraktor in zufälliger Reihenfolge. Sie lautet beispielsweise

> „If someone is rude, then they are young. If someone is early, then they are joking. Frances is beautiful. If someone is joking, then they are careful. If someone is happy, then they are old. If someone is careful, then they are rude. If someone is beautiful, then they are early.“

Ein erweiterter Kontext besteht aus der Problembeschreibung, der Problemelaboration und der finalen Frage. Mit der Zwischenschluss-Elaboration und der finalen Frage FQ1 würde sich für das Problem 7.1 der folgende Kontext ergeben:

> „If someone is rude, then they are young. If someone is early, then they are joking. Frances is beautiful. If someone is joking, then they are careful. If someone is happy, then they are old. If someone is careful, then they are rude. If someone is beautiful, then they are early. Frances is early. Frances is joking. Frances is careful. Frances is rude. Therefore,“

Um die Aufgabe anhand dieses Kontexts zu lösen, muss man nur noch den Zwischenschluss „Frances is rude.“ mit der Regel „If someone is rude, then they are young.“ kombinieren und einmal deduktiv ableiten. In dieser Hinsicht entspricht die Aufgabe also einer Aufgabe der Tiefe 1. Verdeutlichen wir uns diesen Zusammenhang, indem wir die Teile des erweiterten Kontexts zu folgendem „Pseudo-Problem“ 7.2 umsortieren:

Pseudo-Fakt	„Frances is rude.“
Pseudo-Regelkette	„If someone is rude, then they are young.“
Ausgl. Distraktor	„If someone is happy, then they are old.“
Pseudo-Distraktoren	„If someone is beautiful, then they are early.“, „If someone is early, then they are joking.“, „If someone is joking, then they are careful.“, „If someone is careful, then they are rude.“
Zusatzfakten	„Frances is early.“, „Frances is joking.“, „Frances is careful.“
Konklusion	„Frances is young.“
Falsche Alternative	„Frances is old.“
Pseudo-Tiefe	1
Pseudo-Breite	5

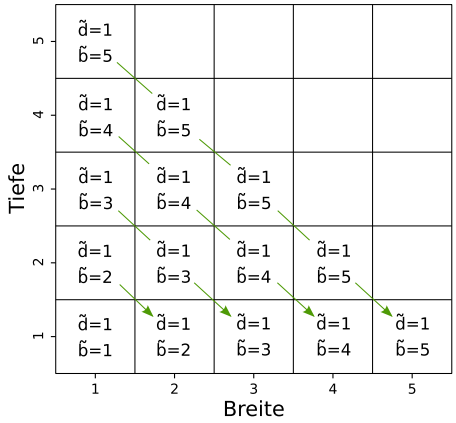


Abbildung 7.7.: $\varphi(d, b)$ für Tiefe d, Breite b

Dieses „Pseudo-Problem“ unterscheidet sich in zwei Punkten von einem echten Problem der Tiefe 1 und Breite 5:

- Der Kontext enthält Zusatzfakten über das Subjekt, die in echten *ModifiedChainRuler*-Problembeschreibungen nicht enthalten sind. Man könnte die zusätzlichen Fakten als eine Kontexterweiterung betrachten.
- Die Pseudo-Distraktoren unterscheiden sich von echten Distraktoren dadurch, dass sie sich mithilfe der zusätzlichen Fakten auf das Subjekt anwenden lassen. Im Gegensatz zu echten Distraktoren bilden sie alternative Pfade, über die man ebenfalls zur Konklusion gelangen kann.

7.2.2. Formalisierung

Betrachten wir die Aufgaben mit gegebenen Zwischenschluss-Elaborationen nun wie in Abschnitt 7.2.1 erläutert als Pseudo-Probleme der Tiefe 1 – analog zur Projektion von Problem 7.1 auf Pseudo-Problem 7.2. Probleme der Tiefe d, Breite b mit Zwischenschluss-Elaborationen lassen sich zu Pseudo-Problemen der Pseudo-Tiefe $\tilde{d}$ und Pseudo-Breite $\tilde{b}$ umformulieren, wenn $d + b \leq 6$ gilt (siehe Abbildung 7.7). Die Pseudo-Komplexität ergibt sich durch $\varphi(d, b) = (1, d + b - 1)$ für Probleme mit $(d + b) \leq 6$. Probleme der Tiefe 1 werden trivialerweise auf sich selbst projiziert.

Ein Problem der Tiefe 3, Breite 2 mit Zwischenschluss-Elaboration lässt sich also beispielsweise zu einem Pseudo-Problem der Pseudo-Tiefe 1, Pseudo-Breite 4 umformulieren (siehe Illustration 7.8). Wie bereits erläutert, ist der einzige Unterschied zwischen so konstruierten Pseudo-Problemen der Pseudo-Komplexität $\tilde{d}, \tilde{b}$ und echten *ModifiedChainRuler*-Problemen dieser Komplexität, dass die Pseudo-Probleme zusätzliche Aussagen über das Subjekt enthalten können, die zusammen mit den Pseudo-Distraktoren wei-

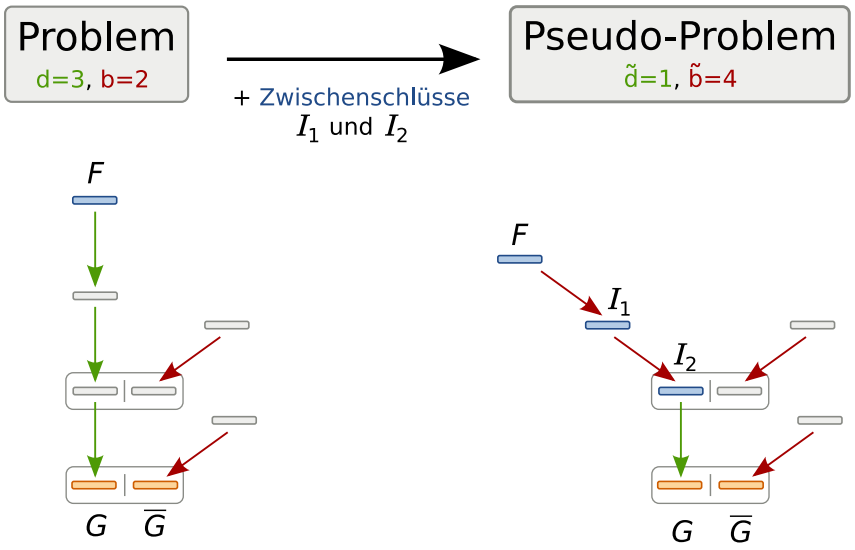


Abbildung 7.8.: Projektion eines Problems der Komplexität $d = 3, b = 2$ auf ein Pseudo-Problem der Komplexität $\tilde{d} = 1, \tilde{b} = 4$

tere Ableitungen über das Subjekt ermöglichen. Diese Ableitungen bilden „Umwege“, über die man ebenfalls zur Konklusion gelangt. Je höher die echte Tiefe der Probleme ist, desto mehr solcher Verknüpfungen sind im Kontext enthalten (siehe Schaubild 7.9). Diesen Überlegungen entsprechend können wir die zusätzlichen Informationen als eine Kontexterweiterung mit $d - 1$ Aussagen über das Subjekt interpretieren. Sie verknüpfen das Subjekt mit zusätzlichen Prädikaten und mit den Pseudo-Distraktoren. Mithilfe der Funktion φ (siehe Tabelle 7.7) sind wir in der Lage, den Einfluss dieser zusätzlichen Informationen zu messen: Wir betrachten Probleme der Komplexität d, b als Pseudo-Probleme der Komplexität $\varphi(d, b) = (\tilde{d}, \tilde{b})$, deren Kontext im Vergleich zu echten Problemen der Komplexität $\tilde{d}, \tilde{b}$ um $(d - 1)$ Aussagen über das Subjekt erweitert ist. Dann vergleichen wir die Korrektklassifizierungsraten auf Pseudo-Problemen der Komplexität $\tilde{d}, \tilde{b}$ mit den Korrektklassifizierungsraten auf echten Problemen dieser Komplexität.

Sei $\alpha(d, b)$ die Akkuratheit auf Problemen der Tiefe d und Breite b und $\tilde{\alpha}(d, b)$ die Akkuratheit auf Problemen der selben Komplexität mit Zwischenschluss-Elaborationen. Für $d = 1$ gilt im Übrigen $\tilde{\alpha}(d, b) = \alpha(d, b)$, da zu diesen Problemen keine Zwischenschlüsse existieren. Sei nun $(\tilde{d}, \tilde{b}) := \varphi(d, b) := (1, d + b - 1)$, falls $(d + b) \leq 6$. Damit lässt sich $\tilde{\alpha}(d, b)$ als eine Akkuratheit auf Pseudo-Problemen der Komplexität $\tilde{d}, \tilde{b}$ interpretieren. Die Akkuratheit auf echten Problemen dieser Komplexität ergibt sich aus $\alpha(\tilde{d}, \tilde{b})$. Die gesuchte Differenz lässt sich also durch

$$\Delta_\varphi(d, b) := \tilde{\alpha}(d, b) - \alpha(\tilde{d}, \tilde{b})$$

ermitteln. Da Δ_φ misst, welche Auswirkungen die zusätzlichen Verknüpfungen auf die Akkuratheit haben, nenne ich es *Maß der Verknüpfungswirkung*.

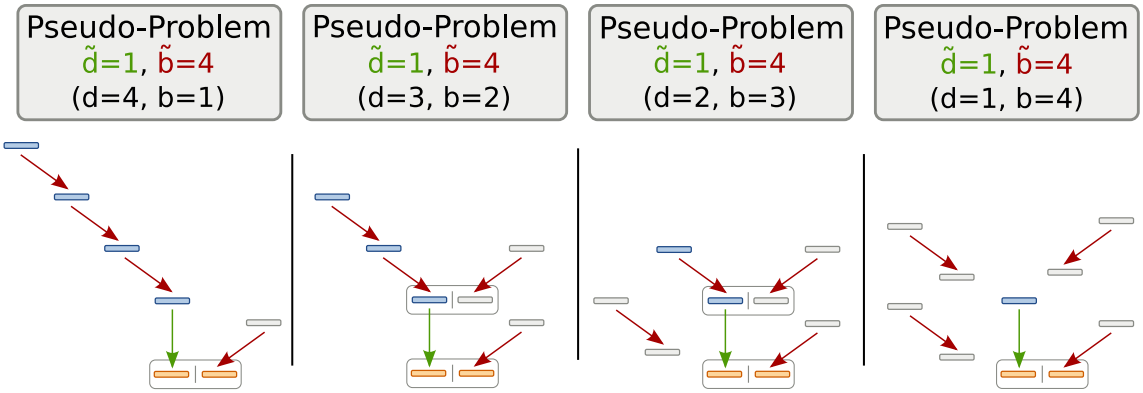
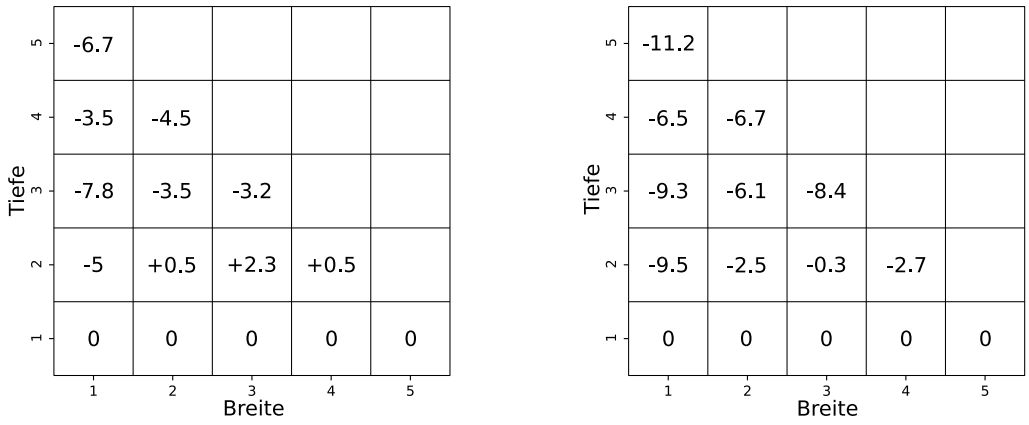


Abbildung 7.9.: Vergleich zwischen verschiedenen Problemen der gleichen Pseudo-Komplexität hinsichtlich ihrer Verknüpftheit. Im Kontext enthaltene Aussagen sind blau markiert.

Betrachten wir als Beispiel *ModifiedChainRuler*-Probleme der Komplexität $d = 3, b = 2$ mit Zwischenschluss-Elaborationen. Die Korrektklassifizierungsrate $\tilde{\alpha}(d, b)$ lässt sich aus den Abbildungen 7.5a (FQ1), beziehungsweise 7.5c (FQ2) ablesen. Sie liegt bei $\tilde{\alpha}(3, 2) = 61$ für FQ1 und bei $\tilde{\alpha}(3, 2) = 56,2$ für FQ2. Es gilt $\varphi(3, 2) = (1, 4)$. Die mit Zwischenschlüssen erweiterten Probleme lassen sich also auf Pseudo-Probleme der Pseudo-Komplexität $\tilde{d} = 1$, $\tilde{b} = 4$ projizieren. Wir ermitteln zum Vergleich nun die Korrektklassifizierungsrate auf echten Problemen dieser Komplexität: $\alpha(\tilde{d}, \tilde{b}) = \alpha(1, 4)$. Sie liegt bei $\alpha(1, 4) = 64,5$ für FQ1 und $\alpha(1, 4) = 62,3$ für FQ2. Daraus folgt $\Delta_\varphi(d, b) = -3,5$ für FQ1 und $\Delta_\varphi(d, b) = -6,1$ für FQ2. Die betrachteten Pseudo-Probleme der Pseudo-Komplexität $\tilde{d} = 1, \tilde{b} = 4$ enthalten im Vergleich zu echten Problemen dieser Komplexität zwei zusätzliche Aussagen über das Subjekt (siehe auch Schaubild 7.9). Wir haben gezeigt, dass die beiden zusätzlichen Aussagen und die Informationen, die sich mithilfe der Distraktoren daraus ableiten lassen, in diesem Beispiel zu einer deutlichen Verschlechterung der Vorhersage führen.

7.2.3. Evaluation der Verknüpfungswirkung auf dem *ModifiedChainRuler*-Datensatz

Die Werte für das Maß der Verknüpfungswirkung Δ_φ auf dem *ModifiedChainRuler*-Datensatz sind in den Tabellen 7.10a (FQ1) und 7.10b (FQ2) aufgeführt. Aus den Tabellen lässt sich erkennen, dass zwei oder mehr zusätzliche Fakten über das Subjekt (echte Tiefe 3) die Performanz von GPT-2 stark negativ beeinflussen. Eine mögliche Erklärung für diese Verschlechterungen könnte in der Weise liegen, wie GPT-2 die Informationen aus dem Kontext verarbeitet. Das Modell berechnet mithilfe einer Methode namens *masked self-attention* schrittweise kontextualisierte Repräsentationen der Eingabe-Tokens. Die Repräsentationen eines Tokens v_i enthalten gewichtete Informationen über v_i und alle Tokens $v_1, ..., v_{i-1}$, die diesem Token vorausgehen. Wie die Informationen in einem gegebenen Kontext gewichtet werden, bestimmt das Sprachmodell anhand der über 117 Milliarden Parameter, die es im Vortraining erlernt hat [30][16][41]. Betrachten wir nun den Kontext des Problems 7.2:



(a) Δ_φ | FQ1 (b) Δ_φ | FQ2

Abbildung 7.10.: Auswertung von $\Delta_\varphi(d, b)$ für Tiefe d und Breite b mit FQ1 und FQ2 auf dem *ModifiedChainRuler*-Datensatz

> „`If someone is rude, then they are young. If someone is early, then they are joking. Frances is beautiful. If someone is joking, then they are careful. If someone is happy, then they are old. If someone is careful, then they are rude. If someone is beautiful, then they are early. Frances is early. Frances is joking. Frances is careful. Frances is rude. Therefore,`“

Um die Warscheinlichkeiten für die beiden Antwortoptionen „`Frances is young.`“ und „`Frances is old.`“ auf Grundlage dieses Kontexts vorherzusagen, muss das Sprachmodell kontextualisierte Einbettungen für die Tokens in den jeweiligen Sequenzen berechnen. Die Einbettungen des Tokens für „`Frances`“ in den Antwortmöglichkeiten enthalten gewichtete Informationen über den kompletten vorhergehenden Kontext. Wir können vermuten, dass das Sprachmodell alle Informationen, die bisher über das Subjekt „`Frances`“ genannt wurden, als relevant gewichtet und in die Repräsentationen kodiert. Dazu gehören die Prädikate „`beautiful`“, „`early`“, „`joking`“ „`careful`“ und „`rude`“. Um die Inhalte dieser Begriffe in eine Einbettung zu integrieren, benutzt das Sprachmodell unter Anderem die Repräsentationen, die es zuvor bereits für diese Prädikate berechnet hat. Die Repräsentationen von „`beautiful`“ enthalten aber mutmaßlich wiederum Informationen darüber, dass dieser Begriff im Zusammenhang mit „`early`“ steht, und so weiter. Im Vergleich dazu ist die Gewichtung des Kontexts bei einem Problem, das *tatsächlich* die Tiefe 1 und Breite 5 hat, sehr viel eindeutiger: Das Subjekt wird bei solchen Problemen nur in Zusammenhang mit einem einzigen Prädikat genannt, das selbst wiederum in Zusammenhang mit nur einem anderen Prädikat vorkommt. Die Ergebnisse für Δ_φ in den Tabellen 7.10a und 7.10b zeigen, dass zwei oder mehr zusätzliche Aussagen über das Subjekt die Performanz von GPT-2 auf dem *ModifiedChainRuler*-Datensatz deutlich verschlechtern. Das legt die Vermutung nahe, dass GPT-2 Informationen, die zur Problemlösung benötigt werden, „verliert“, wenn es zu viele Informationen über das Subjekt kodieren muss.

Abbildung 7.11.: Auswertung von $\Delta_\varphi(d, b)$ für Tiefe d und Breite b auf dem Fantasie-Datensatz mit FQ2

7.2.4. Evaluation der Verknüpfungswirkung auf dem Fantasie-Datensatz

Tabelle 7.11 zeigt die Auswertung von Δ_φ auf dem *ModifiedChainRuler*-Fantasie-Datensatz. Bei Problemen aus dem Fantasie-Datensatz haben zusätzliche Informationen über das Subjekt einen gemischten Einfluss. Wie man Tabelle 7.11 entnehmen kann, verbessern die zusätzlichen Fakten die Akkuratheit bei Problemen mit geringer Komlexität. Das bestätigt die Hypothese, die ich in Abschnitt 6.2.2 formuliert habe: Das Sprachmodell scheint bei Fantasie-Problemen mit geringer Komplexität nicht in der Lage zu sein, die Funktion der erfundenen Prädikate richtig zu interpretieren. Erweitert man den Kontext aber durch zusätzliche Aussagen, die ebenfalls Fantasieworte enthalten, so kann das Sprachmodell diese nutzen, um zu lernen, wie die Neologismen richtig verwendet werden. Bei Problembeschreibungen, die ohne Kontexterweiterung aus wenigen Sätzen bestehen, führen die zusätzlichen Fakten daher zu einer deutlichen Verbesserung der Akkuratheit. Der Tabelle lässt sich entnehmen, dass die zusätzlichen Fakten bei Problemen der Pseudo-Breiten $\tilde{b} \leq 3$ Verbesserungen von bis zu 6 absoluten Prozentpunkten bewirken. Enthalten die Problembeschreibungen hingegen schon eine ausreichende Anzahl an Sätzen, tritt der in Abschnitt 7.2.3 beschriebene Effekt ein und die Vorhersagen verschlechtern sich. Deutliche Verschlechterungen sind bei Problemen der Pseudo-Breite $\tilde{b} = 5$ zu erkennen. Dabei handelt es sich um Probleme, die ohne Kontexterweiterung schon sieben Sätze enthalten. Anhand der Probleme mit Pseudo-Breite $\tilde{b} = 4$ (Problembeschreibung aus sechs Sätzen ohne Kontexterweiterung) lässt sich gut erkennen, wie die beiden Effekte sich überlagern: Bei Problemen der Komplexität $d = 2, b = 3$ führt der positive Einfluss des zusätzlichen Satzes noch zu einer Verbesserung von 3 absoluten Prozentpunkten. Sobald jedoch weitere Aussagen hinzukommen (Komplexitäten $d = 3, b = 2$ und $d = 4, b = 1$), gleichen die zusätzlichen Informationen über das Subjekt den positiven Effekt aus und führen zu einer leichten Verschlechterung der Akkuratheit.

7.2.5. Zusammenfassung

Die Performanz von GPT-2 auf dem *ModifiedChainRuler*-Datensatz verbessert sich merklich, wenn es zu den Problemen Kontexterweiterungen gegeben bekommt, die alle möglichen Zwischenschlüsse enthalten. Das Sprachmodell bezieht die Zwischenschlüsse also wie vorgesehen in seine Vorhersagen ein. In diesem Experiment zeigen sich leichte Hinweise darauf, dass die finale Frage „`Therefore,`“ dem Modell besser signalisiert, dass es noch eine Ableitung durchführen muss, um zur Konklusion zu gelangen. Vergleicht man Probleme, bei denen nur eine Regelanwendung nötig ist, deren Kontext aber zusätzliche Informationen über das Subjekt enthält, mit Problemen der Tiefe 1, welche die gleiche Anzahl an Regeln enthalten, so zeigt sich

> (i) eine deutlich geringere Akkuratheit ab zwei zusätzlichen Informationen auf *ModifiedChainRuler*-Problemen und
>
> (ii) eine höhere Akkuratheit bei Problemen von geringer Komplexität und eine geringere Akkuratheit bei Problemen von hoher Komplexität auf dem Fantasie-Datensatz.

7.3. Vorwärts- und rückwärtsgerichtete Beweise

In diesem Abschnitt untersuche ich, wie gut GPT-2 in der Lage ist, zwei Arten von hochqualitativen Problemelaborationen für seine Vorhersage zu nutzen: Vorwärts- und rückwärtsgerichtete Beweise. Diese Idee ist inspiriert von der Arbeit von Gontier et al. [14]. Die Autoren zeigen anhand einer ähnlichen NLI-Aufgabe, dass Sprachmodelle nach entsprechendem Training besser in der Lage sind, vorwärtsgerichtete Beweise zu generieren, aber rückwärtsgerichtete Beweise leichter für ihre Vorhersagen nutzen können (siehe Abschnitt 3.2). In diesem Abschnitt untersuche ich, ob die Beobachtung von Gontier et al. sich bei meinem Versuchsaufbau bestätigt: Verbessern gegebene rückwärtsgerichtete Beweise die Zero-Shot-Performanz von GPT-2 auf dem *ModifiedChainRuler*-Datensatz stärker als gegebene vorwärtsgerichtete Beweise? Ich untersuche zunächst, wie stark Kontexterweiterungen durch Vorwärts- und Rückwärtsbeweise die Performanz beeinflussen. Anschließend überprüfe ich, welchen Anteil die verschiedenen Komponenten der Problemelaborationen an dem beobachteten Effekt haben.

Bei vorwärtsgerichteten Beweisen wird die richtige Lösung abgeleitet, indem man ausgehend vom Fakt schrittweise die Regelkette in der richtigen Reihenfolge anwendet, um alle Zwischenschlüsse zu ziehen und schließlich zur richtigen Lösung zu gelangen. Das entspricht der Heuristik, sich einen unbekannten Sachverhalt zu erschließen, indem man versucht, aus seinem bisherigen Wissen so viele neue Informationen abzuleiten wie möglich. Bei rückwärtsgerichteten Beweisen wird stattdessen zunächst die Lösung genannt und durch umgekehrtes Verketten der Regelkette begründet. Dieses Vorgehen entspricht der meta-kognitiven Strategie, testweise eine Antwortmöglichkeit als wahr anzunehmen und auszuprobieren, ob sie sich mit dem bisherigen Wissensstand begründen lässt.

Betrachten wir wieder Beispiel 5.2:

Fakt	„Maria is charming.“
Regelkette	„If someone is charming, then they are careless.“, „If someone is careless, then they are happy.“, „If someone is happy, then they are awake.“
Zwischenergebnisse	„Maria is careless.“, „Maria is happy.“
Distraktoren	„If someone is young, then they are sad.“, „If someone is serious, then they are sleeping.“
Konklusion	„Maria is awake.“
Falsche Alternative	„Maria is sleeping.“
Tiefe	3
Breite	2
Gruppe	1

Der vorwärtsgerichtete Beweis für dieses Problem lautet folgendermaßen:

> „Maria is charming. If someone is charming, then they are careless. Therefore, Maria is careless. If someone is careless, then they are happy. Therefore, Maria is happy. If someone is happy, then they are awake. Therefore, Maria is awake.“

Der rückwärtsgerichtete Beweis lautet:

> „Maria is awake. That’s because if someone is happy, then they are awake. Maria is happy. That’s because if someone is careless, then they are happy. Maria is careless. That’s because if someone is charming, then they are careless. Maria is charming.“

Ich untersuche den Einfluss vorwärts- und rückwärtsgerichteter Beweise auf die Vorhersage, indem ich synthetische Beweise zu allen Problemen der *ModifiedChainRuler*-Datensätze erstelle und als Kontexterweiterungen zu den jeweiligen Problemen hinzufüge. Den Einfluss, den diese hochqualitativen Elaborationen auf die Vorhersage haben, messe ich wie bisher durch die Differenz der Korrektklassifizierungsrate im Vergleich zur Korrektklassifizierungsrate ohne Kontexterweiterung. Betrachten wir zunächst, wodurch der Einfluss solcher vorgegebenen Beweise zustande kommen könnte:

1. Das Sprachmodell vollzieht die Beweisstruktur nach und festigt dadurch sein „Wissen“ über die richtige Antwortoption
2. Das Sprachmodell memorisiert die in Beweis enthaltene Lösung (hier „Maria is awake.“)

3. Das Sprachmodell nutzt die gleiche Heuristik wie beim *ChainRuler*-Datensatz

Um genauer zu verstehen, wie vorgegebene Problemelaborationen die Vorhersage von GPT-2 beeinflussen, möchte ich den Anteil dieser drei Komponenten am Einfluss der Kontexterweiterungen isolieren. Ich führe dazu die folgenden zusätzlichen Experimente durch:

Einerseits teste ich maskierte Varianten der Vorwärts- und Rückwärtsbeweise. Anlass für diesen Versuch ist die Überlegung, dass sowohl vorwärts- als auch rückwärtsgerichtete Beweise die richtige Lösung enthalten. Wie die Experimente in Abschnitt 7.1.2 zeigen, haben Kontexterweiterungen, welche die Lösung enthalten, einen starken Einfluss auf die Akkuratheit. Bei vorwärtsgerichteten Beweisen steht die Lösung im letzten, bei rückwärtsgerichteten Beweisen im ersten Satz. In maskierten Beweisen wird dieser Satz durch einen Satz ersetzt, der signalisiert, dass an dieser Stelle etwas stehen müsste, aber das Lösungsprädikat nicht enthält. Nach einem Versuch mit mehreren manuell erstellten Sequenzen habe ich die Sequenz „`(...)`“ ausgewählt, um das Lösungsprädikat zu ersetzen. Im obigen Problem wird die im Beweis enthaltene Lösung also beispielsweise durch „`Maria is (...).`“ ersetzt (Abbildung 7.12b und 7.12f). Indem ich den Unterschied zwischen dem Einfluss eines Beweises und dem Einfluss des maskierten Beweises auf die Akkuratheit messe, isoliere ich den Effekt, der auf das Vorhandensein der richtigen Lösung im Kontext zurückzuführen ist.

Andererseits teste ich ausgeglichene Varianten aller Beweise. Die Probleme der *Modified-ChainRuler*-Datensätze sind hinsichtlich der Antwort-Prädikate ausgeglichen: Durch die Konstruktion der Distraktoren enthalten die Problembeschreibungen genau einmal das Lösungsprädikat und einmal das Prädikat der falschen Alternative. GPT-2 kann die jeweiligen Häufigkeiten der Prädikate im Kontext also nicht nutzen, um sich für eine der Antwortmöglichkeiten zu entscheiden. Erweitert man den Kontext allerdings durch einen Vorwärts- oder Rückwärtsbeweis, so ist er nicht mehr ausgeglichen. Die Problembeschreibung enthält beide Antwort-Prädikate genau einmal. Im Beweis kommt das Lösungsprädikat zweimal vor, einmal in der letzten Regel (im Beispiel „`If someone is happy, then they are awake.`“) und einmal in der enthaltenen Konklusion („`Maria is awake.`“). Das Lösungsprädikat kommt im Kontext also insgesamt dreimal vor, das Prädikat der falschen Alternative aber nur einmal. Wenn das Sprachmodell seine Heuristik einsetzt, hat es demzufolge gute Chancen, die richtige Lösung zu finden. Aus diesem Grund erstelle ich Varianten aller Beweise, denen der ausgleichende Distraktor vorangestellt ist. Im Beispiel lautet dieser Distraktor „`If someone is serious, then they are sleeping.`“. Damit wird der statistische Effekt der Regel „`If someone is happy, then they are awake.`“ ausgeglichen, ohne den Beweis zu unterbrechen. Die Lösung ist in ausgeglichenen Beweisen aber weiterhin enthalten (siehe Abbildung 7.12c und 7.12g). Vollständig ausgeglichen sind hingegen die ausgeglichenen Varianten der maskierten Beweise. Darin kommt sowohl das Lösungsprädikat als auch das Prädikat der falschen Alternative genau zweimal vor (Abbildung 7.12d und 7.12h). Der Einfluss, der darauf entfällt, dass das Sprachmodell die einfache Heuristik verwendet, lässt sich also messen, indem man den Unterschied zwischen dem Einfluss maskierter Beweise und dem Einfluss maskierter und ausgeglichener Beweise auf die Akkuratheit misst.

(a) Vorwärtsbeweis

(b) Vorwärtsbeweis maskiert

(c) Vorwärtsbeweis ausgeglichen

(d) Vorwärtsbeweis maskiert und ausgeglichen

(e) Rückwärtsbeweis

(f) Rückwärtsbeweis maskiert

(g) Rückwärtsbeweis ausgeglichen

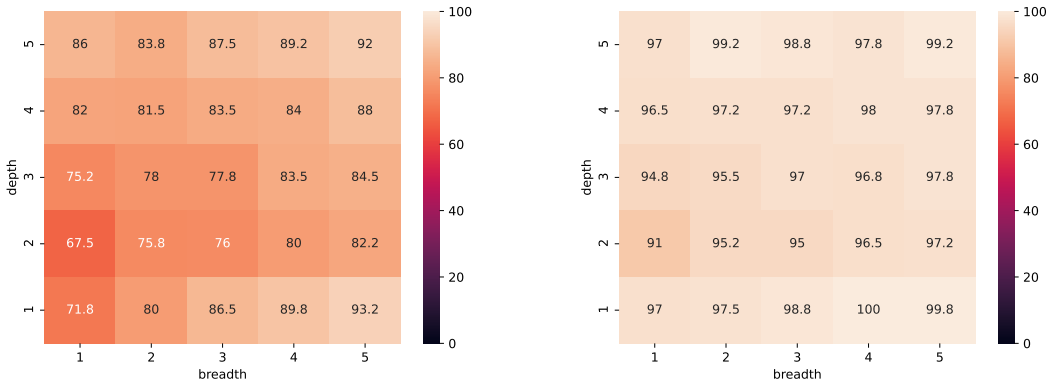
(h) Rückwärtsbeweis maskiert und ausgeglichen

Abbildung 7.12.: Übersicht über die verschiedenen Varianten der Beweise

In Schaubild 7.12 ist eine Übersicht über die verschiedenen Varianten der Beweise zu sehen. Einflüsse der Vorwärts- und Rückwärtsbeweise, die nicht auf die einfache Heuristik oder die enthaltene Lösung zurückzuführen sind, werte ich als erwünschte Effekte. Sie lassen sich darauf zurückführen, dass das Sprachmodell die Struktur der Beweise generalisiert und die aufgeführten Argumente ausnutzt, um die Konklusion abzuleiten. Ich messe diese erwünschten Effekte durch den Einfluss maskierter und ausgeglichener Beweise auf die Akkuratheit.

7.3.1. Evaluation vorwärtsgerichteter Beweise

Als Erstes untersuchen wir Kontexterweiterungen mit vorwärtsgerichteten Beweisen. Abbildung 7.13 zeigt die Akkuratheit von GPT-2 auf dem *ModifiedChainRuler*-Datensatz mit dieser Kontexterweiterung. Wie sich am Unterschied zwischen den beiden Abbildungen erkennen lässt, gehen die Korrektklassifizierungsraten mit der finalen Frage „`Therefore,`“ (Abbildung 7.13a) und „`Therefore, we know that`“ (Abbildung 7.13b) stark auseinander.



(a) Akkuratheit(Vorwärtsbeweis) | FQ1

(b) Akkuratheit(Vorwärtsbeweis) | FQ2

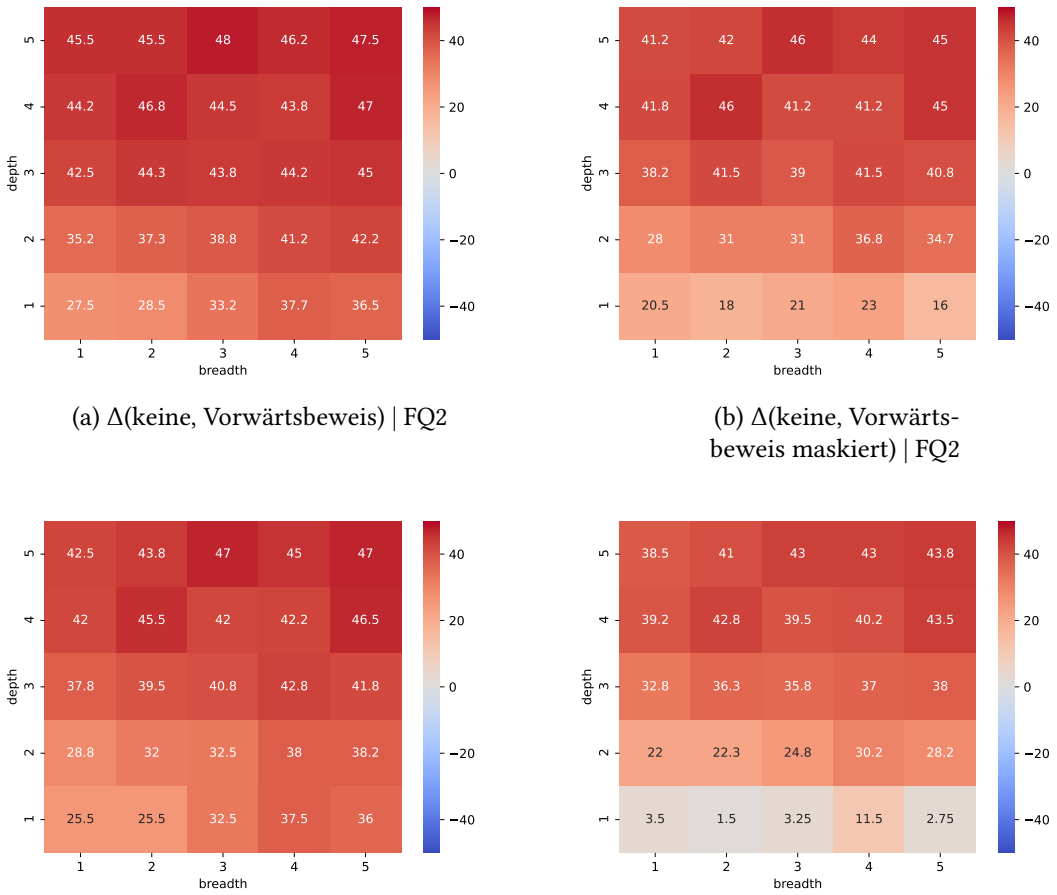
Abbildung 7.13.: Akkuratheit mit Kontexterweiterungen durch vorwärtsgerichtete Beweise

Ich betrachte daher zunächst Vorhersagen, die mit der finalen Frage „`Therefore, we know that`“ (FQ2) erstellt werden und gehe anschließend auf die Besonderheiten der finalen Frage „`Therefore,`“ (FQ1) ein.

7.3.1.1. Evaluation vorwärtsgerichteter Beweise mit FQ2

Abbildung 7.13b zeigt die Korrektklassifizierungsrate bei Kontexterweiterungen mit vorwärtsgerichteten Beweisen und finaler Frage FQ2. Wie wir dieser Grafik entnehmen können, ist die Akkuratheit bei diesem Versuchsaufbau für Probleme aller Komplexitäten extrem gut – auf einer Tiefe und Breite sogar 100%. Die Kontexterweiterungen mit Vorwärtsbeweisen funktionieren also noch besser als Kontexterweiterungen mit Orakel-Elaborationen (vgl. Abb. 7.3a), in denen die richtige Lösung viermal wiederholt wird. Ich möchte diese Verbesserungen im Folgenden auf die einzelnen Komponenten der Beweise zurückführen. Abbildung 7.14a zeigt die Verbesserung der Korrektklassifizierungsrate im Vergleich zum Experiment ohne Kontexterweiterung. Die Verbesserungen, die durch gegebene Vorwärtsbeweise mit maskierter Lösung auftreten, sind in Abbildung 7.14b zu sehen und die Abbildungen 7.14c und 7.14d zeigen die Verbesserungen, die mit ausgeglichenen Varianten dieser beiden Beweise erzielt werden.

Wir können daraus unter Anderem die folgenden Beobachtungen ablesen: Maskieren der Lösung hat auf Problemen mit geringer Tiefe erwartungsgemäß einen deutlichen Einfluss auf die Akkuratheit. Gleicht man zusätzlich noch den statistischen Effekt davon aus, dass das Lösungsprädikat im Beweis wiederholt wird, verbessert sich die Vorhersage bei Tiefe 1 nur noch geringfügig. Das ist nicht überraschend. Ein maskierter und ausgeglichener Vorwärtsbeweis der Tiefe 1 hat die Form: „`If someone is [`X`], then they are [`$\overline{G}$`]. [Subjekt] is [`F`]. If someone is [`F`], then they are [`G`]. Therefore, [Subjekt] is (...).`“ Das Sprachmodell bekommt also keine zusätzlichen Informationen. Dass sich die Akkuratheit auf Problemen der Tiefe 1 durch maskierte und ausgeglichene Vorwärtsbeweise trotzdem noch verbessert, lässt sich damit erklären, dass das Sprachmodell den Fakt



(a) Δ(keine, Vorwärtsbeweis) | FQ2

(b) Δ(keine, Vorwärtsbeweis maskiert) | FQ2

(c) Δ(keine, Vorwärtsbeweis ausgeglichen) | FQ2

(d) Δ(keine, Vorwärtsbeweis maskiert und ausgeglichen) | FQ2

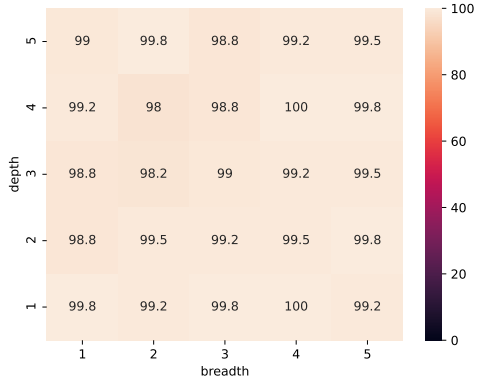
Abbildung 7.14.: Differenz zwischen Akkuratheit mit Elaboration und Basis-Akkuratheit mit FQ2, jeweils für folgende Elaborationen: (a) Vorwärtsbeweis, (b) maskierter Vorwärtsbeweis, (c) ausgeglichener Vorwärtsbeweis, (d) maskierter und ausgeglichener Vorwärtsbeweis

und die relevante Regel nacheinander in der richtigen Reihenfolge präsentiert bekommt. Hier zeigt sich eine Besonderheit, die vermutlich durch die Unidirektionalität von GPT-2 entsteht. Sie lässt sich mit der folgenden Hypothese erklären: Wenn die Regeln in der richtigen Reihenfolge angeordnet sind, kann das Sprachmodell sich die Zusammenhänge besser schrittweise erschließen. In dem Moment, in dem GPT-2 das Prädikat $[F]$ in der Regel verarbeitet, hat es die Information, dass „`[Subjekt] is` $[F]$“ gilt, schon verarbeitet. Die kontextualisierten Einbettungen des ersten Vorkommens von $[F]$ enthalten also schon Informationen über das Subjekt. Da das unidirektionale Sprachmodell zur Berechnung der Einbettungen eines Tokens alle bereits berechneten Einbettungen vorhergehender Tokens nutzen kann, kann GPT-2 sein Wissen über die Verbindung zwischen dem Subjekt und $[F]$ also auch in die Repräsentationen dieses zweiten Tokens integrieren. Der Zusammenhang zwischen diesem Token und $[G]$ wird anschließend in die Einbettungen von $[G]$ kodiert.

Wenn es das Prädikat [G] schließlich in der Antwortoption auftaucht, kann GPT-2 auf all diese kontextualisierten Einbettungen zurückgreifen, die jeweils Informationen über das Subjekt enthalten.

Bei höheren Tiefen macht es nur einen geringfügigen Unterschied, ob der Beweis die richtige Lösung enthält oder nicht. Vergleichen wir die Verbesserungen, die durch maskierte und ausgeglichene Vorwärtsbeweise erzielt werden (Abbildung 7.14d) mit den Verbesserungen, die man durch Zwischenschluss-Elaborationen erreicht (Abbildung 7.5c), so zeigt sich, dass der positive Einfluss nur zu einem kleinen Teil auf die im Beweis enthaltenen Zwischenschlüsse zurückzuführen ist.

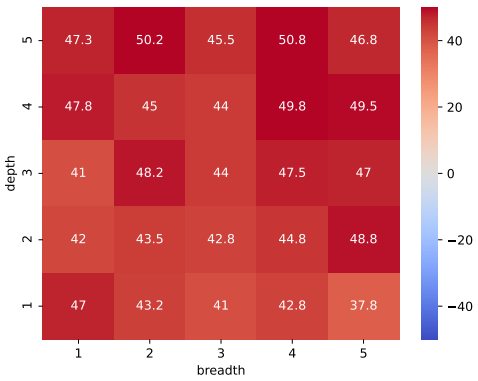
Mit zunehmender Breite der Probleme werden die Verbesserungen leicht ausgeprägter. Ein möglicher Erklärungsversuch ist der folgende: Je höher die Breite eines Problems ist, desto mehr verwirrende Regeln enthält das Problem. Im Beweis tauchen diese Regeln aber nicht mehr auf. Die Auswahl der Regeln für den Beweis aus der gesamten Menge an Regeln könnte dem Sprachmodell also einen zusätzlichen Hinweis darauf geben, welcher Zusammenhang durch den Beweis gezeigt wird. Sehr viel deutlicher ausgeprägt ist die positive Korrelation zwischen Akkuratheit und Tiefe der Probleme. Das zeigt, dass GPT-2 sehr gut dazu in der Lage ist, das Argumentationsschema des Beweises weiterzuführen. Die einzelnen Beweisschritte dienen dem Sprachmodell als Demonstrationen dafür, wie korrekte Ableitungen vollzogen werden. Es imitiert anschließend dieses Vorgehen und führt beim letzten Beweisschritt selbst eine korrekte Ableitung durch. Bei Problemen der Tiefe 4 und 5 erreicht GPT-2 mit maskierten und ausgeglichenen Beweisen durchgängig eine Akkuratheit von über 90% (vgl. Abbildung A.5d für absolute Akkuratheit). Betrachten



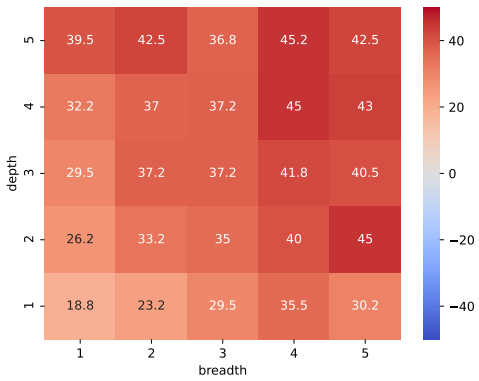
(a) Akkuratheit(Vorwärtsbeweis) | FQ2 Fantasie-Datensatz

Abbildung 7.15.: Akkuratheit mit Kontexterweiterung durch vorwärtsgerichteten Beweis auf dem *ModifiedChainRuler*-Fantasie-Datensatz

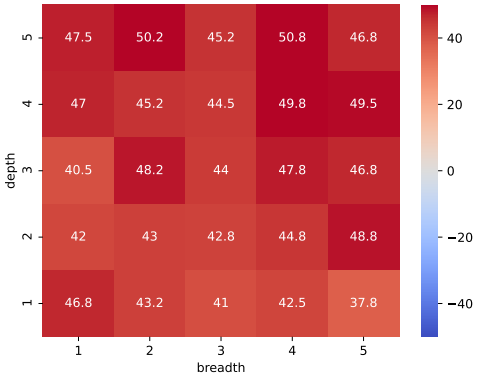
wir nun die Akkuratheit mit dem gleichen Versuchsaufbau auf dem Fantasie-Datensatz (Abb. 7.15). Hier funktionieren gegebene Vorwärtsbeweise sogar noch besser. Das Modell erreicht damit eine Korrektklassifizierungsrate von über 98% auf allen Komplexitäten. In 19 von 25 Komplexitäten sagt es sogar über 99% der Antworten richtig vorher. Das bestätigt



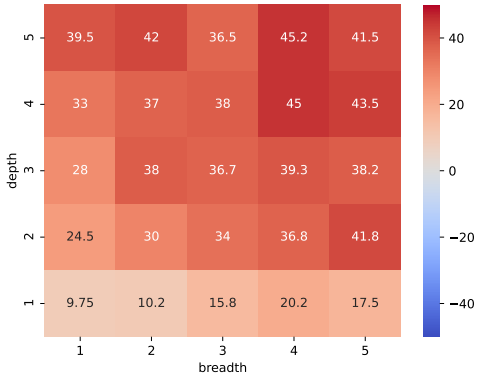
(a) Δ(keine, Vorwärtsbeweis) | FQ2 Fantasie-Datensatz



(b) Δ(keine, Vorwärtsbeweis maskiert) | FQ2 Fantasie-Datensatz



(c) Δ(keine, Vorwärtsbeweis ausgeglichen) | FQ2 Fantasie-Datensatz

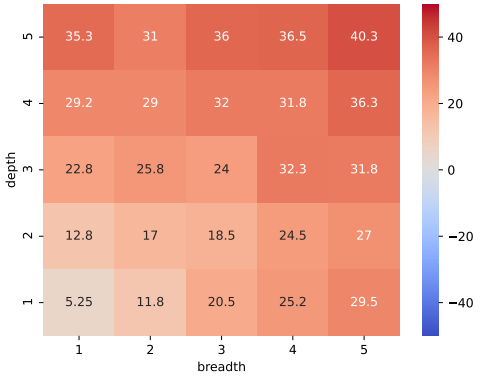


(d) Δ(keine, Vorwärtsbeweis maskiert und ausgeglichen) | FQ2 Fantasie-Datensatz

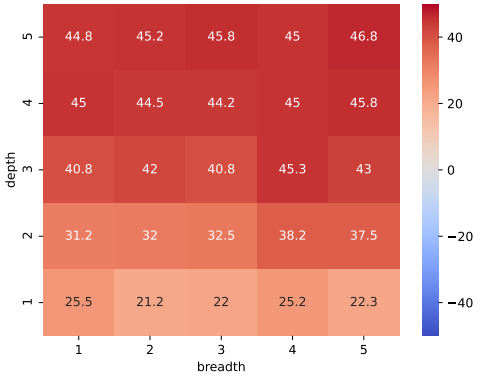
Abbildung 7.16.: Differenz zwischen Akkuratheit mit Elaboration und Basis-Akkuratheit auf dem Fantasie-Datensatz, jeweils für Varianten der Vorwärtsbeweise

die in Abschnitt 7.1.2 formulierte Überlegung: GPT-2 scheint auf Grundlage einer Problembeschreibung im Fantasie-Datensatz weniger gefestigtes Wissen über den Sachverhalt zu erlangen als bei einem *ModifiedChainRuler*-Problem. Wenn die Kontexterweiterung dann eine der Antwortmöglichkeiten enthält, basiert es seine Vorhersagen beinahe vollständig auf dieser neuen Information. Der Blick auf die Einflüsse der einzelnen Elaborationen in Abbildung 7.16 bestätigt diese Vermutung. Zwischen dem Einfluss gegebener Vorwärtsbeweise (Abb. 7.16a) und dem Einfluss ausgeglichener Vorwärtsbeweise (Abb. 7.16c) besteht fast kein Unterschied. Die stärkste Veränderung zwischen Problemen der selben Komplexität liegt hier bei nur 0,8 absoluten Prozentpunkten. Das zeigt, dass GPT-2 die gegebenen Regeln und Distraktoren fast gar nicht in seine Vorhersage einbezieht, so lange die Elaboration die richtige Lösung enthält. Maskieren der richtigen Lösung hat erwartungsgemäß einen deutlichen Effekt (siehe Abbildungen 7.16a und 7.16b). Trotzdem lässt sich auch

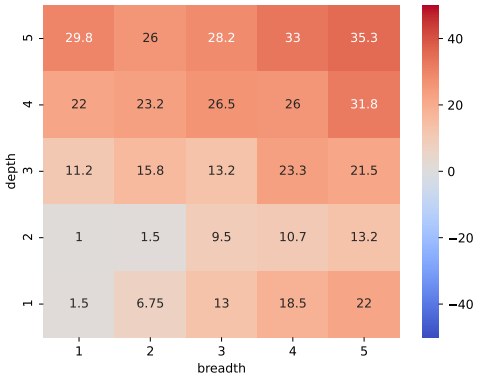
mit maskierten und ausgeglichenen Vorwärtsbeweisen noch eine starke Verbesserung gegenüber Problemen ohne Kontexterweiterung feststellen (siehe Abb. 7.16d). Auch hier zeigt sich, dass GPT-2 gut dazu in der Lage ist, das Beweisschema zu generalisieren und selbst einzusetzen. Obwohl diese Elaborationen weder die Konklusion enthalten, noch mithilfe der einfachen Heuristik lösbar sind, führen sie bei Problemen hoher Komplexität zu absoluten Korrektklassifizierungsraten von über 90% (siehe Abbildung A.5l).



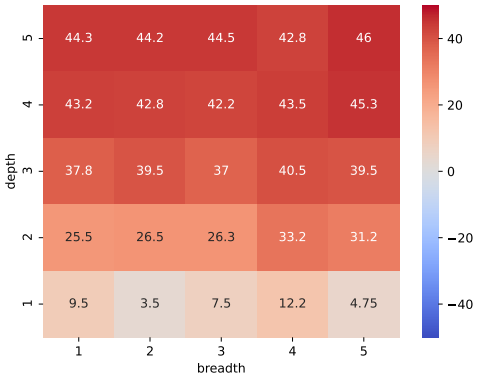
(a) Δ(keine, Vorwärtsbeweis) | FQ1



(b) Δ(keine, Vorwärts-beweis maskiert) | FQ1



(c) Δ(keine, Vorwärtsbeweis ausgeglichen) | FQ1



(d) Δ(keine, Vorwärtsbeweis maskiert und ausgeglichen) | FQ1

Abbildung 7.17.: Differenz zwischen Akkuratheit mit Elaboration und Basis-Akkuratheit, jeweils für Varianten der Vorwärtsbeweise mit FQ1

7.3.1.2. Evaluation vorwärtsgerichteter Beweise mit FQ1

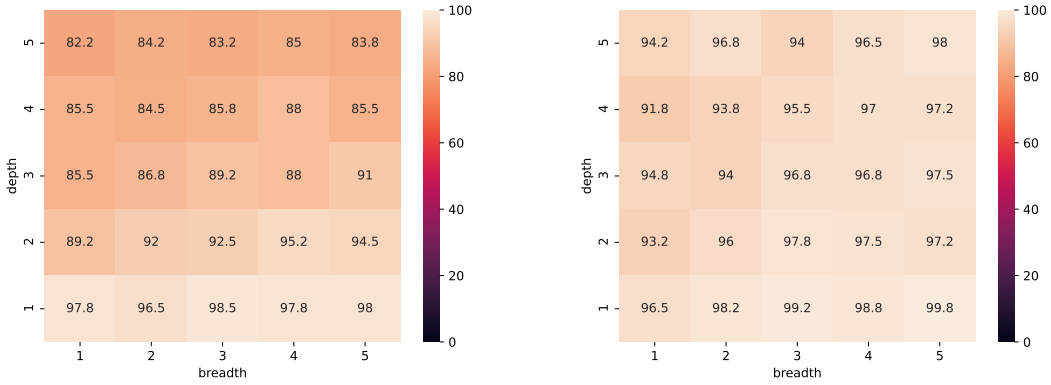
Wir gehen nun auf die Versuche mit Vorwärtsbeweisen unter Verwendung der finalen Frage „`Therefore,`“ (FQ1) ein (Abbildung 7.13a). Wieso ist GPT-2 mit dieser finalen Frage so viel schlechter in der Lage, die gegebenen Elaborationen auszunutzen? Eine Antwort darauf liefern wieder die einzelnen Varianten der Vorwärtsbeweise. Abbildung 7.17 zeigt

die Einflüsse der jeweiligen Varianten auf die Korrektklassifizierungsrate. Werden die Antwortmöglichkeiten mit der finalen Frage „`Therefore,`“ eingeleitet, so verbessern maskierte Vorwärtsbeweise die Akkuratheit sehr viel *stärker* als nicht maskierte Beweise. Fassen wir noch einmal die Unterschiede zwischen den beiden Varianten zusammen: Vorwärtsbeweise enthalten als letzten Satz „`Therefore, [Subjekt] is [`G`].`“ mit der richtigen Lösung „`[Subjekt] is [`G`].`“. Maskierte Vorwärtsbeweise unterscheiden sich von Vorwärtsbeweisen dadurch, dass die Lösung durch eine Aussage der Form „`[Subjekt] is (...).`“ ersetzt wird. Auf den letzten Satz des jeweiligen Beweises folgt wiederum die finale Frage „`Therefore,`“. Man könnte das Phänomen also folgendermaßen umformulieren: Gegeben die gleiche vorausgehenden Problemstellung mit „abgeschnittenem“ Vorwärtsbeweis berechnet GPT-2 im Durchschnitt eine geringere Wahrscheinlichkeit für die Sequenz „`[Subjekt] is [`G`]. Therefore, [Subjekt] is [`G`].`“ als für die Sequenz „`[Subjekt] is (...). Therefore, [Subjekt] is [`G`].`“. Wie lässt sich das begründen? Man könnte glauben, dass das Sprachmodell generell eine geringe Wahrscheinlichkeit dafür berechnen würde, dass auf „`Therefore,`“ der gleiche Satz folgt wie davor. Wir können aber ausschließen, dass das der Fall ist. Im Gegenteil, Sprachmodelle berechnen gewöhnlich eine hohe Wahrscheinlichkeit für sich wiederholende Sätze [15]. Das hat unter Anderem zur Folge, dass sie ohne *sampling* oft repetitive Textsequenzen generieren (siehe Abschnitt 2.2). Dieser Effekt zeigt sich auch in den Experimenten aus Abschnitt 7.1.2. Bei Orakel-Elaborationen sagt das Sprachmodell auch mit der finalen Frage FQ1 in den allermeisten Fällen die richtige Lösung vorher. Das Phänomen lässt sich also nicht dadurch erklären, dass GPT-2 die Sequenz „`[Subjekt] is [`G`]. Therefore, [Subjekt] is [`G`].`“ *im Allgemeinen* für unwahrscheinlich halten würde. Die Begründung ist meiner Theorie zufolge eine andere: Wie wir schon anhand der Vorwärtsbeweise mit der finalen Frage FQ2 gesehen haben, ist GPT-2 sehr gut dazu in der Lage, die Beweisstruktur zu erkennen und fortzuführen. In gegebenen Vorwärtsbeweisen taucht nun immer das gleiche Schema auf: „`[Subjekt] is [`I`]. If someone is [`I`], then they are [`J`]. Therefore, [Subjekt] is [`J`].`“ Die Prädikate $[I]$ und $[J]$ sind dabei nie gleich. Aus den bisherigen Beobachtungen können wir die Vermutung ableiten, dass das Sprachmodell die Textstruktur erfasst und daraus übernimmt, dass auf die Sequenz „`Therefore, [Subjekt] is`“ ein *neues* Prädikat folgen muss. Dem Modell zuvor das Lösungsprädikat $[G]$ zu zeigen, verschlechtert dementsprechend die Vorhersage. Wird die Lösung hingegen maskiert, generiert das Sprachmodell mit hoher Wahrscheinlichkeit die richtige Lösung.

Dieser Theorie zufolge zeigt sich der negative Effekt des Lösungsprädikats, sobald das Sprachmodell eine vollständige Ableitung gesehen hat. Wir können aber annehmen, dass der Effekt stärker ausgeprägt ist, wenn das Schema mindestens zweimal in der Elaboration auftaucht, also ab einer Tiefe von 2. Gleichzeitig können wir davon ausgehen, dass unmaskierte vorwärtsgerichtete Beweise die Vorhersage durch die oben beschriebenen Einflüsse verbessern: Sortieren der relevanten Regeln und Demonstrationen korrekter Ableitungen. Diese Effekte treten mit zunehmender Komplexität verstärkt auf. Eine Überlagerung der beiden Einflüsse würde also dazu führen, dass unmaskierte Vorwärtsbeweise bei Problemen mit geringer Komplexität, insbesondere ab Tiefe 2, die geringsten Verbesserungen bewirken. Mit höherer Tiefe und Breite würde ihre Effektivität aber zunehmen. Die Theorie erklärt das Muster in Abbildung 7.13a also vollständig.

7.3.2. Evaluation rückwärtsgerichteter Beweise

Da die Effekte von Kontexterweiterungen durch rückwärtsgerichtete Beweise mit beiden finalen Fragen beinahe identische Einflüsse auf die Akkuratheit haben, betrachte ich hier beispielhaft „`Therefore, we know that`“ (FQ2). Die Auswertungen für FQ1 finden sich in Anhang A (Grafik A.5). Abbildung 7.18a zeigt die Akkuratheit auf dem *ModifiedChainRuler*-Datensatz, Abbildung 7.18b auf dem Fantasie-Datensatz. Vergleicht man die Korrektklassifizierungsraten mit den entsprechenden Korrektklassifizierungsraten mit Vorwärtsbeweisen (Abbildungen 7.13b und 7.15), so zeigt sich, dass Rückwärtsbeweise einen leicht schwächeren Einfluss auf die Vorhersage haben. Die Beobachtung von Gontier et al. bestätigt sich also in diesem Versuchsaufbau nicht [14]. Bei der Interpretation dieses Ergebnisses ist allerdings zu beachten, dass die Vorhersage sensibel von den Versuchs-Parametern abhängen kann, wie ich in Abschnitt 7.3.1 anhand der Unterschiede zwischen FQ1 und FQ2 bei identischen Vorwärtsbeweisen demonstriert habe.



(a) Akkuratheit(Rückwärts-beweis) | FQ2

(b) Akkuratheit(Rückwärts-beweis) | FQ2
Fantasie-Datensatz

Abbildung 7.18.: Akkuratheit mit rückwärtsgerichtetem Beweis auf *ModifiedChainRuler*-Datensatz und *ModifiedChainRuler*-Fantasie-Datensatz

Ebenso wie in Abschnitt 7.1.2 und Abschnitt 7.3 zeigt sich auch bei rückwärtsgerichteten Beweisen, dass Elaborationen, welche die richtige Lösung enthalten, auf dem Fantasie-Datensatz besser funktionieren als auf dem *ModifiedChainRuler*-Datensatz. Maskiert man die richtige Lösung (Abbildungen 7.19b und 7.20b), verringert sich der Unterschied erwartungsgemäß. Die Verbesserungen durch Kontexterweiterungen mit Rückwärtsbeweisen zeigen auf beiden Datensätzen das gleiche Muster. Mit zunehmender Breite wird der positive Einfluss ausgeprägter – wie auch schon bei Vorwärtsbeweisen. Der Erklärungsversuch aus Abschnitt 7.3.1 lässt sich auch auf Rückwärtsbeweise anwenden: Auch hier wird bei höheren Breiten besser verdeutlicht, welche Regeln für den Beweis ausgewählt werden und welche nicht. Das könnte dem Modell einen Hinweis darauf geben, welche Eigenschaften die relevanten Regeln verbinden. Hinsichtlich der Tiefe zeigen sich bei Vorwärts- und Rückwärtsbeweisen jedoch gegenteilige Effekte. Bei rückwärtsgerichteten, maskierten

Beweisen *verringert* sich der positive Einfluss mit zunehmende Tiefe der Probleme. Dieser Zusammenhang erscheint einleuchtend, wenn man die Struktur dieser Beweise bedenkt. Rückwärtsgerichtete, maskierte Beweise beginnen mit der maskierten Konklusion und verketten die Regeln in umgekehrter Reihenfolge, bis sie beim Fakt ankommen. In den bisherigen Experimenten haben wir festgestellt, dass GPT-2 sehr gut in der Lage ist, die Struktur der Beweise zu erkennen und zu imitieren. Bei vorwärtsgerichteten, maskierten Beweisen führt das in den meisten Fällen dazu, dass GPT-2 korrekt aus der letzten Zwischenkonklusion ableitet und damit zur richtigen Lösung gelangt. Je mehr Ableitungen es zuvor gesehen hat (höhere Tiefe), desto besser ist es darin. Übertragen wir diese Beobachtung nun auf rückwärtsgerichtete Beweise, so bedeutet das, dass GPT-2 auch hier das Argumentationsschema, Aussagen zu begründen, generalisiert. Es will demzufolge für die letzte Aussage – den Fakt – eine Begründung vorhersagen. Wir können auch hier wieder annehmen, dass GPT-2 dieses Muster stärker in seine Vorhersagen integriert, je höher die Tiefe des Problems ist. Da die gesuchte Lösung aber keine Begründung für den Fakt ist, sondern eine Aussage, die maskiert am Anfang des Beweises steht, ergibt sich eine negative Beziehung zwischen Tiefe der Probleme und Nutzen der maskierten Rückwärtsbeweise.

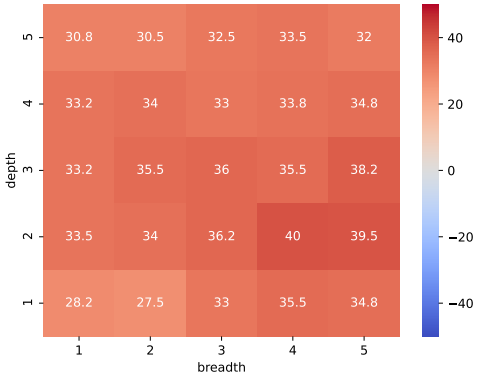
Der Effekt des Maskierens ist bei Rückwärtsbeweisen sehr viel stärker ausgeprägt als bei Vorwärtsbeweisen. Das liegt vermutlich daran, dass die gegebene Lösung einen größeren Einfluss auf die Vorhersage hat, wenn sie am Anfang des Beweises steht. Durch seine Unidirektionalität kann GPT-2 das Wissen über die Lösung besser festigen, wenn sie früher in der Textsequenz vorkommt. Das entspricht im Übrigen der Theorie, die Gontier et al. formulieren, um zu erklären, weshalb rückwärtsgerichtete Beweise die Vorhersage in ihrem Versuchsaufbau stärker beeinflussen als vorwärtsgerichtete Beweise [14]. Die maskierten Rückwärtsbeweise auszugleichen führt zu einer deutlichen Verschlechterung auf dem *ModifiedChainRuler*-Datensatz (Abb. 7.19d). Auf dem Fantasie-Datensatz hat das Ausgleichen der maskierten Beweise einen geringeren Effekt (Abb. 7.20d). Hier zeigt sich wieder, dass GPT-2 seine Vorhersagen bei Problemen aus dem Fantasie-Datensatz weniger stark auf die gegebenen Regeln aufbaut.

Interessanterweise führt das dazu, dass die Korrektklassifizierungsrate bei maskierten und ausgeglichenen Rückwärtsbeweisen auf dem Fantasie-Datensatz insgesamt höher ist als auf dem *ModifiedChainRuler*-Datensatz (vgl. Abbildungen A.4h und A.4l). Insgesamt haben maskierte und ausgeglichene Rückwärtsbeweise aber einen deutlich schwächeren Einfluss auf die Akkuratheit als maskierte und ausgeglichene Vorwärtsbeweise.

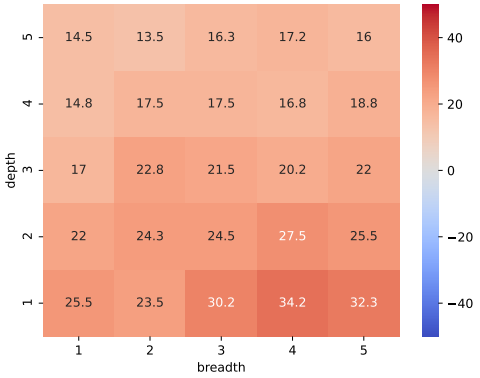
7.3.3. Zusammenfassung

Ich fasse nun einige der Erkenntnisse aus diesem Abschnitt kurz zusammen.

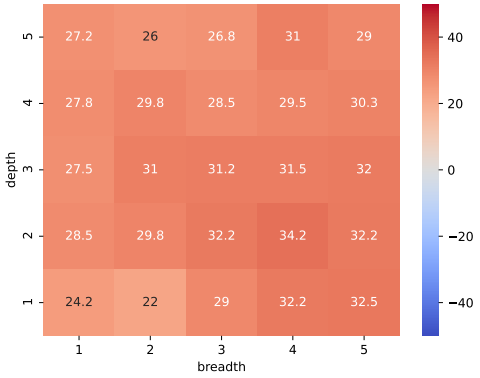
- Die Experimente mit gegebenen vorwärts- und rückwärtsgerichteten Beweisen zeigen, dass GPT-2 extrem gut in der Lage ist, die Struktur der vorgegebenen Beweise zu verallgemeinern und selbstständig weiterzuführen. Abhängig von der Struktur



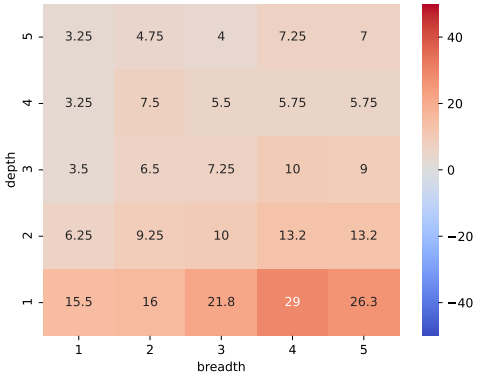
(a) Δ(keine, Rückwärtsbeweis) | FQ2



(b) Δ(keine, Rückwärtsbeweis maskiert) | FQ2



(c) Δ(keine, Rückwärtsbeweis ausgeglichen) | FQ2

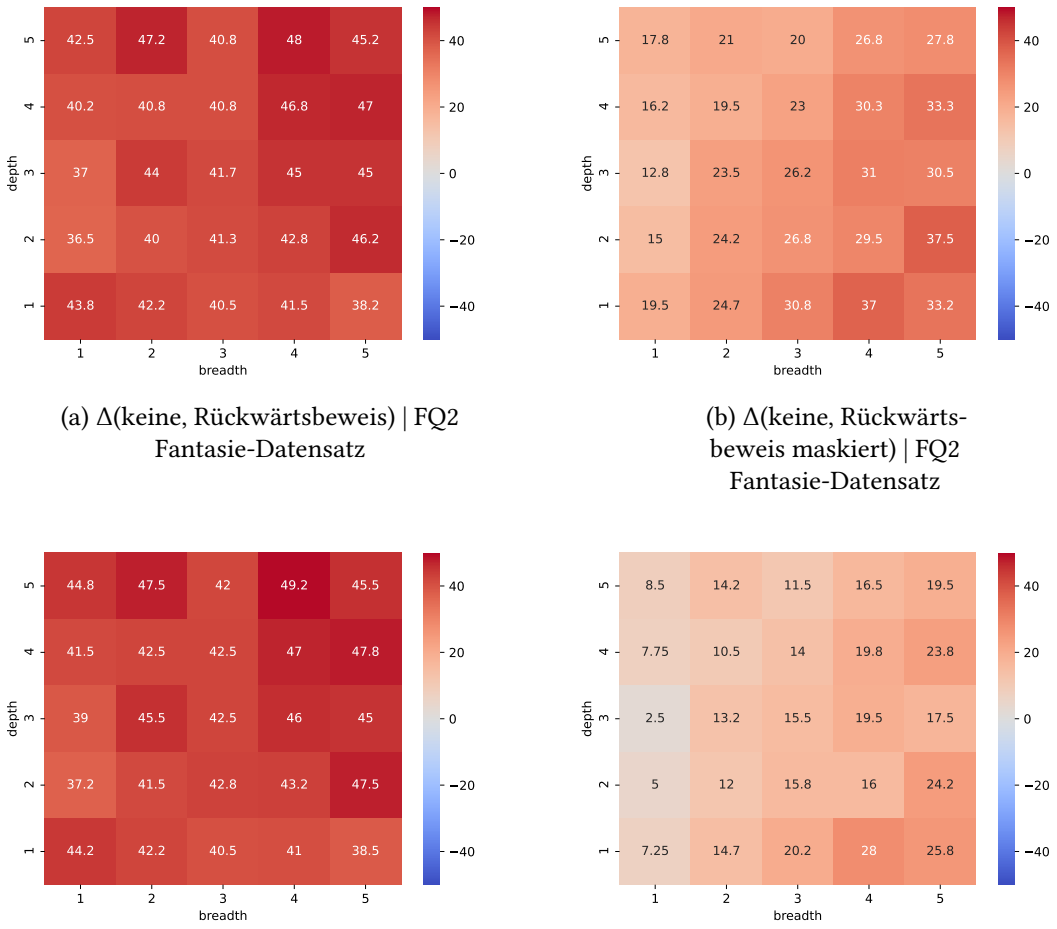


(d) Δ(keine, Rückwärtsbeweis maskiert und ausgeglichen) | FQ2

Abbildung 7.19.: Differenz zwischen Akkuratheit mit Elaboration und Basis-Akkuratheit, jeweils für Varianten der Rückwärtsbeweise mit **FQ2**

der Problemstellung kann das allerdings auch zu unerwünschten Nebeneffekten führen (siehe Abschnitt 7.3.1.2).

- Das Vorkommen der richtigen Lösung in den Beweisen beeinflusst die Vorhersage bei Rückwärtsbeweisen stärker als bei Vorwärtsbeweisen (vermutlich wegen der Unidirektionalität von GPT-2).
- Die einfache Heuristik, die das Sprachmodell beim *ChainRuler*-Datensatz verwendet, spielt bei den vorgegebenen Beweisen auf dem *ModifiedChainRuler*-Datensatz ebenfalls eine Rolle. Auf dem Fantasie-Datensatz verändert sie die Vorhersage hingegen kaum.
- Isoliert man die Beweise von Einflüssen, die durch die Heuristik und die gegebene Lösung entstehen, verbessern sie die Vorhersagen trotzdem noch stark. Das lässt darauf schließen, dass GPT-2 die Beweise korrekt interpretiert.



(a) Δ(keine, Rückwärtsbeweis) | FQ2 Fantasie-Datensatz

(b) Δ(keine, Rückwärtsbeweis maskiert) | FQ2 Fantasie-Datensatz

(c) Δ(keine, Rückwärtsbeweis ausgeglichen) | FQ2 Fantasie-Datensatz

(d) Δ(keine, Rückwärtsbeweis maskiert und ausgeglichen) | FQ2 Fantasie-Datensatz

Abbildung 7.20.: Differenz zwischen Akkuratheit mit Elaboration und Basis-Akkuratheit auf dem *ModifiedChainRuler*-**Fantasie**-Datensatz, jeweils für Varianten der Rückwärtsbeweise

- Betrachtet man die maskierten und ausgeglichenen Varianten der Beweise, so
  - ◇ haben vorwärtsgerichtete Beweise ab Tiefe 2 einen stärkeren Einfluss als rückwärtsgerichtete Beweise
  - ◇ funktionieren Vorwärtsbeweise mit zunehmender Tiefe deutlich und mit zunehmender Breite leicht besser
  - ◇ funktionieren Rückwärtsbeweise mit zunehmender Breite besser und mit zunehmender Tiefe schlechter
- Alle Arten von Beweisen führen auf dem *ModifiedChainRuler*-Fantasie-Datensatz sowohl relativ als auch absolut zu höheren Korrektklassifizierungsraten als auf dem *ModifiedChainRuler*-Datensatz. Das entspricht der Beobachtung, dass GPT-2 seine

Vorhersagen auf dem Fantasie-Datensatz stärker auf die jeweiligen Kontexterweiterungen stützt als beim *ModifiedChainRuler*-Datensatz (vgl. Abschnitt 7.1).

8. Generierung von Problemelaborationen

In Kapitel 7 habe ich gezeigt, dass die Korrektklassifizierungsraten auf den *ModifiedChainRuler*-Datensätzen sich mithilfe von Kontexterweiterungen dramatisch verbessern lassen. In diesem Kapitel untersuche ich, ob GPT-2 in der Lage ist, selbst Kontexterweiterungen zu generieren, die seine Vorhersagen verbessern. Analog zu Betz et al. nenne ich Texte, die das Sprachmodell auf Grundlage der jeweiligen Problembeschreibung neu erzeugt, auch *dynamische Problemelaborationen* [3].

Im ersten Schritt generiert das Sprachmodell selbst dynamische Problemelaborationen. In Abschnitt 8.1 untersuche ich drei verschiedene Methoden, das Sprachmodell im Zero-Shot-Setting dazu zu animieren, Problemelaborationen zu erstellen. In Abschnitt 8.3 generiert das Modell die Elaborationen mithilfe von Few-Shot-Learning. Bei dieser Methode werden jeder Problembeschreibung zwei Beispiel-Probleme inklusive hochqualitativer Elaborationen vorangestellt. Das Sprachmodell soll aus diesen Beispielen das gewünschte Vorgehen erlernen (siehe Abschnitt 2.3).

Im zweiten Schritt werden die generierten Problemelaborationen als Kontexterweiterungen zu den Problembeschreibungen hinzugefügt. Auf Grundlage der Problembeschreibung, der selbst erzeugten Elaboration und einer finalen Frage trifft das Sprachmodell schließlich seine Antwortvorhersage. Das Vorgehen in diesem Schritt gleicht dem Vorgehen mit vorgegebenen Kontexterweiterungen in Kapitel 7.

8.1. Experimente im Zero-Shot-Setting

In diesem Abschnitt betrachte ich drei verschiedene Methoden, das Sprachmodell im Zero-Shot-Setting dazu anzuregen, auf Grundlage einer Problembeschreibung eine geeignete Elaboration zu generieren. Das hier beschriebene Vorgehen entspricht dem Vorgehen von Betz et al. [3].

Freie Elaborationen sind Kontexterweiterungen, die das Sprachmodell ohne zusätzliche Strukturierung von außen erstellt. Mit dieser Methode untersuche ich, welchen Einfluss Kontexterweiterungen auf die Vorhersage haben, die vom Sprachmodell alleine auf Grundlage der Problembeschreibung und einer generischen Frage erstellt werden. Das Vorgehen dabei ist folgendermaßen: Zunächst wird zu jedem Problem auf den *ModifiedChainRuler*-Datensätzen synthetisch eine Frage über die beiden Antwortmöglichkeiten erzeugt.

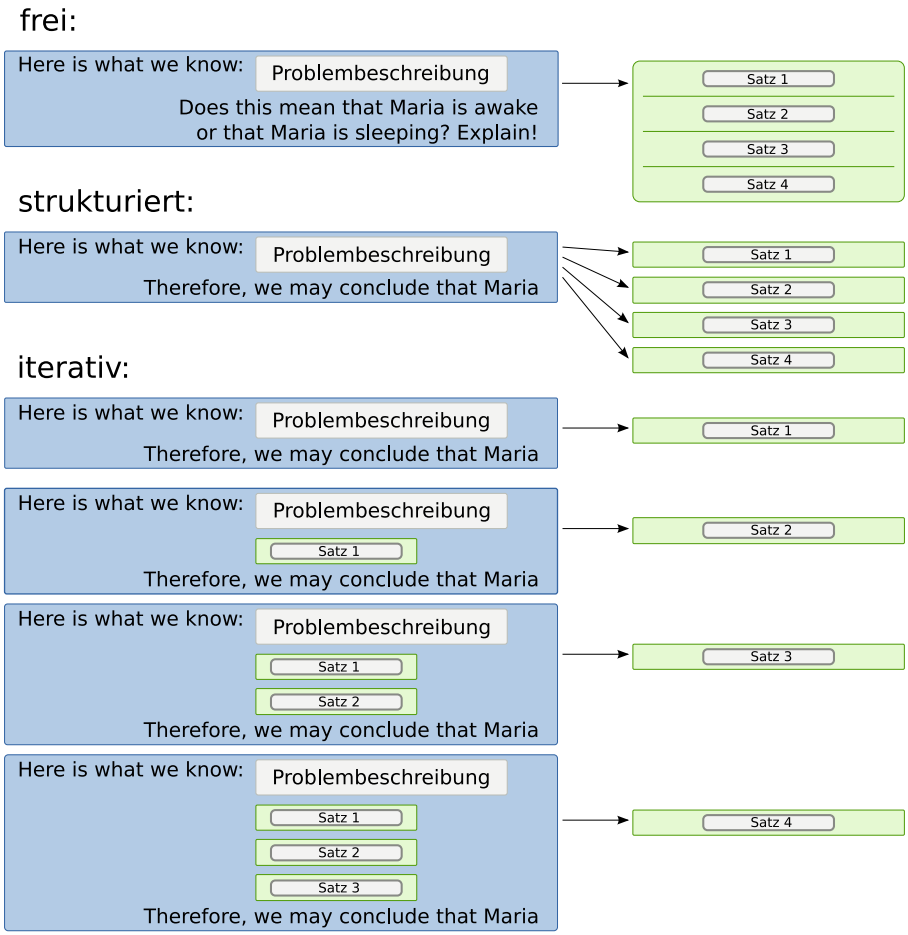


Abbildung 8.1.: Übersicht über die Methoden zur Generierung dynamischer Kontexterweiterungen im Zero-Shot-Setting am Beispiel von Problem 5.2

Für Problem 5.2 lautet diese Frage beispielsweise „`Does this mean that Maria is awake or that Maria is sleeping? Explain!`“. Anschließend wird die Einleitung „`Here is what we know:`“ mit der Problembeschreibung und der dazugehörigen Frage zu einem Kontext verkettet. Anschließend bekommt das Sprachmodell die Möglichkeit, auf Grundlage dieses Kontexts vier Sätze zu generieren. Um Elaborationen zu erhalten, die möglichst wenig repetitiv sind (siehe Abschnitt 2.2), verwende ich dafür *nucleus sampling* mit $p = 0,5$.

Die Idee hinter **strukturierten Elaborationen** ist dagegen, Aussagen über das Subjekt zu sammeln, die das Sprachmodell jeweils einzeln aus dem Kontext ableitet. Der Kontext beginnt wieder mit der Einleitung „`Here is what we know:`“ und der Problemstellung. Darauf folgt diesmal der Prompt „`Therefore, we may conclude that`“, gefolgt vom Namen des Subjekts. Für Problem 5.2 würde der Prompt also beispielsweise „`Therefore, we may conclude that Maria`“ lauten. GPT-2 bekommt diesen Kontext viermal gegeben und generiert mit *nucleus sampling* jeweils einen Satz. Den vier generierten Sätzen wird jeweils

wieder der Subjektname vorangestellt, anschließend werden sie zu einer Elaboration konkateniert.

Da die Aussagen bei strukturierten Elaborationen unabhängig voneinander generiert werden, hat das Sprachmodell nicht die Möglichkeit, Schlüsse zu ziehen, die aufeinander aufbauen. Das wird durch **iterative Elaborationen** ermöglicht. Der erste Kontext entspricht dem Kontext bei strukturierten Elaborationen. Das Sprachmodell vervollständigt wieder den Prompt „`Therefore, we may conclude that [Subjekt]`“. Der generierte Satz wird anschließend mit dem Subjektnamen ergänzt und als Kontexterweiterung zwischen die Problembeschreibung und den Prompt eingefügt. Das bildet den neuen Kontext, der dem Modell wieder übergeben wird. Es generiert daraufhin einen zweiten Satz, der dann wieder zwischen den ersten Satz und den Prompt eingefügt wird, um einen neuen Kontext zu erstellen. Auf diese Weise werden vier Sätze generiert, die zusammen die Problemelaboration bilden. Für iterative Elaborationen wird *beam decoding* verwendet. Diese Dekodierstrategie eignet sich besonders gut, wenn ein spezifischer, kurzer Text generiert werden soll (siehe Abschnitt 2.2). Im Gegensatz zu freien und strukturierten Elaborationen ist eine geringe Diversität bei iterativen Elaborationen wünschenswert. Das Sprachmodell soll dadurch auf seine zuvor generierten Aussagen aufbauen.

Abbildung 8.1 zeigt das Vorgehen beim Erstellen der drei Varianten. Die Elaborationen werden erst generiert und anschließend als Kontexterweiterungen an die Probleme angefügt. Zusammen mit der finalen Frage „`Therefore,`“ (FQ1) oder „`Therefore, we know that`“ (FQ2) bilden sie den Kontext, auf den das Sprachmodell seine Vorhersagen basiert.

Betrachten wir nun die Elaborationen, die GPT-2 zu Beispiel 5.2 generiert hat.

Freie Elaboration:

> Kontext: „`Here is what we know: If someone is happy, then they are awake. If someone is careless, then they are happy. If someone is serious, then they are sleeping. If someone is charming, then they are careless. If someone is young, then they are sad. Maria is charming. Does this mean that Maria is awake or that Maria is sleeping? Explain!`“
>
> Von GPT-2 generierte freie Elaboration: „`The subject's attitude toward the object of attention is not always as obvious as it might seem. The subject might not be consciously aware of the fact that they are being observed. In fact, they might not even be aware that they are being...`“

Strukturierte und iterative Elaborationen:

> Kontext: „`Here is what we know: If someone is happy, then they are awake. If someone is careless, then they are happy. If someone is serious, then they are sleeping. If someone is charming, then they are careless. If someone is young, then they are sad. Maria is charming. Therefore, we may conclude that Maria`“

> Von GPT-2 generierte strukturierte Elaboration: „`Maria is asleep. Maria is sleeping. Maria is asleep. Maria is happy.`“
>
> Von GPT-2 generierte iterative Elaboration: „`Maria is happy. Maria is asleep. Maria is asleep because she is not happy. Maria is asleep because she is not happy.`“

Die dynamischen Problemelaborationen weichen in diesem Beispiel stark von den hochqualitativen Elaborationen ab, die wir in Kapitel 7 betrachtet haben. Die freie Elaboration hat keinen erkennbaren Bezug zur Problembeschreibung. Sowohl die strukturierte als auch die iterative Elaboration enthalten mehrmals die falschen Alternative „`Maria is sleeping.`“, beziehungsweise „`Maria is asleep.`“. Bei der iterativen Elaboration generiert GPT-2 zunächst den falschen Schluss und begründet diesen anschließend: „`Maria is asleep because she is not happy.`“. Allerdings ist „`If someone is not happy, then they are asleep.`“ nicht logisch aus der gegebenen Regel „`If someone is happy, then they are awake.`“ ableitbar. Doch selbst durch eine korrekten Regelanwendung (beispielsweise „`Maria is asleep because she is serious.`“) könnte das Modell nicht mehr zur Lösung gelangen. Hier zeigt sich ein Nachteil iterativer Elaborationen: Wenn das Sprachmodell auf die zuvor generierten Aussagen aufbaut, kann es durch korrekte Schlussfolgerungen nicht mehr zur Konklusion gelangen, nachdem es eine falsche Aussage erzeugt hat.

8.2. Evaluation der im Zero-Shot-Setting generierten Elaborationen

Abbildung 8.2 zeigt die Korrektklassifizierungsraten durch Kontexterweiterungen mit Elaborationen, die GPT-2 im Zero-Shot-Setting generiert hat, sowie die jeweiligen Veränderungen gegenüber der Basis-Akkuratheit. Keine der drei Arten von Elaborationen verbessert die Schlussfolgerungsfähigkeiten des Sprachmodells eindeutig.

Auf dem *ModifiedChainRuler*-Datensatz lassen sich folgende Tendenzen erkennen: Bei Problemen mit geringer Tiefe verschlechtern dynamisch erstellten Elaborationen die Akkuratheit. Freie Elaborationen haben hier – mit Verlusten von bis zu -9,5 absoluten Prozentpunkten – den stärksten negativen Einfluss (siehe Abbildung 8.2e und 8.2c). Das stimmt mit der Beobachtung von Petroni et al. überein, dass freie, selbst generierte Kontexterweiterungen die Performanz vortrainierter Sprachmodelle verschlechtern [26, Kapitel 4.2]. In einer zufälligen Stichprobe von 5 freien Elaborationen pro Komplexität (125 Elaborationen insgesamt) enthalten 84 Elaborationen (67,2%) keine einzige Aussage, die logisch aus der Aufgabenbeschreibung folgt. Die Stichprobengröße ist zu klein, um eine gesicherte Aussage über alle Elaborationen zu treffen. Sie legt aber die folgende Hypothese nahe: Freie Elaborationen bauen größtenteils nicht logisch auf die Problemstellung auf. In vielen Fällen kommen darin aber Prädikate aus der Aufgabenstellung vor. Bei Problemen mit hoher Tiefe, aber geringer Breite ist die Wahrscheinlichkeit hoch, dass diese Prädikate aus der Regelkette stammen. Dadurch werden mit hoher Wahrscheinlichkeit Aussagen generiert, die zufällig auf die relevanten Regeln verweisen.

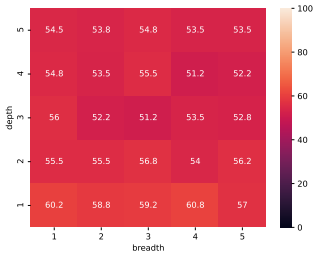
Auch strukturierte Elaborationen wirken sich bei Problemen der Tiefe 1 deutlich negativ aus (siehe Abbildung 8.2j und 8.2k). Auf Problemen höherer Tiefe haben freie und strukturierte Elaborationen einen gemischten Effekt. Bei Problemen mit hoher Tiefe, aber geringer Breite scheinen diese Elaborationen noch am besten zu funktionieren. Das liefert einen Hinweis darauf, dass das Sprachmodell sich bei der Textgenerierung möglicherweise leicht von Distraktoren beeinflussen lässt und Aussagen generiert, die der Problembeschreibung widersprechen. Im Kontext kommen $(d + 2b + 1)$ Prädikate vor: Ein Prädikat im Fakt, zusätzliche d Prädikate in der Regelkette und zwei Prädikate in jedem Distraktor. Wenn GPT-2 ein beliebiges Prädikat aus dem Kontext einsetzt, um den Satzanfang „`Therefore, we may conclude that [Subjekt]`“ fortzuführen, ist die Wahrscheinlichkeit, dass die Aussage logisch aus dem Kontext folgt $(d + 1)/(d + 2b + 1)$. Bei Aufgaben der Tiefe und Breite 1 wäre die Wahrscheinlichkeit, dass mindestens 2 der 4 generierten Sätze aus dem Kontext folgen, also 0, 5. Die Basis-Akkuratheit in dieser Komplexität ist aber deutlich besser, sodass diese „zufällige“ Kontexterweiterung mutmaßlich zu einer Verschlechterung der Akkuratheit führen würde. Betrachten wir nun die Fälle, in denen die Wahrscheinlichkeit, dass mindestens zwei der vier „zufällig“ generierten Sätze Ableitungen aus dem Kontext sind, höher ist als die Basis-Akkuratheit auf dieser Komplexität. Das trifft für die Komplexitäten $(d = 2, b = 1)$, $(d = 3, b = 1)$, $(d = 4, b = 1)$, $(d = 4, b = 2)$, $(d = 5, b = 1)$ und $(d = 5, b = 2)$ zu. Das entspricht den Komplexitäten, in denen freie und strukturierte Elaborationen verhältnismäßig gut funktionieren. Das legt die Vermutung nahe, dass Zufallseffekte bei den Ergebnissen eine Rolle spielen.

Iterative Elaborationen zeigen ähnliche Effekte wie freie und strukturierte Elaborationen, allerdings mit etwas besseren Werten. Ab Tiefe 2 zeigen sich hier nur noch wenige Verschlechterungen, die zudem schwächer ausgeprägt sind als bei freien und strukturierten Elaborationen. Bei Problemen der Tiefe 4 und 5 kommt es nur in einer einzigen Komplexität zu einer leichten Verschlechterungen von -0,25 Prozentpunkten. Das Sprachmodell scheint also eingeschränkt fähig zu sein, aus bereits gezogenen Schlüssen korrekte Schlussfolgerungen zu ziehen. Insgesamt sind die Effekte allerdings nicht stark genug ausgeprägt, um gesicherte Erkenntnisse aus der Analyse abzuleiten.

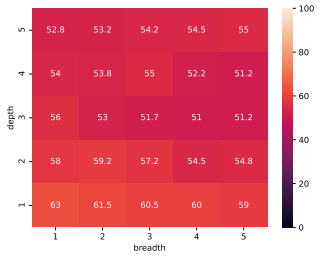
Die Ergebnisse auf dem *ModifiedChainRuler*-Fantasie-Datensatz (Abbildung 8.2c, 8.2i und 8.2o) sind für alle Elaborationstypen auf allen Komplexitäten durchwachsen. Im Vergleich zum *ModifiedChainRuler*-Datensatz ergeben sich ausgeprägtere Schwankungen. Das ist nicht überraschend, wenn man bedenkt, dass alle Versuche aus Kapitel 7 darauf hindeuten, dass Kontexterweiterungen die Vorhersage auf dem Fantasie-Datensatz stärker beeinflussen.

8.3. Few-Shot-Learning

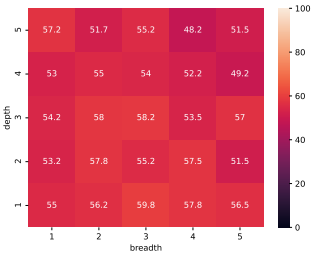
Die Ergebnisse aus Abschnitt 7.3 zeigen, dass Kontexterweiterungen mit vorwärtsgerichteten und – zu etwas geringerem Ausmaß – rückwärtsgerichteten Beweisen die Performanz von GPT-2 auf den *ModifiedChainRuler*-Datensätzen drastisch verbessern. Im Gegensatz dazu haben Problemelaborationen, die das Modell im Zero-Shot-Setting selbst generiert,



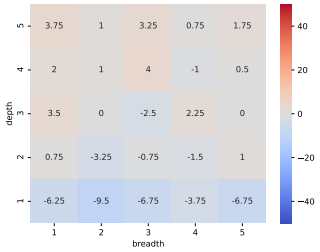
(a) Akkuratheit (frei) | FQ1



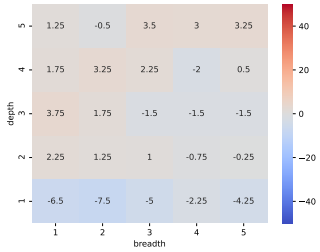
(b) Akkuratheit (frei) | FQ2



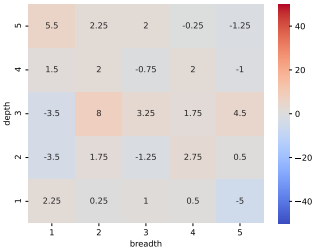
(c) Akkuratheit (frei) | FQ2 Fantasie-Datensatz



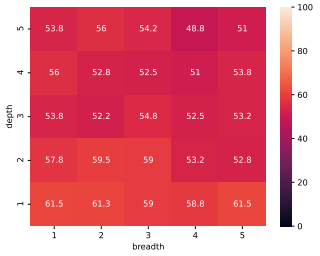
(d) Δ(keine, frei) | FQ1



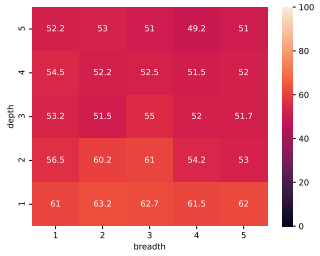
(e) Δ(keine, frei) | FQ2



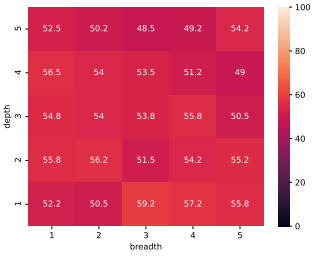
(f) Δ(keine, frei) | FQ2 Fantasie-Datensatz



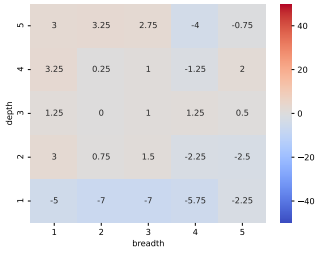
(g) Akkuratheit (strukturiert) | FQ1



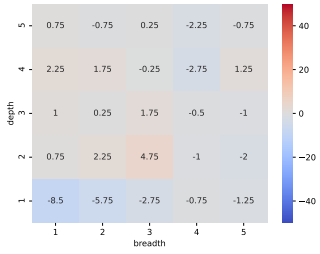
(h) Akkuratheit (strukturiert) | FQ2



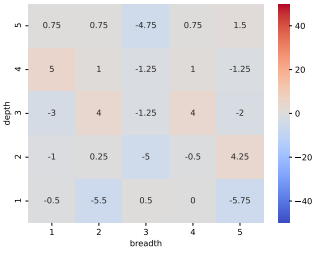
(i) Akkuratheit (strukturiert) | FQ2 Fantasie-Datensatz



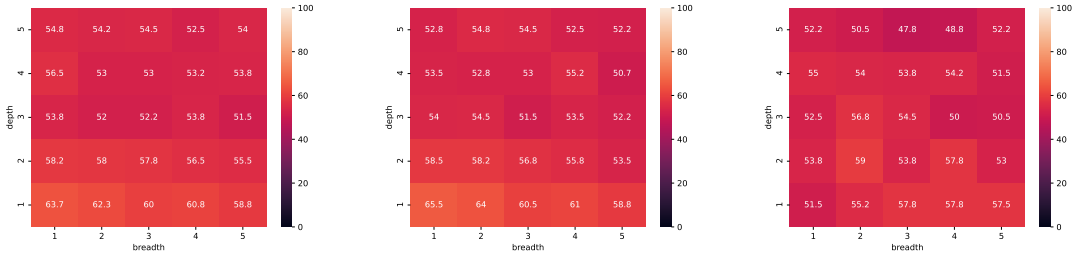
(j) Δ(keine, strukturiert) | FQ1



(k) Δ(keine, strukturiert) | FQ2



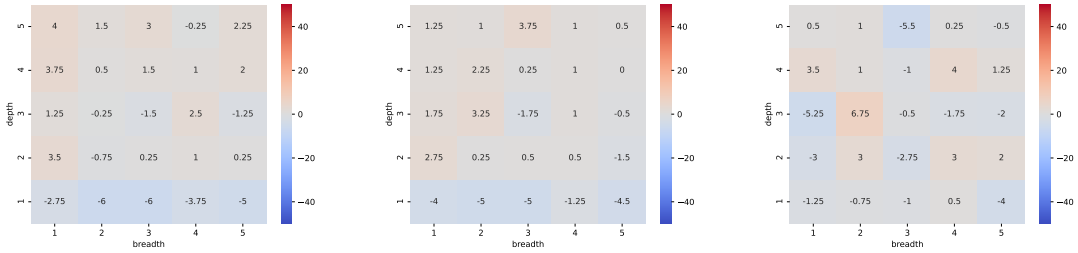
(l) Δ(keine, strukturiert) | FQ2 Fantasie-Datensatz



(m) Akkuratheit (iterativ) | FQ1

(n) Akkuratheit (iterativ) | FQ2

(o) Akkuratheit (iterativ) | FQ2 Fantasie-Datensatz



(p) Δ(keine, iterativ) | FQ1

(q) Δ(keine, iterativ) | FQ2

(r) Δ(keine, iterativ) | FQ2 Fantasie-Datensatz

Abbildung 8.2.: Akkuratheit mit Elaborationen, die GPT-2 im Zero-Shot-Setting erstellt hat und jeweilige Differenzen zu Akkuratheit ohne Kontexterweiterungen (d)-(f)

keine eindeutig positiven Effekte auf die Vorhersage (siehe Abschnitt 8.1). Dieses Ergebnis lässt darauf schließen, dass die dynamisch generierten Elaborationen stark von den vorgegebenen Beweisen abweichen. Mithilfe von Few-Shot-Learning teste ich, ob das Sprachmodell anhand von zwei hochqualitativen Beispielen lernen kann, selbst wirkungsvollere Problemelaborationen zu generieren.

Zu jedem Problem werden zunächst zwei Beispiele erstellt. Ein Beispiel besteht aus den folgenden Komponenten: Der Einleitung „`Here's what we know:`“, der Problembeschreibung eines Beispiel-Problems, der Überleitung „`In particular, we know that`“ und ein zur Problembeschreibung gehörender Beweis. Betz et al. benutzen statische Beispiele fürs Few-Shot-Learning [3, Anhang A]. Da die Performanz vortrainierter Sprachmodelle von der genauen Formulierung der Prompts abhängen kann [19], versuche ich stattdessen, eine möglichst diverse Auswahl zu verwenden. Ich wähle die Beispiel-Probleme daher zufällig aus dem Datensatz aus. Sie haben die gleiche Komplexität, aber eine andere Gruppe als das zu lösende Problem. Probleme unterschiedlicher Gruppen können bis auf das Quell-Prädikat [F] des Fakts „`[Subjekt] is [`F`]`“ keine Prädikate und auch keine Subjektnamen teilen. Mit dieser Konstruktion ist es also unmöglich, dass aus den Regeln eines Beispiels die falsche Alternative des eigentlichen Problems gefolgert werden kann. Als Beweise nutze ich die in Abschnitt 7.3 synthetisierten Vorwärts- oder Rückwärtsbeweise zu den jeweiligen Beispiel-Problemen.

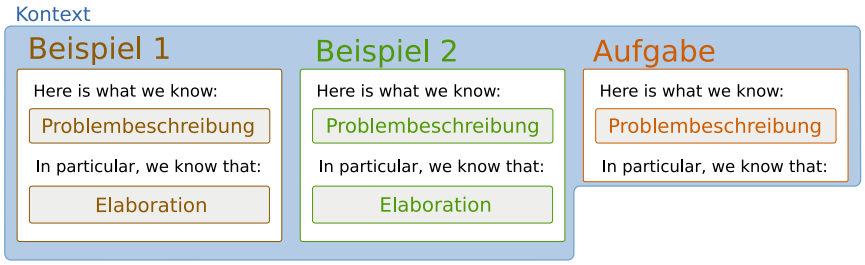


Abbildung 8.3.: Kontext beim Few-Shot-Learning

Mithilfe der beiden Beispiele wird nun der Kontext fürs Few-Shot-Learning erstellt. Auf die beiden Beispiele folgt noch einmal die Einleitung „`Here is what we know:`“ sowie die eigentliche Aufgabe. Die Überleitung „`In particular, we know that`“ dient nun als Prompt für die Textgenerierung (siehe Schaubild 8.3). Die Beweise, die dem Sprachmodell im Few-Shot-Learning als Musterlösungen dienen, enthalten jeweils den Fakt, die Regelkette, alle Zwischenschlüsse sowie die Konklusion der Beispiel-Aufgaben. Bei einem Problem der Tiefe d und Breite b entspricht das $(2d + 1)$ Sätzen. Da GPT-2 die Struktur der Beweise imitieren soll, erhält es die Möglichkeit, ebenso viele Sätze zu generieren. Zur Generierung wird wieder *beam decoding* verwendet, da ein Abweichen von der vorgegebenen Struktur in diesem Fall nicht erwünscht ist.

Im zweiten Schritt werden die dynamischen Problemelaborationen, die mithilfe von Few-Shot-Learning generiert wurden, als Kontexterweiterungen für die Vorhersage verwendet. Zusammen mit der Problembeschreibung und der finalen Frage bilden sie jeweils den Kontext für die Vorhersage. Die Beispiele sind nicht Teil dieses Kontexts, sie können die Vorhersage also höchstens indirekt beeinflussen.

Die Experimente werden auf beiden Datensätzen durchgeführt – einmal mit Beispielen, die als Musterlösungen vorwärtsgerichtete Beweise enthalten (FSL vorwärts) und einmal mit Beispielen, die als Musterlösungen Rückwärtsbeweise enthalten (FSL rückwärts). Betrachten wir wieder die generierten Elaborationen zu Problem 5.2. Die Beispiel-Problembeschreibungen und dazugehörigen Beweise sind hier dem Schaubild 8.3 entsprechend farbig hervorgehoben.

FSL vorwärts:

> Kontext: „`Here is what we know: If someone is wise, then they are sitting. If someone is blunt, then they are warm. If someone is inattentive, then they are standing. If someone is sitting, then they are cold. If someone is intelligent, then they are wise. Olivia is intelligent. In particular, we know that Olivia is intelligent. If someone is intelligent, then they are wise. Therefore, Olivia is wise. If someone is wise, then they are sitting. Therefore, Olivia is sitting. If someone is sitting, then they are cold. Therefore,`

> Olivia is cold. Here is what we know: If someone is beautiful, then they are warm. If someone is fresh, then they are cold. If someone is warm, then they are standing. If someone is stingy, then they are popular. Julie is beautiful. If someone is standing, then they are unpopular. In particular, we know that Julie is beautiful. If someone is beautiful, then they are warm. Therefore, Julie is warm. If someone is warm, then they are standing. Therefore, Julie is standing. If someone is standing, then they are unpopular. Therefore, Julie is unpopular. Here is what we know: If someone is happy, then they are awake. If someone is careless, then they are happy. If someone is serious, then they are sleeping. If someone is charming, then they are careless. If someone is young, then they are sad. Maria is charming. In particular, we know that“
>
> Von GPT-2 generierte Elaboration: „Maria is charming. If someone is charming, then they are happy. Therefore, Maria is happy. If someone is happy, then they are awake. Therefore, Maria is awake. If someone is asleep, then they are sad. Therefore, Maria is sad.“

GPT-2 beginnt die FSL-vorwärts-Elaboration richtigerweise mit dem Fakt. Es verbindet anschließend die Regeln „If someone is charming, then they are careless.“ und „If someone is careless, then they are happy.“ zu der Regel „If someone is charming, then they are happy.“. Dieser Schluss ist zwar korrekt, entspricht aber nicht dem Vorgehen, das in den Beispielen demonstriert wird. GPT-2 generiert die richtige Lösung „Maria is awake.“, ist aber anschließend nicht in der Lage, auf diese Aussage aufzubauen. Da das Sprachmodell eine Ableitung übersprungen hat, gelangt es zur richtigen Lösung, bevor es das Ende der Elaboration erreicht hat. Auf die Lösung lässt sich aber keine Regel nach dem vorgegebenen Schema anwenden. Stattdessen wendet GPT-2 den Umkehrschluss einer Regel („If someone is asleep, then they are sad.“) inkorrekt an und generiert eine falsche Aussage. Möglicherweise lernt GPT-2 aus den Beispielen im Few-Shot-Learning also nicht, wie korrekte Ableitungen gebildet werden, sondern imitiert nur deren syntaktische Struktur. In der hier betrachteten Elaboration zieht GPT-2 keine korrekten Schlüsse mehr, sobald es dieses einfache Schema nicht mehr anwenden kann.

FSL rückwärts:

> Kontext: „Here is what we know: If someone is wise, then they are sitting. If someone is blunt, then they are warm. If someone is inattentive, then they are standing. If someone is sitting, then they are cold. If someone is intelligent, then they are wise. Olivia is intelligent. In particular, we know that Olivia is cold. That’s because if someone is sitting, then they are cold. Olivia is sitting. That’s because if someone is wise, then they are sitting. Olivia is wise. That’s because if someone is intelligent, then they are wise. Olivia is intelligent. Here is what we know: If someone is beautiful, then they are warm. If someone is fresh, then they are cold. If someone is warm, then they are standing. If someone is stingy, then they are popular. Julie is beautiful. If someone is standing, then they are unpopular. In particular, we know that Julie is unpopular. That’s because if someone

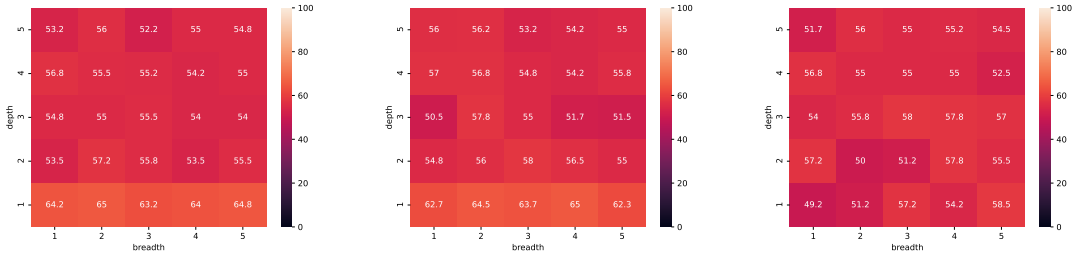
> is standing, then they are unpopular. Julie is standing. That's because if someone is warm, then they are standing. Julie is warm. That's because if someone is beautiful, then they are warm. Julie is beautiful. Here is what we know: If someone is happy, then they are awake. If someone is careless, then they are happy. If someone is serious, then they are sleeping. If someone is charming, then they are careless. If someone is young, then they are sad. Maria is charming. In particular, we know that"
>
> Von GPT-2 generierte Elaboration: „Maria is sad. That's because if someone is charming, then they are sad. Maria is charming. That's because if someone is happy, then they are awake. Maria is happy. That's because if someone is careless, then they are happy. Maria is careless."

Auf den ersten Blick wirkt die von GPT-2 generierte FSL-rückwärts-Elaboration wie ein Rückwärtsbeweis. Bei näherem Hinsehen zeigt sich aber, dass die Regel „If someone is charming, then they are sad." nicht in der Problembeschreibung vorkommt. Das Sprachmodell generiert daraufhin den Fakt („Maria is charming."), der aber nicht mithilfe einer Regel begründet werden kann. Es fügt stattdessen eine unpassende Regel ein und generiert dann den Zwischenschluss „Maria is happy.", den es korrekterweise mit dem Zwischenschluss „Maria is careless." begründet.

8.4. Evaluation der durch Few-Shot-Learning generierten Elaborationen

Die Korrektklassifizierungsraten mit Elaborationen, die durch Few-Shot-Learning mit Vorwärtsbeweisen erstellt wurden, finden sich in Abbildung 8.4. Gegenüber den im Zero-Shot-Setting generierten Elaborationen lassen sich leichte Verbesserungen erkennen. Beispiele hoher Tiefe scheinen dabei am besten zu funktionieren. Bei Problemen der Tiefe 4 und 5 kommt es durch die Kontexterweiterungen sowohl auf dem Fantasie-Datensatz als auch auf dem *ModifiedChainRuler*-Datensatz nicht mehr zu Verschlechterungen. Zusätzlich fällt auch der negative Einfluss auf einfachen Problemen durchschnittlich schwächer aus als mit den Elaborationen, die im Zero-Shot-Setting generiert wurden. Insgesamt scheint das Sprachmodell eingeschränkt dazu fähig zu sein, die Struktur der Vorwärtsbeweise durch Few-Shot-Learning zu imitieren.

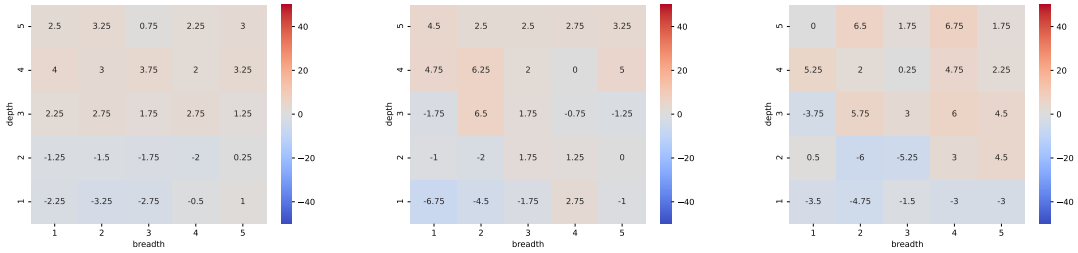
Verwendet man rückwärtsgerichtete Beweise für die Beispiele im Few-Shot-Learning, zeigt sich allerdings ein gegenteiliger Effekt (siehe Abbildung 8.5). In manchen Komplexitäten kommt es zwar noch zu leichten Verbesserungen gegenüber der Basis-Akkuratheit, aber sie sind schwächer ausgeprägt und weniger häufig als die Verschlechterungen, die über alle Tiefen und Breiten hinweg auftreten. Das Sprachmodell scheint also nicht in der Lage zu sein, die Struktur der Rückwärtsbeweise korrekt zu generalisieren. Das deckt sich mit der Beobachtung von Gontier et al., dass Sprachmodelle Schwierigkeiten haben, rückwärtsgerichtete Beweise zu erstellen. Wie in Abschnitt 7.3.2 beschrieben, kann GPT-2 hochqualitative Rückwärtsbeweise, welche die Lösung nicht enthalten, schlechter nutzen



(a) Akkuratheit (FSL vorwärts) | FQ1

(b) Akkuratheit (FSL vorwärts) | FQ2

(c) Akkuratheit (FSL vorwärts) | FQ2 Fantasie-Datensatz



(d) Δ(keine, FSL vorwärts) | FQ1

(e) Δ(keine, FSL vorwärts) | FQ2

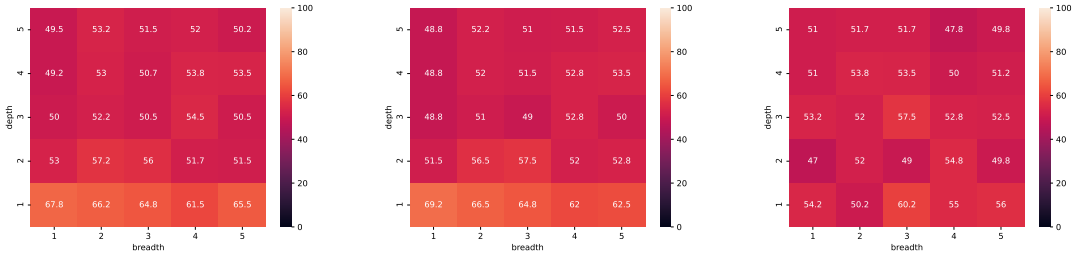
(f) Δ(keine, FSL vorwärts) | FQ2 Fantasie-Datensatz

Abbildung 8.4.: Akkuratheit mit Kontexterweiterungen durch FSL-vorwärts-Elaborationen (a)-(c) und jeweilige Differenzen zu Akkuratheit ohne Kontexterweiterungen (d)-(f)

als entsprechende Vorwärtsbeweise. Dementsprechend ist nicht überraschend, dass GPT-2 selbst generierte FSL-rückwärts-Elaborationen schlechter nutzen kann als selbst generierte FSL-vorwärts-Elaborationen.

Um einen ersten Einblick zu bekommen, was das Sprachmodell aus den Beispielen im Few-Shot-Learning übernimmt, analysiere ich manuell eine Stichprobe der generierten Elaborationen. Die Stichprobe besteht aus den FSL-vorwärts- und FSL-rückwärts-Elaborationen zu 125 zufällig ausgewählten Problemen des *ModifiedChainRuler*-Datensatzes. Sie ist stratifiziert nach Tiefe und Breite der Probleme, enthält also 5 Probleme jeder Komplexität. Diese Stichprobengröße ist nicht ausreichend, um Aussagen über die gesamte Population von 10.000 Elaborationen zu treffen. Die manuelle Analyse dient also nur dazu, eine erste Hypothese über den Einfluss des Few-Shot-Learnings zu bilden. Ich benutze dabei die folgenden Definitionen:

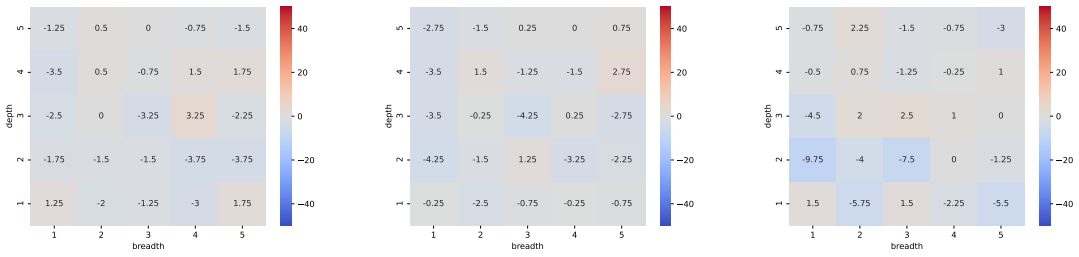
Eine *vorwärtsgerichtete, schlussähnliche Figur* hat die Form „`[Subjekt] is [`P_1`]. If someone is [`P_2`], then they are [`P_3`]. Therefore, [Subjekt] is [`P_4`].“. Eine *rückwärtsgerichtete, schlussähnliche Figur* hat die Form „`[Subjekt] is [`$P_4$`]. That's because if someone is [`$P_2$`], then they are [`$P_3$`]. [Subjekt] is [`$P_1$`].“. Eine schlussähnliche Figur ist genau dann ein *gültiger Schluss*, wenn [P_1]=[P_2] und [P_3]=[P_4] gilt. [P_2] und [P_3] müssen dabei nicht zwangsläufig unterschiedliche Prädikate sein, Tautologien sind also erlaubt. Ein Schluss ist genau dann *wahr*, wenn er gültig ist und aus der Problembeschreibung



(a) Akkuratheit (FSL rückwärts) | FQ1

(b) Akkuratheit (FSL rückwärts) | FQ2

(c) Akkuratheit (FSL rückwärts) | FQ2 Fantasie-Datensatz



(d) Δ(keine, FSL rückwärts) | FQ1

(e) Δ(keine, FSL rückwärts) | FQ2

(f) Δ(keine, FSL rückwärts) | FQ2 Fantasie-Datensatz

Abbildung 8.5.: Akkuratheit mit Kontexterweiterungen durch FSL-rückwärts- Elaborationen (a)-(c) und jeweilige Differenzen zu Akkuratheit ohne Kontexterweiterungen (d)-(f)

folgt. Ein vollständiger Beweis enthält d wahre Schlüsse. Ich untersuche jede Elaboration der Stichprobe nach den folgenden Kriterien:

1. Wie viele schlussähnliche Figuren enthält die generierte Elaboration?
2. Wie viele gültige Schlüsse enthält die generierte Elaboration?
3. Wie viele wahre Schlüsse enthält die generierte Elaboration?

Eine geringe Anzahl schlussähnlicher Figuren deutet darauf hin, dass GPT-2 die Beispiele im Few-Shot-Learning schlecht auf die Aufgaben übertragen kann. Enthalten die generierten Elaborationen mehr schlussähnliche Figuren als gültige Schlüsse, so bedeutet das, dass GPT-2 die syntaktische Struktur der Beispiele nur oberflächlich imitiert. Gültige Schlüsse, die keine wahren Schlüsse sind, deuten darauf hin, dass GPT-2 zwar aus den Beispielen lernt, wie korrekte Schlüsse durchgeführt werden, aber die Problembeschreibung nicht benutzt. Es „erfindet“ in diesem Fall sozusagen Aussagen und Regeln.

Beim Few-Shot-Learning mit Vorwärtsbeweisen bildet das Sprachmodell in dieser Stichprobe insgesamt 202 schlussähnliche Figuren (von 375 möglichen). Davon sind 181 gültige Schlüsse und 69 wahre Schlüsse. Bei 112 FSL-vorwärts-Elaborationen generiert GPT-2 den ersten Satz (den Fakt) wie vorgesehen. Etwa 90% der schlussähnlichen Figuren in dieser Stichprobe sind gültige Schlüsse. Hier zeigt sich wieder, dass GPT-2 gut dazu in

der Lage ist, die Struktur vorwärtsgerichteter Schlüsse zu verallgemeinern. Der Anteil wahrer Schlüsse an den gültigen Schlüssen beträgt allerdings nur etwa 38%. Die restlichen Schlüsse sind zwar korrekt gebildet, folgen aber nicht logisch aus der Problembeschreibung. Betrachten wir ein Beispiel für einen solchen Schluss anhand einer Elaboration aus der Stichprobe:

> Problembeschreibung: „`If someone is careless, then they are rich enough. If someone is full, then they are early. If someone is still, then they are late. If someone is happy, then they are moving. If someone is intelligent, then they are still. Linda is intelligent.`“
>
> Idealer Vorwärtsbeweis: „`Linda is intelligent. If someone is intelligent, then they are still. Therefore, Linda is still. If someone is still, then they are late. Therefore, Linda is late.`“
>
> Von GPT-2 generierte FSL-vorwärts-Elaboration: „`Linda is intelligent. If someone is intelligent, then they are full. Therefore, Linda is full. If someone is full, then they are early. Therefore, Linda is early.`“

Das Sprachmodell generiert hier eine Regel „`If someone is intelligent, then they are full.`“, die nicht aus der Problembeschreibung stammt, und gelangt damit zu der falschen Aussage „`Linda is full.`“. Anschließend verwendet es den ausgleichenden Distraktor „`If someone is full, then they are early.`“, um die falsche Alternative abzuleiten. In dieser Elaboration kommen also zwei gültige Schlüsse vor, die keine wahren Schlüsse sind.

GPT-2 scheint im Few-Shot-Learning also nicht zu erkennen, dass die Regeln in den Beispiel-Beweisen (i) immer den zugehörigen Beispiel-Problembeschreibungen entstammen und (ii) ausschließlich auf Aussagen angewandt werden, die aus der jeweils vorhergehenden Ableitung folgen. Das könnte einen Hinweis darauf liefern, dass GPT-2 nicht genügend Repräsentationsfähigkeiten hat, um diese komplexen Zusammenhänge im Few-Shot-Learning mit zwei Beispielen zu lernen. Hinzu kommt ein umfassenderes Problem mit vortrainierten neuronalen Sprachmodellen: Da sie nur mit unüberwachten Textdaten trainiert werden, können sie ihr Wissen nicht in Beziehung zur Welt setzen. Sie haben kein Konzept einer *Grundwahrheit*, die faktische und fiktiven Aussagen unterscheidet [2][24][20]. In unserer Anwendung soll das Sprachmodell die Problembeschreibung als wahr annehmen. Aussagen, die nicht logisch aus dem gegebenen Kontext folgen, haben für uns nicht den gleichen Wahrheitswert – auch nicht, wenn das Sprachmodell sie generiert. Dass Aussagen sich in ihrem Wahrheitsgehalt unterscheiden können, ist dem Sprachmodell allerdings nicht zugänglich. Es kann nur vorhersagen, wie *wahrscheinlich* es ist, dass der gegebene Kontext mit gewissen Aussagen fortgesetzt wird. Die Überleitung „`In particular, we know that`“ scheint nicht auszureichen, um ihm zu signalisieren, dass es nur Aussagen generieren soll, die logisch aus der Problembeschreibung folgen.

Durch Few-Shot-Learning mit Rückwärtsbeweisen bildet GPT-2 in der gleichen Stichprobe insgesamt 309 schlussähnliche Figuren (von 375 möglichen). Davon sind allerdings nur 102 gültige Schlüsse und 31 wahre Schlüsse. 20 der FSL-rückwärts-Elaborationen enthalten als ersten Satz wie vorgesehen die Konklusion. In dieser Stichprobe beträgt der

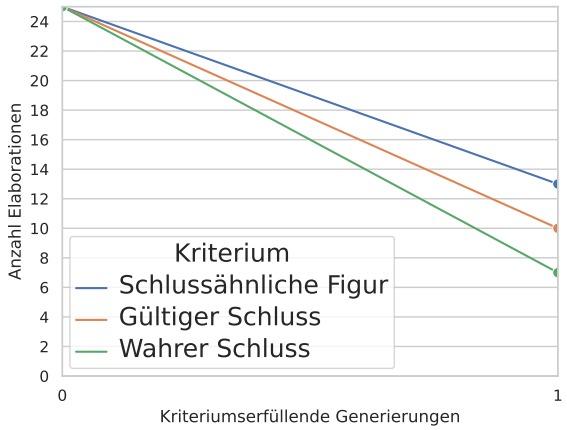
Anteil gültiger Schlüsse an den schlussähnlichen Figuren bei FSL-rückwärts-Elaborationen dementsprechend etwa 33%, er ist also deutlich kleiner als bei FSL-vorwärts-Elaborationen. Dieses Ergebnis legt nahe, dass GPT-2 Schwierigkeiten hat, die Struktur rückwärtsgerichteter Schlüsse korrekt zu erfassen. Betrachten wir dazu ein Beispiel:

> Problembeschreibung: „`If someone is warm, then they are tall. Judith is charming. If someone is unpopular, then they are calm. If someone is charming, then they are angry. If someone is timid, then they are stingy. If someone is angry, then they are generous. If someone is inattentive, then they are secretive.`“
>
> Idealer Rückwärtsbeweis: „`Judith is generous. That's because if someone is angry, then they are generous. Judith is angry. That's because if someone is charming, then they are angry. Judith is charming.`“
>
> Von GPT-2 generierte FSL-rückwärts-Elaboration: „`Judith is charming. That's because if someone is warm, then they are tall. Judith is warm. That's because if someone is unpopular, then they are calm. Judith is calm.`“

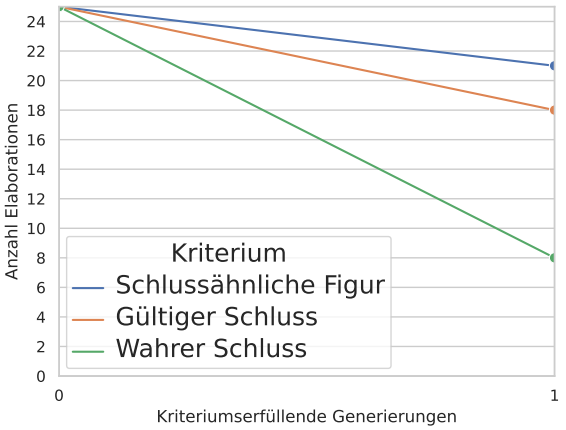
Die von GPT-2 generierte Elaboration enthält 2 schlussähnliche Figuren, aber keine gültigen Schlüsse. Das Modell gibt in beiden Fällen zwar Regeln aus dem Kontext wieder, verwendet diese aber nicht korrekt. Es imitiert die syntaktische Struktur rückwärtsgerichteter Schlüsse nur oberflächlich.

Wie in Abschnitt 7.3.2 diskutiert, kann GPT-2 rückwärtsgerichtete Schlüsse schlechter für die Vorhersage nutzen als vorwärtsgerichtete Schlüsse. Die manuelle Analyse legt nahe, dass es sie zusätzlich auch schlechter generieren kann. Das könnte erklären, wieso Kontexterweiterungen mit FSL-rückwärts-Elaborationen die Akkuratheit auf Problemen fast aller Komplexitäten verschlechtern.

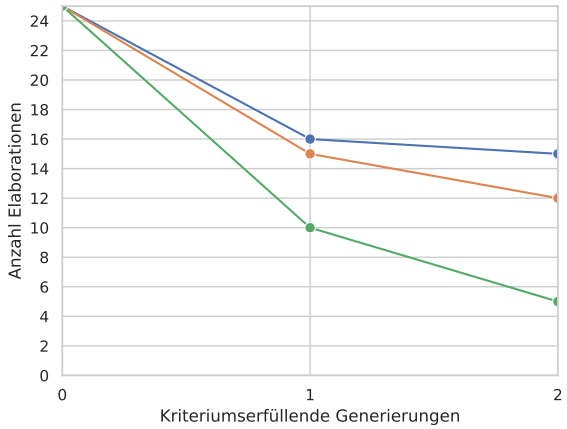
In Grafik 8.6 werden die Ergebnisse der manuellen Analyse veranschaulicht. Die einzelnen Abbildungen zeigen, wie viele Elaborationen einer Tiefe d mindestens k schlussähnliche Figuren, korrekte und gültige Schlüsse enthalten, für $k = 0, ..., d$. In Abbildung 8.6c ist beispielsweise zu sehen, dass in der Stichprobe 16 FSL-vorwärts-Elaborationen für Probleme der Tiefe 2 vorkommen, die mindestens eine schlussähnliche Figur enthalten ($k = 1$). 15 dieser Elaborationen enthalten mindestens einen gültigen Schluss, 10 mindestens einen wahren Schluss. Betrachtet man in dieser Abbildung $k = 2$, so sieht man, dass 13 Elaborationen zwei schlussähnliche Figuren enthalten. 12 Elaborationen enthalten zwei gültige Schlüsse und 5 Elaboration enthalten zwei wahre Schlüsse. Bei der Interpretation dieser Ergebnisse ist zu beachten, dass pro Tiefe nur 25 Probleme ausgewertet werden und Zufallseinflüsse dementsprechend eine bedeutende Rolle spielen. Sie dienen aber zur Visualisierung der beschriebenen Tendenzen: Der Anteil gültiger Schlüsse an den schlussähnlichen Figuren ist bei den FSL-rückwärts-Elaborationen deutlich geringer als bei den FSL-vorwärts-Elaborationen. In beiden Varianten entfällt nur ein kleiner Anteil der generierten schlussähnlichen Figuren auf wahre Schlüsse.



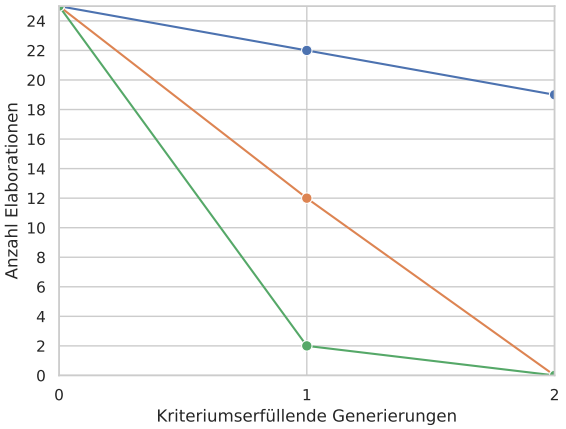
(a) Tiefe 1, FSL vorwärts



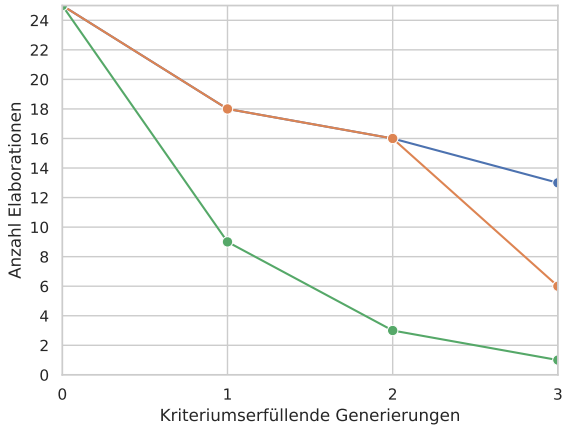
(b) Tiefe 1, FSL rückwärts



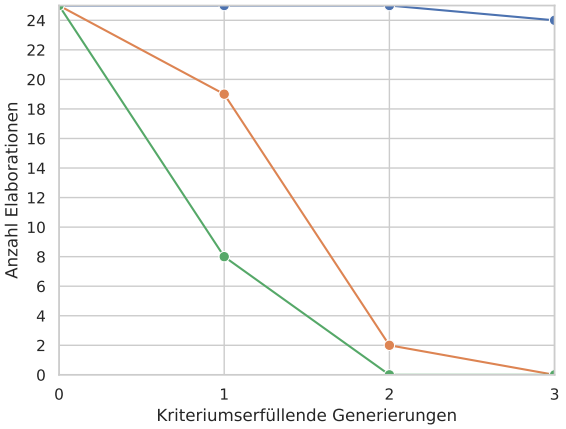
(c) Tiefe 2, FSL vorwärts



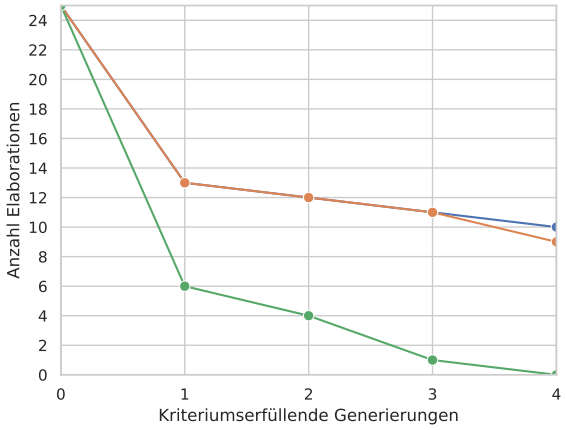
(d) Tiefe 2, FSL rückwärts



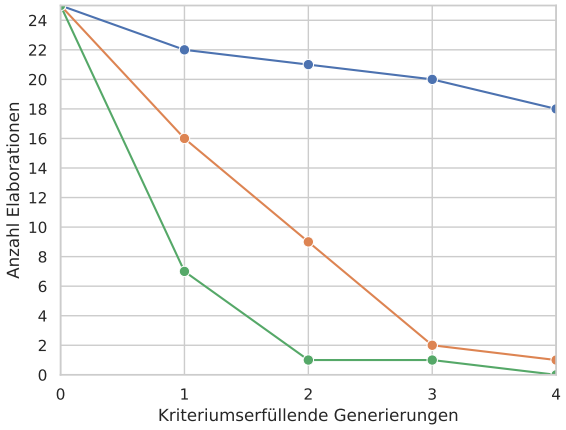
(e) Tiefe 3, FSL vorwärts



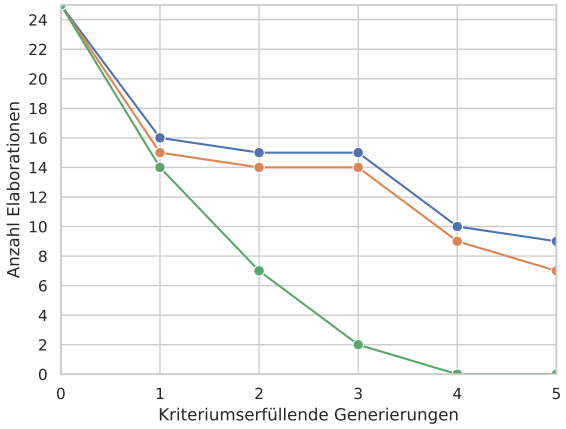
(f) Tiefe 3, FSL rückwärts



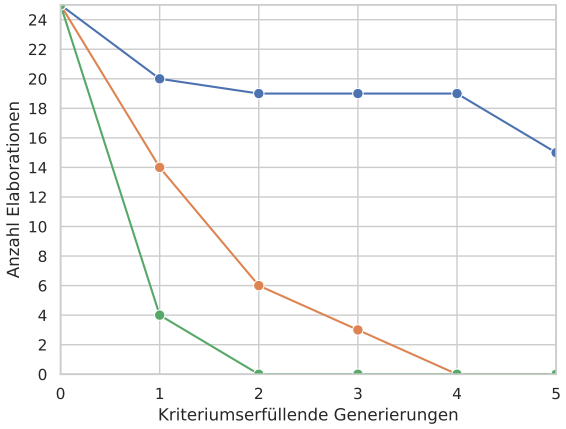
(g) Tiefe 4, FSL vorwärts



(h) Tiefe 4, FSL rückwärts



(i) Tiefe 5, FSL vorwärts



(j) Tiefe 5, FSL rückwärts

Abbildung 8.6.: Anzahl Elaborationen in der Stichprobe, die mindestens k Generierungen enthalten, welche das jeweilige Kriterium erfüllen, für $k = 0, ..., d$. Die Stichprobe besteht aus jeweils 25 Problemen einer Tiefe d, mit 5 Problemen pro Breite. FSL-vorwärts-Elaborationen sind links aufgeführt, FSL-rückwärts-Elaborationen zu den selben Problemen rechts. Kriterien der Analyse sind schlussähnliche Figuren (blau), gültige Schlüsse (orange) und wahre Schlüsse (grün).

9. Zusammenfassung und weiterführende Arbeit

Ich stelle in dieser Arbeit *ModifiedChainRuler* vor – einen Datensatz mit natürlichsprachlichen, deduktiven Schlussfolgerungsproblemen. *ModifiedChainRuler* ist angelehnt an den *ChainRuler*-Datensatz von Betz et al., umgeht aber durch seine Konstruktion dessen Schwierigkeiten (siehe Kapitel 5). Wie ich in Abschnitt 5.5 zeige, ist der *ModifiedChainRuler*-Datensatz dazu geeignet, die Schlussfolgerungsfähigkeiten neuronaler Sprachmodelle zu testen. Mit einer Variation dieses Datensatzes, dem *ModifiedChainRuler*-Fantasie-Datensatz, kann außerdem untersucht werden, ob Sprachmodelle in der Lage sind, Schlussfolgerungen über erfundene Prädikate zu ziehen (Abschnitt 5.6).

Anhand dieser Datensätze zeige ich, dass GPT-2 ohne Finetuning moderate Schlussfolgerungsfähigkeiten auf Problemen hat, die nur wenige deduktive Ableitungen erfordern. Bei einfachen Problemen erreicht GPT-2 im Zero-Shot-Setting eine Korrektklassifizierungsrate von bis zu 69,5%. Mit Problemen, bei denen mehrere Schlussfolgerungsschritte nötig sind, um zur richtigen Lösung zu gelangen, hat GPT-2 allerdings Schwierigkeiten. Seine Performanz ist in diesen Fällen nur unwesentlich besser als zufälliges Raten (siehe Abschnitt 6.2). Wie ich in Abschnitt 6.2.2 und Abschnitt 7.2.4 demonstriere, kann GPT-2 sogar Schlussfolgerungen über erfundene Prädikate vollziehen, sofern es zuvor genug Sätze gesehen hat, in denen solche Fantasiewörter die syntaktische Funktion von Prädikaten erfüllen.

In Abschnitt 7.1 demonstriere ich, dass Kontexterweiterungen die Akkuratheit der Vorhersage stark beeinflussen können. Auf dem *ModifiedChainRuler*-Fantasie-Datensatz haben synthetische Kontexterweiterungen stärkere Effekte als auf dem *ModifiedChainRuler*-Datensatz. Die Vorhersage des Sprachmodells verbessert sich merklich, wenn es die Zwischenergebnisse zu den Problemen gegeben bekommt (Abschnitt 7.2). Das ist nicht verwunderlich, da sich die Komplexität der Aufgaben durch solche Kontexterweiterungen deutlich reduziert. Mit Zwischenschluss-Kontexterweiterungen ist unabhängig von der Tiefe des Problems nur noch eine Ableitung nötig, um zur richtigen Lösung zu gelangen. Diesem Gedanken folgend vergleiche ich die erweiterten Probleme in Abschnitt 7.2.1 mit Problemen der Tiefe 1, welche die gleichen Gesamtzahl an Regeln haben. Mithilfe eines geeigneten Maßes (Abschnitt 7.2.2) zeige ich, dass das Sprachmodell Schlussfolgerungsprobleme schlechter lösen kann, wenn die darin enthaltenen Aussagen stärker miteinander verknüpft sind. Eine mögliche Erklärung für diesen Effekt liegt in der Architektur von GPT-2 (siehe Abschnitt 7.2.3).

Wie ich in Abschnitt 7.3 zeige, ist GPT-2 sehr gut dazu in der Lage, die Struktur vorgegebener Beweise zu verallgemeinern und selbstständig weiterzuführen. Durch Kontexterweiterungen mit synthetischen Beweisen lassen sich Korrektklassifizierungsraten von bis zu 100% erreichen. Indem ich verschiedene Einflüsse isoliere, weise ich nach, dass die Verbesserungen zum Teil darauf zurückzuführen sind, dass das Sprachmodell eine einfache Heuristik einsetzt oder die Lösung memorisiert. Wenn man diese unerwünschten Effekte unterbindet, verbessern vorgegebene Beweise die Vorhersage trotzdem noch stark. Daraus schließe ich, dass GPT-2 in der Lage ist, die Beweisschritte korrekt zu interpretieren und auszunutzen. Dass die relevanten Regeln geordnet und in die richtige Reihenfolge gebracht werden, scheint die Vorhersagen des Sprachmodells ebenfalls zu verbessern. Diese Beobachtung kann in zukünftiger Arbeit systematisch untersucht werden: Wie groß ist der Einfluss, den die Reihenfolge der Sätze auf die Korrektklassifizierungsrate hat? Verbessern sich die Schlussfolgerungsfähigkeiten eines Sprachmodells, wenn es darauf trainiert wird, den gegebenen Kontext zu sortieren, bevor es eine Antwort vorhersagt?

In Abschnitt 7.3.1.2 zeigt sich eine Schwierigkeit bei der Verwendung neuronaler Sprachmodelle: Es ist nicht möglich, zu überprüfen, was ein Sprachmodell aus einem gegebenen Text *weiß*, sondern nur, wie es diesen *fortsetzen* würde. Eine finale Frage wie „`Therefore,`“ oder „`Therefore, we know that`“ gibt dem Sprachmodell keinen Anhaltspunkt, welche der möglichen Schlussfolgerungen aus dem Kontext gefragt ist – die Konklusion oder einer der Zwischenschlüsse. Gegebene Vorwärtsbeweise demonstrieren dem Sprachmodell im Gegensatz zu Rückwärtsbeweisen, dass die letztmögliche Schlussfolgerung aus dem Kontext die gesuchte Lösung ist. Interessante Forschungsfragen für eine zukünftige Arbeit sind daher: Welche Wahrscheinlichkeiten berechnet das Sprachmodell für Zwischenschlüsse und deren konzeptuelle Komplemente? Wie kann die finale Frage umformuliert werden, um dem Sprachmodell zu vermitteln, welche Aussage die gesuchte Lösung ist?

Wie ich in Kapitel 8 gezeigt habe, beeinflussen dynamisch vom Sprachmodell generierte Problemelaborationen die Korrektklassifizierungsrate nur leicht. Im Zero-Shot-Setting untersuche ich freie, strukturierte und iterative Elaborationen (Abschnitt 8.1). Außerdem betrachte ich Kontexterweiterungen, die durch Few-Shot-Learning mit je zwei Beispielen von vorwärtsgerichteten oder rückwärtsgerichteten Beweisen erstellt wurden (Abschnitt 8.3). Freie und strukturierte Elaborationen verschlechtern die Vorhersage auf *ModifiedChainRuler*-Problemen der Tiefe 1 und führen auf Problemen höherer Tiefen teilweise zu leichten Verbesserungen. Diese Effekte lassen sich durch wahrscheinlichkeitstheoretische Überlegungen erklären (siehe Abschnitt 8.2). Iterative Elaborationen funktionieren im Vergleich dazu minimal besser. GPT-2 scheint bereits korrekt generierte Aussagen teilweise nutzen zu können, um daraus weitere korrekte Schlüsse zu ziehen. Auf dem *ModifiedChainRuler*-Fantasie-Datensatz haben dynamische Elaborationen, die das Modell im Zero-Shot-Setting erstellt, durchwachsene Effekte.

Few-Shot-Learning mit Rückwärtsbeweisen verschlechtert die Qualität der dynamisch generierten Elaborationen. Few-Shot-Learning mit Vorwärtsbeweisen führt hingegen zu leichten Verbesserungen (siehe Abschnitt 8.4). Hier zeigt sich wieder, dass das Sprachmodell die Struktur vorwärtsgerichteter Schlussfolgerungen gut generalisieren kann. Am besten funktioniert Few-Shot-Learning mit Vorwärtsbeweisen auf Problemen hoher Tiefe. Eine

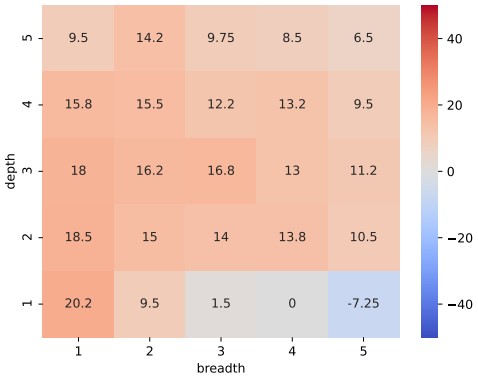
mögliche Erklärung dafür ist, dass das Sprachmodell eine gewisse Anzahl an Ableitungen „gesehen“ haben muss, um sie selbst durchführen zu können. In einer weiterführenden Arbeit kann untersucht werden, ob Few-Shot-Learning mit zusätzlichen Beispielen einen stärkeren Effekt hat. Alternativ könnte man nur Beispiel-Probleme der Tiefe 5 verwenden – unabhängig von der Komplexität der eigentlichen Aufgabe.

Kontexterweiterungen, die mithilfe von Few-Shot-Learning generiert wurden, verbessern die Akkuratheit im besten Fall um 6,75 absolute Prozentpunkte. Sie haben also einen deutlich schwächeren Effekt als vorgegebene Kontexterweiterungen. GPT-2 scheint einzelne Beweisschritte besser imitieren zu können als die Struktur eines vollständigen Beweises. Um die Diskrepanz zwischen der Wirksamkeit vorgegebener Beweise und dynamisch generierter Elaborationen zu überbrücken, könnte man in einer weiterführenden Arbeit folgendermaßen vorgehen: Zunächst wird jede Problembeschreibung durch eine feste Anzahl z zusätzlicher Regeln erweitert. Die Regelkette hat dann die Länge $(d + z)$. Nach der finalen Frage enthält der Kontext die ersten z Ableitungsschritte des vorwärtsgerichteten Beweises. Wie verändern sich die Schlussfolgerungsfähigkeiten des Sprachmodells, wenn es zuvor z Schlussfolgerungen gesehen hat? Gibt es eine bestimmte Anzahl solcher Demonstrationen, ab der es Probleme beliebiger Tiefe lösen kann?

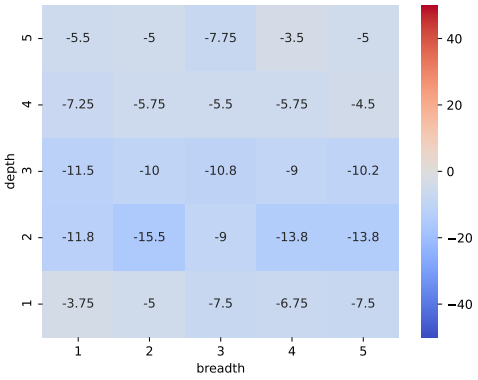
Die manuelle Analyse einer Stichprobe von je 125 im FSL erstellten Elaborationen gibt einen Hinweis darauf, dass vorwärtsgerichtete Schlüsse für das Modell leichter zu generieren sind als rückwärtsgerichtete Schlüsse. Außerdem sind etwa 62% der gültigen Schlüsse in den analysierten FSL-vorwärts-Elaborationen und etwa 70% der gültigen Schlüsse in FSL-rückwärts-Elaborationen nicht aus der Problembeschreibung ableitbar. Das Sprachmodell „erfindet“ also Argumente oder Fakten, um Schlüsse zu bilden.

Neuronale Sprachmodelle haben kein Konzept von einer Grundwahrheit, welche den sprachlichen Ausdrücken zugrunde liegt [24][2][20]. Dass Aussagen sich in ihrem Wahrheitsgehalt unterscheiden können, ist ihnen nicht zugänglich. Daher ist es herausfordernd, einem Sprachmodell zu signalisieren, dass es die gegebenen Aussagen *als wahr annehmen* und nur aus diesen Aussagen weitere Schlussfolgerungen ziehen soll. Eine Möglichkeit dafür könnte sein, den Prompt für die Textgenerierung (hier „`In particular, we know that`“) umzuformulieren, beispielsweise zu „`Without any additional information, we know that`“. In einer weiterführenden Arbeit kann untersucht werden, welche Prompts geeignet sind, den Anteil wahrer Schlüsse in den generierten gültigen Schlüssen zu erhöhen.

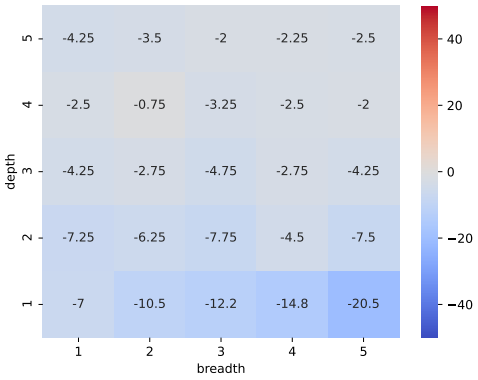
A. Abbildungen



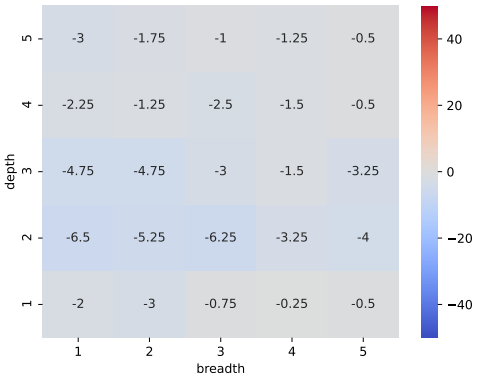
(a) Δ(Vorwärtsbeweis, Vorwärtsbeweis maskiert) | FQ1



(b) Δ(Vorwärtsbeweis maskiert, Vorwärtsbeweis maskiert und ausgeglichen) | FQ1

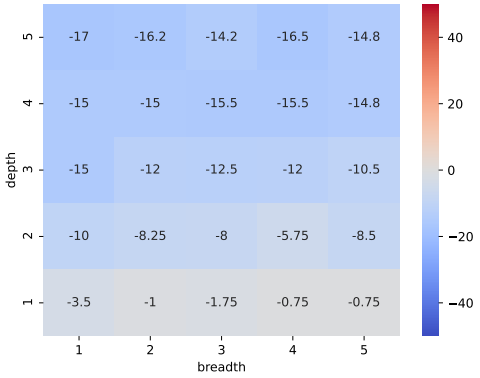


(c) Δ(Vorwärtsbeweis, Vorwärtsbeweis maskiert) | FQ2

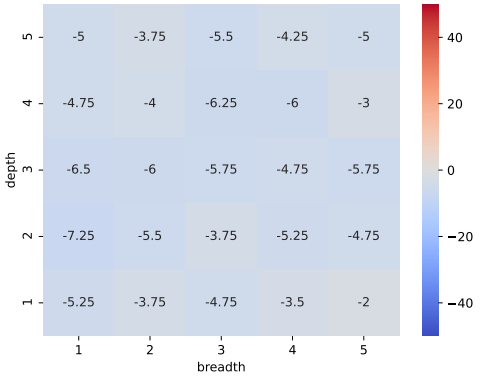


(d) Δ(Vorwärtsbeweis maskiert, Vorwärtsbeweis maskiert und ausgeglichen) | FQ2

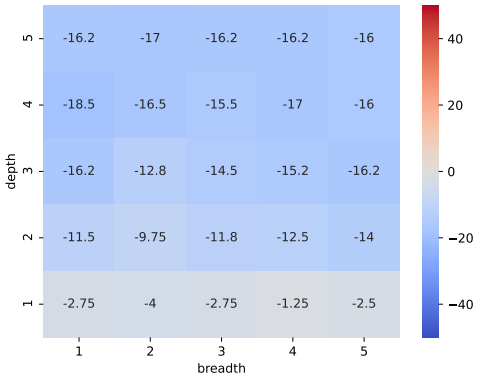
Abbildung A.1.: Veränderungen der Akkuratheit durch (a) Maskieren der Antwort und (b) Ausgleichen der Elaboration bei Vorwärtsbeweisen auf dem *ModifiedChainRuler*-Datensatz



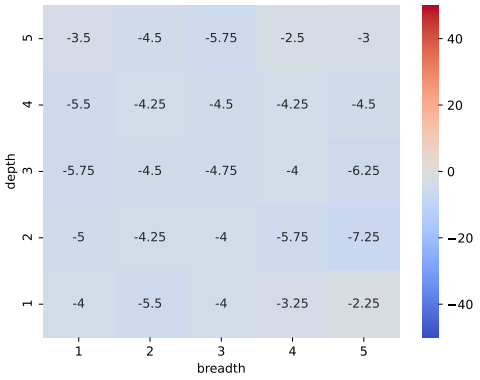
(a) Δ(Rückwärtsbeweis, Rückwärtsbeweis maskiert) | FQ1



(b) Δ(Rückwärtsbeweis maskiert, Rückwärtsbeweis maskiert und ausgeglichen) | FQ1

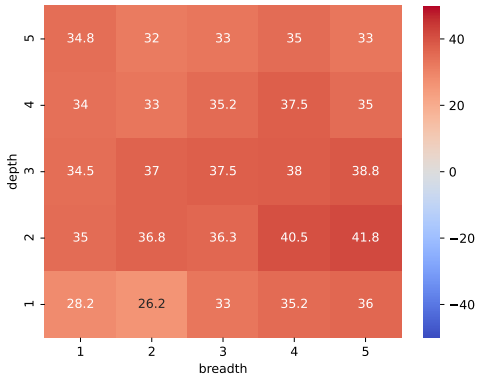


(c) Δ(Rückwärtsbeweis, Rückwärtsbeweis maskiert) | FQ2

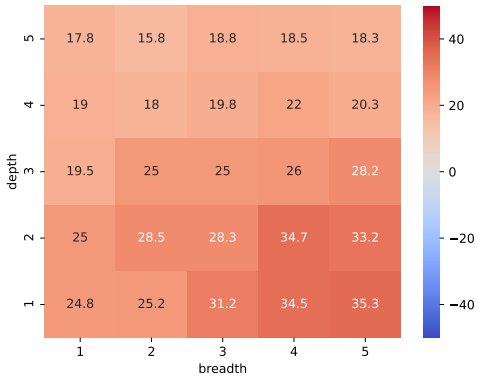


(d) Δ(Rückwärtsbeweis maskiert, Rückwärtsbeweis maskiert und ausgeglichen) | FQ2

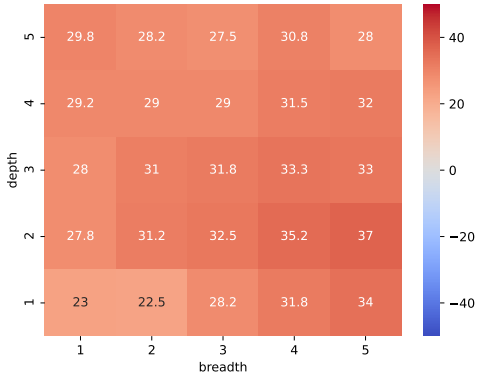
Abbildung A.2.: Veränderungen der Akkuratheit durch (a) Maskieren der Antwort und (b) Ausgleichen der Elaboration bei Rückwärtsbeweisen auf dem *ModifiedChainRuler*-Datensatz



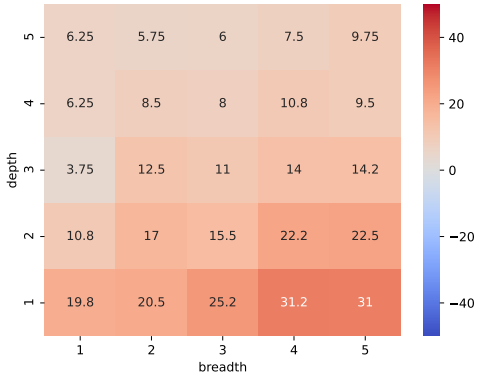
(a) Δ(keine, Rückwärtsbeweis) | FQ1



(b) Δ(keine, Rückwärtsbeweis maskiert) | FQ1

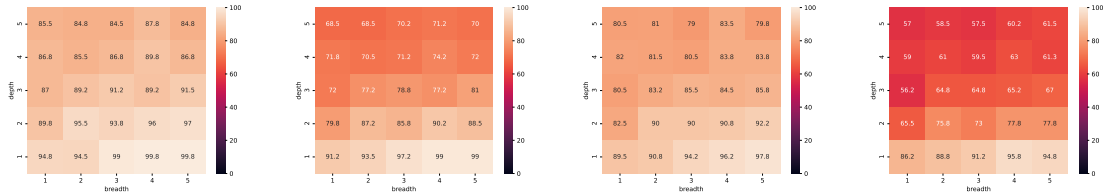


(c) Δ(keine, Rückwärtsbeweis ausgeglichen) | FQ1



(d) Δ(keine, Rückwärtsbeweis maskiert und ausgeglichen) | FQ1

Abbildung A.3.: Differenz zwischen Akkuratheit mit Elaboration und Akkuratheit ohne Elaboration, jeweils für folgende Elaborationen: (a) Rückwärtsbeweis, (b) maskierter Rückwärtsbeweis, (c) ausgeglichener Rückwärtsbeweis, (d) maskierter und ausgeglichener Rückwärtsbeweis

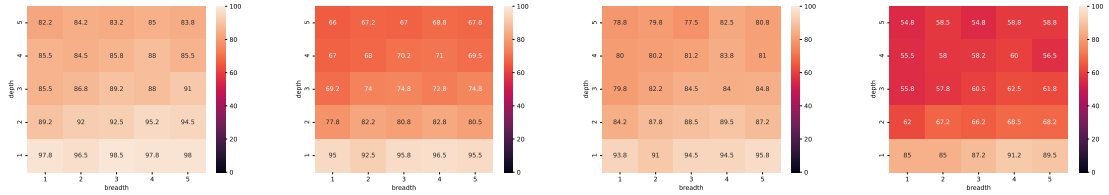


(a) Akkuratheit (Rückwärts-beweis) | FQ1

(b) Akkuratheit (Rückwärtsbeweis maskiert) | FQ1

(c) Akkuratheit (Rückwärtsbeweis ausgeglichen) | FQ1

(d) Akkuratheit (Rückwärtsbeweis maskiert und ausgeglichen) | FQ1

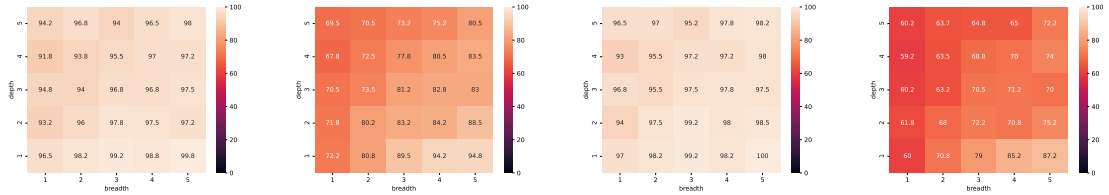


(e) Akkuratheit (Rückwärts-beweis) | FQ2

(f) Akkuratheit (Rückwärtsbeweis maskiert) | FQ2

(g) Akkuratheit (Rückwärtsbeweis ausgeglichen) | FQ2

(h) Akkuratheit (Rückwärtsbeweis maskiert und ausgeglichen) | FQ2



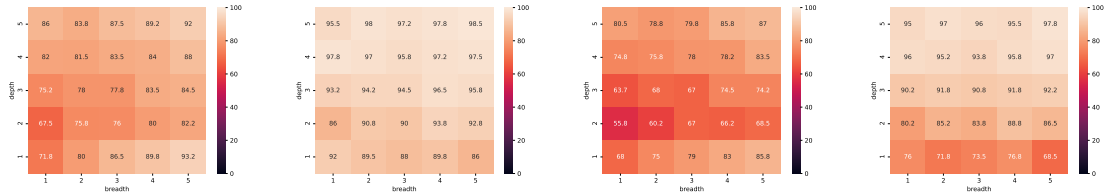
(i) Akkuratheit (Rückwärts-beweis) | FQ2 Fantasie-Datensatz

(j) Akkuratheit (Rückwärtsbeweis maskiert) | FQ2 Fantasie-Datensatz

(k) Akkuratheit (Rückwärtsbeweis ausgeglichen) | FQ2 Fantasie-Datensatz

(l) Akkuratheit (Rückwärtsbeweis maskiert und ausgeglichen) | FQ2 Fantasie-Datensatz

Abbildung A.4.: Akkuratheit mit Rückwärtsbeweis (erste Spalte), maskiertem Rückwärtsbeweis (zweite Spalte), ausgeglichenem Rückwärtsbeweis (dritte Spalte), maskiertem und ausgeglichenem Rückwärtsbeweis (vierte Spalte), jeweils für finale Fragen FQ1 ((a)-(d)) und FQ2 ((e)-(h)) auf dem *ModifiedChainRuler*-Datensatz und für FQ2 auf dem *ModifiedChainRuler*-Fantasie-Datensatz ((i)-(l))

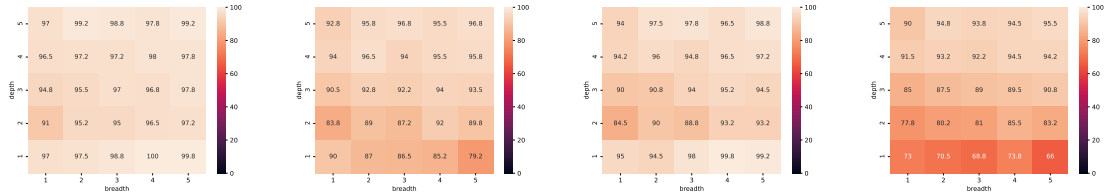


(a) Akkuratheit (Vorwärtsbeweis) | FQ1

(b) Akkuratheit (Vorwärtsbeweis maskiert) | FQ1

(c) Akkuratheit (Vorwärtsbeweis ausgeglichen) | FQ1

(d) Akkuratheit (Vorwärtsbeweis maskiert und ausgeglichen) | FQ1

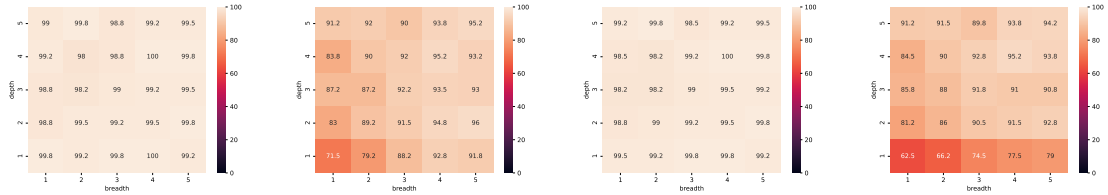


(e) Akkuratheit (Vorwärtsbeweis) | FQ2

(f) Akkuratheit (Vorwärtsbeweis maskiert) | FQ2

(g) Akkuratheit (Vorwärtsbeweis ausgeglichen) | FQ2

(h) Akkuratheit (Vorwärtsbeweis maskiert und ausgeglichen) | FQ2



(i) Akkuratheit (Vorwärtsbeweis) | FQ2 Fantasie-Datensatz

(j) Akkuratheit (Vorwärtsbeweis maskiert) | FQ2 Fantasie-Datensatz

(k) Akkuratheit (Vorwärtsbeweis ausgeglichen) | FQ2 Fantasie-Datensatz

(l) Akkuratheit (Vorwärtsbeweis maskiert und ausgeglichen) | FQ2 Fantasie-Datensatz

Abbildung A.5.: Akkuratheit mit Vorwärtsbeweis (erste Spalte), maskiertem Vorwärtsbeweis (zweite Spalte), ausgeglichenem Vorwärtsbeweis (dritte Spalte), maskiertem und ausgeglichenem Vorwärtsbeweis (vierte Spalte), jeweils für finale Fragen FQ1 ((a)-(d)) und FQ2 ((e)-(h)) auf dem *ModifiedChainRuler*-Datensatz und für FQ2 auf dem *ModifiedChainRuler*-Fantasie-Datensatz ((i)-(l))

beautiful
intelligent
charming

(a) Quell-Prädikate

awake	sleeping
hungry	full
warm	cold
attentive	inattentive
happy	sad
moving	still
in need of money	rich enough
careful	careless
humble	proud
joking	serious

(b) Gruppe 1

friendly	hostile
old	young
big	small
popular	unpopular
brave	timid
healthy	sick
blunt	secretive
angry	calm
standing	sitting
tired	fresh

(c) Gruppe 2

Tabelle A.1.: Prädikate zum Synthetisieren des *ModifiedChainRuler*-Datensatzes

blad
foaby
dreen

(a) Quell-Prädikate

oching
potaugh
baly
fanough
wouring
soaly
rakouming
benoosy
trouby
blodely
patoozing
pawounig
canimery
croumy
steaby
laphiful
calooning
paritting

(b) Gruppe 1

wooking
parouly
funely
stold
lanig
wouring
soaful
gloosy
maming
cadeling
gouby
lirful
patouny
toumy
gotaugh
conill
gleered
freebing

(c) Gruppe 2

Tabelle A.2.: Prädikate zum Synthetisieren des *ModifiedChainRuler*-Fantasie-Datensatzes

Literatur

[1] Emily M. Bender und Alexander Koller. „Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data“. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Juli 2020, S. 5185–5198. DOI: 10.18653/v1/2020.acl-main.463. URL: https://aclanthology.org/2020.acl-main.463.

[2] Emily M. Bender u. a. „On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?“ In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (2021).

[3] Gregor Betz, Kyle Richardson und Christian Voigt. *Thinking Aloud: Dynamic Context Generation Improves Zero-Shot Reasoning Performance of GPT-2*. 2021. arXiv: 2103.13033 [cs.CL].

[4] Gregor Betz, Christian Voigt und Kyle Richardson. *Critical Thinking for Language Models*. 2020. arXiv: 2009.07185 [cs.CL].

[5] Tom B. Brown u. a. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[6] Kevin Clark u. a. *What Does BERT Look At? An Analysis of BERT's Attention*. 2019. arXiv: 1906.04341 [cs.CL].

[7] Peter Clark, Oyvind Tafjord und Kyle Richardson. *Transformers as Soft Reasoners over Language*. 2020. arXiv: 2002.05867 [cs.CL].

[8] Zihang Dai u. a. *Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context*. 2019. arXiv: 1901.02860 [cs.LG].

[9] Jacob Devlin u. a. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[10] Charlotte Dignath und Gerhard Büttner. „Components of fostering self-regulated learning among students“. In: *Metacognition Learning* 3 (2008), S. 231–264. DOI: https://doi.org/10.1007/s11409-008-9029-x.

[11] Arthur K. Ellis, David W. Denton und John B. Bond. „An Analysis of Research on Metacognitive Teaching Strategies“. In: *Procedia - Social and Behavioral Sciences* 116 (2014). 5th World Conference on Educational Sciences, S. 4015–4024. ISSN: 1877-0428. DOI: https://doi.org/10.1016/j.sbspro.2014.01.883. URL: https://www.sciencedirect.com/science/article/pii/S1877042814009008.

[12] Allyson Ettinger. *What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models*. 2020. arXiv: 1907.13528 [cs.CL].

[13] Allyson Ettinger u. a. „Assessing Composition in Sentence Vector Representations“. In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, S. 1790–1801. URL: https://aclanthology.org/C18-1152.

[14] Nicolas Gontier u. a. *Measuring Systematic Generalization in Neural Proof Generation with Transformers*. 2020. arXiv: 2009.14786 [cs.LG].

[15] Ari Holtzman u. a. *The Curious Case of Neural Text Degeneration*. 2020. arXiv: 1904.09751 [cs.CL].

[16] Benjamin Hoover, Hendrik Strobelt und Sebastian Gehrmann. „exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models“. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, Juli 2020, S. 187–196. URL: https://www.aclweb.org/anthology/2020.acl-demos.22.

[17] Md Mosharaf Hossain u. a. *It's not a Non-Issue: Negation as a Source of Error in Machine Translation*. 2020. arXiv: 2010.05432 [cs.CL].

[18] Liwei Jiang u. a. „"I'm Not Mad": Commonsense Implications of Negation and Contradiction“. In: *NAACL*. 2021.

[19] Zhengbao Jiang u. a. *How Can We Know What Language Models Know?* 2020. arXiv: 1911.12543 [cs.CL].

[20] Nora Kassner und Hinrich Schütze. „Negated and Misprimed Probes for Pretrained Language Models: Birds Can Talk, But Cannot Fly“. In: *The 58th Annual Meeting of the Association for Computational Linguistic*. Stroudsburg, USA: Association for Computational Linguistics, 2020. URL: http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-72195-2.

[21] M. Kejriwal und Ke Shen. „Do Fine-tuned Commonsense Language Models Really Generalize?“ In: *ArXiv* abs/2011.09159 (2020).

[22] Najoung Kim u. a. „Probing What Different NLP Tasks Teach Machines about Function Word Comprehension“. In: *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*. Minneapolis, Minnesota: Association for Computational Linguistics, Juni 2019, S. 235–249. DOI: 10.18653/v1/S19-1026. URL: https://aclanthology.org/S19-1026.

[23] Yinhan Liu u. a. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL].

[24] William Merrill u. a. „Provable Limitations of Acquiring Meaning from Ungrounded Form: What will Future Language Models Understand?“ In: *ArXiv* abs/2104.10809 (2021).

[25] Timothy Niven und Hung-Yu Kao. *Probing Neural Network Comprehension of Natural Language Arguments*. 2019. arXiv: 1907.07355 [cs.CL].

[26] Fabio Petroni u. a. *How Context Affects Language Models' Factual Predictions*. 2020. arXiv: 2005.04611 [cs.CL].

[27] Fabio Petroni u. a. *Language Models as Knowledge Bases?* 2019. arXiv: 1909.01066 [cs.CL].

[28] XiPeng Qiu u. a. „Pre-trained models for natural language processing: A survey“. In: *Science China Technological Sciences* 63.10 (Sep. 2020), S. 1872–1897. ISSN: 1869-1900. DOI: 10.1007/s11431-020-1647-3. URL: http://dx.doi.org/10.1007/s11431-020-1647-3.

[29] A. Radford. „Improving Language Understanding by Generative Pre-Training“. In: 2018.

[30] A. Radford u. a. „Language Models are Unsupervised Multitask Learners“. In: 2019.

[31] Colin Raffel u. a. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2020. arXiv: 1910.10683 [cs.LG].

[32] Kyle Richardson und Ashish Sabharwal. *What Does My QA Model Know? Devising Controlled Probes using Expert Knowledge*. 2020. arXiv: 1912.13337 [cs.CL].

[33] Kyle Richardson u. a. *Probing Natural Language Inference Models through Semantic Fragments*. 2019. arXiv: 1909.07521 [cs.CL].

[34] Anna Rogers, Olga Kovaleva und Anna Rumshisky. *A Primer in BERTology: What we know about how BERT works*. 2020. arXiv: 2002.12327 [cs.CL].

[35] Swarnadeep Saha u. a. *PRover: Proof Generation for Interpretable Reasoning over Rules*. 2020. arXiv: 2010.02830 [cs.CL].

[36] Rico Sennrich, Barry Haddow und Alexandra Birch. *Neural Machine Translation of Rare Words with Subword Units*. 2016. arXiv: 1508.07909 [cs.CL].

[37] Vered Shwartz u. a. *Unsupervised Commonsense Question Answering with Self-Talk*. 2020. arXiv: 2004.05483 [cs.CL].

[38] Oyvind Tafjord, Bhavana Dalvi und P. Clark. „ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language“. In: *ArXiv* abs/2012.13048 (2021).

[39] Alon Talmor u. a. *Leap-Of-Thought: Teaching Pre-Trained Models to Systematically Reason Over Implicit Knowledge*. 2020. arXiv: 2006.06609 [cs.CL].

[40] Ian Tenney, Dipanjan Das und Ellie Pavlick. „BERT Rediscovers the Classical NLP Pipeline“. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Juli 2019, S. 4593–4601. DOI: 10.18653/v1/P19-1452. URL: https://aclanthology.org/P19-1452.

[41] Ashish Vaswani u. a. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[42] Changhan Wang, Kyunghyun Cho und Jiatao Gu. *Neural Machine Translation with Byte-Level Subwords*. 2019. arXiv: 1909.03341 [cs.CL].

[43] Yaqing Wang u. a. *Generalizing from a Few Examples: A Survey on Few-Shot Learning*. 2020. arXiv: 1904.05046 [cs.LG].

[44] Yile Wang, Leyang Cui und Yue Zhang. *How Can BERT Help Lexical Semantics Tasks?* 2020. arXiv: 1911.02929 [cs.CL].

[45] Thomas Wolf u. a. *HuggingFace's Transformers: State-of-the-art Natural Language Processing*. 2020. arXiv: 1910.03771 [cs.CL].

[46] Ziang Xie. *Neural Text Generation: A Practical Guide*. 2017. arXiv: 1711.09534 [cs.CL].

[47] Hitomi Yanaka u. a. *Can neural networks understand monotonicity reasoning?* 2019. arXiv: 1906.06448 [cs.CL].