Towards enhancing LLMs with logic-based reasoning: A Meta Interpretive Learning approach

Dany Varghese and Alireza Tamaddoni-Nezhad

Department of Computer Science, University of Surrey, Guildford, UK
{dany.varghese,atn}@surrey.ac.uk

Abstract. Large Language Models (LLMs) have achieved remarkable success in various natural language processing tasks. However, their ability to perform tasks requiring formal representation and reasoning remains limited. This paper explores the integration of Meta-Interpretive Learning (MIL) with LLMs to enhance their reasoning capabilities. Specifically, we employ a novel MIL framework called Meta Inverse Entailment to efficiently learn grammar from example sentences. In our experiments, we examine a hybrid GPT-MIL model on a simplified natural language grammar. The results suggest that the accuracy of GPT is significantly improved when it incorporates the grammar learned using MIL. This hybrid approach demonstrates the potential of combining LLMs' linguistic proficiency with MIL's rigorous formalism, leading to better performance in tasks demanding logical representation and reasoning.

1 Introduction

Large Language Models (LLMs) have revolutionised the field of Natural Language Processing (NLP), achieving unprecedented success across a broad spectrum of tasks such as machine translation, text summarisation, question answering, and more [28,7]. These models, exemplified by OpenAI's GPT-3 [3] and Google's BERT [7], leverage vast amounts of data and sophisticated neural network architectures to produce human-like text and understand context with remarkable accuracy. Their ability to generate coherent and contextually relevant text has found applications in diverse areas, from customer service automation to creative writing and beyond.

Despite their impressive achievements, LLMs face significant challenges when it comes to tasks requiring formal reasoning and representation. While they excel in tasks involving pattern recognition and language generation, their performance drops when dealing with problems that necessitate logical inference and generalisation from a limited number of examples [12]. One critical area where this limitation is evident is in grammar learning from examples, where the ability to abstract rules and apply them to new contexts is essential. LLMs, although powerful, often struggle to deduce and generalise grammatical rules from a few instances, highlighting a gap in their reasoning capabilities.

The importance of integrating 'ultra-strong' machine learning into LLMs becomes apparent in this context. Michie [13] proposed criteria to evaluate machine learning based on predictive performance and the comprehensibility of the learned knowledge. The weak criterion involves the machine learner's ability to enhance predictive accuracy as data increases. The strong criterion not only requires improved performance but also demands that the system presents its hypotheses in a symbolic form. The ultra-strong criterion further extends this by requiring the system to teach the hypothesis to a human, thereby improving the human's performance beyond what they could achieve by studying the training data alone. In the era of explainable AI (XAI), where transparency and interpretability are as crucial as performance, the ability to understand and replicate the reasoning behind model predictions is paramount [1]. 'Ultra-strong' learning embodies these principles, aiming to create models that are both powerful and understandable.

Incorporating 'ultra-strong' learning into LLMs can address their current limitations by enhancing their ability to reason and generalise. This is particularly relevant in tasks such as grammar induction, where understanding and applying rules across different contexts is necessary. A promising approach to achieve this integration is through Meta Interpretive Learning (MIL), a form of Inductive Logic Programming (ILP) that combines logic-based learning with meta-level reasoning [18,17]. MIL allows models to learn compact, human-readable representations of knowledge, which can then be used to reason about new instances effectively.

Inductive Logic Programming (ILP) has long been recognised for its potential to perform reasoning tasks efficiently by inducing hypotheses from observations [15]. ILP systems, such as Progol [16] and Aleph [22], have demonstrated the ability to learn relational theories and generate explanations in a way that is both interpretable and generalisable. MIL extends this capability by employing a meta-level framework that enables the learning of higher-order programs, thus enhancing the expressiveness and applicability of the induced knowledge.

Our research explores the integration of MIL with LLMs to bridge the gap between linguistic proficiency and formal reasoning. We introduce a novel MIL framework called Meta Inverse Entailment (MIE), designed to enhance the reasoning capabilities of LLMs. MIE employs a compact language model implemented using PyGol, a Python-based ILP system. In our experiments, we utilise simplified natural languages and MIL to learn the underlying grammar rules, which are then incorporated into the GPT model. This hybrid approach leverages the strengths of both LLMs and MIL: the linguistic proficiency of LLMs and the rigorous formalism of MIL.

The results of our experiments indicate a significant improvement in the accuracy of GPT when it incorporates the learned grammar from MIL. This suggests that the hybrid model can better generalise and reason about new instances, aligning with the principles of 'ultra-strong' learning. Our findings demonstrate the potential of combining LLMs with MIL to create AI systems that are not

only adept at understanding and generating language but also capable of sophisticated problem-solving and decision-making.

2 Formal language notation

The following definitions are taken from Muggleton et. al. [18].

Let Σ be a finite alphabet. Σ^* is the infinite set of strings made up of zero or more letters from Σ. λ is the empty string. uv is the concatenation of strings u and v. $\mid u \mid$ is the length of string u. A language L is any subset of Σ^*. Let ν be a set of non-terminal symbols disjoint from Σ. A production rule $r = LHS \rightarrow RHS$ is well-formed in the case that $LHS \in (\nu \cup \Sigma)^*$, $RHS \in (\nu \cup \Sigma \cup \lambda)^*$, and when applied replaces LHS by RHS in a given string. A grammar G is a pair $\langle s, R \rangle$ consisting of a start symbol $s \in \nu$ and a finite set of production rules R.

A grammar is regular Chomsky-normal in the case that it contains only production rules of the form $S \rightarrow \lambda$ or $S \rightarrow aB$ where $S, B \in \nu$ and $a \in \Sigma$. A grammar is linear context-free in the case that it contains only regular Chomsky-normal production rules or rules of the form $S \rightarrow Ab$ where $S, A \in \nu$ and $b \in \Sigma$. A grammar is context-free in the case that it contains only linear context-free Chomsky-normal production rules or rules of form $S \rightarrow AB$ where $S, A, B \in \nu$. A context-free grammar is said to be deterministic in the case that it does not contain two regular Chomsky-normal production rules $S \rightarrow aB$ and $S \rightarrow aC$ where $B \neq C$.

A sentence $\sigma \in \Sigma^*$ is in $L(G)$ iff given a start symbol $S \in \nu$ there exists a sequence of production rule applications where $R_i \in G$. A language L is regular, linear context-free or context-free in the case there exists a grammar G for which $L = L(G)$ where G is regular, linear context-free, or context-free, respectively. According to the context-free pumping lemma [9], if a language L is context-free, then there exists some integer $p \geq 1$ such that any string s in L with $\mid s \mid \geq p$ (where p is a constant) can be written as $s = uvxyz$ with substrings u, v, x, y and z, such that $\mid vxy \mid \leq p$, $\mid vy \mid \geq 1$ and uv^nxy^nz is in L for every integer $n \geq 0$.

3 Meta interpretive learning

Meta Interpretive Learning is a specialised subdomain of ILP. The primary objective of a MIL system is to find a hypothesis, denoted as H, in the context of provided background knowledge, B, and a set of examples, E. A MIL problem is a tuple $\langle B, E, M, I, H \rangle$, where $E = \{E^+, E^-\}$ is a set of ground atoms (negated ground atoms) of one or more target predicates, the positive and negative examples, respectively; B is the background knowledge, a set of definite clause definitions with datalog heads; M is a set of second-order metarules; I is a set of additional symbols reserved for invented predicates not defined in B or E^+ and H is a set of hypotheses.

Operating on the foundation of a Prolog meta-interpreter, MIL stands out in its learning approach compared to a typical Prolog meta-interpreter. The conventional meta-interpreter's objective is to validate a goal, achieved by consistently sourcing first-order clauses matching heads with the designated goal. In contrast, MIL takes a more comprehensive approach. In addition to the usual first-order clause retrieval, MIL also seeks higher-order metarules. These metarules, components of the background knowledge, are designed to align seamlessly with the goal.

Traditional MIL approaches always use metarules, a declarative language bias. A primary concern is the dependence on these declarative biases within machine learning, which, while crucial for directing the hypothesis search, often requires significant human input and expertise in specific domains. To avoid the issues associated with declarative language bias, this study employs a novel MIL approach called Meta Inverse Entailment (MIE) [25,26].

The MIE methodology employs the Bottom Clause of Relevant Literals (BCRL) and Meta Theory (MT) to generate individual hypothesis clauses. BCRL is an efficient tool for collecting all literals related to an example derived from background knowledge, thus delineating a boundary to search for a hypothesis during the learning process. Conversely, meta theory encapsulates a higher-order language bias autonomously extracted from background knowledge. The complete explanation of MIE is out of the scope of this paper, so we explain only the main concepts.

Definition 1 (Relevant literals ($\overrightarrow{\mathcal{R}_e}$)). *Let B be the background knowledge, $\overrightarrow{\mathcal{B}}$ be the ordered clause of ground literals of B, and e be an example. Then the relevant literals of e is defined as $\overrightarrow{\mathcal{R}_e} = e \leftarrow L_1, L_2, \cdots, L_n$ if and only if $L_i \in \overrightarrow{\mathcal{B}}$ and L_i be related literal (See Definition 3 in [25]) of either L_j or e such that $1 \leq j < i$.*

Definition 2 (Bottom clause of relevant literals ($\bot_{e,B}$)). *Let B be the background knowledge, e be a definite clause representing an example, $\mathcal{K}$ be a set of constant terms and $\overrightarrow{\mathcal{R}_e}$ be the relevant literals of example e as defined in the Definition 1. Then bottom clause of relevant literals of e, denoted as $\bot_{e,B}$ is $\overrightarrow{\mathcal{R}_e}$ where each unique occurrence of term $t_i \notin \mathcal{K}$ replaced with a new variable.*

The bottom clause of relevant literals introduced in MIE can resemble the bottom clause concept in Progol [16], but it does not use language bias such as mode declaration.

Definition 3 (Meta theory of relevant literals ($\mathcal{M}_e$)). *Let e be an example, $\overrightarrow{\mathcal{R}_e}$ be the relevant literals of e. Then, a clause $\overrightarrow{C} \in \mathcal{M}_e$ if and only if;*

1. *$\exists\Theta$ such that $\overrightarrow{C}\Theta \succeq_s \overrightarrow{\mathcal{R}_e}$ (Refrer Definition 19 in [23]) and*
2. *$\overrightarrow{C}$ is head_connected (Refer Definition 3 in [20]).*

The significant difference between a meta clause and a metarule is that a metarule specifies a relationship between the variables in the body of a rule. In meta theory, however, we avoid the tightly coupled nature of metarules to reduce computational complexity, deriving directly from background knowledge.

In MIE, We introduce a novel concept called the double-bounded hypothesis set. This set is defined as a sequential subsumption of meta theory, relative to the bottom clause of related literals. In this new framework, the hypothesis set of an example is constrained within specific boundaries: the bottom clause of relevant literals at the lower boundary and the encompassing meta theory at the upper boundary.

Definition 4 (Double-bounded hypothesis set ($HS(\bot_{e,B}, \mathcal{M}_e)$)). *Let e be an example, $\bot_{e,B}$ be a bottom clause of relevant literals of e as defined in Definition 2, $\mathcal{M}_e$ be a meta theory as defined in Definition 3. Then*

$$HS(\bot_{e,B}, \mathcal{M}_e) = \{h| \ s.t: \ \ (1)\ h \in \overrightarrow{\mathcal{L}_{\bot}},\ and\ (2)\ h \in \overrightarrow{\mathcal{L}_{\mathcal{M}}}\}$$

Now, Consider the problem specification of ILP. Suppose we are given a Horn program for a background knowledge B and a single Horn clause as an example E, then the hypothesis H should follow;

$$B \wedge H \models E \tag{1}$$

$$B \wedge H \ is\ consistent \tag{2}$$

Equations 1 & 2 are equivalent to

$$B \wedge \neg E \models \neg H \tag{3}$$

$$B \not\models \neg H \tag{4}$$

Like inverse entailment, MIE generates some theories that are not derived from B alone but from $B \wedge \neg E$. Instead of the bottom clause, we consider a bridge formula CT. So Equation 3 can be written as

$$B \wedge \neg E \models CT \tag{5}$$

$$CT \models \neg H \tag{6}$$

Finally, Equation 6 can also written as

$$H \models \neg CT \tag{7}$$

Also by Equations 4 and 6, we have

$$B \not\models CT \tag{8}$$

Example Grammar	L	BCRL
S $\rightarrow$ 0A A $\rightarrow$ 1B B $\rightarrow$ 0B B $\rightarrow \lambda$	$(01)0^*$	string(S):- move(S,0,A), move(A,1,B), move(B,0,B), accept(B).

Table 1: Grammar, Language and BCRL for a regular language

By Equation 5, the set of clausal theories, CT, can be obtained by generating the double-bounded hypothesis set. Then

$$HS(\bot_{e,B}, \mathcal{M}_e) \models \neg CT \tag{9}$$

The candidate hypothesis (H) will be a subset of $HS(\bot_{e,B}, \mathcal{M}_e)$.

$$H = \{h | h \in HS(\bot_{e,B}, \mathcal{M}_e) \text{ and } h \models \neg CT\} \tag{10}$$

In Equation 5, we introduce a bridge formula, a set of clausal theories as CT, which replaces the concept of the bottom clause and characteristic clauses introduced in [10].

3.1 MIE framework applied to grammar learning

The Chomsky language types are organised in a hierarchy of inclusion, wherein regular languages are a subset of context-free languages (regular $\subseteq$ context-free). Within the realm of grammatical inference [5], methods for learning regular languages have been the subject of substantial research. A significant number of these methods start with a prefix tree acceptor and then proceed to merge states in a more gradual manner.

Table 1 demonstrates how the MIE approach can be used to learn a regular language. Upon a closer examination of the BCRL for the regular grammar in Table 3, it is evident that it shares similar properties with prefix trees.

Proposition 1 (Unique $\bot_R$ for regular languages). *Prefix trees act as a compact bottom theory in the MIL setting for regular languages.*

Proof. Refer to Proposition 6 from [18] for the proof.

Proposition 2 (Unique $\bot$ for context-free languages). *Any $\bot_C$ for a context-free language contains a set of atoms representing a regular prefix tree.*

Proof. Refer to Proposition 7 from [18] for the proof.

Using, Equation 10, MIE grammar learning generates all the possible sets of hypotheses for regular grammar as follows;

$$H_R = \{h | h \in HS(\bot_R, \mathcal{M}_e) \text{ and } h \models \neg CT\} \tag{11}$$

Equation 11 can also be applied to Context-Free Grammars (CFGs) as follows:

$$H_R = \{h | h \in HS(\bot, \mathcal{M}_e) \text{ and } h \models \neg CT\} \tag{12}$$

Also, it has to be noted that the resulting hypothesis (grammar) is then tested for non-coverage of each of the negative examples.

4 Experiments

The following section provides a detailed overview of experiments conducted to evaluate the ability of machine learning models to learn grammar from a simplified natural language. We investigate the following Null Hypothesis throughout the experiments.

Null Hypothesis $\mathbf{H_1}$. PyGol cannot learn a simplified natural language grammar.

Null Hypothesis $\mathbf{H_2}$. PyGol cannot outperform GPT in learning a simplified natural language grammar.

Null Hypothesis $\mathbf{H_3}$. GPT's reasoning efficiency in learning simplified natural grammar cannot be significantly improved by including production rules from MIL.

These experiments assess the performance of the MIL system, PyGol [25,27], in comparison to the GPT model (gpt-4-turbo), highlighting their respective capabilities in grammar learning tasks. All datasets and learning systems used in these experiments are available at https://github.com/hmlr-lab/Compact-Language-Model.

4.1 Learning a simplified natural language grammar

In this experiment, we investigate the capability of PyGol and the GPT model to learn simplified natural grammar.

Materials & methods Our approach to generating and evaluating grammar is based on training examples from the same domain as described by Muggleton et al. [19]. The dataset includes 50 sentences from a simplified natural language grammar (SNG), such as "a ball hits the small dog" with an equal distribution of positive and negative examples, leading to a baseline accuracy of 50%. We used the training data to generate the grammar and evaluated it against a separate test dataset. For each leave-out size, we sampled ten times, and for each sample, predictive accuracies were computed using 10-fold cross-validation. The query used in GPT API is shown in Fig. 1.

```
f"""
Hey ChatGPT,
The language 'L' accepts sentences in positive examples and
                                rejects them in the negative
                                examples of Training Data.
Training Data:
    Positive examples: {training_positive}
    Negative examples: {training_negative}
Testing Data: {testing_example}
Instructions:
1. For each test sentence in the Testing Data, determine if
                                it belongs to the language 'L
                                '.
2. ans = 1 if sentences in the  Testing data that belong to
                                language 'L'.
3. ans = 0 if sentences in  Testing data that do not belong
                                to language 'L'.
4. Make a list of 'ans' as in the same order fas in Testing
                                data.

Return the array of 'ans'.
"""
```

Fig. 1: GPT API query for experiment 1

Results & discussion From the results shown in Fig. 2, we can reject the Null Hypothesis H_1, as PyGol produces hypotheses with predictive accuracies significantly higher than the baseline accuracy. These results suggest that GPT models can learn simplified natural grammar more accurately, even with a limited number of examples. This improved performance is likely due to GPT models having access to extensive background information about natural languages.

From the results, it is clear that PyGol surpasses the GPT model, leading to the rejection of the Null Hypothesis H_2. PyGol's superior performance on NLP tasks, such as those involving the SNG dataset, is mainly due to its ability to generalise grammars from very few examples through predicate invention and recursion. This ability enables PyGol to effectively infer underlying grammatical rules with minimal data, showcasing its efficiency and robustness in learning tasks. The compactness of the grammar learned using PyGol is evident from an example in Table. 2, showcasing concise production rules for a target language. This highlights PyGol's efficiency in deriving effective and compact grammatical structures using recursion.

4.2 Evaluating GPT's reasoning efficiency with MIL production rules

In this experiment, we evaluate the Null Hypothesis H_3.

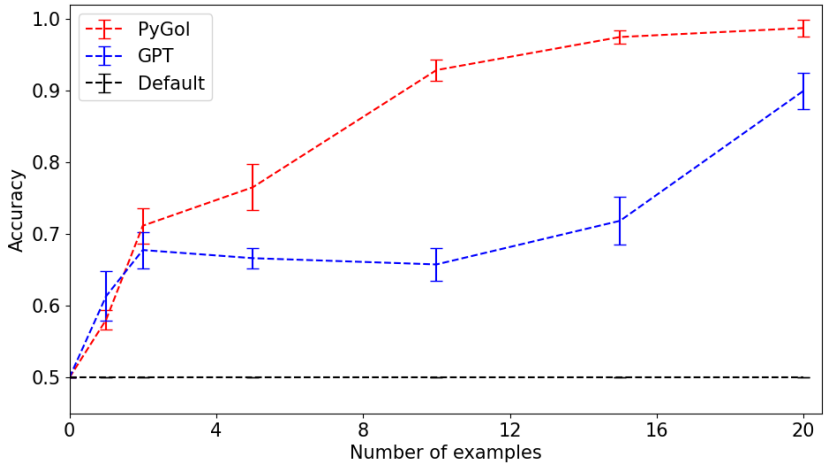

Fig. 2: Average predictive accuracies on simplified natural language grammars

Examples	Language	Type	Grammar
a dog hits a ball	$s(S_1, S_2) \rightarrow np(S_1, S_3), vp(S_3, S_4), np(S_4, S_2)$ $np(S_1, S_2) \rightarrow det(S_1, S_3), noun(S_3, S_2)$ $vp(S_1, S_2) \rightarrow verb(S_1, S_2)$	Natural grammar	$S \rightarrow aB$ $S \rightarrow$ NP VP NP $NP \rightarrow$ DT NOUN $VP \rightarrow VERB$

Table 2: An overview of the examples used in the PyGol learning process, the target languages for grammar induction, the type of grammar, and the specifics of the grammar learned

Materials & methods In this study, we utilised the same dataset as described in Section 4.1. The dataset comprises positive and negative examples from which we first generate production rules for a language. These production rules, along with the examples, were then fed into the GPT model. To evaluate the accuracy of the GPT model, we compared it against PyGol and a GPT model that does not use production rules. For each leave-out size, we sampled ten times, and for each sample, predictive accuracies were computed using 10-fold cross-validation. In the API (Refer Fig. 3) used to evaluate the GPT model with production rules, we have clearly mentioned that grammar learned from GPT is 'partial grammar' for the language.

Results & discussion The results from the experiment are illustrated in Fig. 4. It is evident that adding production rules from PyGol increases the learning efficiency of GPT, resulting in an average accuracy improvement of 8 ± 6 ($max = 18$). This evidence allows us to refute Null Hypothesis H_3. Furthermore,

```
f"""
Hey ChatGPT,
The language 'L' accepts sentences in  positive examples and
rejects in the negative examples of Training Data.
The possible production rules of  partial  grammar for 'L' is
given, in which "vp" means verb phrase, "np" means noun
phrase,  "det" means determiner, "adj" means  adjective
and "prep"  means peposition.

production rules : {H}
Training Data:
    Positive examples: {training_positive}
    Negative examples: {training_negative}
Test Data:
    Positive examples: {testing_positive}
    Negative examples: {testing_negative}
Instructions:
1.For each test sentence in the Test Data, determine
if it belongs to the language 'L'.
3.ans = 1 if  sentences in the  Testing data that
belong to language 'L'.
4.ans = 0 if sentences in  Testing data that do not
belong to language 'L'.
5.Make a list  of 'ans' as  in the same order fas in Testing
                                data.
Return just the array of 'ans'."""
```

Fig. 3: GPT API query for experiment 2

Fig. 4 shows that combining ChatGPT with production rules from PyGol significantly enhances performance compared to using ChatGPT alone, highlighting the value of integrating structured knowledge. However, as the number of examples increases, the combined model might start overfitting to specific patterns in the training data, which do not generalise well to the test data. Overfitting occurs when a model becomes too tailored to the nuances of the training set, causing it to lose effectiveness on unseen data. This phenomenon likely explains why the combined model's accuracy drops after incorporating more than 10 examples with respect to PyGol.

Additionally, the results of the experiment show that the True Negative (TN) value is much better than the True Positive (TP) value. This indicates that the model is more effective at correctly identifying negative examples (i.e., correctly rejecting instances that do not belong to the target class) than it is at correctly identifying positive examples (i.e., correctly accepting instances that do belong to the target class). This disparity suggests that the model might be more conservative in its predictions, leaning towards correctly identifying non-target examples but potentially missing some target examples.

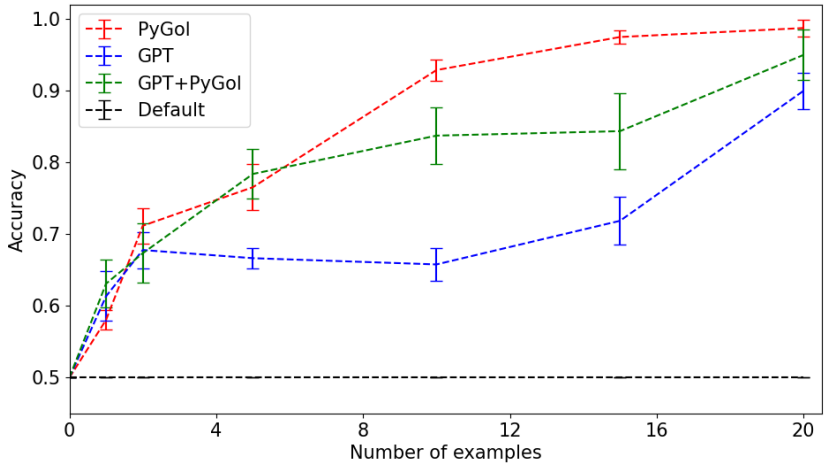

Fig. 4: Average predictive accuracies of GPT enhanced with production rules

5 Related work

Grammatical inference, or grammatical induction, is learning grammar from examples, closely linked to machine learning and formal language theory. It has significant real-world applications, including speech recognition [29], computational linguistics [8], and computational biology [11]. The problem of learning regular languages, represented by deterministic finite state automata, has been well-studied, with efficient algorithms existing since the 1950s [14]. Learning context-free languages is widely considered intractable, with the current state of the art largely comprising negative results [5]. Some positive PAC (Probably Approximately Correct) learning results exist for regular languages [6], but these have not been extended to context-free grammar.

ILP has been applied to grammatical inference along with other machine-learning approaches. The first seminal work for grammatical inference in ILP involved the system MERLIN 2.0 [2], a successor to an earlier system where predicate invention is guided by sequences of input clauses in SLD-refutations of positive and negative examples relative to an overly general theory. Unlike its predecessor, which searches for the minimal finite-state automaton to generate all positive and no negative sequences, MERLIN 2.0 induces Hidden Markov Models from positive sequences only. ILP systems typically require predicate invention even for learning regular languages, a problem recognised since the early days of ILP. Although Cussens and Pulman [4] applied ALEPH to learning natural language grammar, their approach avoids predicate invention by assuming all predicates, like noun phrases (np), are known in the background knowledge. The first seminal work to learn context-free grammar using ILP was developed

by Muggleton et al. [18] using Meta Interpretive Learning. The MIL framework implements predicate invention and recursive generalisations through abduction with respect to a meta-interpreter. This approach is based on a previously unexplored case of Inverse Entailment for grammatical inference of regular languages, marking a significant advancement in the field.

Large language models (LLMs) [24], such as GPT (Generative Pre-trained Transformer) [30], have revolutionised the field of natural language processing (NLP). These models leverage the Transformer architecture, introduced by Vaswani et al. [28], which is based on self-attention mechanisms to efficiently capture long-range dependencies and contextual relationships within text data. LLMs are typically pre-trained on extensive and diverse corpora, allowing them to learn rich linguistic patterns, syntactic structures, and semantic nuances. This pre-training phase equips LLMs with a broad understanding of language, enabling them to perform a wide array of NLP tasks with remarkable proficiency [21]. Technically, LLMs consist of multiple layers of Transformer encoders and decoders, which process input text through a series of attention heads and feed-forward neural networks. Each layer refines the representation of the input by focusing on different aspects of the context, effectively enabling the model to attend to relevant parts of the text. The pre-training phase involves unsupervised learning, where the model predicts the next word in a sentence, thereby learning from the sequential structure of the language.Following pre-training, LLMs undergo fine-tuning on task-specific datasets, enhancing their performance on particular applications such as machine translation, text summarisation, and question answering.The scalability and adaptability of LLMs make them a cornerstone of modern NLP research, driving advancements in the development of more sophisticated and human-like language processing systems.

6 Conclusion

This paper first, explores the implementation of the meta-interpretive learning (MIL) approach for a simplified natural grammar. The MIL framework offers several advantages over the standard Inductive Logic Programming (ILP) framework, particularly in incorporating predicate invention and mutual recursion. Our experiments utilised a novel MIL framework called MIE, implemented using PyGol. A key advantage of MIE is its ability to avoid the use of declarative language bias, such as meta-rules, making MIL more user-friendly and reliable. Additionally, MIE's hybrid learning approach, with its double-bounded hypothesis space, enables efficient hypothesis learning, predicate invention, and recursion. The results suggest that the generalisation power of MIE is both efficient and accurate, closely matching the MIL approach introduced by Muggleton et al. [18].

Learning natural language grammar from a small number of examples presents significant challenges for models like ChatGPT. The primary issues include insufficient data coverage, leading to poor generalisation and a tendency to overfit specific examples. Although GPT models possess extensive background knowl-

edge of natural language, they lack the structured information necessary for formal grammar and are not inherently designed for explicit rule learning, which is crucial for understanding and applying the strict grammatical rules of natural languages.

To address these limitations, we proposed a hybrid model that combines production rules learned from Meta-Interpretive Learning and GPT models. This amalgamation represents a form of 'ultra-strong' machine learning, which traditionally emphasises the explainability of machine models using symbolic tools. However, in our approach, symbolic rules are used to enhance the learnability of Large Language Models (LLMs), a popular domain in AI. This integration aims to improve the reasoning capabilities of language models, addressing a fundamental issue in LLMs.

The results of our study indicate that integrating production rules from PyGol with ChatGPT leads to a significant improvement in performance, demonstrating that structured knowledge enhances GPT's learning efficiency. However, as the number of examples increased, the combined model's accuracy declined after incorporating more than 10 examples. This decline suggests overfitting, where the model becomes too tailored to the specific patterns in the training set, leading to reduced generalisation to unseen data.

These findings underscore both the potential and challenges of integrating structured knowledge with machine learning models. While the combination of production rules with GPT enhances performance, the standalone PyGol approach remains superior for simplified natural grammar tasks. Future research could explore more sophisticated integration strategies or hybrid models that leverage the strengths of both PyGol and GPT-based approaches. By doing so, we can aim to create more robust models that effectively balance structured rule learning with the extensive background knowledge inherent in GPT models.

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: A survey on explainable artificial intelligence (xai). IEEE Access **6**, 52138–52160 (2018)
2. Boström, H.: Predicate invention and learning from positive examples only. In: European Conference on Machine Learning. pp. 226–237. Springer (1998)
3. Brown, T., Mann, B., Ryder, et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)
4. Cussens, J., Pulman, S.: Experiments in Inductive Chart Parsing, p. 143âĂŞ156. Springer-Verlag, Berlin, Heidelberg (2001)
5. De La Higuera, C.: A bibliographical study of grammatical inference. Pattern recognition **38**(9), 1332–1348 (2005)
6. Denis, F.: Learning regular languages from simple positive examples. Machine Learning **44**, 37–66 (2001)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186 (2019)

8. Heinz, J., De la Higuera, C., Van Zaanen, M.: Grammatical inference for computational linguistics. Springer Nature (2022)
9. Hopcroft, J. E., .U.J.D.: An introduction to automata and formal languates (1979)
10. Inoue, K.: Induction as consequence finding. Machine Learning **55**, 109–135 (01 2004), https://doi.org/10.1023/B:MACH.0000023149.72125.e2
11. López, V.F., Aguilar, R., Alonso, L., Moreno, M.N., Corchado, J.M.: Grammatical inference with bioinformatics criteria. Neurocomputing **75**(1), 88–97 (2012)
12. Marcus, G., Davis, E.: Rebooting AI: Building artificial intelligence we can trust. Vintage (2020)
13. Michie, D.: Machine learning in the next five years. In: Proceedings of the 3rd European Conference on European Working Session on Learning. p. 107âĂŞ122. EWSL'88, Pitman Publishing, Inc., USA (1988), https://dl.acm.org/doi/10.5555/3108771.3108781
14. Moore, E.F., et al.: Gedanken-experiments on sequential machines. Automata studies **34**, 129–153 (1956)
15. Muggleton, S.: Inductive logic programming. New Generation Computing **8**, 295–318 (1991), http://dx.doi.org/10.1007/BF03037089
16. Muggleton, S.: Inverse entailment and progol. New Generation Computing **13**, 245–286 (1995). https://doi.org/10.1007/BF03037227
17. Muggleton, S., Lin, D., Tamaddoni, N.A.: Meta-interpretive learning of higher-order dyadic datalog: predicate invention revisited. Machine Learning **100**, 49–73 (2015), http://dx.doi.org/10.1007/s10994-014-5471-y
18. Muggleton, S., Lin, D., Pahlavi, N., Tamaddoni-Nezhad, A.: Meta-interpretive learning: application to grammatical inference. Machine Learning **94**(1), 25 – 49 (2014), https://github.com/metagol/metagol
19. Muggleton, S., Lin, D., Tamaddoni-Nezhad, A.: Mc-toplog: Complete multi-clause learning guided by a top theory. pp. 238–254 (07 2011)
20. Muggleton, S., Tamaddoni-Nezhad, A.: Qg/ga: A stochastic search for progol. In: Inductive Logic Programming: 16th International Conference, ILP 2006, Santiago de Compostela, Spain, August 24-27, 2006, Revised Selected Papers 16. pp. 37–39. Springer (2007)
21. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training (2018)
22. Srinivasan, A.: Aleph: A learning engine for proposing hypotheses. Machine Learning **38**, 95–133 (2001)
23. Tamaddoni-Nezhad, A., Muggleton, S.: The lattice structure and refinement operators for the hypothesis space bounded by a bottom clause. Machine learning **76**, 37–72 (2009)
24. Thirunavukarasu, A.J., Ting, D.S.J., Elangovan, K., Gutierrez, L., Tan, T.F., Ting, D.S.W.: Large language models in medicine. Nature medicine **29**(8), 1930–1940 (2023)
25. Varghese, D., Barroso-Bergada, D., Bohan, D.A., Tamaddoni-Nezhad, A.: Efficient Abductive Learning of Microbial Interactions using Meta Inverse Entailment. In Proceedings of the 31st International Conference on ILP (2022), (in press)
26. Varghese, D., Patel, U., Krause, P., Tamaddoni-Nezhad, A.: Few-shot learning forÂăplant disease classification using ilp. In: Advanced Computing. pp. 321–336. Springer Nature Switzerland, Cham (2023)
27. Varghese, D., Tamaddoni-Nezhad, A.: PyGol. https://github.com/PyGol/ (2022)
28. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

29. Vidal, E., Casacuberta, F., García, P.: Grammatical inference and automatic speech recognition. In: Ayuso, A.J.R., Soler, J.M.L. (eds.) Speech Recognition and Coding. pp. 174–191. Springer Berlin Heidelberg, Berlin, Heidelberg (1995)
30. Zhu, Q., Luo, J.: Generative pre-trained transformer for design concept generation: an exploration. Proceedings of the design society **2**, 1825–1834 (2022)