PARADISE: Evaluating Implicit Planning Skills of Language Models with Procedural Warnings and Tips Dataset

Arda Uzunoğlu[‡], **Abdulfattah Safa**[†], **Gözde Gül Şahin**[†]

[‡]Computer Science Department, Johns Hopkins University, Maryland, USA
[†]Computer Engineering Department, Koç University, Istanbul, Türkiye
[†]KUIS AI, Koç University, Istanbul, Türkiye
[†]https://gglab-ku.github.io/

Abstract

Recently, there has been growing interest within the community regarding whether large language models are capable of planning or executing plans. However, most prior studies use LLMs to generate high-level plans for simplified scenarios lacking linguistic complexity and domain diversity, limiting analysis of their planning abilities. These setups constrain evaluation methods (e.g., predefined action space), architectural choices (e.g., only generative models), and overlook the linguistic nuances essential for realistic analysis. To tackle this, we present PARADISE, an abductive reasoning task using Q&A format on practical procedural text sourced from wikiHow. It involves warning and tip inference tasks directly associated with goals, excluding intermediary steps, with the aim of testing the ability of the models to infer implicit knowledge of *the plan* solely from the given goal. Our experiments, utilizing fine-tuned language models and zero-shot prompting, reveal the effectiveness of task-specific small models over large language models in most scenarios. Despite advancements, all models fall short of human performance. Notably, our analysis uncovers intriguing insights, such as variations in model behavior with dropped keywords, struggles of BERT-family and GPT-4 with physical and abstract goals, and the proposed tasks offering valuable prior knowledge for other unseen procedural tasks. The PARADISE dataset and associated resources are publicly available for further research exploration with https://github.com/GGLAB-KU/paradise.

1 Introduction

Recent breakthroughs in emergent (or lack of) abilities of large language models (LLMs) have given rise to empirical studies that employ language models as planners (Huang et al., 2022; Zhao et al., 2023; Song et al., 2023) (i.e., agentic models) and analysis studies that investigate their planning capabilities (Valmeekam et al., 2023; Pallagani et al., 2023; Valmeekam et al., 2022). Majority of these studies use toy simulation environments like ALFRED (Shridhar et al., 2020), BlocksWorld, and VirtualHome (Puig et al., 2018) which have little lexical and domain variance and limited number of actions (e.g., verbs). Additionally, the planning task is mostly formulated as a generation problem which can only be evaluated on the closed problem domain; and models with decoder-based architectures. Hence evaluating *open-domain* planning abilities for a wide range of models still remain as a challenge.

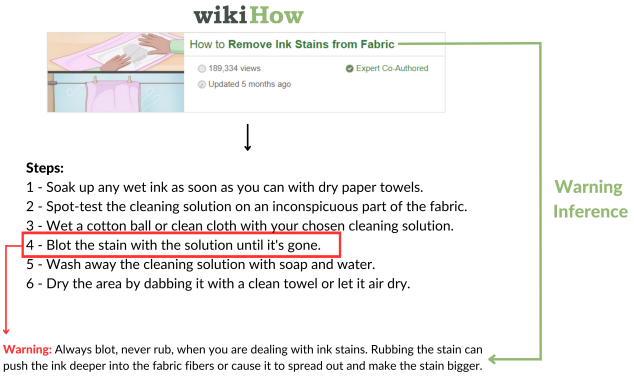

Figure 1: A procedural tutorial on "Removing Ink Stains from Fabric". Here, one can damage the fabric if they ignore the warning "Always blot, never rub, when dealing with ink stains".

Planning requires a combination of a wide range of complex reasoning abilities. One line of research focuses on distinct set of reasoning abilities (e.g., commonsense (Huang et al., 2019), arithmetic (Cobbe et al., 2021), logical (Han et al., 2022), temporal (Wang and Zhao, 2023) etc...) of language models

on more realistic, open-domain text. However, such open-domain text mostly does not contain *a goal* or *a plan*, hence lack the complex linguistic phenomenon (e.g., implicit relations, complex temporal, and co-reference links, large event-cause chains etc...) that is common in procedural text. On the other hand, existing studies that utilize plans as a test bed for reasoning are mostly limited to extracting direct and explicit relations (Dalvi et al., 2019; Zhang et al., 2020b), i.e., more simplistic reasoning compared to implicit reasoning. Here, we formulate the implicit planning skills as abductive reasoning skills over the procedural plans with missing information. We hypothesize that a model with planning abilities would be able to *infer the warnings and tips* about the plan *without seeing the explicit instructions* (see Fig 1). Although abductive reasoning has been studied in the past, they either employ additional source of information (Huang et al., 2019; Bhagavatula et al., 2020) or focus on consecutive elements (Zellers et al., 2018, 2019; Tandon et al., 2019), both of which diminish the notion of implicity.

In this work, we present PARADISE, an extensive, expert-curated dataset for warning and tip inference tasks covering a wide range of domain derived from wikiHow[1]. Unlike previous works, our tasks focus on the implicit relationship between goals and warnings/tips, bypassing intermediate steps (instructions). This requires a model to possess implicit knowledge of intermediate steps (i.e., the plan) solely based on the provided goal in the absence of explicit instructions. Furthermore, we use a question answering formulation, which allows for easier evaluation with standard metrics, and testing of broader-range of model architectures including encoder-based ones. We establish robust baselines by fine-tuning pretrained language models like DeBERTa (He et al., 2020) and zero-shot prompting with a varied set of large language models such as Mistral-7B (Jiang et al., 2023) and GPT-4 (OpenAI, 2023b). Our extensive experiments address a broad range of research questions, delving into the relationship between memorization and performance (§4.1), the differences in failures between PLMs and LLMs (§4.2), and the knowledge transfer capacity of the proposed tasks to unseen tasks (§4.5). We observe that fine-tuning small models tailored to specific tasks proves more effective than zero-shot prompting across all LLMs, including GPT-4. However, it's noteworthy that all models, despite these efforts, still lag behind human performance. Our exhaustive analysis also provide interesting insights such as large models getting less affected from dropping matched keywords; BERT-family struggling more with physical goals, while GPT-4 struggling with abstract, digital and social objectives; and proposed tasks providing beneficial prior knowledge to unseen procedural tasks. We release all the resources at https://github.com/GGLAB-KU/paradise.

2 PARADISE

Initially, we augment the wikiHow corpus (Zhang et al., 2020b) by integrating it with a recent compilation[2] of 21K tutorials. The extended corpus maintains the JSON format, except for the segregation of warnings and tips, an example of which can be seen in Appendix A. Each wikiHow tutorial comprises procedural steps to achieve its objective, with some tutorials featuring step-specific or general warnings and tips. We automatically generate downstream task data incorporating these warnings and tips, as elaborated in Sec. 2.2. The process also involves expert human annotation, detailed in Sec. 2.3.

2.1 Task Formulation

We define warning and tip inference tasks as multiple-choice question answering tasks, in which a system needs to choose the correct warning or tip for a given goal among candidates. In this context, goals are questions, while warnings and tips are the choices. An example for both tasks can be seen in Fig. 2.

2.2 Candidate Sampling

Acquiring the goals and positive candidates is straightforward, involving iterative selection from each tutorial in our corpus. For negative candidate sampling, we modify the approach outlined by Zhang et al. (2020b). In contrast to their reliance solely on verbs, we note that verbs prove inadequate in capturing meaning because warnings and tips, on average, are much longer than individual steps (~40 versus ~11 tokens). This leads to the generation

[1] https://www.wikihow.com

[2] Scrape date: November, 2022

	Category Distribution									Size		
	Other	**C&E**	**HE**	**F&E**	**H&C**	**H&G**	**E&C**	**F&B**	**PC&S**	**Train**	**Dev**	**Test**
WikiHow Corpus	27.25%	14.51%	10.18%	10.09%	9.27%	8.89%	7.35%	6.63%	5.83%		133K	
Warning Inference	34.75%	7.09%	14.13%	7.26%	4.12%	14.02%	5.19%	4.92%	8.52%	33K	5K	500
Tip Inference	30.13%	7.45%	11.33%	10.61%	5.44%	12.11%	7.46%	5.29%	10.18%	71K	5K	500

Table 1: Category distribution and size of PARADISE. C&E - Computer and Electronics, HE - Health, F&E - Food and Entertaining, H&C - Hobbies and Crafts, H&G - Home and Garden, E&C - Education and Communications, F&B - Finance and Business, PC&S - Personal Care and Style.

1. Goal: Sit up Straight at a Computer
   (a) Remember that people can see some of your surroundings you while you chat. Be mindful of what is in the camera's field of view.
   (b) **Do not remain in any one position in front of a computer for too long.**
   (c) Avoid moving around in this pose. Any movements you make within the pose should be deliberate and slow.
   (d) Keep an appropriate distance between your eyes and computer screen.
2. Goal: Avoid Oil Splatter when Frying
   (a) Remember to have lots of sides apart from just the barbecued food.
   (b) Wear clear, plastic gloves if you are going to use your hands to mix the meat.
   (c) Never use extra virgin olive oil to stir-fry. It has a low smoking point.
   (d) **Wear long sleeves when you plan on frying food.**

Figure 2: Example questions for warning (1) and tip (2) inference tasks. Correct choices are **bold**.

of low-quality negative candidates. To address this limitation, we enhance our negative candidate sampling strategy by incorporating embeddings of noun tokens.

We begin by encoding each warning and tip using BERT (Devlin et al., 2019). We calculate the average of verb and noun tokens, identified with spaCy (Honnibal et al., 2020). Subsequently, we employ FAISS (Johnson et al., 2019) to conduct a semantic similarity search, identifying the top three warnings and tips with the highest cosine similarity score relative to the positive candidate.

Following Zhang et al. (2020b), we randomly reassign one of the negative candidates as positive and correct the labels and goals accordingly with a probability of 0.15 to avoid sampling bias and filter the examples as described in Appendix B.

2.3 Test Set Construction

As our datasets are automatically generated, they may include undesired elements like multiple plausible candidates for a given goal. For instance, consider the goal "Deal with Your Step Mother", which has a positive candidate of "Stay connected with relatives such as grandparents and close friends for extra support" and a negative candidate of "Recruit help from friends and family". Although the negative candidate is chosen due to its high semantic similarity with the positive candidate, it is also a reasonable choice for the given goal, introducing noise into the dataset. To mitigate such issues, we employ expert annotation to validate the test splits.

The expert annotation process consists of three stages. First, experts verify that each example contains no more than one plausible candidate. Second, they meticulously examine examples to ensure that the positive candidate is genuinely relevant to the content of the associated wikiHow tutorial. Finally, experts assess the appropriateness of examples for gauging reasoning skills, excluding instances that demand expert-level knowledge or domain-specific high-level information. This annotation process yields approximately 80% of annotations as valid examples. Consequently, the test splits for each task are expanded with such valid examples from the pool of automatically generated examples until reaching the predetermined size of 500 examples.

Apart from expert annotation, we leverage the dataset cartography tool (Swayamdipta et al., 2020) to uphold the high quality of our data, probe our datasets, and gain a deeper understanding of their features. Further details can be found in Appendix C.

2.4 Dataset Statistics

The statistics of the corpus and final datasets are given in Table 1. We specify the validation and test split sizes as 5K and 500, respectively, with the remaining data serving as the training set. Tip inference dataset is nearly twice the size of the warning inference dataset, but the average token counts per goal and candidate are comparable (~7 for goal, ~40 for candidate). We employ a nearly uniform sampling approach across various categories to ensure a high level of domain diversity.

3 Experimental Setup

To evaluate language models in our tasks, we establish two setups: 1) finetuning setup for pretrained encoder LMs such as BERT (Devlin et al., 2019) and 2) zero-shot setup for large language models such as GPT-4 (OpenAI, 2023b).

3.1 Finetuning Setup

We fine-tune a set of models from the BERT family: DistilBERT (Sanh et al., 2019), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), DeBERTa (He et al., 2020), which show strong performance in procedural and reasoning tasks (Zhou et al., 2022; Tandon et al., 2019; Zellers et al., 2019; Zhang et al., 2020b). For fine-tuning, we concatenate each candidate (warning or tip) with the question (goal) using a `[CLS]` token, i.e., the model receives `[CLS]` `question` `[SEP]` `candidate` as input. Subsequently, we apply an additional projection layer, followed by a softmax function that receives the representation of the `[CLS]` token for each `candidate`. The `candidate` with the highest probability is selected as the answer. The models are optimized through cross-entropy loss. Further implementation details can be found in Appendix D.

3.2 Zero-Shot Setup

We identify five popular and capable[3] large language models that are diverse in architecture, scale, avalability, and performance, namely as GPT-4[4] (OpenAI, 2023b), PALM-2 [5] (Anil et al., 2023), LLaMA-2 70B (Touvron et al., 2023), Mistral 7B[6] (Jiang et al., 2023), and Vicuna 33B (Chiang et al., 2023).

We first perform preliminary experiments with default prompts on a small subset of the validation set. We, then, iteratively refine the prompts to fit the specific model's template. For instance Vicuna expects a certain template with `###Human` and `###Assistance` roles specified in the text. We use the respective model APIs, where available. Preliminary tests on the subsets were conducted for each model to identify optimal `temperature` and `top_p` parameters. The best-performing configurations were then applied to the entire datasets for a thorough evaluation. Further details on the prompt templates and parameters are given in Appendix E.

Model	Warning	Tip
Random	25.0	25.0
Majority	26.0	26.0
PLMs		
DistilBERT	22.44 ±3.88	21.48 ±4.84
BERT	**25.52 ±4.56**	**26.88 ±5.02**
RoBERTa	20.88 ±4.80	20.36 ±4.20
DeBERTa	23.40 ±6.76	22.60 ±8.61
Fine-Tuned PLMs		
DistilBERT	82.16 ±0.79	87.48 ±0.65
BERT	83.92 ±0.68	89.80 ±0.91
RoBERTa	87.92 ±0.60	91.00 ±0.34
DeBERTa	**90.68 ±0.41**	**93.68 ±0.48**
Open-Source LLMs		
Mistral 7B	**71.8**	**72.4**
Vicuna 33B	53.2	57.0
LLaMA-2 70B	65.2	64.5
Proprietary LLMs		
PALM-2	83.6	82.4
GPT-4	**86.2**	**88.8**
Human	**94.0**	**96.0**

Table 2: Main accuracy results for fine-tuning and zero-shot setups.

4 Experiments and Results

We experiment with the PLMs and LLMs on PARADISE using the setup explained in Sec 3. We also

[3]These LLMs are chosen from the models that rank high in the Hugging Face Chatbot Leaderboard.

[4]Model variant: `gpt-4-1106-preview`

[5]Model variant: `text-bison`

[6]Model variant: `Mistral-7B-Instruct-v0.1`

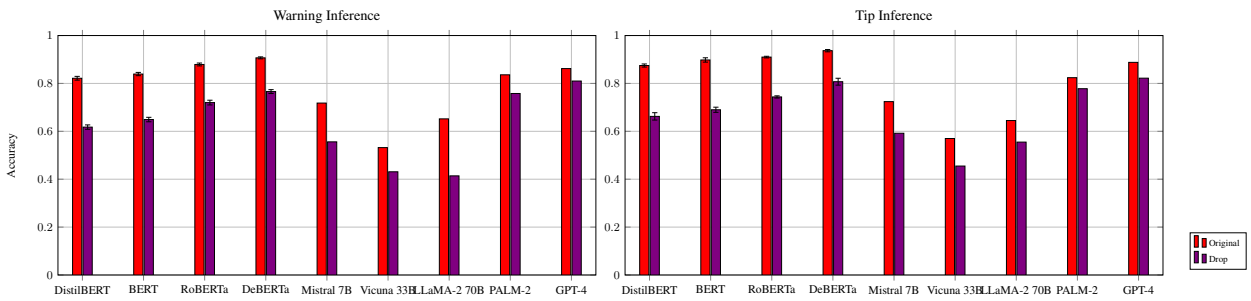

Figure 3: Model performances tested on manipulated test data.

calculate the random and majority baselines, and evaluate the human performance by averaging the accuracy of two human annotators[7] on a random set of 100 examples. Our main results are given in Table 2.

Fine-tuning Among the fine-tuned models, DeBERTa performs the best in both tasks; yet, it still falls behind human performance. Considering DeBERTa's performance in previous abductive reasoning datasets, such as CosmosQA (Huang et al., 2019), SWAG (Zellers et al., 2018), and HellaSWAG (Zellers et al., 2019), such results indicate that its success is not task-specific and its performance is transferable to warning and tip inference. DistilBERT, BERT, and RoBERTa cannot perform on par with DeBERTa and fall well behind human performance, although they perform considerably well on the proposed tasks. All models perform better in tip inference compared to warning inference.

Zero-shot Although their performances are worse than fine-tuned PLMs, proprietary LLMs perform considerably better than open-source LLMs. GPT-4, which also tops many benchmarks (OpenAI, 2023b), is the best-performing LLM, while PALM-2 is a close runner-up. One surprising finding is that the performances of open-source LLMs are not correlated with their size, as Mistral 7B outperforms Vicuna 33B and LLaMA-2 70B in both tasks. Similar to PLMs, LLMs also perform better in tip inference compared to warning inference, with LLaMA-2 70B being an exception.

To gain further insights on behaviours of the models we ask the following research questions:

- **RQ1:** Do the models perform well due to simple keyword matching?
- **RQ2:** Do different model families fail on different instances? Is there a certain pattern?
- **RQ3:** How does the performance compare for the explicit (i.e., directly related to a step) and implicit warnings/tips (i.e., more general and not directly related to any of the steps)?
- **RQ4:** Are the models also good at the reverse task, i.e., can they find the goal most related to the warning/tip?
- **RQ5:** Can the proposed tasks help improve performance in other procedural tasks?

4.1 RQ1: Keyword Matching

If the goal and only one candidate share a common keyword, an example might become trivial and the task might develop into simple keyword matching. For example, when the goal and the positive candidate are about cats while negative candidates are about other animals, the positive candidate becomes easily distinguishable. Therefore, we drop such keywords in both positive and negative candidates in order to evaluate our tasks' dependency on keywords.

This manipulation, averaging approximately 2 tokens per candidate, induces a 4.5% change on average. As illustrated in Fig. 3, the omission of such keywords results in a 15% to 20% decrease in prediction accuracy for Pre-trained Language Models (PLMs). The impact diminishes with increasing original model performance; DistilBERT is most affected, while DeBERTa is least affected. In contrast, Language Models (LLMs) experience a milder accuracy decline of 5% to 15%. In addition to dropping keywords, we also experiment with other keyword manipulation methods as detailed in Appendix F.

[7]Two university students majoring in computer science, between the ages of 20 - 24.

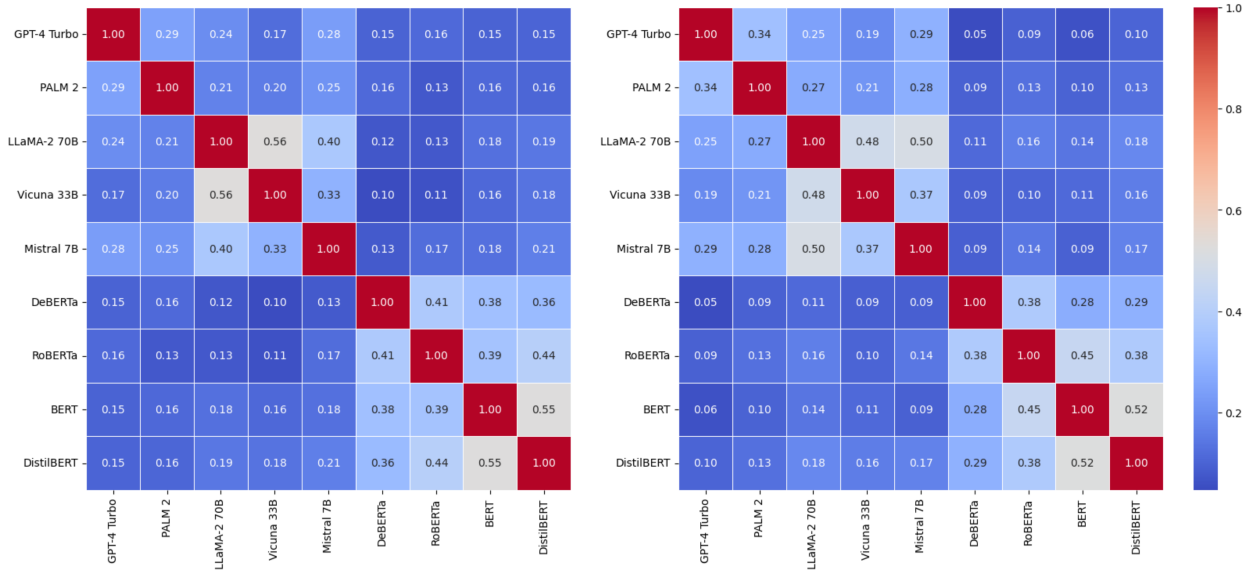

Figure 4: Correlation matrices of incorrect predictions of each model for tip (left) and warning (right) inference.

4.2 RQ2: Failures of Different Model Families

To better understand the behaviours of the models we experiment with, we generate correlation matrices for their incorrect predictions in both tasks. As depicted in Fig. 4, models within the same group (PLMs, Open-Source LLMs, and Proprietary LLMs) exhibit the highest correlation. Notably, incorrect predictions diverge more between PLMs and LLMs, while open-source LLMs and proprietary LLMs display higher inter-group correlation. Task-wise, correlation levels remain consistent, underscoring data quality and overall consistency.

As evident in Fig. 4, DeBERTa and GPT-4 exhibit divergent failure patterns. To understand their distinctions in distinguishing PLMs and LLMs, we manually inspect instances of failure. Our analysis reveals that DeBERTa struggles more with tangible, physical, and craft-related goals, while GPT-4 encounters challenges with abstract, digital, and social objectives. We validate these findings by generating the category distribution of unique failures for DeBERTa and GPT-4, as shown in Table 3. Notably, GPT-4 falters in social and digital categories like Youth, Relationships, and Computer & Electronics, while DeBERTa encounters difficulties in tangible categories such as Sports & Fitness, Pets & Animals, and Home & Garden. Specific instances of failures for DeBERTa and GPT-4 are detailed in Appendix G.

Rank	DeBERTa		GPT-4	
	Warning	Tip	Warning	Tip
#1	H&G (24.5%)	F&E (25.0%)	H&G (24.4%)	HE (18.1%)
#2	S&F (12.8%)	H&G (21.9%)	HE (16.3%)	F&E (13.4%)
#3	E&C (11.7%)	HE (15.6%)	C&E (13.0%)	C&E (10.7%)
#4	PC&S (11.7%)	E&C (10.9%)	RE (9.8%)	YO (8.7%)
#5	P&A (8.5%)	P&A (9.4%)	PC&S (8.9%)	PC&S (8.7%)

Table 3: Top categories that DeBERTa and GPT-4 fail.

4.3 RQ3: Implicit versus Explicit

As outlined in Sec. 2, warnings and tips within the dataset exhibit a distinction: some are specific to individual steps, while others are general. Although they are related to steps from the wikiHow tutorials, step-specific warnings and tips are not matched with their associated steps manually by editors. To assess the implicit reasoning skills of language models, we curate distinct subsets of test splits for both tasks, comprising warnings and tips demonstrating high semantic similarity with steps from relevant wikiHow tutorials. Employing SBERT (Reimers and Gurevych, 2019) for encoding steps and warnings/tips, we conduct a semantic similarity search. We retain examples with a cosine similarity score

Model	Warning			Tip		
	All	Sim>0.5	Sim>0.75	All	Sim>0.5	Sim>0.75
Fine-Tuned PLMs						
DistilBERT	82.16 ±0.79	**87.12 ±0.66**	85.71 ±0.00	87.48 ±0.65	88.43 ±0.67	**90.02 ±1.46**
BERT	83.92 ±0.68	86.84 ±0.59	**90.29 ±2.29**	89.80 ±0.91	91.10 ±0.89	**92.69 ±0.77**
RoBERTa	87.92 ±0.60	89.57 ±0.33	**92.57 ±1.40**	91.00 ±0.34	91.83 ±0.26	**91.93 ±0.77**
DeBERTa	90.68 ±0.41	91.63 ±0.58	**94.86 ±2.14**	93.68 ±0.48	95.34 ±0.39	**100.0 ±0.0**
Open-Source LLMs						
Mistral 7B	71.8	64.4	**70.3**	72.4	72.8	**73.6**
Vicuna 33B	53.2	54.1	**59.5**	57.0	58.9	**66.0**
LLaMA-2 70B	**65.2**	58.4	62.2	64.5	70.9	**77.4**
Proprietary LLMs						
PALM-2	83.6	83.2	**86.5**	82.4	82.8	**84.9**
GPT-4	86.2	86.1	**89.2**	88.8	88.7	**92.5**

Table 4: The accuracy results of PLMs and LLMs on different subsets of the test splits with varying level of similarity to the instructions from associated wikiHow tutorials.

surpassing a threshold with at least one step, resulting in 35 warnings and 52 tips with high similarity scores (cosine similarity score with the step > 0.75) and 368 warnings and 382 tips with decent similarity scores (cosine similarity score with the step > 0.5).

We assess the performance of PLMs and LLMs, as detailed in Sec. 3, on subsets outlined in Table 4. Results show improved model performance as the similarity between warnings/tips and steps increases. Higher accuracy in these subsets suggests a strong capacity for implicit reasoning, given that warnings and tips become more representative of intermediate steps with increased similarity. Notably, BERT and DeBERTa excel among PLMs, while Mistral 7B and LLaMA-2 70B lead among LLMs, exhibiting the highest accuracy increase in warning and tip inference tasks, respectively.

4.4 RQ4: Reverse Inference Tasks

If models can effectively reason about the relationship between warnings/tips and goals, it suggests they can correctly identify the goal corresponding to a given warning or tip. To examine this hypothesis, we construct reversed versions of our tasks, requiring the system to select the correct goal for a provided warning or tip. Using the candidate sampling method detailed in Sec 2.2, we randomly select 500 examples for evaluation with fine-tuned PLMs and LLMs. Results in Table 5 indicate zero-shot performances for PLMs align closely with the random baseline, except for DeBERTa, which achieves a 10% higher accuracy. This suggests limited inherent reasoning capabilities over procedural warnings and tips. However, fine-tuned models exhibit a significant performance increase, supporting our hypothesis. In contrast, LLMs maintain similar accuracy scores in reverse tasks without experiencing a performance loss observed in fine-tuned PLMs.

4.5 RQ5: Transfer Learning

To examine the impact of our tasks on reasoning over procedural documents, we conduct cross-tests between warning and tip inference tasks and out-of-domain transfer learning on goal and step inference tasks from Zhang et al. (2020b).

4.5.1 Cross Domain

While categorized separately, both warnings and tips share the common objective of enhancing reader understanding in a wikiHow tutorial. As a result, they often exhibit similarities in structure and semantics, with occasional interchangeability. To assess the generalizability of Pre-trained Language Models (PLMs) across warning and tip inference tasks, we conduct cross-tests. Specifically, we evaluate PLMs fine-tuned on tip inference data using the test split of the warning inference dataset and vice versa.

Model	Reverse Warning	Reverse Tip
Random	25.0	25.0
PLMs		
DistilBERT	20.52 ±3.93	20.00 ±5.27
BERT	25.08 ±3.73	25.40 ±4.93
RoBERTa	26.52 ±5.42	26.36 ±7.03
DeBERTa	**33.76 ±4.83**	**35.72 ±4.99**
Fine-Tuned PLMs		
DistilBERT	65.44 ±1.53	68.48 ±1.25
BERT	69.08 ±1.69	72.36 ±1.18
RoBERTa	72.64 ±1.44	77.00 ±1.26
DeBERTa	**79.92 ±0.63**	**80.44 ±0.79**
Open-Source LLMs		
Mistral 7B	**74.6**	**79.2**
Vicuna 33B	62.8	61.4
LLaMA-2 70B	76.2	76.0
Proprietary LLMs		
PALM-2	83.6	85.8
GPT-4	**86.4**	**86.0**

Table 5: Accuracy results of the reverse task evaluation.

Model	Warning	Tip
Random	25.0	25.0
BERT	84.68 ±0.27	86.20 ±0.77
RoBERTa	88.28 ±0.30	88.80 ±0.72

Table 6: Accuracy results of BERT and RoBERTa fine-tuned with tip inference on warning inference, and vice versa.

As depicted in Table 6, models fine-tuned on tip inference data demonstrate comparable performance to those fine-tuned on warning inference data for the warning inference task. Conversely, models fine-tuned on warning inference data exhibit slightly lower performance than those fine-tuned on tip inference data for the tip inference task. The nearly identical results in cross tests affirm the high similarity between warnings and tips, highlighting their interchangeability.

4.5.2 Out-of-Domain

The goal and step inference tasks focus on identifying goal-step relationships in procedural how-to tutorials. *Goal inference* involves selecting the plausible goal from candidates for a given step, while *step inference* entails the reverse process. Although similar to the proposed tasks, they aim to measuring *explicit* procedural reasoning abilities.

For both step and goal inference tasks, we fine-tune three BERT models: i) BERT trained from scratch, ii) BERT previously fine-tuned on warning inference data, and iii) BERT previously fine-tuned on tip inference data. We report their performances on the test split throughout the training. Notably, prior fine-tuning on warning and tip inference tasks consistently improves performance during training for both goal and step inference tasks, as illustrated in Fig. 5. The second and third BERT models exhibit significantly enhanced zero-shot performances, with average accuracy increases of 21.16% and 22.98% for goal inference, and 34.75% and 39.77% for step inference, respectively. While the performance gap diminishes, the second and third BERT models continue to outperform at the end of training, with average accuracy increases of 2.09% and 2.27% for goal inference, and 0.15% and 0.53% for step inference, respectively.

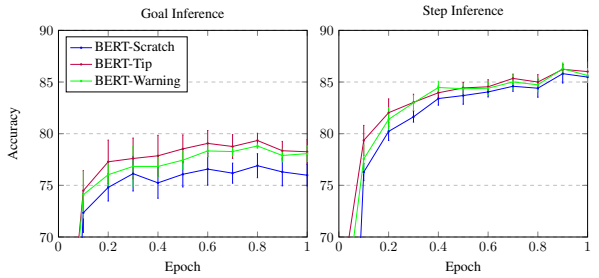

Figure 5: Accuracy of the three BERT variations plotted over the training epoch.

5 Related Work

Commonsense reasoning is a broad domain that branches into a wide range of subdomains such as linguistic reasoning (Şahin et al., 2020; Liu et al., 2022; Lin et al., 2021), abductive reasoning (Tandon et al., 2019; Zellers et al., 2019), reasoning about the physical world (Bisk et al., 2020; Khot et al., 2019; Aroca-Ouellette et al., 2021), temporal reasoning (Zhang et al., 2020b; Qin et al., 2021), etc. (Bhargava and Ng, 2022). Although there exists

some abductive reasoning tasks that involve finding the most likely explanation for a set of incomplete observations (Bhargava and Ng, 2022), they are outside of the domain of procedural language understanding. For example, CosmosQA (Huang et al., 2019) present commonsense abductive reading comprehension and ART (Bhagavatula et al., 2020) propose abductive natural language inference with narrative contexts. Thus, they depend on additional paragraphs provided by the question, which enrich the level of information provided to the model. Furthermore, abductive reasoning resources obtained from procedural texts are either artificially made difficult with targeted models (such as SWAG (Zellers et al., 2018) and HellaSWAG (Zellers et al., 2019)) or covers another form of procedural texts (e.g., natural phenomenons of WIQA (Tandon et al., 2019)). Moreover, they focus on the continuous elements (i.e. consecutive steps or events), which we believe diminishes the degree of implicity.

Furthermore our main resource wikiHow has been extensively used for a wide range of tasks thanks to its rich body of well-structured procedural documents, including but not limited to summarization (Koupaee and Wang, 2018; Ladhak et al., 2020), intent detection (Zhang et al., 2020a), reasoning (Zhang et al., 2020b), linking actions (Lin et al., 2020; Zhou et al., 2022), and next event prediction (Nguyen et al., 2017; Zellers et al., 2018, 2019).

6 Conclusion

There has been a growing interest in procedural data, tasks, and reasoning. However, the spotlight has been on explicit and direct relations when studying reasoning within procedural documents. To address the the lack of resources to study implicit relations and reasoning, we introduce PARADISE and strong baseline models evaluated and analyzed with extensive experiments. PARADISE contains +104K warnings and tips in total and serves as a reliable testbed for the evaluation of abductive and implicit commonsense reasoning skills of language models. Moreover, it brings improvement to zero-and-few-shot performances in out-of-domain procedural tasks. Our experiments reveal that PLMs do not possess inherent reasoning skills; yet, most of them outperform LLMs when fine-tuned. However, best-performing models from both groups fall behind human performance, indicating room for improvement. We release all the resources publicly to further research.

Limitations

We evaluate LLMs with their respective APIs due to their proprietary nature or heavy computation costs. Therefore, although we explain our evaluation setup in detail, the performances of LLMs we evaluate might not always be reproducible due to potential future changes or deprecations of their APIs.

Ethics Statement

We utilize the content from wikiHow, adhering to the specific circumstances outlined in the Creative Commons license. We fully comply with all conditions stipulated by the Creative Commons license, and these requirements facilitate the utilization of the wikiHow corpus upon which we build.

Acknowledgements

This work has been supported by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) as part of the project "Automatic Learning of Procedural Language from Natural Language Instructions for Intelligent Assistance" with the number 121C132. We also gratefully acknowledge KUIS AI Lab for providing computational support. We thank our anonymous reviewers and the members of GGLab who helped us improve this paper. We especially thank Aysha Gurbanova, Şebnem Demirtaş, and Mahmut İbrahim Deniz for their contributions to evaluating human performance on warning and tip inference tasks.

References

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi

Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. 2023. Palm 2 technical report.

Stéphane Aroca-Ouellette, Cory Paik, Alessandro Roncone, and Katharina Kann. 2021. PROST: Physical reasoning about objects through space and time. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4597–4608. Association for Computational Linguistics.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Prajjwal Bhargava and Vincent Ng. 2022. Commonsense knowledge reasoning and generation with pre-trained language models: A survey. In *AAAI Conference on Artificial Intelligence*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Tony F. Chan, Gene H. Golub, and Randall J. LeVeque. 1979. Updating formulae and a pairwise algorithm for computing sample variances. Technical report, Stanford, CA, USA.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wentau Yih, and Peter Clark. 2019. Everything happens for a reason: Discovering the purpose of actions in procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4496–4505. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910. Association for Computational Linguistics.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Luke Benson, Lucy Sun, Ekaterina Zubova, Yujie Qiao, Matthew Burtell, et al. 2022. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *CoRR*, abs/2006.03654.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401. Association for Computational Linguistics.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9118–9147. PMLR.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Alexander Jansen, and Ashish Sabharwal. 2019. Qasc: A dataset for question answering via sentence composition. *ArXiv*, abs/1910.11473.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *ArXiv*, abs/1810.09305.

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4034–4048. Association for Computational Linguistics.

Angela Lin, Sudha Rao, Asli Celikyilmaz, Elnaz Nouri, Chris Brockett, Debadeepta Dey, and Bill Dolan. 2020. A recipe for creating multimodal aligned datasets for sequential tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4871–4884. Association for Computational Linguistics.

Bill Yuchen Lin, Ziyi Wu, Yichi Yang, Dong-Ho Lee, and Xiang Ren. 2021. RiddleSense: Reasoning about riddle questions featuring linguistic creativity and commonsense knowledge. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1504–1515. Association for Computational Linguistics.

Emmy Liu, Chenxuan Cui, Kenneth Zheng, and Graham Neubig. 2022. Testing the ability of language models to interpret figurative language. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4437–4452. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

George A. Miller. 1994. WordNet: A lexical database for English. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Dai Quoc Nguyen, Dat Quoc Nguyen, Cuong Xuan Chu, Stefan Thater, and Manfred Pinkal. 2017. Sequence to sequence learning for event prediction. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 37–42. Asian Federation of Natural Language Processing.

OpenAI. 2023a. Api reference - openai api.

OpenAI. 2023b. Gpt-4 technical report.

Vishal Pallagani, Bharath Muppasani, Keerthiram Murugesan, Francesca Rossi, Biplav Srivastava, Lior Horesh, Francesco Fabiano, and Andrea Loreggia. 2023. Understanding the capabilities of large language models for automated planning. *CoRR*, abs/2305.16151.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. 2018. Virtualhome: Simulating household activities via programs. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8494–8502. Computer Vision Foundation / IEEE Computer Society.

Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqui. 2021. TIMEDIAL: Temporal commonsense reasoning in dialog. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7066–7076. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Gözde Gül Şahin, Clara Vania, Ilia Kuznetsov, and Iryna Gurevych. 2020. LINSPECTOR: Multilingual probing tasks for word representations. *Computational Linguistics*, 46(2):335–385.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10737–10746. Computer Vision Foundation / IEEE.

Chan Hee Song, Brian M. Sadler, Jiaman Wu, Wei-Lun Chao, Clayton Washington, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 2986–2997. IEEE.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293. Association for Computational Linguistics.

Gugger Sylvain, Debut Lysandre, Wolf Thomas, Schmid Philipp, Mueller Zachary, and Mangrulkar Sourab. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. `https://github.com/huggingface/accelerate`.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6076–6085. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Karthik Valmeekam, Alberto Olmo Hernandez, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (A benchmark for llms on planning and reasoning about change). *CoRR*, abs/2206.10498.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. 2023. On the planning abilities of large language models - A critical investigation. *CoRR*, abs/2305.15771.

Yuqing Wang and Yun Zhao. 2023. TRAM: benchmarking temporal reasoning for large language models. *CoRR*, abs/2310.00835.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Perric Cistac, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. pages 38–45. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020a. Intent detection with WikiHow. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 328–333. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020b. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639. Association for Computational Linguistics.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2023. Large language models as commonsense knowledge for large-scale task planning. In *RSS 2023 Workshop on Learning for Task and Motion Planning*.

Shuyan Zhou, Li Zhang, Yue Yang, Qing Lyu, Pengcheng Yin, Chris Callison-Burch, and Graham Neubig. 2022. Show me more details: Discovering hierarchies of procedures from semi-structured web data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2998–3012. Association for Computational Linguistics.

A Example JSON File

An example JSON file from our corpus can be seen in Listing 1.

Listing 1: Example JSON file for the procedural tutorial with the goal "Rip Your Own Jeans" from wikiHow.


```
{
    "title": "How to Rip Your Own Jeans",
    "url": "https://www.wikihow.com/Rip-Your-Own-Jeans",
    "title_description": "Distressed denim is a popular style,
          but buying jeans that are already ripped can be
         expensive. Luckily, you can create this trend
         yourself by roughing up the fabric with a piece of
         sandpaper, then snipping a hole with a pair of
         scissors.",
    "category_hierarchy": [
        "Hobbies and Crafts",
        "Crafts",
        "Decoration Projects"
    ],
    "author": {
        "name": null,
        "n_coauthors": 128,
        "blurb": null
    },
    "time_updated": "April 12, 2020",
    "n_views": 3443113,
    "rating": {
        "n_votes": 76,
        "helpful_percent": 77
    },
    "methods": [],
    "parts": [],
    "video": "https://www.wikihow.com/Video/Rip-Your-Own-Jeans
        ",
    "related_articles": ["ABRIDGED"],
    "tips": [
        "Washing the jeans right after ripping them causes the
             fibers to loosen more and create a more
            distressed look.",
        "Avoid adding rips too near the seams, as they may
            cause them to begin to unravel.",
        "For exact rips, use a sewing needle to pull out
            individual stitches from the fabric."
    ],
    "warnings": [
        "Don't make the rip too big at first. Washing the
            fabric will increase the size and fray of the
            hole.",
        "Never attempt to rip or fray your jeans while you're
            wearing them.",
        "Use caution with sharp tools."
    ],
    "QAs": ["ABRIDGED"],
    "refs": [
        "https://www.marieclaire.co.uk/news/fashion-news/how-
            to-rip-jeans-821",
        "https://www.marieclaire.co.uk/news/fashion-news/how-
            to-rip-jeans-821",
        "https://www.cosmopolitan.com/style-beauty/fashion/a58
            592/5-easy-tricks-for-distressing-your-jeans/"
    ],
    "quizzes": [],
    "other_languages": {"ABRIDGED"},
    "steps": ["ABRIDGED"]
}
```


B Filtering Examples

After sampling the negative candidates, we apply a set of filters introduced by Zhang et al. (2020b) to ensure the high-quality of the pairs and the challenge they bring. However we change some of these filters as follows:

Similarity filter: We obtain the cosine similarity scores using Supervised SimCSE-RoBERTa-Large (Gao et al., 2021), since sentence embeddings capture sentence-level information, resulting in better filtering.

Length filter: We also set an upper bound to ensure they are long (> 8 tokens) enough to contain relevant information yet short (< 128 tokens) enough to be on-point and coherent.

Category filter: We exclude examples from some categories (e.g., Philosophy and Religion, Celebrities, Holidays and Traditions, etc.) that might not be suitable for evaluating language models' reasoning skills due to their subjectivity.

C Dataset Analysis with Cartography

We utilize the dataset cartography tool (Swayamdipta et al., 2020), which processes a model's behaviour on training instances (also known as the training dynamics) for mapping the dataset, to better understand the characteristics of our datasets. To this end, cartography derives three metrics from the training dynamics: confidence (the mean model probability of the true label across epochs), correctness (the fraction of times the model correctly labels an observation across epochs), and variability (the spread of the model's probability of correctly labeling observations across epochs). With these metrics, cartography reveals three regions in a data map, each with distinct features.

Using the cartography tool, we generate data maps for warning and tip inference datasets. As seen in Fig. 6, both of our datasets have a high density in the positive ends of confidence and correctness and in the negative end of variability, indicating that BERT is capable of confidently choosing the correct warnings and tips throughout the training. However, tip inference dataset shows greater density in those ends, demonstrating higher easiness that reinforces our reasoning in Appendix H.

Additionally, we implement the noise detector (Swayamdipta et al., 2020) using a Gaussian Naive Bayes classifier model (Chan et al., 1979) to identify mislabeled instances in our datasets. We train our classifier model on a small set of equally distributed mislabeled (randomly re-assinged) and correctly labeled data instances. Although simple, it achieves 95.2% accuracy on the test set. We, then, use the

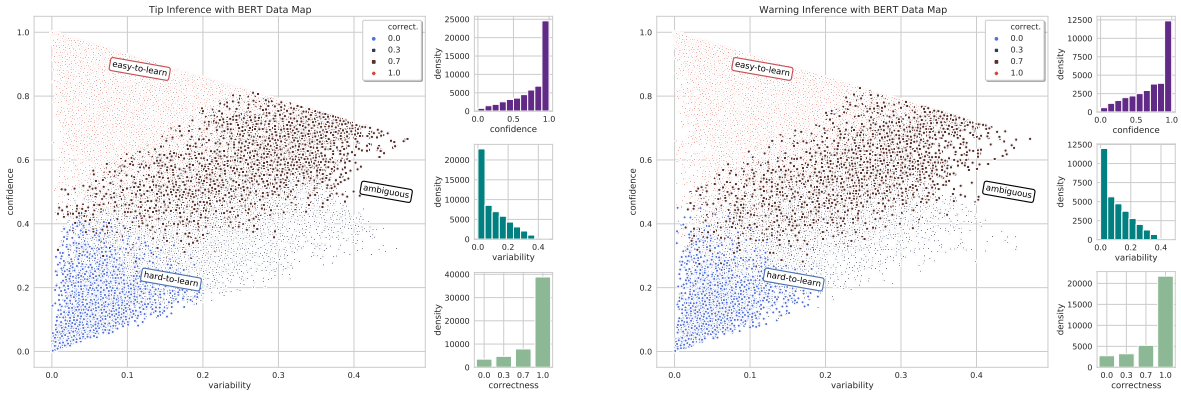

Figure 6: Data maps for warning and tip inference tasks obtained with BERT.

classifier model on the entire warning and tip inference datasets. Our classifier model finds 1022 noisy instances in the tip inference and 746 noisy instances in the warning inference datasets, indicating a 1.3% and 1.9% of noise respectively. Such levels of noise in automatically generated datasets with no human supervision (other than curating the test splits) illustrates the success of the filtering described in Appendix B.

D Implementation Details

We use the base versions of DistilBERT[8], BERT[9], RoBERTa[10], and DeBERTa[11] and implement them using Hugging Face libraries, namely transformers (Wolf et al., 2020), evaluate, and accelerate (Sylvain et al., 2022).

We fine-tune each model for 1 epoch with its unique set of hyperparameters that can be seen in Table 7. We use a batch size of 8 and keep the default values for all other hyperparameters for testing. In fine-tuning, we use five different seeds (42, 2717, 6802, 9893, and 7818) to conduct statistical significance analysis. In testing, we set the seed to 42.

For both fine-tuning and testing, we utilize half-precision floating point format (FP16) with the accelerate library. DistilBERT, BERT, and RoBERTa models are fine-tuned on a single NVIDIA T4, while DeBERTa is fine-tuned on two NVIDIA T4s. Computational costs of fine-tuning each model across each task can be seen in Table 8.

[8] https://huggingface.co/distilbert-base-uncased

[9] https://huggingface.co/bert-base-uncased

[10] https://huggingface.co/roberta-base

[11] https://huggingface.co/microsoft/deberta-v3-base

Hyperparameter	DistilBERT	BERT	RoBERTa	DeBERTa
Total Batch Size	32	64	64	128
Gradient Acc. Steps	1	2	2	4
Learning Rate	2e-5	2e-5	1e-5	1e-5
Max. Sequence Length	128	128	128	128

Table 7: Hyperparameters used in the fine-tunings of models.

Model	GPU	Time
WARNING INFERENCE		
DistilBERT	1 x NVIDIA T4	6 mins 10 secs
BERT	1 x NVIDIA T4	11 mins 53 secs
RoBERTa	1 x NVIDIA T4	11 mins 45 secs
DeBERTa	2 x NVIDIA T4	9 mins 32 secs
TIP INFERENCE		
DistilBERT	1 x NVIDIA T4	12 mins 47 secs
BERT	1 x NVIDIA T4	24 mins 57 secs
RoBERTa	1 x NVIDIA T4	24 mins 51 secs
DeBERTa	2 x NVIDIA T4	19 mins 55 secs

Table 8: Computational costs of fine-tuning each model across each task.

E Prompting LLMs

In order to effectively engage with a language model, it is essential to meticulously construct a prompt template and finetune specific parameters, notably temperature (sampling temperature between 0 and 1 or 0 and 2) and top p (nucleus sampling, where the model considers the results of the tokens with top p probability mass) (OpenAI, 2023a).

For the prompt template construction, we referred to the official API documentation of each model to ascertain the recommended templates. In cases

where no specific templates were provided, we directly used the questions as prompts, omitting any additional tokens.

Regarding the calibration of the temperature and top p settings, we initiated our tests with the default values as specified in the model's API and playground interface. This approach was followed by a systematic tuning process to optimize performance. Notably, we observed that a lower temperature setting, as compared to the default, yielded more accurate results. This aligns with the general understanding that lower temperatures are preferable for fact-based prompts, while higher temperatures are better suited for tasks requiring creativity and an element of randomness.

The accompanying tables 9 and 10 show the specific prompts and parameter settings employed for each model (all the experiments were done in December 2023).

F Additional Keyword Manipulation Methods

In addition to dropping, we manipulate the common keywords in the candidates using the following methods, as exemplified in Table 11:

1. Synonym Replacement: We replace such keywords with their synonyms using WordNet (Miller, 1994) with NLTK (Bird et al., 2009).

2. Replacing with a Placeholder: We replace such keywords with a placeholder word, which is simply PLACEHOLDER.

3. Replacing with the BERT Prediction: We mask such keywords out with the [MASK] token and use BERT to predict the most likely token other than the original one.

We evaluate fine-tuned PLMs on these manipulated examples. As seen in Fig. 7, synonym replacement causes the least decrease in the performance with an approximate average of 10% drop in accuracy across models and tasks. Dropping, placeholder replacement, and BERT prediction replacement closely follow each other respectively, with average decreases in accuracy varying from 15% to 20%.

G DeBERTa and GPT-4 Failures

Specific examples that DeBERTa and GPT-4 fail can be seen in Table 12.

H A Comparison Between Warnings and Tips

As discussed in Sec. 4.5.1, warnings and tips share the common purpose of presenting additional information to user for better instruction execution within procedural wikiHow tutorials. Thus, they generally have high similarity in semantics and structure.

Overall, warning inference poses a greater challenge compared to tip inference. We believe this is due to two main reasons. First, tips are more goal-specific, while warnings are more general, yet still distinguishable. For example, any goal that contains sharp objects might have a warning towards using those sharp objects carefully; yet, the same does not hold true for tips. Second, tips are ample in quantity; therefore, there are more examples to learn from.

Our reasoning behind tips being more goal-specific and informative regarding the procedural tutorial compared to warnings is reinforced by the following findings:

1. Both PLMs and LLMs perform better in tip inference compared to warning inference, indicating a greater capability in reasoning with tips due to tips being more directly connected to their goals.

2. Previous fine-tuning on tip inference better improves the performance in transfer learning compared to previous fine-tuning on warning inference, demonstrating that tip inference contributes to model's learning over procedural tasks more.

3. Tip inference is affected more by keyword manipulation, showing that tip inference data is more vulnerable to keyword alteration because its positive candidates contain more goal-specific vocabulary.

Model	Settings	Prompt
GPT-4 **PALM2**	**Temp. = 0.3** **Top p = 0.9** **Temp. = 0.3** **Top p = 0.9**	I will give you a goal below and 4 tips. Can you pick the most related tip to it? **Goal**: Prepare for a Long Car Trip **Tips**: **Tip 0-** While performing any exercise, make sure you are drinking water to stay hydrated. **Tip 1-** Do drink plenty of water to keep your skin hydrated. **Tip 2-** If you are traveling for a long time, bring a bottle of water to keep you hydrated. **Tip 3-** Bring a bottle of water with you to stay hydrated. **Response format**: Return the tip number in json with 'tip' as key and no more details.
Llama2 **Mistral**	**Temp. = 0.3** **Top p = 0.9** **Temp. = 0.0** **Top p = 0.1**	<s>[INST] I will give you a goal below and 4 tips. Can you pick the most related tip to it? **Goal**: Prepare for a Long Car Trip **Tips**: **Tip 0-** While performing any exercise, make sure you are drinking water to stay hydrated. **Tip 1-** Do drink plenty of water to keep your skin hydrated. **Tip 2-** If you are traveling for a long time, bring a bottle of water to keep you hydrated. **Tip 3-** Bring a bottle of water with you to stay hydrated. **Response format**: Return the tip number in json with 'tip' as key and no more details. [\/INST]
Vicuna	**Temp. = 0.3** **Top-p = 0.9**	A chat between a human and an assistant. ### Human: **Goal**: Prepare for a Long Car Trip **Tips**: **Tip 0-** While performing any exercise, make sure you are drinking water to stay hydrated. **Tip 1-** Do drink plenty of water to keep your skin hydrated. **Tip 2-** If you are traveling for a long time, bring a bottle of water to keep you hydrated. **Tip 3-** Bring a bottle of water with you to stay hydrated. **Response format**: Return the tip number in json with 'tip' as key and no more details. ### Assistance:

Table 9: LLM settings and prompts for tip inference.

Model	Settings	Prompt
GPT-4 **PALM2**	**Temp. = 0.3** **Top p = 0.9** **Temp. = 0.3** **Top p = 0.9**	I will give you a goal below and 4 warnings. Can you pick the most related warning to it? **Goal**: Make Laundry Detergent Slime **Warnings**: **Warn 0-** Warning 0- Avoid flooding the floor with cleaner or water. A thin layer of water should be enough for a dry cloth to wipe **Warn 1-** Don't apply heat (dryer, iron) to the stained area until the stain is gone. **Warn 2-** Don't place the slime in a cold area when it's finished. It may become less stretchy. **Warn 3-** Don't let the resurfacer dry on your skin since it may cause irritation and is difficult to remove. **Response format**: Return the warning number in json with 'warn' as key and no more details.
Llama2 **Mistral**	**Temp. = 0.3** **Top p = 0.9** **Temp. = 0.0** **Top p = 0.1**	<s>[INST] I will give you a goal below and 4 warnings. Can you pick the most related warning to it? **Goal**: Make Laundry Detergent Slime **Warnings**: **Warn 0-** Warning 0- Avoid flooding the floor with cleaner or water. A thin layer of water should be enough for a dry cloth to wipe **Warn 1-** Don't apply heat (dryer, iron) to the stained area until the stain is gone. **Warn 2-** Don't place the slime in a cold area when it's finished. It may become less stretchy. **Warn 3-** Don't let the resurfacer dry on your skin since it may cause irritation and is difficult to remove. **Response format**: Format the answer in json with the warning number as value and 'warn' as key and no more details. [\/INST]
Vicuna	**Temp. = 0.3** **Top-p = 0.9**	A chat between a human and an assistant. ### Human: I will give you a goal below and 4 warnings. Can you pick the most related warning to it? **Goal**: Make Laundry Detergent Slime **Warnings**: **Warn 0-** Warning 0- Avoid flooding the floor with cleaner or water. A thin layer of water should be enough for a dry cloth to wipe **Warn 1-** Don't apply heat (dryer, iron) to the stained area until the stain is gone. **Warn 2-** Don't place the slime in a cold area when it's finished. It may become less stretchy. **Warn 3-** Don't let the resurfacer dry on your skin since it may cause irritation and is difficult to remove. **Response format**: Return the warning number in json with 'warn' as key and no more details.

Table 10: LLM settings and prompts for warning inference.

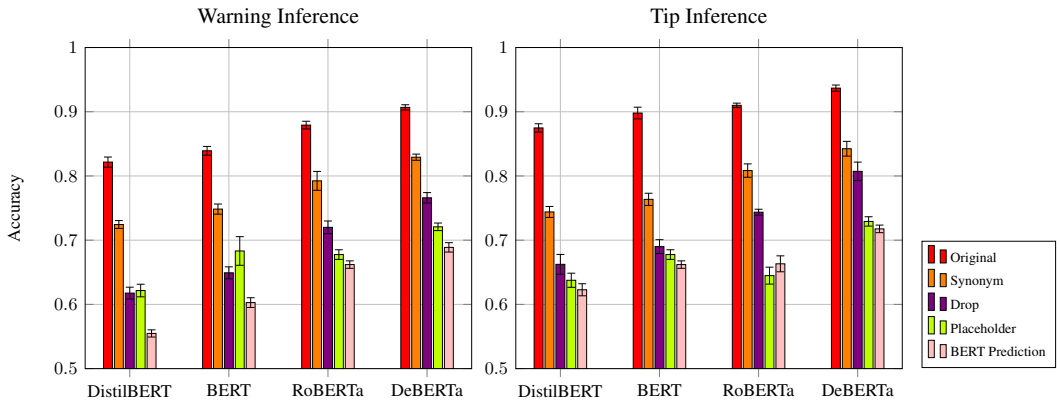

Figure 7: Model performances tested on manipulated test data.

Manipulation	**Positive Candidate (abridged)**	**Negative Candidate (abridged)**
GOAL: Stop Eating Fast Food COMMON KEYWORDS: Fast, Food, Eating		
Original	Beating a fast food addiction is a lot easier when you're not going it alone. Talk to friends about your goals for your diet.	Being vulnerable about your eating disorder is a hard thing to do. Remember that you need other people to support you.
Synonym	Beating a flying food addiction is a lot easier when you're not going it alone. Talk to friends about your goals for your diet.	Being vulnerable about your eat disorder is a hard thing to do. Remember that you need other people to support you.
Dropping	Beating a addiction is a lot easier when you're not going it alone. Talk to friends about your goals for your diet.	Being vulnerable about your disorder is a hard thing to do. Remember that you need other people to support you.
Placeholder	Beating a placeholder placeholder addiction is a lot easier when you're not going it alone. Talk to friends about your goals for your diet.	Being vulnerable about your placeholder disorder is a hard thing to do. Remember that you need other people to support you.
BERT Prediction	Beating a new drug addiction is a lot easier when you're not going it alone. Talk to friends about your goals for your diet.	Being vulnerable about your anxiety disorder is a hard thing to do. Remember that you need other people to support you.

Table 11: Examples of keyword manipulation for a pair from the tip inference dataset. The goal is "Stop Eating Fast Food", which shares the common keywords of "Fast" and "Food" with the positive candidate and "Eating" with one of the negative candidates.

DeBERTa	**GPT-4**
Install Ceramic Wall Tile	Block People on Facebook
Build a Nerf Fort	See Your WiFi Password on an iPhone
Turn a Cardboard Box Into a Basket	Change the Password in Outlook 365
Prevent Water Stains on Bathroom Walls	Deal with Catty Coworkers
Make a Rope Braid	Be Confident As a Short Person
Remove a Red Wine Stain Ring from a Wood Table	Help an Autistic Family Member

Table 12: Goals of examples that DeBERTa and GPT-4 fail.