Attempts to Improve Performance of LLMs on the TRIP Dataset

Ishita Deshmukh and Carlos Figueredo and Hanning Li and Fei Wu
Computer Science and Engineering Division, University of Michigan
{ideshmuk, carlosfc, lhanning, feiyuwu}@umich.edu

Abstract

This report outlines four distinct approaches to improve the performance of Large Language Models (LLMs) on the Tiered Reasoning for Intuitive Physics (TRIP) dataset. The TRIP dataset is specifically designed by the SLED lab in order to evaluate commonsense reasoning in physical scenarios through a multi-tiered framework. The proposed methods include manipulation among ALBERT, GPT-2 XL, and ELECTRA models, with each offering different perspectives from variant architecture advantages. One newer approach, integrating Answer Set Programming (ASP), is also discussed in this report. ASP was integrated into the prompting process to enhance logical reasoning capabilities. As for our project, we aim to improve upon the shortcomings identified by the SLED research lab. Existing models oftentimes rely on superficial patterns instead of genuine reasoning skills. We are targeting to evaluate our approaches among key reasoning tasks — conflict detection, consistency assessment, and verifiability. A detailed comparison is given in this report. We believe that our work contributes to improving LLM's performance on complex reasoning benchmarks and providing new insights for commonsense reasoning in AI systems.

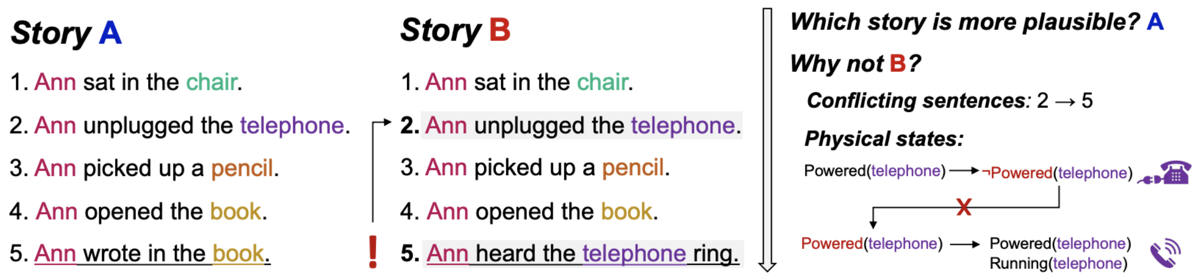
1 Introduction

Current evaluations of LLMs have shown increasing performance on commonsense reasoning benchmarks. However, some concerns have been raised that these results may stem from dataset biases instead of genuine reasoning abilities. In an effort to test underlying reasoning capabilities, the Tiered Reasoning for Intuitive Physics (TRIP) dataset has been developed by the SLED research group (Storks et al., 2021). This novel benchmark challenges LLMs to demonstrate coherent reasoning about physical scenarios through a multi-tiered evaluation of the reasoning process. This data set consists of pairs of highly similar stories, differing only by one sentence which makes one of the stories implausible. The high-level end task is story plausibility classification, which consists of determining which story is more feasible. The data set enables a multi-tiered evaluation where tested systems must jointly identify story plausibility, detect conflicting sentences, and classify underlying physical states causing the conflict. It is also possible to re-process the data set to test these tasks independently. The stories in the data set involve concrete physical actions in household settings and avoid subjective understandings and complex language. Additionally, implausible stories are designed to only become implausible when considering the full text (e.g., avoiding sentences like "John ate the spoon.").

For clarity, an example story from the TRIP dataset is illustrated in Figure 1, which is reproduced from its original appearance in the SLED group's original paper. In Figure 1, Story A describes Ann who sits in a chair, then unplugs the telephone, then picks up a pencil, opens a book, and writes in the book. This story is physically plausible, because none of the actions in the story are physically impossible. Story B describes Ann who sits in a chair, then unplugs the telephone, then picks up a pencil and opens a book, and then hears the telephone ringing. This story is not physically plausible because the telephone could not ring if it was unplugged. A model would need to identify that Story A is more plausible, identify that sentences 2 and 5 from Story 5 are contradictory, and indicate that the conflict is related to the state of the telephone object.

Existing models are able to achieve a reasonable accuracy when asked to determine which story is more plausible. However, these models perform very poorly when asked to detect the conflicting sentences and to classify the underlying physical states that cause the conflict. This result suggests that the models are not performing any actual reasoning. As we aim to build more robust LLMs, uncovering reasoning processes is a key challenge. By focusing on interpretability, we can move beyond mere accuracy metrics and develop LLMs that not only perform well but also demonstrate coherent and verifiable reasoning processes. This approach is essential for advancing the field of AI towards more robust, trustworthy, and truly intelligent systems capable of human-like reasoning about complex scenarios.

In this paper, we discuss four approaches that we attempted in order to improve the performance



Story A
1. Ann sat in the chair.
2. Ann unplugged the telephone.
3. Ann picked up a pencil.
4. Ann opened the book.
5. Ann wrote in the book.
Story B
1. Ann sat in the chair.
2. Ann unplugged the telephone.
3. Ann picked up a pencil.
4. Ann opened the book.
5. Ann heard the telephone ring.
Which story is more plausible? A
Why not B?
Conflicting sentences: 2 → 5
Physical states:
Powered(telephone) → ¬Powered(telephone)
Powered(telephone) → Powered(telephone) Running(telephone)

Figure 1: Example stories from the TRIP dataset. The diagram is from the SLED Group's research paper (Storks et al., 2021).

of large language models on these reasoning tasks. Our first three models build on the work made by the SLED research group, trying similar strategies with different models. Our fourth approach takes a new angle by using TRIP to measure the performance of a generative language model in the first and second tasks of the data set. Specifically, we use Answer Set Programming code to enhance the reasoning capabilities of Mistral-7B LLM. Our work verifies the results obtained in the original paper, and our fourth approach shows promising results to enhance LLMs' reasoning ability through ASP prompting. The report is divided as follows: section 2 shows related work to our approaches and an Answer Set Programming overview, section 3 describes in detail our experiments and approaches, section 4 shows immediate comparisons between models, section 5 delves into the discussion of our results, section 6 describes what could be next steps if given more time, and section 7 concludes the major results of our work.

2 Related Work

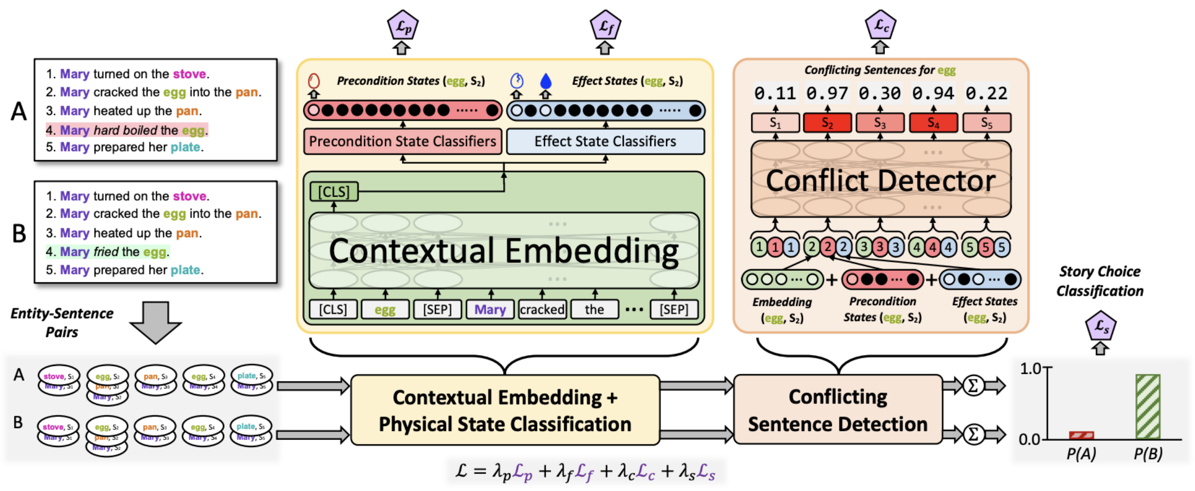
The TRIP dataset was developed by the SLED research group. The SLED research group trained various BERT-like models (BERT, RoBERTa, and DeBERTa) on the TRIP dataset and evaluated their performance on identifying the implausible story in a pair and providing supporting reasoning by identifying the conflicting statements and object states in the story. The models consisted of four modules: (1) a contextual embedding model to learn the entities in the story; (2) precondition and effect classifiers to identify the physical states of each entity as the story progresses; (3) a conflict detector to identify which sentences in the story conflict; (4) a story choice prediction module to identify which of two stories is physically possible. Each module had its own loss function, and the overall loss function was a weighted average of the individual loss functions. This overall loss function was then minimized through gradient descent. Figure 2, which is reproduced from its original appearance in the SLED Group's research paper, illustrates the model architecture.

The models had worse performance on the task of identifying the implausible story with the architecture described in the paper compared to the standard model, which suggests that the standard models were not actually learning any commonsense reasoning skills. In our research, we aim to explore if other models can be trained to learn commonsense reasoning skills.

2.1 Answer Set Programming Overview

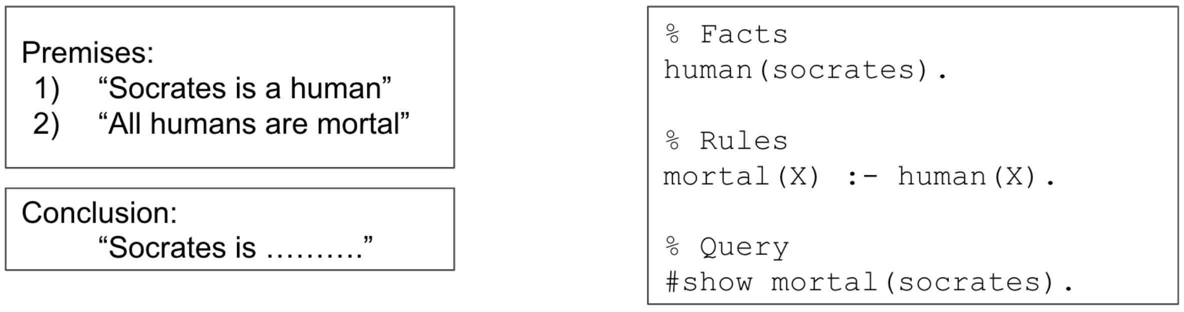
ASP is a type of logic programming that helps solve complex problems by setting up constraints and desired outcomes. We expect this language will allow us to model logically the relationships between the sentences in the stories. Specifically, ASP works by creating "answer sets" or "stable models," which are groups of facts that satisfy all the rules in a given program. The existence of a stable model will indicate that the story is plausible. Conversely, if no stable model can be found, it suggests that the story contains a logical inconsistency. Clingo is a logic programming language and solver that can be used to write and execute ASP code. Figure 3 provides an example that illustrates the concepts of ASP that essentially allow automatic resolution of declarative logical problems. We will use Clingo syntax to set a framework that the LLM can leverage to solve reasoning problems methodically.

Our initial challenge is to translate natural language to ASP code. An architecture that performs translation of text to ASP code was developed in the paper "Towards Automatic Composition of ASP Programs from Natural Language Specifications"



A
1. Mary turned on the stove.
2. Mary cracked the egg into the pan.
3. Mary heated up the pan.
4. Mary hard boiled the egg.
5. Mary prepared her plate.
B
1. Mary turned on the stove.
2. Mary cracked the egg into the pan.
3. Mary heated up the pan.
4. Mary fried the egg.
5. Mary prepared her plate.
Entity-Sentence Pairs
Precondition States (egg, S₂)
Effect States (egg, S₂)
Precondition State Classifiers
Effect State Classifiers
[CLS]
Contextual Embedding
[CLS] egg [SEP] Mary cracked the ... [SEP]
Conflicting Sentences for egg
0.11 0.97 0.30 0.94 0.22
S₁ S₂ S₃ S₄ S₅
Conflict Detector
Embedding (egg, S₂)
Precondition States (egg, S₂)
Effect States (egg, S₂)
Story Choice Classification
Contextual Embedding + Physical State Classification
Conflicting Sentence Detection
1.0
0.0
P(A)
P(B)
$\mathcal{L} = \lambda_p\mathcal{L}_p + \lambda_f\mathcal{L}_f + \lambda_c\mathcal{L}_c + \lambda_s\mathcal{L}_s$

Figure 2: Model architecture from the SLED Group's research paper (Storks et al., 2021).


Premises:
1) "Socrates is a human"
2) "All humans are mortal"

Conclusion:
"Socrates is ...."

```
% Facts
human(socrates).

% Rules
mortal(X) :- human(X).

% Query
#show mortal(socrates).
```

Figure 3: A diagram illustrating the concepts behind answer set programing. Given the fact that Socrates is a human and the rule that all humans are mortal, the code can conclude that Socrates is mortal.

(Borroto et al., 2024). The architecture consists of two stages: (1) a tool called NL2ASP converts natural language text into Controlled Natural Language (CNL) statements; (2) a tool called CNL2ASP converts CNL statements into ASP code. NL2ASP uses the BART and T5 transformer models to perform the translation from natural language to CNL statements. CNL2ASP simply applies the rules of ASP to the CNL statements to convert them into ASP code. CNL2ASP was developed in the paper "CNL2ASP: converting controlled natural language sentences into ASP" (Caruso et al., 2024). Although previous work has been applied to translation from natural language to ASP code, our approach takes one additional step by modeling instead of just translating. This is a significant difference because, instead of translating each sentence independently, our aim is to solve jointly the plausibility problem. Moreover, we had to change our approach in order to assess our research question correctly. In the following section, we will delve deeper into the rationale behind abandoning the translation approach that initially appeared promising. We will explore the intuition that led us to adopt a prompting strategy and discuss how this shift in perspective enables more effective reasoning about physical scenarios in the context of the TRIP dataset.

3 Approaches and Methods

We developed four methods for potentially improving the reasoning performance of LLMs on the TRIP dataset.

3.1 Method #1: ALBERT Model

In this method, we used the same methodology as the original paper (Storks et al., 2021), but with a different BERT-like model: ALBERT. We evaluated ALBERT's performance on the TRIP dataset to determine if it would perform better on the three tasks of the TRIP dataset.

We hypothesized that ALBERT would perform well on the dataset because it was designed to handle inputs consisting of multiple sentences well.

ALBERT was trained using sentence-order prediction loss, which resulted in improved coherence across multiple sentences (Lan, 2019). Understanding multiple sentences and the relationship between them is critical for the physical commonsense reasoning tasks present in the TRIP dataset, leading us to conclude that using ALBERT would lead to improved performance when compared to the BERT-like models that were used in the original paper.

3.2 Method #2: GPT-2 XL Model

In this method, we used another commonly used model GPT-2 XL (Radford et al., 2019), which was also used in our homework assignments. We picked this model because GPT models are getting more and more attention from the audience, and we would like to evaluate an early stage model like GPT-2 XL. Also, given that our goal is to evaluate based on text, and that GPT models were created for text generation purposes, we decided that GPT models would be a good choice for our goal.

We initially expected that GPT-2 XL's strengths in capturing contextual relationships and generating coherent text (Radford et al., 2019) would allow it to perform adequately on the three tasks of the TRIP dataset: story consistency, verifiability, and reasoning accuracy. However, given its architecture and pretraining objectives, we also anticipated challenges in structured reasoning, particularly in tasks requiring fine-grained token-level or logical inferences. We will talk about our findings in later sections.

To evaluate GPT-2 XL, we fine-tuned it on the TRIP dataset using the same loss functions as the original paper, including story choice loss, conflict detection loss, and state classification losses, while ensuring hyperparameter tuning to optimize its performance. We analyzed its outputs on each task and compared its results to other methods to assess its ability to reason effectively in structured contexts.

3.3 Method #3: ELECTRA Model

ELECTRA is a transformer-based model for natural language processing tasks. It is more efficient than BERT and achieves similar or better performance on many tasks. The main idea is to replace token detection. Instead of masking tokens like BERT, ELECTRA replaces some tokens in the input with alternatives generated by a smaller model. The main model, called the discriminator, learns to identify which tokens are replaced. This method uses all tokens in the input, making learning more effective.

ELECTRA has two parts: a generator that creates token replacements and a discriminator that detects them. This design provides more detailed feedback than BERT's approach. It also requires less computing power and is faster to train. ELECTRA's architecture is illustrated in Figure 4.

After pretraining, ELECTRA can be fine-tuned for many tasks like text classification or question answering. It comes in different sizes, and even the smaller models perform well, making it a good option for limited resources. Compared to BERT, ELECTRA is faster, smaller, and more efficient, which makes it useful for a variety of applications. (Clark, 2020).

In terms of its application on the TRIP dataset, ELECTRA is built for token-level discrimination instead of generation, which makes it suitable for the specific challenges of this dataset. Its architecture, which includes a generator and a discriminator, enables the model to make use of all tokens during training, improving its overall efficiency. This design is especially helpful for tasks like conflict detection and state classification in the TRIP dataset, where it is crucial to understand detailed token-level interactions.

To evaluate how effective ELECTRA is, we fine-tuned it on the TRIP dataset with different loss configurations. The main purpose of our experiments was to check whether ELECTRA's token-level pretraining could enhance its ability to reason about physical states and detect implausible parts in stories. We examined its performance on the three main tasks in TRIP: assessing story plausibility, identifying conflicts, and tracking state changes.

3.4 Method #4: Answer Set Programming Prompting

Initially, the idea was to use a solver-augmented LLM (Feng et al., 2023), but this approach failed. Solver-augmented LLMs have an initial stage of parsing. However, when we used Mistral-7B (AI, 2023) to parse the TRIP's sentences into ASP code, the language model was omitting implicit rules. Generally, all rules are available in logical reasoning problems, and it suffices to translate the rules into the solver's code to find a solution. Nevertheless, in our problem setting, not all rules are available, and the inference of implicit rules is the key of the TRIP dataset tasks. The main issue with our initial approach is that the LLM was translating

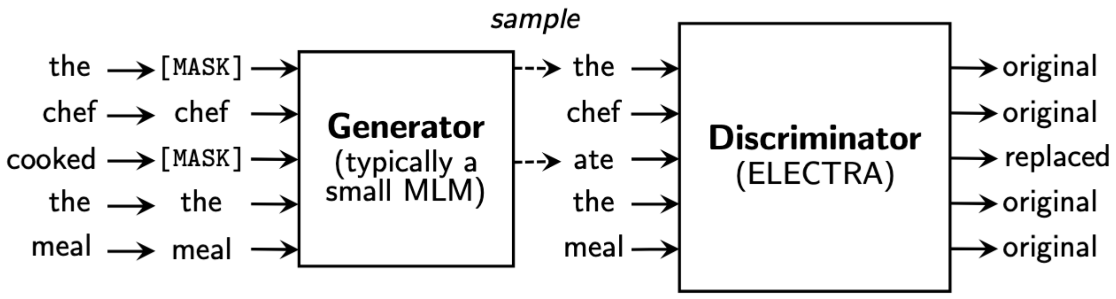


Figure 4: A diagram of ELECTRA's architecture. (Clark, 2020)

correctly sentences like "Tom destroyed the radio" and "Tom switched the radio on" to ASP code, but the implicit rule "If the radio is destroyed, Tom can't switch it on" was never going to appear in the program because it isn't an original sentence of the story. Although the initial approach failed, we remained interested in the use of ASP's rich logical syntax in LLMs, and we developed a prompting strategy to test our ideas.

Our experiment consists in measuring whether incorporating Answer Set Programming in a prompt can increase the performance of LLMs in the TRIP dataset tasks. We begin by evaluating the first task of the TRIP dataset, which can be measured using the accuracy metric. We tested three prompting strategies: Zero-shot, Few-shot, and the proposed ASP Few-shot strategy. Our proposed strategy is a Few-shot prompting that first asks the LLM to model the stories in ASP code to later give a final answer. With this approach, we expect to simulate the process a human would follow to solve a complex logical problem: model, run a solver algorithm, and get results. We expect that the preliminary modeling performed by the LLM will enhance its ability to follow sound reasoning.

In our experiment setting, we drew 100 samples from the TRIP's test split, and we measured Mistral-7B's ability to classify the most plausible story between the two offered. First, we used a zero-shot approach in which we directly asked which story is more plausible. Second, we used a Few-shot approach in which we showed 2 examples before asking for the most plausible story. For the Few-shot examples, we sampled 10 instances of the TRIP's train split, and we used ChatGPT to generate good quality example answers which were validated by our team. For the regular Few-shot approach, the few-shot examples contained an analysis of the plausibility of both stories. Finally, we tested our proposed ASP approach in which the few-shot examples contained three elements: an ASP program, an analysis of the plausibility of both stories and the final selection of story plausibility.

Although the previous experiment mainly aimed at measuring the performance on the 1st task of TRIP, we also aimed at the second task because the text generated by our approach can be processed to extract the consistency metric. This is because the modeling in Answer Set Programming requires the generation of the implicit rules that govern the plausibility selection, and we can use regular expressions and text processing to match these rules with the sentences in the stories. Thus, the second stage of our ASP approach processed the answers generated in the previous section and we were able to detect and extract the conflicting sentences in the implausible story that were contrasted with the ground truth. Additionally, further processing of the generated ASP program allowed us to have a second look at the plausibility selection because the program could differ in the implausible story selected by the LLM at the end of the generated text.

4 Evaluation

Table 1 contains the results for Methods #1-3, along with the results from the SLED Group's paper for comparison.

4.1 Method #1: ALBERT Model

We used Google Colab Pro's A100 GPU to conduct the experiments described in section 3.1. The results achieved by using the ALBERT model were comparable to the results achieved by the BERT-like models used in the SLED Group's paper.

Model	Accuracy (%)	Consistency (%)	Verifiability (%)
random	47.8	11.3	0.0
All Losses			
BERT	78.3	2.8	0.0
RoBERTa	75.2	6.8	0.9
DeBERTa	74.8	2.2	0.0
ALBERT	72.6	5.7	2.0
GPT-2 XL	10.0	3.1	0.1
ELECTRA	79.8	4.6	1.4
Without Story Choice Loss			
BERT	73.9	28.0	9.0
RoBERTa	73.6	22.4	10.6
DeBERTa	75.8	24.8	7.5
ALBERT	72.1	20.8	6.8
GPT-2 XL	8.6	3.8	1.1
ELECTRA	70.1	3.7	19.7
Without Conflict Detection Loss			
BERT	50.9	0.0	0.0
RoBERTa	49.7	0.0	0.0
DeBERTa	52.2	0.0	0.0
ALBERT	46.4	0.0	0.0
GPT-2 XL	10.1	0.0	0.0
ELECTRA	41.0	0.0	0.0
Without State Classification Losses			
BERT	75.2	17.4	0.0
RoBERTa	71.4	2.5	0.0
DeBERTa	72.4	9.6	0.0
ALBERT	74.6	8.8	0.0
GPT-2 XL	6.7	1.8	0.0
ELECTRA	74.6	10.5	0.0

Table 1: Results for Methods # 1-3 along with the SLED Group's Results

Analysis of ALBERT's Performance with Different Losses

1. **All Losses**
   When using all losses, ALBERT achieved an accuracy of 72.6%, which fell within the 74.8% to 78.3% range achieved by the models in the SLED Group's paper. ALBERT achieved a consistency of 5.7%, which fell within the 2.2% to 6.8% range achieved by the models in the SLED Group's paper. ALBERT achieved a verifiability of 2.0%, which was marginally better than the 0.0-0.9% range achieved by the models in the SLED Group's paper.

2. **Without Story Choice Loss**
   When excluding the story choice loss, ALBERT achieved an accuracy of 72.1%, which was lower than the 73.6% to 75.8% range achieved by the models in the SLED Group's paper. ALBERT achieved a consistency of 20.8%, which was lower than the 22.4% to 28.0% range achieved by the models in the SLED Group's paper. ALBERT achieved a verifiability of 6.8%, which was lower than the 7.5-10.6% range achieved by the models in the SLED Group's paper.

3. **Without Conflict Detection Loss**
   When excluding the conflict detection loss, ALBERT achieved an accuracy of 46.4%, which was lower than the 49.7% to 52.2% range achieved by the models in the SLED Group's paper. ALBERT achieved 0.0% consistency and verifiability, which was the same result as achieved by the models in the SLED Group's paper.

4. **Without State Classification Losses**
   When excluding the state classification losses, ALBERT achieved an accuracy of 74.6%, which fell within the 71.4% to 75.2% range achieved by the models in the SLED Group's paper. ALBERT achieved 8.8% consistency, which fell within the 2.5% to 17.4% ranged achieved by the models in the SLED Group's paper. ALBERT achieved 0.0% verifiability, which was the same result as achieved by the models in the SLED Group's paper.

While the ALBERT model achieved a relatively high accuracy, it failed to achieve a high consistency or verifiability with any loss configuration, indicating that it was unable to perform physical commonsense reasoning.

In the SLED Group's paper, the BERT, RoBERTa, and DeBERTa models were used. The BERT model contains 336 million parameters (HuggingFace, 2024a). The RoBERTa model contains 356 million parameters (HuggingFace, 2024b). The DeBERTa model contains 100 million parameters (HuggingFace, 2022). By contrast, ALBERT contains only 12 million parameters (Lan, 2019). While ALBERT was not able to improve upon the performance of the other BERT-like models used previously, it still achieved comparable performance in spite of having roughly 10-30x fewer parameters.

Table 2 provides a more compact representation of the results achieved by the ALBERT model. From this table, we can see that ALBERT achieved the highest accuracy when state classification losses were excluded and achieved the highest consistency and verifiability when the story choice loss was excluded. ALBERT achieved the worst performance when the conflict detection loss was excluded. The models from the SLED Group's paper also achieved the highest consistency and verifiability when the story choice loss was excluded and the worst performance when the conflict detection loss was excluded. These consistent trends suggest that excluding story choice loss improves model performance on the low-level tasks of identifying conflicting sentences and identifying object states, while including conflict detection loss is important for improving performance on the high-level task of identifying which story is more plausible.

Model	Accuracy (%)	Consistency (%)	Verifiability (%)
All Losses	72.6	5.7	2.0
Without Story Choice Loss	72.1	20.8	6.8
Without Conflict Detection Loss	46.4	0.0	0.0
Without State Classification Losses	74.6	8.8	0.0

Table 2: Results achieved by ALBERT model

4.2 Method #2: GPT-2 XL Model

We used Google Colab Pro's A100 GPU to conduct all the experiments described in section 3.2. Similar to using either ALBERT or Electra Model, we used GPT-2 XL Model to evaluate the performance of LLMs on the TRIP dataset. The TRIP dataset is rather challenging, as it evaluates not only the model's accuracy in story comprehension but also its ability to detect conflicts and provide verifiable reasoning.

While we do not have a ground truth value for this comparison, we were expecting the results would be better than what we have gathered in the end. More specifically, we would expect GPT-2 XL perform comparatively similar to other models, but we have got worse results in the end. The following results are what we have observed.

Analysis of GPT-2 XL Performance with Different Losses

1. **All Losses**
   GPT-2 XL achieves an accuracy of 10.0%, a consistency of 3.1%, and a verifiability of 0.1%. This indicates that the model struggles significantly with identifying which story is more plausible, maintaining logical consistency, and producing verifiable outputs.

2. **Without Story Choice Loss**
   When the story choice loss is excluded, the accuracy drops to 8.6%, indicating its importance for maintaining performance. However, consistency improves slightly to 3.8%, and verifiability increases to 1.1%, suggesting that removing this loss makes the outputs marginally more interpretable.

3. **Without Conflict Detection Loss**
   Excluding the conflict detection loss has the most detrimental impact. While the accuracy remains stable at 10.1%, both consistency and verifiability drop to 0%. This demonstrates that conflict detection is crucial for the model to maintain coherent and verifiable results.

4. **Without State Classification Losses**
   When the state classification losses are excluded, the accuracy decreases significantly to 6.7%. Consistency also drops to 1.8%, and verifiability remains at 0%. This shows that the state classification losses play a critical role in enhancing the model's overall performance and coherence.

Table 3 provides a more compact representation of the results achieved by the GPT-2 XL model. From this table, we can see that GPT-2 XL achieved the highest accuracy when the conflict detection loss was excluded, achieved the highest consistency when the state classification losses were excluded, and achieved the highest verifiability when the story choice loss was excluded. The models from the SLED Group's paper also achieved the highest verifiability when the story choice loss was excluded.

These results indicate a lack of robustness in GPT-2 XL's performance across the TRIP dataset metrics.

Model	Accuracy (%)	Consistency (%)	Verifiability (%)
All Losses	10.0	3.1	0.1
Without Story Choice Loss	8.6	3.8	1.1
Without Conflict Detection Loss	10.1	0.0	0.0
Without State Classification Losses	6.7	10.5	0.0

Table 3: Results achieved by GPT-2 XL model

This poor performance is unexpected, especially considering other similar models achieved much higher accuracy compared with this single model.

The unexpected poor performance of GPT-2 XL on the TRIP dataset indicates a few questions to consider. Given that similar models, such as ALBERT and ELECTRA, achieved significantly higher accuracy and consistency on the same dataset, it is possible that the fine-tuning process for GPT-2 XL was not as expected.

Factors resulting in this issue may include:

1. **Critical bugs in the codebase.**
   We may have unseen bugs in the implemention, which may cause the results to be wrong.

2. **Hyperparameter Tuning**
   The learning rate, batch size, or number of training epochs may not have been optimal for this task.

3. **Dataset Preprocessing**
   If the TRIP dataset was tokenized incorrectly for this model, it may have hindered the model's performance.

4. **Other Limitations**
   GPT-2 XL may naturally struggle with this type of logical inference or conflict detection task.

The results obtained for GPT-2 XL on the TRIP dataset were disappointing. The accuracy and consistency metrics were significantly lower than other similar models. This highlights the importance of careful experimental setup to prevent critical bugs. Also, we need to consider dataset-specific factors when fine-tuning large language models.

Some possible improvements would be to fine-tune this model and find possible bugs. If none of the approaches work, we could find another model to work on.

4.3 Method #3: ELECTRA Model

Analysis of ELECTRA'S Performance

The ELECTRA model performs differently depending on the loss configuration, as summarized in Table 4, which provides a more compact representation of the results achieved by the ELECTRA model.

1. **All Losses**
   The ELECTRA model reaches the highest accuracy at 79.8%, but consistency is low at 4.6%, and verifiability is only 1.4%. This shows that the model prioritizes accuracy but struggles with logical consistency and producing verifiable results.

2. **Without Story Choice Loss**
   The accuracy drops to 70.1%, which shows that this loss is important for keeping performance high. At the same time, the verifiability improves a lot to 19.7%, meaning that the model's outputs are easier to interpret. However, consistency stays low at 3.7% with only a slight improvement.

3. **Without Conflict Detection Loss**
   The accuracy drops to 41.0%, and both consistency and verifiability drop to 0%. This shows that conflict detection is critical for the model to function effectively.

4. **Without State Classification Losses**
   The accuracy decreases slightly to 74.6%. However, consistency improves significantly to 10.5%, which is better than with all losses. Verifiability remains at 0%, so state classification losses do not seem to affect this aspect.

Overall, the model achieves the best accuracy with all losses, but consistency and verifiability are weak. Removing the story choice loss improves verifiability but harms accuracy. Conflict detection loss is essential for reasoning, as its removal results in complete failure. Removing state classification losses shows a trade-off, improving consistency but slightly lowering accuracy. These results highlight the importance of carefully balancing loss configurations for better performance.

Model	Accuracy (%)	Consistency (%)	Verifiability (%)
All Losses	79.8	4.6	1.4
Without Story Choice Loss	70.1	3.7	19.7
Without Conflict Detection Loss	41.0	0.0	0.0
Without State Classification Losses	74.6	10.5	0.0

Table 4: Results achieved by ELECTRA model

Comparison of ELECTRA to Other Models

Compared to the other models, ELECTRA achieves the highest accuracy at 79.8% with all losses included, which is better than BERT at 78.3% and RoBERTa at 75.2%. However, its consistency at 4.6% and verifiability at 1.4% are lower than RoBERTa at 6.8% and 0.9%, and ALBERT at 5.7% and 2.0%. This shows that while ELECTRA performs well in identifying plausible stories, it is not as strong in tasks that require logical reasoning or generating interpretable results.

When the story choice loss is removed, the accuracy drops to 70.1%. At the same time, the verifiability improves significantly to 19.7%, which is the highest among all models in this configuration. This result suggests that removing the story choice loss allows the model to produce outputs that are easier to interpret. However, its consistency remains low at 3.7%, much lower than ALBERT at 20.8% and BERT at 28.0%, which perform better at maintaining logical connections in their predictions.

Removing the conflict detection loss has the most negative impact on all models, including ELECTRA. The trends are similar across models, which confirms that conflict detection is critical for reasoning tasks in the TRIP dataset.

Without state classification losses, ELECTRA's accuracy decreases slightly to 74.6%, which is comparable to BERT at 75.2% and DeBERTa at 72.4%. The consistency improves to 10.5%, which is higher than ALBERT at 8.8% but lower than BERT at 17.4%. However, verifiability remains at 0%, showing that state classification losses have little impact on this metric for any of the models.

In summary, ELECTRA achieves the best accuracy but struggles in consistency and verifiability compared to models like ALBERT and RoBERTa. This makes it effective for identifying plausible stories but less effective for tasks that require deeper reasoning or interpretable outputs.

4.4 Method #4: ASP Prompting

We used Google Colab Pro's A100 GPU to run the experiment described in section 3. Additionally, we saved the few-shot examples and the generated answers in separate `.csv` files available in the code base. See Table 5 for the accuracy and consistency scores obtained in the experiment.

Prompting Strategy	Accuracy	Consistency
Zero-shot	58%	-
Few-shot	68%	-
ASP Few-shot	65%	21%

Table 5: Performance Scores of Prompting Strategies

Unlike the original paper, we can notice that the Table 5 above doesn't show a **verifiability** metric. This metric is the hardest to measure because it requires the model to understand the underlying physical states (i.e., preconditions and effects) that contribute to the conflictive story. We skipped this metric because it would require additional prompting and text-processing efforts that escape the scope of this project. Although incorporating the verifiability metric can't change the results in Table 5, its presence is relevant to make a fair comparison between the models in the original paper and the new results. Now, assuming that we can restrict the comparison to the first two metrics, we can compare all the models.

We notice that zero-shot and few-shot strategies don't report consistency. This is because both prompting strategies don't require Mistral-7B-instruct to assess the second TRIP task. Instead, they rely on assessing the selection of the most plausible model. On the other hand, our ASP few-shot approach doesn't directly report consistency, but this metric can be extracted from the ASP program generated by the model.

We recall that the maximum accuracy obtained by the models in the original paper was 72.9%. We can observe that our prompting strategies perform below this accuracy line. Our best accuracy is obtained by the regular few-shot approach with 68% accuracy; however, this score doesn't have a relation with the other two tasks of TRIP. For the consistency metric, our ASP few-shot approach scores 21%, similar to the maximum consistency of 22.2% reported in the original paper. It is important to notice that our experiment only uses 100 samples from the test split in TRIP, and using the full test split would allow a comparison in better terms.

4.5 Comparison of Approaches

Approach	Accuracy (%)	Consistency (%)	Verifiability (%)
BERT	73.9	28.0	9.0
RoBERTa	73.6	22.4	10.6
DeBERTa	75.8	24.8	7.5
ALBERT	72.1	20.8	6.8
GPT-2 XL	8.6	3.8	1.1
ELECTRA	70.1	3.7	19.7
ASP	65	21	-

Table 6: Best Performance Achieved by Each Approach (When Prioritizing Verifiability and Consistency)

Table 5 provides a comparison of the best performance of the models from the SLED Group's paper and our approaches. Our approaches all performed slightly worse than the best performances achieved by the SLED Group, with the notable exceptions of GPT-2 XL, which performed very poorly, and ELECTRA, which achieved a remarkable 19.7% verifiability when story choice loss was excluded, nearly doubling the next highest verifiability of 10.6%, achieved by the RoBERTa model. However, given that ELECTRA's consistency was still very low, it is unlikely that ELECTRA was performing any physical commonsense reasoning.

The ASP approach was a novel strategy to improve the Mistral-7B model's performance. However, this strategy did not result in improved accuracy or consistency compared to the results achieved in the original SLED Group's paper. Although the Mistral-7B model has 7 billion parameters (AI, 2023), which is far more than any other model used on the TRIP dataset, it still did not outperform the other models.

5 Results Discussion

We tested four approaches to improve the baseline performance of LLMs on the TRIP dataset. However, our approaches did not improve upon the results achieved in the SLED Group's paper.

For Methods #1-3, the new models selected did not result in improved performance. Due to hardware constraints, we used relatively smaller models, which were on the order of millions of parameters, in our experiments. These smaller models were possibly not powerful enough to achieve improved performance on physical commonsense reasoning tasks.

The ALBERT model performed fairly similarly to the BERT-like models that were used in the SLED Group's paper. In fact, all the BERT-like models had relatively similar performance. Although the BERT-like models were trained differently from each other and had some differences in architecture from each other, their differences were not significant enough for them to perform differently in the context of physical commonsense reasoning tasks.

The GPT-2 XL model performed very poorly on the task. Since GPT-2 XL was released in 2019 (Radford et al., 2019), it is possible that a newer GPT model, such as GPT-4, would achieve better results.

The ELECTRA model also performed fairly similarly to the BERT-like models that were used in the SLED Group's paper. It did achieve a higher verifiability when story choice loss was excluded, but it was not able to improve on the consistency achieved in the SLED Group's paper.

For Method #4, although our prompting strategies score below the original performance of TRIP, we can still look carefully at the results to draw important discoveries. Comparing among the different strategies can allow us to answer whether Answer Set Programming can add value in prompting LLMs. Additionally, our strategies doesn't incur in training steps which could eventually improve the performance on these two tasks. The results showed in section 4.4 can be used as a baseline of comparison with new approaches that don't go over a training or fine-tuning step. Finally, from the text-processing step of our ASP approach, we can have a look at why LLM might be misreporting performance scores.

We can observe that our ASP approach performs better than the zero-shot prompting. This is evidence in favor of the beneficial addition of ASP directives in logical-reasoning prompts. These results also reinforce that using a few-shot approach can increase performance against a zero-shot strategies. Although our ASP approach doesn't perform better than the regular Few-shot, we gain interpretability.

This is because our model can justify with more arguments its selection. This is very important in a logical reasoning setting because explainable models can give more confidence in its predictions. Thus, our results support the idea that including ASP directives to the prompt can promote a logical reasoning with intermediate reasonable steps.

Our ASP approach differs from previous attempts because it doesn't go over a fine-tuning or instruction-tuning setting. This highlights that the results in section 4.4 are a underestimate of the real achievable performance of Mistral-7b-Instruct on these tasks. In this sense, our results show good performance considering that we didn't have to incur in training costs. Additionally, our ASP approach performs similar to the best performing model in the consistency metric. This is also very important because the LLM wasn't asked directly to find the inconsistent pairs of sentences, but it was able to do so through the modeling of the stories in Answer Set Programming code. Although the text-processing to extract the conflictive pairs isn't trivial, this step isn't adding logical reasoning because it is only extracting what was already modeled in the ASP program. Thus, the addition of the ASP approach does increase the quality of the reasoning because it builds on the implicit rules to make a final decision of plausibility. Moreover, because the ASP approach doesn't incur in training costs, it is more efficient than the original models in the consistency metric.

6 Future Work

While the approaches used in this paper did not result in improved performance on the TRIP dataset, there remain many avenues for future work that could potentially result in improved performance on the TRIP dataset.

1. **Larger Models**
   For Approaches 1-3, the ALBERT, GPT-2 XL, and ELECTRA models were chosen. These models were chosen partially due to hardware constraints, as the A100 GPU on Google Colab Pro did not have sufficient RAM to support significantly larger models. Future work should explore larger and newer models, as they may have better performance.

2. **Further Exploration of ASP Approach**
   We were unable to achieve improved performance with the ASP approach. However, further research into the ASP approach could include:

   (a) **Using the Full TRIP Dataset**
   Our experimentation with the ASP approach only used a subset of the TRIP dataset, but future work should explore using the entire dataset.

   (b) **Different Models**
   Using the Mistral-7B model did not result in improved performance. However, another model may be better suited to this approach. Future work could include using the ASP methodology from this paper with different models.

   (c) **Comparison of ASP Approach to Traditional Approach**
   To determine if the ASP approach improves performance compared to the traditional training approaches (i.e., the approach used by the SLED Group), future work should explore using the traditional training approach and the ASP approach with the same model to determine which approach yields the best performance.

   (d) **Investigating Impact of ASP Approach on Verifiability**
   Due to time constraints, investigating the impact of the ASP approach on verifiability was outside of the scope of this paper. Future work should explore if using the ASP approach is able to significantly improve the verifiability achieved by the model.

   (e) **Fine-Tuning for Solver-Augmented LLM**
   Our initial idea of using a solver-augmented LLM failed because the model would not generate ASP code for implicit rules. Further research should explore approaches to improve the model's ability to generate implicit rules, such as by fine-tuning the model on ASP code generation examples.

3. **Fine-Tuning Approaches**
   Fine-tuning the model on example physical commonsense reasoning data could potentially result in improved model performance and should be explored in future work.

7 Conclusion

We presented four approaches that aimed to improve the ability of large language models to perform physical commonsense reasoning. To evaluate the model's reasoning ability, the TRIP dataset was used, and the accuracy, consistency, and verifiability achieved by the model was recorded. The first three approaches involved using the ALBERT, GPT-2 XL, and ELECTRA models. These models were not able to improve upon the performance achieved in the SLED Group's research paper, likely because they are too small (in terms of number of parameters). Future work should explore using larger models and fine-tuning approaches to improve the performance of the models on the TRIP dataset. The final approach involved using answer set programming to improve the model's performance. However, this approach did not result in improved results compared to the SLED Group's original results. Future work should explore using the ASP approach with different models, comparing the ASP approach to the traditional training approach, investigating the impact of the ASP approach on the verifiability metric, and using fine-tuning to enable a solver-augmented LLM approach.

By assessing the TRIP dataset's tasks from different angles, we were able to see how different LLMs behave when facing logical reasoning tasks. Primarily, we found that all implementations faced a trade-off between interpretability and performance. For the first three models, like in the original paper, we observed that the selection of different loss functions can determine the distribution of high-, intermediate- and low-level understanding of the problem. The varying performance across different loss configurations highlights the critical role of carefully balancing loss functions for optimal performance. For example, when omitting the state classification losses, we observed an increase in consistency when compared to the all losses scenario. This means that we can artificially increase the performance of in higher level tasks by eliminating the objective of low-level understanding. Similarly, for our ASP few-shot approach, we observed that when incorporating the second task in the approach, the accuracy decreased with respect to the regular few-shot strategy. Additionally, we noticed that the ASP framework can add consistency in the reasoning process because it forces the deduction of plausibility by first exposing the implicit rules in the stories. Although the ASP approach doesn't perform better, it shows promise because we didn't finetune the model, and it perform similarly in the consistency metric.

Our work holds significant importance in advancing the understanding of LLM reasoning abilities, particularly in the domain of physical commonsense reasoning. By testing state-of-the-art models like ALBERT, GPT-2 XL, and ELECTRA on the challenging TRIP dataset, this research provides valuable insights into the current limitations of LLMs in physical reasoning tasks. The inclusion of ASP prompting as an alternative method showcases the importance of exploring novel strategies to enhance LLM reasoning. Moreover, the refining of prompting strategies and the potential fine-tuning on ASP directives open new alternatives to improve performance. Finally, the trade-off between model performance and interpretability is a critical consideration in the development of AI systems for real-world applications. This insight is valuable for researchers and practitioners aiming to create LLMs that are both powerful and transparent in their reasoning processes.

8 Division of Work

All group members contributed to creating the project proposal, final report, and final presentation. The remaining work for this project was divided by approach.

1. Ishita worked on Approach #1, using the ALBERT model, and also assisted with debugging of the other approaches.

2. Carlos worked on Approach #4, the ASP Prompting Strategy.

3. Hanning worked on Approach #3, using the ELECTRA model.

4. Fei worked on Approach #2, using the GPT-2 XL model.

9 Project's Code Base

The code for this project can be found at the following GitHub repository: `https://github.com/IshitaDeshmukh13/cse595-research-project-team-24`. The files contained in the GitHub repository are:

1. **"Verifiable_Coherent_NLU_ALBERT.ipynb"**: This file contains the code for running Ap-

proach #1. This code is adapted from (Storks et al., 2021).

2. **"Verifiable_Coherent_NLU_GPT_2XL.ipynb"**: This file contains the code for running Approach #2. This code is adapted from (Storks et al., 2021).

3. **"Verifiable_Coherent_NLU_ELECTRA.ipynb"**: This file contains the code for running Approach #3. This code is adapted from (Storks et al., 2021).

4. **"nlp_project_asp\few_shot.csv"**: This file contains the examples generated by ChatGPT to be used as few-shot examples. They contain the ASP program in text, and an analysis of plausibility also in text.

5. **"nlp_project_asp\nlp_project.ipynb"**: This file contains the implementation of Mistral 7B prompting strategies. It implements the three prompting strategies, and it requires the few_shot file to run few-shot and ASP few-shot experiments.

6. **"nlp...\clingo_few_shot_mistral_results.csv"**: This file contains the answers generated by Mistral 7B in the ASP few-shot strategy. This file is created by the nlp_project notebook, and it stores the generated answer and the ground truth of the experiment.

7. **"nlp_project_asp\consitency_metric.csv"**: This file contains the implementation of the text processing of Mistral 7B's answers. Specifically, it extracts and determines conflictive sentences by extracting implicit rules in the ASP generated program and matching them with the sentences.

References

Mistral AI. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Manuel Borroto, Irfan Kareem, and Francesco Ricca. 2024. Towards automatic composition of asp programs from natural language specifications. *arXiv preprint arXiv:2403.04541*.

Simone Caruso, Carmine Dodaro, Marco Maratea, Marco Mochi, and Francesco Riccio. 2024. Cnl2asp: converting controlled natural language sentences into asp. *Theory and Practice of Logic Programming*, 24(2):196–226.

K Clark. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jiazhan Feng, Ruochen Xu, Junheng Hao, Hiteshi Sharma, Yelong Shen, Dongyan Zhao, and Weizhu Chen. 2023. Language models can be logical solvers.

HuggingFace. 2022. Deberta: Decoding-enhanced bert with disentangled attention. `"https://huggingface.co/microsoft/deberta-base/blob/main/README.md"`.

HuggingFace. 2024a. Bert large model (uncased). `"https://huggingface.co/google-bert/bert-large-uncased"`.

HuggingFace. 2024b. roberta-large-mnli. `"https://huggingface.co/FacebookAI/roberta-large-mnli"`.

Z Lan. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Shane Storks, Qiaozi Gao, Yichi Zhang, and Joyce Chai. 2021. Tiered reasoning for intuitive physics: Toward verifiable commonsense language understanding. *arXiv preprint arXiv:2109.04947*.