Prompting Language Models for Abductive Reasoning Tasks

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Kristýna Onderková
School of Computer Science
University of Galway

Supervisor(s)
Dr. Matthias Nickles

In partial fulfillment of the requirements for the degree of
MSc in Computer Science (Data Analytics)
August 31, 2023

Acknowledgement

I would like to thank my supervisor Dr. Matthias Nickles for his support, guidance and invaluable insights and recommendations throughout the journey. I am deeply appreciative of the diverse literature and valuable resources he introduced me to, which profoundly enriched this thesis.

I also express my thanks to all my teachers for their exceptional lectures, which have significantly contributed to my knowledge. To my fellow schoolmates, your support and camaraderie have made this educational experience truly enjoyable. Last but not least, I thank my family for their enduring patience with me.

Abstract

The development in language models enables their application in diverse tasks through prompt-based techniques, allowing for widespread adoption. However, these models often face limitations in reasoning. Interestingly, abductive reasoning remains underexplored within this context, despite its integral role in logic, capacity to handle incomplete information, and commonplace human use.

This motivates our work to explore how various models handle abductive tasks and optimize prompts for optimal results. Among the four studied language models, only Flan-T5 exhibits abductive reasoning capabilities. Despite employing various prompting strategies, performance is not improved, highlighting the need for more research. Given the other models' struggles with abductive tasks, we conduct a comparative assessment of other key prompting-related abilities.

Leveraging acquired insights, we construct a data generation pipeline using a sequence of prompts, inspired by the traditional natural language generation approach. We demonstrate the utility of abductive reasoning in selecting relevant messages for generation. Although we endeavour to devise an ordering plan for these messages, the constraints of the utilized models limit our success. Nevertheless, our approach shows potential for data-to-text generation and suggests the potential of incorporating symbolic planning into language models.

Keywords: Language Models, Prompting, Abductive Reasoning, Data-to-Text Generation, Natural Language Generation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research Topic and Motivation

Large Language Models (LLMs), based on transformer architecture [2], have recently become widely used by the general public thanks to the simplicity of their natural language interface and their accessibility through platforms like ChatGPT [3]. These models possess remarkable emergent properties such as following instructions, learning tasks from a few examples, and multistep reasoning [4]. Therefore, they are also of high interest to researchers and we may presume that LLMs will continue to grow in importance in the upcoming years.

However their shortcomings as the lack of exact knowledge and inability to plan ahead and properly reason [5, 6] become increasingly more visible and serious. Their capabilities may be misunderstood and misused by the users [5]. Those shortcomings and lack of explainability hinder their use in many rigorous areas like meteorology, law, medicine, or journalism, where risk of generating incorrect information poses a significant concern [7]. Additionally, the cost of training and customizing LLMs [8] makes it challenging to equip them with expert knowledge tailored to specific domains.

To overcome those limitations, various approaches to augment the LLMs are explored [9]. One approach involves training and fine-tuning the model using generated data that contains factual knowledge and correct reasoning examples [10]. Another approach is to utilize external formal reasoners or calculators [9]. LLMs can also directly generate programs to solve specific tasks [11] and knowledge bases can be used to supplement the model with factual information [12]. Moreover, the model's emerging capabilities enable the use of diverse prompting techniques, like to mitigate mentioned limitations without requiring costly training. These techniques encompass zero-shot [13], few-shot, chain of thought (CoT) [14] prompting and many more strategies and variations are being developed [9].

By employing prompting techniques, neural networks in LLMs can be combined with symbolic semantic representation methods. This integration could provide LLMs with expert knowledge and reasoning capabilities tailored to specific tasks and provide greater control over the model's results [15]. Reasoning can be categorised into deductive, inductive, and abductive tasks [16]. While deductive reasoning has been extensively studied [17, 10, 18, 19, 20], there is still significant potential for investigating inductive and abductive reasoning [21], particularly in tasks related to Natural Language Inference and Generation [22]. Currently, abductive tasks primarily focus on explanation classification, where the model assesses the plausibility of hypotheses based on given observations [23] and employs abductive reasoning to predict future events [24].

LLMs have the potential to benefit various fields by generating data-to-text (D2T) output, including popular D2T applications like weather reports in meteorology [25, 26, 27, 28]. Nonetheless, addressing issues such as hallucinations and omissions [29] in these models is crucial to ensure the accuracy of generated texts. Classical Natural Language Generation (NLG) methods, which use templates and rule-based approaches with a pipeline architecture with various stages including planning [26], avoid such issues but are expensive to create. To mitigate said concerns, researchers are exploring content selection and planning with LLMs [30]. One approach involves a stage that translates data into meaning representations and those are then used as input for LLMs to generate the desired texts [31]. Considering that abductive reasoning seeks the most plausible explanations based on given data and rules and is used in automated planning [32, 16], it holds potential in planning D2T generation through the use of prompting LLMs.

1.2 Research Questions

This thesis seeks to investigate the following research questions:

- ***RQ1***: How can abductive reasoning be implemented in language models through prompting?
- ***RQ2***: When prompted with abductive tasks, how do different language models compare?
- ***RQ3***: Can abductive reasoning be effectively employed for the purpose of content selection and discourse planning in data-to-text generation?

1.3 Structure

This thesis is organised into five chapters. Chapter 1 outlines the topic, motivation and research aims for this work. Chapter 2 establishes the terminology, provides some fundamental knowledge about language models and abductive reasoning and explores the relevant literature. Chapter 3 presents the used approach, employed techniques, used datasets and conducted experiments. Chapter 4 then summarizes the results, analyzes them and provides interpretation. Finally, Chapter 5 discusses the achieved results and proposes potential future work.

Chapter 2

Background and Related Work

2.1 Language Models

The rise of computers led to Natural Language Processing (NLP) emerging as a distinct discipline, highlighting language's fundamental role in human communication. Moreove, computer understanding of language holds crucial implications for artificial intelligence, exemplified by the Turing test [33] as an intelligence benchmark. With algorithmic advancements, computational power, and data availability, Language Models have gained prominence in this field.

Early NLP systems from the 1950s relied on symbolic approaches based on grammatical rules and dictionaries. However, creating these systems is laborious, and capturing intricate patterns is challenging. The advent of machine learning led to led to development of statistical models that calculate the probability of word sequences (n-grams) based on a given corpus, showing remarkable utility. For instance, the renowned Hidden Markov Model [36] incorporates hidden variables commonly represented by parts of speech. However, a key limitation of these models is their inability to capture long-distance dependencies.

Another transformation came with the rise of neural network architectures.

Recurrent Neural Networks (RNNs) [37], designed for sequential data processing, demonstrated great suitability for language applications. However, they faced challenges such as vanishing and exploding gradients, hindering their ability to capture long-distance dependencies. This limitation was mitigated through the introduction of the Long Short-Term Memory (LSTM) architecture [38].

The next advancement was Deep Neural Networks, employing large number of hidden layers, alongside the transformer architecture with an attention mechanism [2]. This attention enables effective focus on relevant contextual information, addressing long-distance dependencies. Recently, language models have undergone remarkable growth in terms of layers and parameters, giving rise to LLMs. Those models require dozens or hundred of billions parameters to manifest specific emergent abilities [4].

2.1.1 Prompting Large Language Models

One notable capability of LLMs is their ability to interpret natural language instructions to perform tasks, without the need for additional training or parameter updates [4]. This advantage reduces the requirement for large data collection and expensive training for specific tasks [8]. However, their performance tends to lag behind the specially trained state-of-the-art models [8]. To address this limitation, a range of prompting techniques and strategies are being developed [15]. This section introduces techniques employed to enhance LMs specifically for reasoning, aligning with the focus of this work.

In a standard LM prompting approach described in [21], the objective is to maximize the likelihood of answer A for a given question Q, with prompt T:

$$p(A|T,Q) = \prod_{i=1}^{|A|} p_{LM}(a_i|T,Q,a_{<i}) \tag{2.1}$$

where $|A|$ represents the length of the answer, a_i denotes the i-th token of the answer, and p_{LM} the probabilistic model.

Prompts in LMs often rely on engineered templates. A template consists of a textual string with two slots: one for the input x and the other for the answer z; for example "Finish: [x] English [z]" [15]. By filling in the template with the input x, the prompt is generated. Selecting the suitable template is known as *Prompt Template Engineering*, which can be accomplished manually or automatically through techniques like paraphrasing, mining, or generation [15]. Additionally, *Prompt Answer Engineering* [15] focuses on specifying the desired format of the output. Prominent prompt techniques include:

- Few-shot prompting [8] involves providing the LM with a few text exemplars to learn a specific task, such as "English: Hi Czech: Ahoj; English: Bye Czech: ". However, this can result in higher token usage increasing cost.
- Zero-shot prompting [8], involves providing a task instruction to the LM, like "Translate English to Czech: Hi $\rightarrow$ ". In this approach, it is crucial to provide a detailed and precise task description [13].
- Chain-of-Thought (CoT) [14] introduces a prompt structure that adds to the exemplar's question and answer a sequence of reasoning steps, leading to the solution. For zero-shot applications, only the phrase "Let's think step by step" is appended to the prompt's question [13]. Another promising variation is Chain-of-Symbol [39], which improves accuracy and reduces token usage by reducing natural language to symbolic representations.

Due to the field's novelty, there is currently limited consensus on technique categorization. This thesis will distinguish two main categories: *pure prompting* methods, employed in this research, and *enhanced prompting* methods, as shown in Figure 2.1. *Enhanced methods* encompass approaches such as the use of external engines [21], which employ code interpreters or tools through APIs, as well as techniques that involve fine-tuning the model, such as *prompt-aware training methods* [15], *self-optimization*, and *iterative optimization* methods [21].

The *pure prompting* methods can be further classified into *single* and *multiple* prompts, where the latter involves multiple input-output pairs. While [21] categorizes *prompt engineering* into *single* and *multi-stage* techniques, [15] places *multi-prompt learning methods* alongside *template engineering* and *prompt answer engineering*, both of which are single stage. Conversely, [40] does not differentiate based on the number of prompts employed.

Study [21] asserts that *single-stage* methods aim to improve prompt quality, aligning with *Prompt Template Engineering* in [15]. However, [21] includes zero-shot, few-shot, and CoT prompting, while [15] considers few-shot prompting as a *multi-prompt* approach. Both studies highlight prompt sensitivity to the choice of exemplars, and [15] introduces *Prompt Augmentation* techniques to enhance robustness through exemplar selection and ordering.

Another approach to categorizing single prompts is proposed by [9], which distinguishes between *naive prompting*, where the answer directly follows the input (e.g., few-shot and zero-shot prompting), and *elicitive prompting*, where the model is encouraged to generate intermediate steps before reaching the final answer to enhance reasoning without the use of multiple prompts (e.g., CoT).

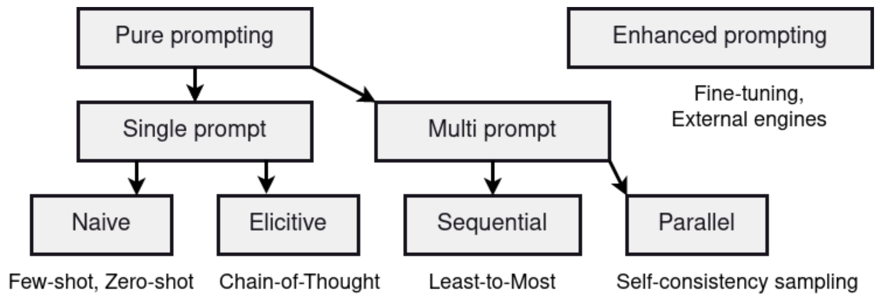


Figure 2.1: Proposed taxonomy of prompt techniques with examples

This aligns with the category *CoT and its variants* in [40].

To enhance clarity, we suggest categorizing *multi-prompt* methods into *sequential* and *parallel* approaches (Figure 2.1). Study [21] describes two ways to create (parallel) *multi-stage* prompt: generating distinct new prompts at each stage, or adding the output to the entire preceding context (T, Q, A). Similarly, *Problem Decomposition* [40] and *Recursive Prompting* [9] both decompose problems into sub-problems and solve them (based on Least-to-Most prompting [41]). This aligns with *prompt composition* [15], combining sub-prompts into composite prompts, hence the confusing terminology. Furthermore, *prompt decomposition* methods [15] provide multiple answers to a single question, which is particularly useful for sequence labeling tasks.

In *parallel multi-prompt* methods, several approaches can be included: *Ensemble methods* [15], *Ensemble optimization* [21] and *Rationale Engineering* [40]. These approaches include self-consistency sampling [42], where multiple prompts with different reasoning paths are provided to a LM. By aggregating the answers, the most likely one is selected, enhancing model robustness. Study [21] also explores verifiers for each reasoning step. Tree-of-Thoughts (ToT) [43], an extension of CoT, similarly explores multiple reasoning steps. Potential next steps in reasoning are generated through CoT or by prompting ”propose a prompt”, and heuristically evaluated. These steps are explored using either breadth-first search(following promising steps to a certain tree depth) or a depth-first search (pursuing the most promising path and backtracking if unsuccessful).

2.2 Abductive Reasoning

Abduction, along with deduction and induction, is a form of logical reasoning [44]. While deduction derives conclusions from premises, abduction works in the opposite direction by selecting or generating the most likely explanation that aligns with the available (and also hypothetical) evidence [44]. Abduction and induction are forms of defeasible inference, distinct from deduction, as their conclusions may not be deductively valid even with valid premises [17, 23]. This allows to incorporate new information and refine conclusions, making it valuable for handling incomplete observations [22, 40]. Therefore, abduction plays a significant role in everyday human reasoning, counterfactual reasoning, and understanding narratives [22]. It stands out as a unique logical operation that introduces new ideas, unlike other operations that encompass all the information in their premises [22]. Additionally, abduction is a powerful tool for extracting latent knowledge from LMs [45]. However, research on abduction in natural language inference and generation is relatively limited [22]. Overall, the exploration of defeasible reasoning lags behind that of deduction and deserves further investigation [17].

The formal definition of a logic abduction is as follows [46]: Given a logical theory T that represents a specific domain and a set of observations O, find and explanation H (hypothesis) for O that satisfies following criteria:

- hypothesis is consistent with theory: $T \cup H$ is consistent
- and logically entails the observations: $T \cup H \models O$

The prominent study [22] explores abduction in LM and introduces the ART dataset with two tasks. In *αNLI* task the model selects the most likely explanation from two hypotheses (plausible and implausible/less plausible) for two subsequent observations (past and future) [22], while for the αNLG task the explanation is generated to maximize plausibility [22]. Their model achieved 68.9%, respective 45% accuracy, notably below human performance of 91.4% and 96%. These results highlight the challenges in seemingly simple tasks. Notably, discarding implausible hypotheses is simpler than comparing the likelihood of two plausible hypotheses [22]. In a separate evaluation [47], ChatGPT impressively achieved 86.7% accuracy on a 30 αNLI questions. This study also revealed that ChatGPT excelled in deduction, followed by abduction and then induction [47]. For clarity, an abductive task example from the ART [22] train set is given:

Observation 1:	Tyler wanted to have a balloon party with tons of balloons.
Observation 2:	It had been a very fun day.
Correct hypothesis:	His parents made that happen.
Incorrect hypothesis:	His parents said no balloons.

Abduction plays a vital role in various tasks. It was employed in LM to propose possible causes for predictions made by an event sequence model [24]. Guided by few-shot prompting on annotated demonstrations, the LM suggested causes, which were then evaluated for feasibility on real historical data [24]. Abduction was also used to consider a wider range of scenarios, generating a recursive tree of explanations for both true and false answers to a question [45]. Subsequently, a SAT solver was employed to eliminate contradictory candidates and determine the best answer [45]. A broader survey [17] explored abduction in backward reasoning, which mirrors how people search for proofs by deriving potential premises from the goal and gradually proving or disproving sub-goals. This can also assist in answering multi-hop questions [17]. Furthermore, abduction can be combined with deduction modules to generate proofs, enabling mutual verification of results, and assessing the plausibility of hypotheses [17, 23].

2.2.1 Event Calculus

To adapt abductive reasoning for planning, a logical formalism of event calculus is often used to construct a hierarchical planning system [32]. This formalism, describing events and their effects, employs first-order logic and extends it with temporal reasoning, allowing for formal reasoning about the effects of events and their consequences [16]. The plan, as a sequence of actions Ω that leat to the goal

Γ, must be consistent with domain description Σ (outcomes of actions), initial state Δ_0 and the event calculus formalism [32]. Implementation or verification of the planning can be then facilitated through logic programming [32].

The central components of event calculus consist of *events* denoted as α, *fluents* β, and *time* points τ [16]. Fluents are variables that undergo changes, such as *temperature*, or binary variables like *rain*, which can be either true or false [16]. Events modify the state of fluents at specific point in time [16]. Predicates of event calculus allow for the description of a narrative, like specification of fluent's initial value (InitiallyP(β)), indication of when event occures (Happens(α, τ)), when event changes value of fluent (Initiates(α, β, τ), Terminates(α, β, τ)), and others [16]. Moreover, event calculus defines axioms that interrelate these predicates, enabling a comprehensive description of specific scenarios [16].

2.3 Natural Language Generation

Natural Language Generation (NLG) [26] is a subfield of NLP dedicated to generating content in natural language, including tasks such as report generation, chatbot development, and image captioning. Prior the rise of end-to-end (E2E) architectures with deep neural models, the traditional approach used a template and rule-based pipelines with distinct stages. This section provides an overview of the pipeline architecture and its application in neural language models.

The classical architecture of an NLG system involves three main stages with six tasks [26]. In the first stage, *content determination* selects the desired information and generates a set of messages to communicate [26]. *Discourse planning* structures these messages into *text plan* [26]. The second stage involves combining individual messages into coherent sentences (*sentence aggregation*), selecting domain-specific words and phrases (*lexicalization*), and generation of *referring* *expressions* to replace certain entities with noun phrases [26]. This forms the *sentence plan*, which is further refined in the final stage through *linguistic realization*, where grammar rules are applied to produce coherent *surface text* [26]. As the latter stages are well-suited for LMs, this work focuses on the plan creation.

The content determination is highly application dependent, as it involves the selection of information [26]. This process includes filtering, summarizing, and processing input data [26]. Two main approaches are employed: *deep reasoning* determines user intentions and information needs using extensive knowledge and sophisticated reasoning, and *domain-specific* approaches, that are mostly used due to their lower complexity [26]. These approaches involve analysing a given corpus by breaking texts into phrases, grouping them into similar messages, and identifying the conditions under which these messages appear [26]. The resulting analysis is refined into a set of rules and discussed with domain experts [26].

Discourse planning relies on understanding the desired message structure [26]. In more general *planning based* approaches, a plan is devised to connect the available messages to a predefined goal structure [26]. Domain-specific *schema-based* approaches are generally used for texts with a relatively small number of patterns [26]. These approaches extract patterns from the corpus and employ them to develop specific schemas that guide the construction of text by using smaller schemas and atomic messages [26]. These schemas often invoke content determination on demand [26]. The resulting *text plan* is typically represented as a tree structure, using logical formulas or templates, with messages as leaves and nodes indicating their grouping [26]. Discourse relations, such as *contrast*, *exception*, or *elaboration*, can be specified to describe relationships between messages [26].

One of the challenges in D2T generation with structured data is how to format the input for LMs, which are primarily designed for sequential data [48]. Table serialization often involves template-based methods, while knowledge graphs are linearized using Abstract Meaning Representations (MR) to create sequences of triples [48]. For instance, the sentence *"Friday will be sunny"* can be represented in flat MR as *"condition[sunny] time[Friday]"* [30], or in triple form as *(Friday, hasWeather, sunny).* Study [31] proposed using tree-structured MRs for planning in dialog systems, employing discourse relations to structure them. This approach imrpoved semantic correctness and expressiveness [31]. In contrast, flat linearization may generate unfaithful texts that contradict the input facts [48].

Researchers explore promising pipeline approaches for LMs, particularly focusing on the planning stage, using RDF triples for generation [49, 50]. Study [49] compared pipeline and end-to-end architectures using neural LMs, founding that E2E models struggle with generalization on unseen data and exhibit more hallucinations. In [50], a purely symbolic text planning stage was followed by LM-based text generation, ensuring fluency and explicit control over the output. The symbolic stage creates unambiguous ordered trees for both discourse and sentence planning [50]. This setup substantially outperformed E2E architectures in terms of input faithfulness. Study [30] trained distinct modules for planning and content selection based on input and required conditions. This lead to improved text quality, better ordering, and reduced redundancy in the output [30].

There is also focus on investigating controlled generation, primarily by using prompt-tuning methods rather than pure prompting [51]. Control prompts provide instructions to guide the LM in generating desired texts [51]. Three main objectives of control include semantic control, enabling manipulation of sentiment, emotions, or text style; structural control, governing the output format and facilitating text ordering; and lexical control, allowing the inclusion of specific words or phrases [51]. Techniques for achieving control include prefix prompts, length specifications, style tokens, domain tags, or relation triples [15].

Chapter 3

Methodology and Experiments

3.1 Implementation

We designed various prompts using techniques as zero-shot, few-shot, elicitive and multi-prompting to address *RQ1*. For answering *RQ2* we used four different LMs: *GPT-2*, *LLaMA*, *Falcon* and *Flan-T5*. Finally, a data-to-text generation pipeline was devised to address *RQ3*.

Python was chosen for implementation due to its rich library support of data handling (os, json, random, pandas, getpass), model frameworks (numpy, PyTorch, transformers, LangChain), and evaluation tools (SciPy, sentence transformers, difflib). Development took place in Jupyter notebooks for swift iteration, result exploration, and structuring with Markdown. Notably, transformers [52] library configured LMs via pipelines, and LangChain framework [53] aided in designing and chaining prompts. The code is accessible on GitHub [54].

3.1.1 Language models

Utilized LMs were obtained from the HuggingFace portal [52]. Due to large model sizes, three computation approaches were used. Compact models like *Flan-T5*

[55] *small*, *base*, and *large* were executed locally, mainly for code and prompt prototyping. Larger models such as *Flan-T5 XL*, *LLaMA 3B* [56], and *GPT-2 XL* [57] were operated on Kaggle and Google Colab. The largest models, including *Flan-T5 XXL*, *Falcon 7B*, and *Falcon Instruct 7B* [58], were accessed using HuggingFace's inference API. This diverse selection of language models from various sources, see Table 3.1, enables varied comparisons for ***RQ2*** and ensures a model-agnostic approach to ***RQ1***. Various *Flan-T5* sizes were employed to assess the impact of model dimensions on its capabilities. Very large models were omitted due to their resource-intensive computational requirements.

The models generated a maximum of 64 tokens each, avoiding excessive text that could complicate evaluation. For API-based queries, a low temperature of 0.1 was chosen to yield conservative outcomes. Other models were integrated into pipelines with their respective HuggingFace model and tokenizer, utilizing float16 precision for numerical values. *Flan-T5* models were configured for a *text2text-generation* task, while *GPT-2* and *LLaMA* were assigned a *text-generation* task.

3.2 Prompting with Abductive Reasoning

3.2.1 Dataset and Evaluation

To answer ***RQ1***, the *ART* dataset [22] was selected to discover effective approaches for prompting LMs with abductive tasks. Although alternative datasets like *AbductionRules* [59] and *D*-Ab* are available, even larger LLMs struggle with them when relying solely on prompting methods [1].

Table 3.1: Model Parameter Sizes in Billions (including models from [1])

model	davinci	ChatGPT	BARD	GPT-2	LLaMA	Falcon	F. Instruct
size [B]	175	175	1560	1.5	3	7	7
Flan-T5	small	base	large	xl	xxl		
size [B]	0.08	0.25	0.78	3	11		

The *ART* dataset is composed of an ID column, two observations, two hypothesis and a label column marking the correct hypotheses. The hypothesis clarifies the second observation in the sequence: first observation, hypothesis, second observation. One of the hypotheses is incorrect, contradicting the observations, or less credible than the correct counterpart. The dataset serves two tasks: *α-NLI* (Abductive Natural Language Inference) selecting the correct hypothesis, and *α-NLG* (Abductive Natural Language Generation) generating a new hypothesis.

The main focus was on α-NLI, a classification task, which is easier to quantitatively measure and evaluate. Evaluation employed a heuristic approach, including direct letter matches (often *Flan-T5*), specific phrasing like *"the correct hypothesis is"* (few-shot prompting), matching the correct hypothesis as a longest substring, and, at times, by cosine similarity of embeddings. Cases without matched hypotheses were assigned a label of 0.

The dataset comprises 3059 test tasks for model evaluation. Most of the models were evaluated on all the tasks, in contrast to [1], who measured 1000 tasks with 80% accuracy and especially [47], who reported accuracy of 87% by measuring only on 30 tasks. Due to extended runtime, *LLaMA* and *GPT-2* models were assessed on the initial 300 tasks, resulting in higher standard error in reported values. Selecting initial tasks over a random sample is unlikely to introduce systematic error, as observed in other models. The training set consists of nearly 170k tasks, with a random subset used for few-shot prompting and a smaller set for CoT tasks; the development set was omitted. Model standard error was determined by randomly splitting results into 30 parts (about 100 and 100 each) and calculating deviations between them.

As per model guidelines, none of the models were trained or fine-tuned on this dataset. Moreover, all models were prompted with first three and three random observations and story IDs, which confirmed no data leakage from the dataset.

3.2.2 Single-Prompting

The initial experiment drew inspiration from [1] which evaluated prompting across three LLMs (*text-davinci-003*, *ChatGPT* and *BARD*) on several reasoning datasets. To ensure a direct comparison we adopted the zero-shot and few-shot prompts *Z0* and *F0*, as outlined in Table 3.2. We extended the few-shot setting to include a five-shot setting for *Flan-T5* models (*small*, *base*, and *large*), abiding by the 512-token limit. This experiment included all mentioned models, however, in later experiments the *Falcon* model was omitted due to its similarity to the *Falcon Instruct* model, and the *Flan-T5 small* due to its limited abilities.

To assess the models' **resilience to prompt reformulation**, the initial zero-shot prompt was modified into prompts *Z1*, *Z2*, *Z3* and *Z4*, outlined in Table 3.2. These specific formulations were selected following experimentation, primarily involving *Flan-T5 large* and *Falcon Instruct* models. Prompt *Z1* simplifies *Z0* while maintaining good results. Prompt *Z2* seeks to determine the score for the model choosing only a more similar text. The text similarity was also measured with cosine similarity of embeddings from a sentence transformer (*all-MiniLM-L6-v2*, as employed in evaluation). Prompt *Z3* aims to convey the sentence order, while prompt *Z4* restructures the tasks to select the more likely of two scenarios presented in chronological order (observation 1, hypothesis, observation 2). This reformulation better aligns with human assessment of the task.

The next task examined the models' capacity to handle **symbolic representations** rather than explicit texts, similarly to Chain-of-Symbol (CoS) prompting [39]. This ability could support genuine reasoning and the use of Event Calculus. This experiment encompassed a one-shot assessment using the *F1* prompt and a three-shot evaluation involving *F1*, *F2* and *F3* prompts. Prompt *F1* was derived form *F0*, presenting complete hypothesis formulations as answers alongside their labels. Prompts *F2* and *F3* were adapted from *Z4*, which selects the more prob-

Table 3.2: Overview of Employed Single-Prompts

{O1}, {O2}, {H1}, {H2}, {label} represent observations, hypotheses, and labels from the dataset. k denotes the count of instances, [: :] iteration through the examples.

prompt	text
task definition	Given a context, the abductive reasoning task is to choose the more likely explanation from a given pair of hypotheses choices. And give simple explanations.
task def. 2	Select the more probable explanation of given context from a pair of hypotheses choices.
task scenario	The reasoning task is to choose the more consistent and more likely scenario from a given pair of scenarios.
k examples	Next, I will give you {k} examples. (an example)
question 0	The context is:{O1}{O2} The hypothesis choices are: A.{H1} B.{H2}
question 1	Context: {O1}{O2} Hypotheses: A. {H1} B. {H2} Answer:
question 2	Scenario A: {O1}{H1}{O2} Scenario B: {O1}{H2}{O2} Answer:
question 3	O1 is: {O1}, O2 is: {O2}, H1 is: {H1}, H2 is: {H2} Scenario A is: O1 H1 O2 Scenario B is: O1 H2 O2 Answer:
zero-shot Z0	(task definition) (question 0)
zero-shot Z1	(task def. 2). (question 1)
zero-shot Z2	Context: {O1} {O2} Text A: {H1} Text B: {H2} Which Text (A or B) si more similar to the Context?
zero-shot Z3	First context C1 happened, then hypothesis happened, then context C2 happened. The task is to choose one hypothesis from a given pair of hypotheses. The chosen hypothesis is coherent with the contexts and does not contradict them. If both hypothesis are correct, the more likely one is chosen. (question 1)
zero-shot Z4	(task scenario) (question 2)
few-shot	task (k examples) [:(question) answer :] (one example) (question)
F0	(task def.), (question 0), answer: The correct choice is: {label}.
F1	(task def.), (question 0), answer: The correct choice is: {label} {Hx}
F2	(task scenario), (question 2), answer: The correct scenario is {label}
F3	(task scenario), (question 3), answer: The correct scenario is {label}
zero-shot T0	(task definition) (question 0) Let's think step by step.
few-shot Tx	(task def. 2) and give an explanation. (k examples) [:(question 1) (reasoning) :] (question 1)
T1 reasoning	(Hypothesis X contradicts the context because...) or (No hypothesis contradicts the context. Hypothesis X relates better to the context because...) The correct hypothesis is X
T2 reasoning	Does one of the hypothesis contradict the context? (Yes, hypothesis X because...) or (No. Hypothesis X relates better to the context because...) The correct hypothesis is X.
T3 reasoning	Does one of the hypothesis contradict the context? (Yes, hypothesis X contradicts context because... ...explaines why...) or (No. It is more likely that...) The context is better explained by hypothesis X.

able scenario. Finally the *F3* prompt employs symbols that represent sentences to outline the scenario in a CoS manner.

As a final step, **elicitive prompting** was implemented with a zero-shot *T0* prompt and few-shot *T1*, *T2* and *T3* prompts. *T1* and *T2* prompts were constructed using four specific examples, while the *T3* prompt featured a different set. The prompt structure, along with the reasoning steps, is in Table 3.2. The reasoning assesses if some of the hypothesis is contradictory to the context and which hypothesis is better related to it. Prompt formulation was iteratively refined through manual experimentation, leveraging the *Flan-T5 large* and *Falcon Instruct* models. The *Flan-T5 base* model was excluded from this task due to its inability to adhere to even simpler prompts.

3.2.3 Multi-Prompting

The application was limited to the *Flan-T5 XXL* and *Falcon Instruct* due to their higher potential. In light of the unsatisfactory CoT results, alternative strategies were adopted (Table 4.5). These prompts integrated keywords to enhance *Falcon* model's performance and incorporated cues to diminish its reliance on general knowledge instead of context. Moreover, efficient implementation of sequential prompts requires answer engineering to prepare responses for subsequent steps. Despite its implementation, the LangChain framework chaining was not employed during execution, enabling intermediate results for enhanced analysis.

The sequential prompt *S1* omitted the first observation in its first step, choosing the better explanation (hypothesis) for only the second observation. This approach reduces potential confusion related to the sequence order and enables assessing the importance of the initial observation for a correct answer. Notably, the first step benefited from a few-shot setting. The second step reintroduced the first observation to determine its compatibility with the chosen hypothesis.

The sequential prompt *S2* employed a chain where the model generated explanation for each hypothesis contradicting the context, aligning with prior Falcon answers. In the second step the model self-assessed the plausibility of these explanations, and in third step determied the more likely explanation. An intriguing approach connected the two models, evaluating *Falcon's* contradiction explanations with the reasoning capabilities of the *Flan-T5* model.

Parallel prompting via self-consistency sampling combined answers from three distinct prompts using the same model, given the considerable performance differences between the models. While mutual independence could not be guaranteed,

Table 3.3: Overview of Sequentially and Parallelly Chained Prompts

{O1}, {O2}, {H1}, {H2} represent observations and hypotheses from the dataset. All the prompts begin with "Instruction:" and end with "Response".

prompt	text
S1 step 1	The task is to select one and only one of two provided sentences, nothing else, to logically and coherently continue the given context. I will give you a few examples. Context is: (observation) Sentence A is: (hypothesis A) Sentence B: (hypothesis B) Response: (observation) That is because (hypothesis B) Sentence B explains the context better. (+ two more examples) Context is: O2 Sentence A is: H1 Sentence B is: H2
S1 step 2	The task is to decide if the sentences in the given text are consistent (answer "yes") or if they contain a logical with each other (answer "no"). Then provide an explanation. Text: {O1} *S1* step 1 answer
S2 step 1	The task is to explain the inconsistency and logical fallacy between the sentences of given text. Answer with phrase "It is because ___", do not use the words "inconsistency" and "logical fallacy" in the explanation. Text: {O1} {Hx} {O2}
S2 step 2	The task is to decide if given explanation is coherent and correct. Explanation: *S2* step 1 answer
S3 step 3	The task is to decide which one of the explanations is more correct and more likely given the provided context. Answer with phrase "Explanation _ is more likely because ___". The context is: {O1} {O2} Explanation A is: *S2* step 2 answer for {H1} Explanation B is: *S2* step 2 answer for {H2}
Z5	There are two independent texts, B and A. The task is to decide which one of the texts is not logically consistent and contains a contradiction. Consider only the information in the texts. Text A: {O1} {H1} {O2} Text B: {O1} {H2} {O2}

efforts were made to employ diverse prompts, including *S1*, *S2*, and *Z0* and the new zero-shot prompt *Z2*, aimed at identifying logically inconsistent text rather than more likely text as in *Z0*. Ensuring the "better than random" criterion posed a challenge with the *Falcon* model, based on prior experiments. However, qualitative analyses indicated potential success with appropriate prompting.

3.3 Text Planning with Abduction

To address ***RQ3***, we developed a three-stage text generation pipeline with content selection, discourse planning and realisation stages. While content selection received primary attention, minimal effort was allocated to realization due to its misalignment with the abduction focus. Implementation again utilized the LangChain [53] framework, involving only the *Flan-T5 XXL* and *Falcon Instruct* models. Given prior experiments indicating that models struggle with symbolic representations, we refrained from the intended event calculus formalism (Chapter 2.2.1), opting for a more straightforward approach, similar to previous experiments. The implemented prompts are listed in Table 3.4. Content selection and discourse planning exclusively employed the *Flan-T5* model, as *Falcon* exhibited inadequacies in reasoning, producing seemingly coherent but nonsensical output.

To evaluate our approach, the initial plan was to use the WeatherGov dataset [60]. However, constrained by time, we generated a simplified dataset to prioritize easier evaluation implementation and emphasize planning with abductive reasoning. This dataset covers four key weather phenomena: sky state (clear, cloudy, overcast), temperature (-10 to 30°C), precipitation (0 to 20 mm/h) and wind (0-100 km/h). Moreover, animals (cat, bird, fish) were included with distinct weather and order preferences, along with locations (beach, city, meadow, mountain) exhibiting varying weather predispositions. The created pipeline (Table 3.4) was then evaluated across six hundred randomly generated instances.

3.3.1 Content Selection

In this stage, the relevant messages for generation are selected based on predefined rules. The choice of weather context was a source of inspiration for experimentation and the fabricated rules lack meteorological validity. These messages encompass three different weather phenomena: sky coverage, precipitation and wind. Each of them serves as a mean to investigate the reasoning capabilities.

The prompt implementation method mirrored the strategy from the AbductionRules dataset, where the model selects a hypothesis that best aligns with provided observations and rules. The prompts were adopted in zero-shot setting, as the few-shot yield no improvement and CoT prompts even worsened results, in line with previous findings. An approach that explicitly stated variables was used, leveraging its applicability to this task and enhancing the response's quality.

The *CSS* (sky coverage) prompt investigated the model's treatment of basic logic elements: equality, 'and,' 'or', and implication by removing parts of the prompt. While the *CSW* (wind) prompt delved into the model's number comparison capabilities and required an abductive reasoning step as there is not explicitly stated when "wind" occurrs.

The most intricate prompt involved precipitation (*CSP*) with four potential outputs and four assessable variables. Againg, the absence of rain selection criteria necessitated an abductive reasoning step. This information was then required in another rule to rigorously test the model. A simpler version of this prompt, featuring three outputs and three variables, without the rule based on abducted information, was tested and then further modified. One modification excluded variables, enabling the model to leverage its world knowledge, another was adjusted to examine its performance in a generative context, omitting provided explanations, and framing the task as, *"The reasoning task is to decide what will happen, given the rules"*. Lastly, the model's handling of additional explanations was evaluated, spanning from the least likely "There is pizza" to the more plausible "There is drizzle" and "There is shower".

The final selection prompt aimed to eliminate sentences stating the absence of precipitation and wind, as they are unnecessary for reporting. However, developing a suitable solution proved challenging; it often either removed too much or too little content. Consequently, we shifted our focus to a simpler task, involving the removal of a sentence based on 'animal requirements' (e.g., fish does not care about wind). This approach still did not yield good enough results, removing too much information, so it was excluded from the pipeline.

3.3.2 Discourse Planning

The objective of discourse planning is to arrange selected messages according to specific preferences. Although utilizing generation rather than inference would be desirable, it appears that even the *Flan-T5* model has limitations in this aspect. We attempted to implement a genuine planning approach, specifying the order of prompts such as which sentence should be placed first, last, or before/after another. However, the sole effective strategy was to place a single sentence at the beginning (by simple rules, without abduction). Frequently, it was necessary to remove the sentence that remained in position after being copied to the beginning.

Thus, this stage again relies on selecting one of the provided explanations (sequence of letters) in prompt *P1*. We experimented with cases where two possible orderings were feasible, allowing us to investigate the model's response. In the following step (*P2*), the letters are simply replaced with corresponding sentences from the content selection stage. The output proceeds to the realization stage, executed with both the *Flan-T5* and *Falcon Instruct* models.

Table 3.4: Overview of Prompts for Text Generation Pipeline

{S}, {P}, {T}, {W} are values of the variables. All the prompts end with "Answer: "

prompt	text
task def.	The reasoning task is to select the more probable explanation, given the rules.
Content Selection Sky (CSS)	(task def.) There are three variables: [VAR:sky], [VAR:place] and [VAR:animal]. Rules: If [VAR:place] is beach and [VAR:animal] is cat, there is mist. If [VAR:place] is city or [VAR:animal] is bird, there is smog. The three explanations are: There is mist. The sky is {S}. There is smog. Values: [VAR:sky] is {S}. [VAR:place] is {L}. [VAR:animal] is {A}.
Precipitation (CSP)	(task def.) The reasoning task is to select the more probable explanation, given the rules. There are four variables: [VAR:temp] [VAR:precip], [VAR:sky] and [VAR:place]. Rules: If [VAR:temp] is less than zero, there is snow. If [VAR:precip] is zero then there is no rain and no snow. If [VAR:sky] is clear, there is no rain and no snow. If there is rain and the [VAR:place] is mountains, there is storm. The four explanations are: There is no rain and no snow. There is snow. There is rain. There is storm. [VAR:temp] is {T}. [VAR:precip] is {P}. [VAR:sky] is {S}. [VAR:place] is {L}.
Wind (CSW)	(task def.) There is a variable: [VAR:wind]. Rules: If [VAR:wind] is less than 18 there is no wind. If [VAR:wind] is more than 50 there is strong wind. The three explanations are: There is wind. There is no wind. There is strong wind. Values: [VAR:wind] is {W}.
Animal (CSA)	Task: Remove sentences from text based on the provided rules. There is variable [VAR:animal]. Rules: If [VAR:animal] is fish then remove the sentence containing word "wind". If the [VAR:animal] is not fish, output all the provided text. [VAR:animal] is {A}. Text: {P2})
Plan P1	The reasoning task is to select the best order of event, given the rules. There are two variables: [VAR:animal] and [VAR:sky]. Rules: If [VAR:animal] is cat then the order is: T, S, P, W. If [VAR:sky] is clear then the order is: P, S, T, W. If [VAR:animal] is bird then the order is: W, S, P, T. Usually the order is: S, P, T, W. The possible explanations are: S, P, T, W; W, S, P, T; P, S, T, W; W, P, T, S; T, S, P, W; [VAR:animal] is {A}. [VAR:sky] is {S}.
Plan P2	Task: Replace letters in a given sequence with their full corresponding sentences. I will give you an example: (example) Replace letters in a given sequence with their full corresponding sentences. Context: Replace letter S with the sentence: "{CSS}." Replace letter P with the sentence: "{CSP} {P} mm/h." Replace letter T with the sentence: "The temperature is {T} °C." Replace letter W with the sentence: "{CSW} {W} km/h." Sequence: {P1} Generate the sequence with the letters replaced by their corresponding sentences and include all the provided sentences.
Realization	Task: Generate a fluent weather forecast in future tense from provided informations. Use all the informations and only the informations provided. Weather forecast must contain information about where it is. Informations: Tomorrow in/on the {L}. {P2}.
Pipeline	CSS $\rightarrow$ CSP $\rightarrow$ SCW $\rightarrow$ P1 $\rightarrow$ P2 $\rightarrow$ Realization

Chapter 4

Results

4.1 Results of Abductive Prompting

4.1.1 Single-Prompting

Initial Experiment

Table 4.1 presents the results for the *α-NLI* dataset, prompted by *Z0* and *F0* prompts, in comparison with models from [1]. In the zero-shot setting, *Flan-T5* models predominantly produce the hypothesis letter; sometimes, the *large* model repeats also the hypothesis. *GPT-2* outputs exhibit notable variability, frequently struggling to generate relevant responses, presenting unrelated information. The *LLaMA* model primarily outputs responses beginning with *"The explanation is:"* followed by the letter and repeated hypothesis. Moreover, both *GPT-2* and *LLaMA* models occasionally experience mismatches between the hypothesis letter and the formulation. *Falcon Instruct's* responses closely resemble those of *LLaMA*, while *Falcon* often repeats hypothesis without its letter marking.

In few-shot setting, *LLaMa*, *Falcon*, and *Falcon Instruct* answers follow the prompt at a rate of 95-99%. *GPT-2* continues to struggle in providing inter-

Table 4.1: Initial Zero-Shot and Few-Shot Prompts Results

model	Z0	SE	NA	1-F0	SE	NA	3-F0	SE	NA	5/10-F0	SE	NA
davinci	74			70			74					
ChatGPT	81			80			79					
Bard	75			74			77					
Flan-T5												
small	52	2	0	52	2	0	53	2	0	53	2	0
base	61	2	0	61	2	0	61	2	0	61	2	0
large	68	2	9	71	2	6	69	2	7	70	2	6
xl	81	2	0	81	2	0	81	2	0	81	2	0
xxl	85	2	0	84	1	0	84	1	0	85	2	0
GPT2	14	4	74	35	3	26	44	4	23	39	4	28
LLaMA	18	3	66	50	4	2	47	4	1	48	4	0
Falcon	34	1	36	52	2	0	52	2	0	51	2	0
F. Instruct	48	1	1	52	2	0	53	2	0	52	2	0

SE: Standard Error, NA: Not Available answers; Green: best results, Yellow: more than 5 times preference for one answer, Blue: statistically significant improvement at $p < 0.05$, Grey: comparable to random choice

pretable answers, achieving alignment with the prompt in only 59% of instances. The *Flan-T5* model consistently outputs the hypothesis letter, following the prompt in only 11% (3-shot) and 6% (5-shot) cases for the *large* model, in 15% and 8% (10-shot) for the *XL*, and in 2% for the *XXL* model.

Instances marked as "not available" (*NA*) frequently arise from the model repeating both hypotheses without making a choice (*LLaMA* in zero-shot setting and *Flan-T5 large*) and from generating a new hypothesis (*LLaMA* in few-shot setting and *Falcon*). *Falcon Instruct* produces authentic blank responses. In the *LLaMA* model, the count of *NA* answers might slightly exceed the reported figures, owing to occasional misclassification due to the token limit cutoff.

Certain models consistently favour a single answer choice, this would result in an accuracy of 52% for choice *A* and 48% for choice *B*. Notably, the *Falcon* model is prone to this, displaying a threefold preference for answer *A* in zero-shot setting and 13 (1-shot) to 38-fold (3 and 10-shot) preference in few-shot setting. *Falcon Instruct* performs better in few-shot setting, favoring answer *A* fourfold

(1-shot and 3-shot) and fivefold (10-shot), compared to predominantly *B* choices in zero-shot setting. The *Flan-T5* model experiences this bias exclusively within the *small* model, generating answer *A* eightfold more in zero-shot and elevenfold more often in few-shot setting. Unbalanced responses yield comparable outcomes across both correct and incorrect answers, minimally affecting the results.

Resilience to prompt reformulation

Table 4.2 displays outcomes for prompts *Z1-Z4*. The models respond conventionally, except for *Z4*, where they follow the prompt and answer *Scenario A/B*, with *Flan-T5 base* aligning only in 74% of cases. *GPT-2* outputs remain challenging to interpret, particularly with *Z3*, where a new hypothesis is often generated. Instances of *NA* responses parallel those previously reported, except for *LLaMA* which now also sometimes repeats the question with prompt *Z2*.

The similarity of the hypotheses with the context, measured with cosine similarity of embeddings achieved 54% accuracy. The *Falcon* and *GPT-2* models choose the same answers in about 50% of cases, *LLaMA* in 44% and *Flan-T5* in 54%, 60%, 59% and 57% respectively for bigger models.

The *Flan-T5* model showed bias towards one answer only with the *base* model for prompt *Z2*, displaying eightfold preference for answer *B*. *GPT-2* model was biased twofold towards *B* for *Z3* and the *LLaMA* model predominantly chose

Table 4.2: Zero-Shot Prompts Results for Testing Resilience to Reformulation

model	Z1	SE	NA	Z2	SE	NA	Z3	SE	NA	Z4	SE	NA
Flan-T5 base	61	2	0	53	2	0	61	2	0	57	2	2
Flan-T5 large	74	2	0	70	2	0	73	2	0	74	2	0
Flan-T5 xl	81	1	0	79	2	0	80	1	0	82	2	0
Flan-T5 xxl	84	2	0	78	1	0	82	1	0	83	1	0
GPT2	27	4	45	20	4	60	30	3	43	26	4	43
LLaMA	31	4	41	15	3	71	50	4	7	39	4	29
F. Instruct	51	2	0	49	2	4	50	2	0	41	2	18

SE: Standard Error, NA: Not Available answers; Yellow: more than 5 times preference for one answer, Blue: superior to *Z0* at $p < 0.05$, Orange: inferior to *Z0* at $p < 0.05$, Grey: comparable to random choice answer *A* with prompt *Z1* and five times more with *Z4*. *Falcon Instruct* chose answer *B* threefold more with *Z3* sevenfold with *Z1* and *A* sevenfold *Z4*.

Symbolic representations

Table 4.3 presents results for one-shot prompt *F1*, and three-shot prompts *F1*, *F2* and *F3*. The responses are mostly similar to those previously reported. The *Flan-T5* follows the *F1* prompt only with the *large* and *XL* models, which adhere to the prompt in 55% and 16% of cases respectively. The aligned answers exhibit lowered accuracy around 62%. The *F2* and *F3* are fully followed, only the *base* model at lower rates of 10% and 40% respectively. *GPT-2*'s responses conform to *F1* in 68% (1-shot) and 74% (3-shot), with 86% alignment for *F2* and *F3*. *LLaMA* consistently adheres to all prompts, occasionally outputting the same phrases, probably from training. the same, possibly training, phrases. *Falcon* aligns with prompts formualtions, except for *F1* in 1-shot setting, producing responses similar to zero-shot.

The *NA* responses parallel those previously reported, except for *Flan-T5* and *GPT-2* models with *F2* and *F3* prompt, where an observation is often selected rather than a hypothesis. The bias towards one answer is observed in the *Flan-T5 base* model for prompt *F3*, that yields exclusively answer *A*. Prompted with *F1* *LLaMA* displays a fourfold (1-shot) and twofold (3-shot) preference for answer *A*

Table 4.3: Few-Shot Prompts Results on the Handling of Symbolic Representations

model	1-F1	SE	NA	3-F1	SE	NA	3-F2	SE	NA	3-F3	SE	NA
Flan-T5 base	61	2	0	61	2	0	52	2	1	50	2	3
Flan-T5 large	69	2	8	72	2	3	70	2	0	53	2	14
Flan-T5 xl	80	2	0	81	2	0	81	1	0	69	2	4
Flan-T5 xxl	84	1	0	84	1	0	84	2	0	73	2	0
GPT2	36	4	22	44	4	14	42	4	14	46	4	10
LLaMA	55	5	0	55	4	0	50	4	0	52	4	0
F. Instruct	52	2	0	51	2	0	51	2	1	51	2	0

SE: Standard Error, NA: Not Available answers; Yellow: more than 5 times preference for one answer, Blue: superior to *F0* at $p < 0.05$, Orange: inferior to *F0* at $p < 0.05$, Grey: comparable to random choice and twofold for *F2*. *Falcon* prefers answer *A* sixfold with prompt *F1* in 3-shot setting, twofold with *F2* and a threefold with *F3*. In 1-shot setting with *F1* the model predominantly chooses answer *B*.

Elicitive prompting

Table 4.4 displays results for elicitive prompts. Interpreting zero-shot results is challenging across models, potentially yielding different values than reported. *Flan-T5* models respond typically, the *XXL* model occasionally includes the second observation before selecting hypothesis. *NA* outputs follow the usual pattern. *GPT-2*'s responses are often inadequate but generate fewer novel hypotheses. LLaMA often rearanges the provided sentences, producing *NA* responses. *Falcon's* detailed answers require a higher 128-token limit, yielding answers resembling explanations, sometimes accurately. *NA* answers stem from newly generated, seemingly plausible, hypotheses. *Flan-T5 large* prefers answer *B* twice as much, while *GPT-2* and *LLaMA* twice as much *A*.

In the few-shot setting, *Flan-T5* does not follow the prompts wording very often. They are followed in 48%, 48% and 98% with the *large*, *XL* and *XXL* models for prompt *T1*, 31%, 31% and 5% for *T2* and 59%, 55% and 30% for *T3*. When the prompts *T2* and *T3* are followed, the accuracy is lowered to 69%, 64% and 56% with *T2* and 47%, 70% and 69% with *T3*. The *GPT-2* model adheres to the prompt in 36%, 38% and 37% respectively for the prompts, while *LLaMA*

Table 4.4: Elicitive Prompt Results

model	T0	SE	NA	T1	SE	NA	T2	SE	NA	T3	SE	NA
Flan-T5 large	49	2	31	71	1	3	68	2	4	43	1	38
Flan-T5 xl	80	2	0	81	1	0	79	1	0	79	2	0
Flan-T5 xxl	79	2	2	80	2	0	80	2	0	80	1	1
GPT-2	24	3	58	35	4	32	36	4	24	31	4	38
LLaMA	14	3	71	45	4	0	55	4	0	54	4	0
F. Instruct	33	2	39	52	2	0	49	2	0	51	2	1

SE: Standard Error, NA: Not Available answers; Yellow: more than 5 times preference for one answer, Blue: superior to *F0* at $p < 0.05$, Orange: inferior to *F0* at $p < 0.05$, Grey: comparable to random choice consistently adheres to them, similarly to *Falcon Instruct* that consistently follows the prompts at rate 90%, 97% and 98%.

Contradictory hypotheses are not identified often with the *Flan-T5* models. The *XXL* model with prompt *T1* identifies 7%, the *large* model and prompt *T2* identifies 4%, predominantly selecting hypothesis *A*. The better related hypotheses are rarely detected. *GPT-2* finds contradiction in 11%, 5% and 4% of cases with respective prompts, identifying better-related hypothesis for *T1* (12%) and *T2* (9%). These related hypotheses are predominantly *B*. *LLaMA* finds contradiction in 54% and better-related hypothesis in 46% with prompt *T1*, while *T2* and *T3* always produce a contradiction. *Falcon* finds contradictions in 10%, 8% and 9% cases respectively, occasionally selecting contradicting solution, like *LLaMa*. The better-related hypothesis is identified in 8% and 4% for prompts *T1* and *T2*.

While the *GPT-2* model exhibits twofold bias for answer *B*, same as the *Flan-T5 large* with prompt *T3*, other models regrettably show significant biases. *LLaMA* outputs predominantly answer *B* for prompt *T1* and *A* for *T2* and *T3*. The *Falcon* favours answer *A* elevenfold and sevenfold for *T1* and *T3* and fourfold for prompt *T2*. *Falcon's NA* responses often use observations as hypotheses.

4.1.2 Multi-Prompting

Self-consistency sampling assumes that we have models with better than random accuracy and they are independent. As only one model (Flan-T5) was able to achieve better than random accuracy, this approach could not be used. Voting of the same model with different prompts or differently sized model with same prompt did not improve results, which confirms their dependence.

Table 4.5: Multi-Prompting Results

model	0CoT	SE	NA	CoT1	SE	NA	CoT2	SE	NA	CoT3	SE	NA
Flan-T large												
Flan-T xl												
Flan-T xxl												
F. Instruct												

4.2 Results of Text Planning

Table 4.6 displays the accuracy of the text generation pipeline steps, along with results from other experiments involving these prompts. The basic *CSP simple* prompt encounters an issue, where it outputs *snow* despite *no precipitation* or clear sky (11% accuracy drop), with correct response when both conditions are fulfilled. When comparing numbers a slight drop in accuracy (2%) is observed as *snow* is selected with temperatures slightly above zero (1 or 2 °C). The *CSP* yields improved results (9% drop) in detecting *no precipitation*. However, it often opts for *storm* over *snow*, reducing accuracy by around 2%.

Similar shortcomings arise in other *CSP*-based prompts. In the *CSP knowledge* variant, the opting for *snow* and *rain* instead of *no precipitation* leads to an 18% and 6% reduction in accuracy, respectively. Comparisons involving numbers decline by 7%. The generation prompt exhibits fewer *"no precipitation"* errors, with a 3% drop. However, its accuracy drops by 14% when handling numbers. Among the prompts with additional hypothesis, both *snow* and *rain* are chosen with the same frequency instead of *no precipitation*, resulting in 30% drop for *pizza* and *drizzle* and 28% for *shower* hypothesis. They prioritize *snow* over *rain* with 5% drop for *pizza* and 3.5% for *drizzle* and *shower*. Among these hypotheses, only *shower* closely aligns with the context to be chosen in about 3% of cases.

The *CSS* prompt exhibits a bias towards selecting *clear sky* over *smog* or *mist* in about 1/3 of their instances, thereby impacting all related prompts. With

Table 4.6: Text Generation Results

prompt	CSP	CSP simple	CSP knowledge	CSP generative	CSP pizza	CSP drizzle	CSP shower	CSW
acc. [%]	93.7	87.7	65.3	83.5	64.8	66.3	65.5	98.5

prompt	CSS	CSS *and*	CSS *or*	CSS *both*	P1	P2	CSA	All
acc. [%]	81.5	64.0	63.2	50.5	100	77.3	56.3	

the *and* constraint, he frequency of *clear sky* selection for *mist* drops to 1/4, while a minor bias towards *cloudy* emerges in about 1/5 of *smog* selections. Additionally, *mist* is mistakenly favoured over *cloudy* and *overcast* when only one of conditions is met. The introduction of the *or* constraint has minimal impact on outputs and the bias towards *clear sky* remains unchanged. When both constraints are employed, the behaviour appears to be a straightforward combination.

The *CSW* prompt inaccurately compares numbers in just 1.5% of instances, particularly when the wind speed is exactly 50 km/h. This prompt fails to remove wind in only 2% of cases, yet mistakenly removes wind in 41.7% of cases where it should remain. The *P2* prompt exhibits a failure to convert letters to sentences in 11.7% of cases, while translation errors occur for sky in 6.7%, wind in 5%, and temperature in only two instances, with precipitation consistently included.

In the realization stage, the *Flan-T5* struggles with language-related tasks, like converting sentences to future tense, selecting suitable prepositions, and overall formulation of a fluent forecast. In contrast, the *Falcon* model excels in these aspects, skilfully integrating messages into sentences. Nonetheless, *Falcon* makes completely accurate responses in only about half of the cases, commonly making four types of mistakes with similar frequency: substituting "gentle breeze" or similar terms for wind, reordering information (often placing sky-related details at the beginning), reporting incorrect sky values, and omitting certain information.

4.3 Interpretation of Abductive Prompting

4.3.1 Optimal Prompting Strategies

Solely *Flan-T5* effectively executes the abductive task, while *GPT-2* struggles with interpretability, and *LLaMA* and *Falcon* tackle answer bias. This leads to significant variations in the recommended prompts.

GPT-2

Despite the *GPT-2* model's inadequacy for abductive tasks at this size, we can still identify a suitable prompting approach to partially address *RQ1*. Quantitative analysis of the model's outcomes is intricate, favouring qualitative evaluation, particularly in the zero-shot context. However, interpretability improves in the few-shot setting, notably with a direct task formulation explicitly mentioning the hypothesis in the explanation (*F1*), reducing generation of novel hypotheses.

Another suggestion involves specifying the task prior to the queries, as the absence of this step reduces interpretability, as evident in *Z2*. Moreover, simpler prompts like *Z1* outperform more intricate ones, such as *Z0* and *Z3*. Elicitive prompting and the use of symbolic representations is not advisable, as seen from Table 4.4 and the prompts' responses.

LLaMA

The most effective prompts, addressing *RQ1*, for this model are the few-shot prompt *F1*, including the complete hypothesis text, or the zero-shot prompt *Z3*. While *F1* exhibits a two-fold bias towards answer *A*, *Z3* lacks bias but has a 7% rate of uninterpreted answers. Both prompts show slightly improved accuracy compared to random choice, although not statistically significant. Hence, we conclude that the *LLaMA* model in its 3B version cannot fully grasp abductive reasoning solely through prompting. Unfortunately, the model generated few explanations, limiting the depth of qualitative exploration.

With this model, caution must be exercised regarding its sensitivity to prompt phrasing and its significant susceptibility to answer bias, especially in the zero-shot setting. Detailed task definitions like *Z3* are preferred, in contrast to *Z1*, which is biased towards one solution, or *Z2*, where most answers lack interpretability. Again, elicitive prompts and symbolic representation do not show promise, as seen from the model's responses.

Falcon

To answer *RQ1*, the few-shot prompt *F2* emerges as the most suitable for *Falcon Instruct*, despite its comparable performance to random choice. This choice is driven by the prompt's least answer bias, a significant challenge for this model. Although the bias appears to improve in few-shot settings, the model still tends to favour answer answers slightly more represented in the dataset as correct (A), revealing a deeper issue. Notably, the *Falcon* model's answer bias worsens in the few-shot setting, in contrast to *Falcon Instruct.* However, this model does not perform well in the zero-shot setting, as it lacks instruct training.

In the zero-shot setting, the model mainly picks choice B, possibly due to training, and then fluently justifies it using its world knowledge. While justifications for the right choice are occasionally reasonable, for the incorrect they can be peculiar or vague; this output might be useful when inventively used. The robustness prompts do not notably alter accuracy, but they display varying answer bias despite our efforts to reduce it, underscoring the importance of prompt formulation. Analyzing *Z4's* responses suggest that the model chooses more plausible scenarios within a world context, rather than focusing on a specific scenario. Symbolic representation appears to counter bias, as seen from prompts *F0* and *F1.* Yet, with prompt *F3*, it amplifies bias, while *F2* prompt diminishes it similarly to *Z4.* This highlights the answer bias as the main concern, impeding in-depth analysis. Elicitive prompting maintains a bias towards a single hypothesis, however qualitative analysis hints at the potential utility of the approach in a distinct task that doesn't require reasoning abilities.

Flan-T5

This is the only model that addresses *RQ1* by effectively performing the abductive task. For most *Flan-T5* models, *Z0* stands out as the optimal prompt, while revised prompts like *Z1* and *Z4* led to improvements for the *large*, as seen in Table 4.2. These adjustments primarily targeted the *large* model, implying potential benefits across other sizes. However, these enhancements corresponds to answers that were previously uncategorized, as *Z0* sometimes led to appending the other hypothesis behind the answer (twice as often for hypothesis A than B). Overall, the model displays resilience to prompt reformulation, with minor variations in accuracies across prompts. A comprehensive examination of these results across different model sizes is available in Appendix A.

Table 4.3 shows that while the models can handle symbolic representations to some extent, even the largest versions experience performance limitations. Although the model should have potential for CoT, as expected from training, our formulation of the prompts for abduction was not optimal, as their usage diminished accuracy, likely contributing to their infrequent use by the models. Thus, those approaches are feasible, they presents challenges.

4.3.2 Comparison of Used Language Models

Figure 4.1 depicts the answer to ***RQ2***, highlighting the models' optimal performance in comparison to others. Moreover, we can see *Flan-T5's* performance enhancement as model size increases, and noteworthy capabilities as instruction following, Chain-of-Thought execution, and symbolic representation comprehension. Overall, Flan-T5 excels as the leading model, exclusively capable of abductive reasoning. Conversly, *GPT-2* preformed the poorest, as it struggled with hypothesis selection. *LLaMA* and *Falcon* faced answer bias issues, however *Falcon* demonstrated a better ability to engage in CoT, offering promising explanations.

Prompt *Z2*, designed to assess **similarity** rather than abductive performance, combined with a naive embedding similarity, demonstrated the task's resilience to such similarity. *Flan-T5* exhibited the closest match with the naive approach among all models, overlapping in 57% of cases for the *XXL* model, however the model's performance extends beyond mere similarity assessment. An illustrative example of prompt preformance: *O1: I received a brand new phone for my birthday. O2: I fixed my phone's screen easily. H1: "The screen was beautiful." H2: "Two days later, I dropped it in the driveway."*. With the *Z2*, the *XXL* model opts for the similar yet incorrect hypothesis (*H1*), while the *Z0* prompt yields the correct solution (*H2*). This alignment with simple similarity diminishes in larger models, with the *XXL* model exhibiting the most significant accuracy drop - an indication of its reasoning abilities. *LLaMA* and *Falcon* responses share similarity (59% match), with *LLaMA* negatively correlated with the embedding. Qualitative analysis of *Falcon's* answers hints at a preference for similar hypotheses over random choices. This underscores these models' task performance compared to *GPT-2*, which appears to merely mimic prompt, explaining its resilience to prompt formulation and answer bias.

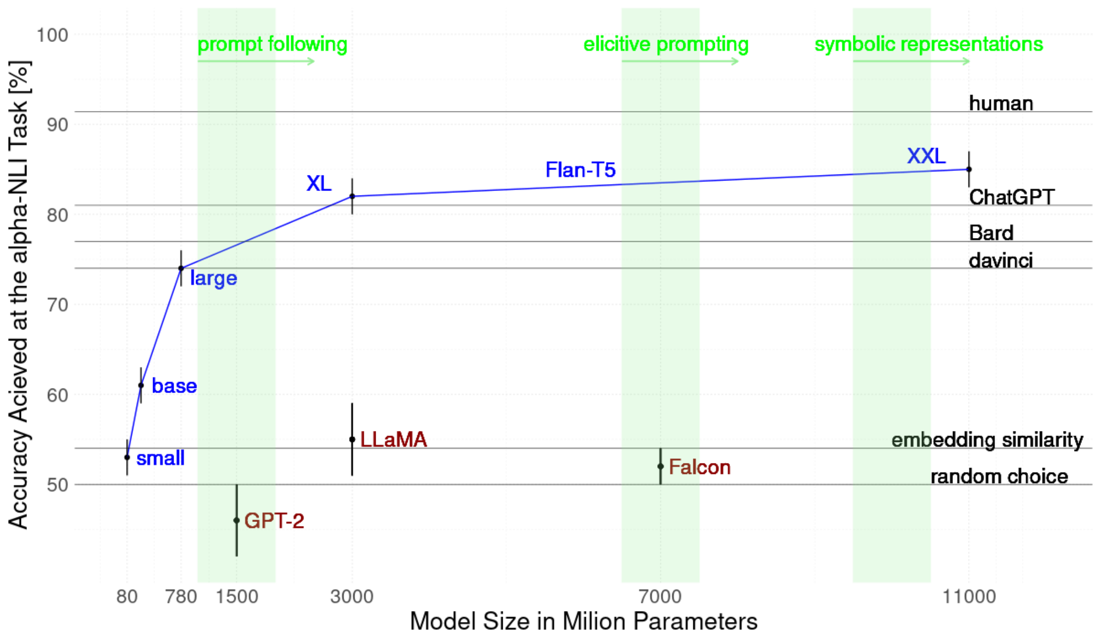
Among the models, only *GPT-2* struggled with **instruction following**, achieving a peak few-shot following of 86% with prompts *F2* and *F3*. In contrast, *LLaMA* and *Falcon* displayed this competence consistently across several prompts. Remarkably, the *Flan-T5* was reluctant to strictly adhere to prompts. The *XXL* version demonstrated this skill with the *T1* prompt, while the *large* and *XL* models exhibited such behaviour in roughly half of the cases. Using the *Z4* prompt, which requires simpler following, the *base* version achieved a 74% following rate. Thus, we hypothesize that a model within the parameter range of 1.5 to 3 billion would likely suffice, implying that the *XL* model could handle those tasks.

With **elicitive prompting**, *GPT-2* and *LLaMA* appear to imitate the task rather than generate genuine insights. *GPT-2* showed some validity in word matching for better-related hypotheses, but identified contradictions lack reliability. *LLaMA* exhibited strong answer bias, even with favourably reformulated prompt with *Z3*'s task definition. Falcon also displayed notable bias, yet in zero-shot setting, it occasionally generated plausible new hypothesis which aligned with world knowledge, albeit not the context. In cases of incorrect answers, the model frequently confused hypotheses with the context. As the *Flan-T5* model lacks training in abductive reasoning, it is expected that the zero-shot *T0* prompt does not yield a thought pattern. Despite its usual brevity, the *XXL* variant occasionally appends the second observation after the correct hypothesis, suggesting an understanding of the relationship. In few-shot setting, the prompt is rarely followed, possibly due to its formulation. With the *T3* prompt the model tends to elicit the correct hypothesis after stating observation 2 followed by *because*. Yet, it sometimes connects two observations or hypotheses together.

The models identified around 10% of contradictory hypotheses, and better-related hypotheses in single-digit percentages, which does not provide significant insight. Moreover, the final choice is often unrelated to the identified cases. This points to potential issues in the prompts' design. However, this analysis might reveal the importance of exemplar selection and reasoning path formulation, which varies among models. *GPT-2's* behaviour with prompt *T3* suggests example selection's importance for interpretability, while *LLaMA's* bias toward the same answer for *T2* and *T3* points to reasoning path significance. This trend is also evident in *Falcon*, as seen in *T1* and *T3* with similar biases but different examples, in contrast to *T2* with the same examples as *T1*, exhibiting opposing bias. *Flan-T5's* behaviour is less clear due to its sporadic adherence to prompts.

Except for *Flan-T5*, none of the models demonstrated proficiency in handling **symbolic representations**. *GPT-2* exhibited inconsistencies between hypothesis letter and its corresponding text, much like *LLaMA*, and its performance improved when presented with hypothesis text (*F1*). *LLaMA* often repeated the same sentence for all symbols, and often confused *O1* with *H1*, and *O2* with *H2*. Its typical answer bias was not present with symbolic prompt *F3*, further supporting its lack of this ability. Moreover, the contradictions from CoT prompts sometimes did not align with the final choice, akin to *GPT-2*, though this could indicate just an incomplete understanding of the task. *Flan-T5's* performance on prompt *F3* (Table 4.3) indicated a lack of this ability in the *base* and *large* models, even impeding the *XXL's* performance. Nevertheless, *Flan-T5's* capability in this aspect supports the idea that it actually engages in reasoning. Interestingly, achieving strong results on this dataset does not solely rely on interpreting symbolic meaning, as evidenced by the *base* and *large* models, although such an ability likely contributes.

Figure 4.1: Comparing Language Model Performance on ART Dataset [22]



4.4 Interpretation of Text Planning

To address ***RQ3***, we established a text generation pipeline incorporating abductive reasoning. Abduction reduces the effort for pipeline creation compared to the traditional approach, as not all information needs explicit specification. It also facilitates the addition of new requirements through the rules. Implementation of the content selection stage the *Flan-T5 XXL* model proved quite feasible, especially as an inference task. However it is advisable to divide the abduction to smaller steps, at least one prompt for a atomic message. Our use of the *Falcon* model for content selection unveiled limitations, as the model's fluency can inadvertently mislead, giving the impression of reasoning within the α*-NLI* dataset.

The *CSP simple* prompt reveals a significant challenge with abductive reasoning: the model's treatment of rules varies based on their order. The results indicate the model's preference for earlier rules, favouring snow (first rule) while employing the last rule less often. The *CSP* prompt exposes the complexity of an added reasoning step (associating *storm* only with *rain*), suggesting improved usability with more explicit rules. CoT prompting or fine-tuning might address this issue. The 6% accuracy gain for *CSP* stems from better utilization of the last *clear sky* rule. The model's knowledge and the lack of explicit variable statements amplify prior errors, causing a 24% accuracy drop, reduced use of the last rule, and diminished numerical accuracy (selecting snow up to 9°C). Surprisingly, the generation prompt performed reasonably well for the simple example, particularly with the last rule. Yet, it tends to omit the temperature rule, selecting snow even at temperatures up to 29°C. It is advisable avoid introducing additional non-relevant hypotheses as it reduces performance, although the model adeptly chooses more plausible explanations, evident in its selection of only the *shower* hypothesis. The 22% accuracy drop across all additional hypotheses signals genuine reasoning, minimally influenced by specific wording.

The *CSS* prompt is similar in complexity to *CSP*, implying similar performance expectations. However, a notable bias towards *clear sky* impacts the results. Findings indicate that the *"and"* constraint behaves almost like an *"or"* constraint, demonstrating the model's ability in basic logic, although specific prompt formulations may require refinement. Conversely, the *"or"* constraint is ineffective, consistently favouring the earlier rule when two solutions are viable, supporting the idea of rule sequence affecting the model's treatment of rules.

The experiment with *CSW* prompt, supported by *CSP* results, indicates the model's competence in handling numerical values. Instances where *strong wind* was chosen instead of *wind* for a speed of 50 km/h might be resolved through improved prompt formulation. However, for more intricate scenarios, utilizing the calculator tool within LangChain could be advisable. On the contrary, the *CSA* prompt yielded unsatisfactory results and requires reconsideration.

The discourse planning stage poses challenges, as genuine planning requires a generation setting, which *Flan-T5* struggles with. The prompt *P1* as an inference task adeptly handled sequence assignment, revealing a notable disparity between this method and actual discourse plan generation. Moreover, prompt *P2* exposed an issue with integrating messages into the plan, mainly impacting the *PSTW* ordering in 80% of cases, indicating a link to a specific prompt formulation.

In the realization stage, the LangChain framework proves beneficial for combining strengths of different models. The inclusion of a fluent model like *Falcon* is valuable, although some hallucination issues need to be addressed. The problem of selecting alternative phrasing instead of "wind" might be resolved by choosing the appropriate adjective in the content selection stage, thereby eliminating the need for hallucination. Other issues could be mitigated by implementing an extended pipeline with multiple steps and incremental reformulations, starting with simpler messages and then gradually combining and refining them.

Chapter 5

Conclusion

5.1 Contributions and Future Work

The code and model outputs are available on the GitHub repository [54].
The main contributions of this work provide answers to the research questions.
RQ1: *How can abductive reasoning be implemented in language models through prompting?*

Flan-T5, only model capable of abductive reasoning, showed best results on the *ART* dataset [22] using zero-shot prompts. Few-shot prompts did not yield accuracy gains, aligning with [1]'s findings. Despite lacking dedicated abductive training, the model's training on deductive tasks [55] proved advantageous and transferable. However, other models failed to exceed random choice accuracy. *GPT-2* struggled with interpretability, partially eased by the few-shot *F1* prompt. *LLaMA* and *Falcon* faced answer bias issues, which were mitigated using prompts *F1* and *Z3* for LLaMA, and *F2* for *Falcon*. Further details are in Chapter 4.3.1.

To comprehensively explore abductive reasoning through prompting, using additional models fine-tuned for basic reasoning abilities would be beneficial. The qualitative analysis of *Flan-T5* results indicates that this specific task could

gain from increased world knowledge via larger models. Moreover, incorporating temporal knowledge would be valuable, as it is a consistent challenge in misclassified instances. Intriguingly, the model also occasionally struggled with sentences differing by just one word (e.g., *pie* and *pizza*), warranting further investigation.
RQ2: *When prompted with abductive tasks, how do different language models compare?*

The *Flan-T5 XXL* model achieved a remarkable 85% accuracy with zero-shot prompt on the *ART* dataset [22], outperforming significantly larger models (Table 3.1) from [1]. It was the sole model capable of handling symbolic representation, akin to CoS [39]. Among the other models, the abduction task proved challenging. *Falcon* demonstrated some ability in elicitive prompting, *LLaMA* excelled in following few-shot prompts, while *GPT-2* encountered struggles even within that context. Further insights are available in Chapter 4.3.2, supplemented by the comparative overview in Table 4.1 illustrating the models' performance across sizes. Future research directions could involve diversifying model selection and utilizing larger versions of the used models.
RQ3: *Can abductive reasoning be effectively employed for the purpose of content selection and discourse planning in data-to-text generation?*

Yes, our data-to-text generation pipeline demonstrates the potential of this approach. By utilizing smaller, non-fine-tuned language models within the LangChain framework, we prompted them for abduction inference tasks to generate simplified weather forecasts. While content selection proves quite effective, discourse planning poses challenges and could benefit from advanced models and a generative approach. For further details, see Chapter 3.3 and specific prompting recommendations in Chapter 4.4. This adds to previous research, like [30, 61, 62], which explored models trained specifically for each stage. We employed abductive reasoning in an pursuit of a more symbolic planning but still in the language model, without using external planners as in [50].

There are several potential paths for future investigation, such as utilizing larger and fine-tuned models for enhanced outcomes, and investigating the model's differential treatment of rules based on their order. Application-wise, time-specific messages for weather phenomena could be generated and combined in subsequent steps. Addressing the challenges observed in the discourse planning stage and devising strategies to tackle actual planning within a generation framework presents another compelling direction. Additionally, the proposed utilization of abduction for personalized content targeting specific groups (e.g., farmers or hikers in the context of meteorology), resonates with research like [51].

5.2 Limitations

The main limitation of this work is our use of relatively small models due to cost considerations. Emergent properties appear in models with over 10 billion parameters [4], yet we used only one such model, *Flan-T5 XXL*. Our choice to prompt models without fine-tuning presents another constraint, especially for *RQ3*, as fine-tuned models offer better capabilities and most of the used models were not tuned on reasoning tasks. Moreover, we exclusively used manual prompt engineering, potentially missing out on better optimization with automatic methods. Additionally, even smaller models require significant runtime for the entire *ART* dataset, with *T5-large* and *GPT-2* taking about 4 hours, and *LLaMA* around 8 hours. While HuggingFace API calls are quick, there is an hourly prompt limit.

References

[1] F. Xu, Q. Lin, J. Han, T. Zhao, J. Liu, and E. Cambria, “Are large language models really good logical reasoners? a comprehensive evaluation from deductive, inductive and abductive views,” *arXiv preprint arXiv:2306.09841*, 2023. viii, 15, 16, 17, 25, 41, 42

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017. 1, 5

[3] T. OpenAI, “Chatgpt: Optimizing language models for dialogue,” *OpenAI*, 2022. 1

[4] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, “Emergent abilities of large language models,” *arXiv preprint arXiv:2206.07682*, 2022. 1, 5, 43

[5] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg *et al.*, “Sparks of artificial general intelligence: Early experiments with gpt-4,” *arXiv preprint arXiv:2303.12712*, 2023. 1

[6] A. Creswell, M. Shanahan, and I. Higgins, “Selection-inference: Exploiting

large language models for interpretable logical reasoning," *arXiv preprint arXiv:2205.09712*, 2022. 1

[7] B. Peng, M. Galley, P. He, H. Cheng, Y. Xie, Y. Hu, Q. Huang, L. Liden, Z. Yu, W. Chen *et al.*, "Check your facts and try again: Improving large language models with external knowledge and automated feedback," *arXiv preprint arXiv:2302.12813*, 2023. 1

[8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020. 1, 5, 6

[9] G. Mialon, R. Dessì, M. Lomeli, C. Nalmpantis, R. Pasunuru, R. Raileanu, B. Rozière, T. Schick, J. Dwivedi-Yu, A. Celikyilmaz *et al.*, "Augmented language models: a survey," *arXiv preprint arXiv:2302.07842*, 2023. 2, 7, 8

[10] M. Saeed, N. Ahmadi, P. Nakov, and P. Papotti, "Rulebert: Teaching soft rules to pre-trained language models," *arXiv preprint arXiv:2109.13006*, 2021. 2

[11] W. Chen, X. Ma, X. Wang, and W. W. Cohen, "Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks," *arXiv preprint arXiv:2211.12588*, 2022. 2

[12] H. Sun, L. B. Soares, P. Verga, and W. W. Cohen, "Adaptable and interpretable neural memory over symbolic knowledge," 2021. 2

[13] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *arXiv preprint arXiv:2205.11916*, 2022. 2, 6

[14] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou, "Chain of thought prompting elicits reasoning in large language models," *arXiv preprint arXiv:2201.11903*, 2022. 2, 6

[15] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023. 2, 5, 6, 7, 8, 13

[16] M. Shanahan, "The event calculus explained," in *Artificial intelligence today: Recent trends and developments.* Springer, 2001, pp. 409–430. 2, 3, 10, 11

[17] F. Yu, H. Zhang, and B. Wang, "Nature language reasoning, a survey," *arXiv preprint arXiv:2303.14725*, 2023. 2, 9, 10

[18] P. Clark, O. Tafjord, and K. Richardson, "Transformers as soft reasoners over language," *arXiv preprint arXiv:2002.05867*, 2020. 2

[19] H. Zhang, J. Huang, Z. Li, M. Naik, and E. Xing, "Improved logical reasoning of language models via differentiable symbolic programming," *arXiv preprint arXiv:2305.03742*, 2023. 2

[20] S. Saha, S. Ghosh, S. Srivastava, and M. Bansal, "Prover: Proof generation for interpretable reasoning over rules," *arXiv preprint arXiv:2010.02830*, 2020. 2

[21] S. Qiao, Y. Ou, N. Zhang, X. Chen, Y. Yao, S. Deng, C. Tan, F. Huang, and H. Chen, "Reasoning with language model prompting: A survey," *arXiv preprint arXiv:2212.09597*, 2022. 2, 5, 7, 8

[22] C. Bhagavatula, R. L. Bras, C. Malaviya, K. Sakaguchi, A. Holtzman, H. Rashkin, D. Downey, S. W.-t. Yih, and Y. Choi, "Abductive common-

sense reasoning," *arXiv preprint arXiv:1908.05739*, 2019. vii, 2, 9, 10, 15, 38, 41, 42

[23] Z. Yang, X. Du, R. Mao, J. Ni, and E. Cambria, "Logical reasoning over natural language as knowledge representation: A survey," *arXiv preprint arXiv:2303.12023*, 2023. 2, 9, 10

[24] X. Shi, S. Xue, K. Wang, F. Zhou, J. Y. Zhang, J. Zhou, C. Tan, and H. Mei, "Language models can improve event prediction by few-shot abductive reasoning," *arXiv preprint arXiv:2305.16646*, 2023. 2, 10

[25] E. Goldberg, N. Driedger, and R. I. Kittredge, "Using natural-language processing to produce weather forecasts," *IEEE Expert*, vol. 9, no. 2, pp. 45–53, 1994. 2

[26] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, no. 1, pp. 57–87, 1997. 2, 3, 11, 12

[27] T. Liu, K. Wang, L. Sha, B. Chang, and Z. Sui, "Table-to-text generation by structure-aware seq2seq learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018. 2

[28] J. G. Corbelle, A. B. Diz, J. Alonso-Moral, and J. Taboada, "Dealing with hallucination and omission in neural natural language generation: A use case on meteorology." in *Proceedings of the 15th International Conference on Natural Language Generation*, 2022, pp. 121–130. 2

[29] O. Dušek, D. M. Howcroft, and V. Rieser, "Semantic noise matters for neural natural language generation," *arXiv preprint arXiv:1911.03905*, 2019. 2

[30] R. Puduppully, L. Dong, and M. Lapata, “Data-to-text generation with content selection and planning,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 6908–6915. 3, 13, 42

[31] A. Balakrishnan, J. Rao, K. Upasani, M. White, and R. Subba, “Constrained decoding for neural nlg from compositional representations in task-oriented dialogue,” *arXiv preprint arXiv:1906.07220*, 2019. 3, 13

[32] M. Shanahan, “An abductive event calculus planner,” *The Journal of Logic Programming*, vol. 44, no. 1-3, pp. 207–240, 2000. 3, 10, 11

[33] A. M. Turing, “Computing machinery and intelligence,” *Mind*, vol. 59, no. 236, pp. 433–460, 1950. [Online]. Available: http://www.jstor.org/stable/2251299 4

[34] J. Hutchins, “The first public demonstration of machine translation: the georgetown-ibm system, 7th january 1954,” *noviembre de*, 2005.

[35] J. R. Searle, “Minds, brains, and programs,” *Behavioral and brain sciences*, vol. 3, no. 3, pp. 417–424, 1980.

[36] P. Blunsom, “Hidden markov models,” *Lecture notes, August*, vol. 15, no. 18-19, p. 48, 2004. 4

[37] L. R. Medsker and L. Jain, “Recurrent neural networks,” *Design and Applications*, vol. 5, pp. 64–67, 2001. 5

[38] A. Graves and A. Graves, “Long short-term memory,” *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012. 5

[39] H. Hu, H. Lu, H. Zhang, W. Lam, and Y. Zhang, “Chain-of-symbol prompting elicits planning in large langauge models,” *arXiv preprint arXiv:2305.10276*, 2023. 6, 17, 42

[40] J. Huang and K. C.-C. Chang, "Towards reasoning in large language models: A survey," *arXiv preprint arXiv:2212.10403*, 2022. 7, 8, 9

[41] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, O. Bousquet, Q. Le, and E. Chi, "Least-to-most prompting enables complex reasoning in large language models," *arXiv preprint arXiv:2205.10625*, 2022. 8

[42] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022. 8

[43] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *arXiv preprint arXiv:2305.10601*, 2023. 8

[44] D. Walton, *Abductive reasoning.* University of Alabama Press, 2014. 8, 9

[45] J. Jung, L. Qin, S. Welleck, F. Brahman, C. Bhagavatula, R. L. Bras, and Y. Choi, "Maieutic prompting: Logically consistent reasoning with recursive explanations," *arXiv preprint arXiv:2205.11822*, 2022. 9, 10

[46] T. Eiter and G. Gottlob, "The complexity of logic-based abduction," *Journal of the ACM (JACM)*, vol. 42, no. 1, pp. 3–42, 1995. 9

[47] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung *et al.*, "A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity," *arXiv preprint arXiv:2302.04023*, 2023. 10, 16

[48] J. Li, T. Tang, W. X. Zhao, and J.-R. Wen, "Pretrained language models

for text generation: A survey," *arXiv preprint arXiv:2105.10311*, 2021. 12, 13

[49] T. C. Ferreira, C. van der Lee, E. Van Miltenburg, and E. Krahmer, "Neural data-to-text generation: A comparison between pipeline and end-to-end architectures," *arXiv preprint arXiv:1908.09022*, 2019. 13

[50] A. Moryossef, Y. Goldberg, and I. Dagan, "Step-by-step: Separating planning from realization in neural data-to-text generation," *arXiv preprint arXiv:1904.03396*, 2019. 13, 43

[51] H. Zhang, H. Song, S. Li, M. Zhou, and D. Song, "A survey of controllable text generation using transformer-based pre-trained language models," *arXiv preprint arXiv:2201.05337*, 2022. 13, 43

[52] I. Hugging Face. Hugging Face. [Online]. Available: https://huggingface.co/ 14

[53] C. Harrison. LangChain. 14, 21

[54] K. Onderková. Diploma Thesis Code on Github. [Online]. Available: https://github.com/Kristyna-Navitas/PromptingLanguageModelsforAbductiveReasoningTasks 14, 41

[55] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, "Scaling instruction-finetuned language models," *arXiv preprint arXiv:2210.11416*, 2022. 15, 41

[56] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023. 15

[57] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019. 15

[58] Y. X. ZXhang, Y. M. Haxo, and Y. X. Mat, "Falcon llm: A new frontier in natural language processing," *AC Investment Research Journal*, vol. 220, no. 44, 2023. 15

[59] N. Young, Q. Bao, J. Bensemann, and M. Witbrock, "Abductionrules: Training transformers to explain unexpected inputs," *arXiv preprint arXiv:2203.12186*, 2022. 15

[60] P. Liang, M. I. Jordan, and D. Klein, "Learning semantic correspondences with less supervision," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009, pp. 91–99. 21

[61] Y. Su, D. Vandyke, S. Wang, Y. Fang, and N. Collier, "Plan-then-generate: Controlled data-to-text generation via planning," *arXiv preprint arXiv:2108.13740*, 2021. 42

[62] Z. Kasner and O. Dušek, "Neural pipeline for zero-shot data-to-text generation," *arXiv preprint arXiv:2203.16279*, 2022. 42

Appendix A

Detailed Results for Abductive Prompting

Further Insight into *Flan-T5* Results

As seen in Table 4.2 there is a slight drop in performance with prompt *Z3* for the *XXL* model, suggesting that the more complex prompt is not helpful. The drops in performance for the *base* model are due to its size, as for prompt *Z3*, where the model follows the prompt only in 74% of case and with prompt *Z2*, where it is biased towards one answer. From *Z3* and *Z4* prompt it seems that the right ordering of the sentences may not be important for the performance.

Table 4.3 shows that explicitly stating the hypothesis with *F1* in few-shot setting, does not seem necessary, except for the *large* model, where it reduces not answered cases similarly to reformulated prompts. The *F2* prompt influences only the performance of the *base* model, where the bigger models are not influenced. The *base* model was not doing well with this prompt in the zero-shot setting (*Z4*), and in few-shot setting this worsens.

The *large* model's confusion with the zero-shot CoT *T0* prompt leads to outputs that closely resemble random choice, likely due to its size.

Further Insight into *Falcon* Results

The prompt *Z4* resulted in numerous *NA* answers, which is probably due to some error in calling the HuggingFace API Interface. Evaluated only on non-empty answers it gives the same accuracy as other prompts.