



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΣΥΣΤΗΜΑΤΩΝ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ ΚΑΙ ΜΑΘΗΣΗΣ

Exploring Transformer-Based Reasoning through Efficient Tuning and Advanced Prompting Strategies

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

Ιωάννη Παναγιωτόπουλου

Επιβλέπων: Γεώργιος Στάμου
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2024



Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Συστημάτων Τεχνητής Νοημοσύνης και Μάθησης

Exploring Transformer-Based Reasoning through Efficient Tuning and Advanced Prompting Strategies

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

Ιωάννη Παναγιωτόπουλου

Επιβλέπων: Γεώργιος Στάμου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 24^η Οκτωβρίου, 2024.

.....
Γεώργιος Στάμου
Καθηγητής Ε.Μ.Π.

.....
Αθανάσιος Βουλόδημος
Επίκουρος Καθηγητής Ε.Μ.Π.

.....
Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2024

.....
ΙΩΑΝΝΗΣ ΠΑΝΑΓΙΩΤΟΠΟΥΛΟΣ
Διπλωματούχος Ηλεκτρολόγος Μηχανικός
και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © – All rights reserved Ιωάννης Παναγιωτόπουλος, 2024.
Με επιφύλαξη παντός δικαιώματος.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Η παρούσα διπλωματική διατριβή διερευνά μεθόδους για τη βελτίωση των ικανοτήτων συλλογισμού των μεγάλων γλωσσικών μοντέλων (MGΜ) μέσω της αξιοποίησης μη σειριακών συλλογιστικών πορειών, που απαιτούν την αμφισβήτηση της παραδοσιακής, συνεπαγωγικής λογικής. Η ανάλυση εστιάζει σε δύο διαφορετικές προσεγγίσεις για την ενίσχυση της απόδοσης των MGΜ: η πρώτη μέσω εκπαίδευσης και η δεύτερη μέσω προηγμένων τεχνικών προτροπής, χωρίς την ανάγκη πρόσθετης εκπαίδευσης.

Η πρώτη προσέγγιση [59] σχετίζεται με τον διαγωνισμό SemEval-2024 Task 9: «BRAINTEASER: A Novel Task Defying Common Sense» [35], και εστιάζει στην επιπλέον εκπαίδευση μοντέλων βασισμένων σε μετασχηματιστές, χρησιμοποιώντας το σύνολο δεδομένων BRAINTEASER [37]. Αυτή η προσέγγιση αφορά την εκπαίδευση των μοντέλων στην επίλυση προκλήσεων που απαιτούν μη σειριακή συλλογιστική, όπως γρίφοι λέξεων και προτάσεων, όπου η λύση δεν βασίζεται στην κοινή λογική συνεπαγωγή. Μέσω στοχευμένης εκπαίδευσης σε μικρότερα μοντέλα κωδικοποιητών και MGΜ, επιδιώκεται η υπέρβαση των τυπικών επιδόσεων. Ένα βασικό στοιχείο αυτής της προσέγγισης είναι η μετατροπή των προβλημάτων πολλαπλών επιλογών σε δυαδικά προβλήματα ταξινόμησης, προβλήματα Σωστού-Λάθους, επιτρέποντας στα μοντέλα να διερευνήσουν πολλαπλά μονοπάτια συλλογισμού. Η ανάλυση εστιάζει στην επιρροή του μεγέθους του μοντέλου και των υπερπαραμέτρων, ενώ εξετάζονται οι λόγοι αποτυχίας των μοντέλων. Στόχος είναι η βελτίωση της ακρίβειας και των ικανοτήτων τους σε μη τυπικές συλλογιστικές πορείες μέσω στοχευμένης εκπαίδευσης.

Η δεύτερη προσέγγιση [60] αποφεύγει την εκπαίδευση και επικεντρώνεται στη βελτίωση της απόδοσης μέσω μιας καινοτόμου τεχνικής προτροπής, που την ονομάζουμε RISCORE (RIddle Solving with COntext REcontruciton). Αυτή η μέθοδος, εμπνευσμένη από το σύνολο δεδομένων BRAINTEASER, βελτιώνει την απόδοση με χρήση λίγων παραδειγμάτων ως προτροπές, παρέχοντας αναδομημένα παραδείγματα γρίφων που αλλάζουν το περιεχόμενο, με σκοπό να ενισχύσουν τις ικανότητες του μοντέλου στην επίλυση σύνθετων προβλημάτων. Το RISCORE διατηρεί την αρχική λογική του προβλήματος αλλά στους αναδιαμορφωμένους γρίφους προσφέρει το ίδιο λογικό πρόβλημα αναδιατυπωμένο σε νέο πλαίσιο - περιεχόμενο. Σε σύγκριση με άλλες μεθόδους προτροπής, το RISCORE αποδείχθηκε αποτελεσματικό στη βελτίωση των μη σειριακών συλλογιστικών πορειών, αλλά και των πιο παραδοσιακών προβλημάτων λογικής, χωρίς την ανάγκη πρόσθετης εκπαίδευσης. Αυτή η προσέγγιση αναδεικνύει τις δυνατότητες των στρατηγικών προτροπής στη βελτίωση της απόδοσης των MGΜ σε σύνθετες εργασίες που αμφισβητούν τη συμβατική κοινή λογική.

Συνολικά, οι δύο προσεγγίσεις παρουσιάζουν διαφορετικές μεθοδολογίες – η πρώτη επικεντρώνεται στην εκπαίδευση και τη βελτιστοποίηση των μοντέλων, ενώ η δεύτερη στις καινοτόμες τεχνικές προτροπής – και συμβάλλουν σημαντικά στην κατανόηση του τρόπου με τον οποίο τα MGΜ μπορούν να βελτιωθούν στην αντιμετώπιση προκλήσεων που απαιτούν μη σειριακές συλλογιστικές πορείες, αμφισβητώντας τις παραδοσιακές λογικές συνεπαγωγές.

Λέξεις-κλειδιά — Μεγάλα Γλωσσικά Μοντέλα (MGΜ), στοχευμένη εκπαίδευση (fine-tuning), Προσαρμογή Χαμηλής Τάξης (LoRA), Κβαντισμένη Προσαρμογή Χαμηλής Τάξης (QLoRA), αναπαραστάσεις λέξεων (word embeddings), μάθηση λίγων παραδειγμάτων (few-shot learning), μάθηση χωρίς παραδείγματα (zero-shot learning), ανακατασκευή πλαισίου - περιεχομένου, σημασιολογική ανακατασκευή, σημασιολογική ομοιότητα, μοντέλα μετασχηματιστών (transformer models), μέθοδος RISCORE, μάθηση εντός πλαισίου (in-context learning), ικανότητες συλλογισμού.

Abstract

This thesis investigates methods to improve the reasoning capabilities of large language models (LLMs) by leveraging lateral thinking, focusing on two distinct proposals that address different approaches to enhancing model performance: the first through fine-tuning and training, and the second through advanced prompting techniques without additional training.

The first proposal [59] is tied to the SemEval-2024 Task 9 competition: "BRAINTEASER: A Novel Task Defying Common Sense," [35] where the focus is on fine-tuning transformer-based models using the BRAINTEASER dataset [37]. This approach involves training models to solve lateral thinking challenges, such as sentence and word puzzles. By employing lightweight tuning on smaller encoder models and LLMs, the aim was to surpass baseline performance. A key element of this proposal was transforming multiple-choice problems into binary classification tasks, allowing the models to explore diverse reasoning paths. The analysis highlighted the influence of model size and hyperparameters, along with an investigation into the reasoning cues that lead to model failures. The goal was to enhance model accuracy and reasoning skills, while providing insights into how LLMs handle lateral thinking problems through targeted fine-tuning.

In contrast, the second proposal [60] takes a different approach by avoiding model training altogether and instead focusing on enhancing performance through a novel prompting technique called RISCORE. This method, inspired by the structure of the BRAINTEASER dataset, augments few-shot learning by providing contextually reconstructed examples of riddles, designed to improve the model's in-context problem-solving abilities. RISCORE operates by preserving the original reasoning process while altering the context to offer a clearer reasoning trajectory for the model to follow. By comparing RISCORE to other popular prompting methods, the results showed its effectiveness in improving both lateral and vertical thinking tasks without the need for additional training. This approach highlights the potential of strategic prompting in enhancing LLM performance, particularly in complex reasoning tasks that challenge common sense.

These two proposals showcase distinct methodologies—one focused on model training and fine-tuning, and the other on innovative prompting techniques—both contributing valuable insights into how LLMs can be improved for lateral thinking challenges.

Keywords — Lateral thinking, vertical thinking, Large Language Models (LLMs), fine-tuning, Low-Rank Adaptation (LoRA), Quantized Low-Rank Adaptation (QLoRA), word embeddings, few-shot learning, zero-shot learning, context reconstruction, semantic reconstruction, semantic similarity, transformer models, RISCORE method, in-context learning, reasoning capabilities.

Ευχαριστίες

Η παρούσα διπλωματική εργασία δεν θα είχε ολοκληρωθεί χωρίς την πολύτιμη υποστήριξη και καθοδήγηση πολλών ανθρώπων, στους οποίους θα ήθελα να εκφράσω την ειλικρινή μου ευγνωμοσύνη.

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντά μου, καθηγητή κ. Στάμου Γεώργιο, για την ευκαιρία που μου προσέφερε να εκπονήσω την διπλωματική μου εργασία στο Εργαστήριο Συστημάτων Τεχνητής Νοημοσύνης και Μάθησης. Η καθοδήγησή του δεν περιορίστηκε στα ακαδημαϊκά πλαίσια, αλλά συνέβαλε καταλυτικά στην προσωπική και επαγγελματική μου πορεία, ανοίγοντας νέους ορίζοντες πέρα από το πλαίσιο της σχολής.

Ευχαριστώ επίσης τους υποψήφιους διδάκτορες, Γιώργο Φιλανδριανό και Μαρία Λυμπεραίου, για τη συνεχή συνεργασία και την αμέριστη υποστήριξή τους. Με την υπομονή και την καθοδήγησή τους, με βοήθησαν να ενταχθώ σε μια δημιουργική και παραγωγική ομάδα, όπου απέκτησα πολύτιμες εμπειρίες και γνώσεις. Η τεχνογνωσία τους, σε συνδυασμό με την διορατικότητά τους, αποδείχθηκαν καθοριστικές για την επιτυχή ολοκλήρωση της διπλωματικής μου εργασίας.

Οι φίλοι μου, και πλέον συνάδελφοι, υπήρξαν πολύτιμοι συνοδοιπόροι σε αυτό το ταξίδι. Μαζί αντιμετωπίσαμε κάθε πρόκληση και δυσκολία των τελευταίων χρόνων, και μέσα από τη συνεργασία μας καταφέραμε να ξεπεράσουμε εμπόδια που φάνταζαν ανυπέρβλητα. Η υποστήριξή τους και η κοινή μας πορεία με έκαναν να αισθάνομαι ότι δεν ήμουν ποτέ μόνος σε αυτή τη διαδρομή, αλλά μέρος μιας δυνατής ομάδας που μοιραζόταν τις ίδιες αξίες και φιλοδοξίες.

Η στήριξη της οικογένειάς μου ήταν αναμφίβολα θεμελιώδης. Οι γονείς μου και ο αδερφός μου, με την απεριόριστη αγάπη, την υπομονή και την ακλόνητη πίστη στις δυνατότητές μου, ήταν μια σταθερή πηγή ενθάρρυνσης και δύναμης σε κάθε βήμα αυτής της πορείας.

Ιωάννης Παναγιωτόπουλος, Οκτώβριος 2024

Contents

Contents	11
List of Figures	14
List of Tables	15
1 Εκτεταμένη Περίληψη στα Ελληνικά	21
1.1 Θεωρητικό Υπόβαθρο	22
1.1.1 Γλωσσικά Μοντέλα	22
1.1.2 Τεχνικές Μάθησης	22
1.1.3 Pre-Training and Fine-Tuning	23
1.1.4 Low-Rank Adaptation and Quantized Low-Rank Adaptation	23
1.1.5 Έννοια της Προτροπής	24
1.1.6 Τεχνικές Προτροπής	24
1.1.7 Γραμμικός τρόπος σκέψης και Αντισυμβατικός - Δημιουργικός τρόπος σκέψης	26
1.2 Προτεινόμενη Μέθοδος 1 - SemEval-2024 Task 9	27
1.2.1 Συνεισφορές	27
1.2.2 Μέθοδοι	27
1.3 Πειραματικό Μέρος - SemEval-2024 Task 9	28
1.3.1 Σύνολα Δεδομένων και Μετρικές	29
1.3.2 Μοντέλα ως Αναφορά	30
1.3.3 Λεπτομέρειες Προτροπής	31
1.3.4 Περιγραφή Πειραμάτων	32
1.3.5 Αποτελέσματα	34
1.4 Προτεινόμενη Μέθοδος 2 - Μέθοδος Προτροπής RISCORE	38
1.4.1 Συνεισφορές	39
1.5 Πειραματικό Μέρος - Μέθοδος Προτροπής RISCORE	40
1.5.1 Σύνολα Δεδομένων και Μετρικές	40
1.5.2 Μέθοδος	41
1.5.3 Βήμα 1: Δημιουργία ζεύγους ερώτησης-απάντησης	41
1.5.4 Βήμα 2: Δημιουργία των λανθασμένων επιλογών	43
1.5.5 Βήμα 3: Δημιουργία του αινίγματος	46
1.5.6 Περιγραφή Πειραμάτων	46
1.5.7 Αποτελέσματα	48
1.6 Συμπεράσματα και Μελλοντικές επεκτάσεις	56
2 Introduction	59
3 Background	61
3.1 Large Language Models (LLMs)	63
3.1.1 Background	63
3.1.2 Transformer	63
3.1.3 Deeper Exploration of LLM Components	65

3.2	Training Techniques	67
3.2.1	Overview	68
3.2.2	Pre-Training and Fine-Tuning	68
3.3	Parameter Efficient Fine-Tuning (PEFT)	70
3.3.1	Introduction to PEFT (Parameter Efficient Fine-Tuning)	70
3.3.2	Low-Rank Parameters in PEFT	70
3.3.3	LoRA: Low-Rank Adaptation	70
3.3.4	Understanding LoRA through an example	72
3.3.5	Introduction to QLoRA (Quantized Low-Rank Adaptation)	75
3.4	Prompting	76
3.4.1	Prompting Methodology	77
3.4.2	Prompt Categories	78
3.4.3	Prompting Techniques	79
3.5	Analysis of Vertical and Lateral Thinking	81
3.5.1	Vertical Thinking in Reasoning	81
3.5.2	Lateral Thinking in Reasoning	81
3.5.3	Differences and Applications of Vertical and Lateral Thinking	82
3.6	Related Work	82
3.6.1	Reasoning in NLP	82
3.6.2	Reasoning with Language Models	83
3.6.3	LLMs and Prompting for Reasoning Tasks	83
4	Proposal 1 - SemEval-2024 Task 9	85
4.1	Contributions	85
4.2	Methods	85
4.2.1	Encoder models	86
4.2.2	LLMs	87
5	Experiments - SemEval-2024 Task 9	89
5.1	Preliminaries	90
5.1.1	Dataset	90
5.1.2	Evaluation metrics	91
5.1.3	Baselines	91
5.1.4	Encoders	91
5.1.5	LLMs	92
5.1.6	Prompting Details	93
5.2	Experimental Setup	94
5.2.1	Encoders	94
5.2.2	LLMs	94
5.3	QLoRA hyperparameters	95
5.4	Results	96
5.4.1	Experimental results and Analysis	96
5.4.2	Assessment and Insights on Dataset Quality	98
5.4.3	Analysis of LoRA Hyperparameters	100
6	Proposal 2 - RISCORE	101
6.1	Contributions	101
6.2	Method for generating context reconstructions	102
6.2.1	Step 1: Generation of Question-Answer pair	102
6.2.2	Step 2: Generation of the distractors	105
6.2.3	Step 3: Creation of the Riddle	108
7	Experiments - RISCORE	109
7.1	Preliminaries	110
7.1.1	Dataset	110
7.1.2	Evaluation metrics	111

7.1.3	Baselines	111
7.1.4	Methods explored	115
7.1.5	RISCORE	115
7.1.6	Models Utilized in RISCORE pipeline	116
7.1.7	Models for Prompting	116
7.2	Experimental Setup	117
7.3	Results	117
7.3.1	BrainTeaser	118
7.3.2	RiddleSense	119
7.3.3	Quality of context reconstructed riddles	120
7.3.4	Detailed Results	120
8	Conclusions - Future Work	127
9	Bibliography	131

List of Figures

1.1.1 (a) Μηδενικά παραδείγματα, (b) Ένα παράδειγμα, (c) Λίγα παραδείγματα [8]	25
1.1.2 Παραδείγματα με ύπαρξη της συλλογιστικής πορείας [87]	26
1.4.1 Ένα παράδειγμα της προτροπής λίγων παραδειγμάτων που αντιπαράβάλλεται με την ίδια μέθοδο αλλά χρησιμοποιώντας ένα παράδειγμα με ανακατασκευασμένο περιεχόμενο. Το κείμενο με πράσινο χρώμα αντιπροσωπεύει την ανακατασκευή του περιεχομένου του παραδείγματος 1, που παράγεται από την προτεινόμενη μέθοδο. Το παράδειγμα με κόκκινο χρώμα αφαιρέθηκε απλώς για να διατηρηθεί ο ίδιος αριθμός παραδειγμάτων.	39
3.1.1 The Transformer - model architecture. The original Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, shown in the left and right halves of figure respectively[83]	65
3.1.2 Self-Attention Mechanism	66
3.3.1 Matrix Multiplication for LoRA Weight Changes	72
3.3.2 Adding the LoRA Weight Changes	73
3.3.3 LoRA Integration in Transformer Architecture	74
3.4.1 Prompting Typology [51]	78
3.4.2 (a) Zero-shot prompting, (b) One-shot prompting, (c) Few-shot prompting [8]	79
3.4.3 Chain-of-thought [87]	80
6.0.1 An example of few-shot (FS) prompting alongside the same method using a context-reconstructed example. The text in green represents the context reconstruction of Example 1, produced by the proposed method. The example in red was removed simply to maintain the same number of shots.	102

List of Tables

1.1	Απεικόνιση της δομής του συνόλου δεδομένων κάθε επιμέρους εργασίας, όπου παρουσιάζεται η αρχική δήλωση μαζί με τις δύο ανακατασκευές της. Οι σωστές απαντήσεις παρουσιάζονται με έντονη γραφή	29
1.2	Παραδείγματα ερωτήσεων που απεικονίζουν και τα δύο επιμέρους υπο-προβλήματα, όπου οι σωστές απαντήσεις παρουσιάζονται με έντονη γραφή . Τα παραδείγματα στα αριστερά αφορούν το υπο-πρόβλημα <i>A: Παζλ προτάσεων</i> , ενώ εκείνα στα δεξιά αντιστοιχούν στο υπο-πρόβλημα <i>B: Παζλ λέξεων</i>	30
1.3	Στατιστικά στοιχεία δεδομένων.	30
1.4	Απόδοση του μοντέλου για το <i>Υπο-πρόβλημα A: Παζλ προτάσεων</i> . Περισσότερα αποτελέσματα στον πίνακα 1.7.	34
1.5	Απόδοση του μοντέλου για το <i>Υπο-πρόβλημα B: Παζλ λέξεων</i> . Περισσότερα αποτελέσματα στον πίνακα 1.7.	35
1.6	Ένα παράδειγμα εξηγήσεων που παράγονται από το Mistral-7b, ενώ αντιμετωπίζεται το πρόβλημα σε μορφή πολλαπλών επιλογών. Οι εξηγήσεις, ανεξάρτητα από την εγκυρότητά τους, είναι λεπτομερείς και επιδεικνύουν υψηλό επίπεδο κατανόησης και συλλογισμού. Οι σωστές απαντήσεις επισημαίνονται με έντονη γραφή, ενώ οι λανθασμένες προβλέψεις παραπέμπουν στην προβλεπόμενη από το μοντέλο απάντηση.	37
1.7	Η απόδοση των διαφόρων υπερπαραμέτρων LoRA για τις Mistral και Mixtral και στα δύο υπο-προβλήματα.	38
1.8	Στατιστικά στοιχεία συνόλου δεδομένων	41
1.9	Απόδοση του μοντέλου για το <i>Brain Teaser</i> με χρήση τεχνικών ως αναφορά και προτροπών RISCORE _m . Τα καλύτερα συνολικά αποτελέσματα επισημαίνονται με έντονη γραφή . Πιο αναλυτικά αποτελέσματα μπορείτε να βρείτε στους πίνακες 1.13 και 1.14.	49
1.10	Επίδοση του μοντέλου για το <i>Brain Teaser</i> με τη χρήση της μεθόδου προτροπής RISCORE. Για την επιλογή όλων των δειγμάτων χρησιμοποιήθηκε επιλογή με βάση την σημασιολογική ομοιότητα. Τα αποτελέσματα που ξεπερνούν τη μέθοδο FS με σημασιολογικά παρόμοια παραδείγματα, χρησιμοποιώντας τον ίδιο αριθμό δειγμάτων, είναι <u>υπογραμμισμένα</u> . Αναλυτικότερα αποτελέσματα μπορείτε να βρείτε στους πίνακες 1.14 και 1.15.	49
1.11	Απόδοση του μοντέλου για το <i>RiddleSense</i> χρησιμοποιώντας βασικές τεχνικές ως αναφορά. Το καλύτερο αποτελέσματα συνολικά είναι επισημασμένο με έντονη γραφή . Πιο λεπτομερή αποτελέσματα μπορείτε να βρείτε στον πίνακα 1.16.	50
1.12	Απόδοση του μοντέλου για το <i>RiddleSense</i> με χρήση της μεθόδου προτροπής RISCORE. Η επιλογή με βάση την ομοιότητα χρησιμοποιήθηκε για την επιλογή όλων των δειγμάτων. Τα αποτελέσματα που ξεπερνούν τη μέθοδο λίγων παραδειγμάτων με σημασιολογικά παρόμοια παραδείγματα, χρησιμοποιώντας τον ίδιο αριθμό λήψεων, είναι <u>υπογραμμισμένα</u> . Αναλυτικότερα αποτελέσματα μπορείτε να βρείτε στον πίνακα 1.17.	50
1.13	Απόδοση μοντέλου για το <i>BrainTeaser</i> (Μέρος 1). Η παρουσία του (Q) στη στήλη μέθοδος υποδεικνύει ότι τα αποτελέσματα αντιστοιχούν στην χβαντισμένη έκδοση του μοντέλου.	51
1.14	Απόδοση μοντέλου για το <i>BrainTeaser</i> (Μέρος 2). Η παρουσία του (Q) στη στήλη μέθοδος υποδεικνύει ότι τα αποτελέσματα αντιστοιχούν στην χβαντισμένη έκδοση του μοντέλου.	52
1.15	Απόδοση μοντέλου για το <i>BrainTeaser</i> (Μέρος 3). Η παρουσία του (Q) στη στήλη μέθοδος υποδεικνύει ότι τα αποτελέσματα αντιστοιχούν στην χβαντισμένη έκδοση του μοντέλου.	53

1.16	Απόδοση μοντέλου για το <i>RiddleSense</i> (Μέρος 1). Η στήλη Quant υποδεικνύει αν το μοντέλο είναι χβαντισμένο ή όχι.	54
1.17	Απόδοση μοντέλου για το <i>RiddleSense</i> (Μέρος 2). Η στήλη Quant υποδεικνύει αν το μοντέλο είναι χβαντισμένο ή όχι.	55
5.1	Illustration of the structure of each sub-task’s dataset, showcasing the original statement along with its two adversarials. The correct answers are highlighted in bold	90
5.2	Example questions illustrating both sub-tasks, with correct answers highlighted in bold . Examples on the left pertain to <i>sub-task A: Sentence Puzzle</i> , while those on the right correspond to <i>sub-task B: Word Puzzle</i>	91
5.3	Data statistics.	91
5.4	Model Performance for <i>sub-task A: Sentence Puzzle</i> . More results in Table 5.7.	97
5.5	Model performance for <i>sub-task B: Word Puzzle</i> . More results in Table 5.7.	98
5.6	An example of explanations generated by Mistral-7b, while addressing the multi-class classification problem. The explanations, regardless of their validity, are detailed and showcase a high level of comprehension and reasoning. Correct answers are indicated in bold, while mispredictions reference the predicted answer by the model.	99
5.7	The performance of various LoRA hyperparameters for Mistral and Mixtral in both sub-tasks.	100
7.1	Data statistics	111
7.2	Model’s performance for <i>Brain Teaser</i> using baselines and RISCORE _m prompting. The best overall results are highlighted in bold . More detailed results can be found in Tables 7.6 and 7.7.	118
7.3	Model’s performance for <i>Brain Teaser</i> using RISCORE prompting. Similarity-based selection was employed for choosing all the exemplars. Results that surpass the FS method with semantically similar examples, using the same number of shots, are <u>underlined</u> . More detailed results can be found in Tables 7.7 and 7.8.	118
7.4	Model’s Performance for <i>RiddleSense</i> using baseline techniques. The best results overall are in bold . More detailed results can be found in Table 7.9.	119
7.5	Model’s performance for <i>RiddleSense</i> using RISCORE prompting. Similarity-based selection was employed for choosing all the exemplars. Results that surpass the FS method with semantically similar examples, using the same number of shots, are <u>underlined</u> . More detailed results can be found in Table 7.10.	119
7.6	Model performance for <i>BrainTeaser</i> (Part 1). The presence of (Q) in the method column indicates that the results correspond to the quantized version of the model.	121
7.7	Model performance for <i>BrainTeaser</i> (Part 2). The presence of (Q) in the method column indicates that the results correspond to the quantized version of the model.	122
7.8	Model performance for <i>BrainTeaser</i> (Part 3). The presence of (Q) in the method column indicates that the results correspond to the quantized version of the model.	123
7.9	Model Performance for <i>RiddleSense</i> (Part 1). The column Quant indicates whether the model is quantized or not.	124
7.10	Model Performance for <i>RiddleSense</i> (Part 2). The column Quant indicates whether the model is quantized or not.	125

Chapter 1

Εκτεταμένη Περίληψη στα Ελληνικά

1.1 Θεωρητικό Υπόβαθρο

1.1.1 Γλωσσικά Μοντέλα

Τα γλωσσικά μοντέλα (ΓΜ) αποτελούν υπολογιστικά εργαλεία που χρησιμοποιούνται για την πρόβλεψη ή τη δημιουργία ανθρώπινου κειμένου, μέσω της μοντελοποίησης της πιθανότητας ακολουθιών λέξεων. Τα παραδοσιακά μοντέλα, όπως τα n -gram, βασίζονται σε έναν σταθερό αριθμό προηγούμενων λέξεων για την πρόβλεψη της επόμενης, χρησιμοποιώντας μια πιθανοτική προσέγγιση. Ωστόσο, αυτά τα μοντέλα παρουσιάζουν δυσκολίες όταν αντιμετωπίζουν σπάνιες ή αθέατες ακολουθίες λέξεων. Για την αντιμετώπιση αυτών των περιορισμών, αναπτύχθηκαν γλωσσικά μοντέλα, ξεκινώντας με τα νευρωνικά δίκτυα πρόσθιας τροφοδότησης (feedforward neural networks). Αυτά τα δίκτυα χρησιμοποιούν ενσωματώσεις λέξεων (word embeddings) για την αναπαράσταση των λέξεων εισόδου, επιτρέποντας καλύτερη γενίκευση σε δεδομένα που δεν έχουν παρατηρηθεί. Στη σύγχρονη εποχή, τα νευρωνικά γλωσσικά μοντέλα έχουν εξελιχθεί περαιτέρω με την εισαγωγή της αρχιτεκτονικής των Μετασχηματιστών (Transformers), η οποία έχει αναδειχθεί ως η κορυφαία προσέγγιση λόγω της ικανότητάς της να μοντελοποιεί πολύπλοκες γλωσσικές εξαρτήσεις και σχέσεις σε μεγάλες ακολουθίες κειμένου.

Τα Μεγάλα Γλωσσικά Μοντέλα (MGM) είναι εξελιγμένα μοντέλα που βασίζονται στην αρχιτεκτονική του Μετασχηματιστή (Transformer), η οποία χρησιμοποιεί μηχανισμό προσοχής για να μοντελοποιεί εξαρτήσεις χωρίς την ανάγκη επαναληπτικών διαδικασιών. Ο Μετασχηματιστής αποτελείται από δύο κύρια μέρη: τον κωδικοποιητή, που επεξεργάζεται τις ακολουθίες εισόδου και τον αποκωδικοποιητή, που παράγει τις ακολουθίες εξόδου. Ο αρχικός σχεδιασμός του Μετασχηματιστή ήταν ως μοντέλο κωδικοποιητή-αποκωδικοποιητή που προοριζόταν για προβλήματα όπως η μηχανική μετάφραση. Μοντέλα όπως το BART και το T5 αξιοποιούν αυτή την αρχιτεκτονική για τέτοιου είδους εφαρμογές.

Ωστόσο, τα σύγχρονα MGM, όπως το GPT, χρησιμοποιούν μια απλουστευμένη προσέγγιση μόνο με τον αποκωδικοποιητή, που επικεντρώνεται στην αυτοπαλίνδρομη πρόβλεψη του επόμενου βήματος, γεγονός που τα καθιστά κατάλληλα για εργασίες, όπως η παραγωγή κειμένου. Από την άλλη πλευρά, μοντέλα όπως το BERT εφαρμόζουν μια αρχιτεκτονική που περιλαμβάνει μόνο τον κωδικοποιητή.

Αυτές οι αρχιτεκτονικές παραλλαγές – μόνο κωδικοποιητής, κωδικοποιητής-αποκωδικοποιητής και μόνο αποκωδικοποιητής – αντικατοπτρίζουν διαφορετικές προσεγγίσεις που είναι κατάλληλες για συγκεκριμένες εργασίες στην Επεξεργασία Φυσικής Γλώσσας (ΕΦΓ), με κάθε αρχιτεκτονική να υπερέχει σε διαφορετικά σενάρια χρήσης.

1.1.2 Τεχνικές Μάθησης

Οι τεχνικές μηχανικής μάθησης (ML) μπορούν να ταξινομηθούν γενικά με βάση τον τύπο των δεδομένων που χρησιμοποιούνται κατά την εκπαίδευση και το επίπεδο εποπτείας που παρέχεται. Αυτές οι κατηγορίες περιλαμβάνουν την επιβλεπόμενη, τη μη επιβλεπόμενη και την ενισχυτική μάθηση, καθώς και υβριδικές προσεγγίσεις, όπως η ημιεπιβλεπόμενη και η αυτοεπιβλεπόμενη μάθηση, οι οποίες συνδυάζουν στοιχεία τόσο από τις επιβλεπόμενες όσο και από τις μη επιβλεπόμενες μεθόδους.

Στην επιβλεπόμενη μάθηση, τα μοντέλα εκπαιδεύονται με επισημασμένα δεδομένα, όπου οι είσοδοι αντιστοιχίζονται στις σωστές εξόδους, απαιτώντας σημαντικά σύνολα δεδομένων με ετικέτες για προβλήματα όπως η ταξινόμηση.

Αντίθετα, η μη επιβλεπόμενη μάθηση λειτουργεί με μη επισημασμένα δεδομένα και έχει στόχο την ανακάλυψη προτύπων και δομών, συχνά χρησιμοποιείται σε εφαρμογές όπως η ομαδοποίηση και η ανίχνευση ανωμαλιών.

Η ημιεπιβλεπόμενη μάθηση συνδυάζει ένα μικρό σύνολο δεδομένων με ετικέτες με ένα μεγαλύτερο σύνολο μη επισημασμένων δεδομένων, επιτρέποντας καλύτερη γενίκευση σε περιπτώσεις όπου τα επισημασμένα δεδομένα είναι περιορισμένα.

Η αυτοεπιβλεπόμενη μάθηση αξιοποιεί μη επισημασμένα δεδομένα μέσω προκαθορισμένων εργασιών που δημιουργούν εποπτεία, μαθαίνοντας έτσι χρήσιμες αναπαραστάσεις. Αυτή η προσέγγιση είναι ιδιαίτερα διαδεδομένη σε τομείς όπως η επεξεργασία φυσικής γλώσσας (ΕΦΓ) και η όραση υπολογιστών.

Τέλος, η ενισχυτική μάθηση επικεντρώνεται στην εκπαίδευση πρακτόρων για την αλληλεπίδραση με το περιβάλλον τους, με στόχο τη μεγιστοποίηση της μακροπρόθεσμης ανταμοιβής.

1.1.3 Pre-Training and Fine-Tuning

Η έννοια του Pre-training και η έννοια του fine-tuning είναι δύο κρίσιμες τεχνικές στη μηχανική μάθηση που επιτρέπουν στα μοντέλα να αποκτήσουν πρώτα ευρεία γνώση από μεγάλα σύνολα δεδομένων και στη συνέχεια να εξειδικευτούν σε συγκεκριμένα είδη προβλημάτων. Κατά το pre-training, τα μοντέλα εκτίθενται σε τεράστια, αδόμητα σύνολα δεδομένων, συχνά χωρίς σαφείς ετικέτες (labels) εργασιών. Στην Επεξεργασία Φυσικής Γλώσσας (ΕΦΓ), οι εργασίες pre-training, βοηθούν τα μοντέλα να μάθουν γενικά πρότυπα, όπως η σύνταξη και η σημασιολογία, τα οποία μπορούν να μεταφερθούν σε μια ποικιλία μεταγενέστερων προβλημάτων. Αυτή η τεχνική παρέχει στο μοντέλο ισχυρά θεμέλια και τη δυνατότητα γενίκευσης σε διάφορες εφαρμογές, καθιστώντας το ιδιαίτερα αποτελεσματικό ακόμη και όταν τα δεδομένα που αφορούν συγκεκριμένα είδη προβλημάτων είναι περιορισμένα.

Το fine-tuning ακολουθεί τη διαδικασία του pre-training και προσαρμόζει το προ-εκπαιδευμένο μοντέλο σε ένα συγκεκριμένο πρόβλημα, χρησιμοποιώντας ένα μικρότερο, επισημασμένο σύνολο δεδομένων. Μέσω αυτής της διαδικασίας, το μοντέλο βελτιώνει την προϋπάρχουσα γενική γνώση του και εξειδικεύεται στην επίλυση του συγκεκριμένου προβλήματος. Το fine-tuning επιτρέπει στο μοντέλο να ανιχνεύσει συγκεκριμένα μοτίβα και σχέσεις στα δεδομένα του προβλήματος, ενώ εκμεταλλεύεται την ευρεία γνώση που απέκτησε κατά τη διάρκεια του pre-training. Αυτή η μέθοδος είναι ιδιαίτερα ωφέλιμη όταν υπάρχει περιορισμένη διαθεσιμότητα επισημασμένων δεδομένων, καθώς το μοντέλο ξεκινά από μια ισχυρή βάση γνώσης που μειώνει τον κίνδυνο υπερπροσαρμογής, ενισχύοντας την απόδοσή του σε νέα και άγνωστα δεδομένα.

Σε περιπτώσεις όπου τα δεδομένα για μια συγκεκριμένη εργασία είναι περιορισμένα, το fine-tuning για τη συγκεκριμένη εργασία ή το pre-training προσαρμοσμένο στον τομέα του προβλήματος μπορεί να αποτελέσει ένα αποτελεσματικό ενδιάμεσο βήμα. Με την προ-εκπαίδευση (pre-training) του μοντέλου σε ένα σύνολο δεδομένων που σχετίζεται στενά με την εργασία-στόχο, το μοντέλο έχει τη δυνατότητα να μάθει μοτίβα και σχέσεις ειδικές για τον τομέα πριν την εφαρμογή του fine-tuning. Αυτή η προσέγγιση διευκολύνει την ταχύτερη προσαρμογή του μοντέλου και επιτυγχάνει καλύτερα αποτελέσματα κατά τη διαδικασία του fine-tuning, αφού το μοντέλο έχει ήδη εκτεθεί σε δεδομένα που σχετίζονται με το πρόβλημα που καλείται να λύσει. Για παράδειγμα, στον τομέα της επεξεργασίας φυσικής γλώσσας (ΕΦΓ), το pre-training ενός μοντέλου σε γενικά δεδομένα, όπως η Βικιπαίδεια, μπορεί να ακολουθηθεί από ένα δεύτερο στάδιο pre-training σε δεδομένα ενός συγκεκριμένου τομέα, όπως νομικά ή ιατρικά κείμενα. Αυτή η στρατηγική βελτιώνει σημαντικά την απόδοση του μοντέλου σε εξειδικευμένες εργασίες, καθώς το μοντέλο έχει ήδη εκπαιδευτεί σε δεδομένα που είναι συναφή με την τελική εργασία.

Αυτός ο συνδυασμός γενικού pre-training, pre-training για συγκεκριμένο τομέα και fine-tuning προσφέρει ένα ισχυρό πλαίσιο για την εκπαίδευση μοντέλων που γενικεύουν καλά σε όλα τα είδη προβλημάτων, ενώ παράλληλα αποδίδουν αποτελεσματικά σε εξειδικευμένα προβλήματα. Με την αξιοποίηση της γνώσης που αποκτήθηκε κατά το pre-training, το fine-tuning γίνεται πιο αποτελεσματικό και απαιτεί λιγότερα δεδομένα για συγκεκριμένα προβλήματα, καθιστώντας αυτή την προσέγγιση δύο σταδίων ιδιαίτερα αποτελεσματική σε ένα ευρύ φάσμα εφαρμογών μηχανικής μάθησης.

1.1.4 Low-Rank Adaptation and Quantized Low-Rank Adaptation

Η αποδοτική εκπαίδευση παραμέτρων (Parameter Efficient Fine-Tuning, PEFT) είναι μια προηγμένη τεχνική που στοχεύει στην προσαρμογή μεγάλων προ-εκπαιδευμένων μοντέλων σε συγκεκριμένες εργασίες, μειώνοντας την κατανάλωση πόρων. Σε αντίθεση με την παραδοσιακή εκπαίδευση, η οποία απαιτεί την ενημέρωση όλων των παραμέτρων του μοντέλου, η PEFT εστιάζει στην προσαρμογή ενός μικρού υποσυνόλου παραμέτρων, διατηρώντας τις περισσότερες από τις προϋπάρχουσες παραμέτρους σταθερές. Αυτό μειώνει σημαντικά τις απαιτήσεις σε υπολογιστική ισχύ και μνήμη, καθιστώντας τη διαδικασία εκπαίδευσης πιο αποδοτική, ιδιαίτερα για μοντέλα μεγάλης κλίμακας. Έτσι, η PEFT επιτρέπει την επίτευξη υψηλής απόδοσης χωρίς την ανάγκη για εκτεταμένους υπολογιστικούς πόρους, καθιστώντας τη μια ιδιαίτερα ελκυστική επιλογή για πολλές εφαρμογές.

Low-Rank Adaptation (LoRA) Η LoRA (Low-Rank Adaptation) είναι μια κορυφαία μέθοδος αποδοτικής εκπαίδευσης παραμέτρων (PEFT) που προσαρμόζει μεγάλα μοντέλα με την εισαγωγή πινάκων χαμηλής τάξης για την αποθήκευση γνώσης σχετικής με το πρόβλημα, ενώ διατηρεί τις περισσότερες από τις παραμέτρους του αρχικού μοντέλου αμετάβλητες. Αυτό επιτρέπει σημαντική μείωση της χρήσης μνήμης και των υπολογιστικών πόρων, καθιστώντας δυνατή την εκπαίδευση σε μηχανήματα με περιορισμένους πόρους. Η LoRA προσαρμόζει

μόνο μικρούς πίνακες που σχετίζονται με το πρόβλημα, οι οποίοι προσφέρουν τις απαραίτητες προσαρμογές στο μοντέλο χωρίς να απαιτείται ενημέρωση του συνόλου των παραμέτρων. Παρά το γεγονός ότι ενημερώνονται πολύ λιγότερες παράμετροι, η LoRA επιτυγχάνει επιδόσεις που είναι συγκρίσιμες με εκείνες της εκπαίδευσης όλων των παραμέτρων, καθιστώντας την ιδανική επιλογή για περιβάλλοντα με περιορισμένους πόρους.

Η αρθρωτή φύση της τεχνικής LoRA την καθιστά ιδανική για την αντιμετώπιση πολλαπλών προβλημάτων, καθώς επιτρέπει την ξεχωριστή αποθήκευση πινάκων για κάθε συγκεκριμένο πρόβλημα, χωρίς να επηρεάζεται το βασικό μοντέλο. Αυτή η προσέγγιση διευκολύνει τη διαδοχική μάθηση, επιτρέποντας την προσαρμογή σε διαφορετικά προβλήματα με την ενημέρωση μόνο των σχετικών πινάκων. Έτσι, η LoRA προσφέρει μια εξαιρετικά ευέλικτη και αποδοτική λύση για την προσαρμογή μεγάλων μοντέλων σε πολλούς τομείς, ενώ παράλληλα ελαχιστοποιεί τη χρήση υπολογιστικών πόρων.

Quantized Low-Rank Adaptation (QLoRA) Η QLoRA (Quantized Low-Rank Adaptation) επεκτείνει την αποδοτικότητα της LoRA, ενσωματώνοντας τεχνικές κβαντισμού για να μειώσει ακόμη περισσότερο τις απαιτήσεις μνήμης και υπολογιστικής ισχύος κατά τη λεπτομερή προσαρμογή μοντέλων. Η QLoRA εφαρμόζει κβάντιση 4-bit στα βάρη του μοντέλου, συμπιέζοντας τα σε χαμηλότερης ακρίβειας μορφή, ενώ ταυτόχρονα εξακολουθεί να ενημερώνει πίνακες χαμηλής τάξης για προσαρμογή σε συγκεκριμένα προβλήματα. Ο συνδυασμός της χαμηλής τάξης προσαρμογής και της κβάντισης των βαρών επιτρέπει τη λεπτομερή ρύθμιση μεγάλων μοντέλων με δισεκατομμύρια παραμέτρους, ακόμη και σε υλικό με περιορισμένη μνήμη, όπως GPU 12 GB, διατηρώντας παράλληλα υψηλή απόδοση.

Οι βασικές βελτιστοποιήσεις της τεχνικής QLoRA, όπως η κβάντιση 4-bit NF4, η κβάντιση κατά μπλοκ και η διπλή κβάντιση, προσφέρουν σημαντική εξοικονόμηση μνήμης, διατηρώντας παράλληλα υψηλή ακρίβεια. Παρόλο που η εκπαίδευση πραγματοποιείται με χαμηλότερη ακρίβεια, το τελικό μοντέλο επιτυγχάνει επιδόσεις σχεδόν ισάξιες με εκείνες των παραδοσιακών μεθόδων εκπαίδευσης. Ένα κρίσιμο πλεονέκτημα της QLoRA είναι ότι μειώνει τη χρήση μνήμης μόνο κατά τη διάρκεια της εκπαίδευσης, χωρίς πρόσθετο κόστος ή μείωση επιδόσεων κατά την εξαγωγή συμπερασμάτων (inference). Αυτό την καθιστά ιδανική λύση για την κλιμάκωση της λεπτομερούς προσαρμογής μεγάλων μοντέλων σε συστήματα με περιορισμένους πόρους, προσφέροντας αποδοτική εκπαίδευση χωρίς συμβιβασμούς στην απόδοση.

1.1.5 Έννοια της Προτροπής

Η **Προτροπή** (Prompting) αποτελεί μια σημαντική αλλαγή στη μηχανική μάθηση, μετακινώντας το επίκεντρο από την παραδοσιακή μάθηση με επίβλεψη σε μια μέθοδο που αξιοποιεί μεγάλα προ-εκπαιδευμένα γλωσσικά μοντέλα. Αντί τα μοντέλα να εκπαιδεύονται με επισήμασμένα δεδομένα για να προβλέπουν τις εξόδους από τις εισόδους, η μάθηση με προτροπές μετατρέπει τις εισόδους σε οδηγίες (προτροπές) που καθοδηγούν το γλωσσικό μοντέλο να παράγει τις επιθυμητές εξόδους. Οι προτροπές αυτές σχεδιάζονται με κατάλληλα κενά ή ενδείξεις, ώστε το μοντέλο να "συμπληρώσει" τις πληροφορίες που λείπουν, με βάση την εκτενή προ-εκπαίδευσή του και την κατανόηση της γλώσσας.

Αυτή η προσέγγιση επιτρέπει στα γλωσσικά μοντέλα να προσαρμόζονται γρήγορα σε νέα προβλήματα, συχνά με ελάχιστα δεδομένα, και είναι ιδιαίτερα αποτελεσματική για μάθηση με λίγα ή καθόλου παραδείγματα (few-shot και zero-shot learning). Τα μοντέλα αξιοποιούν τη γενική τους γνώση από την εκπαίδευσή τους σε μαζικά σύνολα δεδομένων κειμένου, παρέχοντας ευέλικτες και αποδοτικές λύσεις σε πλήθος εφαρμογών χωρίς την ανάγκη εκτεταμένης εκπαίδευσης από την αρχή.

1.1.6 Τεχνικές Προτροπής

Η επιλογή και εξερεύνηση κατάλληλων προτροπών, όπου τα μοντέλα μπορούν να μάθουν νέες εργασίες μόνο μέσω εντολών φυσικής γλώσσας, έχει προταθεί και χρησιμοποιηθεί ευρέως σε πολλές εφαρμογές. Ορισμένες από τις πιο γνωστές μεθόδους περιλαμβάνουν την εκμάθηση χωρίς παραδείγματα, την εκμάθηση με ένα παράδειγμα, την εκμάθηση με λίγα παραδείγματα, καθώς και την προτροπή με αλυσίδα συλλογιστικής μέχρι το τελικό αποτέλεσμα.

Μηδενικά παραδείγματα (Zero-shot Prompting)

Στην τεχνική με μηδενικά παραδείγματα (zero-shot prompting), το μοντέλο λαμβάνει απευθείας μια οδηγία χωρίς να του παρέχονται παραδείγματα. Το μοντέλο χρησιμοποιεί την προ-εκπαιδευμένη γνώση του για να

ολοκληρώσει την εργασία βασισμένο μόνο στην οδηγία.

Ένα παράδειγμα (One-shot Prompting) ή Λίγα παραδείγματα (Few-shot learning)

Η τεχνική με χρήση ενός παραδείγματος (one-shot learning) και η τεχνική λίγων παραδειγμάτων (few-shot learning) επεκτείνουν αυτή την προσέγγιση παρέχοντας ένα ή περισσότερα παραδείγματα στο μοντέλο. Στην εκμάθηση ενός παραδείγματος, το μοντέλο λαμβάνει ένα μόνο παράδειγμα της εργασίας. Στην εκμάθηση λίγων παραδειγμάτων, στο μοντέλο παρέχονται πολλαπλά παραδείγματα (συνήθως k παραδείγματα) ζευγών εισόδου-εξόδου (ερώτηση/περιεχόμενο και απάντηση), ακολουθούμενα από μια νέα είσοδο (άλυτη ερώτηση) για την οποία το μοντέλο καλείται να παράγει την κατάλληλη έξοδο (απάντηση).

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 cheese => ..... ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← example
3 cheese => ..... ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← examples
3 peppermint => menthe poivrée ←
4 plush girafe => girafe peluche ←
5 cheese => ..... ← prompt
```

Figure 1.1.1: (a) Μηδενικά παραδείγματα, (b) Ένα παράδειγμα, (c) Λίγα παραδείγματα [8]

Τα παραδείγματα για τις παραπάνω τεχνικές μπορούν να επιλεγούν με διάφορα κριτήρια:

- **Τυχαία επιλογή:** Τα παραδείγματα μπορούν να επιλεγούν τυχαία από τα δεδομένα εκπαίδευσης.
- **Σημασιολογική ομοιότητα:** Μπορούν να επιλεγούν παραδείγματα που είναι σημασιολογικά παρόμοια με τη νέα ερώτηση ή το πρόβλημα, καθοδηγώντας πιο αποτελεσματικά το μοντέλο.
- **Ποικιλομορφία:** Η επιλογή ενός ποικίλου φάσματος παραδειγμάτων μπορεί να βοηθήσει το μοντέλο να γενικεύσει καλύτερα σε διαφορετικές εισόδους.

Παραδείγματα με ύπαρξη της συλλογιστικής πορείας (Chain-of-Thought Prompting)

Η προτροπή με παράδειγμα που περιέχει και την συλλογιστική πορεία (Chain-of-Thought Prompting) είναι μια μέθοδος κατά την οποία το μοντέλο καθοδηγείται μέσω μιας αλληλουχίας ενδιάμεσων βημάτων συλλογισμού. Αυτή η προσέγγιση έχει αποδειχθεί ότι βελτιώνει σημαντικά την ικανότητα των μεγάλων γλωσσικών μοντέλων (MGF) να εκτελούν σύνθετα βήματα συλλογισμού, ενθαρρύνοντας το μοντέλο να αναλύει τα προβλήματα σε μικρότερα, διαχειρίσιμα βήματα [87].

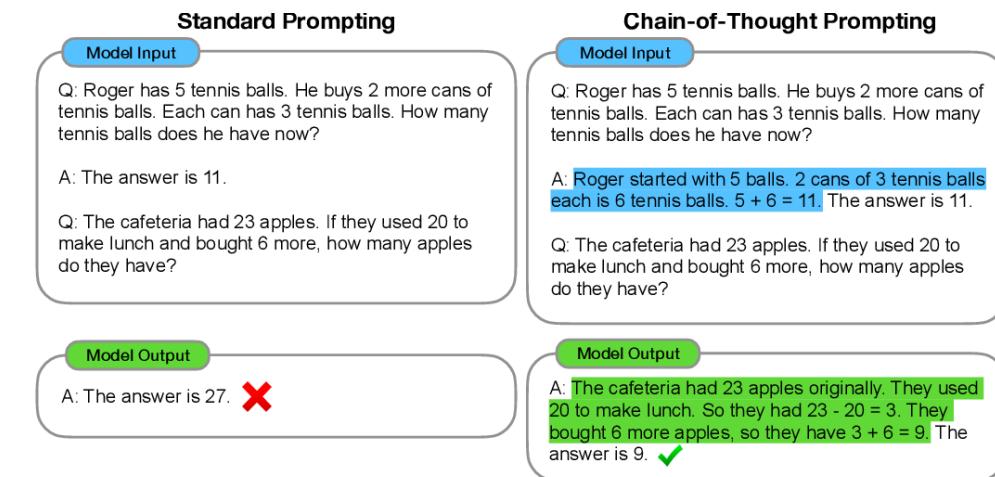


Figure 1.1.2: Παραδείγματα με ύπαρξη της συλλογιστικής πορείας [87]

1.1.7 Γραμμικός τρόπος σκέψης και Αντισυμβατικός - Δημιουργικός τρόπος σκέψης

Στην **Επεξεργασία Φυσικής Γλώσσας (NLP)**, η συλλογιστική είναι ζωτικής σημασίας για τα συστήματα ώστε να προχωρήσουν πέρα από την κατανόηση επιφανειακού επιπέδου και να επιτύχουν βαθύτερη κατανόηση. Ενώ τα **Μεγάλα Γλωσσικά Μοντέλα (MGM)** είναι πολύ ικανά σε πλήθος προβλημάτων, οι ικανότητές τους για συλλογισμό είναι περιορισμένες, προκαλώντας συζητήσεις σχετικά με το, αν πραγματικά δύνανται να χρησιμοποιήσουν συλλογισμό, για να οδηγηθούν στην απάντηση ή απλώς απομνημονεύουν μοτίβα [18, 4, 39, 98, 72, 81, 21, 97]. Για να ενισχυθούν οι ικανότητες συλλογισμού αυτών των μοντέλων, είναι σημαντικό να διερευνηθούν δύο τύποι σκέψης: ο **Γραμμικός** και ο **Αντισυμβατικός - Δημιουργικός** καθένας από τους οποίους εξυπηρετεί έναν μοναδικό ρόλο στην επίλυση προβλημάτων.

Γραμμική σκέψη στη συλλογιστική

Η **Γραμμική σκέψη** ακολουθεί μια λογική, γραμμική προσέγγιση στην επίλυση προβλημάτων, βασιζόμενη σε καθιερωμένους κανόνες και σε επαγωγικούς συλλογισμούς. Είναι αποτελεσματική όταν οι σχέσεις μεταξύ των υποθέσεων, των δεδομένων είναι σαφείς, επιτρέποντας στα μοντέλα να επεξεργάζονται συστηματικά τα προβλήματα βήμα προς βήμα.

Για παράδειγμα, πάρτε τον γρίφο:

«Ο Ρωμαίος και η Ιουλιέτα κείτονται νεκροί στο πάτωμα ενός δωματίου. Το δωμάτιο είναι τελείως στεγνό, αλλά υπάρχει σπασμένο γυαλί σε κοντινή απόσταση. Πώς πέθαναν;»

«Ο Ρωμαίος και η Ιουλιέτα κείτονται νεκροί στο πάτωμα ενός δωματίου. Το δωμάτιο είναι τελείως στεγνό, αλλά υπάρχει σπασμένο γυαλί σε κοντινή απόσταση. Πώς πέθαναν;» Η γραμμική σκέψη θα ανέλυε συστηματικά την κατάσταση, υποθέτοντας ότι ο Ρωμαίος και η Ιουλιέτα είναι άνθρωποι και εστιάζοντας στις πιθανές αιτίες θανάτου με βάση τα γνωστά γεγονότα. Ωστόσο, αυτή η προσέγγιση οδηγείται σε αδιέξοδο, όταν το πρόβλημα απαιτεί σκέψη πέρα από τις συμβατικές υποθέσεις.

Αντισυμβατική - Δημιουργική σκέψη στη συλλογιστική

Η **Αντισυμβατική σκέψη**, από την άλλη πλευρά, ενθαρρύνει τη δημιουργικότητα και απομακρύνεται από τις τυπικές παραδοχές. Στον ίδιο γρίφο, ένας δημιουργικός τρόπος σκέψης αποκαλύπτει ότι ο Ρωμαίος και η Ιουλιέτα είναι στην πραγματικότητα ψάρια και πέθαναν επειδή έσπασε η γυάλα τους (εξ ου και το σπασμένο γυαλί). Αυτή η αντισυμβατική λύση αμφισβητεί την υπόθεση ότι ο Ρωμαίος και η Ιουλιέτα είναι άνθρωποι, επιτρέποντας μια νέα εξήγηση. Αυτό το είδος σκέψης λοιπόν ευδοκιμεί όταν οι παραδοσιακές λογικές προσεγγίσεις αποτυγχάνουν, ενθαρρύνοντας τους λύτες προβλημάτων να επαναπροσδιορίσουν την κατάσταση και να βρουν δύσκολες και δημιουργικές συνδέσεις μεταξύ των δεδομένων του προβλήματος.

Διαφορές και εφαρμογές της Γραμμικής και της Αντισυμβατικής - Δημιουργικής σκέψης

Ο γραμμικός και ο αντισυμβατικός τρόπος σκέψης είναι πολύτιμοι, αλλά ταιριάζουν σε διαφορετικά είδη προβλημάτων. Η γραμμική σκέψη είναι ιδανική για εργασίες που απαιτούν αυστηρή λογική και εξαγωγή συμπερασμάτων, ενώ η δημιουργική σκέψη υπερέχει σε καταστάσεις που απαιτούν ευελιξία και δημιουργικότητα, ιδίως όταν οι αρχικές παραδοχές μπορεί να παραπλανήσουν. Στην Επεξεργασία Φυσικής Γλώσσας (ΕΦΓ), ο συνδυασμός και των δύο μορφών συλλογισμού ενισχύει τις ικανότητες των μοντέλων να αντιμετωπίζουν ένα ευρύ φάσμα προβλημάτων, οδηγώντας στην ευκολότερη επίλυσή τους.

1.2 Προτεινόμενη Μέθοδος 1 - SemEval-2024 Task 9

Παρουσιάζουμε την πρότασή μας για το **SemEval-2024 Task 9: «BRAINTEASER: A Novel Task Defying Common Sense»** [36], η οποία έχει γίνει δεκτή για το συνέδριο SemEval 2024 [59]. Εστιάζουμε στη βελτίωση των επιδόσεων των μοντέλων που βασίζονται σε μετασχηματιστές σε εργασίες μη σειριακής συλλογιστικής πορείας, στο Υπο-Πρόβλημα Α (παζλ σε δομή προτάσεων) και στο Υπο-Πρόβλημα Β (παζλ λέξεων), μέσω της εκπαίδευσης και της ανάλυσης των δυνατοτήτων συλλογισμού των μοντέλων.

Διευρυνούμε την επίδραση του μεγέθους των μοντέλων, των τεχνικών fine-tuning, καθώς και τις επιδράσεις διαφόρων υπερπαραμέτρων, προκειμένου να επιτύχουμε ανταγωνιστικά αποτελέσματα. Επιπλέον, παρέχουμε πληροφορίες για το πώς τα μοντέλα διαχειρίζονται προκλήσεις συλλογισμού που αντιπαράτίθενται στην κοινή λογική.

1.2.1 Συνεισφορές

1. Εκπαίδευση τόσο σε μικρά μοντέλα όσο και σε Μεγάλα Γλωσσικά Μοντέλα (MGM), ξεπερνώντας τις βασικές επιδόσεις αναφοράς.
2. Μετατροπή των προβλημάτων πολλαπλών επιλογών σε προβλήματα δυαδικής ταξινόμησης (πρόβλημα Σωστού - Λάθους).
3. Αξιολόγηση της απόδοσης των μοντέλων με βάση την προϋπάρχουσα γνώση τους ή την εμπειρία τους σε παρόμοια προβλήματα.
4. Ανάλυση των περιπτώσεων αποτυχίας για την κατανόηση των προκλήσεων που παρουσιάζουν οι μη σειριακές συλλογιστικές πορείες.

Ο κώδικάς μας είναι διαθέσιμος στο GitHub [εδώ](#).

1.2.2 Μέθοδοι

Η προσέγγισή μας επικεντρώνεται στη διαδικασία fine-tuning γλωσσικών μοντέλων σε δύο βασικές κατηγορίες: **μοντέλα κωδικοποιητή** και **μεγάλα γλωσσικά μοντέλα (MGM)**. Αρχικά, εφαρμόζουμε fine-tuning σε παραλλαγές μοντέλων **κωδικοποιητή**, όπως τα BERT [14], RoBERTa-large [52] και DeBERTaV3-base [29], χρησιμοποιώντας σύνολα δεδομένων που περιλαμβάνουν προκλήσεις συλλογισμού, πέραν του BrainTeaser. Πειραματιζόμαστε τόσο με την παραδοσιακή μορφή πολλαπλών επιλογών όσο και με την προσαρμογή σε πρόβλημα δυαδικής ταξινόμησης (Σωστού - Λάθους).

Στη συνέχεια, εφαρμόζουμε fine-tuning στα **MGM**, όπως τα Llama 2 [79], Phi-2 [26], και Mistral-7b [34], συγκρίνοντας τις επιδόσεις τους σε διάφορα μεγέθη μοντέλων, καθώς έχει αποδειχθεί ότι το μέγεθος του μοντέλου επηρεάζει τις ικανότητες συλλογισμού [79, 86].

Μοντέλα κωδικοποιητή

Για τα μοντέλα κωδικοποιητή, αξιολογούμε τόσο τις εκδόσεις vanilla (αρχικές εκδόσεις) όσο και τις προ-εκπαιδευμένες σε σύνολα δεδομένων κοινής λογικής πριν από τη διαδικασία fine-tuning στο BrainTeaser. Οι προ-εκπαιδευμένες εκδόσεις περιλαμβάνουν τα BERT-SE¹, RoBERTa-WNGRD² και DeBERTaV3-TS³,

¹JazibEijaz/bert-base-uncased-finetuned-semeval2020-task4b

²DeepPavlov/roberta-large-winogrande

³sileod/deberta-v3-large-tasksources-nli

τα οποία έχουν προηγουμένως προ-εκπαιδευτεί σε σύνολα δεδομένων όπως το CommonsenseQA [77], WinoGrande [68], ai2_arc [10], Physical Interaction (PIQA) [6], Choice of Plausible Alternatives (COPA) [22], καθώς και το σύνολο δεδομένων SemEval-2020 Task 4 [84], προκειμένου να ενισχυθούν οι ικανότητες κοινής λογικής των μοντέλων.

Αυτή η μεθοδολογία μάς επιτρέπει να συγκρίνουμε τις επιδόσεις των μοντέλων τόσο σε ακατέργαστη μορφή όσο και όταν έχουν εμπλουτιστεί με επιπλέον γνώσεις, διερευνώντας έτσι πώς η προ-εκπαίδευση σε δεδομένα κοινής λογικής επηρεάζει τη συλλογιστική τους απόδοση σε σύνθετες εργασίες όπως το BrainTeaser.

Για το πρόβλημα με δομή πολλαπλών επιλογών, αντιμετωπίσαμε κάθε ερώτηση ως πρόβλημα όπου το μοντέλο έπρεπε να επιλέξει μία σωστή απάντηση από τέσσερις διαθέσιμες επιλογές. Αυτή η προσέγγιση ενθάρρυνε τη δημιουργική σκέψη του μοντέλου, καθώς έπρεπε να αναλύσει όλες τις επιλογές ταυτόχρονα για να επιλέξει τη σωστή απάντηση. Από την άλλη πλευρά, η ρύθμιση δυαδικής ταξινόμησης (Σωστού-Λάθους) μετέτρεπε κάθε ερώτηση σε μια σειρά από δυαδικές αποφάσεις (σωστή ή λανθασμένη απάντηση). Ωστόσο, αυτή η μέθοδος απέδωσε υποδεέστερα αποτελέσματα, καθώς τα μοντέλα δυσκολεύονταν να αξιολογήσουν τις επιλογές μεμονωμένα, οδηγώντας μας να επιστρέψουμε στην αρχική προσέγγιση πολλαπλών επιλογών.

Μεγάλα Γλωσσικά Μοντέλα (MGM)

Με την τεχνική fine-tuning, εκπαιδεύσαμε MGM τελευταίας τεχνολογίας, όπως τα Llama 2, Phi-2 και Mistral-7b, σε προβλήματα πολλαπλών επιλογών. Τα μοντέλα αυτά προτρέπονταν να παρέχουν όχι μόνο την απάντηση, αλλά και μια εξήγηση κατά τη διάρκεια της διαδικασίας επεξεργασίας. Αυτή η προσέγγιση βελτίωσε τις επιδόσεις των μοντέλων, αποκαλύπτοντας παράλληλα αδυναμίες στη συλλογιστική τους.

Πειραματιστήκαμε επίσης με διαφορετικές υπερπαραμέτρους LoRA, όπως οι τιμές r και a (16, 32, 64, 128), προκειμένου να βρούμε τον βέλτιστο συνδυασμό για τις συγκεκριμένες εργασίες. Περισσότερες λεπτομέρειες σχετικά με τις υπερπαραμέτρους και τις ιδιαιτερότητες των τεχνικών προτροπής που χρησιμοποιήθηκαν, μπορείτε να βρείτε στις ενότητες 1.3.4, 1.3.4, 1.3.3.

1.3 Πειραματικό Μέρος - SemEval-2024 Task 9

Ας εξετάσουμε τον παρακάτω γρίφο:

«Ποιος ξυρίζει κάθε μέρα αλλά εξακολουθεί να έχει μούσι;»

Η απάντηση, «Ένας κουρέας», αναδεικνύει έναν **αντισυμβατικό τρόπο σκέψης**, ο οποίος προκαλεί τις αρχικές παραδοχές. Στην περίπτωση αυτή, υποθέτουμε ότι ο κουρέας ξυρίζει τους άλλους, αλλά διατηρεί το δικό του μούσι. Αυτός ο τύπος μη γραμμικής και δημιουργικής συλλογιστικής είναι κεντρικός στον διαγωνισμό **SemEval-2024 Task 9**, με τίτλο **"BRAINTEASER: A Novel Task Defying Common Sense"** [36], που αξιολογεί την ικανότητα των **Μεγάλων Γλωσσικών Μοντέλων (MGM)** τόσο στη **γραμμική σκέψη** (λογική συλλογιστική) όσο και στη **δημιουργική σκέψη** (δημιουργική επίλυση προβλημάτων).

Τα πειράματά μας επικεντρώνονται στην ενίσχυση των επιδόσεων των MGM μέσω τεχνικών λεπτομερούς ρύθμισης (fine-tuning) και μάθησης μεταφοράς (transfer learning). Ο διαγωνισμός περιλαμβάνει δύο βασικά υποπροβλήματα:

1. **Υπο-πρόβλημα Α (παζλ σε μορφή προτάσεων):** Προβλήματα που απαιτούν λογική και δημιουργική σκέψη σε επίπεδο προτάσεων.
2. **Υπο-πρόβλημα Β (παζλ λέξεων):** Προβλήματα σε επίπεδο λέξεων που απαιτούν την ερμηνεία πολλαπλών εννοιών και εναλλακτικών οπτικών.

Τα πειράματά μας στοχεύουν να απαντήσουν στα εξής ερευνητικά ερωτήματα:

- **EE1:** Μπορεί η τεχνική fine-tuning με τη χρήση συνόλων δεδομένων που σχετίζονται με τη συλλογιστική να βελτιώσει τις ικανότητες δημιουργικής σκέψης των μοντέλων, συγκριτικά με την απλή χρήση προτροπών;
- **EE2:** Πώς μπορεί η μάθηση μεταφοράς να ενισχύσει τις επιδόσεις των MGM σε προβλήματα δημιουργικής και γραμμικής συλλογιστικής;

Αυτό το κεφάλαιο εξετάζει πώς το μέγεθος του μοντέλου, η προ-εκπαίδευση και οι τεχνικές fine-tuning επηρεάζουν τις επιδόσεις των M_{GM}, παρέχοντας σημαντικές πληροφορίες για τη βελτίωση των ικανοτήτων τους στη δημιουργική σκέψη και την επίλυση μη παραδοσιακών προβλημάτων.

1.3.1 Σύνολα Δεδομένων και Μετρικές

Σύνολα Δεδομένων

Το πρόβλημα που εισάγεται από το **BrainTeaser** SemEval-2024 [38, 36] αποτελείται από γρίφους που χρειάζονται αντισυμβατική σκέψη και παρουσιάζονται ως ερωτήσεις πολλαπλής επιλογής (QAs), με τέσσερις επιλογές: μία σωστή απάντηση και τρεις λανθασμένες (αποπροσανατολισμούς), συμπεριλαμβανομένου του «Κανένα από τα παραπάνω». Το πρόβλημα χωρίζεται σε δύο επιμέρους υπο-προβλήματα: **Υπο-πρόβλημα A (παζλ προτάσεων)** και **Υπο-πρόβλημα B (παζλ λέξεων)**.

Επιπλέον, το σύνολο δεδομένων περιέχει παραλλαγές που δημιουργήθηκαν τροποποιώντας τους αρχικούς γρίφους, διατηρώντας όμως τα ίδια μονοπάτια συλλογισμού. Οι τροποποιήσεις περιλαμβάνουν:

1. **Σημασιολογική ανακατασκευή:** Επαναδιατύπωση της ερώτησης χωρίς να τροποποιούνται οι απαντήσεις ή οι αποπροσανατολιστικές λανθασμένες απαντήσεις (πχ. παράφραση της ερώτησης).
2. **Ανακατασκευή περιεχομένου:** Αλλαγή του περιεχομένου - πλαισίου του γρίφου διατηρώντας την αρχική συλλογιστική πορεία για την εύρεση της απάντησης.

Κάθε δείγμα δεδομένων περιλαμβάνει την αρχική, τη σημασιολογική και την ανακατασκευασμένη σε πλαίσιο εκδοχή του ίδιου γρίφου. Ένα παράδειγμα αυτών των τριπλετών παρουσιάζεται στον πίνακα 1.1.

Question	Choice
<i>Original</i>	
What kind of nut has no shell?	A peanut.
	A doughnut.
	A walnut.
	None of above.
<i>Semantic Reconstruction</i>	
Which nut doesn't have a shell?	A doughnut.
	A walnut.
	A peanut.
	None of above.
<i>Context Reconstruction</i>	
Which type of bell doesn't make a sound?	A fire bell.
	A cow bell.
	A bluebell.
	None of above.

Table 1.1: Απεικόνιση της δομής του συνόλου δεδομένων κάθε επιμέρους εργασίας, όπου παρουσιάζεται η αρχική δήλωση μαζί με τις δύο ανακατασκευές της. Οι σωστές απαντήσεις παρουσιάζονται με **έντονη γραφή**

Υπο-πρόβλημα A: Παζλ προτάσεων Σε αυτή την υποεργασία, τα ζεύγη προτάσεων είναι σχεδιασμένα έτσι ώστε να είναι εύκολα για τον άνθρωπο να τα λύσει, αλλά δύσκολα για τα συστήματα. Το σύνολο εκπαίδευσης περιέχει 169 σύνολα ερωτήσεων πολλαπλής επιλογής, με σημασιολογικές και συμφραστικές ανακατασκευές, με αποτέλεσμα να προκύπτουν συνολικά 507 ερωτήσεις (3×169). Παραδείγματα αυτών των γρίφων προτάσεων παρέχονται στον πίνακα 1.2 (αριστερά).

Υπο-πρόβλημα B: Παζλ λέξεων Αυτή η υποεργασία περιλαμβάνει γρίφους λέξεων όπου η απάντηση βασίζεται στη σύνθεση της λέξης και όχι στην προεπιλεγμένη σημασία της. Το σύνολο δεδομένων περιλαμβάνει

132 σύνολα QA πολλαπλών επιλογών, με σημασιολογικές και ανακατασκευές πλαίσιου, συνολικού αριθμού 396 QA (3×132). Αυτοί οι γρίφοι περιλαμβάνουν λογοπαίγνια, ομόφωνα και διαφορούμενες λέξεις, θέτοντας προκλήσεις τόσο για τα συστήματα όσο και για τους ανθρώπους (βλ. παραδείγματα στον Πίνακα 1.2, δεξιά).

Sentence Puzzle		Word Puzzle	
Question	Choice	Question	Choice
A man shaves everyday, yet keeps his beard long.	He is a barber.	What has toes but no feet or legs?	Cabbages.
	He wants to maintain his appearance.		Tomatoes.
	He wants his girlfriend to buy him a razor.		Onions.
	None of above.		None of above.
You go to the doctor because you're sick, and he gives you three medicines to take every half hour. How long do the drugs keep you going?	One and a half hours.	What did the little lobster get on its math test?	Sea-plus.
	Two hours.		Very-bad.
	An hour.		Very-Good.
	None of above.		None of above.
How many times can you deduct 10 from 100?	Once.	What's the beginning of an argument?	The letter T.
	Infinite time.		The letter A.
	Twice.		The letter U.
	None of above.		None of above.

Table 1.2: Παραδείγματα ερωτήσεων που απεικονίζουν και τα δύο επιμέρους υπο-προβλήματα, όπου οι σωστές απαντήσεις παρουσιάζονται **με έντονη γραφή**. Τα παραδείγματα στα αριστερά αφορούν το υπο-πρόβλημα A: Παζλ προτάσεων, ενώ εκείνα στα δεξιά αντιστοιχούν στο υπο-πρόβλημα B: Παζλ λέξεων.

Στατιστικά στοιχεία δεδομένων

Το σύνολο δεδομένων χωρίζεται σε σύνολα εκπαίδευσης, ανάπτυξης (που χρησιμοποιούνται κατά την εξάσκηση) και κρυφά σύνολα δοκιμών. Τα στατιστικά στοιχεία των δεδομένων συνοψίζονται στον πίνακα 1.3. Ο πίνακας κατάταξης παρέμεινε κρυμμένος κατά τη διάρκεια της φάσης αξιολόγησης.

Sub-task	Train	Dev	Test
A - Sentence Puzzle	507	120	120
B - Word Puzzle	396	96	96

Table 1.3: Στατιστικά στοιχεία δεδομένων.

Μετρικές

Και τα δύο υπο-προβλήματα αξιολογούνται με τη χρήση της **ακρίβειας** μέσω δύο διαφορετικών προσεγγίσεων:

1. **Ακρίβεια με βάση την περίπτωση:** Αυτή η μετρική αξιολογεί κάθε ερώτηση μεμονωμένα, συμπεριλαμβανομένων των αρχικών ερωτήσεων και των σημασιολογικών και πλαισιακών ανακατασκευών τους. Η ακρίβεια με βάση την περίπτωση παρέχει λεπτομερείς πληροφορίες για την ικανότητα του μοντέλου να αιτιολογεί διαφορετικά σενάρια και να επιλύει ατομικές προκλήσεις με σωστή λογική.
2. **Ακρίβεια με βάση την ομάδα:** Αυτή η μετρική αξιολογεί ομάδες τριών συναφών ερωτήσεων (αρχικές ερωτήσεις, σημασιολογικές ανακατασκευές και ανακατασκευές περιεχομένου). Ένα μοντέλο κερδίζει βαθμολογία 1, μόνο αν λύσει σωστά όλες τις ερωτήσεις εντός της ομάδας. Αυτή η μετρική παρέχει μια πιο συνολική εικόνα της απόδοσης του μοντέλου, αναδεικνύοντας τη σταθερότητα της συλλογιστικής του σε διαφορετικά πλαίσια.

Συνδυάζοντας την ακρίβεια με βάση την περίπτωση και την ακρίβεια με βάση την ομάδα, αυτές οι μετρικές προσφέρουν μια ολοκληρωμένη αξιολόγηση της ικανότητας ενός μοντέλου να χειρίζεται προκλήσεις μη σειριακών συλλογιστικών πορειών και δημιουργικής σκέψης και στα δύο υπο-προβλήματα του διαγωνισμού.

1.3.2 Μοντέλα ως Αναφορά

Για τα δύο υπο-προβλήματα του **BRAINTEASER**, χρησιμοποιήσαμε τρία βασικά μοντέλα ως αναφορά: **Human**, **ChatGPT**, και **RoBERTa-L**. Αυτά τα μοντέλα αξιολογήθηκαν σε μορφή πολλαπλών επιλογών

χωρίς να έχουν υποβληθεί σε διαδικασία εκπαίδευσης (fine-tuning). Το βασικό μοντέλο **ανθρώπου** λειτουργεί ως το άνω όριο αναφοράς, αντιπροσωπεύοντας την ιδανική απόδοση στην εκτέλεση της εργασίας. Αντίθετα, τα **ChatGPT** και **RoBERTa-L** χρησιμοποιήθηκαν ως αυτοματοποιημένες γραμμές βάσης για τη μέτρηση των επιδόσεων χωρίς προηγούμενη προσαρμογή ή fine-tuning.

Στον σχολιασμό των αποτελεσμάτων, οι βαθμολογίες αναφοράς θα επισημαίνονται με γκρι χρώμα, ώστε να διευκολύνεται η σύγκριση με τα μοντέλα που έχουν υποστεί fine-tuning. Αυτό θα καταδεικνύει ξεκάθαρα τον αντίκτυπο της τεχνικής fine-tuning στην απόδοση των MGM, αναδεικνύοντας τις βελτιώσεις που επιφέρει στην ικανότητα επίλυσης των προκλήσεων του BRAINTEASER.

1.3.3 Λεπτομέρειες Προτροπής

Εδώ, παρέχουμε μια ολοκληρωμένη επισκόπηση της προτροπής που χρησιμοποιήθηκε με συνέπεια καθ' όλη τη διάρκεια της τεχνικής fine-tuning των MGM, η οποία τελικά οδήγησε στη βέλτιστη απόδοση και στα δύο υπο-προβλήματα. Τα MGM διερευνήθηκαν αποκλειστικά με τη μορφή πολλαπλής επιλογής, καθώς αυτή η ρύθμιση προσέφερε το απαραίτητο πλαίσιο αναφορικά με το περιεχόμενο και τις πληροφορίες για τη συλλογιστική μέσω των παρεχόμενων επιλογών. Διατηρώντας τη συνέπεια στη δομή των προτροπών κατά τη διάρκεια της τεχνικής fine-tuning, μπορέσαμε να ενισχύσουμε αποτελεσματικά την ικανότητα των μοντέλων να χειρίζονται την πολυπλοκότητα και των δύο υπο-προβλημάτων, οδηγώντας σε βελτιωμένη απόδοση.

System Prompt:

Below is an instruction that describes a multiple choice task. Answer the following multiple choice question by giving the most appropriate response. Answer should be one among options provided after the question. Select the most suitable answer while making the necessary assumptions. Give only answer and a short explanation of two or three sentences. Nothing else.

User Prompt:

Question: {question}

- 1) {a}
- 2) {b}
- 3) {c}
- 4) {d}

Answer:

The correct answer is: {label}) {answer}

Ακολουθεί η επεξήγηση κάθε παραμέτρου που χρησιμοποιείται στην προτροπή του χρήστη:

- **question:** Αυτή η μεταβλητή αντιπροσωπεύει την πραγματική ερώτηση που τέθηκε στο μοντέλο. Χρησιμεύει ως η κύρια είσοδος για την οποία το μοντέλο πρέπει να σκεφτεί και να παράγει μια κατάλληλη απάντηση.
- **{a}, {b}, {c}, {d}:** Αυτές οι μεταβλητές αντιστοιχούν στις τέσσερις πιθανές επιλογές απάντησης που παρουσιάζονται για κάθε ερώτηση σε μορφή πολλαπλών επιλογών. Κάθε επιλογή συνδέεται με ένα αριθμητικό αναγνωριστικό (1, 2, 3 ή 4) και αντιπροσωπεύει μια ξεχωριστή επιλογή που μπορεί να επιλέξει το μοντέλο.
- **{label}:** Αυτή η μεταβλητή δηλώνει τον αριθμό (1, 2, 3 ή 4) που αντιστοιχεί στη σωστή επιλογή απάντησης μεταξύ των τεσσάρων παρεχόμενων επιλογών. Δείχνει το δείκτη της σωστής επιλογής.
- **{answer}:** Αυτή η μεταβλητή περιέχει το κείμενο της σωστής επιλογής απάντησης. Αντιστοιχεί στο περιεχόμενο της επιλεγμένης σωστής επιλογής (είτε **{a}**, **{b}**, **{c}**, ή **{d}**) όπως προσδιορίζεται από την ετικέτα.

Εδώ, η προτροπή παρουσιάζει μια ερώτηση μαζί με τέσσερις πιθανές επιλογές απάντησης και το μοντέλο είναι ρυθμισμένο ώστε να παράγει τη σωστή απάντηση, επιλέγοντας την κατάλληλη ετικέτα και το αντίστοιχο κείμενο απάντησης.

1.3.4 Περιγραφή Πειραμάτων

Στα πειράματά μας, χρησιμοποιήσαμε την πλατφόρμα Google Colab και το Kaggle, αξιοποιώντας διάφορα πακέτα Python ανοιχτού κώδικα, όπως τα Transformers, TRL (Transformer Reinforcement Learning) [91], PEFT (Parameter-Efficient Fine-Tuning) [57], BitsAndBytes, Accelerate [25], και Sentence-Transformers.

Κωδικοποιητές

Το μοντέλο BERT-SE ⁴ Κατά τη διάρκεια της τεχνικής fine-tuning, χρησιμοποιήθηκε ρυθμός μάθησης $3e^{-5}$, με μέγεθος παρτίδας 16 δειγμάτων, τα οποία επεξεργάστηκαν σε κάθε επανάληψη, κατά τη διάρκεια 3 εποχών. Αυτή η διαδικασία είχε ως στόχο την προσαρμογή του προ-εκπαιδευμένου μοντέλου ώστε να ταιριάζει καλύτερα στο εκάστοτε υπο-πρόβλημά μας. Ο βελτιστοποιητής μας (optimizer) ήταν ο AdamW και ο ρυθμιστής μάθησης (learning scheduler) ήταν γραμμικός. Η ίδια ρύθμιση χρησιμοποιήθηκε και για την τεχνική fine-tuning του κωδικοποιητή BERT.

Το μοντέλο RoBERTa-WNGRD ⁵ υποβλήθηκε σε fine-tuning στο σύνολο εκπαίδευσης κάθε συνόλου δεδομένων, χρησιμοποιώντας ρυθμό μάθησης $3e^{-5}$, μέγεθος παρτίδας 16 και τρέχοντας για 3 εποχές. Ο optimizer ήταν επίσης AdamW και ο ρυθμιστής μάθησης (learning scheduler) ήταν γραμμικός. Το RoBERTa-large ρυθμίστηκε λεπτομερώς στο σύνολο εκπαίδευσης του συγκεκριμένου συνόλου δεδομένων κάθε υπο-προβλήματος, χρησιμοποιώντας πανομοιότυπες διαμορφώσεις.

Το μοντέλο DeBERTaV3-TS ⁶, όπως και το DeBERTaV3-base, υποβλήθηκε σε fine-tuning παρόμοια με το σύστημα RoBERTa-WNGRD, με μόνη διαφορά το μέγεθος της παρτίδας, το οποίο ορίστηκε σε 4.

MGFM

Το μοντέλο Phi-2 ⁷ υποβλήθηκε σε fine-tuning χρησιμοποιώντας τη μορφή της προτροπής που περιγράφεται στην ενότητα *Λεπτομέρειες Προτροπής* 1.3.3. Η διαδικασία της τεχνικής fine-tuning περιελάμβανε τον καθορισμό ρυθμού μάθησης $2e^{-5}$ και μεγέθους παρτίδας 2, με το μοντέλο να εκπαιδεύεται για 250 βήματα. Πραγματοποιήσαμε πειράματα με διαφορετικές διαμορφώσεις των r και lora_alpha , περιλαμβάνοντας συνδυασμούς όπως $r = 64, 128$ και $\text{lora_alpha} = 64, 128$. Το ποσοστό του dropout ρυθμίστηκε σταθερά στο 0,1 σε όλα τα πειράματα. Χρησιμοποιήσαμε έναν βελτιστοποιητή (optimizer) AdamW και έναν ρυθμιστή σταθερής μάθησης. Παρά τα πολλά υποσχόμενα benchmarks που συνόδευαν την κυκλοφορία του, η απόδοση του μοντέλου κατά την εξαγωγή συμπερασμάτων στο κρυφό σύνολο δεδομένων και των δύο υπο-προβλημάτων ήταν υποδεέστερη, σημειώνοντας χαμηλότερη επίδοση σε σύγκριση με τους κωδικοποιητές που αναφέρθηκαν παραπάνω. Αυτή η ασυμφωνία εγείρει την πιθανότητα, η οποία υποστηρίζεται από διάφορες αναφορές, ότι η διαδικασία εκπαίδευσης του μοντέλου με τη χρήση μεθόδων όπως η χβάντιση και η LoRA μπορεί να μην είναι ακόμη πλήρως βελτιστοποιημένη.

Και οι δύο παραλλαγές του Llama 2 ⁸, με 7 δισεκατομμύρια και 13 δισεκατομμύρια παραμέτρους, υποβλήθηκαν στην ίδια διαδικασία fine-tuning που περιγράφηκε προηγουμένως, χρησιμοποιώντας την τεχνική QLoRA. Η διαδικασία fine-tuning ακολούθησε τη μορφή της προτροπής που περιγράφεται στην ενότητα 1.3.3 (*Λεπτομέρειες Προτροπής*), χρησιμοποιώντας ρυθμό μάθησης $2e^{-5}$ και μέγεθος παρτίδας 1, με κάθε μοντέλο να εκπαιδεύεται για 250 βήματα. Παρά τον πειραματισμό με διάφορους συνδυασμούς τιμών για τα r και a (32, 64, 128), ενώ το dropout ρυθμίστηκε σταθερά στο 0.1, τα αποτελέσματα ήταν απογοητευτικά. Ως μοντέλο παραγωγής κειμένου, το Llama 2 παρήγαγε επεξηγήσεις για κάθε προτροπή πολλαπλής επιλογής. Ωστόσο, ακόμη και όταν προέβλεπε λανθασμένα μια επιλογή ως σωστή, οι παραγόμενες εξηγήσεις συχνά δεν είχαν λογική συνοχή. Πολλές εξηγήσεις που παρήχθησαν κατά τη φάση εξαγωγής συμπερασμάτων ήταν άσχετες με το πλαίσιο του εκάστοτε γρίφου - προβλήματος, υποδεικνύοντας την αποτυχία να συλλάβουν τη συλλογιστική πορεία των περισσότερων ερωτήσεων πολλαπλής επιλογής. Συνοπτικά, και οι δύο παραλλαγές του Llama 2, παρά τη μεγάλη κλίμακα τους, αποδείχθηκαν ανίκανες να κατανοήσουν και να αιτιολογήσουν αποτελεσματικά τις παρεχόμενες ερωτήσεις πολλαπλής επιλογής.

⁴JazibEijaz/bert-base-uncased-finetuned-semeval2020-task4b-append-e3-b32-l4e5

⁵DeepPavlov/roberta-large-winogrande

⁶sileod/deberta-v3-large-tasks-source-nli

⁷microsoft/phi-2

⁸meta-llama/Llama-2

Το μοντέλο Mistral-7b⁹ υπερέιχε σημαντικά έναντι όλων των άλλων. Πριν από τη διαδικασία fine-tuning, εφαρμόσαμε την τεχνική QLoRA. Χρησιμοποιώντας ρυθμό μάθησης $2e^{-5}$ και μέγεθος παρτίδας 2, κάθε μοντέλο υποβλήθηκε σε λεπτομερή ρύθμιση για 250 βήματα χρησιμοποιώντας το σύνολο εκπαίδευσης κάθε συνόλου δεδομένων των υπο-προβλημάτων. Τα αρχικά αποτελέσματα ήταν ελπιδοφόρα. Κατά τη διάρκεια του πειραματισμού με τις παραμέτρους r και a , διατηρώντας παράλληλα ένα dropout 0,1, αναδύθηκαν ορισμένα πρότυπα. Συγκεκριμένα, παρατηρήσαμε εξηγήσεις και βαθμολογίες υψηλότερης ποιότητας όταν χρησιμοποιούσαμε υψηλότερες τιμές κατάταξης (r), που κυμαίνονταν από (16, 32, 64, 128). Αυτό το αποτέλεσμα ήταν αναμενόμενο, καθώς οι υψηλότερες τιμές κατάταξης αντιστοιχούν σε αλλαγές βαρών υψηλότερης ακρίβειας, με αποτέλεσμα καλύτερης ποιότητας ανανέωση βαρών και συνολική απόδοση του μοντέλου. Είναι ενδιαφέρον ότι, όταν ο λόγος a/r ήταν χαμηλός (0.5 - 1), οι εξηγήσεις διατηρούσαν υψηλή ποιότητα ανεξάρτητα από τις προβλέψεις, υποδηλώνοντας μια συνεκτική πορεία συλλογισμού ακόμη και αν η προβλεπόμενη επιλογή ήταν λανθασμένη. Ωστόσο, ο καθορισμός του λόγου a/r σε 2 ή 4 ενδεχομένως ενίσχυσε τα αποτελέσματα, υποδηλώνοντας ισχυρότερη επιρροή από τα στρώματα QLoRA στο βασικό μοντέλο. Ωστόσο, αυτή η προσαρμογή οδήγησε σε μείωση της ποιότητας των εξηγήσεων. Η βελτίωση θα μπορούσε να αποδοθεί στη χαμηλή εγγενή διαστατικότητα του μοντέλου. Παρά τις πολλές παραμέτρους, η πραγματική διαστατικότητα των αναπαραστάσεων που μαθαίνει το μοντέλο είναι χαμηλή. Κατά συνέπεια, μετά τη διεξαγωγή αρκετών πειραμάτων, το μοντέλο με τις καλύτερες επιδόσεις όσον αφορά στους γρίφους λέξεων, ευθυγραμμίζεται με αυτή την έννοια. Μετά τη διεξαγωγή πολυάριθμων δοκιμών, επιτύχαμε τις καλύτερες επιδόσεις με το πρώτο μοντέλο χρησιμοποιώντας $r=128$ και $\alpha=128$, και το τρίτο καλύτερο χρησιμοποιώντας $r=64$ και $\alpha=32$. Αυτά τα μοντέλα συμβολίζονται ως Mistral-7b_lora_r_lora_a, αντιπροσωπεύοντας τις διαμορφώσεις Mistral-7b_128_128 και Mistral-7b_64_32, αντίστοιχα.

Το μοντέλο Mistral-8x7b¹⁰ δεν εξερευνήθηκε σε βάθος, ωστόσο τα αρχικά αποτελέσματα ήταν ελπιδοφόρα, παρά τις περιορισμένες διαμορφώσεις. Περαιτέρω πειραματισμός με διάφορες ρυθμίσεις υπερπαραμέτρων μπορεί να οδηγήσει σε βελτιωμένη απόδοση. Στη μοναδική μας προσπάθεια με αυτό το σύστημα, χρησιμοποιήσαμε ρυθμό μάθησης $2e^{-5}$ και μέγεθος παρτίδας 2, με fine-tuning των μοντέλων για 250 βήματα χρησιμοποιώντας, το σύνολο εκπαίδευσης κάθε συνόλου δεδομένων των υπο-προβλημάτων. Τόσο το r όσο και το a ορίστηκαν σε 128, συνοδευόμενα από dropout 0,1. Αυτή η διαμόρφωση επιλέχθηκε με βάση τις τιμές r και a του μοντέλου με τις καλύτερες επιδόσεις και στις δύο υποεργασίες, Mistral-7b. Παρά τη μεγαλύτερη κλίμακά του, το Mistral-8x7b πέτυχε τη δεύτερη καλύτερη ακρίβεια κατά τη διάρκεια των κρυφών συνόλων δοκιμών όσον αφορά στο πρώτο υπο-πρόβλημα, αφήνοντας πίσω του τη μικρότερη παραλλαγή του, το Mistral-7b. Αυτό το μοντέλο αναφέρεται στον πίνακα αποτελεσμάτων και των δύο υπο-προβλημάτων ως Mistral-8x7b_128_128. Περαιτέρω πειραματισμός με διάφορες διαμορφώσεις μπορεί να αποφέρει βελτιώσεις, ιδίως όταν αξιοποιείται η χαμηλή εγγενής διαστατικότητα και ο πλεονασμός που ενυπάρχει στο μοντέλο.

Υπερπαραμέτροι QLoRA

Χρησιμοποιήσαμε την τεχνική QLoRA [13] για να βελτιστοποιήσουμε τα μοντέλα κβαντίζοντάς τα σε **ακρίβεια 4 bit** (μέσω BitsAndBytes) και εφαρμόζοντας **LoRA** [31] μέσω της βιβλιοθήκης PEFT. Αυτή η διαδικασία **Quantized LoRA (QLoRA)** μειώνει τη χρήση μνήμης και τις υπολογιστικές απαιτήσεις, καθιστώντας τον ιδανικό για την χρήση μεγάλων μοντέλων σε υλικό με περιορισμένους πόρους.

Οι βασικές υπερπαραμέτροι περιλαμβάνουν:

- **Rank (r):** Ελέγχει τη διαστατικότητα της προσέγγισης χαμηλού βαθμού.
- **Άλφα (a):** Ένας παράγοντας κλιμάκωσης που εξισορροπεί τα ειδικά για την εργασία βάρη έναντι των προ-εκπαιδευμένων βαρών.

Χαμηλότερες τιμές r οδήγησαν μερικές φορές σε καλύτερα αποτελέσματα λόγω ενός αποτελέσματος κανονικοποίησης, το οποίο βοηθά στην αποφυγή της υπερβολικής προσαρμογής, ιδίως με μικρότερα σύνολα δεδομένων. Ενθαρρύνοντας το μοντέλο να εστιάζει σε ουσιαστική μοτίβα, οι προσεγγίσεις χαμηλότερης τάξης προωθούσαν την καλύτερη γενίκευση σε άορατα δεδομένα.

⁹ [mistralai/Mistral-7B-v0.1](https://mistral.ai/Mistral-7B-v0.1)

¹⁰ [mistralai/Mistral-8x7B-v0.1](https://mistral.ai/Mistral-8x7B-v0.1)

1.3.5 Αποτελέσματα

Πειραματικά αποτελέσματα και ανάλυση

Οι μετρικές μας για το υπο-πρόβλημα σχετικό με παζλ προτάσεων παρουσιάζονται στον Πίνακα 1.4 και για το υπο-πρόβλημα με παζλ λέξεων στον Πίνακα 1.5 μαζί με τις επιδόσεις αναφοράς των μοντέλων που αναφέραμε παραπάνω. Είναι ενδιαφέρον ότι η απόδοση του προβλήματος δυαδικής ταξινόμησης (πρόβλημα Σωστού-Λάθους) είναι σημαντικά χαμηλότερη από εκείνη του προβλήματος πολλαπλών επιλογών. Αρχικά, αυτή η συμπεριφορά φαινόταν αντιφατική, καθώς φαινόταν ευκολότερο να προσδιοριστεί αν μια ερώτηση είναι σωστή ή όχι από το να επιλεγεί η σωστή απάντηση από τέσσερις διαφορετικές επιλογές. Ωστόσο, η υπόθεση αυτή δεν είναι ακριβής. Σκεφτείτε τον γρίφο με τις λέξεις: ‘What is the capital in France?’. Με την πρώτη ματιά, η επιλογή ‘F’ φαίνεται λανθασμένη, αλλά όταν εξετάζονται οι επιλογές ‘F,’ ‘E,’ ‘A,’ και ‘Καμία από τις παραπάνω’, η ‘F’ αναδεικνύεται ως η μόνη σωστή απάντηση, καθώς γίνεται φανερό ότι η ερώτηση αναφέρεται στο κεφαλαίο γράμμα και όχι στην πρωτεύουσα πόλη. Επομένως, οι ποικίλες επιλογές παρέχουν κρίσιμο πλαίσιο στα μοντέλα, εξηγώντας την ανώτερη απόδοση των μοντέλων σε προβλήματα μορφής πολλαπλής επιλογής. Αυτή η έλλειψη πλαισίου είναι ο λόγος για τον οποίο απέχουμε από την περαιτέρω διερεύνηση αυτής της μεθοδολογίας σε όλα τα μοντέλα της μελέτης μας.

System	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Overall
Multi-class classification problem						
Human	.907	.907	.944	.907	.889	.920
ChatGPT	.608	.593	.679	.507	.397	.627
RoBERTa-L	.435	.402	.464	.330	.201	.434
Mistral-7b_128_128	.850	.825	.775	.825	.700	.817
Mistral-7b_64_128	.850	.825	.775	.825	.700	.817
Mistral-7b_16_64	.800	.800	.850	.750	.725	.817
Mixtral-8x7b_128_128	.850	.825	.725	.800	.700	.800
Llama 2-7b_64_128	.725	.650	.700	.575	.475	.692
Llama 2-13b_64_64	.665	.614	.645	.550	.400	.641
Llama 2-7b_64_64	.625	.600	.675	.550	.400	.633
Llama 2-7b_64_32	.250	.250	.425	.075	.000	.308
Phi-2_64_128	.625	.575	.550	.525	.425	.583
Phi-2_128_128	.625	.575	.550	.500	.375	.583
Phi-2_64_64	.525	.425	.550	.375	.300	.500
RoBERTa-WNGRD	.800	.775	.775	.750	.675	.784
DeBERTaV3-TS	.800	.775	.725	.750	.625	.767
DeBERTaV3-base	.725	.750	.675	.725	.625	.717
BERT-SE	.750	.725	.650	.700	.550	.708
RoBERTa-large	.700	.700	.725	.675	.550	.708
BERT	.675	.650	.650	.600	.475	.658
Binary classification problem						
DeBERTaV3-TS	.725	.650	.550	.650	.650	.642
RoBERTa-WNGRD	.575	.600	.500	.550	.550	.558
BERT-SE	.625	.550	.375	.525	.525	.517

Table 1.4: Απόδοση του μοντέλου για το Υπο-πρόβλημα A: Παζλ προτάσεων. Περισσότερα αποτελέσματα στον πίνακα 1.7.

Υπο-πρόβλημα A: Παζλ Προτάσεων Ο πίνακας 1.4 απεικονίζει τις ελάχιστες διακυμάνσεις μεταξύ όλων των μετρικών που βασίζονται σε περιπτώσεις. Αυτή η συνέπεια επεκτείνεται και στις σχετικές μετρικές που βασίζονται σε ομάδες για όλα τα μοντέλα, αναδεικνύοντας μια συστηματική συμπεριφορά προς την κατεύθυνση της ανίχνευσης διαφόρων μονοπατιών συλλογισμού. Αυτή η παρατήρηση ισχύει τόσο για τους ταξινομητές που βασίζονται στον κωδικοποιητή όσο και για τα ΜΓΜ που χρησιμοποιήθηκαν. Οι γρίφοι προτάσεων προσφέρουν εγγενώς πιο λεπτομερείς πληροφορίες, επιτρέποντας στα μοντέλα να ανιχνεύουν και να εντοπίζουν τα ίδια μοτίβα συλλογισμού πιο εύκολα, ανεξάρτητα από τις αλλαγές στο πλαίσιο, σε αντίθεση με τους γρίφους λέξεων, οι οποίοι

συνήθως διαθέτουν πιο σύντομες δηλώσεις πλαισίου, παρουσιάζοντας μεγαλύτερη πρόκληση για τα μοντέλα να διακρίνουν συνεπή μοτίβα συλλογισμού.

Αρχικά, καθίσταται προφανές ότι η προ-εκπαίδευση των κωδικοποιητών σε διάφορα σύνολα δεδομένων συλλογιστικής κοινής λογικής οδηγεί σε ουσιαστικές βελτιώσεις των επιδόσεων, καθώς επιτρέπει στο σύστημα να αντιλαμβάνεται χαρακτηριστικά γνωρίσματα που δεν σχετίζονται με τον τομέα, τα οποία αποδεικνύονται πλεονεκτικά για την επακόλουθη εργασία. Επιπλέον, διάφοροι προ-εκπαιδευμένοι κωδικοποιητές κοινής λογικής που έχουν ρυθμιστεί σε δεδομένα BrainTeaser υπερτερούν των Llama 2 και Phi-2.

Μια άλλη αξιοσημείωτη παρατήρηση από τον πίνακα 1.4 είναι ότι μόνο το Mistral-7b από τα MFM είναι σε θέση να ξεπεράσει τα μοντέλα των κωδικοποιητών, ενώ τόσο το Llama 2 όσο και το Phi-2 σημείωσαν σταθερά χαμηλότερες βαθμολογίες.

Σε αντίθεση με το Llama 2 και το Mistral-7b, το Phi-2 δεν έχει υποβληθεί σε μάθηση για να ακολουθεί συστηματικά οδηγίες [26], γεγονός που, σε συνδυασμό με τον περιορισμένο αριθμό παραδειγμάτων στο σύνολο δεδομένων Υπο-πρόβλημα A: Παζλ Προτάσεων, συμβάλλει στη χαμηλότερη απόδοσή του, ως αποτέλεσμα της αδυναμίας του Phi να συλλάβει την πολυπλοκότητα των δεδομένων BrainTeaser.

System	Original	Semantic	Context	Ori.+Sem.	Ori.+Sem.+Con.	Overall
Multi-class classification problem						
Human	.917	.917	.917	.917	.900	.917
ChatGPT	.561	.524	.518	.439	.292	.535
RoBERTa-L	.195	.195	.232	.146	.061	.207
Mistral-7b_16_64	.875	.906	.781	.813	.719	.854
Mistral-7b_128_128	.844	.844	.813	.719	.625	.833
Mistral-7b_8_16	.781	.938	.781	.719	.562	.833
Mixtral-8x7b_128_128	.625	.719	.625	.531	.375	.656
Llama 2-13b_64_64	.354	.344	.438	.125	.031	.379
Llama 2-7b_64_64	.375	.344	.375	.125	.031	.365
Llama 2-7b_64_128	.281	.188	.438	.031	.031	.302
Phi-2_64_64	.688	.625	.688	.562	.438	.667
Phi-2_64_128	.656	.656	.625	.594	.406	.646
Phi-2_16_64	.625	.500	.688	.438	.312	.604
DeBERTaV3-base	.750	.750	.562	.656	.438	.687
DeBERTaV3-TS	.812	.781	.406	.719	.281	.666
RoBERTa-WNGRD	.750	.656	.500	.625	.312	.635
BERT	.562	.594	.469	.562	.312	.542
BERT-SE	.562	.500	.406	.500	.281	.489
RoBERTa-large	.281	.344	.344	.156	.094	.323
Binary classification problem						
DeBERTaV3-TS	.250	.313	.313	.219	.219	.292
RoBERTa-WNGRD	.375	.313	.125	.281	.281	.271
BERT-SE	.000	.000	.000	.000	.000	.000

Table 1.5: Απόδοση του μοντέλου για το Υπο-πρόβλημα B: Παζλ λέξεων. Περισσότερα αποτελέσματα στον πίνακα 1.7.

Υπο-πρόβλημα B: Παζλ Λέξεων Στον Πίνακα 1.5, παρατηρούμε μια έντονη αντίθεση στις επιδόσεις των μοντέλων στην κατανόηση και τον εντοπισμό μονοπατιών συλλογισμού, όταν το περιεχόμενο αλλάζει. Υπάρχουν αξιοσημείωτες αποκλίσεις στην ακρίβεια μεταξύ του αρχικού και του σημασιολογικού πλαισίου σε σύγκριση με την ανακατασκευή του πλαισίου, ιδιαίτερα εμφανείς στην περίπτωση των μοντέλων κωδικοποιητών μικρότερου μεγέθους.

Όσον αφορά στους κωδικοποιητές, είναι προφανές ότι, ειδικά το vanilla RoBERTa-large στερείται

ισχυρής λογικής και δυσκολεύεται να χειριστεί συστηματικά την ασάφεια. Αντίθετα, το RoBERTa-large προ-εκπαιδευμένο στο WinoGrande παρουσιάζει ανταγωνιστικές επιδόσεις. Αυτή η αξιοσημείωτη βελτίωση (πάνω από 40%) που οφείλεται στην προ-εκπαίδευση στο WinoGrande υποδηλώνει ότι το συγκεκριμένο σύνολο δεδομένων εξοπλίζει αποτελεσματικά το μοντέλο με την ικανότητα να κατανοεί τις πολυπλοκότητες της συλλογιστικής που σχετίζονται με τα λεκτικά παζλ, καθιστώντας τα αποτελέσματά του ανταγωνιστικά συγκριτικά με το DeBERTaV3 σε αυτό το υπο-πρόβλημα, παρά την υψηλότερη επίδοση του vanilla DeBERTaV3 έναντι του RoBERTa-large στα βασικά συγκριτικά κριτήρια συλλογιστικής [29]. Εκτός από αυτό, η προ-εκπαίδευση σε άλλα σύνολα δεδομένων συλλογιστικής κοινής λογικής δεν βελτιώνει σημαντικά τη συνολική απόδοση των κωδικοποιητών. Συμπερασματικά, εκτός από το WinoGrande, τα υπόλοιπα σύνολα δεδομένων με επιπλέον προ-εκπαίδευση δεν κατέχουν στοιχεία συλλογιστικής που να πορσεγγίζουν τους γρίφους λέξεων του BrainTeaser.

Όσον αφορά τα MFM, το Mistral-7b ξεπέρασε σημαντικά όλα τα υπόλοιπα, ξεπερνώντας ακόμη και το 8 φορές μεγαλύτερο μοντέλο που εκπαιδεύτηκε με τις ίδιες υπερπαραμέτρους (*Mistral-8x7b*). Το Llama 2 παρουσίασε τα χειρότερα αποτελέσματα ανεξάρτητα από το μέγεθος (7/13 δισεκατομύρια) και τις υπερπαραμέτρους LoRA (r και a). Αντίθετα, το Phi-2 επέδειξε σχετικά καλύτερες επιδόσεις, ιδίως αν ληφθεί υπόψη ο μικρότερος αριθμός παραμέτρων του (2,7 δισεκατομύρια) σε σύγκριση με τα άλλα MFM. Ωστόσο, και τα δύο μοντέλα παρουσίασαν χειρότερες επιδόσεις σε σύγκριση με τους περισσότερους κωδικοποιητές μετά από fine-tuning. Οι παρατηρήσεις αυτές επιβεβαιώνουν έντονα ότι οι γρίφοι λέξεων διαθέτουν μια κατανομή που αποκλίνει από την αναλυτική κοινή λογική συλλογιστική που απαιτείται για τους γρίφους προτάσεων, επιφέροντας ένα μοναδικό σύνολο γνωστικών απαιτήσεων.

Το Mistral-7b εμφανίζει μια τάση όπου εξηγήσεις υψηλότερης ποιότητας παράγονται με υψηλότερες τιμές του lora rank r . Ωστόσο, το μοντέλο με τις καλύτερες επιδόσεις παρουσίασε μια διαμόρφωση με $r=16$ και $a=64$. Η μέθοδος QLoRA [31] εξηγεί γιατί το κορυφαίο μοντέλο μας έχει βαθμό 16 αντί για 128, σε αντίθεση με τις κοινές προσδοκίες. Αντλώντας από την ευρέως διαδεδομένη παρουσία δομών χαμηλού βαθμού, όπως τονίζεται από προηγούμενες μελέτες [45, 46, 23], αξιοποιούμε την εγγενή δομή χαμηλού βαθμού στο πρόβλημά μας, όπως τονίζεται στο [31]. Είναι καλά τεκμηριωμένο ότι πολλά προβλήματα, ιδίως με έντονα υπερπαραμετροποιημένα μοντέλα, παρουσιάζουν ιδιότητες χαμηλής τάξης μετά την εκπαίδευση [58].

Συνολικά, τα συστήματά μας επιδεικνύουν αξιοσημείωτα υψηλή συνολική ακρίβεια, η οποία είναι λιγότερο από 10% χαμηλότερη από την ανθρώπινη απόδοση και περισσότερο από 30% μεγαλύτερη από την ChatGPT. Αυτό υποδηλώνει την επάρκεια των μεθόδων μας στην κατανόηση και την ανίχνευση μοτίβων λογοπαίγνιων, αντιμετωπίζοντας με συνέπεια την ασάφεια ανεξάρτητα από τα συμφραζόμενα και τις σημασιολογικές παραλλαγές στα brain teasers. Κατά την εξέταση των σύντομων εξηγήσεων που παρέχονται με κάθε πρόβλεψη (Πίνακας 1.6), παρατηρούμε εμπειριστωμένες αιτιολογήσεις ακόμη και για λανθασμένες απαντήσεις. Τα λάθη συνήθως ακολουθούν συγκεκριμένα μοτίβα λεξιλογίου σε όλες τις αρχικές, σημασιολογικές και συμφραζόμενες ερωτήσεις πολλαπλής επιλογής (λεπτομέρειες στο 1.3.5).

Αξιολόγηση και πληροφορίες σχετικά με την ποιότητα του συνόλου δεδομένων

Κατά την εξέταση των εσφαλμένων προβλέψεών μας και στα δύο υπο-προβλήματα, μετά την δημοσίευση των ετικετών (σωστών απαντήσεων) από τον διοργανωτή του διαγωνισμού για το κρυφό σύνολο δεδομένων, καταλήξαμε σε διάφορα συμπεράσματα. Σε όλες τις τριπλέτες, που περιλαμβάνουν αρχικές, σημασιολογικές και ερωτήσεις ανακατασκευής πλαισίου, παρατηρούμε ένα σημαντικό βαθμό ασάφειας σε διάφορα μοτίβα. Αυτή η ασάφεια οδηγεί συχνά σε ασυνεπή επιλογή σωστών απαντήσεων, ακόμη και όταν απαντώνται από ανθρώπους. Αυτό υπογραμμίζει την ανάγκη για σαφέστερη διατύπωση των ερωτήσεων και μονοσήμαντη έκφραση για την ενίσχυση της ακρίβειας των προβλέψεων του μοντέλου. Ένα άλλο αξιοσημείωτο μοτίβο που εντοπίσαμε αφορά στον ποιοτικό έλεγχο της σημασιολογικής αναδόμησης σε ορισμένες ερωτήσεις. Σε αυτές τις περιπτώσεις, ορισμένες λέξεις δεν αντικαταστάθηκαν με ακριβή συνώνυμα, με αποτέλεσμα την αλλαγή του προβλήματος που παρουσίαζε η ερώτηση. Ενώ αυτό μπορεί να μην είναι εγγενώς προβληματικό, οι σωστές απαντήσεις του συνόλου δεδομένων παρέμειναν αμετάβλητες σε σύγκριση με την αρχική έκδοση της ερώτησης. Αυτή η ασυμφωνία υποδηλώνει ότι η αλλαγή στον ορισμό της ερώτησης πέρασε απαρατήρητη από τους διοργανωτές της εργασίας, οδηγώντας σε κάποιες λανθασμένες προβλέψεις από το μοντέλο μας, ενώ στην πραγματικότητα το σωστό πλαίσιο της παρεχόμενης δήλωσης πολλαπλών επιλογών είχε καταγραφεί από το σύστημά μας. Οι δύο παραπάνω παρατηρήσεις αναδεικνύουν την εγγενή δυσκολία στη δημιουργία σαφών και ακριβών γρίφων, καθώς και την πρόκληση που αντιμετωπίζουν τα μοντέλα στην κατανόησή τους. Στα παραπάνω σενάρια, το

<i>Example of correct prediction with explanation</i>	
Question	Choices
If a dog is tied to a piece of rope that is 6m long how can it reach a bone that is 7m away?	The other end is not tied to anything.
	The rope stretches proportionally, providing the extra length needed for the horse to reach the hay seven meters away.
	The dog dug a 1m tunnel underground to reach the bone that was 7m away while still tied to the 6m long rope.
	None of above.
Explanation: A dog cannot reach a bone that is 7m away if it is tied to a piece of rope that is 6m long. This is because the distance between the dog and the bone is greater than the length of the rope. Therefore, the dog will not be able to reach the bone unless the rope is untied.	
<i>Example of misprediction with explanation</i>	
Question	Choices
How can you drop a raw egg onto a concrete floor and not crack it?	You need to drop the egg gently.
	You need to drop the egg hard.
	No problem, because concrete floors are very hard to crack.
	None of above.
Predicted Answer: None of above.	
Explanation: There is no way to drop a raw egg onto a concrete floor and not crack it. Concrete is a very hard material and will easily break any object that falls on it. The only way to avoid this would be to use some sort of protective covering for the egg before dropping it.	

Table 1.6: Ένα παράδειγμα εξηγήσεων που παράγονται από το Mistral-7b, ενώ αντιμετωπίζεται το πρόβλημα σε μορφή πολλαπλών επιλογών. Οι εξηγήσεις, ανεξάρτητα από την εγκυρότητά τους, είναι λεπτομερείς και επιδεικνύουν υψηλό επίπεδο κατανόησης και συλλογισμού. Οι σωστές απαντήσεις επισημαίνονται με έντονη γραφή, ενώ οι λανθασμένες προβλέψεις παραπέμπουν στην προβλεπόμενη από το μοντέλο απάντηση.

μοντέλο μας με τις καλύτερες επιδόσεις είτε εντοπίζει την ύπαρξη αντίφασης στις ερωτήσεις και επιλέγει να επιλέξει «Κανένα από τα παραπάνω», όπως διευκρινίζεται στη σύντομη και επεξηγηματική αιτιολόγησή του, είτε παρέχει μια λανθασμένη απάντηση με βάση την απάντηση του συνόλου δεδομένων, αλλά αντικατοπτρίζει σωστά το πλαίσιο του προβλήματος, το οποίο μπορεί να έχει μεταβληθεί λόγω ακούσιας χρήσης συνωνύμων.

Ανάλυση των υπερπαραμέτρων LoRA

Ο πίνακας 1.7 απεικονίζει περαιτέρω ανάλυση των υπερπαραμέτρων LoRA για τα μοντέλα Mistral και Mixtral, τα οποία παρουσίασαν τα καλύτερα αποτελέσματα μεταξύ όλων των άλλων μοντέλων και στις δύο εργασίες. Λόγω υπολογιστικών περιορισμών, εκπαιδεύσαμε το μοντέλο Mixtral, το οποίο είναι οκτώ φορές μεγαλύτερο, μόνο για τις υπερπαραμέτρους με τις καλύτερες επιδόσεις του Mistral.

System	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Overall
Task A						
Mistral-7b_64_128	.850	.825	.775	.825	.700	.817
Mistral-7b_16_64	.800	.800	.850	.750	.725	.817
Mixtral-8x7b_128_128	.850	.825	.725	.800	.700	.800
Mistral-7b_128_64	.850	.800	.725	.775	.625	.792
Mistral-7b_64_32	.850	.775	.725	.750	.675	.783
Mistral-7b_8_16	.800	.800	.700	.750	.625	.767
Mistral-7b_128_32	.825	.775	.725	.750	.600	.775
Task B						
Mistral-7b_128_128	.844	.844	.813	.719	.625	.833
Mistral-7b_8_16	.781	.938	.781	.719	.562	.833
Mistral-7b_16_16	.812	.812	.875	.688	.625	.833
Mistral-7b_8_8	.875	.812	.812	.750	.688	.833
Mistral-7b_16_32	.875	.812	.781	.750	.594	.823
Mistral-7b_64_32	.844	.875	.719	.750	.562	.812
Mistral-7b_128_64	.844	.812	.781	.688	.531	.812
Mistral-7b_64_64	.719	.812	.625	.625	.406	.719
Mixtral-8x7b_128_128	.625	.719	.625	.531	.375	.656

Table 1.7: Η απόδοση των διαφόρων υπερπαραμέτρων LoRA για τις Mistral και Mixtral και στα δύο υπο-προβλήματα.

1.4 Προτεινόμενη Μέθοδος 2 - Μέθοδος Προτροπής RISCORE

Σκεφτείτε τους ακόλουθους δύο γρίφους: **R1**: "*A man shaves every day, yet keeps his beard long*" και **R2**: "*What has a beard but never needs to shave?*". Ενώ αυτοί οι γρίφοι είναι παρόμοιοι στη δομή και αναφέρονται στα ίδια αντικείμενα, διαφέρουν στη συλλογιστική. Ο **R1** (απάντηση: "*A barber*") χρησιμοποιεί το "μούσι" κυριολεκτικά, ενώ ο **R2** (απάντηση: "*A tree*") το χρησιμοποιεί μεταφορικά, αναφερόμενος στο "beard" των δέντρων όπως η βελανιδιά. Στην αγγλική η λέξη beard έχει διπλό νόημα, αρχικά σημαίνει μούσι για τους ανθρώπους αλλά αναφέρεται συνήθως και σε ένα είδος βρύου ή λειχήνας που κρέμεται από τα κλαδιά των δέντρων και μοιάζει με γενειάδα.

Όταν χρησιμοποιείτε αυτούς τους γρίφους ως πλαίσιο για την επίλυση ενός νέου γρίφου - π.χ., "*I plant seeds every day, yet don't have a single plot*" - η ερμηνεία του "*plant seeds*" θα μπορούσε να οδηγήσει σε διαφορετικές απαντήσεις ανάλογα με το σκεπτικό (μεταφορικά: "*A teacher*" ή κυριολεκτικά: "*A hydroponicist*" - ο υδροπόνος). Αυτό δείχνει πώς η συλλογιστική, είτε κυριολεκτική είτε μεταφορική, μπορεί να επηρεάσει την επίλυση γρίφων και να οδηγήσει σε διαφορετικές κατευθύνσεις σκέψης.

Ωστόσο, η χρήση μιας ανακατασκευασμένης με βάση τα συμφραζόμενα εκδοχής της **R1** - όπως η **R3**: "*Tom attends class every day but doesn't do any homework*" - θα παρείχε σαφέστερη συλλογιστική σε σχέση με την **R2**, λόγω της συνέπειάς της με την κυριολεκτική σημασία της **R1**. Αυτό καταδεικνύει τη σημασία της ευθυγράμμισης των διαδικασιών συλλογισμού στο πλαίσιο που παρέχεται για την επίλυση νέων γρίφων. Όταν οι γρίφοι και η συλλογιστική τους είναι ευθυγραμμισμένοι με το πλαίσιο της ερώτησης, το μοντέλο μπορεί να ανταποκριθεί με μεγαλύτερη ακρίβεια, ενώ όταν χρησιμοποιούνται μεταφορικές ή πιο αφηρημένες έννοιες, η διαδικασία μπορεί να γίνει πιο περίπλοκη.

Βασίζόμενοι σε αυτή την ιδέα, προτείνουμε τη μέθοδο RISCORE, η οποία ενισχύει τις ικανότητες επίλυσης των ΜΓΜ αναφορικά με τους γρίφους, χρησιμοποιώντας την τεχνική λίγων παραδειγμάτων με παραδείγματα που έχουν ανακατασκευαστεί με βάση το πλαίσιο. Όπως φαίνεται στο Σχήμα 1.4.1, κάθε παράδειγμα συμπληρώνεται με μια ανακατασκευασμένη εκδοχή που διατηρεί τη διαδικασία συλλογισμού αλλά αλλάζει το περιεχόμενο. Αυτό βοηθά το μοντέλο να ακολουθήσει μια πιο συνεκτική πορεία συλλογισμού για να λύσει αποτελεσματικά νέους γρίφους.

Η προσέγγισή μας βασίζεται στις υπάρχουσες τεχνικές λίγων παραδειγμάτων, αλλά το RISCORE επικεντρώνεται στην επαύξηση των δειγμάτων της τεχνικής λίγων παραδειγμάτων με παραδείγματα ανακατασκευασμένα με βάση

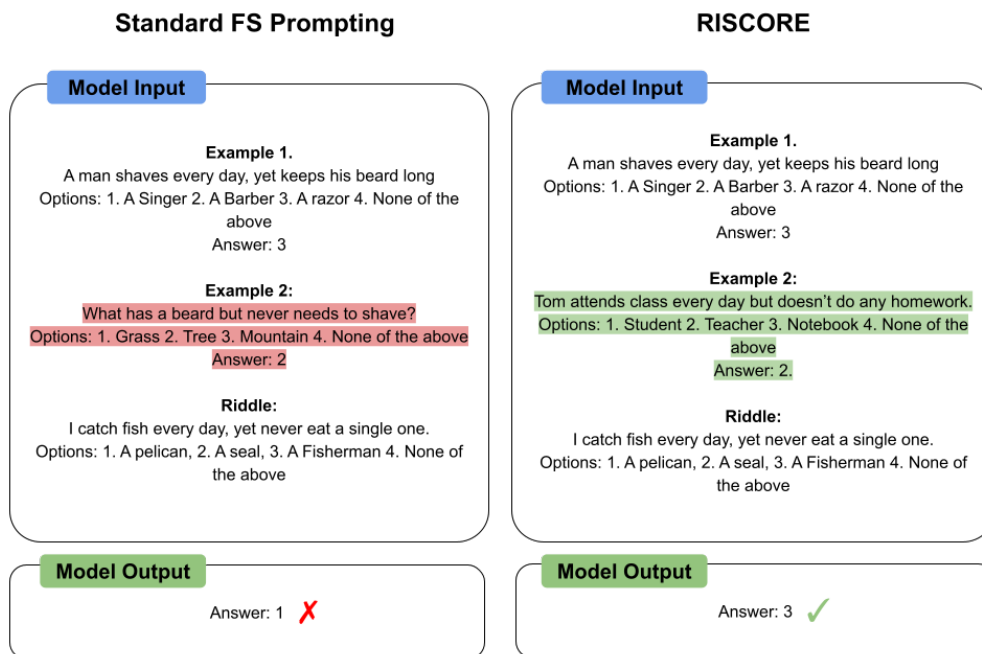


Figure 1.4.1: Ένα παράδειγμα της προτροπής λίγων παραδειγμάτων που αντιπαράβάλλεται με την ίδια μέθοδο αλλά χρησιμοποιώντας ένα παράδειγμα με ανακατασκευασμένο περιεχόμενο. Το κείμενο με πράσινο χρώμα αντιπροσωπεύει την ανακατασκευή του περιεχομένου του παραδείγματος 1, που παράγεται από την προτεινόμενη μέθοδο. Το παράδειγμα με κόκκινο χρώμα αφαιρέθηκε απλώς για να διατηρηθεί ο ίδιος αριθμός παραδειγμάτων.

το πλαίσιο, τα οποία συχνά αποδεικνύονται πιο ωφέλιμα από τα υπάρχοντα παραδείγματα στο σύνολο δεδομένων. Αυτή η πρόταση βρίσκεται επί του παρόντος υπό εξέταση σε ένα σημαντικό συνέδριο που αφορά το αντικείμενο μελέτης [60].

Διερευνήσαμε δύο κύρια ερευνητικά ερωτήματα:

- **EE1:** Βελτιώνει η χρήση ανακατασκευασμένων παραδειγμάτων την ικανότητα ενός μοντέλου να λύνει γρίφους δημιουργικής σκέψης σε σύγκριση με την τυπική προτροπή;
- **EE2:** Μπορεί η αυτόματη παραγωγή ανακατασκευασμένων αιτιγμάτων να βελτιώσει τις επιδόσεις τόσο σε εργασίες αντισυμβατικής όσο και σε εργασίες γραμμικής σκέψης;

1.4.1 Συνεισφορές

Συνοψίζοντας, οι συνεισφορές μας είναι:

- Επαληθεύουμε πειραματικά ότι η παροχή ενός αιτίγματος μαζί με την ανακατασκευασμένη με βάση το πλαίσιο εκδοχή του μπορεί να ενισχύσει την απόδοση τόσο σε προβλήματα γραμμικής όσο και δημιουργικής σκέψης.
- Προτείνουμε το RISCORE, μια νέα μέθοδο προτροπής που έχει σχεδιαστεί για να ενισχύει τις ικανότητες επίλυσης αιτιγμάτων των MFM. Συμπληρωματικά, παρουσιάζουμε έναν αλγόριθμο για τη δημιουργία ανακατασκευασμένων με βάση το πλαίσιο αιτιγμάτων σε μορφή πολλαπλής επιλογής.
- Συγκρίνουμε το RISCORE με ένα ευρύ φάσμα δημοφιλών τεχνικών προτροπής, αναδεικνύοντας την αποτελεσματικότητά του σε σύγκριση με διάφορες εναλλακτικές προτροπές, ενώ πειραματιζόμαστε με μοντέλα διαφορετικών μεγεθών.

1.5 Πειραματικό Μέρος - Μέθοδος Προτροπής RISCORE

Τα πειράματα αποσκοπούν στην αξιολόγηση της αποτελεσματικότητας του **RISCORE** στην ενίσχυση των ικανοτήτων συλλογισμού των γλωσσικών μοντέλων σε εργασίες γραμμικής και δημιουργικής σκέψης. Συγκεκριμένα, διερευνήσαμε κατά πόσον η χρήση ανακατασκευασμένων με βάση το περιεχόμενο αινιγμάτων ως δείγματα στην τεχνική προτροπής λίγων παραδειγμάτων παράλληλα με τα πρωτότυπά τους, βελτιώνει τις επιδόσεις των μοντέλων.

Για να διερευνήσουμε αυτά, δοκιμάσαμε διάφορα γλωσσικά μοντέλα, συγκρίνοντας το RISCORE με την παραδοσιακή προτροπή λίγων στιγμών, και επικεντρωθήκαμε σε μοτίβα συλλογισμού και όχι σε σημασιολογία επιφανειακού επιπέδου. Η εγγκατάστασή μας διαφοροποίησε τον αριθμό των υποδειγμάτων και την πολυπλοκότητα της εργασίας για να διασφαλίσουμε ισχυρές συγκρίσεις.

1.5.1 Σύνολα Δεδομένων και Μετρικές

Τα πειράματά μας χρησιμοποίησαν δύο σύνολα δεδομένων: **BrainTeaser** και **RiddleSense**, και τα δύο σχεδιασμένα για να δοκιμάσουν την δημιουργική και την γραμμική σκέψη και τις ικανότητες συλλογισμού των γλωσσικών μοντέλων.

- Το **BrainTeaser** αποτελείται από γρίφους με παραλλαγμένα υποσύνολα όπου οι ερωτήσεις τροποποιούνται διατηρώντας τις διαδρομές συλλογισμού τους. Οι τροποποιήσεις περιλαμβάνουν **σημασιολογική αναδόμηση** (αναδιατύπωση με διατήρηση των απαντήσεων και διάφορων αποσπασματικών στοιχείων ανέπαφων) και **αναδόμηση περιεχομένου** (αλλαγή του περιεχομένου αλλά διατήρηση της συλλογιστικής). Κάθε ερώτηση έχει τέσσερις επιλογές, με την τελευταία να είναι πάντα «Κανένα από τα παραπάνω». Επικεντρωθήκαμε στα πειράματά μας στο υπο-πρόβλημα **Sentence Puzzle (SP)**, το οποίο προκαλεί τις ικανότητες συλλογισμού του μοντέλου μέσω γλωσσικών παραλλαγών.
- Το **RiddleSense** παρουσιάζει ερωτήσεις τύπου γρίφου που δοκιμάζουν την κοινή λογική και την ερμηνεία της μεταφορικής γλώσσας. Ακολουθεί μορφή πολλαπλής επιλογής με μία σωστή απάντηση και τέσσερις λανθασμένες απαντήσεις, αλλά σε αντίθεση με το BrainTeaser, δεν περιλαμβάνει την επιλογή «Κανένα από τα παραπάνω», δημιουργώντας μια ελαφρώς διαφορετική πρόκληση συλλογισμού. Οι γρίφοι του συνόλου δεδομένων απαιτούν συχνά αφηρημένη σκέψη και κατανόηση μεταφορικής γλώσσας, ευθυγραμμίζόμενοι με το BrainTeaser, αλλά με πιο σύντομες απαντήσεις.

Στατιστικά στοιχεία συνόλου δεδομένων

Το σύνολο δεδομένων BrainTeaser χωρίζεται σε σύνολα εκπαίδευσης, ανάπτυξης (που χρησιμοποιούνται κατά την εξάσκηση) και κρυφά σύνολα δοκιμών. Αν και το RiddleSense ακολουθεί παρόμοια δομή, δεν είχαμε πρόσβαση στο κρυφό σύνολο δοκιμών του. Ως εκ τούτου, για τα πειράματά μας, χρησιμοποιήσαμε το σύνολο ανάπτυξης του RiddleSense για την αξιολόγηση του μοντέλου. Αυτό μας επέτρεψε να εξακολουθήσουμε να πραγματοποιούμε εύρωστες συγκρίσεις μεταξύ των μοντέλων και στα δύο σύνολα δεδομένων παρά τη διαφορά στη διαθεσιμότητα των δεδομένων.

Μετρικές

BrainTeaser Για το **BrainTeaser**, παρακολουθούμε τη συνολική ακρίβεια καθώς και την ακρίβεια για κάθε τύπο παραδείγματος - αρχική, σημασιολογική και ανακατασκευή περιεχομένου - δεδομένου ότι το σύνολο δεδομένων είναι ισορροπημένο. Παρακολουθούμε επίσης την **ακρίβεια βάσει ομάδας**, μετρώντας την απόδοση του μοντέλου όταν όλες οι ερωτήσεις σε μια ομάδα (αρχικές και ανακατασκευασμένες) απαντώνται σωστά. Αυτό παρέχει μια ευρύτερη άποψη της συλλογιστικής ικανότητας του μοντέλου σε διαφορετικές μορφές ερωτήσεων, ακολουθώντας τις κατευθυντήριες γραμμές αξιολόγησης από τους δημιουργούς του συνόλου δεδομένων.

RiddleSense Για το **RiddleSense**, μπορούμε να αξιολογήσουμε μόνο τη συνολική ακρίβεια λόγω της απλούστερης δομής των δεδομένων του, χωρίς τις πολλαπλές παραλλαγές ανακατασκευής του BrainTeaser.

Προεπεξεργασία και Φιλτράρισμα Συνόλου Δεδομένων

Για να αποφύγουμε την επικάλυψη δεδομένων, φιλτράραμε προσεκτικά παρόμοιες ερωτήσεις στο **RiddleSense** που ταίριαζαν με εκείνες στο **BrainTeaser**. Χρησιμοποιώντας **Sentence Transformers** [66] και το μοντέλο **gte-large-en-v1.5** [99], δημιουργήσαμε αναπαραστάσεις λέξεων και υπολογίσαμε την ομοιότητα συνημίτονου μεταξύ των ερωτήσεων και από τα δύο σύνολα δεδομένων. Ορίστηκε κατώφλι ομοιότητας 0,9, διασφαλίζοντας ότι μόνο οι πολύ παρόμοιες ερωτήσεις αφαιρούνται για να διατηρηθούν μοναδικά σύνολα δεδομένων.

Λόγω περιορισμών υλικού, το σύνολο δοκιμών RiddleSense χωρίστηκε σε δύο μισά για επεξεργασία. Μετά το φιλτράρισμα, μόνο μοναδικές, μη επικαλυπτόμενες ερωτήσεις διατηρήθηκαν για την αξιολόγηση, διατηρώντας την ακεραιότητα των αποτελεσμάτων και εξασφαλίζοντας ταυτόχρονα την αποτελεσματική χρήση των υπολογιστικών πόρων. Τα στατιστικά στοιχεία του συνόλου δεδομένων βρίσκονται στον πίνακα 1.8.

Dataset	Train	Dev	Test
BrainTeaser - SP	507 _(169x3)	120 _(40x3)	120 _(40x3)
RiddleSense (initial)		1021	
RiddleSense (filtered)	3510	720	—
RiddleSense (sampled 50%)		360	

Table 1.8: Στατιστικά στοιχεία συνόλου δεδομένων

1.5.2 Μέθοδος

Αυτή η ενότητα περιγράφει λεπτομερώς την προσέγγισή μας για τη δημιουργία υψηλής ποιότητας, ανακατασκευασμένων με βάση το περιεχόμενο ζευγών ερωτήσεων-απαντήσεων, τα οποία χρησιμοποιούνται παράλληλα με τα αρχικά τους για την ενίσχυση των επιδόσεων σε προβλήματα αντισυμβατικής σκέψης. Εμπνευσμένοι από την ημιαυτοματοποιημένη διαδικασία που παρουσιάστηκε από το BrainTeaser [38], την επεκτείνουμε με την πλήρη αυτοματοποίηση της διαδικασίας χρησιμοποιώντας Μεγάλα Γλωσσικά Μοντέλα (MGΜ).

1.5.3 Βήμα 1: Δημιουργία ζεύγους ερώτησης-απάντησης

Το πρώτο βήμα περιλαμβάνει τη δημιουργία ενός ζεύγους ερωτήσεων-απαντήσεων ανακατασκευασμένου με βάση το πλαίσιο ανά επιλεγμένη περίπτωση, αγνοώντας προσωρινά τις υπόλοιπες λανθασμένες επιλογές για την δημιουργία ενός προβλήματος σε μορφή πολλαπλής επιλογής. Για να το επιτύχουμε αυτό, παρέχουμε στο MGΜ τον γρίφο, τη σωστή απάντηση και μια προτροπή συστήματος. Η προτροπή δίνει εντολή στο μοντέλο να αναλύσει το ζεύγος ερώτησης-απάντησης, να κατανοήσει τη συλλογιστική που συνδέει το αίνιγμα με την απάντησή του και να δημιουργήσει ένα παρόμοιο αίνιγμα που διατηρεί την ίδια συλλογιστική διαδικασία σε ένα νέο πλαίσιο, μαζί με μια νέα απάντηση. Αυτή η προσέγγιση διατηρεί την κοινή λογική της αρχικής ερώτησης, ενώ προσαρμόζει τον γρίφο και την απάντηση σε μια νέα κατάσταση.

Εφαρμόζουμε αυτή τη διαδικασία τόσο στις ρυθμίσεις **Μηδενικών παραδειγμάτων** όσο και στις ρυθμίσεις **Λίγων παραδειγμάτων**:

- Στη ρύθμιση **μηδενικών παραδειγμάτων**, το μοντέλο παράγει τα ανακατασκευασμένα ζεύγη Ερώτησης-Απάντησης (QA) χωρίς να έχει προηγούμενα παραδείγματα ή ειδική εκπαίδευση για το συγκεκριμένο πρόβλημα [76].
- Στη ρύθμιση **λίγων παραδειγμάτων**, ενισχύουμε την κατανόηση του μοντέλου παρέχοντας παραδείγματα. Αυτά τα παραδείγματα περιλαμβάνουν τα αρχικά ζεύγη ερώτησης-απάντησης, καθώς και χειροκίνητα επιμελημένες, υψηλής ποιότητας, ανακατασκευασμένες εκδοχές με βάση το περιεχόμενο από το σύνολο δεδομένων BrainTeaser. Για να αποφευχθεί η επικάλυψη, τα δείγματα λίγων παραδειγμάτων επιλέγονται προσεκτικά από τα δεδομένα εκπαίδευσης και είναι διακριτά από τα ζεύγη που ανακατασκευάζονται.

Οι συγκεκριμένες προτροπές που χρησιμοποιήθηκαν για τη δημιουργία αυτών των ζευγαριών περιγράφονται λεπτομερώς στο 1.5.3.

Φιλτράρισμα και έλεγχος ποιότητας Αφού δημιουργήσουμε τα ανακατασκευασμένα, με βάση τα συμφραζόμενα, ζεύγη ερώτησης-απάντησης, εφαρμόζουμε μια διαδικασία φιλτραρίσματος για να διασφαλίσουμε ότι κάθε περίπτωση ευθυγραμμίζεται με τους γρίφους του συνόλου δεδομένων. Αυτό περιλαμβάνει την εφαρμογή συγκεκριμένων κανόνων για τη δομή και τη λογική του αινίγματος και της απάντησής του. Αυτοί οι κανόνες είναι προσαρμοσμένοι, εξασφαλίζοντας ευελιξία σε διάφορα σύνολα δεδομένων και περιβάλλοντα.

Διασφάλιση ποιότητας για παραγόμενο κείμενο Αφού δημιουργηθούν τα ανακατασκευασμένα από το πλαίσιο ζεύγη ερωτήσεων-απαντήσεων, μια διαδικασία ελέγχου ποιότητας διασφαλίζει ότι τα ζεύγη πληρούν συγκεκριμένα κριτήρια.

Για το σύνολο δεδομένων **BrainTeaser**, το φιλτράρισμα είναι ευέλικτο και αρκετά χαλαρό, δεδομένου ότι οι ερωτήσεις και οι απαντήσεις μπορεί να ποικίλλουν σε μήκος. Ο κύριος περιορισμός είναι ότι οι ερωτήσεις πρέπει να έχουν μήκος τουλάχιστον 7 λέξεις, ενώ οι απαντήσεις μπορούν να έχουν οποιοδήποτε μήκος. Ως αποτέλεσμα, μόνο το 2% περίπου των παραγόμενων ζευγών φιλτράρεται, διατηρώντας υψηλή συνολική ποιότητα.

Αντίθετα, το σύνολο δεδομένων **RiddleSense** έχει αυστηρότερους κανόνες λόγω της μικρότερης μορφής απαντήσεων. Εδώ, οι ερωτήσεις πρέπει να έχουν μήκος τουλάχιστον 6 λέξεις και οι απαντήσεις δεν πρέπει να υπερβαίνουν τις 7 λέξεις. Περίπου το 10% των παραγόμενων ζευγών φιλτράρονται, επειδή δεν πληρούν αυτά τα αυστηρότερα πρότυπα. Αυτά τα όρια λέξεων διασφαλίζουν ότι οι ερωτήσεις έχουν αρκετή πολυπλοκότητα και βάθος, ενώ οι απαντήσεις παραμένουν συνοπτικές και συνεκτικές, διατηρώντας τη δομή του γρίφου και εμπλέκοντας αποτελεσματικά το μοντέλο.

Μοντέλα που χρησιμοποιήθηκαν Πειραματιστήκαμε με δύο μοντέλα για να εκτιμήσουμε πώς το μέγεθος του μοντέλου επηρεάζει την ποιότητα των ανακατασκευών πλαισίου και τη συνολική απόδοση. Χρησιμοποιήσαμε το **Llama3-8B** [1], ένα μικρότερο μοντέλο, και το **Llama3-70B** [1], ένα μεγαλύτερο μοντέλο της ίδιας οικογένειας. Αυτά τα μοντέλα επιλέχθηκαν για τις ισχυρές επιδόσεις τους σε διάφορα προβλήματα. Συγκρίνοντας το μικρότερο Llama3-8B με το μεγαλύτερο Llama3-70B, επιδιώξαμε να διερευνήσουμε πώς το μέγεθος του μοντέλου επηρεάζει την αποτελεσματικότητα των ανακατασκευών περιεχομένου και την επίδρασή τους στην απόδοση και στα δύο σύνολα δεδομένων. Αυτή η σύγκριση παρέχει βαθύτερη κατανόηση του ρόλου του μεγέθους του μοντέλου στη δημιουργία παραδειγμάτων υψηλής ποιότητας.

Προτροπές

Πρώτον, θα παρουσιάσουμε τη διαμόρφωση των προτροπών **μηδενικών** παραδειγμάτων που χρησιμοποιείται για τη δημιουργία των ανακατασκευασμένων από το πλαίσιο ζευγαριών Ερώτησης-Απάντησης.

System Prompt:

You are an expert in context reconstruction. Your task is to receive a question along with its correct answer and adapt them to a new scenario while maintaining the misleading commonsense premise.

Please follow these steps:

- First, you will receive an unsolved riddle along with five answer options. Analyze the given setting and identify the connection between the question and its correct answer.
- Modify the original question and correct answer to fit a different situational context, ensuring that the underlying logic and relationship between them are preserved.
- Ensure that both the new question and the new correct answer are distinct from the originals.

User Prompt:

Question: ````{QUESTION}```

Correct answer: ````{ANSWER}```

Στη συνέχεια, με μικρές τροποποιήσεις, θα παρουσιάσουμε τη διαμόρφωση προτροπής **λίγων παραδειγμάτων** που ακολουθεί την ίδια λογική αλλά ενσωματώνει παραδείγματα για να καθοδηγεί τις απαντήσεις του μοντέλου.

System Prompt:

You are an expert in context reconstruction. Your task is to receive a question along with its correct answer and adapt them to a new scenario while maintaining the misleading commonsense premise.

Please follow these steps:

First, review an example provided with its context reconstruction, which illustrates the type of transformation you will need to perform.

Next, you will receive an unsolved riddle along with five answer options. Analyze the given setting and identify the connection between the question and its correct answer.

Modify the original question and correct answer to fit a different situational context, ensuring that the underlying logic and relationship between them are preserved.

Ensure that both the new question and the new correct answer are distinct from the originals.

User Prompt:

{EXAMPLES}

Adapt the following riddle - answer pair while taking into consideration the examples above regarding context reconstruction:

Question: ````{QUESTION}```

Correct answer: ````{ANSWER}```

Στην παραπάνω ρύθμιση, εννοείται ότι οι τιμές του **EXAMPLES** αναπαριστούν ζεύγη απαντημένων παραδειγμάτων Ερώτησης-Απάντησης: το αρχικό και το αντίστοιχο ανακατασκευασμένο με βάση το περιεχόμενο, τα οποία προέρχονται και τα δύο από το σύνολο δεδομένων BrainTeaser.

1.5.4 Βήμα 2: Δημιουργία των λανθασμένων επιλογών

Η δημιουργία των λανθασμένων επιλογών για τις ερωτήσεις πολλαπλής επιλογής είναι μια κρίσιμη αλλά δύσκολη διαδικασία. Οι λάθος επιλογές πρέπει να είναι αληθοφανείς αλλά σαφώς εσφαλμένες σε σύγκριση με τη σωστή απάντηση. Θα πρέπει επίσης να παραμένουν κοντά στο νόημα της σωστής απάντησης, αποφεύγοντας την υπερβολική απόκλιση που θα μπορούσε να μειώσει την πρόκληση του ανίγματος. Αυτό είναι ιδιαίτερα σημαντικό σε περιπτώσεις όπου η σωστή απάντηση είναι «Κανένα από τα παραπάνω», όπως φαίνεται στο σύνολο δεδομένων BrainTeaser. Το μήκος των απαντήσεων διαφέρει επίσης σημαντικά μεταξύ των συνόλων δεδομένων: Το BrainTeaser έχει συνήθως μεγαλύτερες απαντήσεις (περισσότερες από τέσσερις ή πέντε λέξεις), ενώ το RiddleSense διαθέτει μικρότερες, συχνά μονολεκτικές απαντήσεις. Για να αντιμετωπίσουμε αυτές τις διαφορές, προτείνουμε δύο μεθόδους για τη δημιουργία distractors, προσαρμοσμένες σε μεγάλες και μικρές απαντήσεις.

Δημιουργία λανθασμένων επιλογών μεγάλου μήκους Για τις μεγάλες σε μήκος λανθασμένες επιλογές, χρησιμοποιούμε μεγάλα γλωσσικά μοντέλα (MGm) για να δημιουργήσουμε τρεις λανθασμένες επιλογές που διατηρούν τη συνάφεια με το πλαίσιο και προκαλούν τη διαδικασία συλλογισμού. Εφαρμόζουμε δύο προσεγγίσεις:

1. **Πρώτη προσέγγιση:** Προτρέπουμε το μοντέλο με το αρχικό ζεύγος Ερώτηση-Απάντηση, ζητώντας του να κατανοήσει τη συλλογιστική πίσω από τον γρίφο και να δημιουργήσει μια εσφαλμένη απάντηση βασισμένη σε δύσκολες ή παραπλανητικές πτυχές της έννοιας. Με αυτόν τον τρόπο παράγεται μία από τις λανθασμένες επιλογές, που στοχεύει να μπερδέψει τον χρήστη μέσω της παρόμοιας λογικής.
-

2. **Δεύτερη προσέγγιση:** Παρέχουμε στο μοντέλο την ανακατασκευασμένη με βάση το περιεχόμενο ερώτηση και τις λανθασμένες απαντήσεις από την αρχική ερώτηση, εξαιρώντας το «Κανένα από τα παραπάνω». Το μοντέλο τροποποιεί αυτές τις λανθασμένες επιλογές με βάση το νέο πλαίσιο, διασφαλίζοντας ότι παραμένουν διαφορετικές από τη σωστή απάντηση. Παρόλο που αυτή η προσέγγιση μπορεί κάποιες φορές να παράγει λανθασμένες επιλογές χαμηλότερης ποιότητας, εξακολουθούν να εξυπηρετούν τον σκοπό της παραπλανητικής και ποικίλης απάντησης, συμβάλλοντας στην ενίσχυση της συλλογιστικής διαδικασίας.

Τα πειράματά μας δείχνουν ότι, όταν χρησιμοποιούνται τόσο τα πρωτότυπα όσο και τα ανακατασκευασμένα με βάση το πλαίσιο παραδείγματα, η συνολική απόδοση του μοντέλου δεν επηρεάζεται σημαντικά από λανθασμένες επιλογές χαμηλότερης ποιότητας. Δημιουργώντας δύο συναφείς με το πλαίσιο εσφαλμένες απαντήσεις και αναμειγνύοντάς τις με τη σωστή απάντηση, εξασφαλίζουμε ότι οι επιλογές πολλαπλών επιλογών παραμένουν συνεκτικές και προκλητικές για το μοντέλο. Τέλος, συνδυάζουμε τη σωστή απάντηση με δύο από τις τρεις παραγόμενες λανθασμένες απαντήσεις και προσθέτουμε την επιλογή «Κανένα από τα παραπάνω» ως τελευταία επιλογή. Εάν η επιλογή «Κανένα από τα παραπάνω» είναι η σωστή απάντηση, τότε χρησιμοποιούμε και τις τρεις παραγόμενες λανθασμένες επιλογές. Αυτή η διαδικασία οδηγεί στη δημιουργία ενός πλήρως προετοιμασμένου συνόλου δεδομένων, κατάλληλου για περαιτέρω χρήση και πειράματα.

Προτροπές Πρώτον, παρουσιάζουμε την προτροπή του συστήματος-χρήστη που χρησιμοποιείται για να αναθέσει στο μοντέλο την κατανόηση του γρίφου, τον εντοπισμό της συλλογιστικής διαδικασίας που συνδέει την ερώτηση με την απάντηση και, στη συνέχεια, να οδηγήσει σε παραγωγή μίας λανθασμένης επιλογής με βάση τις πιο δύσκολες ή παραπλανητικές πτυχές της έννοιας.

System Prompt:

Your task is to act as a concept grasper. You will be given a riddle and its correct answer.

Your goal is to understand the connection between the riddle and the correct answer, focusing on the tricky parts. Based on these tricky aspects, propose a plausible wrong answer that someone might give.

The wrong answer should be short, concise, and limited to one sentence.

- Riddle:

- Correct Answer:

Response format:

- Wrong Answer:

User Prompt:

- Riddle: {QUESTION}

- Correct Answer: {ANSWER}

Τώρα θα δώσουμε την προτροπή συστήματος-χρήστη που χρησιμοποιείται για τη δεύτερη μέθοδο.

System Prompt:

You will be given a sentence without context and then provided with a specific context.

Your task is to rewrite the sentence so that it aligns with the given context, while keeping it as close as possible to the original meaning.

The purpose is to adapt the sentence to the context, not to answer any questions related to the context.

- Sentence (out of context):

- Context:

Response format:

- Sentence:

User Prompt:

- Sentence (out of context): {ORI_CHOICE}

- Context:{QUESTION}

Εδώ, η τιμή του **ORI_CHOICE** αναφέρεται στις λανθασμένες επιλογές από τον αρχικό γρίφο, εξαιρουμένης της επιλογής «Κανένα από τα παραπάνω». Δημιουργούμε προτροπές για το μοντέλο με κάθε μία από αυτές τις εσφαλμένες επιλογές ξεχωριστά.

Δημιουργία λανθασμένων επιλογών μικρού μήκους Για σύνολα δεδομένων με μονολεκτικές απαντήσεις, όπως το **RiddleSense**, χρησιμοποιούμε μια διαφορετική προσέγγιση για τη δημιουργία εσφαλμένων επιλογών που συμπληρώνουν τις πιο λεπτομερείς, δομημένες ερωτήσεις. Χρησιμοποιούμε δύο διαφορετικές προσεγγίσεις προσαρμοσμένες σε αυτή τη μορφή σύντομων απαντήσεων.

Πρώτη προσέγγιση Αυτή η μέθοδος τεμαχίζει την ανακατασκευασμένη με βάση το πλαίσιο ερώτηση σε μικρότερες υπο-φράσεις χρησιμοποιώντας σημεία στίξης ή συνδετικές λέξεις. Για ερωτήσεις που δεν έχουν άμεση ερώτηση, προσθέτουμε **What am I?** - μια κοινή μορφή ερώτησης στο σύνολο δεδομένων- για να εξασφαλίσουμε την ευθυγράμμιση με την αναμενόμενη μορφή απάντησης.

Αφού το αίνιγμα χωριστεί σε υπο-φράσεις, ζητάμε από το μοντέλο να δημιουργήσει μια λανθασμένη απάντηση για κάθε υπο-φράση που συνδέεται με την ερώτηση. Ωστόσο, ορισμένες υπο-φράσεις μπορεί να εξακολουθούν να μοιάζουν υπερβολικά με τη σωστή απάντηση. Για να μετριάσουμε αυτό το πρόβλημα, εισάγουμε ένα ενδιάμεσο βήμα χρησιμοποιώντας ταξινόμηση μηδενικών παραδειγμάτων με το μοντέλο **facebook/bart-large-mnli** [43] από το Hugging Face. Αυτό το μοντέλο κατηγοριοποιεί τη σωστή απάντηση σε μία από τις οκτώ αμοιβαία αποκλειόμενες κατηγορίες "τρόφιμο", "άτομο", "αντικείμενο", "ζώο", "φύση", "χρόνος", "τόπος", "έννοια". Στη συνέχεια προτρέπουμε το ΜΓΜ να δημιουργήσει εσφαλμένες απαντήσεις χρησιμοποιώντας τις δύο πλησιέστερες κατηγορίες (εξαιρώντας την κατηγορία της σωστής απάντησης), διασφαλίζοντας ότι οι λανθασμένες επιλογές θα είναι συναφείς με το πλαίσιο. Εφαρμόζεται φιλτράρισμα για να εξασφαλιστούν διακριτές και κατάλληλες λανθασμένες απαντήσεις.

Δεύτερη προσέγγιση Όταν η πρώτη προσέγγιση δε δημιουργεί υψηλής ποιότητας λανθασμένες απαντήσεις, χρησιμοποιούμε το **WordNet** [11] για να επαυξήσουμε το σύνολο των διαθέσιμων λανθασμένων επιλογών. Για κάθε παραγόμενη ή αρχική λάθος επιλογή, αναχτούμε συνώνυμα και υπώνυμα από το WordNet, επεκτείνοντας τη δεξαμενή των πιθανών εσφαλμένων επιλογών για να διατηρήσουμε την πρόκληση και την ποικιλία στις επιλογές πολλαπλής επιλογής.

Επιλογή τελικών λανθασμένων επιλογών Αφού δημιουργήσουμε τις εσφαλμένες επιλογές, επιλέγουμε τυχαία τέσσερις και τις συνδυάζουμε με τη σωστή απάντηση, ανακατεύοντας τη σειρά για να ολοκληρώσουμε το σύνολο πολλαπλών επιλογών. Για να διατηρήσουμε υψηλής ποιότητας περισπασμούς, απαιτούμε τουλάχιστον δύο από τους τέσσερις περισπασμούς να έχουν παραχθεί με την πρώτη προσέγγιση (περισπασμοί που δημιουργούνται από το μοντέλο). Αυτό εξασφαλίζει υψηλότερο επίπεδο πολυπλοκότητας και συνάφειας, καθώς οι εσφαλμένες επιλογές που δημιουργούνται από το **WordNet** τείνουν να είναι λιγότερο ποιοτικές. Εάν αυτό το κριτήριο δεν πληρείται, παραλείπουμε τη συγκεκριμένη περίπτωση για την παραγωγή ενός παραδείγματος με ανακατασκευή περιεχομένου. Αυτή η στρατηγική συμβάλλει στη διατήρηση της συνολικής ποιότητας του συνόλου δεδομένων.

Προτροπές Για την πρώτη προσέγγιση, όπου παράγουμε περισπασμούς με βάση τις υποφράσεις, παρέχουμε στο μοντέλο δύο κατηγορίες (εξαιρουμένης της κατηγορίας της σωστής απάντησης) για κάθε υποφράση συνδεδεμένη με την ερώτηση. Η διαμόρφωση της προτροπής συστήματος-χρήστη δίνει εντολή στο μοντέλο να δημιουργήσει μια λανθασμένη απάντηση που ταιριάζει εντός της καθορισμένης κατηγορίας.

System Prompt:

Task: Provide a concise, relevant answer to the given question within the specified category.

Constraints:

- The answer should not exceed three words.
- Follow the exact format provided below.

Response Format:

Answer: ...

User Prompt:

Question: ````{QUESTION}```

Category: {CATEGORY}

1.5.5 Βήμα 3: Δημιουργία του αινίγματος

Αφού δημιουργήσαμε τις επιθυμητές ανακατασκευές περιεχομένου και για τα δύο σύνολα δεδομένων, αντιμετωπίσαμε ένα πρόβλημα: ορισμένα αρχικά επιλεγμένα παραδείγματα δεν είχαν αντίστοιχα ανακατασκευασμένα ζεύγη λόγω της διαδικασίας φιλτραρίσματος του ποιοτικού ελέγχου.

Για να το αντιμετωπίσουμε αυτό, χρησιμοποιούμε πρώτα τα αρχικά παραδείγματα με τη μεγαλύτερη σημασιολογική ομοιότητα ως παραδείγματα μάθησης, προσθέτοντας τα αυτόματα παραγόμενα ανακατασκευασμένα ζεύγη με βάση το περιεχόμενο. Εάν αυτά τα παραδείγματα δεν επαρκούν για να καλύψουν τις απαιτούμενες ρυθμίσεις (π.χ. δύο, τέσσερα ή οκτώ παραδείγματα για το **RISCORE**), χρησιμοποιούμε μια πιο δομημένη προσέγγιση αντί να προσθέτουμε παραδείγματα τυχαία.

Δημιουργούμε αναπαραστάσεις λέξεων για τα υπόλοιπα πρωτότυπα και ανακατασκευασμένα παραδείγματα που δεν έχουν ακόμη συμπεριληφθεί στα παραδείγματα. Χρησιμοποιώντας την ομοιότητα συνημίτονου, εντοπίζουμε τα πιο όμοια παραδείγματα από αυτή τη δεξαμενή, η οποία μπορεί να περιλαμβάνει ανακατασκευασμένα παραδείγματα που δεν αποτελούσαν μέρος του αρχικού συνόλου εκπαίδευσης. Στη συνέχεια, επιλέγουμε τα πιο παρόμοια παραδείγματα και αντιστοιχίζουμε το καθένα με το αντίστοιχό του, διασφαλίζοντας ότι συμπεριλαμβάνουμε είτε την αρχική είτε την ανακατασκευασμένη έκδοση, ανάλογα με τις ανάγκες. Αυτή η διαδικασία επαναλαμβάνεται μέχρι να επιτύχουμε τον απαιτούμενο αριθμό παραδειγμάτων για το πρόβλημα.

1.5.6 Περιγραφή Πειραμάτων

RISCORE

Το **RISCORE** εφαρμόζεται με την προσθήκη ανακατασκευασμένων ερωτήσεων με βάση το περιεχόμενο σε κάθε παράδειγμα που χρησιμοποιείται στην τεχνική λίγων παραδειγμάτων. Ο αριθμός των παραδειγμάτων αναφέρεται στο συνολικό αριθμό των παραδειγμάτων στην προτροπή, συνδυάζοντας τόσο τους αρχικούς γρίφους όσο και τις ανακατασκευασμένες εκδοχές τους. Για παράδειγμα, σε μια ρύθμιση **RISCORE** με 4 παραδείγματα, υπάρχουν 2 πρωτότυπα παραδείγματα και 2 ανακατασκευασμένες με βάση το πλαίσιο εκδοχές, εξισορροπώντας την είσοδο μεταξύ νέων και ανακατασκευασμένων αινιγμάτων.

Πραγματοποιήθηκαν πειράματα με ρυθμίσεις 2, 4 και 8 βολών, δηλαδή 1, 2 ή 4 πρωτότυπα παραδείγματα συνδυάστηκαν με τις αντίστοιχες ανακατασκευές τους. Χρησιμοποιήσαμε τα μοντέλα **Llama3-8B** και

Llama3-70B για να δημιουργήσουμε ζεύγη ερωτήσεων-απαντήσεων και εσφαλμένων επιλογών - περισπασμούς, χρησιμοποιώντας τόσο τεχνική μηδενικών όσο και λίγων παραδειγμάτων.

Τέλος, διακρίνουμε μεταξύ **RISCORE_m** (χρησιμοποιώντας χειροποίητα δημιουργημένες ανακατασκευές όταν είναι διαθέσιμες) και **RISCORE** (χρησιμοποιώντας πλήρως αυτοματοποιημένες ανακατασκευές). Η δομή της προτροπής ακολουθεί την ίδια μορφή με την τυπική μέθοδο **λίγων παραδειγμάτων**, με μόνη αλλαγή τα παραδείγματα που παρέχονται.

Μοντέλα που χρησιμοποιούνται στην μέθοδο RISCORE

Αυτή η ενότητα περιγράφει τα μοντέλα που χρησιμοποιήθηκαν στα πειράματά μας. Χρησιμοποιήθηκαν δύο διαφορετικά μοντέλα: το ένα για το πρόβλημά σημασιολογικής ομοιότητας (π.χ. φιλτράρισμα δεδομένων ή προτροπή) και το άλλο για τη δημιουργία σύντομων περισπασμών - εσφαλμένων επιλογών στο πλαίσιο της προτεινόμενης μεθόδου μας.

Μοντέλο αναπαράστασης λέξεων Για τις αναπαραστάσεις λέξεων, χρησιμοποιήσαμε το μοντέλο **gte-large-en-v1.5**¹¹ [99], το οποίο υλοποιείται μέσω της βιβλιοθήκης **Sentence Transformers**¹² [66]. Αυτό το μοντέλο επιλέχθηκε για την ισορροπία μεταξύ αποδοτικότητας και επιδόσεων, με καλές επιδόσεις σε προβλήματα του **MTEB leaderboard**¹³ και κυρίως στη σημασιολογική ομοιότητα κειμένου (STS). Το εύχρηστο μέγεθός του και οι ισχυρές επιδόσεις του το κατέστησαν κατάλληλη επιλογή για τον χειρισμό των αναγκών αναπαράστασης λέξεων στα πειράματά μας.

Μοντέλο ταξινόμησης μηδενικών παραδειγμάτων Για την ταξινόμηση μηδενικών παραδειγμάτων, χρησιμοποιήσαμε το μοντέλο **bart-large-mnli**¹⁴ [43] από το Hugging Face. Αυτό το μοντέλο χρησιμοποιήθηκε για την ταξινόμηση των σωστών απαντήσεων σε μία από οκτώ αμοιβαία αποκλειόμενες κατηγορίες: *τρόφιμα, πρόσωπο, αντικείμενο, ζώο, φύση, χρόνος, τόπος, έννοια*. Το επιλέξαμε για τις ισχυρές ικανότητές του στην ταξινόμηση μηδενικών πυροβολισμών, ιδιαίτερα για σύντομες εισόδους (π.χ. μεμονωμένες λέξεις ή φράσεις) χωρίς την ανάγκη για περαιτέρω εκπαίδευση ειδικά για το πρόβλημά μας. Η ικανότητά του να γενικεύει καλά σε διάφορες εργασίες το κατέστησε ιδανικό για την αξιόπιστη ταξινόμηση των απαντήσεων στα πειράματά μας.

Μοντέλα που χρησιμοποιήθηκαν για Προτροπή

Σε αυτή την ενότητα, αναλύουμε τα μοντέλα που χρησιμοποιήθηκαν για την προτροπή και τον αντίκτυπό τους στα τελικά αποτελέσματα για κάθε σύνολο δεδομένων. Επιλέξαμε παραλλαγές αυτών των μοντέλων που έχουν εκπαιδευτεί να ακολουθούν οδηγίες, επειδή ευθυγραμμίστηκαν καλύτερα με τις απαιτήσεις της εργασίας. Τα συντονισμένα με οδηγίες μοντέλα αποδίδουν γενικά καλά σε προβλήματα όπου η κατανόηση και η τήρηση συγκεκριμένων οδηγιών είναι ζωτικής σημασίας, καθιστώντας τα πιο κατάλληλα για τον χειρισμό της πολυπλοκότητας των εργασιών μας και την παραγωγή ορθών αποτελεσμάτων.

Οι τιμές της θερμοκρασίας και της ποινής επανάληψης επιλέχθηκαν μέσω διερευνητικών πειραμάτων. Για λόγους συνέπειας, εφαρμόσαμε **θερμοκρασία 0,5** και **ποινή επανάληψης** είτε 1,0 είτε 1,15 σε όλα τα πειράματα. Αυτές οι ρυθμίσεις εξασφάλισαν ισορροπημένες εξόδους, διατηρώντας παράλληλα την ποικιλία στο παραγόμενο κείμενο.

Llama 3 [1] Στα πειράματά μας, επιλέξαμε δύο παραλλαγές του μοντέλου: τις εκδόσεις 8B και 70B. Το μοντέλο Llama3-8B¹⁵ χρησιμοποιήθηκε για εξαγωγή συμπερασμάτων χωρίς χβάντιση, γεγονός που του επέτρεψε να παρέχει αποτελέσματα με πλήρη ακρίβεια. Αυτή η προσέγγιση ήταν ιδανική όταν οι υπολογιστικοί πόροι ήταν επαρκείς και η ακρίβεια ήταν κρίσιμη. Από την άλλη πλευρά, το μοντέλο Llama3-70B¹⁶, λόγω περιορισμών υλικού, υποβλήθηκε σε χβάντιση. Αυτή η διαδικασία μείωσε το μέγεθος του μοντέλου και τις υπολογιστικές ανάγκες του, καθιστώντας το προσιτό παρά τους περιορισμούς του υλικού μας. Αν και η χβάντιση μπορεί να

¹¹ [Alibaba-NLP/gte-large-en-v1.5](#)

¹² [SentenceTransformers](#)

¹³ [MTEB leaderboard](#)

¹⁴ [facebook/bart-large-mnli](#)

¹⁵ [meta-llama/Meta-Llama-3-8B-Instruct](#)

¹⁶ [meta-llama/Meta-Llama-3-70B-Instruct](#)

οδηγήσει σε κάποια απώλεια ακρίβειας, ήταν απαραίτητο βήμα για να χρησιμοποιήσουμε αυτό το μοντέλο, κάτι που διαφορετικά δεν θα ήταν εφικτό.

Mistral [32] Στα πειράματά μας, χρησιμοποιήσαμε το Mistral-7B-Instruct-v0.2¹⁷ και το Mixtral-8x7B-Instruct-v0.1¹⁸ στις μη κβαντισμένες μορφές τους. Το Mistral 7B-Instruct v0.2 επιλέχθηκε για τις ισχυρές ικανότητές του να ακολουθεί εντολές, διατηρώντας πλήρη ακρίβεια. Ομοίως, το Mixtral-8x7B-Instruct-v0.1, το οποίο ενσωματώνει οκτώ μοντέλα 7B, χρησιμοποιήθηκε χωρίς κβαντισμό για να επωφεληθούμε από τη συνδυασμένη απόδοσή των μοντέλων από τα οποία αποτελείται. Ο στόχος μας ήταν να διερευνήσουμε την αποτελεσματικότητα μικρότερων μοντέλων για τις εργασίες ανίγματος και προτιμήσαμε αυτές τις παραλλαγές, ώστε να διασφαλίσουμε ότι ανταποκρίνονταν καλά στις συγκεκριμένες απαιτήσεις, οδηγώντας σε βελτιωμένα αποτελέσματα.

Qwen2 [95] Για την περαιτέρω διερεύνηση μικρότερων παραλλαγών, επιλέξαμε την έκδοση Qwen2-7B-Instruct¹⁹. Η οικογένεια μοντέλων Qwen2, συμπεριλαμβανομένης αυτής της παραλλαγής 7B-Instruct, είναι γνωστή για τις ισχυρές επιδόσεις της σε προβλήματα συλλογισμού. Τα μοντέλα Qwen2 είναι ιδιαίτερα αποτελεσματικά στην επίλυση σύνθετων προβλημάτων και στη λογική συλλογιστική, γεγονός που τα καθιστά κατάλληλα για τους στόχους μας.

1.5.7 Αποτελέσματα

Σε αυτή την ενότητα, παρουσιάζουμε και αναλύουμε τα αποτελέσματα των πειραμάτων μας.

BrainTeaser

Στον πίνακα 1.9 παρουσιάζονται τα αποτελέσματα των επιδόσεων των διαφόρων τεχνικών προτροπής που εφαρμόζονται στο σύνολο δεδομένων BrainTeaser. Αξίζει να σημειωθεί ότι η μέθοδος CoT_FS υπολείπεται σε απόδοση σε σύγκριση με τις τεχνικές λίγων παραδειγμάτων, ακόμη και όταν παρέχονται παραδείγματα που περιλαμβάνουν χειροκίνητα παραγόμενες επεξηγήσεις. Η τάση αυτή ισχύει ανεξάρτητα από το μέγεθος του χρησιμοποιούμενου μοντέλου. Αντίθετα, υπάρχει σημαντική βελτίωση όταν η προσέγγιση των λίγων παραδειγμάτων ενσωματώνει τα παραγόμενα ανίγματα που έχουν ανακατασκευαστεί με βάση το πλαίσιο μαζί με τα αρχικά ανίγματα. Αυτό υποδηλώνει ότι τα ανίγματα που έχουν ανακατασκευαστεί με βάση το πλαίσιο καθοδηγούν αποτελεσματικά το μοντέλο στην ανίχνευση της κατάλληλης διαδρομής συλλογισμού. Είναι αξιοσημείωτο ότι το όφελος από τη χρήση αυτών των ανακατασκευασμένων παραδειγμάτων είναι πιο έντονο σε μικρότερα μοντέλα σε σύγκριση με μεγαλύτερα μοντέλα της ίδιας αρχιτεκτονικής. Για παράδειγμα, η βελτίωση στο μοντέλο Llama3-8B είναι σημαντική: ενώ η τεχνική των 4 δειγμάτων με χρήση σημασιολογικής ομοιότητας επιτυγχάνει βαθμολογία 0,717, η ενίσχυση αυτής της προσέγγισης με τέσσερα ανακατασκευασμένα με βάση το περιεχόμενο παραδείγματα (συνολικά οκτώ παραδείγματα) αυξάνει την απόδοση σε 0,758 - μια βελτίωση περίπου 5,7%. Για το μεγαλύτερο μοντέλο, η αύξηση αυτή είναι από 0,792 σε 0,808, οδηγώντας σε βελτίωση περίπου 2,02%. Το ίδιο μοτίβο είναι πιο εμφανές στο Mistral-7B, όταν για 4 παραδείγματα παρατηρούμε αύξηση 9,2% ενώ για το Mixtral-8x7B έχουμε αύξηση 2,5%. Παρόμοια τάση παρατηρείται και στο Qwen2-7B, όπου η ρύθμιση 4-shot με σημασιολογικά παρόμοια παραδείγματα έχει περίπου 3,4% χαμηλότερη απόδοση από τα επανημένα παραδείγματα 8-shot, τα οποία περιλαμβάνουν τα ίδια παραδείγματα μαζί με τις ανακατασκευές περιεχομένου τους, με αποτέλεσμα να έχουμε συνολικά 8 παραδείγματα.

Σε αυτό το σημείο, αξίζει να σημειωθεί ότι η βελτίωση των επιδόσεων δεν προέρχεται μόνο από τον αυξημένο αριθμό παραδειγμάτων, αλλά και από την ποιότητα των δειγμάτων που προστίθενται. Αυτό είναι εμφανές από το γεγονός ότι τα καλύτερα αποτελέσματα στο σύνολο δεδομένων επιτυγχάνονται όταν συμπεριλαμβάνουμε παραδείγματα με γρίφος ανακατασκευής με βάση το περιεχόμενό τους. Επιπλέον, συγκρίνοντας τις επιδόσεις και διατηρώντας σταθερό τον αριθμό των παραδειγμάτων, παρατηρούμε ότι οι περιπτώσεις που χρησιμοποιούν τα ανίγματα ανακατασκευής έχουν καλύτερες επιδόσεις, ειδικά όταν ο αριθμός των λήψεων είναι μεγαλύτερος από δύο. Στην τελευταία περίπτωση, η προσθήκη διαφορετικών παραδειγμάτων φαίνεται να βοηθάει περισσότερο το μοντέλο στη σωστή κατανόηση της εργασίας, καθώς η συμπερίληψη ενός μόνο παραδείγματος με την ανακατασκευή με βάση το πλαίσιο του καθιστά την προτροπή υπερβολικά συγκεκριμένη.

¹⁷[mistralai/Mistral-7B-Instruct-v0.2](#)

¹⁸[mistralai/Mixtral-8x7B-Instruct-v0.1](#)

¹⁹[Qwen/Qwen2-7B-Instruct](#)

Method	N.	Llama3-70B	Mistral-8x7B	Llama3-8B	Mistral-7B	Qwen2-7B
CoT_ZS	0	0.725	0.550	0.633	0.450	0.458
Randomly Selected Shots						
CoT_FS	2	0.758	0.617	0.633	0.475	0.608
	4	0.683	0.583	0.608	0.508	0.650
	8	0.708	0.642	0.658	0.508	0.667
FS	2	0.775	0.617	0.633	0.517	0.642
	4	0.808	0.683	0.642	0.483	0.608
	8	0.775	0.617	0.675	0.483	0.642
RISCORE _m	2	0.783	0.625	0.667	0.458	0.608
	4	0.758	0.617	0.675	0.517	0.625
	8	0.800	0.650	0.667	0.400	0.592
Semantically Similar Shots						
FS	2	0.825	0.692	0.700	0.517	0.600
	4	0.792	0.683	0.717	0.458	0.633
	8	0.783	0.667	0.767	0.533	0.650
RISCORE _m	2	0.783	0.675	0.767	0.483	0.667
	4	0.833	0.708	0.742	0.567	0.642
	8	0.808	0.708	0.758	0.550	0.667

Table 1.9: Απόδοση του μοντέλου για το *Brain Teaser* με χρήση τεχνικών ως αναφορά και προτροπών RISCORE_m. Τα καλύτερα συνολικά αποτελέσματα επισημαίνονται με **έντονη γραφή**. Πιο αναλυτικά αποτελέσματα μπορείτε να βρείτε στους πίνακες [1.13](#) και [1.14](#).

Method	N.	Llama3-70B	Mistral-8x7B	Llama3-8B	Mistral-7B	Qwen2-7B
Llama3-70B zeroshot for QA & Llama3-8B distractors						
RISCORE	2	0.792	0.667	0.625	0.492	0.625
	4	0.792	0.642	0.675	0.467	0.625
	8	0.808	0.683	0.700	0.475	0.642
Llama3-70B fewshot for QA & Llama3-8B distractors						
RISCORE	2	0.750	0.675	0.683	0.475	0.625
	4	0.792	0.650	0.658	0.558	0.658
	8	0.808	0.675	0.742	0.517	0.658
Llama3-70B fewshot for QA & Llama3-70B distractors						
RISCORE	2	0.783	0.667	0.683	0.500	0.617
	4	0.792	0.642	0.667	0.508	0.617
	8	0.767	0.683	0.700	0.500	0.617

Table 1.10: Επίδοση του μοντέλου για το *Brain Teaser* με τη χρήση της μεθόδου προτροπής RISCORE. Για την επιλογή όλων των δειγμάτων χρησιμοποιήθηκε επιλογή με βάση την σημασιολογική ομοιότητα. Τα αποτελέσματα που ξεπερνούν τη μέθοδο FS με σημασιολογικά παρόμοια παραδείγματα, χρησιμοποιώντας τον ίδιο αριθμό δειγμάτων, είναι **υπογραμμισμένα**. Αναλυτικότερα αποτελέσματα μπορείτε να βρείτε στους πίνακες [1.14](#) και [1.15](#).

Χρησιμοποιώντας την αυτοματοποιημένη μέθοδο RISCORE Στον πίνακα [1.10](#) παρουσιάζεται η απόδοση του RISCORE με τη χρήση της αυτοματοποιημένης μεθόδου για τη δημιουργία παραδειγμάτων με ανακατασκευή πλαισίου. Όπως αναμενόταν, η χρήση της αυτοματοποιημένης μεθόδου οδηγεί σε μικρότερη βελτίωση στην απόδοση του μοντέλου σε σύγκριση με τα χειροκίνητα επιμελημένα παραδείγματα που παρέχονται στο σύνολο δεδομένων BrainTeaser. Ωστόσο, η απόδοση όλων των μοντέλων βελτιώθηκε σταθερά με την προσθήκη αιγιμάτων ανακατασκευασμένων από το πλαίσιο, ακόμη και όταν χρησιμοποιήθηκε η πλήρως αυτοματοποιημένη διαδικασία για τη δημιουργία τους. Για παράδειγμα, για το μοντέλο Llama3-8B, η επίδοση με τη χρήση 4 δειγμάτων από το σύνολο δεδομένων είναι 0,717 και με τη χρήση των ίδιων παραδειγμάτων μαζί με τις ανακατασκευές τους που παράγονται από την αυτοματοποιημένη μέθοδο (8 λήψεις συνολικά), η επίδοση αυξάνεται σε 0,742. Η ίδια τάση είναι εμφανής σε όλα τα σχετικά μικρότερα μοντέλα, όπου η επαύξηση των 4 παραδειγμάτων που επιλέγονται με βάση τη σημασιολογική ομοιότητα με τα αυτοματοποιημένα ζεύγη ανακατασκευών με βάση το περιεχόμενό τους, αποδίδει αξιοσημείωτες βελτιώσεις στην απόδοση. Συγκεκριμένα, στην περίπτωση του Qwen2-7B, η τεχνική των 4 παραδειγμάτων που συμπληρώνεται με αντίστοιχα ανακατασκευασμένα παραδείγματα οδηγεί σε βελτίωση της απόδοσης κατά 2,5%. Στο Mistral-7B, η βελτίωση είναι ακόμη πιο έντονη, με την απόδοση να αυξάνεται κατά 10% από το μοντέλο αναφοράς στο καλύτερο σενάριο. Είναι σημαντικό ότι ο Πίνακας [1.10](#) δείχνει ότι στις περισσότερες περιπτώσεις (υπογραμμισμένες), είναι ακόμη καλύτερο να χρησιμοποιείται η αυτοματοποιημένη μέθοδος για τη δημιουργία πρόσθετων παραδειγμάτων σε σύγκριση με την προσθήκη του ίδιου αριθμού παραδειγμάτων από το σύνολο δεδομένων. Για παράδειγμα, η απόδοση του Llama3-70B με 8 δείγματα από το σύνολο δεδομένων είναι 0,783, ενώ η χρήση μόνο 4 παραδειγμάτων από το σύνολο δεδομένων και η αυτόματη δημιουργία των υπόλοιπων 4 με τη χρήση της μεθόδου ανακατασκευής περιεχομένου, οδηγεί σε απόδοση 0,808.

RiddleSense

Σε αυτό το σύνολο δεδομένων, το RISCORE μπορεί να εφαρμοστεί μόνο σε αυτόματα παραγόμενα παραδείγματα, επειδή η μορφή του συνόλου δεδομένων δε διαθέτει ανακατασκευές περιεχομένου για τις ερωτήσεις του.

Method	N.	Llama3-70B	Mistral-8x7B	Llama3-8B	Mistral-7B	Qwen2-7B
CoT_ZS	0	0.775	0.675	0.619	0.589	0.608
Randomly Selected Shots						
CoT_FS	2	0.789	0.692	0.625	0.594	0.667
	4	0.783	0.686	0.672	0.603	0.656
	8	0.783	0.697	0.658	0.597	0.625
FS	2	0.769	0.706	0.672	0.586	0.689
	4	0.772	0.719	0.639	0.586	0.683
	8	0.800	0.711	0.672	0.586	0.700
Semantically Similar Shots						
FS	2	0.792	0.714	0.706	0.608	0.722
	4	0.817	0.692	0.711	0.633	0.714
	8	0.800	0.675	0.681	0.611	0.731

Table 1.11: Απόδοση του μοντέλου για το *RiddleSense* χρησιμοποιώντας βασικές τεχνικές ως αναφορά. Το καλύτερο αποτέλεσμα συνολικά είναι επισημασμένο **με έντονη γραφή**. Πιο λεπτομερή αποτελέσματα μπορείτε να βρείτε στον πίνακα 1.16.

Method	N.	Llama3-70B	Mistral-8x7B	Llama3-8B	Mistral-7B	Qwen2-7B
Llama3-70B fewshot for QA & Llama3-70B distractors						
RISCORE	2	0.792	0.672	0.692	0.600	0.697
	4	0.783	0.689	0.722	0.600	0.717
	8	0.789	0.700	0.708	0.597	0.731
Llama3-70B fewshot for QA & Llama3-8B distractors						
RISCORE	2	0.786	0.719	0.681	0.603	0.681
	4	0.789	0.686	0.686	0.606	0.697
	8	0.775	0.689	0.706	0.617	0.719
Llama3-8B zeroshot for QA & Llama3-8B distractors						
RISCORE	2	0.792	0.681	0.689	0.589	0.694
	4	0.778	0.714	0.700	0.600	0.683
	8	0.806	0.689	0.686	0.614	0.689

Table 1.12: Απόδοση του μοντέλου για το *RiddleSense* με χρήση της μεθόδου προτροπής RISCORE. Η επιλογή με βάση την ομοιότητα χρησιμοποιήθηκε για την επιλογή όλων των δειγμάτων. Τα αποτελέσματα που ξεπερνούν τη μέθοδο λίγων παραδειγμάτων με σημασιολογικά παρόμοια παραδείγματα, χρησιμοποιώντας τον ίδιο αριθμό λήψεων, είναι **υπογραμμισμένα**. Αναλυτικότερα αποτελέσματα μπορείτε να βρείτε στον πίνακα 1.17.

Στον πίνακα 1.11 παρουσιάζονται τα αποτελέσματα των βασικών τεχνικών με τη χρήση διαφόρων μοντέλων στο σύνολο δεδομένων *RiddleSense*. Για άλλη μια φορά, τα αποτελέσματα επιβεβαιώνουν ότι η τεχνική λίγων βολών, χρησιμοποιώντας σημασιολογικά παρόμοια παραδείγματα, παρέχει σταθερά την καλύτερη απόδοση σε όλα τα μοντέλα που δοκιμάστηκαν.

Στον Πίνακα 1.12, παρουσιάζουμε τα αποτελέσματα του *RiddleSense* χρησιμοποιώντας την προτεινόμενη μέθοδο για την ανακατασκευή περιεχομένου κάθε εισόδου.

Μια σαφής τάση προκύπτει όταν συγκρίνουμε μια απλή επιλογή 8 δειγμάτων με βάση τη σημασιολογική ομοιότητα με τη ρύθμιση 8 δειγμάτων της μεθόδου μας, όπου τα 4 περισσότερα σημασιολογικά όμοια παραδείγματα εμπλουτίζονται με τα παραγόμενα από εμάς ζεύγη ανακατασκευασμένων δειγμάτων με βάση το περιεχόμενο. Ειδικότερα, τα αποτελέσματα δείχνουν ότι η μεθόδός μας ξεπερνά σταθερά την τυπική προσέγγιση των 8 παραδειγμάτων, επιδεικνύοντας σημαντική βελτίωση της απόδοσης του μοντέλου σε διάφορες περιπτώσεις. Ένα παράδειγμα αυτής της τάσης παρατηρείται με το μοντέλο Llama3-8B, όπου η μεθόδός μας σημειώνει βαθμολογία 0,708, περίπου 2% υψηλότερη από τη ρύθμιση λίγων δειγμάτων που βασίζεται στη σημασιολογική ομοιότητα, η οποία επιτυγχάνει βαθμολογία 0,681. Το ίδιο μοτίβο είναι εμφανές κατά τη σύγκριση των δύο μεθόδων με 4 παραδείγματα. Επιτυγχάνουμε παρόμοια ή οριακά καλύτερη ακρίβεια χρησιμοποιώντας συνολικά μόλις 4 παραδείγματα - δύο πρωτότυπα και δύο παραγόμενες ανακατασκευές με βάση το πλαίσιο. Αυτό υπογραμμίζει την αποτελεσματικότητα της ενσωμάτωσης ζευγών με ανακατασκευή περιεχομένου στην ενίσχυση της ακρίβειας του μοντέλου. Δεν επιτυγχάνουμε σημαντική αύξηση των επιδόσεων- ωστόσο, επιτυγχάνουμε παρόμοια ή οριακά καλύτερα αποτελέσματα, ενώ βασιζόμαστε σε λιγότερη θεμελιωμένη γνώση. Αυτό αποδεικνύει την αποτελεσματικότητα της μεθόδου μας, καθώς διατηρεί συγκρίσιμες επιδόσεις με λιγότερα, αλλά στρατηγικά επιλεγμένα, παραδείγματα.

Ποιότητα των ανακατασκευασμένων αινιγμάτων Για να δημιουργήσουμε αινίγματα ανακατασκευασμένα με βάση το περιεχόμενο, χρησιμοποιήσαμε μοντέλα Llama3 με 8 δισεκατομμύρια και 70 δισεκατομμύρια παραμέτρους τόσο σε ρυθμίσεις λίγων παραδειγμάτων (FS) όσο και σε ρυθμίσεις μηδενικών (ZS). Διαπιστώσαμε ότι το μοντέλο Llama3-8B δυσκολευόταν να παράγει υψηλής ποιότητας ζεύγη

ερωτήσεων-απαντήσεων για το σύνολο δεδομένων BrainTeaser και, ως εκ τούτου, δεν χρησιμοποιήθηκε στο RISCORE. Αυτή η δυσκολία πιθανώς προκύπτει από την απαίτηση του συνόλου δεδομένων BrainTeaser για δημιουργική σκέψη, η οποία είναι ιδιαίτερα δύσκολη για το μικρότερο μοντέλο. Τα ζεύγη Ερωτήσεων-Απαντήσεων είναι ουσιώδη, και εάν η ποιότητά τους είναι ανεπαρκής -όπως παρατηρήθηκε με το Llama3-8B στο σύνολο δεδομένων BrainTeaser- οι υψηλής ποιότητας λανθασμένες απαντήσεις δεν μπορούν από μόνες τους να αντισταθμίσουν αυτή την ανεπάρκεια. Ωστόσο, για τα αινίγματα γραμμικής σκέψης, το μικρότερο μοντέλο παράγει αποτελεσματικά αινίγματα ακόμη και στη ρύθμιση ZS, τα οποία, όταν χρησιμοποιούνται στη ρύθμιση λίγων δειγμάτων (FS), μπορούν να οδηγήσουν σε αυξημένες επιδόσεις σε σύγκριση με πραγματικά παραδείγματα που αντλούνται από το σύνολο δεδομένων.

Αναλυτικά αποτελέσματα

Σε αυτή την ενότητα, παρουσιάζουμε τα λεπτομερή αποτελέσματα και για τα δύο σύνολα δεδομένων σε όλες τις πειραματικές μας τεχνικές. Λόγω της εκτεταμένης φύσης των πειραμάτων, τα αποτελέσματα για κάθε σύνολο δεδομένων οργανώνονται σε υποπίνακες. Ξεκινάμε με το σύνολο δεδομένων BrainTeaser, όπου οι μετρικές είναι πιο λεπτομερείς λόγω της δομής του συνόλου δεδομένων, και ακολουθούν τα αποτελέσματα για το σύνολο δεδομένων RiddleSense. Η δομή είναι οργανωμένη ανά μέθοδο ανά μοντέλο, με τα αποτελέσματα να παρουσιάζονται σε φθίνουσα σειρά με βάση τη βαθμολογία για κάθε μέθοδο και όχι με βάση τον αριθμό των παραδειγμάτων. Αυτό διαφέρει από την προσέγγιση που περιγράφεται στο 1.5.7.

Model	Method	Num.Ex	Task	Temp	Rep_Pen	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Average
Chain-of-Thought Zero-shot											
Meta-Llama-3-70B-Instruct	CoT_ZS (Q)	0	SP	0.5	1.15	0.725	0.775	0.675	0.675	0.550	0.725
Mixtral-8x7B-Instruct-v0.1	CoT_ZS	0	SP	0.5	1.0	0.575	0.550	0.525	0.475	0.275	0.550
Meta-Llama-3-8B-Instruct	CoT_ZS	0	SP	0.5	1.0	0.625	0.650	0.625	0.500	0.325	0.633
Mistral-7B-Instruct-v0.2	CoT_ZS	0	SP	0.5	1.0	0.375	0.475	0.500	0.300	0.250	0.450
Qwen2-7B-Instruct	CoT_ZS	0	SP	0.5	1.15	0.475	0.450	0.450	0.350	0.200	0.458
Few-shot with CoT Explanations											
Meta-Llama-3-70B-Instruct	CoT_FS (Q)	2	SP	0.5	1.15	0.850	0.750	0.675	0.700	0.500	0.758
Meta-Llama-3-70B-Instruct	CoT_FS (Q)	8	SP	0.5	1.0	0.675	0.750	0.700	0.600	0.425	0.708
Meta-Llama-3-70B-Instruct	CoT_FS (Q)	4	SP	0.5	1.15	0.700	0.650	0.700	0.575	0.450	0.683
Mixtral-8x7B-v0.1	CoT_FS	8	SP	0.5	1.0	0.650	0.675	0.600	0.575	0.375	0.642
Mixtral-8x7B-v0.1	CoT_FS	2	SP	0.5	1.0	0.625	0.625	0.600	0.525	0.350	0.617
Mixtral-8x7B-v0.1	CoT_FS	4	SP	0.5	1.0	0.575	0.600	0.575	0.475	0.350	0.583
Meta-Llama-3-8B-Instruct	CoT_FS	8	SP	0.5	1.0	0.675	0.700	0.600	0.525	0.325	0.658
Meta-Llama-3-8B-Instruct	CoT_FS	2	SP	0.5	1.0	0.725	0.625	0.550	0.550	0.350	0.633
Meta-Llama-3-8B-Instruct	CoT_FS	4	SP	0.5	1.0	0.650	0.625	0.550	0.500	0.300	0.608
Mistral-7B-Instruct-v0.2	CoT_FS	8	SP	0.5	1.0	0.525	0.550	0.450	0.375	0.300	0.508
Mistral-7B-Instruct-v0.2	CoT_FS	4	SP	0.5	1.0	0.525	0.550	0.450	0.375	0.300	0.508
Mistral-7B-Instruct-v0.2	CoT_FS	2	SP	0.5	1.0	0.525	0.425	0.475	0.300	0.225	0.475
Qwen2-7B-Instruct	CoT_FS	8	SP	0.5	1.0	0.600	0.725	0.675	0.550	0.425	0.667
Qwen2-7B-Instruct	CoT_FS	4	SP	0.5	1.0	0.650	0.675	0.625	0.550	0.450	0.650
Qwen2-7B-Instruct	CoT_FS	2	SP	0.5	1.0	0.675	0.550	0.600	0.450	0.375	0.608
Few-shot with Random Selection											
Meta-Llama-3-70B-Instruct	FS Rand (Q)	4	SP	0.5	1.15	0.825	0.850	0.750	0.800	0.700	0.808
Meta-Llama-3-70B-Instruct	FS Rand (Q)	8	SP	0.5	1.15	0.775	0.800	0.750	0.675	0.550	0.775
Meta-Llama-3-70B-Instruct	FS Rand (Q)	2	SP	0.5	1.15	0.750	0.800	0.775	0.725	0.600	0.775
Mixtral-8x7B-Instruct-v0.1	FS Rand	4	SP	0.5	1.0	0.675	0.775	0.600	0.600	0.400	0.683
Mixtral-8x7B-Instruct-v0.1	FS Rand	2	SP	0.5	1.0	0.650	0.625	0.575	0.475	0.300	0.617
Mixtral-8x7B-Instruct-v0.1	FS Rand	8	SP	0.5	1.0	0.675	0.650	0.525	0.525	0.350	0.617
Meta-Llama-3-8B-Instruct	FS Rand	8	SP	0.5	1.0	0.750	0.625	0.650	0.575	0.400	0.675
Meta-Llama-3-8B-Instruct	FS Rand	4	SP	0.5	1.0	0.625	0.650	0.650	0.500	0.375	0.642
Meta-Llama-3-8B-Instruct	FS Rand	2	SP	0.5	1.0	0.600	0.700	0.600	0.550	0.375	0.633
Mistral-7B-Instruct-v0.2	FS Rand	2	SP	0.5	1.0	0.525	0.550	0.475	0.425	0.275	0.517
Mistral-7B-Instruct-v0.2	FS Rand	8	SP	0.5	1.0	0.450	0.575	0.425	0.400	0.300	0.483
Mistral-7B-Instruct-v0.2	FS Rand	4	SP	0.5	1.0	0.550	0.425	0.475	0.325	0.225	0.483
Qwen2-7B-Instruct	FS Rand	2	SP	0.5	1.15	0.675	0.650	0.600	0.550	0.425	0.642
Qwen2-7B-Instruct	FS Rand	8	SP	0.5	1.15	0.700	0.600	0.625	0.525	0.425	0.642
Qwen2-7B-Instruct	FS Rand	4	SP	0.5	1.15	0.675	0.575	0.575	0.525	0.400	0.608

Table 1.13: Απόδοση μοντέλου για το *BrainTeaser* (Μέρος 1). Η παρουσία του **(Q)** στη στήλη μέθοδος υποδεικνύει ότι τα αποτελέσματα αντιστοιχούν στην χβαντισμένη έκδοση του μοντέλου.

Model	Method	Num.Ex	Task	Temp	Rep_Pen	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Average
Few-shot with Semantic Similarity											
Meta-Llama-3-70B-Instruct	FS Sim (Q)	2	SP	0.5	1.15	0.850	0.875	0.750	0.800	0.675	0.825
Meta-Llama-3-70B-Instruct	FS Sim (Q)	4	SP	0.5	1.15	0.825	0.775	0.775	0.700	0.575	0.792
Meta-Llama-3-70B-Instruct	FS Sim (Q)	8	SP	0.5	1.15	0.775	0.825	0.750	0.700	0.600	0.783
Mixtral-8x7B-Instruct-v0.1	FS Sim	2	SP	0.5	1.0	0.700	0.700	0.675	0.575	0.450	0.692
Mixtral-8x7B-Instruct-v0.1	FS Sim	4	SP	0.5	1.0	0.750	0.650	0.650	0.575	0.425	0.683
Mixtral-8x7B-Instruct-v0.1	FS Sim	8	SP	0.5	1.0	0.750	0.625	0.625	0.525	0.375	0.667
Meta-Llama-3-8B-Instruct	FS Sim	8	SP	0.5	1.0	0.750	0.775	0.775	0.700	0.600	0.767
Meta-Llama-3-8B-Instruct	FS Sim	4	SP	0.5	1.0	0.675	0.825	0.650	0.675	0.475	0.717
Meta-Llama-3-8B-Instruct	FS Sim	2	SP	0.5	1.0	0.800	0.675	0.625	0.625	0.525	0.700
Mistral-7B-Instruct-v0.2	FS Sim	8	SP	0.5	1.0	0.550	0.600	0.450	0.500	0.325	0.533
Mistral-7B-Instruct-v0.2	FS Sim	2	SP	0.5	1.0	0.55	0.475	0.525	0.375	0.275	0.517
Mistral-7B-Instruct-v0.2	FS Sim	4	SP	0.5	1.0	0.475	0.450	0.450	0.350	0.225	0.458
Qwen2-7B-Instruct	FS Sim	8	SP	0.5	1.15	0.675	0.625	0.650	0.575	0.500	0.650
Qwen2-7B-Instruct	FS Sim	4	SP	0.5	1.15	0.650	0.600	0.650	0.550	0.450	0.633
Qwen2-7B-Instruct	FS Sim	2	SP	0.5	1.15	0.600	0.550	0.650	0.500	0.400	0.600
RISCORE _m Rand											
Meta-Llama-3-70B-Instruct	RISCORE _m Rand (Q)	8	SP	0.5	1.15	0.775	0.825	0.800	0.700	0.600	0.800
Meta-Llama-3-70B-Instruct	RISCORE _m Rand (Q)	2	SP	0.5	1.15	0.850	0.775	0.725	0.725	0.650	0.783
Meta-Llama-3-70B-Instruct	RISCORE _m Rand (Q)	4	SP	0.5	1.15	0.775	0.800	0.700	0.725	0.600	0.758
Mixtral-8x7B-Instruct-v0.1	RISCORE _m Rand	8	SP	0.5	1.0	0.700	0.625	0.625	0.600	0.450	0.650
Mixtral-8x7B-Instruct-v0.1	RISCORE _m Rand	2	SP	0.5	1.0	0.725	0.600	0.550	0.600	0.400	0.625
Mixtral-8x7B-Instruct-v0.1	RISCORE _m Rand	4	SP	0.5	1.0	0.675	0.650	0.525	0.550	0.325	0.617
Meta-Llama-3-8B-Instruct	RISCORE _m Rand	4	SP	0.5	1.0	0.650	0.700	0.675	0.625	0.500	0.675
Meta-Llama-3-8B-Instruct	RISCORE _m Rand	2	SP	0.5	1.0	0.700	0.725	0.575	0.525	0.375	0.667
Meta-Llama-3-8B-Instruct	RISCORE _m Rand	8	SP	0.5	1.0	0.700	0.750	0.550	0.575	0.425	0.667
Mistral-7B-Instruct-v0.2	RISCORE _m Rand	4	SP	0.5	1.0	0.575	0.475	0.500	0.425	0.300	0.517
Mistral-7B-Instruct-v0.2	RISCORE _m Rand	2	SP	0.5	1.0	0.550	0.425	0.400	0.325	0.200	0.458
Mistral-7B-Instruct-v0.2	RISCORE _m Rand	8	SP	0.5	1.0	0.350	0.450	0.400	0.275	0.200	0.400
Qwen2-7B-Instruct	RISCORE _m Rand	4	SP	0.5	1.15	0.625	0.650	0.600	0.550	0.400	0.625
Qwen2-7B-Instruct	RISCORE _m Rand	2	SP	0.5	1.15	0.650	0.575	0.600	0.500	0.400	0.608
Qwen2-7B-Instruct	RISCORE _m Rand	8	SP	0.5	1.15	0.625	0.550	0.600	0.450	0.325	0.592
RISCORE _m Sim											
Meta-Llama-3-70B-Instruct	RISCORE _m Sim (Q)	4	SP	0.5	1.15	0.850	0.850	0.800	0.800	0.650	0.833
Meta-Llama-3-70B-Instruct	RISCORE _m Sim (Q)	8	SP	0.5	1.15	0.800	0.775	0.850	0.700	0.625	0.808
Meta-Llama-3-70B-Instruct	RISCORE _m Sim (Q)	2	SP	0.5	1.15	0.850	0.775	0.725	0.725	0.600	0.783
Mixtral-8x7B-Instruct-v0.1	RISCORE _m Sim	8	SP	0.5	1.0	0.775	0.625	0.725	0.575	0.500	0.708
Mixtral-8x7B-Instruct-v0.1	RISCORE _m Sim	4	SP	0.5	1.0	0.675	0.750	0.700	0.600	0.425	0.708
Mixtral-8x7B-Instruct-v0.1	RISCORE _m Sim	2	SP	0.5	1.0	0.725	0.675	0.625	0.675	0.475	0.675
Meta-Llama-3-8B-Instruct	RISCORE _m Sim	2	SP	0.5	1.0	0.750	0.875	0.675	0.675	0.575	0.767
Meta-Llama-3-8B-Instruct	RISCORE _m Sim	8	SP	0.5	1.0	0.775	0.800	0.700	0.775	0.650	0.758
Meta-Llama-3-8B-Instruct	RISCORE _m Sim	4	SP	0.5	1.0	0.825	0.700	0.700	0.675	0.525	0.742
Mistral-7B-Instruct-v0.2	RISCORE _m Sim	4	SP	0.5	1.0	0.575	0.550	0.575	0.400	0.325	0.567
Mistral-7B-Instruct-v0.2	RISCORE _m Sim	8	SP	0.5	1.0	0.575	0.575	0.500	0.450	0.325	0.550
Mistral-7B-Instruct-v0.2	RISCORE _m Sim	2	SP	0.5	1.0	0.500	0.475	0.475	0.350	0.200	0.483
Qwen2-7B-Instruct	RISCORE _m Sim	8	SP	0.5	1.15	0.700	0.650	0.650	0.575	0.425	0.667
Qwen2-7B-Instruct	RISCORE _m Sim	2	SP	0.5	1.15	0.700	0.625	0.675	0.550	0.375	0.667
Qwen2-7B-Instruct	RISCORE _m Sim	4	SP	0.5	1.15	0.750	0.625	0.550	0.575	0.425	0.642

Table 1.14: Απόδοση μοντέλου για το *BrainTeaser* (Μέρος 2). Η παρουσία του (Q) στη στήλη μέθοδος υποδεικνύει ότι τα αποτελέσματα αντιστοιχούν στην χβαντισμένη έκδοση του μοντέλου.

Model	Method	Num.Ex	Task	Temp	Rep_Pen	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Average
RISCORE Results											
Llama3-70B zeroshot for QA & Llama3-8B for distractors											
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	8	SP	0.5	1.15	0.850	0.825	0.750	0.775	0.600	0.808
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	2	SP	0.5	1.15	0.800	0.775	0.800	0.750	0.650	0.792
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	4	SP	0.5	1.15	0.825	0.750	0.800	0.675	0.600	0.792
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	8	SP	0.5	1.0	0.700	0.750	0.600	0.650	0.475	0.683
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	2	SP	0.5	1.0	0.750	0.750	0.500	0.650	0.375	0.667
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	4	SP	0.5	1.0	0.725	0.625	0.575	0.575	0.375	0.642
Meta-Llama-3-8B-Instruct	RISCORE Sim	8	SP	0.5	1.0	0.700	0.725	0.675	0.650	0.525	0.700
Meta-Llama-3-8B-Instruct	RISCORE Sim	4	SP	0.5	1.0	0.650	0.675	0.700	0.550	0.425	0.675
Meta-Llama-3-8B-Instruct	RISCORE Sim	2	SP	0.5	1.0	0.700	0.600	0.575	0.525	0.350	0.625
Mistral-7B-Instruct-v0.2	RISCORE Sim	2	SP	0.5	1.0	0.550	0.450	0.475	0.375	0.300	0.492
Mistral-7B-Instruct-v0.2	RISCORE Sim	4	SP	0.5	1.0	0.525	0.400	0.475	0.300	0.200	0.467
Mistral-7B-Instruct-v0.2	RISCORE Sim	8	SP	0.5	1.0	0.425	0.525	0.475	0.400	0.325	0.475
Qwen2-7B-Instruct	RISCORE Sim	8	SP	0.5	1.15	0.600	0.700	0.625	0.550	0.425	0.642
Qwen2-7B-Instruct	RISCORE Sim	2	SP	0.5	1.15	0.650	0.650	0.575	0.550	0.400	0.625
Qwen2-7B-Instruct	RISCORE Sim	4	SP	0.5	1.15	0.600	0.650	0.625	0.550	0.450	0.625
Llama3-70B fewshot for QA & Llama3-8B for distractors											
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	8	SP	0.5	1.15	0.800	0.800	0.825	0.750	0.650	0.808
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	4	SP	0.5	1.15	0.775	0.800	0.800	0.700	0.600	0.792
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	2	SP	0.5	1.15	0.800	0.725	0.725	0.725	0.600	0.750
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	2	SP	0.5	1.0	0.725	0.725	0.575	0.675	0.475	0.675
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	8	SP	0.5	1.0	0.700	0.675	0.65	0.625	0.450	0.675
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	4	SP	0.5	1.0	0.700	0.675	0.575	0.625	0.400	0.650
Meta-Llama-3-8B-Instruct	RISCORE Sim	8	SP	0.5	1.0	0.775	0.750	0.700	0.700	0.600	0.742
Meta-Llama-3-8B-Instruct	RISCORE Sim	2	SP	0.5	1.0	0.725	0.700	0.625	0.600	0.400	0.683
Meta-Llama-3-8B-Instruct	RISCORE Sim	4	SP	0.5	1.0	0.725	0.625	0.625	0.550	0.450	0.658
Mistral-7B-Instruct-v0.2	RISCORE Sim	4	SP	0.5	1.0	0.625	0.575	0.475	0.475	0.350	0.558
Mistral-7B-Instruct-v0.2	RISCORE Sim	8	SP	0.5	1.0	0.500	0.550	0.500	0.425	0.350	0.517
Mistral-7B-Instruct-v0.2	RISCORE Sim	2	SP	0.5	1.0	0.550	0.425	0.450	0.375	0.225	0.475
Qwen2-7B-Instruct	RISCORE Sim	4	SP	0.5	1.15	0.625	0.700	0.650	0.575	0.450	0.658
Qwen2-7B-Instruct	RISCORE Sim	8	SP	0.5	1.15	0.650	0.675	0.650	0.575	0.475	0.658
Qwen2-7B-Instruct	RISCORE Sim	2	SP	0.5	1.15	0.625	0.650	0.600	0.525	0.400	0.625
Llama3-70B fewshot for QA & Llama3-70B for distractors											
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	4	SP	0.5	1.15	0.875	0.775	0.725	0.750	0.600	0.792
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	2	SP	0.5	1.15	0.775	0.825	0.750	0.775	0.675	0.783
Meta-Llama-3-70B-Instruct	RISCORE Sim (Q)	8	SP	0.5	1.15	0.775	0.750	0.775	0.725	0.600	0.767
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	8	SP	0.5	1.0	0.700	0.725	0.625	0.600	0.450	0.683
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	2	SP	0.5	1.0	0.700	0.700	0.600	0.625	0.500	0.667
Mixtral-8x7B-Instruct-v0.1	RISCORE Simm	4	SP	0.5	1.0	0.725	0.650	0.550	0.600	0.425	0.642
Meta-Llama-3-8B-Instruct	RISCORE Sim	8	SP	0.5	1.0	0.800	0.675	0.625	0.625	0.475	0.700
Meta-Llama-3-8B-Instruct	RISCORE Sim	2	SP	0.5	1.0	0.675	0.700	0.675	0.600	0.475	0.683
Meta-Llama-3-8B-Instruct	RISCORE Sim	4	SP	0.5	1.0	0.725	0.650	0.625	0.550	0.475	0.667
Mistral-7B-Instruct-v0.2	RISCORE Sim	4	SP	0.5	1.0	0.575	0.500	0.450	0.375	0.300	0.508
Mistral-7B-Instruct-v0.2	RISCORE Sim	2	SP	0.5	1.0	0.625	0.400	0.475	0.350	0.275	0.500
Mistral-7B-Instruct-v0.2	RISCORE Sim	8	SP	0.5	1.0	0.600	0.475	0.425	0.400	0.300	0.500
Qwen2-7B-Instruct	RISCORE Sim	2	SP	0.5	1.15	0.650	0.600	0.600	0.500	0.400	0.617
Qwen2-7B-Instruct	RISCORE Sim	4	SP	0.5	1.15	0.650	0.625	0.575	0.600	0.475	0.617
Qwen2-7B-Instruct	RISCORE Sim	8	SP	0.5	1.15	0.625	0.625	0.600	0.575	0.450	0.617

Table 1.15: Απόδοση μοντέλου για το *BrainTeaser* (Μέρος 3). Η παρουσία του (Q) στη στήλη μέθοδος υποδεικνύει ότι τα αποτελέσματα αντιστοιχούν στην κβαντισμένη έκδοση του μοντέλου.

Model	Method	Num.Ex	Quant	Temp	Rep_Pen	Average
Chain-of-Thought Zero-shot						
Meta-Llama-3-70B-Instruct	CoT_ZS (Q)	0	4bit	0.5	1.15	0.775
Mixtral-8x7B-v0.1	CoT_ZS	0	False	0.5	1.0	0.675
Meta-Llama-3-8B-Instruct	CoT_ZS	0	False	0.5	1.0	0.619
Mistral-7B-Instruct-v0.2	CoT_ZS	0	False	0.5	1.0	0.589
Qwen2-7B-Instruct	CoT_ZS	0	False	0.5	1.15	0.608
Few-shot with CoT Explanations						
Meta-Llama-3-70B-Instruct	CoT_FS (Q)	2	4bit	0.5	1.15	0.789
Meta-Llama-3-70B-Instruct	CoT_FS (Q)	4	4bit	0.5	1.15	0.783
Meta-Llama-3-70B-Instruct	CoT_FS (Q)	8	4bit	0.5	1.15	0.783
Mixtral-8x7B-v0.1	CoT_FS	8	False	0.5	1.0	0.697
Mixtral-8x7B-v0.1	CoT_FS	2	False	0.5	1.0	0.692
Mixtral-8x7B-v0.1	CoT_FS	4	False	0.5	1.0	0.686
Meta-Llama-3-8B-Instruct	CoT_FS	4	False	0.5	1.0	0.672
Meta-Llama-3-8B-Instruct	CoT_FS	8	False	0.5	1.0	0.658
Meta-Llama-3-8B-Instruct	CoT_FS	2	False	0.5	1.0	0.625
Mistral-7B-Instruct-v0.2	CoT_FS	4	False	0.5	1.0	0.603
Mistral-7B-Instruct-v0.2	CoT_FS	8	False	0.5	1.0	0.597
Mistral-7B-Instruct-v0.2	CoT_FS	2	False	0.5	1.0	0.594
Qwen2-7B-Instruct	CoT_FS	2	False	0.5	1.15	0.667
Qwen2-7B-Instruct	CoT_FS	4	False	0.5	1.15	0.656
Qwen2-7B-Instruct	CoT_FS	8	False	0.5	1.15	0.625
Few-shot with Random Selection						
Meta-Llama-3-70B-Instruct	FS Rand (Q)	4	4bit	0.5	1.15	0.800
Meta-Llama-3-70B-Instruct	FS Rand (Q)	8	4bit	0.5	1.15	0.772
Meta-Llama-3-70B-Instruct	FS Rand (Q)	2	4bit	0.5	1.15	0.769
Mixtral-8x7B-v0.1	FS Rand	4	False	0.5	1.0	0.719
Mixtral-8x7B-v0.1	FS Rand	8	False	0.5	1.0	0.711
Mixtral-8x7B-v0.1	FS Rand	2	False	0.5	1.0	0.706
Meta-Llama-3-8B-Instruct	FS Rand	2	False	0.5	1.0	0.672
Meta-Llama-3-8B-Instruct	FS Rand	8	False	0.5	1.0	0.672
Meta-Llama-3-8B-Instruct	FS Rand	4	False	0.5	1.0	0.639
Mistral-7B-Instruct-v0.2	FS Rand	2	False	0.5	1.0	0.586
Mistral-7B-Instruct-v0.2	FS Rand	4	False	0.5	1.0	0.586
Mistral-7B-Instruct-v0.2	FS Rand	8	False	0.5	1.0	0.586
Qwen2-7B-Instruct	FS Rand	8	False	0.5	1.15	0.700
Qwen2-7B-Instruct	FS Rand	2	False	0.5	1.15	0.689
Qwen2-7B-Instruct	FS Rand	4	False	0.5	1.15	0.683
Few-shot with Semantic Similarity						
Meta-Llama-3-70B-Instruct	FS Sim (Q)	4	4bit	0.5	1.15	0.817
Meta-Llama-3-70B-Instruct	FS Sim (Q)	8	4bit	0.5	1.15	0.800
Meta-Llama-3-70B-Instruct	FS Sim (Q)	2	4bit	0.5	1.15	0.792
Mixtral-8x7B-v0.1	FS Sim	2	False	0.5	1.0	0.714
Mixtral-8x7B-v0.1	FS Sim	4	False	0.5	1.0	0.692
Mixtral-8x7B-v0.1	FS Sim	8	False	0.5	1.0	0.675
Meta-Llama-3-8B-Instruct	FS Sim	4	False	0.5	1.0	0.711
Meta-Llama-3-8B-Instruct	FS Sim	2	False	0.5	1.0	0.706
Meta-Llama-3-8B-Instruct	FS Sim	8	False	0.5	1.0	0.681
Mistral-7B-Instruct-v0.2	FS Sim	4	False	0.5	1.0	0.633
Mistral-7B-Instruct-v0.2	FS Sim	8	False	0.5	1.0	0.611
Mistral-7B-Instruct-v0.2	FS Sim	2	False	0.5	1.0	0.608
Qwen2-7B-Instruct	FS Sim	8	False	0.5	1.15	0.731
Qwen2-7B-Instruct	FS Sim	2	False	0.5	1.15	0.722
Qwen2-7B-Instruct	FS Sim	4	False	0.5	1.15	0.714

Table 1.16: Απόδοση μοντέλου για το *RiddleSense* (Μέρος 1). Η στήλη **Quant** υποδεικνύει αν το μοντέλο είναι κβαντισμένο ή όχι.

Model	Method	Num.Ex	Quant	Temp	Rep_Pen	Average
RISCORE Results						
Llama3-70B Fewshot for QA & Llama3-70B for distractors						
Meta-Llama-3-70B-Instruct	RISCORE Sim	2	True	0.5	1.15	0.792
Meta-Llama-3-70B-Instruct	RISCORE Sim	8	True	0.5	1.15	0.789
Meta-Llama-3-70B-Instruct	RISCORE Sim	4	True	0.5	1.15	0.783
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	8	False	0.5	1.0	0.700
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	4	False	0.5	1.0	0.689
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	2	False	0.5	1.0	0.672
Meta-Llama-3-8B-Instruct	RISCORE Sim	4	False	0.5	1.0	0.722
Meta-Llama-3-8B-Instruct	RISCORE Sim	8	False	0.5	1.0	0.708
Meta-Llama-3-8B-Instruct	RISCORE Sim	2	False	0.5	1.0	0.692
Mistral-7B-Instruct-v0.2	RISCORE Sim	2	False	0.5	1.0	0.600
Mistral-7B-Instruct-v0.2	RISCORE Sim	4	False	0.5	1.0	0.600
Mistral-7B-Instruct-v0.2	RISCORE Sim	8	False	0.5	1.0	0.597
Qwen2-7B-Instruct	RISCORE Sim	8	False	0.5	1.0	0.731
Qwen2-7B-Instruct	RISCORE Sim	4	False	0.5	1.0	0.717
Qwen2-7B-Instruct	RISCORE Sim	2	False	0.5	1.0	0.697
Llama3-70B Fewshot for QA & Llama3-8B for distractors						
Meta-Llama-3-70B-Instruct	RISCORE Sim	4	True	0.5	1.15	0.789
Meta-Llama-3-70B-Instruct	RISCORE Sim	2	True	0.5	1.15	0.786
Meta-Llama-3-70B-Instruct	RISCORE Sim	8	True	0.5	1.15	0.775
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	2	False	0.5	1.0	0.719
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	8	False	0.5	1.0	0.689
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	4	False	0.5	1.0	0.686
Meta-Llama-3-8B-Instruct	RISCORE Sim	8	False	0.5	1.0	0.706
Meta-Llama-3-8B-Instruct	RISCORE Sim	4	False	0.5	1.0	0.686
Meta-Llama-3-8B-Instruct	RISCORE Sim	2	False	0.5	1.0	0.681
Mistral-7B-Instruct-v0.2	RISCORE Sim	8	False	0.5	1.0	0.617
Mistral-7B-Instruct-v0.2	RISCORE Sim	4	False	0.5	1.0	0.606
Mistral-7B-Instruct-v0.2	RISCORE Sim	2	False	0.5	1.0	0.603
Qwen2-7B-Instruct	RISCORE Sim	8	False	0.5	1.0	0.719
Qwen2-7B-Instruct	RISCORE Sim	4	False	0.5	1.0	0.697
Qwen2-7B-Instruct	RISCORE Sim	2	False	0.5	1.0	0.681
Llama3-8B Fewshot for QA & Llama3-8B for distractors						
Meta-Llama-3-70B-Instruct	RISCORE Sim	8	True	0.5	1.15	0.806
Meta-Llama-3-70B-Instruct	RISCORE Sim	2	True	0.5	1.15	0.792
Meta-Llama-3-70B-Instruct	RISCORE Sim	4	True	0.5	1.15	0.778
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	4	False	0.5	1.0	0.714
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	8	False	0.5	1.0	0.689
Mixtral-8x7B-Instruct-v0.1	RISCORE Sim	2	False	0.5	1.0	0.681
Meta-Llama-3-8B-Instruct	RISCORE Sim	4	False	0.5	1.0	0.700
Meta-Llama-3-8B-Instruct	RISCORE Sim	2	False	0.5	1.0	0.689
Meta-Llama-3-8B-Instruct	RISCORE Sim	8	False	0.5	1.0	0.686
Mistral-7B-Instruct-v0.2	RISCORE Sim	8	False	0.5	1.0	0.614
Mistral-7B-Instruct-v0.2	RISCORE Sim	4	False	0.5	1.0	0.600
Mistral-7B-Instruct-v0.2	RISCORE Sim	2	False	0.5	1.0	0.589
Qwen2-7B-Instruct	RISCORE Sim	2	False	0.5	1.0	0.694
Qwen2-7B-Instruct	RISCORE Sim	8	False	0.5	1.0	0.689
Qwen2-7B-Instruct	RISCORE Sim	4	False	0.5	1.0	0.683

Table 1.17: Απόδοση μοντέλου για το *RiddleSense* (Μέρος 2). Η στήλη **Quant** υποδεικνύει αν το μοντέλο είναι κβαντισμένο ή όχι.

1.6 Συμπεράσματα και Μελλοντικές επεκτάσεις

Συμπεράσματα

Σε αυτή τη μελέτη, αξιολογήσαμε συστηματικά τόσο προ-εκπαιδευμένους όσο και εκπαιδευμένους με την τεχνική fine-tune κωδικοποιητές, καθώς και μεγάλα γλωσσικά μοντέλα (LLM), σε μια ποικιλία προβλημάτων που περιλαμβάνουν σύνθετους συλλογισμούς. Τα ευρήματά μας καταδεικνύουν ότι η αξιοποίηση της προ-εκπαίδευσης σε συγκεκριμένο εννοιολογικό περιεχόμενο, σε συνδυασμό με την τεχνική fine-tune σε σχετικά σύνολα δεδομένων, οδηγεί σε σημαντικές βελτιώσεις σε σχέση με τις βασικές μεθόδους. Τα μοντέλα που ενσωμάτωσαν πρόσθετα σύνολα δεδομένων συλλογιστικής κατά την προ-εκπαίδευση παρουσίασαν βελτιωμένες επιδόσεις, υπογραμμίζοντας τη σημασία της κατανόησης γενικών συλλογιστικών ικανοτήτων που μπορούν να εφαρμοστούν σε συγκεκριμένα προβλήματα.

Σε όλα τα προβλήματα, το χάσμα επιδόσεων μεταξύ των μοντέλων καταδεικνύει την επίδραση της αρχιτεκτονικής, του μεγέθους και των στρατηγικών εκπαίδευσης. Τα προ-εκπαιδευμένα μοντέλα κωδικοποιητή, ιδίως εκείνα που εκτέθηκαν σε σύνολα δεδομένων κοινής λογικής, παρουσίασαν σημαντικές βελτιώσεις στο χειρισμό περίπλοκων μονοπατιών συλλογισμού. Αντίθετα, τα fine-tuned MFM συχνά υπολείπονται σε σχέση με τα προ-εκπαιδευμένα μοντέλα, τονίζοντας την ανάγκη για την χρήση της τεχνικής fine-tune ακόμη και σε μεγάλα μοντέλα για να προσαρμοστούν αποτελεσματικά στις προκλήσεις συγκεκριμένων προβλημάτων.

Μια βασική διαπίστωση από τα πειράματά μας είναι ότι η δόμηση σε πρόβλημα πολλαπλών επιλογών υπερτερεί σταθερά των προβλημάτων δυαδικής ταξινόμησης (πρόβλημα Σωστού-Λάθους). Αυτό υποδεικνύει ότι η παροχή ποικίλων επιλογών προσφέρει κρίσιμες πληροφορίες σχετικά με το περιεχόμενο που καθοδηγούν τα μοντέλα προς πιο ακριβή συλλογισμό. Η αρχική υπόθεση ότι το πρόβλημα Σωστού-Λάθους είναι απλούστερο αποδείχθηκε ανακριβής, καθώς συχνά δε διαθέτουν το βάθος του περιεχομένου που απαιτείται για τη σωστή λήψη αποφάσεων.

Επιπλέον, παρουσιάσαμε την τεχνική RISCORE, μια νέα μέθοδο που ενισχύει την τυπική μάθηση λίγων παραδειγμάτων με δείγματα ανακατασκευασμένα με βάση το περιεχόμενο. Η RISCORE αποδείχθηκε ιδιαίτερα αποτελεσματική στη βελτίωση των επιδόσεων των MFM βοηθώντας τα μοντέλα να εντοπίσουν τα κατάλληλα μονοπάτια συλλογισμού. Ειδικότερα, η μέθοδος RISCORE ξεπέρασε σταθερά τις τεχνικές που χρησιμοποιούνται ως αναφορά που βασίζονται στη σημασιολογική ομοιότητα, ιδίως σε μικρότερα μοντέλα όπου η γενίκευση της συλλογιστικής είναι πιο δύσκολη. Η ευρωστία της μεθόδου αποδείχθηκε τόσο με χρήση χειροκίνητα επιμελημένων όσο και αυτόματα παραγόμενων ζευγών ανακατασκευασμένων με βάση το πλαίσιο.

Εν κατακλείδι, η μελέτη μας υπογραμμίζει τη σημασία του συνδυασμού της προ-εκπαίδευσης σε διαφορετικά σύνολα δεδομένων συλλογιστικής με τη χρήση της τεχνικής fine-tuning σε συγκεκριμένα προβλήματα για τη βελτίωση της απόδοσης του μοντέλου σε σύνθετες εργασίες συλλογιστικής. Η εισαγωγή του RISCORE προσθέτει ένα ακόμη επίπεδο βελτίωσης, προσφέροντας μια κλιμακούμενη λύση για τη αναβάθμιση των συλλογιστικών ικανοτήτων των MFM, ιδίως σε τεχνικές λίγων δειγμάτων. Προχωρώντας, η περαιτέρω διερεύνηση της ανακατασκευής του περιεχομένου και της ανίχνευσης μοτίβων συλλογισμού θα είναι το κλειδί για την προώθηση των ικανοτήτων των MFM σε εργασίες που απαιτούν διαφοροποιημένη, πολυεπίπεδη επίλυση προβλημάτων.

Μελλοντικές επεκτάσεις

Με βάση την παρούσα μελέτη, θα πρέπει να διερευνηθούν διάφοροι βασικοί ερευνητικοί τομείς για την περαιτέρω ενίσχυση των δυνατοτήτων των MFM σε σύνθετες εργασίες συλλογισμού. Πρώτον, η **χρήση της τεχνικής fine-tune σε διαφορετικά σύνολα δεδομένων**, ιδίως για μικρότερα μοντέλα, θα μπορούσε να βελτιώσει τη γενίκευση. Ένα βασικό ερώτημα είναι: *Πώς μπορεί η τεχνική fine-tune σε συγκεκριμένο τομέα να ενισχύσει τη συλλογιστική σε μικρότερα FM;* Δεύτερον, η **σειρά των παραδειγμάτων RISCORE** παρουσιάζει μια άλλη ερευνητική ευκαιρία: *Επηρεάζει η αλληλουχία των παραγόμενων και των αρχικών παραδειγμάτων την ακρίβεια της συλλογιστικής;* Επιπλέον, η διερεύνηση της **επιλογής σημασιολογικά παρόμοιων παραδειγμάτων** θα μπορούσε να αποκαλύψει πώς οι κακές επιλογές επηρεάζουν την απόδοση.

Μια άλλη πολλά υποσχόμενη κατεύθυνση είναι η ανάπτυξη ενός **ντετερμινιστικού πλαισίου** για την κατηγοριοποίηση των τεχνικών συλλογισμού σε αινίγματα. *Μπορεί αυτό να βελτιώσει την επιλογή παραδειγμάτων για εκμάθηση με λίγες λήψεις;* Τέλος, η **ανθρώπινη αξιολόγηση των αυτοματοποιημένων παραδειγμάτων που παράγουμε** θα μπορούσε να βοηθήσει στη βελτίωση της

ποιότητάς τους, ανακαλύπτοντας τα σφάλματα μας κατά την διαδικασία παραγωγής τους, με αποτέλεσμα την βελτίωση της μεθόδου μας.

Chapter 2

Introduction

As natural language processing (NLP) models become increasingly integral to decision-making processes, the need for explainability and interpretability has become paramount. These models are widely used in various applications, including sentiment analysis, topic classification, machine translation, and conversational agents. Their decisions often have significant impacts, influencing everything from product recommendations to loan approvals and healthcare diagnostics. However, the black-box nature of many sophisticated NLP models, particularly deep learning-based approaches, poses challenges for understanding how specific decisions are made. This opacity can undermine trust and limit the broader adoption of these technologies in critical and sensitive domains.

Given the high stakes involved, there is a pressing need for methods that can provide clear, actionable insights into the behavior of NLP models. One promising approach is the generation of counterfactual explanations. Counterfactual explanations are hypothetical scenarios that show how minimal changes to the input can lead to different model outcomes. By presenting these alternate scenarios, counterfactual explanations help users understand what features are most influential in the model’s decision-making process. This type of explanation is particularly valuable because it aligns with human reasoning: understanding "what-if" scenarios is a natural way for people to grasp causal relationships and make informed decisions.

In this work, we propose a graph-based counterfactual editor designed to generate semantically edited inputs, referred to as *counterfactual interventions*, which alter the model’s prediction. These interventions serve as counterfactual explanations, offering a clear and intuitive understanding of the model’s decision-making process. The primary objective of this research is to develop a method that can create these counterfactual interventions in a way that is contrastive, fluent, and minimal. "Contrastive" means that the changes should effectively alter the model’s prediction; "fluent" means that the edited inputs should remain coherent and grammatically correct; and "minimal" means that the changes should be as small as possible to achieve the desired effect.

We evaluate the effectiveness and efficiency of our framework and compare it against existing state-of-the-art counterfactual editors using two NLP tasks, namely binary sentiment classification and topic classification. Our experiments demonstrate that our framework generates edits that are not only contrastive, fluent, and minimal but also produced significantly faster than those from other methods. This speed is crucial for practical applications where timely insights are necessary.

This thesis introduces a novel approach to generating counterfactual explanations for NLP models, addressing a critical need for interpretability in AI. It draws inspiration from [49] and presents a GNN model capable of solving the Rectangular Linear Assignment Problem (RLAP). Using this model instead of the traditional graph assignment algorithms, such as the *Hungarian algorithm* our work achieve a strong improvement in runtime. It also provides a comprehensive evaluation of the editor across different tasks, demonstrating its versatility and effectiveness. By emphasizing minimal and fluent edits, it ensures that the generated explanations are both understandable and practically useful, while the whole process remains faster than others state of the art editors.

The outline of this thesis is as follows:

- We will firstly provide all the background needed in basic Machine Learning algorithms and concepts as well as bipartite graphs in order to be able to explain and justify the idea of Graph Neural Networks. After doing so, we will provide a thorough description of GNN variants relevant to this work.
- We will give a more detailed definition of Counterfactual Explanations and related work. In a similar fashion, we will formally explain the problem of rectangular linear assignment and provide the theoretical background for methods already used to tackle it, such as the Hungarian algorithm.
- Lastly, we will propose our GNN-based editor for counterfactual explanations and highlight the performance of different variants used for its components. We will compare these results with two state of the art counterfactual editors across two NLP tasks and showcase how our editor outperforms them while also being significantly faster. We will also discuss different trade-offs in our editor and how multiple components influence its performance.

Chapter 3

Background

Before delving into the experimental portion of this thesis, it is important to establish a comprehensive understanding of the core concepts and techniques that underpin the research. This background section introduces key topics in Artificial Intelligence (AI), with a particular focus on large language models (LLMs) and Transformer-based architectures. These models are at the heart of the experiments, and the knowledge presented here will provide the necessary foundation for understanding how they can be used to tackle complex reasoning tasks, such as those requiring lateral thinking in a multiple-choice format.

Transformer-based models and LLMs Transformer-based models, especially LLMs, have become the backbone of modern Natural Language Processing (NLP) due to their ability to model intricate patterns in language, allowing them to perform a wide range of tasks such as text generation, question answering, and reasoning over complex inputs. In this thesis, we focus on exploring whether these models can demonstrate *lateral thinking*—a form of creative problem-solving that involves non-linear reasoning and the ability to approach problems in unconventional ways. Understanding the mechanics behind these models and how they can be adapted to such tasks is critical to the later analysis.

Training of Large Language Models One of the primary techniques used in this research involves the *training* of these large models on specific tasks. Training refers to the process of adjusting the internal parameters of the model to improve its ability to perform task-specific operations. While large models like LLMs come pre-trained on vast datasets, additional task-specific training is often required to fine-tune their performance on more specialized tasks, such as solving complex multiple-choice questions. This section will cover the various training methodologies, including conventional approaches as well as more advanced techniques, before we introduce the specific approach used in this thesis.

Quantization Techniques: LoRA and QLoRA A significant challenge when working with large models is the computational resources required for training and tuning them. To address this, we will also introduce *quantization* techniques, particularly methods such as *LoRA* (Low-Rank Adaptation) and *QLoRA* (Quantized Low-Rank Adaptation) [31, 13]. These techniques allow for efficient training and fine-tuning of large models using limited hardware by reducing the computational footprint and memory usage. LoRA and QLoRA enable models to be adapted to specific tasks without the need for prohibitively expensive hardware, making the fine-tuning of LLMs more accessible. This section will explain these techniques in detail, highlighting how they make it feasible to experiment with large models on constrained resources.

Prompting Techniques In addition to traditional training, *prompting* techniques play a crucial role in this work. Prompting involves carefully crafting the input text to guide the model’s output, allowing it to perform specific tasks without extensive retraining. This is an efficient method for controlling model behaviour and has been widely adopted in the use of large language models for various applications. In this thesis, we will explore baseline prompting techniques and analyze how they can influence the model’s reasoning capabilities, especially in tasks requiring lateral thinking.

Conclusion Through this chapter, we aim to provide a detailed overview of LLMs, Transformer architectures, prompting techniques, training and quantization methods, ensuring that readers have the necessary background to fully comprehend the experimental work and the analysis that follows.

3.1 Large Language Models (LLMs)

3.1.1 Background

Language Models (LMs) are computational models that have the capability to understand and generate human language. More specifically, a language model is a probability distribution over word sequences, having the ability to predict the likelihood of these sequences or generate new text based on a given input.

The fundamental methodology for probabilistic language modeling since 1980's has been **n-grams models**. These models are grounded on the Markov chain rule, assuming that the probability of the coming word in a word sequence depends only on a fixed size window of previous words. Hence a bigram model considers one previous word, a trigram two, and in general a n-gram $n-1$ previous words.

For example a bigram language model models the probability of the sequence $w_1, w_2, \dots w_n$ as:

$$P(w_1, w_2, \dots w_n) = P(w_2, w_1) \cdot P(w_3|w_2) \cdot \dots \cdot P(w_n|w_{n-1}) \quad (3.1.1)$$

where the conditional probability $P(w_k|w_{k-1})$ can be estimated the proportion of occurrences of the word w_{k-1} followed by the word w_k in the corpora. The estimation of these probabilities constitutes the training of an n-gram model on text corpora in one or more languages. However, given that a language can be used to express an infinite variety of valid sentences, an n-gram model struggles to assign non-zero probabilities to word sequences that may never be encountered in the training corpora. To address this problem, a variety of smoothing techniques has been proposed over the years.

Subsequently, the advancements in the development of neural networks suggest the neural language models, which was first introduced with simple **feedforward neural language models** by [5]. A feedforward neural language model is a simple feedforward network that takes a sequence of previous words as input and gives a probability distribution over possible next words as output. Therefore, similarly to a n-gram LM, the feedforward neural LM is trained to predict the probability of a word considering the $n-1$ previous words.

A major difference between n-grams and feedforward neural language models is found in the way the two models represent the sequence of input words. An n-gram model assigns the word identity i to each word w_i included in the prior word sequence which should be represented. On the contrary, a feedforward neural language model mainly uses an embedding vector for each word. Thus, the latter generalizes better to unseen word sequences of the test set.

While feedforward language models introduced many of the foundational concepts of neural language modeling, modern neural language models have advanced significantly by utilizing more powerful architectures. In this thesis, we will focus on analyzing the state-of-the-art (SotA) architecture: Transformer networks, which have revolutionized the field with their superior ability to model complex dependencies in language.

3.1.2 Transformer

Large Language Models (LLMs) are advanced language models with vast parameter sizes and remarkable learning capabilities that are typically pretrained on large unstructured text corpora. All modern LLMs are now built on Transformer architecture [83], which eschews recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output. The *encoder* and *decoder* are the two major components of the Transformer architecture. These components are employed in sequence-to-sequence operations like machine translation, where the encoder processes the input sequence and the decoder generates the output sequence.

The Transformer architecture is summarised below:

1. **Input Representation:** The input sequence is first embedded into continuous vector representations. Positional embeddings are added to these embeddings to provide information about the position of each token in the sequence.
2. **Encoder:** The encoder consists of a stack of identical layers. Each layer has two sub-layers:

- *Multi-Head Self-Attention*: This sub-layer computes a weighted sum of embeddings, allowing each word to focus on different parts of the input sequence. Multiple attention heads run in parallel, capturing different relationships between words.
- *Position-wise Feed-Forward Neural Network*: After the attention mechanism, each token's representation is passed through a position-wise feed-forward neural network. This introduces non-linearity and further refines the token representations.

Residual connections[28], followed by layer-normalization[3] are employed around each of the sub-layer.

3. **Decoder**: The decoder also consists of a stack of identical layers, each containing three sub-layers:

- *Masked Multi-Head Self-Attention*: This sub-layer acts similar to the corresponding encoder's sub-layer, but with a mask applied to prevent attending to future positions during training.
- *Multi-Head Encoder-Decoder Attention*: This sub-layer focuses on the encoded input sequence, allowing the decoder to consider the relevant parts of the input during sequence generation.
- *Position-wise Feed-Forward Neural Network*: Similar to the encoder, this sub-layer follows the attention mechanisms.

As with the encoder, residual connections are used around each sub-layer, followed by layer-normalization.

4. **Output Generation**: The output of the final decoder layer is transformed into probability distributions over the output vocabulary using a linear transformation followed by a softmax activation. Throughout the training process, the model is fed with a word sequence as input to predict the subsequent word.

Transformer variations

Several architectural variations of the Transformer have been proposed since it was first introduced by [83]. The masking pattern used on the inputs, which acts as contextual information for the model to generate a prediction, is a key distinction between these systems.

Encoder-Decoder. As previously indicated and originally proposed, the Transformer consisted of two stacks (Fig. 3.1.1): the encoder and the decoder. The encoder processes the input sequence and generates context-rich representations, which are used by the decoder to generate the output sequence step by step. Notable pretrained language models using an encoder-decoder architecture include BART[44] and T5 [65].

Decoder-Only. While the encoder-decoder design serves as the foundational variation of the Transformer model, contemporary LLMs predominately employ a decoder-only architecture. These models have the capability to train as a conventional language model, wherein they learn to predict the next token in a given sequence. Decoder-only models lack the ability to process or represent the input sequence and output sequence separately. All tokens are treated equally during processing, and conditioning is only dependent on prior tokens due to the casual masking pattern, implying that the representation of any conditioning text is intrinsically weaker. However, this produces a simpler architecture that is well-suited to a standard autoregressive next-step-prediction pretraining objective. Notably, this architecture is the foundation of the GPT series of models [64, 9] as well as numerous other recent LLMs.

Encoder-Only. As an aside, there is an additional prevalent architectural variant that employs only a Transformer encoder layer stack. This model architecture serves as the foundation for the ubiquitous BERT [15] and its derivatives.

Transformers have revolutionized NLP by eliminating the need for convolutions or recurrence mechanisms while achieving state-of-the-art performance on various tasks. Their ability to model long-range dependencies and efficiently process sequential data using the attention mechanism has set a new standard for language models.

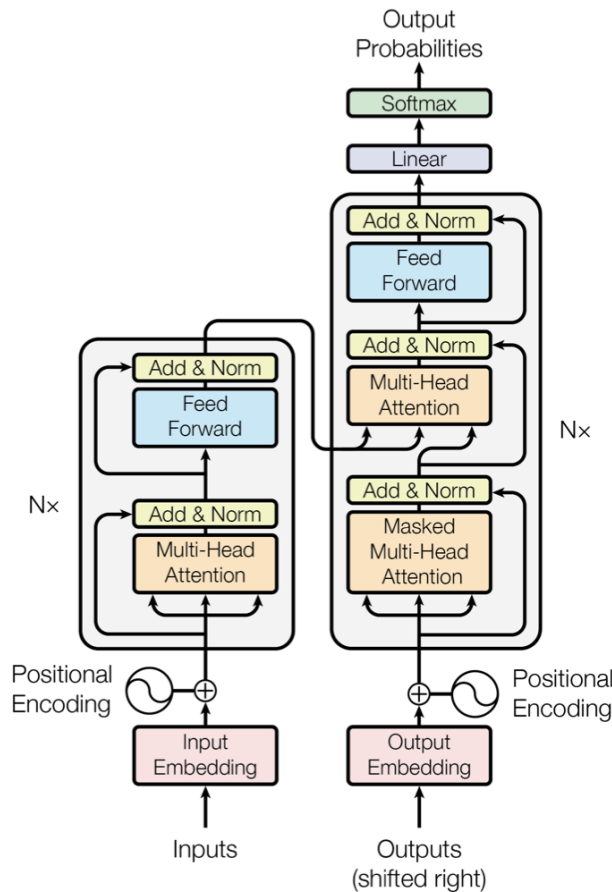


Figure 3.1.1: **The Transformer - model architecture.** The original Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, shown in the left and right halves of figure respectively[83]

3.1.3 Deeper Exploration of LLM Components

The LLMs used in this work are composed of multiple components that allow them to process, generate, and understand natural language. The following sections will delve deeper into the mathematical formulations of these key components. We will analyze each of these, including tokenization, embedding layers, self-attention mechanisms, multi-head attention, position encoding, feed-forward layers, and the final output layer.

Tokenization and Embedding Layer

Tokenization: Text is split into tokens, which can be words, subwords, or characters. Let $x = (x_1, x_2, \dots, x_n)$ represent an input sequence of n tokens.

Embedding: Each token x_i is transformed into a dense vector $e_i \in \mathbb{R}^d$ using an embedding matrix $E \in \mathbb{R}^{V \times d}$, where V is the vocabulary size and d is the embedding dimension. The embedding for the entire input sequence can be represented as:

$$E(x) = (e_1, e_2, \dots, e_n)$$

where $e_i = E[x_i]$.

Self-Attention Mechanism

The self-attention mechanism is central to transformer-based LLMs. For each token in the input sequence, the attention mechanism computes a weighted sum of the other tokens, determining how much focus the model should place on each token when processing the sequence.

For a given input $x = (x_1, x_2, \dots, x_n)$, the attention mechanism relies on three matrices: the query (Q), key (K), and value (V) matrices. These matrices are computed as:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V$$

where $X \in \mathbb{R}^{n \times d}$ is the input embedding matrix, and $W^Q, \in \mathbb{R}^{d \times d_q}$, $W^K, \in \mathbb{R}^{d \times d_k}$, $W^V, \in \mathbb{R}^{d \times d_v}$ are learned parameter matrices that project the input embeddings into the query, key, and value spaces, respectively.

The attention scores are computed by taking the dot product of the query with the keys, scaled by the square root of the dimensionality d_k , followed by a softmax operation:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

This equation computes a weighted sum of the value vectors, where the weights are determined by the similarity between the query and key vectors. The softmax function ensures that the attention weights sum to 1, giving a probabilistic interpretation to the weights. An visual representation of the self-attention mechanism can be seen in Figure 3.1.2.

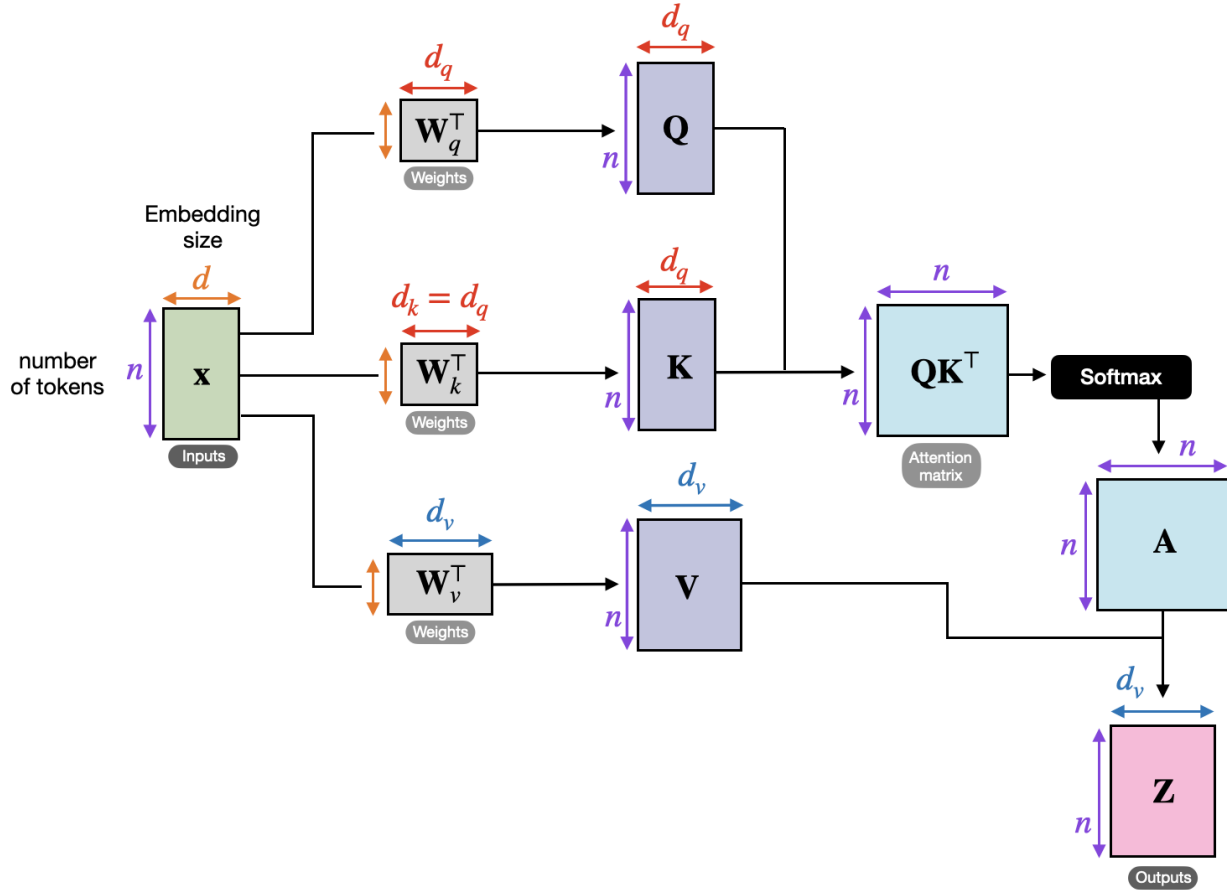


Figure 3.1.2: Self-Attention Mechanism

Multi-Head Attention

Instead of computing a single set of attention scores, LLMs use multi-head attention to capture different types of relationships between tokens. In multi-head attention, multiple attention mechanisms (heads) are run in parallel, and their outputs are concatenated:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W^O$$

where each head is computed as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

and $W^O \in \mathbb{R}^{hd_k \times d}$ is a learned output projection matrix. The use of multiple heads allows the model to attend to different parts of the input in different ways, increasing its representational power.

Position Encoding

Transformers do not have an inherent notion of word order, so position encoding is added to capture the sequential nature of the input. A positional encoding matrix $P \in \mathbb{R}^{n \times d}$ is added to the input embeddings. The positional encoding is typically defined using sinusoidal functions:

$$P_{i,2j} = \sin\left(\frac{i}{10000^{2j/d}}\right), \quad P_{i,2j+1} = \cos\left(\frac{i}{10000^{2j/d}}\right)$$

where i is the position and j is the dimension. This encoding allows the model to differentiate between positions in the sequence while maintaining generalization to longer or shorter sequences.

Feed-Forward Layer

Each transformer block includes a feed-forward network (FFN) applied independently to each position. The FFN consists of two linear transformations with a non-linearity in between:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

where $W_1 \in \mathbb{R}^{d \times d_{ff}}$ and $W_2 \in \mathbb{R}^{d_{ff} \times d}$ are learned weight matrices, $b_1 \in \mathbb{R}^{d_{ff}}$ and $b_2 \in \mathbb{R}^d$ are bias terms, and $\max(0, \cdot)$ represents the ReLU activation function. The FFN is applied separately to each token in the sequence.

Final Output Layer

The final output of the transformer model is passed through a linear transformation and softmax layer to generate probabilities for each class or token prediction. Let h_i represent the hidden state of the i -th token after the transformer layers. The probability distribution over the vocabulary is computed as:

$$P(x_i|h_i) = \text{softmax}(h_iW^T + b)$$

where $W \in \mathbb{R}^{V \times d}$ is the learned weight matrix, and $b \in \mathbb{R}^V$ is the bias term.

3.2 Training Techniques

Machine Learning (ML) techniques can be broadly categorized based on the type of data used during training and the level of supervision provided. These categories include supervised, unsupervised, and reinforcement learning, with hybrid approaches such as semi-supervised and self-supervised learning blending aspects of both supervised and unsupervised methods. To provide a comprehensive context for the techniques used in this thesis, it is essential to first explore these learning paradigms before diving deeper into the specific approach of supervised fine-tuning, which will be analyzed in detail in the following sections.

3.2.1 Overview

Supervised Learning Supervised learning is one of the most common ML techniques, where models are trained on labeled data. Each input feature vector x is paired with a known target y , and the model learns to map inputs to outputs by estimating the conditional probability distribution $p(y|x)$. Supervised learning is used for tasks such as classification, regression, and forecasting, where the relationships between inputs and outputs are explicitly defined. While highly effective in many domains, supervised learning requires large amounts of labeled data, which can be resource-intensive to obtain, particularly for complex tasks.

Unsupervised Learning In unsupervised learning, models are trained on data without labeled outputs. The goal is to learn the underlying structure of the data by modeling the distribution $p(x)$. This approach is commonly used for tasks like clustering, dimensionality reduction, and anomaly detection, where the model uncovers patterns and relationships within the data without explicit supervision. While unsupervised learning can provide valuable insights, it lacks the direct feedback present in supervised learning, making it harder to evaluate and interpret the results.

Semi-supervised Learning Semi-supervised learning bridges the gap between supervised and unsupervised learning by utilizing a small set of labeled data alongside a larger set of unlabeled data. This is particularly useful in cases where labeled data is scarce or expensive to collect, but unlabeled data is abundant. By leveraging the unlabeled data to help the model generalize better, semi-supervised learning aims to improve performance over a purely supervised approach. Applications of semi-supervised learning include tasks like link prediction in graphs and fraud detection, where having even a limited amount of labeled data can significantly enhance model performance [92].

Self-supervised Learning Self-supervised learning is an increasingly popular approach that lies between supervised and unsupervised learning. In this paradigm, the model is trained on unlabeled data by solving pretext tasks designed to learn useful representations. For example, models can be tasked with predicting missing parts of a sequence or reconstructing corrupted data. The learned representations can then be transferred to downstream tasks with minimal fine-tuning. Self-supervised learning has proven particularly effective in fields such as natural language processing and computer vision, where massive amounts of unlabeled data are available [53]. This technique forms the foundation of many large language models, which are pretrained using self-supervised methods before being fine-tuned for specific tasks.

Reinforcement Learning Reinforcement learning involves training an agent to interact with an environment and make decisions that maximize a reward signal. Unlike the static datasets used in supervised and unsupervised learning, the agent in reinforcement learning continuously learns from the outcomes of its actions. This method has been successfully applied in areas such as robotics, game playing, and decision-making systems. However, reinforcement learning is beyond the scope of this thesis and will not be explored further.

3.2.2 Pre-Training and Fine-Tuning

In machine learning, the training process typically follows a two-phase approach: pre-training and fine-tuning. These stages enable models to first develop a broad understanding of data and subsequently specialize in solving specific tasks. This approach has been particularly successful in natural language processing (NLP) and other domains where data variability and task specificity are critical for achieving high performance.

Pre-Training

Pre-training is the initial phase, where a model is exposed to large, unstructured datasets, often without explicit task labels. During this stage, the model learns general patterns, features, and representations from the data, which form a solid foundation for subsequent fine-tuning. For language models, pre-training often involves tasks such as predicting masked words in a sentence (masked language modeling) or predicting the next word in a sequence (autoregressive modeling). This phase equips the model with broad domain knowledge that can be transferred to more specific tasks during fine-tuning.

The key benefit of pre-training lies in its ability to help the model build generalizable representations that can be leveraged for various downstream tasks. Pre-trained models have demonstrated superior performance across a wide range of applications due to their capacity to capture essential patterns, such as semantic relationships and syntactic structures. This enables them to adapt quickly and efficiently to specific tasks, even when task-specific data is limited.

Pre-Training Benefits in this Context

In the context of this investigation, pre-training offers several key advantages:

- **Generalization:** Pre-training on both large general datasets and task-specific datasets enhances the model's ability to generalize to unseen examples, which is particularly important given the limited size of the fine-tuning dataset.
- **Efficiency:** By leveraging pre-trained knowledge, the model requires less task-specific data to perform effectively. Instead of learning from scratch, it uses the information acquired during pre-training to adapt efficiently during fine-tuning.
- **Faster Adaptation:** Pre-training on related tasks reduces the need for extensive training iterations during fine-tuning. The model can quickly converge to optimal performance, as it is already aligned with the task requirements.

Fine-Tuning

Fine-tuning builds on pre-training by adapting the pre-trained model to a specific task. In this phase, the model is trained on a smaller, task-specific dataset, typically with labeled examples, enabling it to specialize in solving the given problem. Fine-tuning is highly effective because it allows models to transfer the general knowledge acquired during pre-training to the specific nuances of a new task. By adjusting the model's parameters to fit task-specific data, fine-tuning enables the model to better capture patterns and relationships unique to the task.

Fine-tuning is particularly valuable in cases where large amounts of task-specific data are difficult to obtain. By leveraging the broad understanding gained during pre-training, fine-tuning enhances the model's performance on small datasets, reducing the risk of overfitting and improving generalization.

Fine-Tuning with Limited Data

A common challenge in machine learning, especially for specialized tasks, is the scarcity of labeled data. Fine-tuning on small datasets can be problematic when training a model from scratch, as the model is likely to overfit, failing to generalize beyond the training set. However, fine-tuning a pre-trained model helps mitigate this issue by providing a strong initialization based on prior knowledge.

Fine-tuning pre-trained models on small datasets offers several benefits:

- **Data Efficiency:** Since the model has already learned general patterns during pre-training, fine-tuning on limited data mainly involves adapting these patterns to the task-specific nuances. This reduces the need for extensive labeled data and helps prevent overfitting.
- **Improved Generalization:** The pre-trained model's existing knowledge allows it to generalize better even with smaller datasets, as it is less dependent on the fine-tuning data alone. This leads to better performance on unseen examples compared to models trained from scratch.

Thus, in scenarios with limited data, fine-tuning pre-trained models proves advantageous by providing robust generalization and minimizing overfitting risks.

Fine-Tuning with Task-Specific Pre-Training

Beyond general pre-training, models can benefit significantly from **domain-adaptive pre-training**, where pre-training data is carefully selected to closely align with the target task. By pre-training the model on datasets similar to the downstream task, the model becomes more adept at capturing domain-specific patterns and representations.

For instance, in NLP tasks, a model pre-trained on a general corpus (e.g., Wikipedia or Common Crawl) may perform reasonably well across a variety of tasks. However, fine-tuning this model on a specialized task (e.g., commonsense reasoning or disambiguation) can be further enhanced by an intermediate pre-training step using a dataset closely related to the final task before fine-tuning.

This approach offers several advantages:

- **Transfer of Domain-Specific Knowledge:** By pre-training on data that shares domain-specific characteristics with the target task, the model captures nuances and representations closely aligned with the downstream task. This leads to faster convergence and typically results in better performance during fine-tuning.
- **Enhanced Performance:** Models that have been pre-trained on similar tasks generally outperform models that only undergo general pre-training, as they require less adaptation and can leverage the domain-specific knowledge already acquired.
- **Efficient Adaptation:** Domain-adaptive pre-training reduces the amount of adaptation required during fine-tuning. Since the model has already encountered related problems, the fine-tuning phase becomes more efficient, improving both generalization and performance.

3.3 Parameter Efficient Fine-Tuning (PEFT)

3.3.1 Introduction to PEFT (Parameter Efficient Fine-Tuning)

PEFT (Parameter Efficient Fine-Tuning) is an advanced technique designed to adapt large pre-trained models to specific tasks in a resource-efficient way. Traditional fine-tuning approaches, which involve updating all the parameters of a model, can be resource-heavy and costly, especially for very large models. PEFT addresses this issue by minimizing the number of trainable parameters, allowing model adaptation to be done with fewer resources.

PEFT strikes a balance between leveraging the knowledge stored in the pre-trained model and efficiently adapting it to new tasks with only minor adjustments. Among the various techniques available for PEFT, two of the most widely used are *Low-Rank Adaptation (LoRA)* and *QLoRA*, both of which are designed to reduce computational load and memory requirements while maintaining high performance.

3.3.2 Low-Rank Parameters in PEFT

One of the most effective ways to implement PEFT is through the use of *Low-Rank Parameters*. Instead of modifying the entire model's parameters, this approach introduces smaller, low-dimensional parameter sets that are modularly added to the network. These low-rank parameter sets represent the task-specific updates, while the original model's parameters remain frozen, ensuring that the adaptation process is lightweight and efficient.

Let's explore *LoRA (Low-Rank Adaptation)*, a widely used technique that exemplifies the principles of low-rank parameter fine-tuning.

3.3.3 LoRA: Low-Rank Adaptation

LoRA [31] is a key PEFT technique designed for efficient fine-tuning of models for domain-specific tasks. By only updating low-rank matrices and keeping the majority of the model's parameters frozen, LoRA offers a highly memory- and computation-efficient way to adapt large models. This modular fine-tuning approach is particularly valuable in settings with limited computational resources, where full-parameter fine-tuning would be prohibitive.

How LoRA Works:

LoRA fine-tunes a model by modifying only small, low-rank matrices that capture task-specific knowledge, rather than adjusting the entire model. Here's how it functions:

1. **Low-Rank Decomposition:** LoRA replaces full weight updates with the introduction of two smaller matrices, denoted as A and B . These matrices have much lower dimensions than the original weight matrix. During fine-tuning, only these smaller matrices are trained, which approximate the necessary task-specific changes.
2. **Frozen Weights:** The majority of the model’s original parameters are kept unchanged (frozen) during fine-tuning. This reduces memory usage and computational cost, as only the newly introduced low-rank matrices require updates.
3. **Matrix Reconstruction:** After training, the low-rank matrices are combined to approximate the full weight update, which is then added to the original frozen parameters. The final model maintains the same structure and size as the original pre-trained model but includes the learned task-specific adaptations.

Benefits of LoRA in PEFT

LoRA offers several advantages when applied within the framework of PEFT:

- **Memory Efficiency:** By only adjusting small, low-rank matrices, LoRA significantly reduces memory requirements. This makes it possible to fine-tune large models on hardware with limited memory capacity.
- **Computational Efficiency:** Since a minimal number of parameters are updated, the computational cost is reduced. LoRA allows for efficient fine-tuning even for very large models, without requiring expensive hardware.
- **Parameter Efficiency:** LoRA enables task-specific fine-tuning without altering the entire model. This modular approach is ideal for scenarios where multiple tasks must be learned sequentially or simultaneously, as only the small LoRA matrices need to be stored for each new task.
- **Comparable Performance:** Despite the reduction in trainable parameters, LoRA maintains comparable performance to full-parameter fine-tuning in many tasks. This makes it a highly effective solution for adapting large language models to specialized domains.

Key Hyperparameters in LoRA Fine-Tuning

While LoRA simplifies the fine-tuning process, several key hyperparameters influence its performance and effectiveness:

1. Rank (r) The rank controls the dimensionality of the low-rank matrices introduced during fine-tuning. The choice of rank directly impacts both the memory efficiency and the precision of the fine-tuning process.

- **Lower Ranks** (e.g., $r=8$ or $r=16$) are typically sufficient for many tasks and provide significant memory savings while delivering strong performance.
- **Higher Ranks** (e.g., $r=32$ or $r=64$) may capture finer task-specific details for more complex tasks, but the performance gains diminish as ranks increase, especially across all layers.

For most tasks, a rank of 8 or 16 offers a good balance between efficiency and performance.

2. Alpha (Scaling Factor) Alpha is a scaling factor applied to the weight updates from the low-rank matrices before they are added back to the original weights.

- A standard setting is to use $\alpha = 2 \times \text{rank}$ ensuring that weight updates are well-integrated with the frozen model parameters.
- Adjusting alpha allows fine-tuning precision to be controlled, with lower alpha values increasing the effect of fine-tuning and higher alpha values reducing it.

To fit the formula within the text, you can write it as:

3. Dropout **Dropout** is applied during training to prevent overfitting by randomly setting a fraction of the trainable parameters to zero in each batch.

- A **0.1 (10%)** dropout rate is typical for smaller models.
- For larger models, a **0.05 (5%)** dropout rate is recommended to avoid overfitting when dealing with billions of parameters.

4. Layer Selection LoRA’s full potential is unlocked when applied across all layers of a model. Studies show that fine-tuning all layers using LoRA achieves the same or better performance compared to full-parameter fine-tuning. Applying LoRA to only a subset of layers may lead to suboptimal performance.

Considerations for LoRA Fine-Tuning

While LoRA provides a highly efficient method for fine-tuning, the following considerations should be kept in mind:

- **Task Complexity:** Simpler tasks may not require a high-rank setting, while more complex tasks might benefit from experimenting with higher ranks and different alpha values.
- **Layer Coverage:** For best results, apply LoRA across all layers. Limiting the scope of fine-tuning to select layers may compromise performance.
- **Resource Constraints:** LoRA is particularly beneficial in resource-constrained environments, reducing memory and computational demands without sacrificing the quality of the fine-tuning.

3.3.4 Understanding LoRA through an example

LoRA (Low-Rank Adaptation) is a fine-tuning technique designed to simplify the adaptation of large pre-trained models by freezing most of the model’s parameters and learning task-specific updates through smaller, trainable low-rank matrices. Let’s walk through a detailed example using visuals and references to LoRA’s key hyperparameters—**rank (r)** and **alpha**.

Matrix Decomposition with LoRA

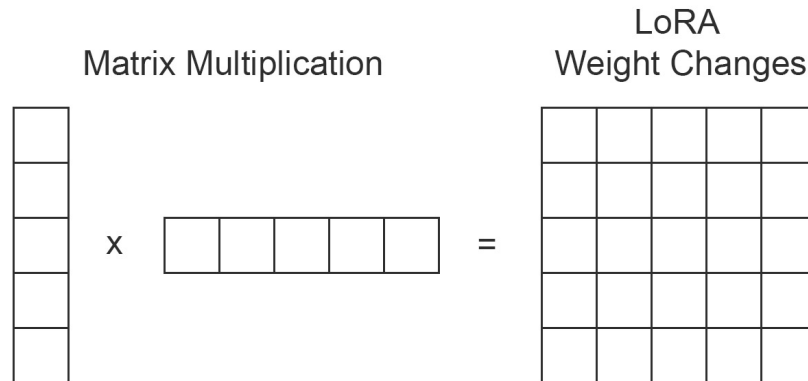


Figure 3.3.1: Matrix Multiplication for LoRA Weight Changes

LoRA operates by approximating weight updates using smaller, low-rank matrices rather than directly updating the full weight matrix of a model. Suppose we have a large pre-trained model where the weight matrix is $W \in \mathbb{R}^{d \times d}$. In traditional fine-tuning, this matrix would be updated directly, but with LoRA, the update is approximated as:

$$\Delta W = A \times B$$

Here:

- $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times d}$, where $r \ll d$ (the rank is much smaller than the original matrix size d).
- **Rank (r)** controls the size of the low-rank matrices. A smaller rank reduces the number of trainable parameters, but too low a rank may affect the precision of the model's task adaptation.

In 3.3.1, you see an illustration of this concept, where:

- A column vector and a row vector are multiplied to form a larger matrix. This demonstrates how small vectors (representing the low-rank matrices) generate a matrix that approximates the original weight update.

The matrix ΔW , formed from the multiplication of $A \times B$, represents the learned updates during fine-tuning. Only these smaller matrices are trained, making the process much more memory and computationally efficient.

Applying LoRA Weight Changes

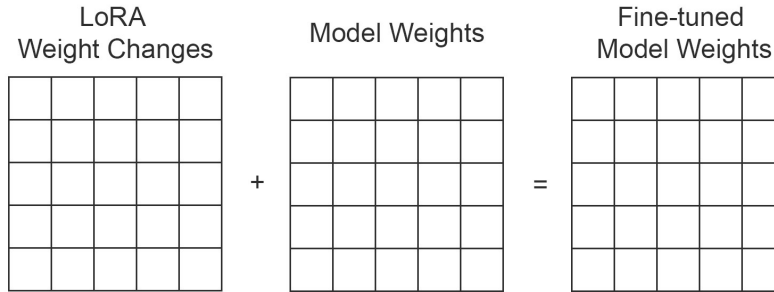


Figure 3.3.2: Adding the LoRA Weight Changes

Once the low-rank matrices A and B have been trained and multiplied to generate ΔW , these updates are then applied to the original frozen weight matrix W . This process is depicted in 3.3.2, where:

1. The **LoRA weight changes** (derived from the multiplication of $A \times B$) are computed.
2. These changes are added to the frozen **model weights**.
3. The result is the **fine-tuned model weights**.

Thus, the overall update to the weight matrix W is given by:

$$W' = W + \alpha \times \Delta W$$

Here, α is the **scaling factor**, another important LoRA hyperparameter that determines how strongly the learned weight changes ΔW are applied to the original model.

Alpha and Its Role:

- **Alpha** is a multiplier that scales the weight changes before they are added back to the original model weights. A common approach is to set $\alpha = 2 \times r$, meaning the weight changes are doubled when applied.
- Smaller values of alpha reduce the impact of the updates (useful if you want more conservative fine-tuning), while larger values increase the effect of the updates, making the model adapt more aggressively to the new task.

In LoRA, adjusting the value of alpha can fine-tune the balance between keeping the original model behavior and allowing for adaptation to new tasks.

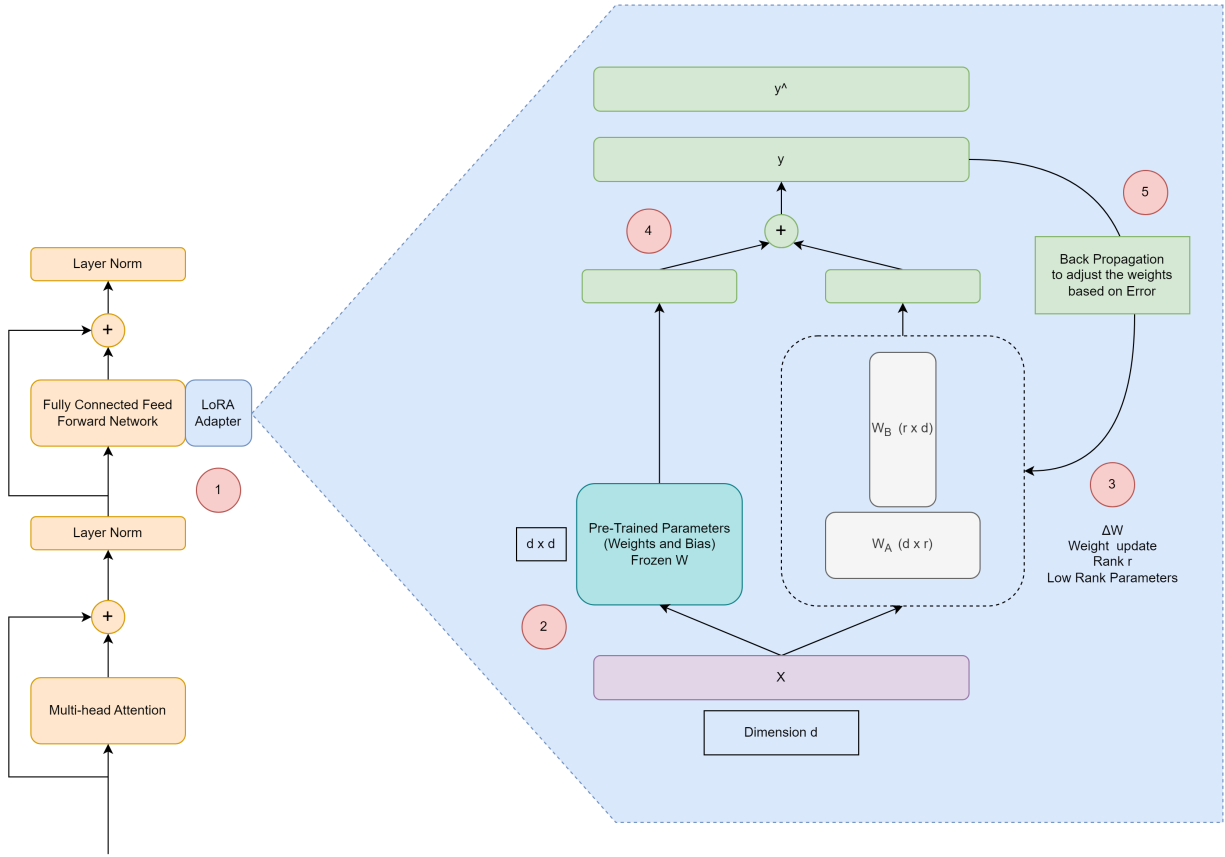


Figure 3.3.3: LoRA Integration in Transformer Architecture

LoRA Integration into the Model Architecture

In 3.3.3, we see how LoRA adapters are injected into a Transformer-based architecture. Let's break it down:

1. **LoRA Implementation** LoRA (Low-Rank Adaptation) is implemented as an adapter that enhances and extends the existing neural network layers. It introduces a set of additional trainable parameters (weights) while keeping the original model parameters frozen. These newly introduced parameters have a significantly lower rank (dimension) compared to the original model's parameters, allowing LoRA to simplify and accelerate the adaptation of pre-trained models for domain-specific tasks.
2. **Freezing the Original Parameters:** The pre-trained parameters of the original model (represented as the weight matrix W) are frozen during fine-tuning. These weights are not updated, reducing the computational load.
3. **Introducing Low-Rank Matrices:** The low-rank matrices $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times d}$ are added to the model, where r is the rank. These matrices are trained, while the original weight matrix W remains unchanged. The smaller the rank r , the more memory-efficient the process becomes, but it also affects the model's ability to capture complex patterns.
4. **Combining Results:** The frozen weights W and the updates ΔW generated by $A \times B$ are combined to produce the final output of the model.
5. **Backpropagation with Adjusted Weights:** During backpropagation, only the weights in the low-rank matrices A and B are updated. The original model weights remain frozen, while $\alpha \times \Delta W$ scales the learned updates, influencing how the model adapts to the new task.

Example with Efficiency Gains (Memory and Speed)

To illustrate the efficiency gains, consider a scenario where we have a **512x512 weight matrix** in the feed-forward layer of the model, resulting in **262,144 parameters**. Using LoRA with a rank of $r = 2$, we reduce the number of trainable parameters as follows:

- Matrix A has dimensions 512×2 , resulting in **1,024 parameters**.
- Matrix B has dimensions 2×512 , also resulting in **1,024 parameters**.

This gives a total of **2,048 parameters** to train, compared to the original 262,144 parameters. This massive reduction in trainable parameters speeds up the training process and reduces memory usage, as fewer calculations are required during backpropagation.

3.3.5 Introduction to QLoRA (Quantized Low-Rank Adaptation)

Quantized Low-Rank Adaptation (QLoRA) [13] builds on the efficiency of LoRA by incorporating **quantization** techniques to further reduce memory usage and computational costs during fine-tuning. The key idea behind QLoRA is to quantize the original model's weights to a lower precision—such as 4-bit integers—while applying low-rank adaptations to adjust the model for task-specific requirements. By combining **low-rank decomposition** with **quantized weights**, QLoRA allows for fine-tuning of large models on memory-limited hardware without compromising performance.

Three major optimizations in QLoRA make it one of the most powerful parameter-efficient fine-tuning (PEFT) methods: **4-bit NF4 quantization**, **double quantization**, and **block-wise quantization**. Let's explore each of these in more detail.

1. 4-bit NF4 Quantization

One of the primary innovations in QLoRA is the use of **4-bit NormalFloat4 (NF4) quantization**, which significantly reduces the memory footprint of the original model by storing weights in a low-precision format. Instead of using high-precision data types like 32-bit floating-point (FP32), QLoRA compresses the weights into a 4-bit format. This process consists of three main steps:

Normalization & Quantization The first step is to normalize the weights. During normalization, the weights are adjusted to have zero mean and a unit variance, ensuring that the values are distributed symmetrically around zero. Once normalized, the weights are mapped to one of 16 discrete values that a 4-bit data type can store, ranging from -1 to 1.

For example, consider a pre-trained FP32 weight with a value of **0.2121**. In the 4-bit quantized representation, the closest available value might be **0.1997** (the 10th position in the range from -1 to 1). Instead of storing the original weight value, the algorithm stores the **index** (10) of the closest quantized position.

The quantization formula is as follows:

$$\text{int4Tensor} = \left(\frac{\text{totalNumberOfPositions}}{\text{absmax}(\text{inputXTensor})} \right) \times \text{FP32WeightsTensor}$$

Where:

- **totalNumberOfPositions** = 16 (for a 4-bit quantization),
- **absmax** represents the maximum absolute value in the input tensor block.

By applying this transformation, the memory required to store weights drops drastically. However, to mitigate potential issues with outliers (which could skew the distribution), QLoRA applies quantization block-wise. This means that weights are quantized independently within smaller blocks (usually groups of 64 weights), preventing any outlier value from significantly affecting the distribution.

Dequantization Once training is complete, QLoRA reverses the quantization process to recover the full-precision values for deployment. This is done using dequantization, where the stored quantized values are converted back to approximate floating-point representations. The dequantization formula mirrors the quantization process:

$$\text{dequantizedTensor} = \frac{\text{int4Tensor}}{\left(\frac{\text{totalNumberOfPositions}}{\text{absmax(inputXTensor)}} \right)}$$

This means the original FP32 weights are reconstructed from the 4-bit representations, allowing for high-quality performance in the fine-tuned model while retaining the memory savings from quantization.

2. Double Quantization

Double quantization is an additional memory-saving technique introduced in QLoRA. After the weights are quantized to 4-bit precision, the **quantization constants** themselves are also quantized, leading to even greater memory efficiency. These quantization constants are the values used to scale the weights during the quantization process.

Here’s how it works:

- During the first quantization step, weights are grouped into small blocks (usually groups of 64), and each block is assigned a **quantization constant**.
- Normally, each of these quantization constants is stored as a 32-bit floating-point value (FP32), which can add considerable overhead, especially in large models.

To reduce this overhead, **double quantization** quantizes the quantization constants themselves. For instance, 256 quantization constants could be grouped together and quantized to **8-bit precision**. This approach reduces the memory required for storing quantization constants from **0.5 bits per parameter** to approximately **0.127 bits per parameter**.

For example, in a model with 1 million parameters, the original quantization constants would require around **500,000 bits** of storage. With double quantization, this is reduced to approximately **125,000 bits**, further shrinking the memory footprint while retaining the quantization benefits.

Advantages of QLoRA

1. **Massive Memory Savings:** By using 4-bit NF4 quantization, QLoRA reduces the memory footprint by **4x** compared to standard 16-bit or 32-bit representations. When combined with double quantization, the memory savings are even greater.
2. **Preserving Model Performance:** Despite reducing the precision of the model weights, QLoRA achieves near-parity with traditional fine-tuning in terms of model performance. This is because the quantization process maintains the core structure of the model, while the low-rank matrices learned during fine-tuning provide the task-specific adjustments.
3. **Fine-Tuning Large Models on Limited Hardware:** QLoRA makes it possible to fine-tune very large models (e.g., models with billions of parameters) on resource-constrained hardware. For instance, a model that would normally require 40GB of memory can be fine-tuned on a **12GB GPU** with QLoRA.
4. **No Additional Inference Costs:** The final model, after fine-tuning, performs inference using the standard weight representation. The 4-bit quantization only affects memory usage during training, meaning the final deployment has no additional overhead.

3.4 Prompting

Prompt-based learning represents a paradigm shift in machine learning, departing from conventional supervised learning approaches. Unlike traditional methods that rely on training models to predict outputs

based on inputs, prompt-based learning harnesses the power of language models to directly model the probability of text. At its core, this approach involves transforming an original input into a textual prompt by using a template wherein certain slots remain unfilled. These slots serve as placeholders for the language model to generate the missing information, resulting in a final string that informs the desired output [51].

One of the defining features of prompt-based learning is its ability to leverage pre-trained language models that have been exposed to massive amounts of raw text data. This pre-training equips the models with a robust understanding of language and enables them to infer missing information in prompts effectively. Moreover, by crafting suitable prompting functions, these models can adapt to new tasks with remarkable flexibility, often requiring only a few examples (*few-shot learning*) or even none at all (*zero-shot learning*).

This framework offers several advantages over traditional approaches. Firstly, it capitalizes on the vast amount of unlabeled text data available, enabling models to capture intricate linguistic patterns and nuances. Secondly, it empowers models to swiftly adapt to novel scenarios, making it well-suited for tasks where labeled data is scarce or costly to obtain. Overall, prompt-based learning emerges as a potent methodology, revolutionizing the landscape of machine learning with its versatility, efficiency, and scalability.

Compared to conventional fine-tuning, which expensively updates the massive LM parameters for each downstream task, prompting concatenates the inputs with an additional piece of text that guides the LM to generate the desired outputs [12].

In general, prompting has rapidly evolved as a standard technique to elicit the "hidden" capabilities of LLMs, successfully serving a vast variety of tasks in NLP and beyond [42, 41, 2, 61, 78].

3.4.1 Prompting Methodology

The primary challenge with traditional **supervised learning** is the need for extensive annotated data to train a model that estimates $P(y|x;\theta)$, where y is the label and x is the input. However, for many tasks, such annotated data can be scarce or costly to obtain. Prompt-based learning techniques in **natural language processing (NLP)** aim to address this challenge by shifting the focus towards training a language model (LM) that estimates the probability of the text $P(x;\theta)$ itself. By doing so, it enables the prediction of y without requiring large labeled datasets. Prompt-based approaches typically involve three core steps to predict the highest scoring answer, as outlined below [51].

1. Prompt Addition

In the first step, a **prompting function** is used to convert the input text into a prompt. This is done by designing a **template** that incorporates two key slots:

- **Input slot [X]**: This slot holds the input text.
- **Answer slot [Z]**: This slot is designed to hold an intermediate answer that will later be mapped to the final output y .

Once the template is created, the input slot **[X]** is filled with the input text. The goal is to structure the task in such a way that the language model can generate a relevant intermediate output by processing the prompt.

2. Answer Search

In this step, the language model searches for the answer that generates the highest probability. To achieve this:

- A **set of possible answers** is defined, typically a list of potential intermediate answers z .
- A function is then used to iteratively fill the answer slot **[Z]** in the prompt with each of the potential answers z .

After each prompt is completed, the language model measures the **probability** of the filled prompt (i.e., the probability of the text generated by the LM) and identifies the **highest-scoring answer**.

3. Answer Mapping

In some tasks, it may be necessary to map the intermediate answer z to the final output y . For example, in **sentiment analysis**, words like "excellent," "good," or "beautiful" might need to be mapped to a final output such as "++" (very positive sentiment). This ensures that the model produces the desired output for the task. However, in tasks such as text generation, this mapping step might be unnecessary, as the output directly corresponds to the generated text.

3.4.2 Prompt Categories

The prompting function can take two forms, either being the same for each input—thus called **static**—or generating a different template for each input—thus called **dynamic**. The categorization can be seen in 3.4.1. In addition to this distinction, prompts are classified into different types [51]:

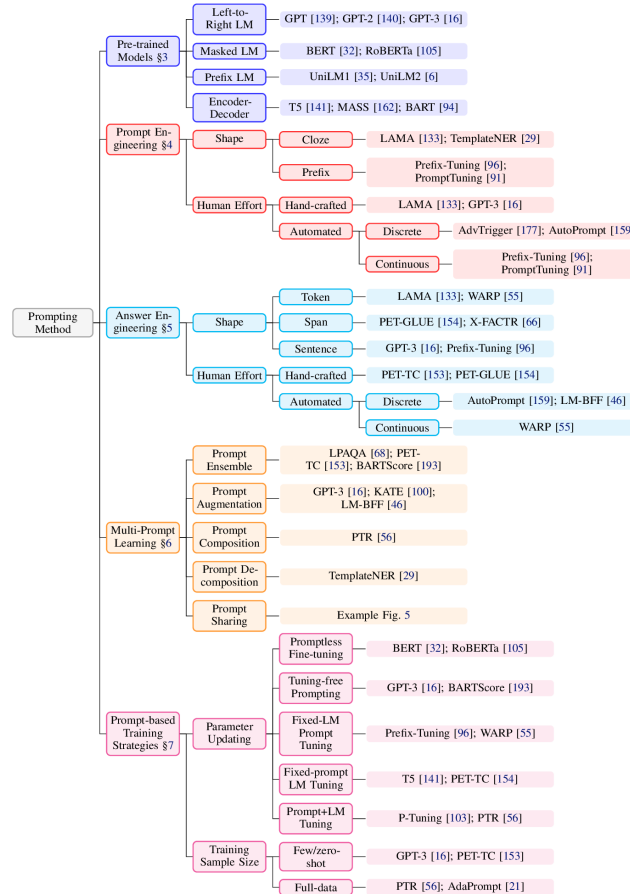


Figure 3.4.1: Prompting Typology [51]

- **Discrete Prompts (a.k.a. hard prompts):** These are specific, hand-crafted text-based prompts written in human-interpretable natural language. For example, a discrete prompt for a sentiment analysis task might be: "The sentiment of the sentence [X] is [Z]," where the model fills in [Z] with appropriate sentiment labels like "positive" or "negative."
- **Continuous Prompts (a.k.a. soft prompts):** These prompts operate directly in the embedding space of the model, rather than in human-readable text. Continuous prompts involve learnable embedding vectors that can be optimized through gradient descent.

While **soft prompts** are popular due to their compatibility with gradient descent, they come with several drawbacks:

- **Lack of interpretability:** Embedding vectors are difficult for humans to comprehend, making soft prompts less interpretable.
- **Incompatibility with other LLMs:** Soft prompts are often incompatible with other large language models and can't be easily transferred, as embedding spaces may differ across models [90].
- **Costly to use:** Soft prompts are typically unavailable for models accessed only through inference APIs, and their use requires access to the model's internal embedding space, making them generally more expensive or impractical to use in deployed environments.

3.4.3 Prompting Techniques

Instruction tuning, where models can learn new tasks based solely on natural language instructions, has been widely proposed and used in various applications [17]. Some of the most prominent methods include zero-shot learning, one-shot learning, few-shot learning, and chain-of-thought prompting.

In **zero-shot prompting**, the model is given an instruction directly without any examples. The model uses its pre-trained knowledge to complete the task based purely on the instruction.

One-shot learning and **few-shot learning** extend this idea by providing one or several examples to the model. In one-shot learning, the model is presented with a single example of the task. In few-shot learning, the model is given multiple examples (typically k examples) of input-output pairs (context and completion), followed by a final input (context) for which the model must generate the appropriate output (completion).

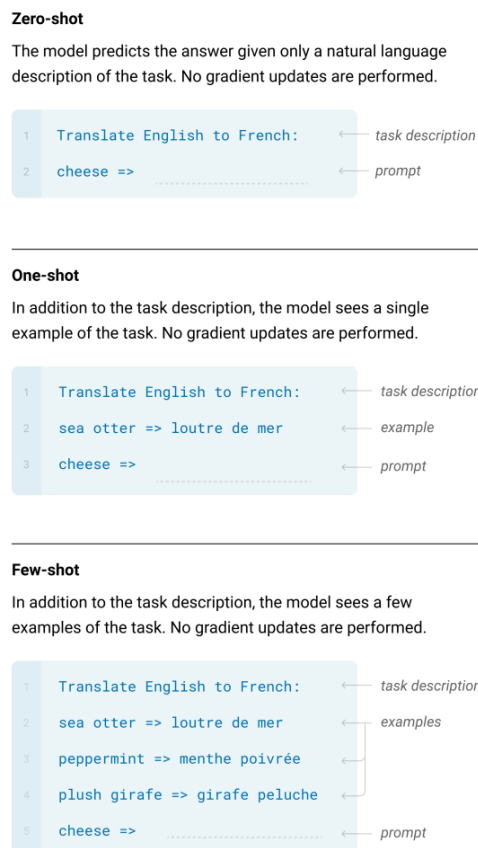


Figure 3.4.2: (a) Zero-shot prompting, (b) One-shot prompting, (c) Few-shot prompting [8]

Few-shot examples can be selected using various criteria:

- **Random selection:** Examples can be chosen randomly from the training data.

- **Semantic similarity:** Examples that are semantically similar to the new task or context can be selected to guide the model more effectively.
- **Diversity:** Choosing a diverse range of examples can help the model generalize better to different inputs.

Chain-of-thought prompting is a method in which the model is guided through a series of intermediate reasoning steps. This approach has been shown to significantly improve the ability of large language models (LLMs) to perform complex reasoning tasks, as it encourages the model to break down problems into smaller steps [87].

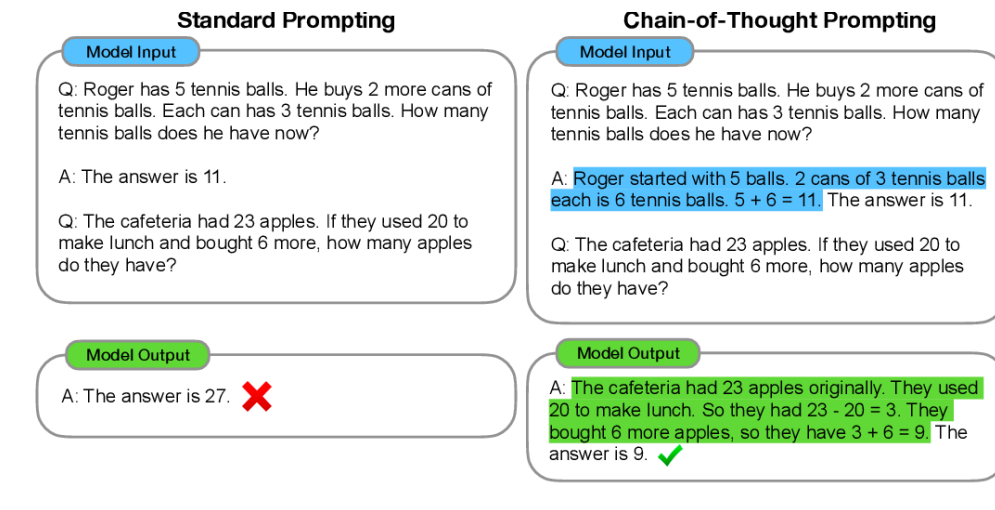


Figure 3.4.3: Chain-of-thought [87]

Few-shot Example Selection Based on Semantic Similarity

When selecting few-shot examples, one of the most effective approaches is to choose examples based on **semantic similarity** between the context of the new task and the existing examples. This selection is typically done using **embeddings** and **cosine similarity** to measure the closeness between different pieces of text.

Embeddings are dense vector representations of words, sentences, or even entire documents. Unlike traditional sparse representations (such as one-hot encoding), embeddings capture the semantic relationships between different linguistic units by mapping them into a continuous vector space. In this space, words or phrases that are semantically similar are positioned closer to each other.

Pre-trained models, like **BERT** or **GPT**, generate embeddings that capture a rich set of features for each word or sentence, derived from vast amounts of training data. These embeddings can be used to compare different text inputs and identify the most relevant examples based on their semantic similarity.

To select few-shot examples based on **semantic similarity**, the process typically involves the following steps:

1. **Embedding Extraction:** First, embeddings are generated for the new task's context (input) and for the potential examples from the dataset.
2. **Cosine Similarity Calculation:** The **cosine similarity** between the embeddings of the new task context and the example contexts is computed. Cosine similarity measures the cosine of the angle between two vectors in a multi-dimensional space. It ranges from -1 (completely dissimilar) to 1 (completely similar). The formula for cosine similarity between two vectors A and B is:

$$\text{Cosine Similarity}(A, B) = \frac{A \cdot B}{\|A\| \|B\|}$$

3. **Example Selection:** The examples with the highest cosine similarity scores are selected as the few-shot examples for prompting.

By selecting examples that are semantically close to the new task, the model is more likely to generate accurate completions or predictions. This method ensures that the selected examples are contextually relevant, thereby improving the performance of few-shot prompting without the need for manual curation.

This approach is particularly useful in tasks where examples with similar structures or semantics can help guide the model towards better performance. Furthermore, embedding-based similarity searches are scalable and efficient, making them well-suited for large datasets and complex tasks.

3.5 Analysis of Vertical and Lateral Thinking

In **Natural Language Processing (NLP)**, reasoning is essential for enabling systems to go beyond surface-level language comprehension and achieve a deeper, more sophisticated understanding. While **Large Language Models (LLMs)** excel in many linguistic tasks, their ability to reason remains limited in key areas, leading to ongoing debates about whether they can truly reason or simply memorize patterns [18, 4, 39, 98, 72, 81, 21, 97]. To address these limitations, it is important to explore different types of thinking—**vertical** and **lateral**—that can challenge and expand a model’s reasoning capabilities.

3.5.1 Vertical Thinking in Reasoning

Vertical thinking refers to a linear, systematic approach to problem-solving. It relies on established rules, logic, and known relationships to reach conclusions. In this mode of thinking, individuals or models work through problems step by step, applying deductive reasoning to derive the correct answer based on the premises provided. Vertical thinking is highly effective in scenarios where the relationships between premises are clear and logical, and where the problem can be solved by following a defined path.

For example, vertical thinking is often used in commonsense reasoning tasks, where models are asked to apply known facts and relationships to reach conclusions. If the premises are consistent and non-contradictory, **Large Language Models (LLMs)** can effectively use their pre-trained knowledge to solve the task by recognizing and applying the correct patterns. However, vertical thinking encounters challenges when the premises appear to conflict, or when creative problem-solving is required. This mode of reasoning is rigid, and it struggles to accommodate situations that demand flexibility or outside-the-box thinking.

Consider the following puzzle:

"How could a cowboy ride into town on Friday, stay two days, and ride out on Wednesday?"

The premises involved include:

- **p1:** The cowboy rides into town on Friday.
- **p2:** The cowboy stays in town for two days.
- **p3:** The cowboy rides out on Wednesday.
- **p4:** Wednesday is the third day of the week.
- **p5:** Sunday is two days after Friday.

Vertical thinking would approach this problem by attempting to reconcile these premises logically. However, the statements appear contradictory, particularly when p4 is introduced, as it conflicts with the timeline suggested by p1 and p2. This leaves the model (or person) stuck, as the conventional logical process leads to a dead end.

3.5.2 Lateral Thinking in Reasoning

Lateral thinking offers an alternative approach to problem-solving, emphasizing creativity, flexibility, and the ability to defy conventional reasoning. Unlike vertical thinking, which is linear and structured, lateral thinking encourages breaking away from established patterns, challenging default assumptions, and exploring

alternative explanations. It involves looking at the problem from new angles and reinterpreting premises that might initially seem fixed.

In the cowboy puzzle, lateral thinking leads to the creative solution:

"The cowboy's horse is named Wednesday."

This solution works because it breaks free from the default assumption that **Wednesday** refers to the day of the week. Instead, lateral thinking reframes **Wednesday** as the name of the cowboy's horse, resolving the apparent contradiction without needing to alter the logical structure of the puzzle itself. Lateral thinking is crucial when a problem seems unsolvable through conventional methods, as it enables the exploration of novel solutions that defy typical expectations.

The essence of lateral thinking lies in identifying and challenging the **implicit premises** that govern our default interpretations. In the cowboy puzzle, p4 ("Wednesday is the third day of the week") is an implicit premise generated through commonsense associations, and it creates a mental "box" that excludes the possibility of Wednesday being anything other than a day. Lateral thinking helps break out of this box by discarding the commonsense association and considering an alternative meaning.

3.5.3 Differences and Applications of Vertical and Lateral Thinking

While both vertical and lateral thinking are essential components of reasoning, they serve different purposes and excel in different contexts:

- **Vertical thinking** is well-suited for tasks that involve logical deduction, strict rules, and clear relationships between premises. It works best in environments where the problem can be systematically deconstructed and solved step by step, without requiring re-interpretation of the premises.
- **Lateral thinking**, on the other hand, is necessary when conventional approaches fail, or when the solution lies outside the bounds of common logic. Lateral thinking promotes creativity and innovation by encouraging the exploration of alternative solutions, particularly in situations where the premises seem contradictory or where common associations create obstacles.

In NLP and LLM reasoning tasks, mastering both vertical and lateral thinking is crucial for solving a wide range of problems. While vertical thinking allows models to handle structured tasks effectively, lateral thinking tests their capacity to think creatively, break away from default assumptions, and generate novel solutions. Both modes of reasoning are valuable, and together they offer a more comprehensive approach to problem-solving.

In the realm of NLP, reasoning is not just about following rules and patterns; it also requires creativity and flexibility. **Vertical thinking** supports structured problem-solving

3.6 Related Work

3.6.1 Reasoning in NLP

The advancements in **pre-trained language models (LLMs)** have led to significant improvements in handling reasoning tasks across various **natural language processing (NLP)** domains, such as commonsense reasoning [67], mathematical reasoning [54], logical reasoning [96], and causal reasoning [20]. Each of these tasks targets different reasoning capabilities: commonsense reasoning involves understanding everyday situations, mathematical reasoning solves numeric problems, and logical reasoning involves systematic deduction based on established rules. Causal reasoning, meanwhile, focuses on identifying cause-and-effect relationships. Several sub-tasks related to reasoning, such as temporal reasoning [82], have also been explored in recent studies [89, 62].

Frameworks like **BigBench** [75] provide large-scale reasoning evaluations with over 204 tasks to challenge the reasoning capabilities of LLMs. Similarly, specific reasoning datasets such as **CommonsenseQA** [77], **WinoGrande** [68], and **RiddleSense** [47] focus on addressing specific reasoning challenges. Many of these datasets are incorporated into platforms like **Tasksource** [73]. For more complex and creative reasoning,

datasets like **BrainTeaser** [38] probe lateral thinking and puzzle-solving abilities, which are crucial for exploring out-of-the-box thinking processes.

3.6.2 Reasoning with Language Models

Language models have been applied to a variety of reasoning challenges, such as **commonsense reasoning** [71], **arithmetic reasoning** [55], **abductive reasoning** [101], **inductive reasoning** [27], **deductive reasoning** [69], and **analogical reasoning** [76]. Reasoning approaches that involve structured, logical thinking—referred to as **vertical thinking**—have been studied extensively across many datasets [48, 7], revealing the systematic approaches LLMs use to solve these tasks.

However, **creative reasoning** remains underexplored in NLP, often omitted from reasoning benchmarks [74, 70], despite the emergent capabilities that larger models display [88]. Creative reasoning tasks such as **puzzle solving** rely on **lateral thinking**, which involves breaking away from conventional patterns of thought. This ability is demonstrated in datasets such as **BrainTeaser** [38], which challenge models to solve problems using unconventional methods.

In our work, we focus on probing both **vertical** and **lateral puzzle-solving reasoning** abilities in LLMs. We take pre-trained **Transformer-based models** that have already been fine-tuned on commonsense reasoning datasets similar to our exploration dataset and further fine-tune them to achieve better reasoning performance. By leveraging pre-trained models in similar reasoning domains, we aim to enhance their adaptability and improve their performance on more specialized tasks.

3.6.3 LLMs and Prompting for Reasoning Tasks

Discovering reasoning patterns in **LLMs** is often done using various prompting techniques [63]. **Zero-shot prompting** involves providing simple instructions like "Let's think step-by-step" to improve reasoning performance without specific examples [40]. More structured approaches, like **Chain-of-Thought (CoT) prompting**, guide models through intermediate reasoning steps, encouraging deeper thought processes in complex tasks [87].

In **few-shot prompting**, where a small set of examples is provided, challenges arise in selecting and placing exemplars effectively. Research shows that choosing and ordering exemplars can influence model performance [16], and selecting based on **semantic similarity** has been the default method [50]. Recently, improvements have been made by ordering exemplars based on complexity or reasoning diversity [19, 100].

Our work proposes a **prompting technique** that builds upon these baseline methods, but rather than focusing solely on semantic similarity for selecting exemplars, we design prompts that capture **hidden reasoning patterns**. While we use similarity-based retrieval for exemplar placement, the goal is to emphasize the reasoning structures within the task, rather than focusing purely on the content's surface meaning. This approach allows us to outperform standard prompting techniques, demonstrating that focusing on reasoning patterns can provide more advanced LLM reasoning capabilities without requiring extensive prompt engineering.

Chapter 4

Proposal 1 - SemEval-2024 Task 9

In this chapter, we outline our proposal for the **SemEval-2024 Task 9 competition: "BRAINTEASER: A Novel Task Defying Common Sense"** [36]. This proposal, which was accepted as a paper for the SemEval 2024 conference [59], details our planned submission, which will focus on both Sub-task A - Sentence Puzzle and Sub-task B - Word Puzzle. The aim of our approach is to improve the performance of transformer-based language models on lateral thinking challenges. By fine-tuning and evaluating various pre-trained models, we seek to gain insights into their reasoning capabilities and provide a deeper understanding of their strengths and weaknesses in this context.

This proposal will involve a comparative analysis of the two categories, focusing on fine-tuning techniques, model size, and the impact of hyperparameters on performance. In particular, we will examine the effect of model size on the BRAINTEASER task, as existing literature has highlighted its significant influence on the reasoning capabilities of models [79, 86]. Additionally, we will explore how various tuning hyperparameters contribute to performance. Through this work, we aim to achieve competitive results on the leaderboard and provide valuable insights for future research into transformer models' effectiveness in reasoning tasks that challenge common sense.

4.1 Contributions

To sum up our contributions are:

1. We perform lightweight tuning on smaller encoder models and LLMs, significantly outperforming the reported baselines.
2. We transform the multiple-choice problem to a binary classification one, aiming to explore diverging reasoning paths for models.
3. We ground final performance on the models' "prior knowledge" in related problems.
4. We delve into models' frequent failures to obtain a deeper understanding of reasoning cues that make models struggle the most.

Our code is available on GitHub ¹.

4.2 Methods

We focus on tuning language models belonging into two categories. First, we fine-tune variations of *encoder* models, namely BERT [14], RoBERTa-large [52] and DeBERTaV3-base [29], to assess the impact of transfer learning using various datasets requiring similar reasoning abilities, apart from BrainTeaser. We study the problem using the provided *multi-choice* setup, but we also transform it into a *binary* classification

¹ [AILS-NTUA-at-SemEval-2024-Task-9-Brainteaser](#)

task. Secondly, the encoders' results are compared with those obtained from *fine-tuned LLMs* using the BrainTeaser dataset. To achieve this, we fine-tune Llama 2 [79], Phi-2 [26] and Mistral-7b [34], which have already demonstrated enhanced reasoning abilities. In this regard, we examine the effect of the model size on our task, which has already been reported in the literature to significantly influence the reasoning abilities of the models [79, 86], along with other tuning hyperparameters. Model details are presented in 5.1.4 and 5.1.5.

4.2.1 Encoder models

Pre-training First, we evaluate the effects of the pre-training on our task. Thus, we select two variations of each encoder: The *vanilla* one (using the default pre-trained basis and fine-tuned on BrainTeaser data only) and one that has undergone additional pre-training using supplementary commonsense reasoning datasets before fine-tuned on BrainTeaser. This enables us to evaluate the differences in performance that the additional pre-training might offer.

The objective is to enhance model performance by allowing it to learn from a diverse range of data sources before fine-tuning on the sub-task-specific dataset.

In the second case, we use the following pre-trained models:

- BERT-SE: a BERT-base-uncased version pre-trained on the multiple-choice dataset used in SemEval-2020 Task 4b [84]
- RoBERTa-WNGRD: a RoBERTa-large version pre-trained on the WinoGrande dataset
- DeBERTaV3-TS: a DeBERTaV3-base model, pre-trained on diverse commonsense reasoning datasets, and fine-tuned with multi-task learning on over 600 tasks from the Tasksource collection.

Datasets employed across both of our sub-tasks include CommonsenseQA [77], WinoGrande [68], ai2_arc [10], Physical Interaction: Question Answering (piqa) [6], Choice of Plausible Alternatives (COPA) [22], and the SemEval-2020 Task 4 dataset [84]. These datasets would offer valuable insights into commonsense reasoning, logical reasoning, and contextual cues, enabling the development of a robust understanding of commonsense principles within our sub-tasks' final system.

Multi-class Classification task This strategy involves treating the problem as multi-class classification: all four provided options are combined with the given question, and consequently these concatenated inputs are fed into the model, which is fine-tuned to select one of the four options as part of a multi-class classification problem.

This multiple-choice format allows the model to consider all options simultaneously, fostering lateral thinking and encouraging the exploration of different reasoning paths. By taking a comprehensive view of the available choices, the model can analyze the relationships between them, leading to more informed and accurate decision-making. The ability to engage with the complexity of the problem enhances the model's understanding, ultimately improving its prediction accuracy by enabling it to select the most plausible answer. This holistic approach, which embraces the full context of the problem, significantly strengthens the model's reasoning capabilities.

Binary Classification task In this approach, both sub-tasks were transformed into binary classification problems. Each original multiple-choice question, which had four options, was converted by pairing each candidate answer—excluding "None of the above"—with the question. A label of 0 was assigned if the option was incorrect and 1 if it was correct. If all three pairings returned 0, "None of the above" was inferred as the correct answer.

This transformation resulted in each multiple-choice question being split into three binary classification problems. Each option was combined with the original question, and labels were assigned accordingly. The "None of the above" option was not directly included, as its correctness was implied when the other three pairings were incorrect. This modification aimed to streamline identifying correct answers when none of the given options were valid.

Initially, we hypothesized that framing the task as binary classification would enhance the model’s ability to evaluate each option independently. By reducing the task to a series of binary decisions, we expected the model to avoid confusion caused by distractor choices and focus on assessing the correctness of each answer.

However, after fine-tuning several pre-trained models for this approach, the results were disappointing. The model often struggled to assess the options correctly, leading to multiple errors. It occasionally predicted more than one answer as correct or labeled all choices as incorrect, defaulting to "None of the above" incorrectly.

These outcomes indicated that the binary classification approach did not capture the nuances of the original problem. The model’s difficulty in reasoning independently through each option suggested that breaking the task down in this way was ineffective. Therefore, we decided to revert to the original multiple-choice format.

4.2.2 LLMs

We demonstrate an in-depth examination of fine-tuning SoTa LLMs (Llama 2, Phi-2, Mistral-7b and Mixtral-8x7b) in the context of multi-class classification. Note that during inference, the models prompted to provide an *explanation* along with the label. This experimental step, which we have observed to improve the performance of the model, also provides a qualitative identification of flaws in the models’ reasoning process. In our experiments, we explore various combinations of LoRA [31] α and r hyperparameters, using values of 16, 32, 64, and 128. For the analysis ahead, LLMs are denoted as `model_r_a`, reflecting these hyperparameters. Additional technical information, including prompting details and specifics about QLoRA hyperparameters, is available in 5.2, 5.3, 5.1.6.

Chapter 5

Experiments - SemEval-2024 Task 9

Consider the following puzzle:

"Who shaves every day but still has a beard?"

The solution—"A barber"—requires lateral thinking, as it defies the immediate assumption that someone who shaves daily would not have a beard. Instead, the puzzle's solution hinges on the unexpected twist that the barber shaves other people but keeps his own beard. This illustrates how **lateral thinking** can be used to solve problems by challenging default assumptions and thinking creatively.

In **Natural Language Processing (NLP)**, **Large Language Models (LLMs)** excel at many linguistic tasks but often struggle with reasoning challenges that demand creative or unconventional thinking. The **SemEval-2024 Task 9** competition, titled "**BRAINTEASER: A Novel Task Defying Common Sense**" [36], is designed to evaluate the ability of LLMs to tackle both **vertical thinking** (logical, step-by-step reasoning) and **lateral thinking** (out-of-the-box problem-solving) challenges.

Our experiments focus on enhancing the performance of LLMs in these reasoning tasks through fine-tuning pre-trained transformer models and leveraging transfer learning. The competition includes two sub-tasks:

1. **Sub-task A (Sentence Puzzle):** Involves solving sentence-level puzzles requiring both logical and creative reasoning.
2. **Sub-task B (Word Puzzle):** Focuses on word-based puzzles that require alternative interpretations of meaning.

The experiments addressed two core research questions:

- **RQ1:** Does fine-tuning pre-trained transformer models on reasoning-specific tasks improve their ability to solve lateral thinking puzzles compared to standard prompting techniques?
- **RQ2:** Can transfer learning from similar reasoning tasks, combined with fine-tuning, be used as an effective strategy to boost performance across both lateral and vertical reasoning tasks?

This chapter explores these research questions, investigating how factors such as model size, pre-training, and fine-tuning strategies impact the performance of LLMs on reasoning tasks. The insights gained provide valuable understanding of how model adaptation can enhance the reasoning abilities of LLMs, particularly in tasks that require lateral thinking.

5.1 Preliminaries

5.1.1 Dataset

The BrainTeaser task at SemEval-2024 [38, 36] features lateral thinking puzzles presented as multiple-choice questions (QAs). Each question offers four options, with one being the correct answer and the others serving as distractors. Additionally, the final option is always "None of above". It consists of two sub-tasks, *Task A: Sentence Puzzle* and *Task B: Word Puzzle*. In addition to the original puzzles, the dataset includes adversarial subsets created by manually modifying the original brain teasers while preserving their reasoning paths. The original data were perturbed in two ways: First, there is *semantic reconstruction* of each original question without altering the answers or the distractors. Second, the original data underwent *context reconstruction*, wherein the original reasoning path remains intact, but the brain teaser describes a new situational context. Overall, the dataset used for training and evaluation consists of triplets of data: original, semantic, and context reconstruction. Table 5.1 provides an example of the triplets of data that constitute the dataset.

Question	Choice
<i>Original</i>	
What kind of nut has no shell?	A peanut.
	A doughnut.
	A walnut.
	None of above.
<i>Semantic Reconstruction</i>	
Which nut doesn't have a shell?	A doughnut.
	A walnut.
	A peanut.
	None of above.
<i>Context Reconstruction</i>	
Which type of bell doesn't make a sound?	A fire bell.
	A cow bell.
	A bluebell.
	None of above.

Table 5.1: Illustration of the structure of each sub-task's dataset, showcasing the original statement along with its two adversarials. The correct answers are **highlighted in bold**

Task A: Sentence Puzzle In this sub-task, the sentence pairs are crafted in a manner that makes it relatively easy for humans to discern the correct statement, yet challenging for systems, even those equipped with commonsense understanding. Table 5.2 contains examples of the Sentence Puzzle dataset (on the left). The training data consists of 169 distinct multiple-choice QA sets, each accompanied by its semantic and context reconstructions, resulting in a total of 507 multiple-choice questions (3×169).

Task B: Word Puzzle involves word-type brain teasers, where the answer defies the default meaning of the word and focuses on the letter composition of the question. The training dataset comprises 132 multiple-choice QAs, each accompanied by its semantic and context reconstructions, resulting in a total of 396 multiple-choice QAs (3×132). These brain teaser categories include puns, homophones, ambiguous words, and various other linguistic puzzles, as showcased in the examples provided in Table 5.2 on the right-hand side. The Word Puzzle sub-task pose challenges not only for systems but also for humans in discerning the correct answer.

Data statistics The BrainTeaser dataset comprises 3 data splits, namely train, development (used during the practice phase), and the hidden test set, which was used for evaluation. Statistics are provided in Table 5.3.

<i>Sentence Puzzle</i>		<i>Word Puzzle</i>	
Question	Choice	Question	Choice
A man shaves everyday, yet keeps his beard long.	He is a barber.	What has toes but no feet or legs?	Cabbages.
	He wants to maintain his appearance.		Tomatoes.
	He wants his girlfriend to buy him a razor.		Onions.
	None of above.		None of above.
You go to the doctor because you're sick, and he gives you three medicines to take every half hour. How long do the drugs keep you going?	One and a half hours.	What did the little lobster get on its math test?	Sea-plus.
	Two hours.		Very-bad.
	An hour.		Very-Good.
	None of above.		None of above.
How many times can you deduct 10 from 100?	Once.	What's the beginning of an argument?	The letter T.
	Infinite time.		The letter A.
	Twice.		The letter U.
	None of above.		None of above.

Table 5.2: Example questions illustrating both sub-tasks, with correct answers **highlighted in bold**. Examples on the left pertain to *sub-task A: Sentence Puzzle*, while those on the right correspond to *sub-task B: Word Puzzle*.

Throughout the evaluation phase, the leaderboard was kept concealed.

Sub-task	Train	Dev	Test
A - Sentence Puzzle	507	120	120
B - Word Puzzle	396	96	96

Table 5.3: Data statistics.

5.1.2 Evaluation metrics

Both sub-tasks are assessed via accuracy metrics to gauge the performance of participating systems in two ways. First, instance-based accuracy evaluates each question individually, considering original questions and their semantic and context adversarials. This metric provides a detailed understanding of a model’s proficiency in reasoning through various scenarios. In contrast, group-based accuracy takes a broader perspective, assessing questions and associated adversarials as cohesive groups. Each group consists of three questions, and a model scores 1 only if it correctly solves *all* questions in a group. This approach evaluates the system’s holistic performance in navigating through lateral thinking challenges. The combined use of instance-based and group-based accuracy metrics provides comprehensive insights into the capabilities of participating systems in tackling the complexities of both sub-tasks.

5.1.3 Baselines

For both sub-tasks of the BRAINTEASER task, we utilized three baseline models: Human, ChatGPT, and RoBERTa-L. These models were not fine-tuned and were used exclusively in the multiple-choice format. The Human baseline provides an upper-bound reference, representing the ideal performance on the task, while ChatGPT and RoBERTa-L serve as automated baselines to assess performance without task-specific adaptation. In our results discussion, these baseline scores will be highlighted in gray to facilitate direct comparison with the performance of our fine-tuned models, illustrating the impact of our fine-tuning efforts.

5.1.4 Encoders

In this subsection, we present the encoder models utilized in our experiments. We first describe the *vanilla* versions of the models, followed by a discussion of pre-trained variations fine-tuned on similar reasoning datasets to explore the impact of additional prior knowledge on our task.

Vanilla variations

BERT [14]: Bidirectional Encoder Representations for Transformers, is a pretrained deep bidirectional transformer model producing context representations. Using a fine-tuning setting, BERT has advanced state-of-the-art performances on a wide range of NLP tasks.

RoBERTa-large [52]: Robustly Optimized BERT pre-training Approach (RoBERTa) is an adaptation of BERT architecture trained with larger batches on 160 GB data from various domains. RoBERTa-large was trained by dynamically modifying language masking while the next sentence prediction loss used in BERT was dropped. Other improvising techniques like larger input text sequences, byte pair encoding are used in training which seemingly improved the model performance in downstream tasks.

DeBERTaV3 [29]: Decoding-enhanced BERT with disentangled attention is an extension of the original DeBERTa model. It builds upon the BERT (Bidirectional Encoder Representations from Transformers) architecture, aiming to enhance its decoding capabilities and overall performance across various natural language processing (NLP) tasks. DeBERTaV3 further improves the efficiency of DeBERTa [30] using ELECTRA-Style pre-training with Gradient Disentangled Embedding Sharing. Compared to DeBERTa, V3 significantly improves the model performance on downstream tasks. It incorporates a disentangled attention mechanism to allow the model to focus on different aspects of input independently, improving its ability to capture diverse linguistic patterns. The model also features enhancements in the decoding process, enabling more accurate text generation and sequence classification.

Pre-trained variations

We now provide details about the pre-trained variations of the encoder models, which were tuned on reasoning datasets similar to our task.

BERT_{base-uncased} We utilize a version pre-trained on a similar multiple-choice dataset employed in SemEval-2020 Task 4b. This dataset comprises natural language statements that contradict commonsense, accompanied by three reasons explaining why they lack coherence. Systems participating in this task were tasked with selecting the correct justification. Upon thorough examination and comparison of the dataset from that task with ours, we conclude that employing this pre-trained version of BERT_{base-uncased}, rather than the original model, would enhance the integration of commonsense logic and reasoning in our configuration. Henceafter, we denote this model as *BERT_{base-uncased-SemEval-2020}*.

RoBERTa_{large} Instead of employing the original version of the model, we opt for a pre-trained version trained on the WinoGrande dataset. As previously mentioned, the WinoGrande dataset serves as a benchmark for several commonsense tasks, making a pre-trained version of the vanilla RoBERTa model more adept at our sub-tasks. This model exhibits enhanced reasoning comprehension. For our experiments, we employ the RoBERTa_{large} pre-trained model on the WinoGrande dataset, which we denote as RoBERTa_{large}-Wngrd.

DeBERTaV3_{base} We choose to utilize a pre-trained version of DeBERTaV3_{base}, which is available in the Hugging Face [93] library of models. More specifically, we are leveraging the DeBERTaV3-base model, which has been fine-tuned with multi-task learning on over 600 tasks from the tasksource collection [73]. This DeBERTa model has undergone pre-training across various commonsense Reasoning datasets mentioned earlier. Consequently, we anticipate notable enhancements in commonsense reasoning, logical reasoning, and the incorporation of knowledge from diverse data sources that align with distributions similar to our sub-task specific dataset. The aforementioned model will be referred to as DeBERTaV3_{base}-TS.

5.1.5 LLMs

In this subsection, we introduce the large language models (LLMs) utilized in our experiments. These models are known for their robust performance in complex natural language processing tasks, such as reasoning, text generation, and code comprehension. Each of these models varies in terms of parameters, architecture, and

training techniques, providing us with diverse approaches to tackle the BRAINTEASER task. Below, we describe the LLMs we employed, focusing on their key features and suitability for the specific challenges presented by our sub-tasks.

Mistral-7b [33] Developed by EleutherAI, is a language model tailored for large-scale natural language processing tasks. With its 7 billion parameters, it excels in handling complex language understanding and generation tasks. Designed to perform exceptionally well across various NLP applications such as text generation, comprehension, and summarization, Mistral-7b surpasses the best open 13b model, Llama 2 [79], and the best released 34b model, Llama 1 [80], in reasoning, mathematics, and code generation tasks. Leveraging grouped-query attention (GQA) and sliding window attention (SWA), Mistral-7b ensures efficient inference and can handle sequences of arbitrary length with reduced inference cost. Its performance across a wide range of benchmarks makes it a promising solution for our sub-tasks, given its extensive task capabilities and superior performance in baseline benchmarks compared to similar or larger language models. While we considered experimenting with its larger variant, Mixtral-8x7b [34], limitations on available resources forced us to deal in depth only with the small variant, Mistral-7b.

Llama 2 [79] A language model that represents a significant advancement in natural language processing. It is a collection of pre-trained and fine-tuned large language models (LLMs) ranging in scale from 7 billion to 70 billion parameters. With its large parameter count and advanced architecture, Llama 2 is designed to tackle complex language understanding and generation tasks effectively. It outperforms many other models, including its predecessor, Llama 1, in various benchmarks, demonstrating superior capabilities in reasoning, mathematics, and code generation. Leveraging its extensive parameterization and innovative techniques, Llama 2 offers state-of-the-art performance across a wide range of NLP applications, making it a notable contender in the field. For our experiments we were able to experiment with various configurations with the 7 billion and the 13 billion models. Our involvement with the 70 billion parameter model has been restricted due to limitations associated with the extensive parameter count, particularly during the fine-tuning process.

Phi-2 [26] An advanced language model designed to address complex natural language processing tasks efficiently. It is part of the small language models (SLMs) released by Microsoft Research team. With its innovative architecture and extensive parameter count, Phi-2 surpasses its predecessor, Phi-1, in various benchmarks, showcasing superior performance in reasoning, comprehension, and text generation. Leveraging cutting-edge techniques and a comprehensive understanding of language patterns, Phi-2 demonstrates remarkable capabilities across a diverse range of NLP applications, solidifying its position as a prominent model in the field. Given its 2.7 billion-parameter architecture, which exhibits exceptional reasoning and language understanding abilities in comparison to various Llama 2 iterations and Mistral-7b, we are confident that this model will deliver noticeable performance for both of our sub-tasks.

5.1.6 Prompting Details

Here, we provide a comprehensive overview of the prompt utilized consistently throughout the fine-tuning process of the LLMs, which ultimately led to optimal performance across both sub-tasks. The LLMs were explored exclusively in the multiple-choice format, as this setup offered the necessary context for reasoning through the provided options. By maintaining consistency in the prompt structure during fine-tuning, we were able to effectively enhance the models' ability to handle the complexities of both the Sentence Puzzle and Word Puzzle sub-tasks, leading to improved performance.

System Prompt:

Below is an instruction that describes a multiple choice task. Answer the following multiple choice question by giving the most appropriate response. Answer should be one among options provided after the question. Select the most suitable answer while making the necessary assumptions. Give only answer and a short explanation of two or three sentences. Nothing else.

User Prompt:

Question: {question}
1) {a}

- 2) {b}
- 3) {c}
- 4) {d}

Answer:

The correct answer is: {label}) {answer}

Here is the explanation of each parameter used in the user prompt:

- **question**: This variable represents the actual question posed to the model. It serves as the main input for which the model must reason and generate an appropriate answer.
- **{a}**, **{b}**, **{c}**, **{d}**: These variables correspond to the four possible answer choices presented for each question in the multiple-choice format. Each choice is associated with a numerical identifier (1, 2, 3, or 4) and represents a distinct option the model can select.
- **{label}**: This variable denotes the number (1, 2, 3, or 4) corresponding to the correct answer option among the four provided choices. It indicates the correct option’s index.
- **{answer}**: This variable contains the text of the correct answer choice. It corresponds to the content of the selected correct option (either **{a}**, **{b}**, **{c}**, or **{d}**) as identified by the label.

In this setup, the prompt presents a question along with four possible answer choices, and the model is fine-tuned to generate the correct answer by selecting the appropriate label and corresponding answer text.

5.2 Experimental Setup

In our experiments, we employed the Google Colab platform and Kaggle, leveraging various open-source Python packages such as Transformers, TRL (Transformer Reinforcement Learning) [91], PEFT (Parameter-Efficient Fine-Tuning) [57], BitsAndBytes, Accelerate [25], and Sentence-Transformers.

5.2.1 Encoders

BERT-SE ¹ During fine-tuning, a learning rate of $3e^{-5}$ was used, with a batch size of 16 samples processed in each iteration, over the course of 3 epochs. This process aimed to adapt the pre-trained model to better suit our sub-task. Our optimizer was AdamW and our learning scheduler was linear. Same setup was used for the fine-tuning of the BERT encoder.

RoBERTa-WNGRD ² underwent fine-tuning on the train split of each dataset, utilizing a learning rate of $3e^{-5}$, a batch size of 16, and running for 3 epochs. The optimizer was also AdamW and the learning scheduler was linear. RoBERTa-large was fine-tuned on the train split of each sub-task’s specific dataset using identical configurations.

DeBERTaV3-TS ³, like DeBERTaV3-base, underwent a fine-tuning process similar to the RoBERTa-WNGRD system, differing only in the batch size, which was set to 4.

5.2.2 LLMs

Phi-2 ⁴ underwent fine-tuning using the prompt format outlined in Section *Prompting Details*. The fine-tuning process involved setting a learning rate of $2e^{-5}$ and a batch size of 2, with the model trained for 250 steps. We conducted experiments with different configurations of r and lora_alpha , encompassing combinations such as $r = 64, 128$ and $\text{lora_alpha} = 64, 128$. The dropout rate was consistently set to 0.1

¹[JazibEijaz/bert-base-uncased-finetuned-semeval2020-task4b-append-e3-b32-14e5](#)

²[DeepPavlov/roberta-large-winogrande](#)

³[sileod/deberta-v3-large-tasks-source-nli](#)

⁴[microsoft/phi-2](#)

across all experiments. We used an AdamW optimizer and a constant learning scheduler. Despite promising benchmarks accompanying its release, the model’s performance during inference on the test split of both sub-tasks’ datasets was subpar, scoring lower compared to the encoders mentioned above. This discrepancy raises the possibility, supported by various reports, that the model’s training process using methods like quantization and LoRA may not be fully optimized yet, particularly given its recent introduction.

Both variations of Llama 2 ⁵, with 7 billion and 13 billion parameters, underwent the same fine-tuning pipeline described earlier, utilizing the QLoRA technique. The fine-tuning process followed the prompt format outlined in Section 5.1.6 (*Prompting Details*), employing a learning rate of $2e^{-5}$ and a batch size of 1, with each model trained for 250 steps. Despite experimenting with various combinations of values for r and a (32, 64, 128), while the dropout rate was consistently set to 0.1, the results were disappointing. As a text generation model, Llama 2 provided explanations for each multiple-choice prompt. However, even when incorrectly predicting a choice as correct, the generated explanations often lacked logical coherence. Many explanations produced during the inference phase were irrelevant to the context of the brain teaser, indicating a failure to capture the reasoning path of most multiple-choice questions. In summary, both variations of Llama 2, despite their large scale, proved incapable of effectively understanding and reasoning through the multiple-choice questions provided.

Mistral-7b ⁶ model outperformed all others significantly. Prior to fine-tuning, we applied the QLoRA technique. Using a learning rate of $2e^{-5}$ and a batch size of 2, each model underwent fine-tuning for 250 steps using the train split of the sub-tasks’ dataset. The initial results were promising. During experimentation with the r and a parameters, while maintaining a dropout of 0.1, certain patterns emerged. Specifically, we observed higher quality explanations and scores when using higher rank values, ranging from (16, 32, 64, 128). This outcome was expected, as higher rank values correspond to higher precision weight changes, resulting in superior weight tuning and overall model performance. Interestingly, when the ratio of a/r was low (0.5 - 1), explanations maintained high quality irrespective of predictions, implying a coherent reasoning path even if the predicted choice was incorrect. However, setting the a/r ratio to 2 or 4 potentially enhanced results, signifying a stronger influence from QLoRA layers on the base model. However, this adjustment led to a decline in the quality of explanations. The improvement could be attributed to the model’s low intrinsic dimensionality. Despite having many parameters, the effective dimensionality of the model’s learned representations is low. Consequently, after conducting several experiments, the best-performing model regarding word puzzles aligns with this concept. After conducting numerous tests, we achieved our best performances with the first model using $r=128$ and $\alpha=128$, and the third best using $r=64$ and $\alpha=32$. These models are denoted as Mistral-7b_lora_r_lora_a, representing Mistral-7b_128_128 and Mistral-7b_64_32 configurations, respectively.

Mistral-8x7b ⁷ Our exploration of Mistral-8x7b was constrained, yet initial results were promising, despite the limited configurations. Further experimentation with various hyperparameter settings may yield improved performance. In our single attempt with this system, we employed a learning rate of $2e^{-5}$ and a batch size of 2, fine-tuning the models for 250 steps using the train split of the sub-task’s dataset. Both r and a were set to 128, accompanied by a dropout rate of 0.1. This configuration was selected based on the r and a values of the best-performing model across both sub-tasks, Mistral-7b. Despite its larger scale, Mistral-8x7b achieved the second-best accuracy during inference on the test split regarding the first subtask, trailing behind its smaller variation, Mistral-7b. This model is referenced in the results table of both sub-tasks as Mistral-8x7b_128_128. Further experimentation with various configurations may yield improvements, particularly when leveraging the low intrinsic dimensionality and redundancy inherent in the model.

5.3 QLoRA hyperparameters

Initially, we employed the QLoRA technique [13] for optimization. The QLoRA technique entails the following steps. First we quantized the models using 4-bit precision to reduce memory usage and computational

⁵[meta-llama/Llama-2](#)

⁶[mistralai/Mistral-7B-v0.1](#)

⁷[mistralai/Mistral-8x7B-v0.1](#)

requirements. The quantization process was facilitated by the BitsAndBytes library. Following quantization, we implemented the LoRA technique [31] using the PEFT library. LoRA, applied to the quantized model, resulted in the creation of Quantized LoRA (QLoRA). This pipeline effectively addresses the challenges posed by memory-intensive models on hardware with limited capabilities, ensuring optimized performance and resource utilization. Regarding the hyperparameters of the QLoRA, the rank (r) determines the dimensionality of the low-rank approximation used in the adapter layers, while alpha (α) is the scaling factor that determines the magnitude of the newly learned weights compared to the original model’s weights. The choice of alpha influences how much emphasis is given to the task-specific information compared to the pre-trained knowledge encoded in the original model.

In our experiments, we observed that lower values of r occasionally yielded slightly superior results. This phenomenon can be attributed to the regularization effect introduced by lower-rank approximations. Essentially, lower-rank approximations act as a form of regularization, discouraging the model from memorizing the training data and instead promoting the learning of more generalizable patterns. This regularization effect becomes particularly significant when dealing with small datasets, as the risk of overfitting is heightened in such scenarios. By limiting the model’s capacity through lower-rank approximations, we encourage it to focus on learning essential features and avoid capturing noise or idiosyncrasies present in the training data. Therefore, in our case where the dataset size is small, the regularization provided by lower-rank approximations becomes crucial. It helps prevent overfitting and encourages the model to generalize better to unseen data, ultimately leading to improved performance in certain cases.

5.4 Results

5.4.1 Experimental results and Analysis

Our metrics for the Sentence Puzzle sub-task are presented in Table 5.4 and for the Word Puzzle sub-task in Table 5.5 along with their baselines. Interestingly, the performance of the binary classification problem is significantly lower than that of the multi-class classification task. Initially, this behavior seemed counterintuitive since it appeared easier to determine whether a question is correct or not than to select the correct answer from four different options. However, this assumption is not accurate. Consider the word riddle: ‘*What is the capital in France?*’ At first glance, the option ‘F’ seems incorrect, but when considering the options ‘F,’ ‘E,’ ‘A,’ and ‘None of the above,’ ‘F’ emerges as the only correct answer, as it becomes apparent that the question refers to the capital *letter* rather than the capital *city*. Therefore, the diverse options provide crucial context to the models, explaining the superior performance of multi-class models. This lack of context is why we refrain from further exploring this methodology across all models in our study.

Task A: Sentence Puzzle

Table 5.4 illustrates minimal fluctuations among all instance-based metrics. This consistency extends to the associated group-based metrics for all models, highlighting a systematic behavior towards detecting various reasoning paths. This observation holds for both the encoder-based classifiers and LLMs utilized in this sub-task. Sentence puzzles inherently offer more detailed information, enabling models to detect and identify the same reasoning patterns more readily, regardless of changes in context, in contrast to word puzzles, which typically feature shorter contextual statements, presenting a greater challenge for models to discern consistent reasoning patterns.

Initially, it becomes apparent that pre-training encoders across various commonsense reasoning datasets results in substantial performance enhancements, as it enables the system to grasp domain-agnostic features which prove advantageous for the subsequent task. Additionally, several commonsense pre-trained encoders fine-tuned on BrainTeaser data outperform Llama 2 and Phi-2.

Another noteworthy observation from Table 5.4 is that only Mistral-7b from LLMs is able to surpass the encoder-type networks, while both Llama 2 and Phi-2 consistently scored lower.

Unlike Llama 2 and Mistral-7b, Phi-2 has not undergone instruction fine-tuning [26], which, coupled with the limited number of examples in the BrainTeaser Sentence Puzzle dataset, contributes to its lower performance,

System	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Overall
Multi-class classification problem						
Human	.907	.907	.944	.907	.889	.920
ChatGPT	.608	.593	.679	.507	.397	.627
RoBERTa-L	.435	.402	.464	.330	.201	.434
Mistral-7b_128_128	.850	.825	.775	.825	.700	.817
Mistral-7b_64_128	.850	.825	.775	.825	.700	.817
Mistral-7b_16_64	.800	.800	.850	.750	.725	.817
Mixtral-8x7b_128_128	.850	.825	.725	.800	.700	.800
Llama 2-7b_64_128	.725	.650	.700	.575	.475	.692
Llama 2-13b_64_64	.665	.614	.645	.550	.400	.641
Llama 2-7b_64_64	.625	.600	.675	.550	.400	.633
Llama 2-7b_64_32	.250	.250	.425	.075	.000	.308
Phi-2_64_128	.625	.575	.550	.525	.425	.583
Phi-2_128_128	.625	.575	.550	.500	.375	.583
Phi-2_64_64	.525	.425	.550	.375	.300	.500
RoBERTa-WNGRD	.800	.775	.775	.750	.675	.784
DeBERTaV3-TS	.800	.775	.725	.750	.625	.767
DeBERTaV3-base	.725	.750	.675	.725	.625	.717
BERT-SE	.750	.725	.650	.700	.550	.708
RoBERTa-large	.700	.700	.725	.675	.550	.708
BERT	.675	.650	.650	.600	.475	.658
Binary classification problem						
DeBERTaV3-TS	.725	.650	.550	.650	.650	.642
RoBERTa-WNGRD	.575	.600	.500	.550	.550	.558
BERT-SE	.625	.550	.375	.525	.525	.517

Table 5.4: Model Performance for *sub-task A: Sentence Puzzle*. More results in Table 5.7.

as a result of Phi’s incapability to capture the complexities of the BrainTeaser data.

In this regard, Mistral-7b, which has already demonstrated superior performance compared to every Llama 2 variation when tested in commonsense reasoning benchmarks [33], is also capable of solving this task more accurately.

Task B: Word Puzzle

In Table 5.5, we observe a stark contrast in the models’ performance in understanding and detecting reasoning paths when the context changes. There are notable discrepancies in accuracy between original and semantic contexts when compared to context reconstruction, particularly evident in the case of smaller encoder models.

Regarding encoders, it is evident that, especially vanilla RoBERTa-large lacks robust commonsense reasoning and struggles to systematically handle ambiguity; in contrast, RoBERTa-large pre-trained on WinoGrande presents competitive performance. This notable enhancement (over 40%) due to WinoGrande pre-training suggests that this particular dataset effectively equips the model with the ability to understand word puzzle-related reasoning complexities, making its scores competitive with DeBERTaV3 in this sub-task, despite the higher DeBERTaV3-base performance over RoBERTa-large in baseline reasoning benchmarks [29]. Other than that, pre-training on other commonsense reasoning datasets does not significantly improve the overall performance for encoders. Conclusively, apart from WinoGrande the rest of the extra pre-training datasets do not hold reasoning cues close to BrainTeaser’s word puzzles.

Regarding LLMs, Mistral-7b notably outperformed all others by a significant margin, even surpassing the 8 times larger model tuned using the same hyperparameters (*Mixtral-8x7b*). Llama 2 exhibited the worst results regardless of size (7/13 billion) and LoRA hyperparameters (r and a). Conversely, Phi-2 demonstrated relatively better performance, particularly considering its smaller parameter count (2.7 billion) compared to

System	Original	Semantic	Context	Ori.+Sem.	Ori.+Sem.+Con.	Overall
Multi-class classification problem						
Human	.917	.917	.917	.917	.900	.917
ChatGPT	.561	.524	.518	.439	.292	.535
RoBERTa-L	.195	.195	.232	.146	.061	.207
Mistral-7b_16_64	.875	.906	.781	.813	.719	.854
Mistral-7b_128_128	.844	.844	.813	.719	.625	.833
Mistral-7b_8_16	.781	.938	.781	.719	.562	.833
Mixtral-8x7b_128_128	.625	.719	.625	.531	.375	.656
Llama 2-13b_64_64	.354	.344	.438	.125	.031	.379
Llama 2-7b_64_64	.375	.344	.375	.125	.031	.365
Llama 2-7b_64_128	.281	.188	.438	.031	.031	.302
Phi-2_64_64	.688	.625	.688	.562	.438	.667
Phi-2_64_128	.656	.656	.625	.594	.406	.646
Phi-2_16_64	.625	.500	.688	.438	.312	.604
DeBERTaV3-base	.750	.750	.562	.656	.438	.687
DeBERTaV3-TS	.812	.781	.406	.719	.281	.666
RoBERTa-WNGRD	.750	.656	.500	.625	.312	.635
BERT	.562	.594	.469	.562	.312	.542
BERT-SE	.562	.500	.406	.500	.281	.489
RoBERTa-large	.281	.344	.344	.156	.094	.323
Binary classification problem						
DeBERTaV3-TS	.250	.313	.313	.219	.219	.292
RoBERTa-WNGRD	.375	.313	.125	.281	.281	.271
BERT-SE	.000	.000	.000	.000	.000	.000

Table 5.5: Model performance for *sub-task B: Word Puzzle*. More results in Table 5.7.

the other LLMs. However, both models performed worse compared to most fine-tuned encoders. This observation strongly confirms that word puzzles possess a distribution that diverges from the analytical commonsense reasoning required for sentence puzzles, entailing a unique set of cognitive demands.

Mistral-7b exhibits a trend where higher quality explanations were generated with higher values of lora rank r . However, the top-performing model showcased a configuration with $r=16$ and $a=64$. The QLoRA method [31] explains why our top model has a rank of 16 instead of 128, contrary to common expectations (more details regarding QLoRA hyperparameters in App. 5.3). Drawing from the widespread presence of low-rank structures, as highlighted by prior studies [45, 46, 23], we leverage the intrinsic low-rank structure in our problem, as emphasized in [31]. It is well-established that many tasks, particularly involving heavily over-parametrized models, exhibit low-rank properties post-training [58].

Overall, our systems demonstrate remarkably high overall accuracy, being less than 10% lower than human performance and more than 30% greater than ChatGPT. This suggests our methods’ proficiency in understanding and detecting wordplay patterns, consistently addressing ambiguity irrespective of contextual and semantic variations in brain teasers. Upon reviewing the short explanations provided with each prediction (Table 5.6), we note thorough justifications even for incorrect answers. Errors typically adhere to specific wordplay patterns across original, semantic, and context multiple-choice questions (details in 5.4.2).

5.4.2 Assessment and Insights on Dataset Quality

Upon reviewing our incorrect predictions across both sub-tasks, subsequent to the task organizer releasing the labels for the test split of the datasets, we reached several conclusions. Across all triplets, encompassing original, semantic, and context reconstruction statements, we observe a considerable degree of ambiguity in various patterns. This ambiguity often leads to inconsistent selection of correct answers, even when answered

<i>Example of correct prediction with explanation</i>	
Question	Choices
If a dog is tied to a piece of rope that is 6m long how can it reach a bone that is 7m away?	The other end is not tied to anything.
	The rope stretches proportionally, providing the extra length needed for the horse to reach the hay seven meters away.
	The dog dug a 1m tunnel underground to reach the bone that was 7m away while still tied to the 6m long rope.
	None of above.
Explanation: A dog cannot reach a bone that is 7m away if it is tied to a piece of rope that is 6m long. This is because the distance between the dog and the bone is greater than the length of the rope. Therefore, the dog will not be able to reach the bone unless the rope is untied.	
<i>Example of misprediction with explanation</i>	
Question	Choices
How can you drop a raw egg onto a concrete floor and not crack it?	You need to drop the egg gently.
	You need to drop the egg hard.
	No problem, because concrete floors are very hard to crack.
	None of above.
Predicted Answer: None of above.	
Explanation: There is no way to drop a raw egg onto a concrete floor and not crack it. Concrete is a very hard material and will easily break any object that falls on it. The only way to avoid this would be to use some sort of protective covering for the egg before dropping it.	

Table 5.6: An example of explanations generated by Mistral-7b, while addressing the multi-class classification problem. The explanations, regardless of their validity, are detailed and showcase a high level of comprehension and reasoning. Correct answers are indicated in bold, while mispredictions reference the predicted answer by the model.

by humans. This underscores the need for clearer formulation of questions and unambiguous expression to enhance the accuracy of model predictions. Another notable pattern we identified pertains to the quality control of semantic reconstruction in certain questions. In these instances, some words were not replaced with accurate synonyms, resulting in a shift in the definition of the brain teaser presented by the question. While this may not inherently be problematic, the dataset’s correct answers remained unchanged compared to the original version of the question. This discrepancy suggests that the alteration in question definition went unnoticed by the task organizers, leading to some erroneous predictions by our model, when in reality the correct context of the provided multiple-choice statement was captured by our system. The two observations above highlight the inherent difficulty in generating clear and precise brain teasers, as well as the challenge that models face in understanding them. In the above scenarios, our top-performing model either detects the presence of a contradiction in the questions and opts to select "None of above," as elucidated in its brief and explanatory justification, or it provides an incorrect answer based on the dataset’s answer but correctly reflects the problem context, which may have been altered due to inadvertent synonym usage.

5.4.3 Analysis of LoRA Hyperparameters

Table 5.7 depicts further analysis of LoRA hyperparameters for Mistral and Mixtral models, which have exhibited the best results among all other models and across the two tasks. Due to computational restrictions, we trained the Mixtral model, which is eight times larger, only for the best performing hyperparameters of Mistral, as a proxy for the performance difference.

System	Original	Semantic	Context	Ori. + Sem.	Ori. + Sem. + Con.	Overall
Task A						
Mistral-7b_64_128	.850	.825	.775	.825	.700	.817
Mistral-7b_16_64	.800	.800	.850	.750	.725	.817
Mixtral-8x7b_128_128	.850	.825	.725	.800	.700	.800
Mistral-7b_128_64	.850	.800	.725	.775	.625	.792
Mistral-7b_64_32	.850	.775	.725	.750	.675	.783
Mistral-7b_8_16	.800	.800	.700	.750	.625	.767
Mistral-7b_128_32	.825	.775	.725	.750	.600	.775
Task B						
Mistral-7b_128_128	.844	.844	.813	.719	.625	.833
Mistral-7b_8_16	.781	.938	.781	.719	.562	.833
Mistral-7b_16_16	.812	.812	.875	.688	.625	.833
Mistral-7b_8_8	.875	.812	.812	.750	.688	.833
Mistral-7b_16_32	.875	.812	.781	.750	.594	.823
Mistral-7b_64_32	.844	.875	.719	.750	.562	.812
Mistral-7b_128_64	.844	.812	.781	.688	.531	.812
Mistral-7b_64_64	.719	.812	.625	.625	.406	.719
Mixtral-8x7b_128_128	.625	.719	.625	.531	.375	.656

Table 5.7: The performance of various LoRA hyperparameters for Mistral and Mixtral in both sub-tasks.

Chapter 6

Proposal 2 - RISCORE

Consider the following two riddles: *R1*: “A man shaves every day, yet keeps his beard long” and *R2*: “What has a beard but never needs to shave?”. While these riddles are semantically similar, follow the same structure, and refer to the same objects, their reasoning processes differ. In *R1*, where the answer is “A barber,” the word “beard” is used literally. In *R2*, where the answer is “A tree,” “beard” is used metaphorically, referring to the “beard” of certain trees, such as the “oak”. Thus, when using these riddles as context for a new riddle—such as “I plant seeds every day, yet don’t have a single plot”—the answer will depend on the interpretation of the phrase “plant seeds.” A metaphorical interpretation might lead to “A teacher,” while a literal one might suggest “A hydroponicist.” This illustrates how riddle context can significantly shape the reasoning process required for solving it.

In this example, however, instead of using *R2*, a contextually reconstructed version of *R1*—*R3*: “Tom attends class every day but doesn’t do any homework”—would provide a clearer reasoning process to the model. Although *R2* is semantically closer to *R1*, its metaphorical use of “beard” contrasts with the literal meaning in the original, potentially obscuring the reasoning pathway that the model should follow for solving a new riddle.

Building on the previous example, we introduce RISCORE (**R**iddle **S**olving with **C**ontext **R**econtruciton), a novel fully automated prompting method that generates and utilizes contextually reconstructed sentence-based puzzles in conjunction with the original examples to create few-shot exemplars, designed to enhance the in-context riddle-solving abilities of LLMs. The RISCORE method supplements each exemplar in FS learning with a contextually reconstructed version of itself. This preserves the original reasoning process while only altering its context. This approach enables the model to delineate a clear and coherent reasoning trajectory, which it can follow to effectively solve new riddles.

As illustrated in Figure 6.0.1, our approach builds on top of existing few-shot (FS) methods [16, 85, 76], with RISCORE not involved in the FS selection process. The goal of our method is to **augment FS samples** with automatically generated context-reconstructed examples (detailed in Section 6.2). The addition of context reconstructed examples has, in most cases, proven to be more beneficial for the models’ performance, surpassing even the utility of real examples extracted from the dataset (see Section 7.3.4).

This proposal has also been published as a draft and is currently under review for acceptance at a major computational linguistics (CL) conference [60].

6.1 Contributions

To sum up our contributions are:

- We experimentally verify that providing a riddle along with its context reconstruction can enhance performance on both lateral and vertical thinking problems.
- We propose RISCORE, a novel prompting method designed to enhance the in-context riddle-solving

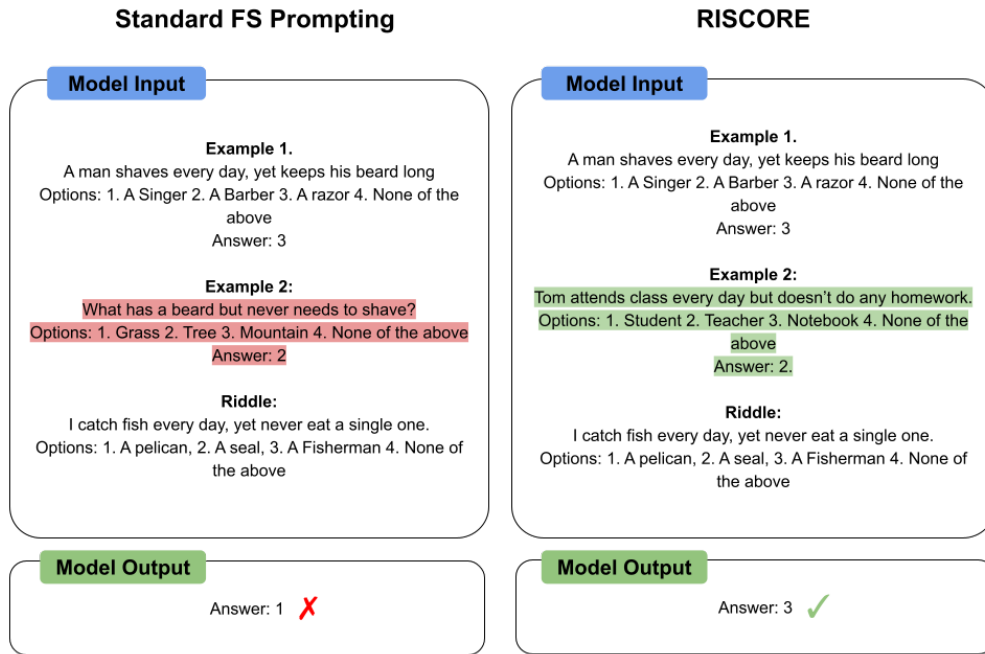


Figure 6.0.1: An example of few-shot (FS) prompting alongside the same method using a context-reconstructed example. The text in green represents the context reconstruction of Example 1, produced by the proposed method. The example in red was removed simply to maintain the same number of shots.

capabilities of LLMs. As a supplement to this, we introduce an algorithm for generating contextually reconstructed multiple-choice riddles.

- We compare RISCORE against a wide array of popular prompting techniques, highlighting its effectiveness across several prompting alternatives and models of varying sizes.

6.2 Method for generating context reconstructions

In this section, we describe our approach for generating high-quality reconstructed Question-Answer pairs, which can be used as input to a model, alongside their counterparts, to enhance performance on lateral thinking problems. This process draws inspiration from the semi-automated pipeline introduced in the BrainTeaser paper [38], but we extend it by fully leveraging the capabilities of LLMs to automate the procedure.

6.2.1 Step 1: Generation of Question-Answer pair

The first step involves generating one contextually reconstructed Question-Answer pair per selected instance, temporarily ignoring distractors. To achieve this, we provide the LLM with the riddle, the correct answer, and a system prompt that outlines the task. The prompt instructs the model to analyse the given Question-Answer pair, understand the riddle, identify the reasoning process that links the question to the answer, and then generate a similar riddle that follows the same reasoning process, along with a corresponding correct answer. This approach maintains the underlying commonsense premise of the original question while altering both the question and the answer to fit a new situational context. Since the model is provided with both the riddle and the correct answer, understanding the riddle becomes easier than solving it independently [76]. To further refine this approach, we apply the method in both Zero-shot (ZS) and Few-shot (FS) settings, using preexisting pairs of original and contextually reconstructed questions from BrainTeaser.

In a zero-shot setting, the model generates the new pairs without prior examples or training specific to the

task. In the few-shot setting, we use the same instructions as in the zero-shot approach, but we enhance the model’s understanding by providing in-context examples. Specifically, we include a pair consisting of an original Question-Answer and its corresponding contextually reconstructed version. This helps the model better grasp the task at hand. It is important to note that these few-shot examples are of high quality, given their manual construction and careful curation. In the few-shot approach, the provided examples are static. It is important to highlight that, in BrainTeaser, when selecting examples for few-shot exemplars from the training dataset, we ensure that the examples chosen are not selected for reconstruction. This avoids any overlap between the few-shot exemplars and the examples for which we are requesting a context change, ensuring that the given pair examples are distinct from the requested ones. The prompts used in this process are presented in 6.2.1.

After generating the Question-Answer pair, a filtering procedure was conducted to ensure the quality of the instance and its alignment with the riddles in the dataset. This process involved applying rules concerning both the structure of the riddle and the answer. These rules are dataset-specific and can be adjusted to suit different datasets, ensuring adaptability across various contexts.

A future direction could involve using Graph Neural Networks (GNNs) to generate efficient text counterfactuals [56]. After generating Question-Answer pairs, a GNN would model the semantic relationships between elements of the riddle (nodes/edges) and identify minimal changes that still preserve the reasoning process. This ensures optimal counterfactual generation, applied in both Zero-shot and Few-shot settings. The GNN would enhance efficiency by altering only necessary components while keeping the logic intact, followed by a filtering step to ensure alignment with dataset-specific rules.

Quality Assurance for Generated Text

As mentioned above, after the generation of the contextual reconstructed Question-Answer pairs, we perform a small but necessary quality control to ensure that the generated pairs adhere to several high-level criteria. One potential area for further exploration is the detection and analysis of **hallucinations** in the generated content. Hallucinations occur when models generate information that is not logically aligned with the input or context, which could affect the quality and coherence of the Question-Answer pairs. Given the growing awareness of this issue, it may be valuable to integrate hallucination detection mechanisms into the quality control process. Several similar works have been investigating techniques to identify and mitigate hallucinations in language models [24], and their findings could inform the development of more robust detection strategies.

For the BrainTeaser dataset, the filtering process is more flexible, as the questions and answers can vary in length. The only constraint is that the questions must have a minimum length of 7 words, while the answers can be as long as necessary. As a result, the number of generated pairs filtered out is relatively low, around 2%, ensuring that the majority of reconstructions meet the quality criteria.

In contrast, the RiddleSense dataset requires stricter limitations due to its typically shorter answers. For this dataset, questions must be at least 6 words in length, while answers must not exceed 7 words, reflecting the dataset’s concise nature. As a result of these stricter rules, approximately 10% of the generated reconstructions for the RiddleSense dataset are filtered out for not meeting the required standards.

It is important to note that the minimum question length requirement is implemented to ensure that the generated questions maintain a sufficient level of complexity and depth. Questions with fewer than 6 or 7 words are unlikely to provide the necessary detail to establish a well-structured and coherent riddle. This limitation helps preserve the quality and integrity of the riddles by ensuring that they are adequately framed to challenge and engage the model effectively.

Models utilized

In this approach, we experiment with two models to assess the impact of model size on the quality of contextual reconstructions and the overall performance across both datasets. We use a relatively small model, Llama3-8B [1], and a larger model from the same family, Llama3-70B [1]. These models were selected due to their strong performance across a diverse range of tasks, outperforming other models of comparable size. By incorporating both a smaller model, Llama3-8B, and a larger model, Llama3-70B, we aim to systematically investigate the effect of model size on the quality of contextual reconstructions and its overall impact on performance. This

exploration allows us to gain deeper insights into how model capacity influences the effectiveness of the generated examples across different datasets.

Prompts

First, we will present the zero-shot prompt configuration used for generating the contextually reconstructed Question-Answer pairs.

System Prompt:

You are an expert in context reconstruction. Your task is to receive a question along with its correct answer and adapt them to a new scenario while maintaining the misleading commonsense premise.

Please follow these steps:

- First, you will receive an unsolved riddle along with five answer options. Analyze the given setting and identify the connection between the question and its correct answer.
- Modify the original question and correct answer to fit a different situational context, ensuring that the underlying logic and relationship between them are preserved.
- Ensure that both the new question and the new correct answer are distinct from the originals.

User Prompt:

Question: ```{QUESTION}```

Correct answer: ```{ANSWER}```

Next, with minor modifications, we introduce the few-shot prompt configuration that follows the same logic but incorporates in-context examples to guide the model's responses.

System Prompt:

You are an expert in context reconstruction. Your task is to receive a question along with its correct answer and adapt them to a new scenario while maintaining the misleading commonsense premise.

Please follow these steps:

First, review an example provided with its context reconstruction, which illustrates the type of transformation you will need to perform.

Next, you will receive an unsolved riddle along with five answer options. Analyze the given setting and identify the connection between the question and its correct answer.

Modify the original question and correct answer to fit a different situational context, ensuring that the underlying logic and relationship between them are preserved.

Ensure that both the new question and the new correct answer are distinct from the originals.

User Prompt:

{EXAMPLES}

Adapt the following riddle - answer pair while taking into consideration the examples above regarding context reconstruction:

Question: ```{QUESTION}```

Correct answer: ```{ANSWER}```

In the above setting, it is understood that the values of the **EXAMPLES** represent pairs of answered Question-Answer examples: the original and its contextually reconstructed counterpart, both sourced from the BrainTeaser dataset.

6.2.2 Step 2: Generation of the distractors

This process involves generating incorrect answers to complement the multiple-choice options. While this task may initially appear straightforward, it presents several significant challenges. First, the distractors must not only match the number of original options but also be guaranteed to be incorrect when compared to the correct answer. Additionally, they should not deviate excessively from the correct answer, as significant divergence could undermine the validity and challenge of the riddle. It is also essential to account for cases where the correct answer is “None of the above”, as observed in the BrainTeaser dataset. The length of the answers also plays a crucial role in this process. For instance, BrainTeaser includes mostly answers with more than four or five words, while RiddleSense provides typically single-word answers. These variations in length introduce additional challenges in crafting effective distractors. To address this, we propose two distinct methods for generating distractors: one for long and another for short distractors, respectively.

Generation of Long Distractors

In this process, we leverage the capabilities of large language models (LLMs) to understand and rephrase context in order to generate distractors. We use two different pipelines to produce at least three distractors, ensuring coverage of various potential situations.

The first approach involves prompting the model with a system-user prompt that instructs it to analyse the given Question-Answer pair. The model is tasked with understanding the riddle, identifying the reasoning process that links the question to the answer, and then suggesting a distractor based on the more challenging or deceptive aspects of the concept. This approach yields one of the three required distractors.

In the second approach, the process is more intricate and can result in distractors of questionable quality. We prompt the model with a system-user prompt, providing it with the reconstructed question without its answer, along with the original question’s incorrect distractors, after removing the option “None of the above”. The model is then tasked with modifying the concept of the given distractors by incorporating elements from the setting described in the question. Importantly, the correct answer is not provided to ensure that the generated distractors are distinctly different from the correct answer and effectively capture a varied interpretation of the question’s context. Certainly, there are instances where the model generates sub-optimal contexts for the distractors provided. Despite these cases, the distractors remain incorrect and serve their purpose, though they may not always be sufficiently challenging.

Our experiments indicate that when the model is given both the original and the contextual reconstructed examples, the minor issue of lower-quality distractors does not significantly impact overall performance. Now, we have also created two new distractors that are somewhat contextually relevant to the setting, adding an additional layer of coherence to the generated options. This ensures that the distractors are not only incorrect but also related to the underlying premise of the riddle, enhancing the overall quality of the multiple-choice options. To create the final dataset, we randomly select two of the three generated distractors and shuffle them with the correct answer in random order. Finally, we append the option “None of the above” as the last choice. In cases where “None of the above” is the correct answer, we use all three generated distractors. With this approach, our dataset is prepared and ready for use.

Prompts used First, we will provide the system-user prompt used to task the model with understanding the riddle, identifying the reasoning process linking the question to the answer, and then suggesting a distractor based on the more challenging or deceptive aspects of the concept.

System Prompt:

Your task is to act as a concept grasper. You will be given a riddle and its correct answer.

Your goal is to understand the connection between the riddle and the correct answer, focusing on the tricky

parts. Based on these tricky aspects, propose a plausible wrong answer that someone might give.

The wrong answer should be short, concise, and limited to one sentence.

- Riddle:

- Correct Answer:

Response format:

- Wrong Answer:

User Prompt:

- Riddle: {QUESTION}

- Correct Answer: {ANSWER}

Now we will provide the system-user prompt used for the second method.

System Prompt:

You will be given a sentence without context and then provided with a specific context.

Your task is to rewrite the sentence so that it aligns with the given context, while keeping it as close as possible to the original meaning.

The purpose is to adapt the sentence to the context, not to answer any questions related to the context.

- Sentence (out of context):

- Context:

Response format:

- Sentence:

User Prompt:

- Sentence (out of context): {ORI_CHOICE}

- Context:{QUESTION}

Here, the value of **ORI_CHOICE** refers to the distractors from the original instance, excluding the "None of the above" option. We create prompts for the model with each of these distractors individually.

Generation of Short Distractors

In this setting, the approach is fundamentally different. The answers and distractors in the original dataset are primarily one-word responses, while the questions feature detailed settings with punctuation, conjunctions, and more complex structures. To handle this, we also generate distractors using two distinct pipelines tailored to this format.

First pipeline The first pipeline involves a more granular approach by splitting the contextual reconstructed question into subphrases based on punctuation or conjunctions. If this yields fewer than three distinct subphrases, we further split the sentence at the position of the word “and” to create additional segments. We also detect the presence of interrogative words, and if the question is purely descriptive without any direct question, we append the phrase “What am I?” at the end. This is not chosen arbitrarily but follows

the standard structure of many riddles in this dataset, where "What am I?" is the common question leading to single-word answers.

Now that we have broken the riddle into sub-phrases and appended the appropriate question, we can prompt the model to generate a wrong answer for each sub-phrase concatenated with the question. This method ensures that the distractors align with different parts of the riddle’s setting. However, since some sub-phrases may contain key ideas central to the riddle, the model may still generate answers too similar to the correct one. To address this issue, we incorporate an additional intermediate step utilizing zero-shot classification. Specifically, we use the *facebook/bart-large-mnli* model¹ [43] provided by Hugging Face. This model classifies the correct answer into one of eight general categories: *'food'*, *'person'*, *'object'*, *'animal'*, *'nature'*, *'time'*, *'place'*, *'concept'*. These categories are chosen to be mutually exclusive to avoid overlap.

In our approach, we first predict the category of the correct answer using this model. We then leverage this classification to guide the generation of distractors. For each sub-phrase concatenated with the question, we provide the LLM with the two most similar categories (excluding the correct answer’s category). The model is prompted with a system-user instruction to produce a correct answer of the given setting in the given category. This method ensures that the generated distractors are not only incorrect but also contextually aligned with the riddle, thus maintaining the challenge for the model. After applying filtering to ensure the distractors are distinct and relevant, we have produced several distractors. The LLMs utilized in this pipeline include the previously mentioned Llama3-8B [1] and its larger counterpart, Llama3-70B [1].

Second pipeline In cases where the distractors produced by the above pipeline are insufficient, we use WordNet [11] to augment our distractor set. For each generated distractor, or if necessary, for the distractors from the original question, we retrieve synonyms and hyponyms from WordNet. These additional terms are then included as potential distractors.

Selection of Final Distractors After compiling these distractors, we randomly select four of them and add the correct answer in a random order to complete the set. This approach ensures that we have a diverse and comprehensive set of distractors for each question.

To ensure the quality of our distractors, we impose a limitation that at least two of the required four distractors must be generated using the first approach. This is due to the fact that the distractors generated through WordNet augmentation tend to be of inferior quality compared to those produced directly by the model. If this requirement is not met, we skip the particular train set instance for producing a contextual reconstruction. This approach helps maintain the overall quality and relevance of the distractors in our dataset.

Prompts used We will present the system-user prompt configuration used in the first pipeline. In this approach, for each sub-phrase concatenated with the question, we provide the model with two categories, excluding the correct answer’s category. For each category, a separate system-user prompt is issued to instruct the model to generate a correct answer that fits the given setting within the specified category.

System Prompt:

Task: Provide a concise, relevant answer to the given question within the specified category.

Constraints:

- The answer should not exceed three words.
- Follow the exact format provided below.

Response Format:

Answer: ...

User Prompt:

Question: ```{QUESTION}```

¹[facebook/bart-large-mnli](https://huggingface.co/facebook/bart-large-mnli)

Category: {CATEGORY}

6.2.3 Step 3: Creation of the Riddle

We have successfully created the desired contextual reconstructions for both datasets. However, an issue remains: some of the originally selected examples do not have corresponding reconstructed examples due to the quality control filtering process.

To address this issue, we first use the original most semantically similar examples as in-context learning exemplars, appending their corresponding automatically generated contextual pairs. If these exemplars are not sufficient for the required settings of two, four, or eight (i.e., **RISCORE**), we employ a more structured approach rather than adding examples randomly.

We begin by generating embeddings for the set of original examples and their contextual reconstructions that have not already been included in the current exemplars. Using cosine similarity, we then identify the most similar examples from this set. Importantly, these similar examples might not be part of the original training set but could be among the reconstructed examples.

We select the most similar examples and pair each with its corresponding pair, ensuring that we add either the original or the reconstructed example as needed. This process is repeated until the number of exemplars meets the required quantity.

Chapter 7

Experiments - RISCORE

The experiments were designed to evaluate the effectiveness of RISCORE in enhancing the reasoning capabilities of language models across both lateral and vertical thinking tasks. Specifically, we aimed to assess whether incorporating contextually reconstructed riddles as few-shot exemplars alongside their original counterparts would improve model performance.

The experiments addressed two core research questions:

- **RQ1**: Does the use of contextually reconstructed instances enhance a model’s ability to solve lateral thinking puzzles when compared to standard prompting techniques?
- **RQ2**: Can the automated generation of these reconstructed riddles be utilized as an effective prompting strategy, boosting performance across both lateral and vertical reasoning tasks?

To explore these questions, we tested a range of language models, comparing RISCORE to traditional few-shot prompting strategies and examining its impact on reasoning performance. The experiments involved using reconstructed riddles as exemplars to emphasize reasoning patterns rather than the surface-level semantics of the data.

Our experimental setup included varying the number of few-shot exemplars, as well as testing across different reasoning task complexities, to ensure robust comparisons between RISCORE and other methods.

7.1 Preliminaries

7.1.1 Dataset

In this work we are using two different datasets of multiple QA format.

The first one is the BrainTeaser dataset, which consists of two sub-tasks regarding both lateral thinking challenges. In addition to the original puzzles, the dataset includes adversarial subsets created by manually modifying the original BrainTeaser while preserving their reasoning paths. The original data were perturbed in two ways: First, there is a *semantic reconstruction* of each original question, ensuring that the answers and distractors remain unchanged. Second, the original data undergoes *context reconstruction*, where the reasoning path is preserved, but the brain teaser is rephrased to describe a new situational context. Each question offers four possible choices, with the last option always being “None of the above”. It is important to note that our experiments are focused exclusively on the sentence puzzle (SP) setting.

The second dataset used in our experiments is RiddleSense. This dataset features riddle-style questions that require sophisticated commonsense reasoning and a strong grasp of figurative language to answer accurately. It is structured as a multiple-choice question answering task, with riddles that test the model’s capacity to navigate and interpret nuanced commonsense scenarios. The reasoning challenges posed by RiddleSense are closely aligned with those found in BrainTeaser. RiddleSense offers one correct answer and four distractors for each riddle. This contrasts with BrainTeaser, which includes a “None of the above” option as the final choice, thereby introducing a more challenging setting.

Data statistics

Format of Datasets The BrainTeaser dataset is divided into three splits: train, development, and a hidden test set used for evaluation. Although RiddleSense follows a similar format, we did not have direct access to its hidden test set. As a result, we utilized the development data split of RiddleSense for our experiments.

Prepossessing and filtering Due to data leakage across the two datasets, we were required to exclude semantically similar examples from the test split utilized. This leakage is understandable, as the sources for English riddles are limited, and the number of unique riddles is finite. After identifying semantically similar riddles, we retained only one instance within the datasets, prioritizing BrainTeaser due to the higher quality of its questions and distractors, which were manually crafted.

Our goal in this process is to ensure that no question from the RiddleSense test set is present in the combined train, dev, and test datasets of BrainTeaser. To achieve this, we are using the Sentence Transformers library¹ [66] with the *gte-large-en-v1.5*² [99] model to generate text embeddings and perform semantic similarity comparisons. We convert each question from the entire BrainTeaser puzzle into embeddings that represent the semantic meaning of each question. We then calculate the cosine similarity between these embeddings and those in the RiddleSense test set to identify duplicates or highly similar questions. A lower threshold is applied to define similarity, meaning that any question with a cosine similarity above this threshold is considered a near-duplicate and excluded from the test set.

After conducting several experiments, we determined that a cosine similarity threshold of 0.9 was optimal. This threshold is a delicate balance, as our questions are riddle-style and often rely on abstract or metaphorical language, making it challenging to clearly assess whether two questions are similar enough to warrant exclusion. Since riddles are not always straightforward in their wording, the threshold needed to be fine-tuned to avoid excluding questions that are conceptually distinct but might have overlapping language, ensuring a fair evaluation without overly aggressive filtering. This means that any question from the RiddleSense test set that had a similarity score of 0.9 or higher to a question in the BrainTeaser datasets was removed to avoid redundancy. As previously mentioned, because we prioritise BrainTeaser questions due to their superior quality from manual annotation, we decided to remove any similar questions from the RiddleSense test set. This approach helps preserve the integrity of the evaluation process and ensures that the test set contains only unique and distinct questions, minimising potential data leakage and improving the robustness of our

¹SentenceTransformers

²Alibaba-NLP/gte-large-en-v1.5

results. Additionally, by removing these instances, we can perform more experiments on the remaining unique questions, optimising our use of hardware resources and potentially yielding more comprehensive insights.

Additionally, hardware limitations made processing the full RiddleSense unfeasible. To manage this, we split its test set into two halves. To ensure the fairness and accuracy of this division, we evaluated various techniques on both halves. Following the completion of these procedures, we present the dataset statistics in the table below. Statistics regarding the datasets are provided in Table 7.1.

Dataset	Train	Dev	Test
BrainTeaser - SP	507 _(169x3)	120 _(40x3)	120 _(40x3)
RiddleSense (initial)	3510	1021	—
RiddleSense (filtered)		720	
RiddleSense (sampled 50%)		360	

Table 7.1: Data statistics

7.1.2 Evaluation metrics

Given that our task is formatted as a multiple-choice question-answering problem, the evaluation metric employed will be accuracy. This metric will provide a measure of the proportion of correctly answered questions out of the total number of questions presented. Although accuracy is a straightforward metric, it is effective in providing useful insights and serves as a reliable measure for evaluation in this context.

BrainTeaser For the BrainTeaser task, we not only track the overall accuracy but also monitor the accuracy for each specific type of instance, including original, semantic, and context reconstructions. This detailed tracking is feasible because the data in each set are balanced, with each original instance having corresponding semantic and context variations. This approach allows us to evaluate the model’s proficiency in handling the same reasoning path under different conditions. The total accuracy is then calculated as the mean of these three individual metrics. We also track group-based accuracy, where a "group" refers to either two questions (original and semantically reconstructed) or three questions (original, semantically reconstructed and contextually reconstructed). This metric assesses the model’s performance when all instances within a group are answered correctly. Group-based accuracy provides a broader perspective on the model’s ability to handle these lateral thinking challenges across different types of question reconstructions. This format of evaluation metrics was both provided and requested by the creators of the dataset in an open Kaggle competition, and we are adopting it for our analysis.

RiddleSense Unfortunately, the simple data row format in RiddleSense restricts our ability to track more detailed metrics comprehensively, limiting us to evaluating only the overall accuracy for this dataset.

7.1.3 Baselines

In our study, we used various prompting techniques as baselines, including zero-shot (**ZS**), few-shot (**FS**), and Chain of Thought (**CoT**) methods.

Chain-of-thought (CoT_ZS) Prompting Technique

This is the most straightforward prompting technique. In this approach, each instance from the test set of each dataset is presented to the model using a specific system-user prompt format. This technique operates in a zero-shot setting, meaning the model is not given any prior examples or training specific to the task at hand. Instead, it relies solely on the system prompt and user prompt to generate responses based on its pre-existing knowledge. The chain of thought character is a result of the usage of the phrase *Let’s think step by step* in the end of each system prompt. The Chain of Thought (CoT) character is achieved by including the phrase “*Let’s think step by step*” at the end of each system prompt. This approach aligns with the findings of [40], which demonstrate that large language models can perform effectively as zero-shot reasoners by incorporating this specific phrase before presenting an unanswered test instance. The following are the system-user prompts that were used:

System Prompt:

You will encounter a riddle that requires analytical thinking and reasoning to solve.

A riddle is a question or statement intentionally phrased so as to require ingenuity in ascertaining its answer or meaning, typically presented as a game.

Different ideas can be used in these riddles:

1. Riddles often employ misdirection, leading you away from the actual solution.
2. They include elements with double meanings, requiring a keen eye for words with dual interpretations.
3. Metaphorical wordplay adds another layer, urging you to decipher figurative language.
4. Look out for exaggeration, as riddles may present overly dramatic details to divert your attention.
5. Common phrases and sayings may hide within the puzzle, demanding familiarity.
6. Associations and irony play a crucial role, introducing unexpected connections.
7. Numerical puzzles can also be part of the mystery, requiring you to decode their significance.
8. Elemental imagery, drawn from nature, might hold key descriptors.
9. Rhyming and sound clues can add a poetic dimension.
10. Word Puzzles: Pay attention to anagrams, acrostics, and other wordplay elements.
11. Also, it is important to note you should decode the upcoming riddle using everyday logic and creativity.

Approach the riddle with everyday logic and creativity, avoiding supernatural explanations.

You will be given an unsolved riddle and five options to choose the answer amongst them. Let's think step by step.

User Prompt:

```
Riddle: ```
{RIDDLE}
```

Options:
[option 1]: ```{OPTION_1}```
[option 2]: ```{OPTION_2}```
[option 3]: ```{OPTION_3}```
[option 4]: ```{OPTION_4}```
[option 5]: ```{OPTION_5}```
```

---

Where the input parameters are self-explanatory. It is worth mentioning that for BrainTeaser, where the multiple-choice options are four, the prompt is adjusted accordingly.

**Few-Shot (FS) Prompting Techniques**

In this approach, we provide a specific number of exemplars from each dataset's training set before asking the model to solve a multiple-choice question from the test set. These exemplars, which include their correct answers, serve as in-context learning examples. By presenting these examples, we aim to guide the model and improve its performance on the subsequent unanswered tasks. This method leverages the provided context to enhance the model's ability to understand and solve new instances more effectively. The system prompt that is used here is the one of the zero-shot prompting with some minor changes. The number of examples that are used are 2, 4 and 8 as mentioned in section 6.

Now, a key decision is how to select the exemplars that will be provided to the model before it attempts to answer the test set questions. The simplest and most straightforward approach is to randomly choose these answered exemplars from the training dataset. This method, referred to as **FS Rand** in our experiments, offers a baseline for comparison, as it doesn't apply any specific strategy for selecting the most relevant examples and relies purely on random sampling.

Another approach implemented in our experiments for selecting exemplars is based on semantic similarity. Specifically, we utilize the Sentence Transformers library <sup>3</sup> [66] in conjunction with the *gte-large-en-v1.5* model <sup>4</sup> [99]. This setup allows us to generate text embeddings and perform semantic similarity comparisons

---

<sup>3</sup>[SentenceTransformers](#)

<sup>4</sup>[Alibaba-NLP/gte-large-en-v1.5](#)

using cosine similarity between these embeddings. For each experiment, we first generate text embeddings for all instances in the training datasets. Then, for each question in the test set, we generate its embedding and calculate its cosine similarity with all training set embeddings. Based on the number of in-context learning examples needed, we select the most similar training instances to use as exemplars. This method ensures that the selected examples are semantically relevant, aiming to improve the model’s performance by presenting it with contextually aligned training examples. This approach is referred to as **FS Sim** in our experiments, as it focuses on semantic similarity to guide exemplar selection for improved task performance.

The system and user prompts used for both of the above settings are outlined as follows:

---

**System Prompt:**

You will encounter a riddle that requires analytical thinking and reasoning to solve.

A riddle is a question or statement intentionally phrased so as to require ingenuity in ascertaining its answer or meaning, typically presented as a game.

Different ideas can be used in these riddles:

1. Riddles often employ misdirection, leading you away from the actual solution.
2. They include elements with double meanings, requiring a keen eye for words with dual interpretations.
3. Metaphorical wordplay adds another layer, urging you to decipher figurative language.
4. Look out for exaggeration, as riddles may present overly dramatic details to divert your attention.
5. Common phrases and sayings may hide within the puzzle, demanding familiarity.
6. Associations and irony play a crucial role, introducing unexpected connections.
7. Numerical puzzles can also be part of the mystery, requiring you to decode their significance.
8. Elemental imagery, drawn from nature, might hold key descriptors.
9. Rhyming and sound clues can add a poetic dimension.
10. Word Puzzles: Pay attention to anagrams, acrostics, and other wordplay elements.
11. Also, it is important to note you should decode the upcoming riddle using everyday logic and creativity.

Approach the riddle with everyday logic and creativity, avoiding supernatural explanations.

First, you’ll encounter **X** examples with their answer provided similar to the riddle you will need to solve.

Then you will be given an unsolved riddle and **X** options to choose the answer amongst them.

**User Prompt:**

{EXAMPLES}

Answer the following riddle while taking into consideration the examples above. Choose the best and the most logical option from the available choices:

Riddle: ```  
{RIDDLE}  
```

Options:
[option 1]: ```{OPTION_1}```
[option 2]: ```{OPTION_2}```
[option 3]: ```{OPTION_3}```
[option 4]: ```{OPTION_4}```
[option 5]: ```{OPTION_5}```

Where the **EXAMPLES** refer to the few-shot examples selected for each test set instance, and the other input parameters are self-explanatory. It is worth mentioning that for BrainTeaser, where the multiple-choice options are four, the prompt is adjusted accordingly.

Few-shot with CoT Explanations (CoT_FS) Prompting Technique

Another promising technique for enhancing model performance across various tasks is Chain of Thought (CoT) prompting. This approach involves not only providing answered examples but also including an explanation consisting of intermediate reasoning steps. By laying out the thought process step-by-step, CoT prompting significantly improves the capability of large language models to tackle complex reasoning tasks. In our task, the Chain of Thought (CoT) prompting technique is anticipated to outperform the previously mentioned methods. This expectation is supported by the performance improvements observed in the CommonsenseQA dataset [77], a domain similar to our setting that involves both lateral and vertical thinking. The dataset demonstrated significant improvements when using a manually composed set of two, four, or eight few-shot exemplars with CoT prompting, highlighting its effectiveness in eliciting successful reasoning. In our case, these explanations were generated manually to ensure they align with human perceptions of the reasoning process required by the riddle. The explanation format we used follows the structure outlined by [87] for few-shot exemplars in full chain-of-thought prompts for CommonsenseQA. Due to the similarity in the multiple-choice question-answer format of our datasets, it served as a good foundation for developing our own approach of creating explanations. Below, we provide the system and user prompts used in the aforementioned technique:

System Prompt:

You will encounter a riddle that requires analytical thinking and reasoning to solve.

A riddle is a question or statement intentionally phrased so as to require ingenuity in ascertaining its answer or meaning, typically presented as a game.

Different ideas can be used in these riddles:

1. Riddles often employ misdirection, leading you away from the actual solution.
2. They include elements with double meanings, requiring a keen eye for words with dual interpretations.
3. Metaphorical wordplay adds another layer, urging you to decipher figurative language.
4. Look out for exaggeration, as riddles may present overly dramatic details to divert your attention.
5. Common phrases and sayings may hide within the puzzle, demanding familiarity.
6. Associations and irony play a crucial role, introducing unexpected connections.
7. Numerical puzzles can also be part of the mystery, requiring you to decode their significance.
8. Elemental imagery, drawn from nature, might hold key descriptors.
9. Rhyming and sound clues can add a poetic dimension.
10. Word Puzzles: Pay attention to anagrams, acrostics, and other wordplay elements.
11. Also, it is important to note you should decode the upcoming riddle using everyday logic and creativity.

Approach the riddle with everyday logic and creativity, avoiding supernatural explanations.

First, you'll encounter **X** examples demonstrating analytical reasoning similar to the riddle you will need to solve.

Then you will be given an unsolved riddle and five options to choose the answer amongst them. Let's think step by step and solve the riddle based on the examples provided above.

User Prompt:

{EXAMPLES_COT}

Answer the following riddle while taking into consideration the examples above. Choose the best and the most logical option from the available choices:

Riddle: ```
{RIDDLE}
```

Options:  
[option 1]: ```{OPTION\_1}```  
[option 2]: ```{OPTION\_2}```  
[option 3]: ```{OPTION\_3}```  
[option 4]: ```{OPTION\_4}```

---

```
[option 5]: ```{OPTION_5}```
```

---

Where the **EXAMPLES\_COT** refer to the few-shot examples with manually generated explanations selected for each test set instance, and the other input parameters are self-explanatory. It is important to note that for BrainTeaser, where there are four multiple-choice options, the prompt is modified accordingly.

### 7.1.4 Methods explored

Before settling on the implementation of our proposed technique, we explored various alternative methods in an effort to improve reasoning performance. One of these methods was **Auto-CoT** [100], where we automatically generated explanations for the entire training dataset, allowing the model to use these explanations as Chain-of-Thought (CoT) reasoning. This approach sought to enhance the model’s ability to reason through lateral thinking problems by breaking down the thought process step-by-step. However, in cases where the explanation or final answer was incorrect, we implemented a pipeline that automatically provided a tip, helping to refine both the explanation and the answer. These corrected explanations were then used as few-shot CoT examples, enabling the model to learn from its mistakes and improve its reasoning capability. While this method showed some promise, it often required significant post-processing and manual intervention to handle the edge cases where the generated explanations were incomplete or unclear.

Another method that we explored, and that holds potential for further investigation, involved **dividing riddles and selecting few-shot examples based on their reasoning process**. The idea here was to enhance the model’s ability to generalize across lateral thinking problems by categorizing riddles according to their underlying reasoning patterns. We attempted to extract these reasoning processes through LLM prompting, intending to group similar riddles and use them to guide the model’s reasoning. However, the techniques applied were not deterministic, which led to inconsistent results. We hypothesize that constructing a more formal taxonomy for categorizing different riddle reasoning types—such as inductive reasoning, deductive reasoning, and analogical reasoning—could enable the selection of few-shot examples in a more structured and reliable manner. This approach might allow the model to better generalize and apply reasoning across different lateral thinking tasks. However, without a well-defined categorization scheme, this method proved challenging to implement effectively.

Given these limitations, we ultimately returned to our proposed method, which combines reconstructed Question-Answer pairs with their original counterparts and fully leverages the capabilities of LLMs. This method, described in detail in the following sections, was more reliable and scalable compared to the alternatives we explored.

### 7.1.5 RISCORE

RISCORE is applied as follows. For each example used in the few-shot (FS) setting, the corresponding contextually reconstructed riddle is appended to the input. For a fair comparison, the number of shots refers to the total number of examples included in the prompt, regardless of whether they are new riddles or context reconstructed examples. Thus, for example RISCORE with 4 shots has the same 2 examples used in 2-shots FS Sim technique augmented with the context reconstruction of them, meaning that keeps only half of the examples drawn from the original dataset and the remainder consisting of reconstructed examples. The experiments were conducted with 2, 4, and 8-shot settings, meaning that 1, 2, and 4 exemplars, respectively, were drawn from the dataset, with the rest being their context reconstructions.

We primarily utilized the Llama3-8B and Llama3-70B model to generate both Question-Answer pairs and distractors. The Question-Answer pairs were produced in both zero-shot and few-shot settings.

Lastly, in our experiments, we differentiate between  $\text{RISCORE}_m$ , which utilizes manually created reconstructions (when available), and RISCORE, which employs the fully automated method for generating reconstructions. The prompt structure for the system-user interaction in this method is the same as in the **FS** method outlined in 7.1.3. The sole distinction is in the examples provided.

### 7.1.6 Models Utilized in RISCORE pipeline

This section analyzes the models used throughout the experiments. Two distinct models were employed, each serving a specific purpose within the overall methodology. The first model was used wherever semantic similarity was required between examples, whether for prompting (FS Sim etc...) or filtering data (see Section 7.1.1). The second model played a crucial role in our proposed method, particularly in generating short distractors (see Section 6.2.2).

#### Choice of Embedding Model

For the embedding representation, we solely experimented with the *gte-large-en-v1.5* model <sup>5</sup> [99], which was utilized through the Sentence Transformers library <sup>6</sup> [66]. This model was selected due to its manageable size, which allows for easy deployment on a local machine. Additionally, it performs well on the corresponding MTEB leaderboard <sup>7</sup>, showing strong results not only in semantic textual similarity (STS) but also across various other leaderboard rankings. This balance of efficiency and high performance made it an ideal choice for our embedding needs.

#### Choice of Zero-shot classification Model

In our approach, we utilize the *bart-large-mnli* model <sup>8</sup> [43], provided by Hugging Face, to classify the correct answer into one of eight distinct categories: 'food', 'person', 'object', 'animal', 'nature', 'time', 'place', 'concept'. These categories were deliberately chosen to be mutually exclusive, avoiding overlap and ensuring clear, distinct classifications.

We opted for this specific model due to its powerful zero-shot classification capabilities, which make it ideal for categorizing short inputs, such as single words or brief phrases, without the need for task-specific fine-tuning. The *bart-large-mnli* model, being instruction-tuned, is particularly adept at handling these minimal inputs while still providing accurate and contextually relevant classifications. Its ability to generalize across various tasks made it well-suited for our requirement to classify concise answers reliably into predefined categories with minimal context.

### 7.1.7 Models for Prompting

In this section, we analyze the models used for prompting with respect to the final results for each dataset. The models employed in the proposed method, such as the sentence transformer for semantic similarity of embeddings and zero-shot classification for mapping answers to eight respective categories, will be further examined in a different section (7.1.6).

We specifically chose instruction-tuned variations of the models because they aligned more closely with our task requirements. Instruction-tuned models typically provide enhanced performance in scenarios where comprehending and following specific instructions is critical. This alignment ensures that the models are better equipped to manage the intricacies of the tasks, leading to more effective and relevant outputs.

The temperature and repetition penalty values were determined through a series of exploratory experiments. To ensure consistency across our work, we systematically applied the same parameters whenever possible. For our experiments, we used a temperature of 0.5 and repetition penalties of either 1.0 or 1.15.

**Llama 3** [1] In our experiments, we chose two variations of the model: the 8B and 70B versions. The Llama3-8B<sup>9</sup> model was used for inference without quantization, which allowed it to deliver results with full precision. This approach was ideal when computational resources were sufficient and precision was critical. On the other hand, the Llama3-70B<sup>10</sup> model, due to hardware limitations, underwent quantization. This process reduced the model's size and computational needs, making it accessible despite the constraints of our

---

<sup>5</sup>[Alibaba-NLP/gte-large-en-v1.5](#)

<sup>6</sup>[SentenceTransformers](#)

<sup>7</sup>[MTEB leaderboard](#)

<sup>8</sup>[facebook/bart-large-mnli](#)

<sup>9</sup>[meta-llama/Meta-Llama-3-8B-Instruct](#)

<sup>10</sup>[meta-llama/Meta-Llama-3-70B-Instruct](#)

hardware. While quantization might lead to some loss of precision, it was necessary to deploy a model with more parameters, which otherwise would not have been feasible.

**Mistral** [32] In our experiments, we used the Mistral-7B-Instruct-v0.2<sup>11</sup> and the Mixtral-8x7B-Instruct-v0.1<sup>12</sup> in their unquantized forms. The Mistral 7B-Instruct v0.2 was selected for its strong instruction-following abilities, maintaining full precision. Similarly, the Mixtral-8x7B-Instruct-v0.1, which integrates eight 7B models, was used unquantized to benefit from its combined performance. Our goal was to explore the effectiveness of smaller models for our riddle tasks, and we preferred these instruction-tuned variations to ensure they were well-suited to our specific requirements, leading to more effective results.

**Qwen2** [95] To further explore smaller model variations, we selected the Qwen2-7B-Instruct<sup>13</sup> version. The Qwen2 family of models, including this 7B-Instruct variant, is known for its strong performance in reasoning tasks. The Qwen2 models are designed with advanced instruction-following capabilities and are particularly effective at complex problem-solving and logical reasoning, making them well-suited for our objectives.

## 7.2 Experimental Setup

In this section, we outline the platforms, tools, and infrastructure used to implement and optimize both the RISCORE pipeline and the baseline techniques for comparison. Our approach required scalable and efficient environments for model development, as well as tools to handle large models under hardware constraints across all techniques. Below, we detail the platforms used for experimentation, the libraries employed to optimize memory usage and performance, and the infrastructure leveraged to scale model deployment beyond local hardware limits, ensuring a thorough comparison between RISCORE and the baseline techniques.

**Cloud Platforms for Experimentation** In our experiments, we primarily utilized Google Colab and Kaggle as the core platforms for running our code and conducting experiments. Both platforms offered accessible, cloud-based environments with GPU acceleration, enabling efficient testing and model development. Google Colab’s free GPU access and Kaggle’s seamless integration with datasets and competition environments provided the flexibility needed for rapid prototyping and scaling up without relying on local hardware resources.

**Libraries for Model Optimization** To handle large models that exceeded our hardware’s memory limits, we used a combination of open-source libraries to optimize performance and reduce resource consumption. The Transformers library [94] allowed us to load and manage large models, while BitsAndBytes played a critical role in quantizing these models to 4-bit precision, significantly reducing memory usage with minimal performance loss. Additionally, the Accelerate [25] library facilitated the efficient distribution of computation across multiple devices, ensuring smooth execution even for large-scale models under hardware constraints.

**Serverless Model Deployment** Beyond these local optimizations, we integrated the Amazon Bedrock API to access foundational models without being bound by local hardware limitations. Bedrock provided a scalable, serverless environment that enabled us to deploy and experiment with large models on demand. This flexibility allowed us to overcome hardware restrictions entirely, ensuring that our model development and testing processes remained efficient and scalable throughout the RISCORE pipeline.

## 7.3 Results

In this section, we present and analyze the results of our experiments. For a more detailed breakdown of the results, including variations across different hyperparameters and configuration settings, please refer to 7.3.4.

---

<sup>11</sup>[mistralai/Mistral-7B-Instruct-v0.2](#)

<sup>12</sup>[mistralai/Mixtral-8x7B-Instruct-v0.1](#)

<sup>13</sup>[Qwen/Qwen2-7B-Instruct](#)

| Method                     | N. | Llama3-70B   | Mistral-8x7B | Llama3-8B    | Mistral-7B   | Qwen2-7B     |
|----------------------------|----|--------------|--------------|--------------|--------------|--------------|
| CoT_ZS                     | 0  | 0.725        | 0.550        | 0.633        | 0.450        | 0.458        |
| Randomly Selected Shots    |    |              |              |              |              |              |
| CoT_FS                     | 2  | 0.758        | 0.617        | 0.633        | 0.475        | 0.608        |
|                            | 4  | 0.683        | 0.583        | 0.608        | 0.508        | 0.650        |
|                            | 8  | 0.708        | 0.642        | 0.658        | 0.508        | 0.667        |
| FS                         | 2  | 0.775        | 0.617        | 0.633        | 0.517        | 0.642        |
|                            | 4  | 0.808        | 0.683        | 0.642        | 0.483        | 0.608        |
|                            | 8  | 0.775        | 0.617        | 0.675        | 0.483        | 0.642        |
| RISCORE <sub>m</sub>       | 2  | 0.783        | 0.625        | 0.667        | 0.458        | 0.608        |
|                            | 4  | 0.758        | 0.617        | 0.675        | 0.517        | 0.625        |
|                            | 8  | 0.800        | 0.650        | 0.667        | 0.400        | 0.592        |
| Semantically Similar Shots |    |              |              |              |              |              |
| FS                         | 2  | 0.825        | 0.692        | 0.700        | 0.517        | 0.600        |
|                            | 4  | 0.792        | 0.683        | 0.717        | 0.458        | 0.633        |
|                            | 8  | 0.783        | 0.667        | 0.767        | 0.533        | 0.650        |
| RISCORE <sub>m</sub>       | 2  | 0.783        | 0.675        | <b>0.767</b> | 0.483        | 0.667        |
|                            | 4  | <b>0.833</b> | 0.708        | 0.742        | <b>0.567</b> | 0.642        |
|                            | 8  | 0.808        | <b>0.708</b> | 0.758        | 0.550        | <b>0.667</b> |

Table 7.2: Model’s performance for *Brain Teaser* using baselines and RISCORE<sub>m</sub> prompting. The best overall results are highlighted in **bold**. More detailed results can be found in Tables 7.6 and 7.7.

### 7.3.1 BrainTeaser

Table 7.2 presents the performance outcomes of various prompting techniques applied to the BrainTeaser dataset. Notably, the CoT\_FS method underperforms compared to few-shot techniques, even when supplied with examples that include manually crafted explanations. This trend holds regardless of the model size utilized. Conversely, there is a significant improvement when the few-shot approach incorporates the provided “context reconstructed” riddles alongside the original riddles. This suggests that context reconstructed riddles effectively guide the model in tracing the appropriate reasoning path. Remarkably, the benefit of using these reconstructed contexts is more pronounced in smaller models compared to larger ones of the same architecture. For instance, the improvement in the Llama3-8B model is substantial: while the 4-shot technique using semantic similarity achieves a score of 0.717, augmenting this approach with four context reconstructed examples (totaling eight shots) enhances performance to 0.758—an improvement of approximately 5.7%. For the larger model, this increase is from 0.792 to 0.808, leading to an improvement of approximately 2.02%. The same pattern is more evident to the Mistral-7B, when for 4-shot we notice an increase of 9.2% while for Mixtral-8x7B we have an increase of 2.5%. A similar trend can be observed in Qwen2-7B, where the 4-shot setting with semantically similar examples performs approximately 3.4% lower than the augmented 8-shot exemplars, which include the same examples along with their context reconstructions, resulting in a total of 8 examples.

At this point, it is worth noting that the performance improvement does not originate solely from the increased number of shots but also from the quality of the shots added. This is evident from the fact that the best results on the dataset are achieved when we include examples with their context reconstruction riddles. Furthermore, by comparing performance while keeping the number of shots constant, we observe that cases using the reconstructed riddles perform better, especially when the number of shots is more than two. In the latter case, the addition of different examples seems more helpful to the model in properly understanding the task, since including only a single example with its contextual reconstruct makes the prompt overly specific.

| Method                                             | N. | Llama3-70B   | Mistral-8x7B | Llama3-8B | Mistral-7B   | Qwen2-7B     |
|----------------------------------------------------|----|--------------|--------------|-----------|--------------|--------------|
| Llama3-70B zeroshot for QA & Llama3-8B distractors |    |              |              |           |              |              |
| RISCORE                                            | 2  | 0.792        | 0.667        | 0.625     | 0.492        | <b>0.625</b> |
|                                                    | 4  | <b>0.792</b> | 0.642        | 0.675     | <b>0.467</b> | 0.625        |
|                                                    | 8  | <b>0.808</b> | <b>0.683</b> | 0.700     | 0.475        | 0.642        |
| Llama3-70B fewshot for QA & Llama3-8B distractors  |    |              |              |           |              |              |
| RISCORE                                            | 2  | 0.750        | 0.675        | 0.683     | 0.475        | <b>0.625</b> |
|                                                    | 4  | <b>0.792</b> | 0.650        | 0.658     | <b>0.558</b> | <b>0.658</b> |
|                                                    | 8  | <b>0.808</b> | <b>0.675</b> | 0.742     | 0.517        | <b>0.658</b> |
| Llama3-70B fewshot for QA & Llama3-70B distractors |    |              |              |           |              |              |
| RISCORE                                            | 2  | 0.783        | 0.667        | 0.683     | 0.500        | <b>0.617</b> |
|                                                    | 4  | <b>0.792</b> | 0.642        | 0.667     | <b>0.508</b> | 0.617        |
|                                                    | 8  | 0.767        | <b>0.683</b> | 0.700     | 0.500        | 0.617        |

Table 7.3: Model’s performance for *Brain Teaser* using RISCORE prompting.

Similarity-based selection was employed for choosing all the exemplars. Results that surpass the FS method with semantically similar examples, using the same number of shots, are underlined. More detailed results can be found in Tables 7.7 and 7.8.

**Using the automated method** Table 7.3 presents the performance of RISCORE using the automated method for generating context reconstructed examples. As expected, the using the automated method yields a smaller improvement in model performance compared to the manually curated examples provided in the BrainTeaser dataset. However, the performance of all models was consistently improved by the addition of context reconstructed riddles, even when using the fully automated algorithm for their generation. For example, for the Llama3-8B model, the performance using 4 shots from the dataset is 0.717, and by using the same shots along with their reconstructions generated by the automated method (8 shots in total), the performance increases to 0.742. The same trend is evident across all relatively smaller models, where augmenting the 4-shot examples selected based on semantic similarity with their automated context pairs yields notable performance improvements. Specifically, in the case of Qwen2-7B, the 4-shot setting augmented with corresponding context exemplars results in a 2.5% performance improvement. In Mistral-7B, the improvement is even more pronounced, with performance increasing by 10% from a baseline of 0.458 in the best-case scenario. Importantly, Table 7.3 shows that in most cases (underlined), it is even better to use the automated method to generate additional examples compared to adding the same number of examples from the dataset. For instance, the performance of Llama3-70B with 8 shots from the dataset is 0.783, whereas using only 4 examples from the dataset and automatically creating the remaining 4 using the context reconstruction method results in a performance of 0.808.

| Method                     | N. | Llama3-70B   | Mistral-8x7B | Llama3-8B    | Mistral-7B   | Qwen2-7B     |
|----------------------------|----|--------------|--------------|--------------|--------------|--------------|
| CoT_ZS                     | 0  | 0.775        | 0.675        | 0.619        | 0.589        | 0.608        |
| Randomly Selected Shots    |    |              |              |              |              |              |
| CoT_FS                     | 2  | 0.789        | 0.692        | 0.625        | 0.594        | 0.667        |
|                            | 4  | 0.783        | 0.686        | 0.672        | 0.603        | 0.656        |
|                            | 8  | 0.783        | 0.697        | 0.658        | 0.597        | 0.625        |
| FS                         | 2  | 0.769        | 0.706        | 0.672        | 0.586        | 0.689        |
|                            | 4  | 0.772        | 0.719        | 0.639        | 0.586        | 0.683        |
|                            | 8  | 0.800        | 0.711        | 0.672        | 0.586        | 0.700        |
| Semantically Similar Shots |    |              |              |              |              |              |
| FS                         | 2  | 0.792        | <b>0.714</b> | 0.706        | 0.608        | 0.722        |
|                            | 4  | <b>0.817</b> | 0.692        | <b>0.711</b> | <b>0.633</b> | 0.714        |
|                            | 8  | 0.800        | 0.675        | 0.681        | 0.611        | <b>0.731</b> |

Table 7.4: Model’s Performance for *RiddleSense* using baseline techniques. The best results overall are in **bold**. More detailed results can be found in Table 7.9.

### 7.3.2 RiddleSense

In this dataset, RISCORE can only be applied to automatically generated examples because the dataset’s format lacks context reconstructions for its questions.

Table 7.4 presents the results of the baseline techniques using various models in the RiddleSense dataset. Once again, the results confirm that the few-shot technique, utilizing semantically similar exemplars for in-context learning, consistently delivers the best performance across all tested models.

In Table 7.5, we present the RiddleSense results using the proposed method for context reconstruction of each input.

A clear trend emerges when comparing a simple 8-shot exemplar selection based on semantic similarity with the 8-shot setting of our method, where the top 4 semantically similar examples are augmented with

| Method                                             | N. | Llama3-70B   | Mistral-8x7B | Llama3-8B    | Mistral-7B   | Qwen2-7B     |
|----------------------------------------------------|----|--------------|--------------|--------------|--------------|--------------|
| Llama3-70B fewshot for QA & Llama3-70B distractors |    |              |              |              |              |              |
| RISCORE                                            | 2  | <b>0.792</b> | 0.672        | 0.692        | 0.600        | 0.697        |
|                                                    | 4  | 0.783        | 0.689        | <b>0.722</b> | 0.600        | <b>0.717</b> |
|                                                    | 8  | 0.789        | <b>0.700</b> | <b>0.708</b> | 0.597        | <b>0.731</b> |
| Llama3-70B fewshot for QA & Llama3-8B distractors  |    |              |              |              |              |              |
| RISCORE                                            | 2  | 0.786        | <b>0.719</b> | 0.681        | 0.603        | 0.681        |
|                                                    | 4  | 0.789        | 0.686        | 0.686        | 0.606        | 0.697        |
|                                                    | 8  | 0.775        | <b>0.689</b> | <b>0.706</b> | <b>0.617</b> | 0.719        |
| Llama3-8B zeroshot for QA & Llama3-8B distractors  |    |              |              |              |              |              |
| RISCORE                                            | 2  | <b>0.792</b> | 0.681        | 0.689        | 0.589        | 0.694        |
|                                                    | 4  | 0.778        | <b>0.714</b> | 0.700        | 0.600        | 0.683        |
|                                                    | 8  | <b>0.806</b> | <b>0.689</b> | <b>0.686</b> | <b>0.614</b> | 0.689        |

Table 7.5: Model’s performance for *RiddleSense* using RISCORE prompting. Similarity-based selection was employed for choosing all the exemplars. Results that surpass the FS method with semantically similar examples, using the same number of shots, are underlined. More detailed results can be found in Table 7.10.

our generated contextual pairs. Notably, the results show that our method consistently outperforms the standard 8-shot exemplar approach, demonstrating a significant improvement in model performance across various instances. An example of this trend is observed with the Llama3-8B model, where our method scores 0.708, approximately 2% higher than the few-shot setting based on semantic similarity, which achieves a score of 0.681. The same pattern is evident when comparing the two 4-shot settings. We achieve similar or marginally better accuracy using a total of just 4 examples—two original and two generated contextual reconstructions. This underlines the effectiveness of integrating contextually reconstructed pairs in enhancing model accuracy. We do not achieve a significant performance boost; however, we attain similar or marginally better results while relying on less grounded knowledge. This demonstrates the efficiency of our method, as it maintains comparable performance with fewer, yet strategically selected, exemplars.

### 7.3.3 Quality of context reconstructed riddles

To generate context reconstructed riddles, we used Llama3 models with 8 billion and 70 billion parameters in both few-shot (FS) and zero-shot (ZS) settings. We found that the Llama3-8B model struggled to produce high-quality Question-Answer pairs for the BrainTeaser dataset and was therefore not used in RISCORE. This difficulty likely arises from the BrainTeaser dataset’s demand for lateral thinking, which is particularly challenging for the smaller model. The Question-Answer pairs are essential, and if their quality is insufficient—as observed with Llama3-8B on the BrainTeaser dataset—high-quality distractors alone cannot compensate for this deficiency. However, for vertical thinking riddles, the smaller model effectively generates riddles even in the ZS setting, which, when used in the few-shot (FS) setting, can lead to increased performance compared to real examples drawn from the dataset.

### 7.3.4 Detailed Results

In this section, we present the detailed results for both datasets across all our experimental techniques. Due to the extensive nature of the experiments, the results for each dataset are organized into subtables. We begin with the BrainTeaser dataset, where the metrics are more detailed due to the dataset’s structure, followed by the results for the RiddleSense dataset. The structure is organized by method per model, with results presented in descending order based on score for each method, rather than by the number of examples. This differs from the approach outlined in 7.3.

| Model                          | Method      | Num.Ex | Task | Temp | Rep_Pen | Original | Semantic | Context | Ori. + Sem. | Ori. + Sem. + Con. | Average |
|--------------------------------|-------------|--------|------|------|---------|----------|----------|---------|-------------|--------------------|---------|
| Chain-of-Thought Zero-shot     |             |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct      | CoT_ZS (Q)  | 0      | SP   | 0.5  | 1.15    | 0.725    | 0.775    | 0.675   | 0.675       | 0.550              | 0.725   |
| Mixtral-8x7B-Instruct-v0.1     | CoT_ZS      | 0      | SP   | 0.5  | 1.0     | 0.575    | 0.550    | 0.525   | 0.475       | 0.275              | 0.550   |
| Meta-Llama-3-8B-Instruct       | CoT_ZS      | 0      | SP   | 0.5  | 1.0     | 0.625    | 0.650    | 0.625   | 0.500       | 0.325              | 0.633   |
| Mistral-7B-Instruct-v0.2       | CoT_ZS      | 0      | SP   | 0.5  | 1.0     | 0.375    | 0.475    | 0.500   | 0.300       | 0.250              | 0.450   |
| Qwen2-7B-Instruct              | CoT_ZS      | 0      | SP   | 0.5  | 1.15    | 0.475    | 0.450    | 0.450   | 0.350       | 0.200              | 0.458   |
| Few-shot with CoT Explanations |             |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct      | CoT_FS (Q)  | 2      | SP   | 0.5  | 1.15    | 0.850    | 0.750    | 0.675   | 0.700       | 0.500              | 0.758   |
| Meta-Llama-3-70B-Instruct      | CoT_FS (Q)  | 8      | SP   | 0.5  | 1.0     | 0.675    | 0.750    | 0.700   | 0.600       | 0.425              | 0.708   |
| Meta-Llama-3-70B-Instruct      | CoT_FS (Q)  | 4      | SP   | 0.5  | 1.15    | 0.700    | 0.650    | 0.700   | 0.575       | 0.450              | 0.683   |
| Mixtral-8x7B-v0.1              | CoT_FS      | 8      | SP   | 0.5  | 1.0     | 0.650    | 0.675    | 0.600   | 0.575       | 0.375              | 0.642   |
| Mixtral-8x7B-v0.1              | CoT_FS      | 2      | SP   | 0.5  | 1.0     | 0.625    | 0.625    | 0.600   | 0.525       | 0.350              | 0.617   |
| Mixtral-8x7B-v0.1              | CoT_FS      | 4      | SP   | 0.5  | 1.0     | 0.575    | 0.600    | 0.575   | 0.475       | 0.350              | 0.583   |
| Meta-Llama-3-8B-Instruct       | CoT_FS      | 8      | SP   | 0.5  | 1.0     | 0.675    | 0.700    | 0.600   | 0.525       | 0.325              | 0.658   |
| Meta-Llama-3-8B-Instruct       | CoT_FS      | 2      | SP   | 0.5  | 1.0     | 0.725    | 0.625    | 0.550   | 0.550       | 0.350              | 0.633   |
| Meta-Llama-3-8B-Instruct       | CoT_FS      | 4      | SP   | 0.5  | 1.0     | 0.650    | 0.625    | 0.550   | 0.500       | 0.300              | 0.608   |
| Mistral-7B-Instruct-v0.2       | CoT_FS      | 8      | SP   | 0.5  | 1.0     | 0.525    | 0.550    | 0.450   | 0.375       | 0.300              | 0.508   |
| Mistral-7B-Instruct-v0.2       | CoT_FS      | 4      | SP   | 0.5  | 1.0     | 0.525    | 0.550    | 0.450   | 0.375       | 0.300              | 0.508   |
| Mistral-7B-Instruct-v0.2       | CoT_FS      | 2      | SP   | 0.5  | 1.0     | 0.525    | 0.425    | 0.475   | 0.300       | 0.225              | 0.475   |
| Qwen2-7B-Instruct              | CoT_FS      | 8      | SP   | 0.5  | 1.0     | 0.600    | 0.725    | 0.675   | 0.550       | 0.425              | 0.667   |
| Qwen2-7B-Instruct              | CoT_FS      | 4      | SP   | 0.5  | 1.0     | 0.650    | 0.675    | 0.625   | 0.550       | 0.450              | 0.650   |
| Qwen2-7B-Instruct              | CoT_FS      | 2      | SP   | 0.5  | 1.0     | 0.675    | 0.550    | 0.600   | 0.450       | 0.375              | 0.608   |
| Few-shot with Random Selection |             |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct      | FS Rand (Q) | 4      | SP   | 0.5  | 1.15    | 0.825    | 0.850    | 0.750   | 0.800       | 0.700              | 0.808   |
| Meta-Llama-3-70B-Instruct      | FS Rand (Q) | 8      | SP   | 0.5  | 1.15    | 0.775    | 0.800    | 0.750   | 0.675       | 0.550              | 0.775   |
| Meta-Llama-3-70B-Instruct      | FS Rand (Q) | 2      | SP   | 0.5  | 1.15    | 0.750    | 0.800    | 0.775   | 0.725       | 0.600              | 0.775   |
| Mixtral-8x7B-Instruct-v0.1     | FS Rand     | 4      | SP   | 0.5  | 1.0     | 0.675    | 0.775    | 0.600   | 0.600       | 0.400              | 0.683   |
| Mixtral-8x7B-Instruct-v0.1     | FS Rand     | 2      | SP   | 0.5  | 1.0     | 0.650    | 0.625    | 0.575   | 0.475       | 0.300              | 0.617   |
| Mixtral-8x7B-Instruct-v0.1     | FS Rand     | 8      | SP   | 0.5  | 1.0     | 0.675    | 0.650    | 0.525   | 0.525       | 0.350              | 0.617   |
| Meta-Llama-3-8B-Instruct       | FS Rand     | 8      | SP   | 0.5  | 1.0     | 0.750    | 0.625    | 0.650   | 0.575       | 0.400              | 0.675   |
| Meta-Llama-3-8B-Instruct       | FS Rand     | 4      | SP   | 0.5  | 1.0     | 0.625    | 0.650    | 0.650   | 0.500       | 0.375              | 0.642   |
| Meta-Llama-3-8B-Instruct       | FS Rand     | 2      | SP   | 0.5  | 1.0     | 0.600    | 0.700    | 0.600   | 0.550       | 0.375              | 0.633   |
| Mistral-7B-Instruct-v0.2       | FS Rand     | 2      | SP   | 0.5  | 1.0     | 0.525    | 0.550    | 0.475   | 0.425       | 0.275              | 0.517   |
| Mistral-7B-Instruct-v0.2       | FS Rand     | 8      | SP   | 0.5  | 1.0     | 0.450    | 0.575    | 0.425   | 0.400       | 0.300              | 0.483   |
| Mistral-7B-Instruct-v0.2       | FS Rand     | 4      | SP   | 0.5  | 1.0     | 0.550    | 0.425    | 0.475   | 0.325       | 0.225              | 0.483   |
| Qwen2-7B-Instruct              | FS Rand     | 2      | SP   | 0.5  | 1.15    | 0.675    | 0.650    | 0.600   | 0.550       | 0.425              | 0.642   |
| Qwen2-7B-Instruct              | FS Rand     | 8      | SP   | 0.5  | 1.15    | 0.700    | 0.600    | 0.625   | 0.525       | 0.425              | 0.642   |
| Qwen2-7B-Instruct              | FS Rand     | 4      | SP   | 0.5  | 1.15    | 0.675    | 0.575    | 0.575   | 0.525       | 0.400              | 0.608   |

Table 7.6: Model performance for *BrainTeaser* (Part 1). The presence of **(Q)** in the method column indicates that the results correspond to the quantized version of the model.

| Model                             | Method                        | Num.Ex | Task | Temp | Rep_Pen | Original | Semantic | Context | Ori. + Sem. | Ori. + Sem. + Con. | Average |
|-----------------------------------|-------------------------------|--------|------|------|---------|----------|----------|---------|-------------|--------------------|---------|
| Few-shot with Semantic Similarity |                               |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct         | FS Sim (Q)                    | 2      | SP   | 0.5  | 1.15    | 0.850    | 0.875    | 0.750   | 0.800       | 0.675              | 0.825   |
| Meta-Llama-3-70B-Instruct         | FS Sim (Q)                    | 4      | SP   | 0.5  | 1.15    | 0.825    | 0.775    | 0.775   | 0.700       | 0.575              | 0.792   |
| Meta-Llama-3-70B-Instruct         | FS Sim (Q)                    | 8      | SP   | 0.5  | 1.15    | 0.775    | 0.825    | 0.750   | 0.700       | 0.600              | 0.783   |
| Mixtral-8x7B-Instruct-v0.1        | FS Sim                        | 2      | SP   | 0.5  | 1.0     | 0.700    | 0.700    | 0.675   | 0.575       | 0.450              | 0.692   |
| Mixtral-8x7B-Instruct-v0.1        | FS Sim                        | 4      | SP   | 0.5  | 1.0     | 0.750    | 0.650    | 0.650   | 0.575       | 0.425              | 0.683   |
| Mixtral-8x7B-Instruct-v0.1        | FS Sim                        | 8      | SP   | 0.5  | 1.0     | 0.750    | 0.625    | 0.625   | 0.525       | 0.375              | 0.667   |
| Meta-Llama-3-8B-Instruct          | FS Sim                        | 8      | SP   | 0.5  | 1.0     | 0.750    | 0.775    | 0.775   | 0.700       | 0.600              | 0.767   |
| Meta-Llama-3-8B-Instruct          | FS Sim                        | 4      | SP   | 0.5  | 1.0     | 0.675    | 0.825    | 0.650   | 0.675       | 0.475              | 0.717   |
| Meta-Llama-3-8B-Instruct          | FS Sim                        | 2      | SP   | 0.5  | 1.0     | 0.800    | 0.675    | 0.625   | 0.625       | 0.525              | 0.700   |
| Mistral-7B-Instruct-v0.2          | FS Sim                        | 8      | SP   | 0.5  | 1.0     | 0.550    | 0.600    | 0.450   | 0.500       | 0.325              | 0.533   |
| Mistral-7B-Instruct-v0.2          | FS Sim                        | 2      | SP   | 0.5  | 1.0     | 0.55     | 0.475    | 0.525   | 0.375       | 0.275              | 0.517   |
| Mistral-7B-Instruct-v0.2          | FS Sim                        | 4      | SP   | 0.5  | 1.0     | 0.475    | 0.450    | 0.450   | 0.350       | 0.225              | 0.458   |
| Qwen2-7B-Instruct                 | FS Sim                        | 8      | SP   | 0.5  | 1.15    | 0.675    | 0.625    | 0.650   | 0.575       | 0.500              | 0.650   |
| Qwen2-7B-Instruct                 | FS Sim                        | 4      | SP   | 0.5  | 1.15    | 0.650    | 0.600    | 0.650   | 0.550       | 0.450              | 0.633   |
| Qwen2-7B-Instruct                 | FS Sim                        | 2      | SP   | 0.5  | 1.15    | 0.600    | 0.550    | 0.650   | 0.500       | 0.400              | 0.600   |
| RISCORE <sub>m</sub> Rand         |                               |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct         | RISCORE <sub>m</sub> Rand (Q) | 8      | SP   | 0.5  | 1.15    | 0.775    | 0.825    | 0.800   | 0.700       | 0.600              | 0.800   |
| Meta-Llama-3-70B-Instruct         | RISCORE <sub>m</sub> Rand (Q) | 2      | SP   | 0.5  | 1.15    | 0.850    | 0.775    | 0.725   | 0.725       | 0.650              | 0.783   |
| Meta-Llama-3-70B-Instruct         | RISCORE <sub>m</sub> Rand (Q) | 4      | SP   | 0.5  | 1.15    | 0.775    | 0.800    | 0.700   | 0.725       | 0.600              | 0.758   |
| Mixtral-8x7B-Instruct-v0.1        | RISCORE <sub>m</sub> Rand     | 8      | SP   | 0.5  | 1.0     | 0.700    | 0.625    | 0.625   | 0.600       | 0.450              | 0.650   |
| Mixtral-8x7B-Instruct-v0.1        | RISCORE <sub>m</sub> Rand     | 2      | SP   | 0.5  | 1.0     | 0.725    | 0.600    | 0.550   | 0.600       | 0.400              | 0.625   |
| Mixtral-8x7B-Instruct-v0.1        | RISCORE <sub>m</sub> Rand     | 4      | SP   | 0.5  | 1.0     | 0.675    | 0.650    | 0.525   | 0.550       | 0.325              | 0.617   |
| Meta-Llama-3-8B-Instruct          | RISCORE <sub>m</sub> Rand     | 4      | SP   | 0.5  | 1.0     | 0.650    | 0.700    | 0.675   | 0.625       | 0.500              | 0.675   |
| Meta-Llama-3-8B-Instruct          | RISCORE <sub>m</sub> Rand     | 2      | SP   | 0.5  | 1.0     | 0.700    | 0.725    | 0.575   | 0.525       | 0.375              | 0.667   |
| Meta-Llama-3-8B-Instruct          | RISCORE <sub>m</sub> Rand     | 8      | SP   | 0.5  | 1.0     | 0.700    | 0.750    | 0.550   | 0.575       | 0.425              | 0.667   |
| Mistral-7B-Instruct-v0.2          | RISCORE <sub>m</sub> Rand     | 4      | SP   | 0.5  | 1.0     | 0.575    | 0.475    | 0.500   | 0.425       | 0.300              | 0.517   |
| Mistral-7B-Instruct-v0.2          | RISCORE <sub>m</sub> Rand     | 2      | SP   | 0.5  | 1.0     | 0.550    | 0.425    | 0.400   | 0.325       | 0.200              | 0.458   |
| Mistral-7B-Instruct-v0.2          | RISCORE <sub>m</sub> Rand     | 8      | SP   | 0.5  | 1.0     | 0.350    | 0.450    | 0.400   | 0.275       | 0.200              | 0.400   |
| Qwen2-7B-Instruct                 | RISCORE <sub>m</sub> Rand     | 4      | SP   | 0.5  | 1.15    | 0.625    | 0.650    | 0.600   | 0.550       | 0.400              | 0.625   |
| Qwen2-7B-Instruct                 | RISCORE <sub>m</sub> Rand     | 2      | SP   | 0.5  | 1.15    | 0.650    | 0.575    | 0.600   | 0.500       | 0.400              | 0.608   |
| Qwen2-7B-Instruct                 | RISCORE <sub>m</sub> Rand     | 8      | SP   | 0.5  | 1.15    | 0.625    | 0.550    | 0.600   | 0.450       | 0.325              | 0.592   |
| RISCORE <sub>m</sub> Sim          |                               |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct         | RISCORE <sub>m</sub> Sim (Q)  | 4      | SP   | 0.5  | 1.15    | 0.850    | 0.850    | 0.800   | 0.800       | 0.650              | 0.833   |
| Meta-Llama-3-70B-Instruct         | RISCORE <sub>m</sub> Sim (Q)  | 8      | SP   | 0.5  | 1.15    | 0.800    | 0.775    | 0.850   | 0.700       | 0.625              | 0.808   |
| Meta-Llama-3-70B-Instruct         | RISCORE <sub>m</sub> Sim (Q)  | 2      | SP   | 0.5  | 1.15    | 0.850    | 0.775    | 0.725   | 0.725       | 0.600              | 0.783   |
| Mixtral-8x7B-Instruct-v0.1        | RISCORE <sub>m</sub> Sim      | 8      | SP   | 0.5  | 1.0     | 0.775    | 0.625    | 0.725   | 0.575       | 0.500              | 0.708   |
| Mixtral-8x7B-Instruct-v0.1        | RISCORE <sub>m</sub> Sim      | 4      | SP   | 0.5  | 1.0     | 0.675    | 0.750    | 0.700   | 0.600       | 0.425              | 0.708   |
| Mixtral-8x7B-Instruct-v0.1        | RISCORE <sub>m</sub> Sim      | 2      | SP   | 0.5  | 1.0     | 0.725    | 0.675    | 0.625   | 0.675       | 0.475              | 0.675   |
| Meta-Llama-3-8B-Instruct          | RISCORE <sub>m</sub> Sim      | 2      | SP   | 0.5  | 1.0     | 0.750    | 0.875    | 0.675   | 0.675       | 0.575              | 0.767   |
| Meta-Llama-3-8B-Instruct          | RISCORE <sub>m</sub> Sim      | 8      | SP   | 0.5  | 1.0     | 0.775    | 0.800    | 0.700   | 0.775       | 0.650              | 0.758   |
| Meta-Llama-3-8B-Instruct          | RISCORE <sub>m</sub> Sim      | 4      | SP   | 0.5  | 1.0     | 0.825    | 0.700    | 0.700   | 0.675       | 0.525              | 0.742   |
| Mistral-7B-Instruct-v0.2          | RISCORE <sub>m</sub> Sim      | 4      | SP   | 0.5  | 1.0     | 0.575    | 0.550    | 0.575   | 0.400       | 0.325              | 0.567   |
| Mistral-7B-Instruct-v0.2          | RISCORE <sub>m</sub> Sim      | 8      | SP   | 0.5  | 1.0     | 0.575    | 0.575    | 0.500   | 0.450       | 0.325              | 0.550   |
| Mistral-7B-Instruct-v0.2          | RISCORE <sub>m</sub> Sim      | 2      | SP   | 0.5  | 1.0     | 0.500    | 0.475    | 0.475   | 0.350       | 0.200              | 0.483   |
| Qwen2-7B-Instruct                 | RISCORE <sub>m</sub> Sim      | 8      | SP   | 0.5  | 1.15    | 0.700    | 0.650    | 0.650   | 0.575       | 0.425              | 0.667   |
| Qwen2-7B-Instruct                 | RISCORE <sub>m</sub> Sim      | 2      | SP   | 0.5  | 1.15    | 0.700    | 0.625    | 0.675   | 0.550       | 0.375              | 0.667   |
| Qwen2-7B-Instruct                 | RISCORE <sub>m</sub> Sim      | 4      | SP   | 0.5  | 1.15    | 0.750    | 0.625    | 0.550   | 0.575       | 0.425              | 0.642   |

Table 7.7: Model performance for *BrainTeaser* (Part 2). The presence of (Q) in the method column indicates that the results correspond to the quantized version of the model.

| Model                                                  | Method          | Num.Ex | Task | Temp | Rep_Pen | Original | Semantic | Context | Ori. + Sem. | Ori. + Sem. + Con. | Average |
|--------------------------------------------------------|-----------------|--------|------|------|---------|----------|----------|---------|-------------|--------------------|---------|
| RISCORE Results                                        |                 |        |      |      |         |          |          |         |             |                    |         |
| Llama3-70B zeroshot for QA & Llama3-8B for distractors |                 |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 8      | SP   | 0.5  | 1.15    | 0.850    | 0.825    | 0.750   | 0.775       | 0.600              | 0.808   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 2      | SP   | 0.5  | 1.15    | 0.800    | 0.775    | 0.800   | 0.750       | 0.650              | 0.792   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 4      | SP   | 0.5  | 1.15    | 0.825    | 0.750    | 0.800   | 0.675       | 0.600              | 0.792   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.700    | 0.750    | 0.600   | 0.650       | 0.475              | 0.683   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.750    | 0.750    | 0.500   | 0.650       | 0.375              | 0.667   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.725    | 0.625    | 0.575   | 0.575       | 0.375              | 0.642   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.700    | 0.725    | 0.675   | 0.650       | 0.525              | 0.700   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.650    | 0.675    | 0.700   | 0.550       | 0.425              | 0.675   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.700    | 0.600    | 0.575   | 0.525       | 0.350              | 0.625   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.550    | 0.450    | 0.475   | 0.375       | 0.300              | 0.492   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.525    | 0.400    | 0.475   | 0.300       | 0.200              | 0.467   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.425    | 0.525    | 0.475   | 0.400       | 0.325              | 0.475   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 8      | SP   | 0.5  | 1.15    | 0.600    | 0.700    | 0.625   | 0.550       | 0.425              | 0.642   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 2      | SP   | 0.5  | 1.15    | 0.650    | 0.650    | 0.575   | 0.550       | 0.400              | 0.625   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 4      | SP   | 0.5  | 1.15    | 0.600    | 0.650    | 0.625   | 0.550       | 0.450              | 0.625   |
| Llama3-70B fewshot for QA & Llama3-8B for distractors  |                 |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 8      | SP   | 0.5  | 1.15    | 0.800    | 0.800    | 0.825   | 0.750       | 0.650              | 0.808   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 4      | SP   | 0.5  | 1.15    | 0.775    | 0.800    | 0.800   | 0.700       | 0.600              | 0.792   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 2      | SP   | 0.5  | 1.15    | 0.800    | 0.725    | 0.725   | 0.725       | 0.600              | 0.750   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.725    | 0.725    | 0.575   | 0.675       | 0.475              | 0.675   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.700    | 0.675    | 0.65    | 0.625       | 0.450              | 0.675   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.700    | 0.675    | 0.575   | 0.625       | 0.400              | 0.650   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.775    | 0.750    | 0.700   | 0.700       | 0.600              | 0.742   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.725    | 0.700    | 0.625   | 0.600       | 0.400              | 0.683   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.725    | 0.625    | 0.625   | 0.550       | 0.450              | 0.658   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.625    | 0.575    | 0.475   | 0.475       | 0.350              | 0.558   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.500    | 0.550    | 0.500   | 0.425       | 0.350              | 0.517   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.550    | 0.425    | 0.450   | 0.375       | 0.225              | 0.475   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 4      | SP   | 0.5  | 1.15    | 0.625    | 0.700    | 0.650   | 0.575       | 0.450              | 0.658   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 8      | SP   | 0.5  | 1.15    | 0.650    | 0.675    | 0.650   | 0.575       | 0.475              | 0.658   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 2      | SP   | 0.5  | 1.15    | 0.625    | 0.650    | 0.600   | 0.525       | 0.400              | 0.625   |
| Llama3-70B fewshot for QA & Llama3-70B for distractors |                 |        |      |      |         |          |          |         |             |                    |         |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 4      | SP   | 0.5  | 1.15    | 0.875    | 0.775    | 0.725   | 0.750       | 0.600              | 0.792   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 2      | SP   | 0.5  | 1.15    | 0.775    | 0.825    | 0.750   | 0.775       | 0.675              | 0.783   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim (Q) | 8      | SP   | 0.5  | 1.15    | 0.775    | 0.750    | 0.775   | 0.725       | 0.600              | 0.767   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.700    | 0.725    | 0.625   | 0.600       | 0.450              | 0.683   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.700    | 0.700    | 0.600   | 0.625       | 0.500              | 0.667   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Simm    | 4      | SP   | 0.5  | 1.0     | 0.725    | 0.650    | 0.550   | 0.600       | 0.425              | 0.642   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.800    | 0.675    | 0.625   | 0.625       | 0.475              | 0.700   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.675    | 0.700    | 0.675   | 0.600       | 0.475              | 0.683   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.725    | 0.650    | 0.625   | 0.550       | 0.475              | 0.667   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 4      | SP   | 0.5  | 1.0     | 0.575    | 0.500    | 0.450   | 0.375       | 0.300              | 0.508   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 2      | SP   | 0.5  | 1.0     | 0.625    | 0.400    | 0.475   | 0.350       | 0.275              | 0.500   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim     | 8      | SP   | 0.5  | 1.0     | 0.600    | 0.475    | 0.425   | 0.400       | 0.300              | 0.500   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 2      | SP   | 0.5  | 1.15    | 0.650    | 0.600    | 0.600   | 0.500       | 0.400              | 0.617   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 4      | SP   | 0.5  | 1.15    | 0.650    | 0.625    | 0.575   | 0.600       | 0.475              | 0.617   |
| Qwen2-7B-Instruct                                      | RISCORE Sim     | 8      | SP   | 0.5  | 1.15    | 0.625    | 0.625    | 0.600   | 0.575       | 0.450              | 0.617   |

Table 7.8: Model performance for *BrainTeaser* (Part 3). The presence of **(Q)** in the method column indicates that the results correspond to the quantized version of the model.

| Model                                    | Method      | Num.Ex | Quant | Temp | Rep_Pen | Average |
|------------------------------------------|-------------|--------|-------|------|---------|---------|
| <b>Chain-of-Thought Zero-shot</b>        |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                | CoT_ZS (Q)  | 0      | 4bit  | 0.5  | 1.15    | 0.775   |
| Mixtral-8x7B-v0.1                        | CoT_ZS      | 0      | False | 0.5  | 1.0     | 0.675   |
| Meta-Llama-3-8B-Instruct                 | CoT_ZS      | 0      | False | 0.5  | 1.0     | 0.619   |
| Mistral-7B-Instruct-v0.2                 | CoT_ZS      | 0      | False | 0.5  | 1.0     | 0.589   |
| Qwen2-7B-Instruct                        | CoT_ZS      | 0      | False | 0.5  | 1.15    | 0.608   |
| Meta-Llama-3-8B-Instruct                 | CoT_ZS      | 0      | 4bit  | 0.5  | 1.15    | 0.633   |
| <b>Few-shot with CoT Explanations</b>    |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                | CoT_FS (Q)  | 2      | 4bit  | 0.5  | 1.15    | 0.789   |
| Meta-Llama-3-70B-Instruct                | CoT_FS (Q)  | 4      | 4bit  | 0.5  | 1.15    | 0.783   |
| Meta-Llama-3-70B-Instruct                | CoT_FS (Q)  | 8      | 4bit  | 0.5  | 1.15    | 0.783   |
| Mixtral-8x7B-v0.1                        | CoT_FS      | 8      | False | 0.5  | 1.0     | 0.697   |
| Mixtral-8x7B-v0.1                        | CoT_FS      | 2      | False | 0.5  | 1.0     | 0.692   |
| Mixtral-8x7B-v0.1                        | CoT_FS      | 4      | False | 0.5  | 1.0     | 0.686   |
| Meta-Llama-3-8B-Instruct                 | CoT_FS      | 4      | False | 0.5  | 1.0     | 0.672   |
| Meta-Llama-3-8B-Instruct                 | CoT_FS      | 8      | False | 0.5  | 1.0     | 0.658   |
| Meta-Llama-3-8B-Instruct                 | CoT_FS      | 2      | False | 0.5  | 1.0     | 0.625   |
| Mistral-7B-Instruct-v0.2                 | CoT_FS      | 4      | False | 0.5  | 1.0     | 0.603   |
| Mistral-7B-Instruct-v0.2                 | CoT_FS      | 8      | False | 0.5  | 1.0     | 0.597   |
| Mistral-7B-Instruct-v0.2                 | CoT_FS      | 2      | False | 0.5  | 1.0     | 0.594   |
| Qwen2-7B-Instruct                        | CoT_FS      | 2      | False | 0.5  | 1.15    | 0.667   |
| Qwen2-7B-Instruct                        | CoT_FS      | 4      | False | 0.5  | 1.15    | 0.656   |
| Qwen2-7B-Instruct                        | CoT_FS      | 8      | False | 0.5  | 1.15    | 0.625   |
| <b>Few-shot with Random Selection</b>    |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                | FS Rand (Q) | 4      | 4bit  | 0.5  | 1.15    | 0.800   |
| Meta-Llama-3-70B-Instruct                | FS Rand (Q) | 8      | 4bit  | 0.5  | 1.15    | 0.772   |
| Meta-Llama-3-70B-Instruct                | FS Rand (Q) | 2      | 4bit  | 0.5  | 1.15    | 0.769   |
| Mixtral-8x7B-v0.1                        | FS Rand     | 4      | False | 0.5  | 1.0     | 0.719   |
| Mixtral-8x7B-v0.1                        | FS Rand     | 8      | False | 0.5  | 1.0     | 0.711   |
| Mixtral-8x7B-v0.1                        | FS Rand     | 2      | False | 0.5  | 1.0     | 0.706   |
| Meta-Llama-3-8B-Instruct                 | FS Rand     | 2      | False | 0.5  | 1.0     | 0.672   |
| Meta-Llama-3-8B-Instruct                 | FS Rand     | 8      | False | 0.5  | 1.0     | 0.672   |
| Meta-Llama-3-8B-Instruct                 | FS Rand     | 4      | False | 0.5  | 1.0     | 0.639   |
| Mistral-7B-Instruct-v0.2                 | FS Rand     | 2      | False | 0.5  | 1.0     | 0.586   |
| Mistral-7B-Instruct-v0.2                 | FS Rand     | 4      | False | 0.5  | 1.0     | 0.586   |
| Mistral-7B-Instruct-v0.2                 | FS Rand     | 8      | False | 0.5  | 1.0     | 0.586   |
| Qwen2-7B-Instruct                        | FS Rand     | 8      | False | 0.5  | 1.15    | 0.700   |
| Qwen2-7B-Instruct                        | FS Rand     | 2      | False | 0.5  | 1.15    | 0.689   |
| Qwen2-7B-Instruct                        | FS Rand     | 4      | False | 0.5  | 1.15    | 0.683   |
| <b>Few-shot with Semantic Similarity</b> |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                | FS Sim (Q)  | 4      | 4bit  | 0.5  | 1.15    | 0.817   |
| Meta-Llama-3-70B-Instruct                | FS Sim (Q)  | 8      | 4bit  | 0.5  | 1.15    | 0.800   |
| Meta-Llama-3-70B-Instruct                | FS Sim (Q)  | 2      | 4bit  | 0.5  | 1.15    | 0.792   |
| Mixtral-8x7B-v0.1                        | FS Sim      | 2      | False | 0.5  | 1.0     | 0.714   |
| Mixtral-8x7B-v0.1                        | FS Sim      | 4      | False | 0.5  | 1.0     | 0.692   |
| Mixtral-8x7B-v0.1                        | FS Sim      | 8      | False | 0.5  | 1.0     | 0.675   |
| Meta-Llama-3-8B-Instruct                 | FS Sim      | 4      | False | 0.5  | 1.0     | 0.711   |
| Meta-Llama-3-8B-Instruct                 | FS Sim      | 2      | False | 0.5  | 1.0     | 0.706   |
| Meta-Llama-3-8B-Instruct                 | FS Sim      | 8      | False | 0.5  | 1.0     | 0.681   |
| Mistral-7B-Instruct-v0.2                 | FS Sim      | 4      | False | 0.5  | 1.0     | 0.633   |
| Mistral-7B-Instruct-v0.2                 | FS Sim      | 8      | False | 0.5  | 1.0     | 0.611   |
| Mistral-7B-Instruct-v0.2                 | FS Sim      | 2      | False | 0.5  | 1.0     | 0.608   |
| Qwen2-7B-Instruct                        | FS Sim      | 8      | False | 0.5  | 1.15    | 0.731   |
| Qwen2-7B-Instruct                        | FS Sim      | 2      | False | 0.5  | 1.15    | 0.722   |
| Qwen2-7B-Instruct                        | FS Sim      | 4      | False | 0.5  | 1.15    | 0.714   |

Table 7.9: Model Performance for *RiddleSense* (Part 1). The column **Quant** indicates whether the model is quantized or not.

| Model                                                  | Method      | Num.Ex | Quant | Temp | Rep_Pen | Average |
|--------------------------------------------------------|-------------|--------|-------|------|---------|---------|
| <b>RISCORE Results</b>                                 |             |        |       |      |         |         |
| Llama3-70B Fewshot for QA & Llama3-70B for distractors |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 2      | True  | 0.5  | 1.15    | 0.792   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 8      | True  | 0.5  | 1.15    | 0.789   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 4      | True  | 0.5  | 1.15    | 0.783   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.700   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.689   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.672   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.722   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.708   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.692   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.600   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.600   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.597   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.731   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.717   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.697   |
| Llama3-70B Fewshot for QA & Llama3-8B for distractors  |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 4      | True  | 0.5  | 1.15    | 0.789   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 2      | True  | 0.5  | 1.15    | 0.786   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 8      | True  | 0.5  | 1.15    | 0.775   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.719   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.689   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.686   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.706   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.686   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.681   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.617   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.606   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.603   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.719   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.697   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.681   |
| Llama3-8B Fewshot for QA & Llama3-8B for distractors   |             |        |       |      |         |         |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 8      | True  | 0.5  | 1.15    | 0.806   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 2      | True  | 0.5  | 1.15    | 0.792   |
| Meta-Llama-3-70B-Instruct                              | RISCORE Sim | 4      | True  | 0.5  | 1.15    | 0.778   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.714   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.689   |
| Mixtral-8x7B-Instruct-v0.1                             | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.681   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.700   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.689   |
| Meta-Llama-3-8B-Instruct                               | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.686   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.614   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.600   |
| Mistral-7B-Instruct-v0.2                               | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.589   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 2      | False | 0.5  | 1.0     | 0.694   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 8      | False | 0.5  | 1.0     | 0.689   |
| Qwen2-7B-Instruct                                      | RISCORE Sim | 4      | False | 0.5  | 1.0     | 0.683   |

Table 7.10: Model Performance for *RiddleSense* (Part 2). The column **Quant** indicates whether the model is quantized or not.



## Chapter 8

# Conclusions - Future Work

### Conclusion

In this study, we systematically evaluated both pre-trained and fine-tuned encoders, as well as instruction-tuned large language models (LLMs), across a variety of classification tasks involving complex reasoning. Our findings demonstrate that leveraging domain-specific pre-training, combined with fine-tuning on relevant datasets, leads to significant improvements over baseline methods. Models that incorporated additional commonsense reasoning datasets during pre-training exhibited enhanced performance, highlighting the importance of capturing general reasoning capabilities that can be applied to specific tasks.

Across all tasks, the performance gap between models illustrates the influence of architecture, size, and training strategies. Pre-trained encoder models, especially those exposed to commonsense reasoning datasets, showed substantial improvements in handling intricate reasoning paths. In contrast, instruction-tuned LLMs, when not fine-tuned on domain-specific data, often underperformed relative to pre-trained models, emphasizing the need for fine-tuning even in large models to adapt effectively to task-specific challenges.

A key insight from our experiments is that multi-class classification tasks consistently outperformed binary classification tasks. This indicates that providing diverse options offers critical contextual information that guides models toward more accurate reasoning. The initial assumption that binary tasks are simpler proved to be inaccurate, as they often lack the depth of context needed for correct decision-making.

Additionally, we introduced the RISCORE technique, a novel method that augments standard few-shot learning with contextually reconstructed examples. RISCORE proved particularly effective in improving LLM performance by helping models trace appropriate reasoning paths. Notably, RISCORE consistently outperformed baseline techniques based on semantic similarity, especially in smaller models where reasoning generalization is more challenging. The method’s robustness was demonstrated through both manually curated and automatically generated contextual pairs, with the latter still producing notable performance improvements.

In conclusion, our study highlights the importance of combining pre-training on diverse reasoning datasets with task-specific fine-tuning to improve model performance on complex reasoning tasks. The introduction of RISCORE adds another layer of refinement, offering a scalable solution for enhancing the reasoning capabilities of LLMs, particularly in few-shot settings. Moving forward, further exploration of context reconstruction and reasoning pattern detection will be key to advancing LLMs’ abilities in tasks that require nuanced, multi-step problem-solving.

### Future Work

Building upon the findings of this study, several promising research directions remain to be explored in order to further enhance the capabilities of large language models (LLMs) in complex reasoning tasks. To guide

these efforts, we propose several key research questions and areas of investigation that could yield significant improvements in model performance.

## Fine-Tuning Across Datasets and Models

One important area for future work is **fine-tuning across various related datasets and models**, especially in small language models. A key research question here is:

*How can fine-tuning on diverse, domain-relevant datasets improve the generalization of smaller LLMs to complex reasoning tasks?*

By systematically exploring fine-tuning strategies across similar datasets, we can gain insights into optimizing performance for smaller models, potentially unlocking their reasoning potential without the need for extensive computational resources.

## Order of RISCORE Contextual Reconstructions

A second research question focuses on the **order of RISCORE contextually reconstructed examples and their original counterparts**. The central question here is:

*Does the sequence in which contextual reconstructions are presented affect the model's ability to follow reasoning paths more accurately?*

Investigating the impact of different presentation orders could provide practical insights into prompt engineering, helping to maximize the model's reasoning accuracy.

## Semantic Similarity Selection

Additionally, **the consequences of poor semantic similarity selection** should be addressed. Specifically, we ask:

*How does incorrect selection of semantically similar examples, when paired with context reconstructions, affect overall model performance?*

Understanding this dynamic could lead to the development of more sophisticated strategies for selecting high-quality, semantically aligned examples.

## Taxonomy for Reasoning Techniques

Another area ripe for exploration is the creation of a **deterministic framework or taxonomy** for detecting and categorizing reasoning techniques in riddles. This leads to the research question:

*Can a taxonomy of reasoning processes improve the selection of riddle examples based on shared reasoning patterns, thereby enhancing few-shot learning?*

By developing such a framework, we can better tailor exemplar selection, allowing models to more consistently apply appropriate reasoning techniques across diverse tasks.

## Human Evaluation of Context Exemplars

**Human evaluation of our automated context exemplars** is also crucial for understanding their efficacy. A guiding research question here is:

*How do human evaluators perceive the quality of automated context exemplars, and how can these insights inform the development of more deterministic, high-quality examples?*

Conducting human surveys or structured assessments will provide valuable feedback on the quality of our context reconstructions and help identify areas for improvement. This could ultimately lead to methods that rely less on LLM-generated content while maintaining the high quality of reconstructed examples.

In summary, these research questions and future investigations will provide a deeper understanding of how fine-tuning strategies, prompt structure, reasoning taxonomies, and human evaluations can collectively

---

enhance the effectiveness of LLMs in complex reasoning tasks. Through these efforts, we aim to not only advance model performance but also contribute to the broader field of artificial intelligence by improving reasoning capabilities across a range of cognitive challenges.



# Chapter 9

## Bibliography

- [1] AI@Meta. “Llama 3 Model Card”. In: (2024). URL:
- [2] Argyrou, G., Dimitriou, A., Lymperaioi, M., Filandrianos, G., and Stamou, G. *Automatic Generation of Fashion Images using Prompting in Generative Machine Learning Models*. 2024. arXiv: [2407.14944 \[cs.CV\]](#). URL:
- [3] Ba, J. L., Kiros, J. R., and Hinton, G. E. *Layer Normalization*. 2016. arXiv: [1607.06450 \[stat.ML\]](#).
- [4] Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. “On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?” In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. FAccT ’21. Virtual Event, Canada: Association for Computing Machinery, 2021, pp. 610–623. ISBN: 9781450383097. DOI: [10.1145/3442188.3445922](#). URL:
- [5] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. “A Neural Probabilistic Language Model”. In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 1137–1155. ISSN: 1532-4435.
- [6] Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. *PIQA: Reasoning about Physical Commonsense in Natural Language*. 2019. arXiv: [1911.11641 \[cs.CL\]](#).
- [7] Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. “PIQA: Reasoning about Physical Commonsense in Natural Language”. In: *Thirty-Fourth AAAI Conference on Artificial Intelligence*. 2020.
- [8] Brown, T. B. et al. *Language Models are Few-Shot Learners*. 2020. arXiv: [2005.14165 \[cs.CL\]](#). URL:
- [9] Brown, T. B. et al. *Language Models are Few-Shot Learners*. 2020. arXiv: [2005.14165 \[cs.CL\]](#).
- [10] Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. *Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge*. 2018. arXiv: [1803.05457 \[cs.AI\]](#).
- [11] Cognitive Science Laboratory, Princeton University. *WordNet - A lexical database for the English language*. , last accessed on March 8th, 2006. 2006.
- [12] Deng, M., Wang, J., Hsieh, C.-P., Wang, Y., Guo, H., Shu, T., Song, M., Xing, E. P., and Hu, Z. *RLPrompt: Optimizing Discrete Text Prompts with Reinforcement Learning*. 2022. arXiv: [2205.12548 \[cs.CL\]](#). URL:
- [13] Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. *QLoRA: Efficient Finetuning of Quantized LLMs*. 2023. arXiv: [2305.14314 \[cs.LG\]](#).
- [14] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by J. Burstein, C. Doran, and T. Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: [10.18653/v1/N19-1423](#). URL:
- [15] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *ArXiv abs/1810.04805* (2019).
- [16] Dong, Q. et al. *A Survey on In-context Learning*. 2024. arXiv: [2301.00234 \[cs.CL\]](#). URL:
- [17] Efrat, A. and Levy, O. *The Turing Test: Can Language Models Understand Instructions?* 2020. arXiv: [2010.11982 \[cs.CL\]](#). URL:

- [18] Floridi, L. and Chiriatti, M. “GPT-3: Its Nature, Scope, Limits, and Consequences”. In: *Minds and Machines* 30 (2020), pp. 681–694.
- [19] Fu, Y., Peng, H., Sabharwal, A., Clark, P., and Khot, T. *Complexity-Based Prompting for Multi-Step Reasoning*. 2023. arXiv: [2210.00720 \[cs.CL\]](#). URL:
- [20] Gendron, G., Witbrock, M., and Dobbie, G. *A Survey of Methods, Challenges and Perspectives in Causality*. 2024. arXiv: [2302.00293 \[cs.LG\]](#).
- [21] Giadikiaroglou, P., Lymperaiou, M., Filandrianos, G., and Stamou, G. *Puzzle Solving using Reasoning of Large Language Models: A Survey*. 2024. arXiv: [2402.11291 \[cs.CL\]](#).
- [22] Gordon, A., Kozareva, Z., and Roemmele, M. “SemEval-2012 Task 7: Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning”. In: *\*SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. Ed. by E. Agirre, J. Bos, M. Diab, S. Manandhar, Y. Marton, and D. Yuret. Montréal, Canada: Association for Computational Linguistics, June 2012, pp. 394–398. URL:
- [23] Grasedyck, L., Kressner, D., and Tobler, C. *A literature survey of low-rank tensor approximation techniques*. 2013. arXiv: [1302.7121 \[math.NA\]](#).
- [24] Grigoriadou, N., Lymperaiou, M., Filandrianos, G., and Stamou, G. “AILS-NTUA at SemEval-2024 Task 6: Efficient model tuning for hallucination detection and analysis”. In: *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Ed. by A. K. Ojha, A. S. Doğruöz, H. Tayyar Madabushi, G. Da San Martino, S. Rosenthal, and A. Rosá. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 1549–1560. DOI: [10.18653/v1/2024.semeval-1.222](#). URL:
- [25] Gugger, S., Debut, L., Wolf, T., Schmid, P., Mueller, Z., Mangrulkar, S., Sun, M., and Bossan, B. *Accelerate: Training and inference at scale made simple, efficient and adaptable*. 2022.
- [26] Gunasekar, S. et al. *Textbooks Are All You Need*. 2023. arXiv: [2306.11644 \[cs.CL\]](#).
- [27] Han, S. J., Ransom, K. J., Perfors, A., and Kemp, C. “Inductive reasoning in humans and large language models”. In: *Cognitive Systems Research* 83 (2024), p. 101155. ISSN: 1389-0417. DOI: <https://doi.org/10.1016/j.cogsys.2023.101155>. URL:
- [28] He, K., Zhang, X., Ren, S., and Sun, J. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: [10.1109/CVPR.2016.90](#).
- [29] He, P., Gao, J., and Chen, W. *DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing*. 2023. arXiv: [2111.09543 \[cs.CL\]](#).
- [30] He, P., Liu, X., Gao, J., and Chen, W. *DeBERTa: Decoding-enhanced BERT with Disentangled Attention*. 2021. arXiv: [2006.03654 \[cs.CL\]](#).
- [31] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: [2106.09685 \[cs.CL\]](#).
- [32] Jiang, A. Q. et al. *Mistral 7B*. 2023. arXiv: [2310.06825 \[cs.CL\]](#). URL:
- [33] Jiang, A. Q. et al. *Mistral 7B*. 2023. arXiv: [2310.06825 \[cs.CL\]](#).
- [34] Jiang, A. Q. et al. *Mistral of Experts*. 2024. arXiv: [2401.04088 \[cs.LG\]](#).
- [35] Jiang, Y., Ilievski, F., and Ma, K. “SemEval-2024 Task 9: BRAINTEASER: A Novel Task Defying Common Sense”. In: *Proceedings of the 18th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2024.
- [36] Jiang, Y., Ilievski, F., and Ma, K. “SemEval-2024 Task 9: BRAINTEASER: A Novel Task Defying Common Sense”. In: *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Ed. by A. K. Ojha, A. S. Doğruöz, H. Tayyar Madabushi, G. Da San Martino, S. Rosenthal, and A. Rosá. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 1994–2008. DOI: [10.18653/v1/2024.semeval-1.274](#). URL:
- [37] Jiang, Y., Ilievski, F., Ma, K., and Sourati, Z. “BRAINTEASER: Lateral Thinking Puzzles for Large Language Models”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023.
- [38] Jiang, Y., Ilievski, F., Ma, K., and Sourati, Z. “BRAINTEASER: Lateral Thinking Puzzles for Large Language Models”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by H. Bouamor, J. Pino, and K. Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 14317–14332. DOI: [10.18653/v1/2023.emnlp-main.885](#). URL:

- 
- [39] Kauf, C., Ivanova, A. A., Rambelli, G., Chersoni, E., She, J. S., Chowdhury, Z., Fedorenko, E., and Lenci, A. “Event knowledge in large language models: the gap between the impossible and the unlikely”. In: *ArXiv abs/2212.01488* (2022).
- [40] Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. *Large Language Models are Zero-Shot Reasoners*. 2023. arXiv: [2205.11916 \[cs.CL\]](#). URL:
- [41] Kritharoula, A., Lymperaioi, M., and Stamou, G. *Language Models as Knowledge Bases for Visual Word Sense Disambiguation*. 2023. arXiv: [2310.01960 \[cs.CL\]](#). URL:
- [42] Kritharoula, A., Lymperaioi, M., and Stamou, G. “Large Language Models and Multimodal Retrieval for Visual Word Sense Disambiguation”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023, pp. 13053–13077. DOI: [10.18653/v1/2023.emnlp-main.807](#). URL:
- [43] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: [1910.13461 \[cs.CL\]](#). URL:
- [44] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: [1910.13461 \[cs.CL\]](#).
- [45] Li, Y., Liang, Y., and Risteski, A. *Recovery guarantee of weighted low-rank approximation via alternating minimization*. 2016. arXiv: [1602.02262 \[cs.LG\]](#).
- [46] Li, Y., Ma, T., and Zhang, H. *Algorithmic Regularization in Over-parameterized Matrix Sensing and Neural Networks with Quadratic Activations*. 2019. arXiv: [1712.09203 \[cs.LG\]](#).
- [47] Lin, B. Y., Wu, Z., Yang, Y., Lee, D.-H., and Ren, X. *RiddleSense: Reasoning about Riddle Questions Featuring Linguistic Creativity and Commonsense Knowledge*. 2021. arXiv: [2101.00376 \[cs.CL\]](#).
- [48] Lin, B. Y., Wu, Z., Yang, Y., Lee, D.-H., and Ren, X. “RiddleSense: Reasoning about Riddle Questions Featuring Linguistic Creativity and Commonsense Knowledge”. In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Ed. by C. Zong, F. Xia, W. Li, and R. Navigli. Online: Association for Computational Linguistics, Aug. 2021, pp. 1504–1515. DOI: [10.18653/v1/2021.findings-acl.131](#). URL:
- [49] Liu, H., Wang, T., Lang, C., Feng, S., Jin, Y., and Li, Y. “GLAN: A graph-based linear assignment network”. In: *Pattern Recognition* 155 (June 2024), p. 110694. DOI: [10.1016/j.patcog.2024.110694](#).
- [50] Liu, J., Shen, D., Zhang, Y., Dolan, B., Carin, L., and Chen, W. “What Makes Good In-Context Examples for GPT-3?” In: *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*. Ed. by E. Agirre, M. Apidianaki, and I. Vulić. Dublin, Ireland and Online: Association for Computational Linguistics, May 2022, pp. 100–114. DOI: [10.18653/v1/2022.deeLIO-1.10](#). URL:
- [51] Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. *Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing*. 2021. arXiv: [2107.13586 \[cs.CL\]](#). URL:
- [52] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: [1907.11692 \[cs.CL\]](#).
- [53] Liu, Y., Jin, M., Pan, S., Zhou, C., Zheng, Y., Xia, F., and Yu, P. “Graph self-supervised learning: A survey”. In: *IEEE Transactions on Knowledge and Data Engineering* (2022).
- [54] Lu, P., Qiu, L., Yu, W., Welleck, S., and Chang, K.-W. “A Survey of Deep Learning for Mathematical Reasoning”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by A. Rogers, J. Boyd-Graber, and N. Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 14605–14631. DOI: [10.18653/v1/2023.acl-long.817](#). URL:
- [55] Luo, L. et al. *Improve Mathematical Reasoning in Language Models by Automated Process Supervision*. 2024. arXiv: [2406.06592 \[cs.CL\]](#). URL:
- [56] Lymperopoulos, D., Lymperaioi, M., Filandrianos, G., and Stamou, G. “Optimal and efficient text counterfactuals using Graph Neural Networks”. In: *The 7th BlackboxNLP Workshop*. 2024. URL:
- [57] Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., and Bossan, B. *PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods*. 2022.
-

- [58] Oymak, S., Fabian, Z., Li, M., and Soltanolkotabi, M. *Generalization Guarantees for Neural Networks via Harnessing the Low-rank Structure of the Jacobian*. 2019. arXiv: [1906.05392](#) [cs.LG].
- [59] Panagiotopoulos, I., Filandrianos, G., Lymperaiou, M., and Stamou, G. “AILS-NTUA at SemEval-2024 Task 9: Cracking Brain Teasers: Transformer Models for Lateral Thinking Puzzles”. In: *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Ed. by A. K. Ojha, A. S. Doğruöz, H. Tayyar Madabushi, G. Da San Martino, S. Rosenthal, and A. Rosá. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 1733–1746. DOI: [10.18653/v1/2024.semeval-1.248](#). URL: [2024.semeval-1.248](#).
- [60] Panagiotopoulos, I., Filandrianos, G., Lymperaiou, M., and Stamou, G. *RISCORE: Enhancing In-Context Riddle Solving in Language Models through Context-Reconstructed Example Augmentation*. 2024. arXiv: [2409.16383](#) [cs.CL]. URL: [2409.16383](#).
- [61] Papadimitriou, C., Filandrianos, G., Lymperaiou, M., and Stamou, G. *Masked Generative Story Transformer with Character Guidance and Caption Augmentation*. 2024. arXiv: [2403.08502](#) [cs.CV]. URL: [2403.08502](#).
- [62] Petersen, M. R. and Plas, L. van der. *Can language models learn analogical reasoning? Investigating training objectives and comparisons to human performance*. 2023. arXiv: [2310.05597](#) [cs.CL].
- [63] Qiao, S., Ou, Y., Zhang, N., Chen, X., Yao, Y., Deng, S., Tan, C., Huang, F., and Chen, H. “Reasoning with Language Model Prompting: A Survey”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by A. Rogers, J. Boyd-Graber, and N. Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 5368–5393. DOI: [10.18653/v1/2023.acl-long.294](#). URL: [10.18653/v1/2023.acl-long.294](#).
- [64] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8 (2019), p. 9.
- [65] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2020. arXiv: [1910.10683](#) [cs.LG].
- [66] Reimers, N. and Gurevych, I. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: [2019.emnlp-main](#).
- [67] Richardson, C. and Heck, L. *Commonsense Reasoning for Conversational AI: A Survey of the State of the Art*. 2023. arXiv: [2302.07926](#) [cs.CL].
- [68] Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. *WinoGrande: An Adversarial Winograd Schema Challenge at Scale*. 2019. arXiv: [1907.10641](#) [cs.CL].
- [69] Sanyal, S., Singh, H., and Ren, X. “FaIRR: Faithful and Robust Deductive Reasoning over Natural Language”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by S. Muresan, P. Nakov, and A. Villavicencio. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 1075–1093. DOI: [10.18653/v1/2022.acl-long.77](#). URL: [2022.acl-long.77](#).
- [70] Sap, M., Le Bras, R., Allaway, E., Bhagavatula, C., Lourie, N., Rashkin, H., Roof, B., Smith, N. A., and Choi, Y. “ATOMIC: an atlas of machine commonsense for if-then reasoning”. In: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI’19/IAAI’19/EAAI’19. Honolulu, Hawaii, USA: AAAI Press, 2019. ISBN: 978-1-57735-809-1. DOI: [10.1609/aaai.v33i01.33013027](#). URL: [10.1609/aaai.v33i01.33013027](#).
- [71] Sap, M., Shwartz, V., Bosselut, A., Choi, Y., and Roth, D. “Commonsense Reasoning for Natural Language Processing”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*. Ed. by A. Savary and Y. Zhang. Online: Association for Computational Linguistics, July 2020, pp. 27–33. DOI: [10.18653/v1/2020.acl-tutorials.7](#). URL: [10.18653/v1/2020.acl-tutorials.7](#).
- [72] Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, E. H.-h., Scharli, N., and Zhou, D. “Large Language Models Can Be Easily Distracted by Irrelevant Context”. In: *ArXiv abs/2302.00093* (2023).
- [73] Sileo, D. “tasksource: Structured Dataset Preprocessing Annotations for Frictionless Extreme Multi-Task Learning and Evaluation”. In: *arXiv preprint arXiv:2301.05948* (2023). URL: [2301.05948](#).
- [74] Speer, R., Chin, J., and Havasi, C. “ConceptNet 5.5: an open multilingual graph of general knowledge”. In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI’17. San Francisco, California, USA: AAAI Press, 2017, pp. 4444–4451.

- 
- [75] Srivastava, A. et al. *Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models*. 2023. arXiv: [2206.04615 \[cs.CL\]](#).
- [76] Sultan, O. and Shahaf, D. “Life is a Circus and We are the Clowns: Automatically Finding Analogies between Situations and Processes”. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Y. Goldberg, Z. Kozareva, and Y. Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 3547–3562. DOI: [10.18653/v1/2022.emnlp-main.232](#). URL:
- [77] Talmor, A., Herzig, J., Lourie, N., and Berant, J. “CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by J. Burstein, C. Doran, and T. Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4149–4158. DOI: [10.18653/v1/N19-1421](#). URL:
- [78] Thomas, K., Filandrianos, G., Lymperaious, M., Zerva, C., and Stamou, G. “*I Never Said That*: A dataset, taxonomy and baselines on response clarity classification”. 2024. arXiv: [2409.13879 \[cs.CL\]](#). URL:
- [79] Touvron, H. et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: [2307.09288 \[cs.CL\]](#).
- [80] Touvron, H. et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: [2302.13971 \[cs.CL\]](#).
- [81] Tyen, G., Mansoor, H., Cărbune, V., Chen, P., and Mak, T. *LLMs cannot find reasoning errors, but can correct them!* 2024. arXiv: [2311.08516 \[cs.AI\]](#).
- [82] Vashishtha, S., Poliak, A., Lal, Y. K., Van Durme, B., and White, A. S. “Temporal Reasoning in Natural Language Inference”. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Ed. by T. Cohn, Y. He, and Y. Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 4070–4078. DOI: [10.18653/v1/2020.findings-emnlp.363](#). URL:
- [83] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL:
- [84] Wang, C., Liang, S., Jin, Y., Wang, Y., Zhu, X., and Zhang, Y. “SemEval-2020 Task 4: Commonsense Validation and Explanation”. In: *Proceedings of The 14th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2020.
- [85] Wang, L., Yang, N., and Wei, F. “Learning to Retrieve In-Context Examples for Large Language Models”. In: *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Y. Graham and M. Purver. St. Julian’s, Malta: Association for Computational Linguistics, Mar. 2024, pp. 1752–1767. URL:
- [86] Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. “Emergent abilities of large language models”. In: *arXiv preprint arXiv:2206.07682* (2022).
- [87] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: [2201.11903 \[cs.CL\]](#). URL:
- [88] Wei, J. et al. *Emergent Abilities of Large Language Models*. 2022. arXiv: [2206.07682 \[cs.CL\]](#). URL:
- [89] Wei, J., Tan, C., Gao, Z., Sun, L., Li, S., Yu, B., Guo, R., and Li, S. Z. *Enhancing Human-like Multi-Modal Reasoning: A New Challenging Dataset and Comprehensive Framework*. 2023. arXiv: [2307.12626 \[cs.AI\]](#).
- [90] Wen, Y., Jain, N., Kirchenbauer, J., Goldblum, M., Geiping, J., and Goldstein, T. *Hard Prompts Made Easy: Gradient-Based Discrete Optimization for Prompt Tuning and Discovery*. 2023. arXiv: [2302.03668 \[cs.LG\]](#). URL:
- [91] Werra, L. von, Belkada, Y., Tunstall, L., Beeching, E., Thrush, T., Lambert, N., and Huang, S. *TRL: Transformer Reinforcement Learning*. 2020.
- [92] Wikipedia contributors. *Semi-supervised learning* — *Wikipedia, The Free Encyclopedia*. [Online; accessed 22-September-2022]. 2022.
-

- [93] Wolf, T. et al. *HuggingFace’s Transformers: State-of-the-art Natural Language Processing*. 2020. arXiv: [1910.03771](#) [[cs.CL](#)].
- [94] Wolf, T. et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. URL:
- [95] Yang, A. et al. *Qwen2 Technical Report*. 2024. arXiv: [2407.10671](#) [[cs.CL](#)]. URL:
- [96] Yang, Z., Du, X., Mao, R., Ni, J., and Cambria, E. *Logical Reasoning over Natural Language as Knowledge Representation: A Survey*. 2023. arXiv: [2303.12023](#) [[cs.CL](#)].
- [97] Yuan, Z., Hu, S., Vulic, I., Korhonen, A., and Meng, Z. “Can Pretrained Language Models (Yet) Reason Deductively?” In: *ArXiv abs/2210.06442* (2022).
- [98] Zhang, M., Press, O., Merrill, W., Liu, A., and Smith, N. A. *How Language Model Hallucinations Can Snowball*. 2023. arXiv: [2305.13534](#) [[cs.CL](#)].
- [99] Zhang, X. et al. *mGTE: Generalized Long-Context Text Representation and Reranking Models for Multilingual Text Retrieval*. 2024. arXiv: [2407.19669](#) [[cs.CL](#)]. URL:
- [100] Zhang, Z., Zhang, A., Li, M., and Smola, A. *Automatic Chain of Thought Prompting in Large Language Models*. 2022. arXiv: [2210.03493](#) [[cs.CL](#)]. URL:
- [101] Zhao, W., Chiu, J., Cardie, C., and Rush, A. “Abductive Commonsense Reasoning Exploiting Mutually Exclusive Explanations”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by A. Rogers, J. Boyd-Graber, and N. Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 14883–14896. DOI: [10.18653/v1/2023.acl-long.831](#). URL: