Leo Steiner, 7906565, leo_steiner@me.com

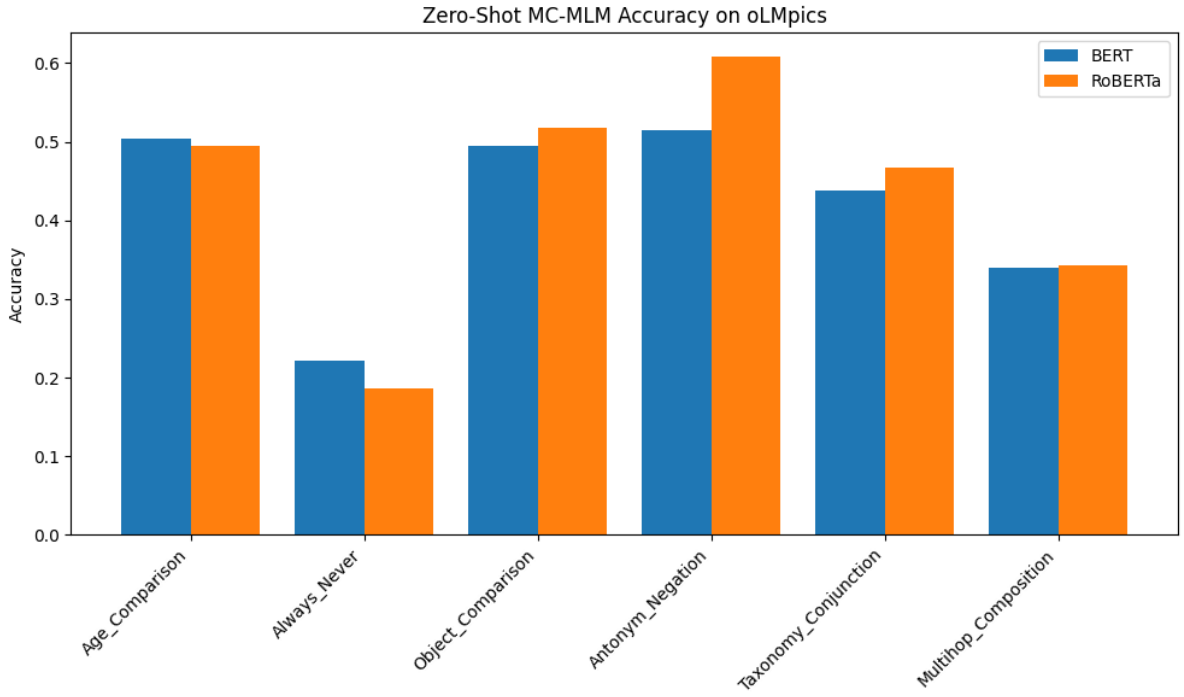
NLP Übung 2

Aufgabe 1

Aufgabe 1.1:

- **Experimentaufbau**: Wie in 3.1 von Talmor et al. (im folgenden als “Paper” bezeichnet) definieren wir für jede Aufgabe ein Statement mit genau einem [MASK] und werten es zero-shot aus . Als Modelle kommen zum einen BERT(bert-base-cased) und anderen RoBERTa(roberta-base) zum Einsatz. Die sechs Testvarianten sind:
  - Age_Comparison
  - Always_Never
  - Object_Comparison
  - Antonym_Negation
  - Taxonomy_Conjunction
  - Multihop_Composition

Für jede Variante wird die Accuracy korrekter Vorhersagen berechnet.



Auswertung Accuracy Aufgabe 1.1

Variante	Zufalls-Baseline	BERT-Base (Eigene Auswertung)	RoBERTa-Base (Eigene Auswertung)	RoBERTa-Large (Paper)	BERT-WWM (Paper)
Age_Comparison	50 %	50.40 %	49.40 %	98 %	70 %
Always_Never	20 %	22.14 %	18.57 %	14 %	10 %
Object_Comparison	50 %	49.40 %	51.80 %	84 %	55 %
Antonym_Negation	50 %	51.40 %	60.80 %	75 %	57 %
Taxonomy_Conjunction	33 %	43.74 %	46.74 %	45 %	46 %
Multihop_Composition	33 %	34.00 %	34.20 %	29 %	33 %

- **Analyse je Set**:
  - **2.1 Age_Comparison**
    - **Ergebnis**: BERT 50.40 %, RoBERTa 49.40 % (Zufall 50 %—2 Antwortmöglichkeiten)
    - **Large-Model-Benchmark**: RoBERTa-Large erreicht im Paper Zeor-Shot fast 98 % .
    - **Ursachen**:
      - **Wertebereich**: Paper: RoBERTa-L kann numerische Vergleiche nur dann korrekt durchführen, wenn die Zahlen innerhalb des bei Pre-Training häufig vorkommenden Altersbereichs liegen; außerhalb bricht die Performance ein. Dieses Phänomen wird im Paper bei Geburtstagsjahren 1920–2000 und folglich den Zahlen (15-105) beschrieben.
        - Bei Durchsicht der Testdaten wird aber klar, dass dies vorliegend keinen Einfluss auf das mäßige Ergebnis haben kann da die Test-Daten allesamt in des oben genannten Zahlenintervalls liegen.
      - **Tokenisierung von Zahlen**: Dies könnte ein Grund für die schlechte Performance der base- Modelle sein sein: Die Grundversionen von BERT/ RoBERTa splitten Zahlen manchmal in Sub-Wort-Einheiten, was die

Repräsentation ungenauer macht, besonders für höhere (und folglich längere) Zahlen außerhalb der Trainingsbeispiele .

- **Zu kleine Datenmenge**: Die Base-Modelle wurden auf zu kleinen Datenmenge trainiert, um besser als bloßes raten ca 50% zu erreichen. Dies würde auch das im vergleich zu den großen Modelle schlechte Ergebnis erklären.

- **2.2 Always_Never**
  - **Ergebnis**: BERT 22.14 %, RoBERTa 18.57 % (Zufall 20 % — 5 Antwortmöglichkeiten)
  - **Large-Model-Benchmark**: RoBERTa-Large 14 %, BERT_WMM .
    Diese Performance wird im Paper auf folgendes zurückgeführt
    - Reporting Bias: “Sometimes” kämmt häufiger vor als “always” oder “never”. -> Systematisch falsche Aussage.
  - Ursachen:
    - **Geringere Reporting Bias**: Bei den hier verwendeten kleinere Modellen, kommt diese Geringere Reporting Bias möglicherweise zum tragen, da die Modell ungefähr die Performance des zufälligen Ratens erreichen - also in diesem Kontext “unbiased” sind.
- **2.3 Object_Comparison**
  - **Ergebnis**: BERT 49.40 %, RoBERTa 51.80 % (Zufall 50 % — 2 Antwortmöglichkeiten)
  - **Large-Model-Benchmark**: RoBERTa-Large ca. 84 % zero-shot Bert WWM 55%.
    Offensichtlich liefert der große Datensatz für Roberta-L genug Kontext um Objektgrößen zu Encoden, wohingegen das WWM-Prozedere diesen Kontext offensichtlich kaum bis gar nicht hergibt. Bert schafft es gerade so Zufallsraten zu schlagen.
  - Ursachen:
    - **Mangel an Relationalem Reasoning**: Vergleichsstrukturen („größer als“, „kleiner als“) erfordern Abstraktion von quantitativen Objekteigenschaften die in Pre-Training nicht oder nicht außreichend gelernt werden, gerade wenn der Datensatz noch einmal kleiner ist als bei den L Modellen.
    - **Fehlende Weltwissen (wie auch bei Bert WMM)**: Ein Modell müsste Konzepte wie „Flugzeug“ und „Haus“ hinsichtlich ihrer typischen

Größenordnungen kennen und vergleichen – dafür ist das Masking-Objective nicht geeignet.

- **Hinweise im Prompt zu schwach**: Anders als bei numerischen Werten gibt es hier keine Zahl im Prompt, nur Konzepte. Die Modelle können daher nicht auf eine verkettete Referenz zurückgreifen.

- Diese Punkte erklären die nahe der “Rate-performance” liegenden Ergebnisse für roberta-base und Bert-base

- **2.4 Antonym_Negation**

  - **Ergebnis**: BERT 51.40 %, RoBERTa 60.80 % (Zufall 50 % — 2 Antwortmöglichkeiten)

  - **Large-Model-Benchmark**: RoBERTa-Large 75 %, BERT-WWM 57 % Zero-Shot. Die bessere Performance von Roberta-large ist hier mit großer Sicherheit auch wieder auf das größere Trainingsdatenset zurückzuführen

  - Ursachen:

    - **Roberta-base schlägt Bert**: RoBERTa mit 10× Trainingsdaten erlernt Kontexte stärker, wodurch „not“ vs. Intensifikator („very“) unterscheidbar wird.

    - **Antonyme im Trainingsdatenset**: Antonym-Paare sowie Negationskonstruktionen wie “not good” “never seen” kommen im Trainingskorpus vermutlich häufig vor, was eine schärfere Trennung beim lernen des Kontext ermöglicht. Dies macht sich bereits bei Roberta-base mit ca 60% bemerkbar.

    - Bert-base liegt wieder ungefähr bei “Rate-Performance”

- **2.5 Taxonomy_Conjunction**

  - **Ergebnis**: BERT 43.74 %, RoBERTa 46.74 % (Zufall 33 % — 3 Antwortmöglichkeiten)

  - **Large-Model-Benchmark**: RoBERTa-Large ca. 45 % Zero-Shot, und Bert Large-WWM bei 46 %
    Teilweises Wissen über Hypernyme: Modelle kennen einzelne Hypernym-Relationen (z. B. „crow” -> „Bird“ statt “crow” -> “Tier"), aber nicht deren Schnittmengenoperation. (Paper 4.4) Außerdem gibt es einen Taxonomie-Bias: In Fehleranalysen wählt RoBERTa-L oft das hypernym näher am ersten Begriff; das zweite Konzept wird ignoriert: Also ein bestimmter Vogel zb Krähe liegt näher am Begriff “Vogel” als an “Tier”.

- Ursachen: Beide Basismodelle erzielen ähnliche Performance und schlagen die 33 % deutlich. Auch sie liegen ungefähr im Bereich der beiden großen Modelle bei der Zero-Shot evaluation. Dies könnte darauf hindeuten, dass bei dieser Aurt von Aufgaben bereits eine moderate Datenbasis genügt, und eine größere Datenmenge nur noch marginal neue Informationen inbseonder bezogen auf die co-occurance-Statistiken bereitstellt.

- **2.6 Multihop_Composition**

  - **Ergebnis**: BERT 34.00 %, RoBERTa 34.20 % (Zufall 33 % — 3 Auswahlmöglichkeiten)

  - **Large-Model-Benchmark**: Zero-Shot um 29–33 % (random)
    Dieses “Zufalls-Ergebniss” zeigt, dass kein echtes multi-hop reasoning statt findet. Es gibt also keinen Mechanismus, der mehrere aufeinander folgende Fakten miteinander verknüpfen kann.

  - Ursachen:

    - Auch die beiden Basis Modelle liegen nahe Zufall und leiden an den gleichen Problemen wie die großen Modelle.
      Eine größere Datenmenge scheint auch hier keine Lösung zu sein.

- **Zusammenfassung**:

  - **Modellgröße & Datenmenge**
    Große Modelle (RoBERTa-L, BERT-WWM) profitieren von mehr Trainingsdaten und Kapazität und lösen drei der sechs Aufgaben teilweise oder vollständig, Base-Varianten jedoch kaum.

  - **Context Abhängigkeit**
    Selbst bei erfolgreichen Tasks (z. B. Age_Comparison) zeigt sich starke Bindung an Wertebereiche und Templates, kein echtes abstraktes Reasoning .

  - **LM-Objective != Symbolisches Denken**
    Die MLM-Loss optimiert für Wort-Co-Occurrence, nicht für logische Operatoren, Mengen- oder Zahlenoperationen. für symbolisches Reasoning sind gezielte Probes und/oder Fine-Tuning unabdingbar.

  - **Folgerung:** Um im Sinne des Papers robuste Resultate zu erreichen, solltest man

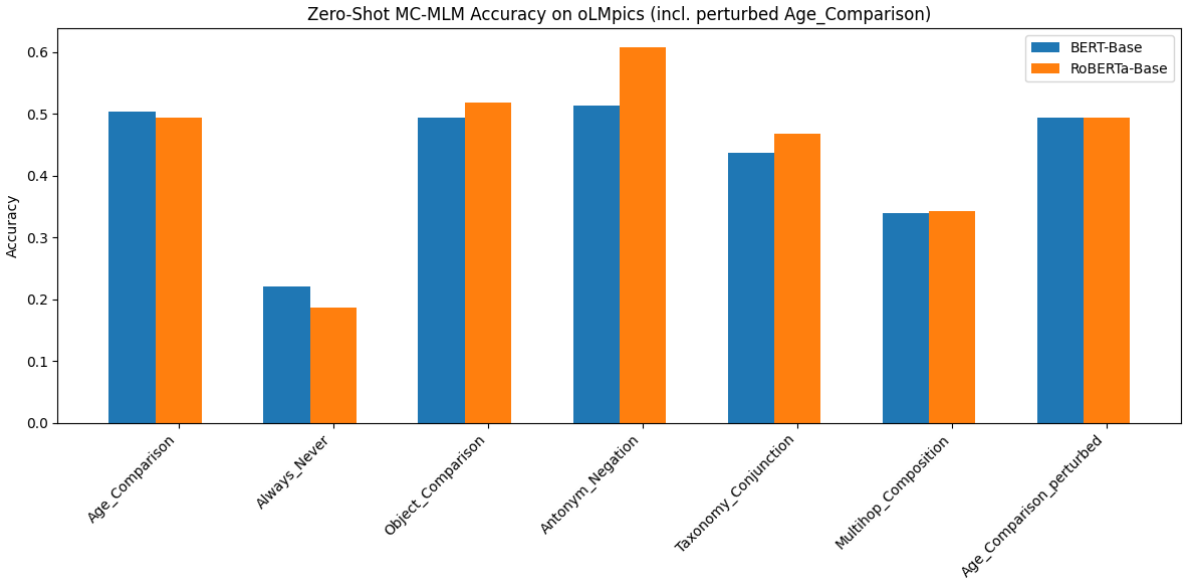
    - **größere Modellvarianten** (roberta-large, bert-wwm) nutzen,

    - **Fine-Tuning** auf kontrollierten Probedaten durchführen (MLP-MLM, LINEAR),

Aufgabe 1.2:

- **Experimentaufbau**: Enspricht dem Aufbau aus 1.1 ergänzt, allerdings werden in dem Age_Comparison Set folgende Wörter ersetzt:
  - age -> blah
  - than -> da

  Dieses sog. Perturbationsexperiment soll herausfinden, ob das Modell abstrakte Konzepte wie hier älter/jünger lernt oder ob es andere oberflächliche bzw. “nicht-beabsichtigte” Patterns lernt. Durch das verwenden von Nonsense-Wörtern (in diesem Zusammenhang) fallen diese Stichwort-Trigger weg



Auswertung Accuracy Aufgabe 1.2

Variante	Zufalls-Baseline (Paper)	BERT-Base (1.1)	RoBERTa-Base (1.1)	BERT-Base (1.2)	RoBERTa-Base (1.2)	RoBERTa-Large (Paper)	BERT-WWM (Paper)	RoBERTa-Large (perturbed) (Paper}	BERT-WWM (perturbed) (Paper)
Age_Comparison	50 %	50.40 %	49.40 %	49.40 %	49.40 %	98 %	70 %	67 %	57 %

- **1.1 Original (BERT 50.4 %, Roberta 49.4 %)**:

  - **Subword-Tokenisierung von Zahlen**
    BERT-Base und RoBERTa-Base zerlegen Zahlen oft in mehrere Sub-Wörter (zB “42” -> “4” + “##2”). Sie lernen folglich u.u. unscharfe, verallgemeinerte Repräsentationen für numerische Werte, insbesondere jenseits häufiger Altersangaben im Pre-Training (15–80). (Siehe auch Paper)

  - **Fehlende arithmetische Fähigkeiten**
    Das MLM-Objective zielt darauf ab, fehlende Wörter vorherzusagen – nicht, einfache Rechenoperationen anzustellen. Eine echte „größer-als“ oder „kleiner-als“ Relation wird nie direkt trainiert und taucht nur in den Co-Occurrences (zB „older than“, „younger than“) auf.

  - **Zufalls-Baseline**
    Da es in diesem Task zwei Antwortmöglichkeitne gibt, liegt ein reiner Zufallsklassifikator bereits bei 50 %.

- **1.2 Perturbed Experiment (“age”->“blah”, “than”->“da”; BERT 49.4 %, Roberta 49.4 %)**:

  - **Geringe Abhängigkeit von Keywords (Trigger)**
    Beide Base-Modelle haben im Original schon kaum echte Alters-Vergleichs-fähigkeiten. Sie “raten” zufällig. Ein Austausch von „age und „than“ verändert daher kaum das Ergebnis

    - Im Gegensatz zu RoBERTa-Large, das im Paper seine 98 % zu weiten Teilen auf starkes Erkennen der Phrasen „older than“ stützt (und beim Perturbieren um 31 % abfällt), nutzen die Base-Varianten kein solches Pattern: Sie haben einfach kein schlicht kein robustes Altersschema gelernt.

- **Fazit zur Perturbation im Paper vs. eigener Auswertung**:

  - **RoBERTa-Large**: stürzt von 98 % auf 67 % ab (-31 %), weil es stark auf die Keywords „age/than“ getrimmt ist.

  - **BERT-WWM**: sinkt von 70 % auf 57 % (-13 %).

  - **Base-Modelle**: bleiben praktisch unverändert (-1 % bzw. 0 %).

Aufgabe 2

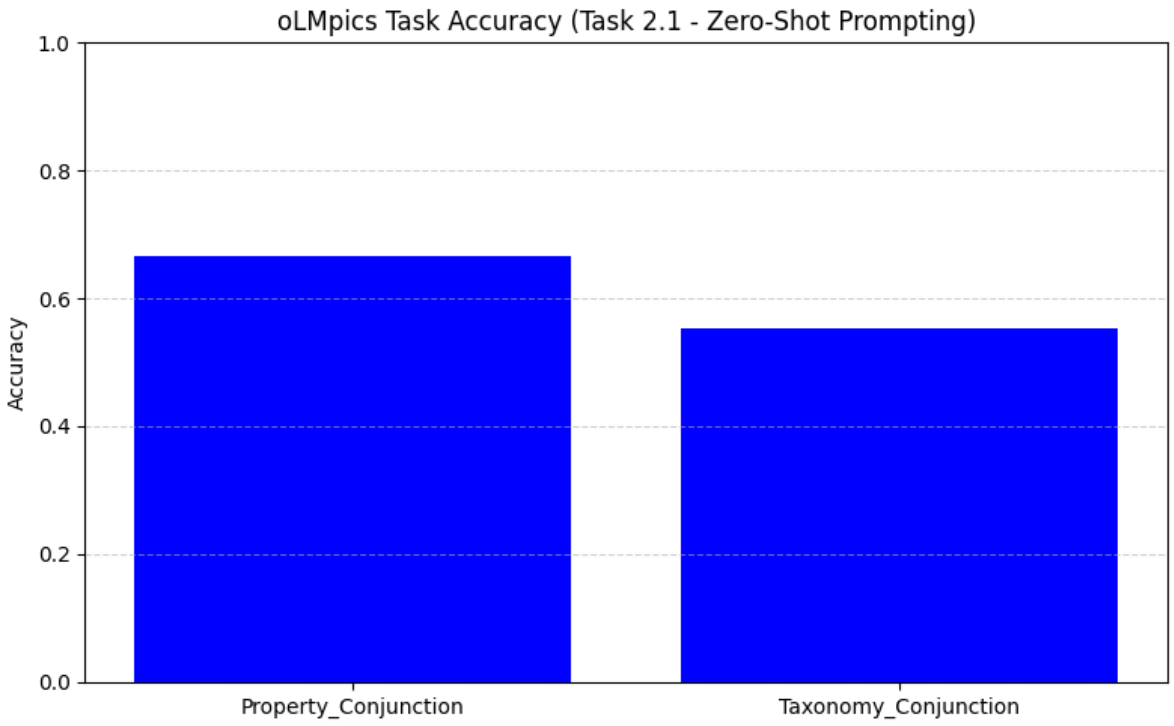
Folgende Version von Llama wurde für das Experiment 2.1 Lokal verwendet: llama3.1:8b: https://ollama.com/library/llama3.1:8b

Folgende Anleitung wurde für das Setup verwendet: https://www.llama.com/docs/llama-everywhere/running-meta-llama-on-mac/

Allerdings bietet das Skript “exercise_2_1.py” die Möglichkeit auch auf “http://llm.lehre.texttechnologylab.org/” zuzugreifen mithilfe des Remote-flags: “python exercise_2_1.py —remote”

Experiment 2.2 läuft nur remote.

Aufgabe 2.1:



Auswertung 2.1

Hier wurde evaluiert, wie gut LLaMA 3.1:8B im Zero-Shot MC-Fragen aus dem oLMpics-Benchmark beantworten kann. Getestet wurden die Sets: Property

Conjunction und Taxonomy Conjunction. Ziel war es, das Modell ohne Fine-Tuning nur mit Prompting zur Auswahl der richtigen Antwort zu bringen.

Der wichtigste Teil des Codes ist dabei das Prompting und die Antwort-Extraktion. Für jede Frage wird ein Prompt erzeugt, der aus dem Fragetext, Platzhalter, den Antwortoptionen A, B, C) und einer klaren Anweisung bestand. Der Benutzerprompt lautet zum Beispiel:

"*What is located at hand and used for writing?*

A) pen
B) spoon
C) computer
Answer:

"

Zusätzliche Instruktion: Ausschließlich den Buchstaben der korrekten Option zurückzugeben, ohne zusätzliche Erklärungen:

"*You are a helpful assistant for multiple-choice questions from the oLMpics benchmark. Answer each question by choosing the correct option and reply only with the corresponding letter (A, B, C, ...). Do not explain your answer.*"

Die Antwort des Modes extrahiert mit einem Regex genau einen Großbuchstabe im Bereich A–C extrahiert.

Ergebnisse:

- Accuracy von 66,67 % für Property Conjunction
- 55,43 % für Taxonomy Conjunction
- Beide Werte liegen deutlich über dem Zufall (33 %), insbesondere bei Property Conjunction zeigt das Modell, dass es in der Lage ist, konjunktive Eigenschaften korrekt zu verknüpfen. Die Leistung bei Taxonomy Conjunction ist geringer, aber immer noch deutlich über Zufall.
- Vergleicht man diese Resultate mit den Ergebnissen aus dem Paper, zeigt sich, dass das LLaMA 3.1:8B-Modell im Zero-Shot schon eine gute Leistung erreicht. Im Paper wurden für Property Conjunction mit fine-tuned Modellen Werte bis zu 87 % Accuracy erzielt, wobei ohne Fine-Tuning die Werte teils deutlich niedriger lagen. Unsere Zero-Shot-Ergebnisse mit LLaMA3 liegen zwar unterhalb der fine-tuned Maxima, aber in ähnlicher Größenordnung wie die besten Zero-Shot-

Ergebnisse im Paper. Für Taxonomy Conjunction erreichte RoBERTa im Paper maximal 59 %, also nur leicht über dem hier erzielten Wert von 55,43 %.

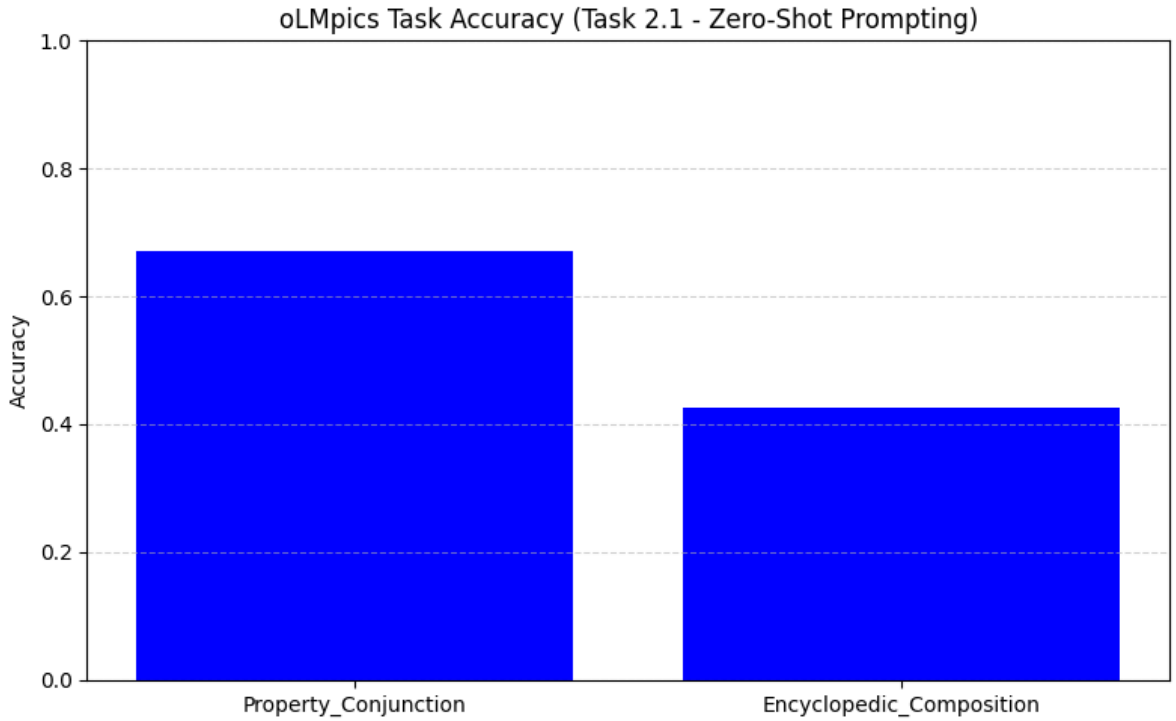
- Bezogen auf die Frage, ob die Modellantworten immer im erwarteten Format geliefert wurde:
  - Nein. Zwar war die Intention im Prompt eindeutig (nur den Buchstaben zurückgeben), dennoch kam es in der Praxis regelmäßig zu Varianten wie „B“, „B)“, „B) examine thing“ oder ähnlichem.
    - Dies zeigt, dass generative Modelle den System-Prompt nicht strikt einhalten, sondern dazu neigen, aus dem Kontext vollständige Antworten zu generieren.
    - Die Ursache liegt vermutlich darin, dass das Modell nicht wie ein klassisches Klassifikationsmodell auf formale Ausgabeformate optimiert wurde, sondern frei formulieren darf.
    - Dies macht es notwendig mithilfe der REGEX die Antworten zu extrahieren
    - Vollständig deterministische Ausgabeformate sind folglich bei generativen Modellen nur mithilfe des Prompts schwer zu erzwingen
  - Diese Ergebnisse sind dem Ordner results 2_1 zu entnehmen, wo noch einmal der Unterschied zwischen dem Originall-Content des Modells und dem bereinigten zu sehen ist

Insgesamt zeigt sich, dass das gewählte Prompting zusammen mit der gezielten Antwort-Extraktion gut funktioniert. Das Modell liefert in vielen Fällen die korrekte Option, ohne dass eine zusätzliche Steuerung oder Nachbearbeitung nötig war. Der Erfolg hängt dabei stark davon ab, dass das Modell genau angewiesen wird, wie es antworten soll, und dass die Antwort anschließend korrekt verarbeitet wird.

Aufgabe 2.1: nach geänderter Aufgabenstellung:

Hier wurde evaluiert, wie gut LLaMA 3.1:8B im Zero-Shot MC-Fragen aus dem oLMpics-Benchmark beantworten kann. Getestet wurden die Sets: Property Conjunction und Encyclopedic Composition. Ziel war es, das Modell ohne Fine-Tuning nur mit Prompting zur Auswahl der richtigen Antwort zu bringen.

Der wichtigste Teil des Codes ist dabei das Prompting und die Antwort-Extraktion. Für jede Frage wird ein Prompt erzeugt, der aus dem Fragetext, Platzhalter, den



Antwortoptionen A, B, C) und einer klaren Anweisung bestand. Der Benutzerprompt lautet zum Beispiel:

"*What is located at hand and used for writing?*
A) pen
B) spoon
C) computer

Answer:"

Zusätzliche Instruktion: Ausschließlich den Buchstaben der korrekten Option zurückzugeben, ohne zusätzliche Erklärungen:

"*You are a helpful assistant for multiple-choice questions from the oLMpics benchmark. Answer each question by choosing the correct option and reply only with the corresponding letter (A, B, C, ...). Do not explain your answer.*"

Die Antwort des Modells wird mit einem Regex ausgewertet, der genau einen Großbuchstaben im Bereich A–C extrahiert.

Ergebnisse:

- Accuracy von 66,67 % für Property Conjunction
  43,02 % für Encyclopedic Composition
  - Beide Werte liegen deutlich über dem Zufall (33 %), insbesondere bei Property Conjunction zeigt das Modell, dass es in der Lage ist, konjunktive

Eigenschaften korrekt zu verknüpfen. Die Leistung bei Encyclopedic Composition ist geringer, aber immer noch deutlich über Zufall.

  - Vergleicht man diese Resultate mit den Ergebnissen aus dem Paper, zeigt sich, dass das LLaMA 3.1:8B-Modell im Zero-Shot schon eine gute Leistung erreicht. Im Paper wurden für Property Conjunction mit fine-tuned Modellen Werte bis zu 87 % Accuracy erzielt, wobei ohne Fine-Tuning die Werte teils deutlich niedriger lagen. Unsere Zero-Shot-Ergebnisse mit LLaMA 3 liegen zwar unterhalb der fine-tuned Maxima, aber in ähnlicher Größenordnung wie die besten Zero-Shot-Ergebnisse im Paper. Für Encyclopedic Composition erreichte RoBERTa im Paper maximal 50 %, also nur leicht über dem hier erzielten Wert von 43,02 %.

- Bezogen auf die Frage, ob die Modellantworten immer im erwarteten Format geliefert wurden:

  - Nein. Zwar war die Intention im Prompt eindeutig (nur den Buchstaben zurückgeben), dennoch kam es in der Praxis regelmäßig zu Varianten wie „B“, „B)“, „B) examine thing“ oder ähnlichem.

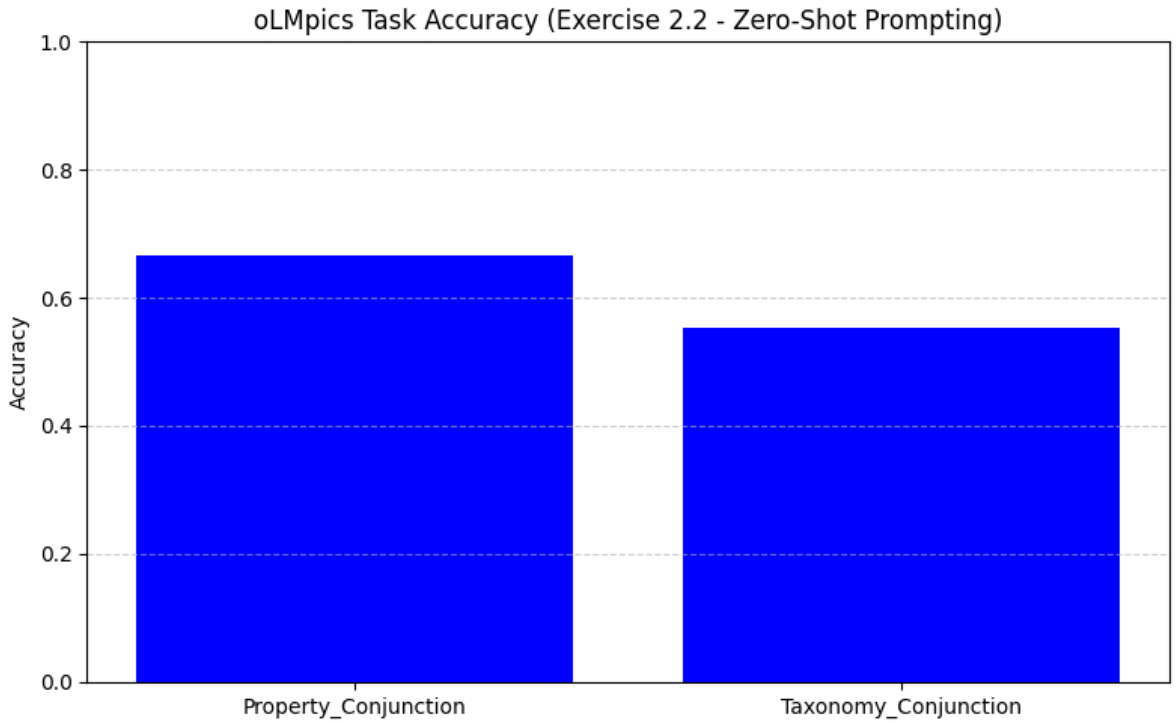
  - Dies zeigt, dass generative Modelle den System-Prompt nicht strikt einhalten, sondern dazu neigen, aus dem Kontext vollständige Antworten zu generieren. Die Ursache liegt vermutlich darin, dass das Modell nicht wie ein klassisches Klassifikationsmodell auf formale Ausgabeformate optimiert wurde, sondern frei formulieren darf.

  - Dies macht es notwendig, mithilfe der REGEX die Antworten zu extrahieren. Vollständig deterministische Ausgabeformate sind folglich bei generativen Modellen nur mithilfe des Prompts schwer zu erzwingen.

  - Diese Ergebnisse sind dem Ordner results_2_1_new zu entnehmen, wo noch einmal der Unterschied zwischen dem Original-Content des Modells und dem bereinigten zu sehen ist.

- Insgesamt zeigt sich, dass das gewählte Prompting zusammen mit der gezielten Antwort-Extraktion gut funktioniert. Das Modell liefert in vielen Fällen die korrekte Option, ohne dass eine zusätzliche Steuerung oder Nachbearbeitung nötig war. Der Erfolg hängt dabei stark davon ab, dass das Modell genau angewiesen wird, wie es antworten soll, und dass die Antwort anschließend korrekt verarbeitet wird.

Aufgabe 2.2:



In Aufgabe 2.2 wurde der Code aus Aufgabe 2.1 mit dem Ziel abgeändert die Ausgabe des Modells strikt auf die erlaubten Antwortoptionen A, B oder C zu beschränken. Anders als in Aufgabe 2.1 wurde hier nicht nur über Prompting gearbeitet, sondern auch auf API-Ebene eine technische Einschränkung der Ausgabe vorgenommen.

- Der entscheidende Unterschied liegt im Einsatz der Parameter
  - max_tokens=1,
  - stop=["\n"] und
  - logit_bias. Dieser erzwingt über gezielte Bevorzugung der Token-IDs von A-C dass das Modell nur eine dieser Optionen auswählt.

Der Prompt selbst ist dabei identisch mit dem aus Aufgabe 2.1.

Ergebnisse:

- Die Ergebnisse waren exakt dieselben wie in Aufgabe 2.1:
  - Property Conjunction Accuracy: 66.67 %
  - Taxonomy Conjunction Accuracy: 55.43 %
- Die Ergebnisse aus 2.2 zeigen, dass das Modell inhaltlich die gleichen Entscheidungen trifft, aber stabilerantwortet.
  - In 2.1 war die Antwortverarbeitung fehleranfällig, da das Modell oft zusätzliche Inhalte wie „B) examine thing“ oder „Answer: B“ zurückgab.
  - In Aufgabe 2.2 hingegen bestand jede Antwort aus exakt einem der zulässigen Buchstaben – wie beabsichtigt.
  - Bzgl. des Konsistenteren Antwortschemas in 2.2 siehe auch im Order results_2_2 die beiden CSVs mit den Aswertungen. Hier ist deutlich zu sehen, dass immer genau ein Buchstabe A,B oder C als Antwort gewählt wurde. Die Einschränkungen auf API Ebene sind also deutlich zu sehen.
- Erhalten Sie immer die Ergebnisse, die Sie erwartet hätten, in dem Format, in dem Sie sie erwartet hätten? Wenn nicht, woran könnte das liegen?
  - In Aufgabe 2.2 wurde das Problem aus 2.1 vollständig gelöst. Durch den Einsatz der parameter logit_bias, max_tokens=1 und einem passenden stop-Token war es unmöglich, dass das Modell mehr als den gewünschten Buchstaben zurückgibt. Dieses Setup ist folglich ideal für echte Multiple-Choice-Aufgaben mit festen Antwortformaten.

Insgesamt is es in Aufgaben wie MC wichtig, das Ausgabeformat des Modells gezielt steuern zu können. 2.2 zeigt, dass dies durch technische Einschränkungen auch bei generativen Modellen zuverlässig funktioniert (auch ohne Veränderung des Prompts oder zusätzliche Nachbearbeitung. Die Accuracy hat sich zwar nicht verbessert aber eben die Konsistenz der Antworten. Eine Höhere Diskrepanz zwischen reinem Prompting und API-Parameter gesetztem Prompten könnte möglicherweise erreicht werden, wenn der Prompt grundsätzlich weniger spezifisch ist oder die Antwortmöglichkeiten komplexer sind.