

Logic Before Language: Pre-pretraining on Formal Derivations Fosters Skill Acquisition and Compressibility

Jo-Ku Cheng Nikolaos Aletras Marco Valentino

School of Computer Science, University of Sheffield, United Kingdom
{jcheng34,n.aletras,m.valentino}@sheffield.ac.uk

Abstract

Pre-pretraining language models (LMs) on symbolic data can accelerate and improve natural language acquisition. However, existing pre-pretraining tasks, such as Dyck and procedural algorithms, rely on narrow primitives that fail to capture the expressive capacity of natural language. Moreover, prior studies remain restricted to relatively small token budgets, offering limited insight into skill emergence and representational dynamics. To address these limitations, we propose logic pre-pretraining (Logic-PPT) as a principled initialization strategy, leveraging formal derivations to impart richer structural and linguistic biases. Formal derivations require abstract mechanisms that are central to natural language, simultaneously binding variables, connecting quantifiers and relational dependencies, and composing predicate-argument structures over long contexts. Scaling our evaluation to a 100B-token regime, logic pre-pretraining substantially accelerates skill acquisition in LMs, achieving 80% accuracy on linguistic tasks with 36B fewer tokens than standard initialization, and outperforming alternative pre-pretraining baselines. Mechanistically, formal derivations induce persistent structural reorganization, distinctively characterized by a lower-rank, spectrally concentrated representation space. Crucially, we show that this internal geometry enables improved model compressibility via pruning, matching the dense baseline performance even at $\approx 33\%$ sparsity.

1 Introduction

Recent work on *symbolic pre-pretraining*, exposing language models (LMs) to procedurally generated symbolic data prior to natural language training, has emerged as a strategy to accelerate natural language acquisition and improve downstream performance and robustness to noise (Hu et al., 2025; Jiang et al., 2026; Guo et al., 2026; Lee et al., 2026a; Mita et al., 2026).

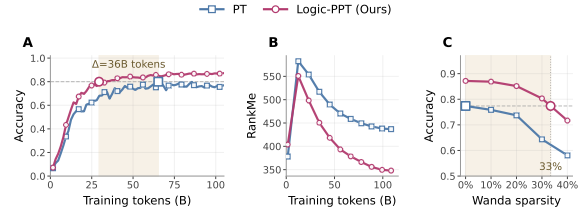


Figure 1: **(A) Faster learning:** Logic-PPT reaches 0.8 mean accuracy using 36B fewer natural language training tokens than the PT baseline in the 100B-token run. **(B) Lower-rank representations:** Logic-PPT promotes a more structured and compact internal geometry, organizing representations around a smaller set of directions. **(C) Greater compression robustness:** At $\approx 33\%$ sparsity, the Logic-PPT model retains accuracy comparable to a dense PT baseline.

However, existing pre-pretraining frameworks remain limited in two critical respects. Qualitatively, the symbolic tasks considered in prior work capture only a narrow range of structural inductive biases relevant to language (Wilcox et al., 2018; Montague, 2002; Hewitt and Manning, 2019). For example, Dyck languages primarily isolate hierarchical nesting and bracket symmetry (Hu et al., 2025), whereas algorithmic tasks such as sorting isolate purely procedural sequence transformations, lacking the expressive capacity inherent to natural language (Jiang et al., 2026). Quantitatively, prior studies have generally trained on fewer than 10B natural language tokens (Hu et al., 2025; Lee et al., 2026a; Jiang et al., 2026), leaving unclear whether the impact of symbolic pre-pretraining persists at longer training and whether it fundamentally changes internal representational dynamics.

These limitations motivate our contribution: *we introduce formal logical derivations as a more expressive and structurally richer source of pre-pretraining data and study its impact on imparting linguistic biases and representational changes at larger training token budget (i.e. 100B tokens).*

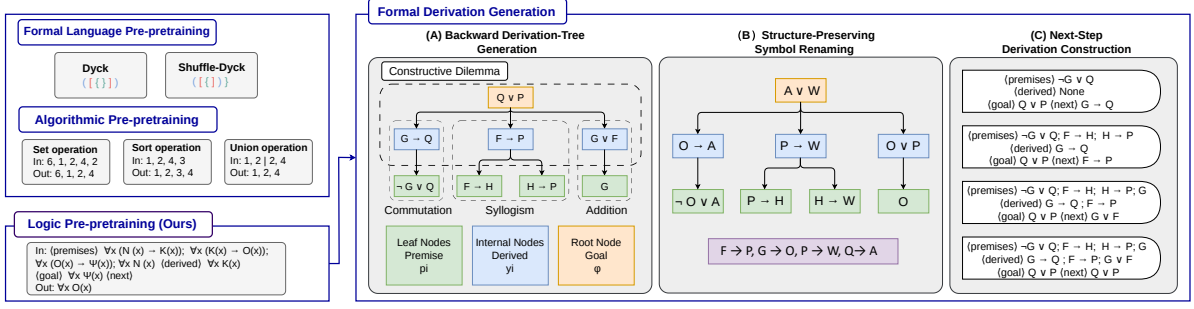


Figure 2: (Left) We introduce *Logic Pre-pretraining (Logic-PPT)*, a new framework for initializing LMs on symbolic data before exposure to natural language. Compared to previous approaches relying on formal languages and procedural algorithms, Logic-PPT adopts more expressive and structurally richer formal derivations to impart deeper linguistic inductive biases. (Right) Overview of the methodology to generate synthetic formal derivations for Logic-PPT: (A) A derivation-tree is constructed by backward chain of rule schemata, starting from a goal and recursively replacing selected leaves with instantiated premises. (B) Each derivation-tree is augmented with symbol-renaming variants that preserve its topology and rule sequence. (C) A post-order traversal produces a leaf-to-root derivation sequence to construct a next-step prediction example.

Specifically, we organize our investigation around three questions: (1) *Behavioral*: Can pre-training on formal derivations accelerate and improve language and skill acquisition in LMs? (2) *Mechanistic*: How does pre-training affect internal representational geometry and dynamics? (3) *Interventional*: can we leverage the structural changes induced by pre-training to further improve LMs’ efficiency?

To answer these questions, we curate a library of 247 formal logic schemata spanning propositional, term, and first-order logic. By leveraging backward chaining algorithms, we construct a large-scale next-step formal derivation dataset to symbolically pre-pretrain Transformer models prior to a 100B-token natural language pretraining run on FineWeb-Edu (Lozhkov et al., 2024). We systematically compare our logic pre-training framework (*Logic-PPT*) against established symbolic methods, including formal languages (Dyck/ShuffleDyck) and procedural algorithms (Set/Sort/Union), as well as a random initialization baseline. Our key findings can be summarized as follows:

- Logic-PPT significantly accelerates language and skill acquisition, achieving 80% accuracy across a suite of linguistic tasks using 36B fewer natural language tokens than standard initialization and outperforming existing symbolic PPT methods by 7.1% points in accuracy at the final checkpoint.
- Mechanistically, formal derivations induce a distinctive lower-rank, spectrally concentrated representation space characterized by clearer

layer differentiation that persists throughout 100B tokens of language pretraining.

- We demonstrate for the first time that this internal geometry translates into superior pruning resilience, matching dense baseline performance at $\approx 33\%$ sparsity. At 40% sparsity, Logic-PPT incurs only a 14.4% relative drop compared to an average 26.8% degradation across alternative symbolic PPT methods.

2 Methodology

Our objective is to investigate how pre-training on formal logical derivations shapes subsequent natural language acquisition. Specifically, we examine whether formal logic instills inductive biases that accelerate skill emergence, reorganize internal representations, and enhance model efficiency. Our framework consists of two main stages: *Logic Pre-Pretraining (Logic-PPT)* and *Natural Language Pretraining (PT)* via backbone transfer.

2.1 Defining Formal Derivations

Formal logical derivations demonstrate the validity of an argument by deriving conclusions from premises through a sequence of rule-governed steps (Hardegree, 1994). Each step is either one of the original premises or a formula that follows from preceding steps according to a derivation rule, with the final step being the conclusion.

Definition. Let Γ be a set of premises, $\mathcal{A}$ a set of logical axioms, and $\mathcal{R}$ a set of inference rules. A formal logical derivation of a formula φ from Γ is

a finite sequence of formulas

$$D = \langle \alpha_0, \dots, \alpha_n \rangle$$

such that α_n is φ and, for each $k \leq n$, either

1. $\alpha_k \in \Gamma \cup \mathcal{A}$, or
2. α_k is obtained from formulas occurring earlier in the sequence by applying an inference rule in $\mathcal{R}$.

If such a derivation exists, we write $\Gamma \vdash_{\mathcal{R}} \varphi$.

2.2 Stage 1: Logic Pre-pretraining

In the first stage, the model is trained exclusively on formal derivations. We denote the randomly initialized model parameters as $\Theta_{\text{logic}} = \{\Theta_{\text{backbone}}, \mathbf{E}_{\text{logic}}, \mathbf{H}_{\text{logic}}\}$, where Θ_{backbone} represents the shared Transformer backbone parameters (multi-head self-attention layers, feed-forward networks, and layer normalizations), $\mathbf{E}_{\text{logic}} \in \mathbb{R}^{|V_{\text{logic}}| \times d}$ is the input token embedding matrix, and $\mathbf{H}_{\text{logic}} \in \mathbb{R}^{d \times |V_{\text{logic}}|}$ is the output language-modeling head. Here, d denotes the hidden dimension and V_{logic} represents a character-level logic vocabulary.

Task Formulation. We cast formal derivation generation as an auto-regressive next-step prediction task. Given a derivation $D = \langle \alpha_1, \dots, \alpha_n \rangle$, we construct one training example for each step i , where the input contains the premises $\Gamma = \{p_1, \dots, p_m\}$, the derivation prefix $D_{<i} = \langle \alpha_1, \dots, \alpha_{i-1} \rangle$, and the target conclusion φ , and its output is the next derivation step $y_i = \alpha_i$. The input is serialized as the token sequence x_i :

$$\begin{aligned} x_i = & \langle \text{premises} \rangle p_1; \dots; p_m \\ & \langle \text{derived} \rangle \alpha_1; \dots; \alpha_{i-1} \\ & \langle \text{goal} \rangle \varphi \\ & \langle \text{next} \rangle. \end{aligned} \quad (1)$$

This construction yields a training pair (x_i, y_i) . All sequences are processed using a domain-specific character-level tokenizer.

Optimization Objective. The model is trained autoregressively to predict the target step y_i . To prevent the model from expending capacity on reconstructing fixed prompt contexts, the cross-entropy loss is computed strictly over the tokens corresponding to the target step y_i :

$$\mathcal{L}_{\text{logic}}(\Theta_{\text{logic}}) = - \sum_{j=1}^{|y_i|} \log P_{\Theta_0}(y_{i,j} \mid x_i, y_{i,<j}). \quad (2)$$

Schema	Premises	Conclusion
Propositional		
Modus ponens	$P \rightarrow Q, P$	Q
Modus tollens	$P \rightarrow Q, \neg Q$	$\neg P$
Resolution	$P \vee Q, \neg P \vee R$	$Q \vee R$
Term		
Barbara	$A \models B, B \models C$	$A \models C$
Ferio	$C \models \neg B, A \not\models \neg C$	$A \not\models B$
CM-Barbara	$A \models \neg B, \neg B \models C$	$A \models C$
First-order		
Universal instantiation	$\forall x \Phi(x)$	$\Phi(t)$
Universal modus ponens	$\forall x(\Phi(x) \rightarrow \Psi(x)), \Phi(a)$	$\Psi(a)$
Equality substitution	$s = t, \Phi(s)$	$\Phi(t)$

Table 1: Representative formal derivation schemata from the three logic families used for Logic-PPT.

After Logic-PPT, the resulting parameters are denoted as $\Theta_{\text{logic}}^* = \{\Theta_{\text{backbone}}^{\text{logic}}, \mathbf{E}_{\text{logic}}^*, \mathbf{H}_{\text{logic}}^*\}$.

2.3 Stage 2: Natural Language Pretraining

Following Logic-PPT, we transfer the inductive biases learned in the Transformer backbone to a standard natural language pretraining setup.

Vocabulary & Head Re-initialization. Because symbolic logic and natural language operate over distinct token spaces, vocabulary-dependent parameters are non-transferable. We discard the symbolic token embedding matrix $\mathbf{E}_{\text{logic}}^*$ and output head $\mathbf{H}_{\text{logic}}^*$. We retain the pre-trained Transformer backbone parameters $\Theta_{\text{backbone}}^{\text{logic}}$ and attach a randomly initialized embedding matrix $\mathbf{E}_{\text{lang}} \in \mathbb{R}^{|V_{\text{lang}}| \times d}$ and output head $\mathbf{H}_{\text{lang}} \in \mathbb{R}^{d \times |V_{\text{lang}}|}$, sized to match a natural language tokenizer vocabulary V_{lang} .

Natural Language Pretraining. The composite parameter set $\Theta = \{\Theta_{\text{backbone}}^{\text{logic}}, \mathbf{E}_{\text{lang}}, \mathbf{H}_{\text{lang}}\}$ is trained on a large-scale natural language text corpus $\mathcal{D}_{\text{text}}$ (FineWeb-Edu). All parameters remain fully trainable under the standard causal language modeling objective:

$$\mathcal{L}_{\text{lang}}(\Theta) = - \sum_{t=1}^T \log P_{\Theta}(w_t \mid w_{<t}), \quad (3)$$

where $w_t \in V_{\text{lang}}$ denotes the t -th natural language token in a sequence of length T .

2.4 Logic Pre-pretraining Data

Collecting Formal Schemata. To cover complementary forms of logical structure and inference, we first curate a library $\mathcal{R}$ of formal derivation rule schemata from three logic families: propositional logic, term logic, and first-order logic. Propositional logic captures truth-functional composi-

tion and transformations involving logical connectives. Term logic captures categorical and syllogistic relations in the Aristotelian tradition, expressed through relations of inclusion and exclusion between terms. First-order logic provides greater expressive power by introducing variables, predicates, relations, and universal and existential quantifiers. Each schema defines an ordered set of premises and a conclusion that together represent a derivation rule. The library contains 247 schemata: 49 propositional, 77 term, and 121 first-order inference-rule schemata. Representative schemata are shown in Table 1 while a full breakdown is provided in Table 5 in the Appendix.

Derivation-Tree Generation. Given a target formula φ and a library of derivation schemata $\mathcal{R}$, we apply the rules in $\mathcal{R}$ backward to recursively derive a finite sequence of intermediate formulas α_i and a corresponding set of premises Γ . Inspired by the LogicTree method (Wang et al., 2025), we construct a derivation tree through backward chaining. Starting from a sampled goal φ , we recursively apply compatible derivation rule schemata backwards for at most L steps. The leaves of the resulting tree constitute the premise set Γ , and its root remains the final goal φ . Further dataset details are provided in the Appendix D.

Example. An example is given in Figure 2, where $Q \vee P$ is selected as the target formula φ . Through backward chaining, the generator identifies constructive dilemma as an applicable inference rule and expands the target into three premises: $G \rightarrow Q$, $F \rightarrow P$, and $G \vee F$. The resulting formulas can then be recursively expanded by applying other schemata until the maximum expansion depth is reached or no applicable rule remains.

Structure-Preserving Symbol Renaming. To increase the diversity of the training data and prevent the model from overfitting to simple surface-level patterns, we generate n structure-preserving variants of each derivation tree. Each variant applies injective random renaming to the symbols throughout the tree while preserving the derivations topology, rule sequence, and logical relations.

Next-Step Derivation Construction. Each derivation is finally expanded into K next-step prediction examples. At derivation step i , the input contains the complete set of premises Γ , the previously derived prefix $(y_1, \dots, y_{i-1})$, and the final goal φ . The prediction target is the next valid

derivation step y_i . The serialized derivations are then adopted for Logic-PPT (Section 2.2).

3 Experimental Setup

Baselines. We compare *Logic-PPT* against two established symbolic PPT methods using a mixture of procedurally generated algorithmic data (i.e., *Alg-PPT*), including SET, SORT, and UNION (Jiang et al., 2026), and a mixture of formal language data (i.e., *Formal-PPT*) combining DYCK and SHUFFLE-DYCK (Hu et al., 2025) (see Figure 2, left). Examples for each class of PPT method are interleaved using balanced round-robin sampling. All PPT methods use an approximately equivalent budget of 2.5B tokens. In addition, we compare our method against a randomly initialized baseline undergoing only language pretraining (i.e., *PT*).

Model Architecture. For all configurations, we use a 14-layer Qwen3 architecture (Team, 2025) as the Transformer backbone. Following the setup of Yamaguchi et al. (2026), we replace the Qwen3 tokenizer with the 32K Mistral tokenizer (Jiang et al., 2023) to reduce the compute cost. The resulting model has a 220M-parameter Transformer backbone and 254M parameters in total.

Language Pretraining. After PPT and weight transfer, all parameters remain trainable during pretraining on approximately 100B FineWeb-Edu tokens (Lozhkov et al., 2024). We use LLaMA Factory (Zheng et al., 2024) with a sequence length of 2,048, a batch size of 96, and checkpoints saved every 2,000 steps to track training progress. Additional implementation and training details, as well as hyperparameters are provided in Table 2, 3, and 6 in the Appendix.

Evaluating Language and Skill Acquisition. We use perplexity on a held-out test set of FineWeb-Edu to evaluate general language modeling capabilities. In addition, to track skill acquisition and emergence during language pretraining, we evaluate the models on BLiMP (Warstadt et al., 2020) and on the elemental tasks by Liu et al. (2026).

Following Liu et al. (2026), we retain the elemental tasks for which at least one PPT configuration achieves 80% accuracy during language pretraining within the 100B token budget. This leads to a total of 17 elemental tasks spanning emergent linguistic capabilities such as string operations, morphology, reading comprehension, and compositional tasks (Table 8 in the Appendix).

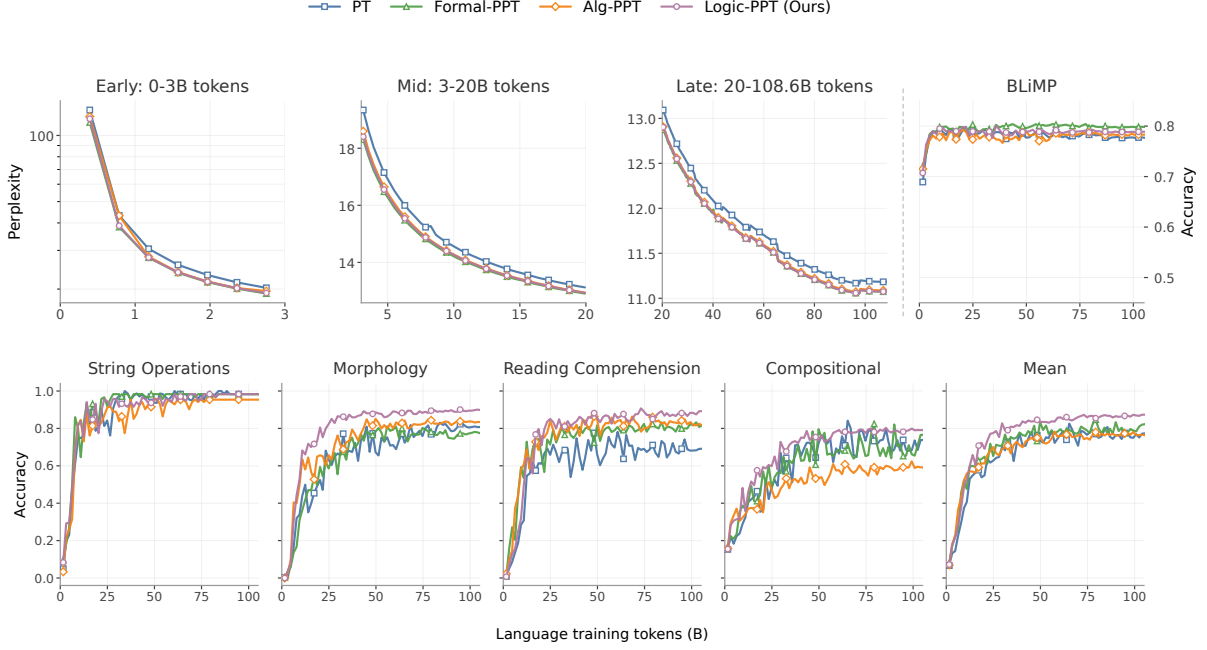


Figure 3: Behavioral trajectories over the 100B-token natural language pretraining run. **Top:** Language modeling perplexity across three training phases, alongside overall BLiMP accuracy. For BLiMP, Formal-PPT performs slightly better than the other variants, but all four saturate quickly and stay close together. **Bottom:** Mean accuracy trajectories across elemental tasks (Liu et al., 2026), shown as an overall mean and broken down by category. Logic-PPT shows substantial improvements on Morphology, Reading Comprehension, and Compositional tasks.

4 Results

4.1 Language Acquisition

As shown in Figure 3, language modeling perplexity evolves through three training phases. During the early phase (0–3B), the perplexity decreases rapidly under all four configurations. Logic-PPT yields a faster reduction than the PT baseline, producing a clear gap within the first 1B tokens. This advantage persists throughout both the middle (3–20B) and late (20–108.6B) phases. At the final checkpoint, the PT baseline reaches a perplexity of 11.183, compared with 11.088 for Alg-PPT (a 0.85% reduction) and 11.074 for both Formal-PPT and Logic-PPT (a 0.97% reduction). These results show that the language modeling advantage of PPT emerges early and remains stable throughout extended natural language training. At the same time, we found that perplexity alone is insufficient to fully characterize the differences between symbolic PPT approaches, with different methods exhibiting comparable trajectories and final perplexity values.

4.2 Skill Acquisition

Analysing the results on BLiMP, we observe that all four models improve rapidly early in training, reaching approximately 80% accuracy within

10-25B tokens, after which performance largely plateaus, with no further gains over the remaining 80B+ tokens. This suggests that, similarly to perplexity, methods relying solely on BLiMP provide little insight into the emergence of more complex capabilities. Despite this, results show that Logic-PPT outperforms the PT baseline and is competitive with alternative symbolic PPT methods.

Performance on the elemental tasks continues to improve throughout the full 100B-token training run, while the separation among training conditions persists, providing deeper insight into skill acquisition dynamics. Across these tasks, the PPT models outperform the PT baseline, indicating that symbolic PPT improves the speed and strength of skill acquisition.

Logic-PPT, in particular, achieves the 80% accuracy threshold earlier than the other configurations. By the final checkpoint, the overall mean accuracy across all 17 tasks is 87.4% for Logic-PPT, versus 80.3% for Formal-PPT, 77.3% for PT, and 76.8% for Alg-PPT. The gains are particularly clear in Morphology, Reading Comprehension, and Compositional tasks, where Logic-PPT outperforms the strongest competing condition by 5.5–6.7% accuracy points. Further details are in Appendix K.

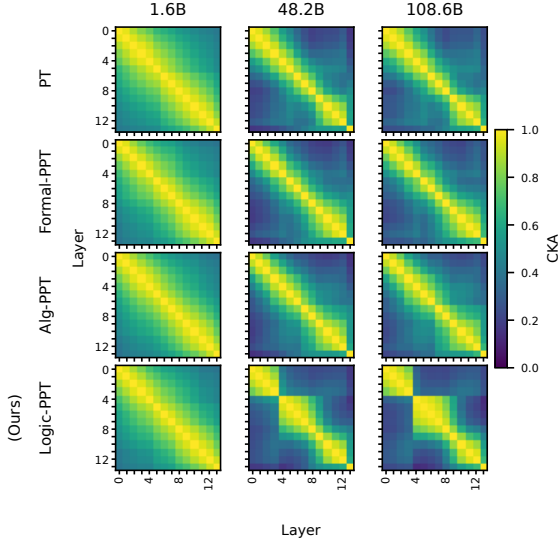


Figure 4: Self layer-to-layer CKA (each model compared against itself) at three checkpoints. Rows are training conditions (PT, Formal-PPT, Alg-PPT, Logic-PPT (Ours)); columns are checkpoints. All four models initially exhibit uniformly high similarity and develop block-diagonal structure as training progresses. However, Logic-PPT exhibits more clear blocks, with lower CKA at the boundary layers.

5 Mechanistic Analysis

We analyze model representational dynamics throughout language PT in both activation and weight space. For the activation-space analysis, we use a held-out subset of FineWeb-Edu containing 15K samples and extract the hidden state of the final non-padding token at each layer. We first use centered kernel alignment (CKA) (Kornblith et al., 2019) to measure activation similarity both across layers within the same model and between models trained under different PPT configuration. We further compute RankMe (Garrido et al., 2023) and the spectral decay (Agrawal et al., 2022) from the centered covariance eigenspectrum of these activations to characterize the geometry and effective complexity. Following Li et al. (2025), we use these metrics to characterize changes in the models’ training dynamics. For the weight-space analysis, we compute the stable rank of the model weight matrices to measure how concentrated or distributed their singular-value spectra are. Formal definitions of all metrics are provided in Appendix A.

5.1 Activation Space

CKA Analysis. Figure 4 shows that within-model layer-wise CKA becomes increasingly struc-

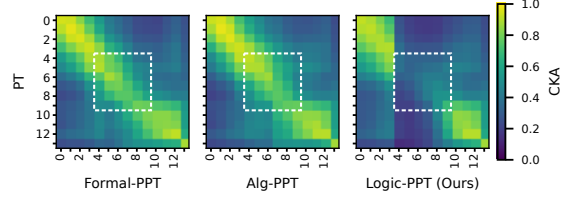


Figure 5: Layer-to-layer CKA between baseline PT and each pre-pretraining (PPT) model, computed at the final checkpoint (108.6B). The y -axis indexes PT’s layers; the x -axis indexes the PPT model’s layers. Logic-PPT exhibits lower similarity with PT’s activations, particularly in the middle layers (layers 4–9, marked by the dashed box).

tured during language PT. Early checkpoints exhibit high similarity across many layer pairs, whereas later checkpoints show lower off-diagonal CKA and clearer block structure, indicating increasing layer differentiation and the formation of distinct representational stages. Logic-PPT produces the clearest block structure, particularly in the middle and upper layers, suggesting a more differentiated and modular organization that persists after extensive language training. Consistent with this finding, pairwise CKA at the final checkpoint (Figure 5) shows lower similarity between the Logic-PPT and the PT in the middle layers, indicating that Logic-PPT induces a distinct and persistent activation space organization that fundamentally differs from other symbolic PPT configurations.

RankMe and Spectral Decay. The CKA analysis reveals clear layer-wise representational differences, motivating a closer examination of activation space geometry and effective complexity. Figure 6 shows the trajectories of RankMe and the spectral decay in three layer-groups. We observe that early in training RankMe increases while the spectral decay decreases, indicating that variance becomes distributed across a broader set of representational directions. Later, this pattern reverses: RankMe decreases and spectral decay increases, showing that variance is progressively concentrated into fewer dominant directions. Although the transition occurs at different times and with different magnitudes across layers, the overall pattern is consistent with the *entropy-seeking* and *compression-seeking* phases described by Li et al. (2025). The effects of PPT become most pronounced during this *compression-seeking* phase. Logic-PPT produces the lowest RankMe in this phase and the

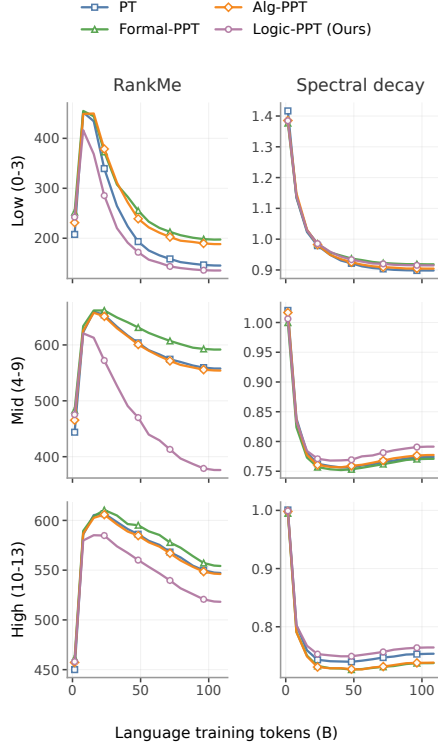


Figure 6: Layer-group trajectories of RankMe and the spectral decay. During the middle and later stages of language PT, Logic-PPT shows much lower RankMe and a higher spectral decay, indicating more compact and lower-dimensional representations.

highest spectral decay in the middle and upper layers, indicating a more compact and anisotropic representation space than the other training configurations. Rather than merely reducing representational dimensionality, Logic-PPT appears to promote a more efficient form of spectral concentration, organizing representations around a smaller set of directions. This interpretation is consistent with our CKA results, which reveal a clearer layer-wise block structure under Logic-PPT.

5.2 Weight Space

A complementary analysis of the weight trajectory in Figure 7 shows that stable rank drops sharply during the early stage of training and then gradually plateaus. The PPT models generally begin with a lower stable rank than the PT baseline, suggesting that symbolic PPT already biases the attention weights toward a more structured and lower-dimensional spectrum before language exposure.

To further analyse this, Figure 8 tracks the stable rank of the attention and MLP weight matrices over the PPT stage. Here, under Logic-PPT, stable rank contracts continually throughout training, for both

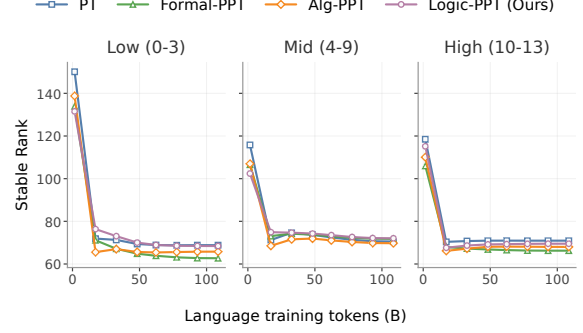


Figure 7: Stable rank trajectories of attention weight matrices across layer groups during language PT. Across all layer groups, PT starts with a higher stable rank than the PPT configurations.

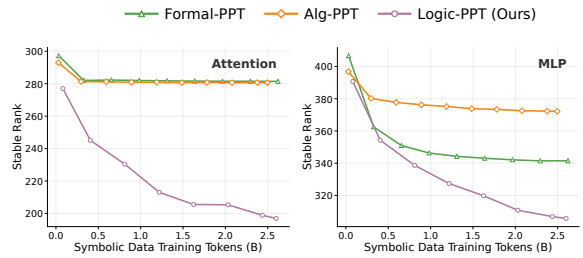


Figure 8: Stable rank of the attention and MLP weight matrices during symbolic PPT. Under Logic-PPT, stable rank continues to decrease, whereas the other PPT conditions plateau.

attention and MLP, with no clear plateau. The other symbolic PPT methods, by contrast, show a much weaker effect, in which stable rank drops only in the earliest training steps and quickly flattens out, remaining largely unchanged for the remainder of training. These results suggest that formal derivation data indeed imposes a stronger and more persistent structural bias on the model’s weights than the formal-language or algorithmic objectives.

6 Model Compression

Prior work using CKA has shown that highly similar upper-layer representations permit some layers to be removed with minimal performance degradation in fine-tuned Transformers (Phang et al., 2021). Although their experiments concern layer truncation rather than weight pruning, they establish a connection between representational organization and model compressibility. Motivated by this connection, together with our representational and spectral analyses, we hypothesize that Logic-PPT induces a more structured and robust computational organization, enabling the model to

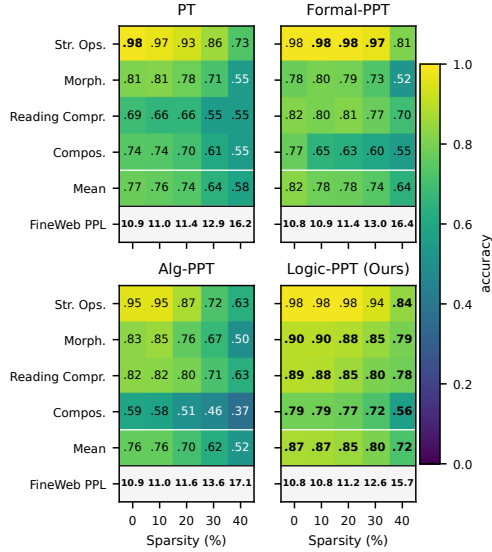


Figure 9: Mean accuracy under Wanda pruning across four elemental task categories, and FineWeb-Edu perplexity under Wanda pruning. Logic-PPT (Ours) performs robustly under compression, retaining the highest accuracy across nearly all categories and sparsity levels.

better preserve its capabilities under weight pruning. To test this, we apply Wanda pruning (Sun et al., 2023) at sparsity levels up to 40% to models trained under different PPT conditions. Wanda scores weights using both parameter magnitude and input activation strength. We evaluate pruning robustness on the elemental tasks and FineWeb-Edu perplexity. Figure 9 shows that Logic-PPT retains the highest mean accuracy across the four task categories. At 40% sparsity, Logic-PPT mean accuracy decreases from 0.87 to 0.745, a relative drop of 14.4%, compared with drops from 24.7% for the PT baseline, 22.0% for Formal-PPT, and 31.6% for Alg-PPT. Logic-PPT also exhibits the smallest increase in FineWeb-Edu perplexity under pruning. Importantly, this robustness advantage does not arise from PPT in general. Formal-PPT and Alg-PPT do not exhibit the same degree of resistance to pruning, with Alg-PPT suffering the largest performance degradation.

7 Related Work

Symbolic and Procedural Pre-training. Recent work explores whether structured synthetic data can provide useful inductive biases before language pretraining (Mita et al., 2026; Guo et al., 2026; Lee et al., 2026a; Shinnick et al., 2026; Wu et al., 2022). Hu et al. (2025) show that pre-training on formal languages improves

language-modeling loss and linguistic generalization, while Jiang et al. (2026) extend this approach to procedurally generated formal and algorithmic data. Despite these promising results, prior work has primarily assessed pre-training through language-modelling loss and downstream accuracy, often at relatively small training scales.

Logic Data Generation. Logic data has been widely used for model evaluation and post-training (Liu et al., 2025; Morishita et al., 2024; Lacombe et al., 2026; Xie et al., 2025; Abdin et al., 2024; Tan et al., 2025; Kim et al., 2025; Lee et al., 2026b; Cao et al., 2026). For example, LogiConBench constructs controllable logical graphs to evaluate model consistency (Chen et al., 2026), while LogicTree generates instantiated multi-step reasoning problems with diverse structures and contexts (Wang et al., 2025). In contrast, we use formal derivation before language pretraining to shape representations and inductive biases for broader language learning, rather than to teach specific logic tasks.

Pretraining Dynamics and Mechanistic Analyses. Dense checkpoint suites, such as Pythia (Biderman et al., 2023) and OLMo (OLMo et al., 2024), have enabled researchers to study how language-model capabilities and internal structures evolve throughout pretraining, rather than evaluating only the final checkpoint. At the behavioural level, language models exhibit broadly consistent skill-acquisition trajectories across random seeds, model families, and training mixtures (Evanson et al., 2023; Liu et al., 2026). Mechanistic studies reveal systematic changes in representation and weight geometry (Kulkarni et al., 2026; Yu et al., 2026; Yunis et al., 2024). Li et al. (2025) identify warm-up, entropy-seeking, and compression-seeking phases using RankMe and spectral decay, while related work tracks effective rank and spectral structure in Transformer weights (Kulkarni et al., 2026). However, these tools have not yet been used to examine how pre-training reshapes subsequent natural language learning dynamics and internal model organization.

8 Conclusion

We investigated how symbolic logic can shape natural language pretraining in LMs, examining its effects from behavioral, mechanistic, and interventional perspectives. Behaviorally, Logic-PPT accel-

erates the acquisition of linguistic skills, reaching strong task accuracy with substantially fewer natural language PT tokens. Mechanistically, these gains are accompanied by persistent changes in model organization: Logic-PPT develops a more modular, compact, and lower-rank representation space that emerges early and persists throughout training. As an intervention, this altered organization translates into greater robustness under pruning, showing that the structure induced by formal logical derivations yields models that are not only more data-efficient but also more compressible.

Limitations

Model Scale and Data. Due to computational constraints, we restrict our experiments to a Qwen architecture with approximately 254M parameters. Following a computationally feasible academic setting like work (Yamaguchi et al., 2026), we train each model from scratch on approximately 100B natural language tokens.

Single Seed. The main behavioral, representational, and pruning results are based on a single 100B-token training run for each condition. Repeating all pretraining configurations was prohibitive within the constraints of the available academic computing infrastructure. To partially assess seed sensitivity, we train each transferred model with three different initialization seeds for 10B natural language tokens (see Appendix F). The resulting self-layer CKA matrices show consistently high correlations across seeds, indicating that the observed layer-wise representational organization is stable at this stage of training.

Evaluation Scope. Our evaluation primarily focuses on capability acquisition during language pretraining, using the elemental tasks and BLiMP, together with analyses of representation geometry and pruning robustness. We do not evaluate whether the benefits of Logic-PPT persist after supervised fine-tuning or transfer to a broader range of downstream tasks, such as natural language reasoning, question answering, or instruction following. Future work should examine whether the observed pretraining advantages translate into consistent improvements under different post-training and downstream evaluation settings.

Hyperparameter Selection. We approximately match the symbolic PPT conditions by the total

number of observed tokens. However, their sequence lengths are task-specific and follow the configurations adopted in prior work, reflecting differences in the structure and serialization of the symbolic tasks.

Language Coverage. Our analysis is limited primarily to English-language pretraining and evaluation. We therefore do not test whether symbolic PPT improves multilingual transfer or whether the same skill-acquisition patterns hold across languages with different morphology, syntax, or writing systems.

Acknowledgments

We acknowledge the University of Sheffield IT Services for providing access to the Stanage high-performance computing facilities. We also acknowledge the Isambard-AI National AI Research Resource (AIRR), operated by the University of Bristol, for providing computational resources under Award ID 0261-5075-3967-1. This work was further supported by computational resources awarded by the EuroHPC Joint Undertaking under Project ID EHPC-DEV-2026D05-083 through access to the Leonardo supercomputer, hosted by CINECA in Italy.

References

- Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, and 8 others. 2024. [Phi-4 technical report](#). *Preprint*, arXiv:2412.08905.
- Kumar K Agrawal, Arnab Kumar Mondal, Arna Ghosh, and Blake Richards. 2022. [\alpha-req : Assessing representation quality in self-supervised learning by measuring eigenspectrum decay](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 17626–17638. Curran Associates, Inc.
- Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O’Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning, ICML’23*. JMLR.org.
- Davis Brown, Charles Godfrey, Nicholas Konz, Jonathan Tu, and Henry Kvinge. 2023. [Understand-](#)

- ing the inner-workings of language models through representation dissimilarity. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6543–6558, Singapore. Association for Computational Linguistics.
- Mingzi Cao, Xingwei Tan, Mahmud Elahi Akhter, Marco Valentino, Maria Liakata, Xi Wang, and Nikolaos Aletras. 2026. [Fundamental reasoning paradigms induce out-of-domain generalization in language models](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 33025–33046, San Diego, California, United States. Association for Computational Linguistics.
- Zheng Chen, Chuan Zhou, Fengxiang Cheng, Yip Tin Po, Fenrong Liu, Yisen Wang, Jiajun Chai, Xiaohan Wang, Guojun Yin, Wei Lin, Bo Li, Haoxuan Li, and Zhouchen Lin. 2026. [Logiconbench: Benchmarking logical consistencies of LLMs](#). In *The Fourteenth International Conference on Learning Representations*.
- Linnea Evanson, Yair Lakretz, and Jean Rémi King. 2023. [Language acquisition: do children and language models follow similar learning stages?](#) In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12205–12218, Toronto, Canada. Association for Computational Linguistics.
- Quentin Garrido, Randall Balestrieri, Laurent Najman, and Yann LeCun. 2023. [Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank](#).
- Xu Guo, Runyu Peng, Jian Tong, Yunhua Zhou, Haijun Lv, Zhihui Lu, and Qipeng Guo. 2026. [Synthetic pre-training improves language model robustness to noisy pre-training data](#). *Preprint*, arXiv:2605.10129.
- Wes Gurnee, Nicholas Sofroniew, Adam Pearce, Mateusz Piotrowski, Isaac Kauvar, Runjin Chen, Anna Soligo, Paul Bogdan, Euan Ong, Rowan Wang, Ben Thompson, David Abrahams, Subhash Kantamneni, Emmanuel Ameisen, Joshua Batson, and Jack Lindsey. 2026. [Verbalizable representations form a global workspace in language models](#). *Preprint*, arXiv:2607.15495.
- Gary M Hardegree. 1994. *Symbolic logic: A first course*. McGraw-Hill.
- John Hewitt and Christopher D. Manning. 2019. [A structural probe for finding syntax in word representations](#). 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 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.
- Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. 2025. [Between circuits and Chomsky: Pre-pretraining on formal languages imparts linguistic biases](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9691–9709, Vienna, Austria. Association for Computational Linguistics.
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. *Mistral 7b*. *Preprint*, arXiv:2310.06825.
- Liangze Jiang, Zachary Shinnick, Anton van den Hengel, Hemanth Saratchandran, and Damien Teney. 2026. [Procedural pretraining: Warming up language models with abstract data](#). In *Forty-third International Conference on Machine Learning*.
- Geonhee Kim, Marco Valentino, and Andre Freitas. 2025. [Reasoning circuits in language models: A mechanistic interpretation of syllogistic inference](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10074–10095, Vienna, Austria. Association for Computational Linguistics.
- Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMIR.
- Atharva Kulkarni, Jacob Mitchell Springer, Arjun Subramonian, and Swabha Swayamdipta. 2026. [Disentangling geometry, performance, and training in language models](#). In *Forty-third International Conference on Machine Learning*.
- Valentin Lacombe, Valentin Quesnel, and Damien Sileo. 2026. [Reasoning core: A scalable procedural data generation suite for symbolic pre-training and post-training](#). *Preprint*, arXiv:2603.02208.
- Dan Lee, Seungwook Han, Akarsh Kumar, and Pulkit Agrawal. 2026a. [Training language models via neural cellular automata](#). *Preprint*, arXiv:2603.10055.
- Isabelle Lee, Sarah Liaw, and Dani Yogatama. 2026b. [FOL-traces: Verified first-order logic reasoning traces at scale](#). In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 2181–2203, Rabat, Morocco. Association for Computational Linguistics.
- Melody Zixuan Li, Kumar Krishna Agrawal, Arna Ghosh, Komal Kumar Teru, Adam Santoro, Guillaume Lajoie, and Blake Aaron Richards. 2025. [Tracing the representation geometry of language models from pretraining to post-training](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Emmy Liu, Kaiser Sun, Millicent Li, Isabelle Lee, Lindia Tjuatja, Jen tse Huang, and Graham Neubig. 2026. [What do language models learn and when? the implicit curriculum hypothesis](#). *Preprint*, arXiv:2604.08510.

- Junteng Liu, Yuanxiang Fan, Zhuo Jiang, Han Ding, Yongyi Hu, Chi Zhang, Yiqi Shi, Shitong Weng, Aili Chen, Shiqi Chen, Mozhi Zhang, Pengyu Zhao, and Junxian He. 2025. [Synlogic: Synthesizing verifiable reasoning data at scale for learning logical reasoning and beyond](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. 2024. [Fineweb-edu: the finest collection of educational content](#).
- Masato Mita, Taiga Someya, Ryo Yoshida, and Yohei Oseki. 2026. [Language acquisition device in large language models](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19564–19577, San Diego, California, United States. Association for Computational Linguistics.
- Richard Montague. 2002. *The Proper Treatment of Quantification in Ordinary English*, chapter 1. John Wiley & Sons, Ltd.
- Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. 2024. [Enhancing reasoning capabilities of LLMs via principled synthetic logic corpus](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Team OLMO, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Taffjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2024. [2 olmo 2 furious](#). *Preprint*, arXiv:2501.00656.
- Jason Phang, Haokun Liu, and Samuel R. Bowman. 2021. [Fine-tuned transformers show clusters of similar representations across layers](#). In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 529–538, Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Zachary Shinnick, Liangze Jiang, Hemanth Saratchandran, Damien Teney, and Anton van den Hengel. 2026. Can you learn to see without images? procedural warm-up for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 27439–27448.
- Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.
- Xingwei Tan, Marco Valentino, Mahmud Elahi Akhter, Maria Liakata, and Nikolaos Aletras. 2025. [Enhancing logical reasoning in language models via symbolically-guided Monte Carlo process supervision](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 31886–31900, Suzhou, China. Association for Computational Linguistics.
- Qwen Team. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- Zehao Wang, Lin Yang, Jie Wang, Kehan Wang, Hanzhu Chen, Bin Wang, Jianye HAO, Defu Lian, Bin Li, and Enhong Chen. 2025. [Logictree: Improving complex reasoning of LLMs via instantiated multi-step synthetic logical data](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. [Blimp: The benchmark of linguistic minimal pairs for english](#). *Transactions of the Association for Computational Linguistics*, 8:377–392.
- Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. [What do RNN language models learn about filler–gap dependencies?](#) In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221, Brussels, Belgium. Association for Computational Linguistics.
- Yuhuai Wu, Felix Li, and Percy S Liang. 2022. Insights into pre-training via simpler synthetic tasks. *Advances in Neural Information Processing Systems*, 35:21844–21857.
- Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhi-rong Wu, and Chong Luo. 2025. [Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning](#). *Preprint*, arXiv:2502.14768.
- Atsuki Yamaguchi, Maggie Mi, and Nikolaos Aletras. 2026. [Enhancing linguistic competence of language models through pre-training with language learning tasks](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 316–336, San Diego, California, United States. Association for Computational Linguistics.
- Junjie Yu, Yue Wang, Zihan Deng, Yan Zhu, Wenxiao Ma, and Quanying Liu. 2026. [Pretraining induces a reusable spectral basis for downstream task adaptation](#). *Preprint*, arXiv:2605.07302.
- David Yunis, Kumar Kshitij Patel, Samuel Wheeler, Pedro Savarese, Gal Vardi, Karen Livescu, Michael Maire, and Matthew R. Walter. 2024. [Approaching deep learning through the spectral dynamics of weights](#). *Preprint*, arXiv:2408.11804.
- Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyuan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. [Llamafactory: Unified efficient fine-tuning of 100+ language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

A Mechanistic Analysis Tools

CKA. To quantify representational similarity between layers, we use linear centered kernel alignment (CKA) (Kornblith et al., 2019). CKA is invariant to orthogonal transformations and isotropic rescaling, making it well suited for comparing representations across language-model layers (Phang et al., 2021; Brown et al., 2023; Gurnee et al., 2026). Let $X_i \in \mathbb{R}^{n \times d_i}$ and $X_j \in \mathbb{R}^{n \times d_j}$ denote the activation matrices extracted from layers i and j , where n is the number of examples and d_i and d_j are the corresponding hidden dimensions.

Linear CKA is defined as

$$\text{CKA}(X_i, X_j) = \frac{\|\tilde{X}_i^\top \tilde{X}_j\|_F^2}{\|\tilde{X}_i^\top \tilde{X}_i\|_F \|\tilde{X}_j^\top \tilde{X}_j\|_F}, \quad (4)$$

where $\tilde{X}_i = HX_i$ and $\tilde{X}_j = HX_j$ are column-centered activation matrices, $H = I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$ is the centering matrix, and $\|\cdot\|_F$ denotes the Frobenius norm.

RankMe and Spectral Decay Rate. Following Li et al. (2025), we use RankMe (Garrido et al., 2023) and the spectral decay rate (Agrawal et al., 2022) to characterize how different pre-pretraining conditions affect the complexity of the representation space. For each checkpoint and layer, we collect a representation matrix $X \in \mathbb{R}^{n \times d}$ from last-token activations on a 15K-example subset of FineWeb-Edu, where n is the number of examples and d is the hidden dimension. We center the features across examples to obtain X_c and compute the empirical covariance matrix $\hat{\Sigma} = \frac{1}{n}X_c^\top X_c$. Let λ_i denote the eigenvalues of $\hat{\Sigma}$. To compute RankMe, we normalize the eigenvalues as $p_i = \frac{\lambda_i}{\sum_j \lambda_j}$ and define RankMe as the exponential entropy of this normalized spectrum:

$$\text{RankMe}(X) = \exp\left(-\sum_i p_i \log p_i\right). \quad (5)$$

To measure spectral concentration more directly, we compute the spectral decay rate by fitting a power law to the same covariance eigenspectrum, $\lambda_i \propto i^{-\alpha}$. Equivalently, we fit the linear regression

$$\log \lambda_i = c - \alpha \log i. \quad (6)$$

The fitted negative slope gives the spectral decay rate α . A larger α indicates faster spectral decay,

meaning that variance is more strongly concentrated along the leading principal directions. Conversely, a smaller α indicates a flatter spectrum and a more uniform spread of variance across dimensions.

A higher RankMe indicates that representational variance is distributed more evenly across dimensions, corresponding to a higher effective dimensionality. Conversely, a lower RankMe indicates that variance is concentrated in fewer dominant directions, resulting in a more anisotropic and lower-dimensional representation. A larger spectral decay rate indicates a more rapidly decaying spectrum and stronger concentration along the leading principal directions, whereas a smaller spectral decay rate corresponds to a flatter spectrum.

Stable Rank. To complement the representation-space analysis, we use stable rank to characterize the effective dimensionality of model weight matrices. For a weight matrix W with singular values $\{\sigma_i\}$, stable rank is defined as

$$\text{srank}(W) = \frac{\|W\|_F^2}{\|W\|_2^2} = \frac{\sum_i \sigma_i^2}{\sigma_1^2}. \quad (7)$$

Higher stable rank indicates that the weight energy is distributed across more singular directions, while lower stable rank indicates that the matrix is dominated by a few leading directions. We use this metric to test whether different pre-pretraining conditions lead to different levels of spectral concentration in the learned weights.

B Pruning

For a weight matrix $\mathbf{W} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ and input activations $\mathbf{X} \in \mathbb{R}^{NL \times C_{\text{in}}}$, Wanda assigns each weight the importance score

$$S_{ij} = |W_{ij}| \|\mathbf{X}_j\|_2. \quad (8)$$

Here, $\mathbf{X}_j$ denotes the activation of the j -th input channel across all tokens.

C Model Architecture

Table 2 summarizes the model architecture used in our experiments.

D Derivation generation details

Inspired by LogicTree (Wang et al., 2025), we represent each reasoning instance as a proof tree constructed through backward chaining. For each source example, we select a goal from a predefined

Table 2: Model architecture used in the experiments.

Hyperparameter	Qwen3-14L
<i>Model architecture</i>	
Hidden size	1,024
Intermediate size	3,072
Number of hidden layers	14
Number of attention heads	16
Number of key-value heads	8
Attention head dimension	128
Maximum position embeddings	40,960
Maximum window layers	14
RoPE theta	1,000,000
RMSNorm epsilon	10^{-6}
Attention dropout	0.0
Hidden activation	SiLU
Transformer parameters	220M
Total parameters	254M

pool consisting of manually specified seed formulas and the conclusions of the available inference-rule schemata. The selected formula becomes the root of the derivation tree.

Starting from the root, we perform at most L backward-expansion steps. At each step, the generator examines the current leaves and identifies all leaf-rule pairs for which the conclusion of the rule can be structurally unified with the formula at the leaf. To encourage broad coverage of the rule library, it preferentially selects among the applicable rules with the lowest cumulative usage counts. One of the corresponding leaf-rule pairs is then sampled using a seeded pseudo-random number generator. The substitution obtained through unification is applied to the rule, and the selected leaf is replaced by the resulting instantiated premises. Schema variables that remain unbound after unification are assigned fresh symbols that do not already occur in the tree, preventing unintended symbol collisions.

Expansion terminates after L rule applications or earlier if no current leaf admits a valid backward expansion. The leaves of the completed tree form the premise set $\Gamma = \{p_1, \dots, p_m\}$, while the root remains the final goal φ . We then traverse the expanded nodes in post-order to obtain a valid leaf-to-root proof sequence

$$Y = (y_1, \dots, y_K), \quad K \leq L, \quad y_K = \varphi. \quad (9)$$

Because each node is visited only after its children, every step y_i is derivable from the initial premises and the preceding steps $y_1, \dots, y_{i-1}$. Thus, although the tree is constructed backward from the goal, its post-order traversal yields a valid forward

Table 3: Hyperparameters of language-pretraining for the 100B-token experiments.

Hyperparameter	Qwen3-14L
<i>Language pretraining</i>	
Training corpus	FineWeb-Edu
Training tokens	$\sim 100\text{B}$
Tokenizer	Mistral
Vocabulary size	32,768
Sequence length	2,048
Packing	Yes
Batch size per GPU	96
GPU	$4 \times \text{NVIDIA GH200}$
Gradient accumulation steps	1
Training steps	127,160
Maximum learning rate	3×10^{-4}
Learning-rate scheduler	Cosine
Warmup steps	2,000
Optimizer	AdamW
Adam ϵ	10^{-8}
Adam β_1	0.9
Adam β_2	0.999
Gradient clipping	1.0
Weight decay	0.01
Training precision	BF16
Initialization seed	42

derivation from Γ to φ . We set the maximum number of backward-expansion steps to $L = 10$ and generate $n = 5$ variants for each derivation tree.

E Pretraining Details

After symbolic pre-pretraining, we transfer the Transformer blocks to the language-pretraining stage and reinitialize the token embeddings and language-modeling head to match the natural-language tokenizer. All model parameters remain trainable during language pretraining. We train all model variants on approximately 100B tokens from FineWeb-Edu using the same architecture, data, and optimization configuration; the no-pre-pretraining baseline is trained under identical conditions. Table 3 summarizes the language-pretraining hyperparameters.

F Seed Stability of CKA

Cross-seed agreement. For a given model, let $\mathbf{c}^{(s)} \in \mathbb{R}^n$ denote the vector of off-diagonal upper-triangular entries of its self-layer CKA matrix under seed s . For each of the three seed pairs (s_a, s_b) , we compute

$$\text{MeanAbsDiff}(s_a, s_b) = \frac{1}{n} \sum_{k=1}^n \left| c_k^{(s_a)} - c_k^{(s_b)} \right|,$$

$$\text{PearsonR}(s_a, s_b) = \text{corr}(\mathbf{c}^{(s_a)}, \mathbf{c}^{(s_b)}).$$

Model	Mean Abs. Diff.	Mean Pearson r
PT	0.0138	0.9955
Formal-PPT	0.0201	0.9942
Alg-PPT	0.0171	0.9941
Logic-PPT	0.0182	0.9926

Table 4: Seed stability of self-layer CKA at the 10B-token checkpoint.

The former measures the absolute magnitude of seed-to-seed drift in CKA values; the latter measures whether the relative pattern of which layer pairs are more or less similar is preserved across seeds.

To examine the effect of random seeds on model representations, we initialize the transferred model with three different seeds and train each run for 10B tokens. As shown in Table 4, the mean absolute difference between CKA matrices is small for all models. At the same time, the mean Pearson correlation between vectorized off-diagonal CKA entries is consistently above 0.99. This indicates that different seeds produce very similar layer-to-layer representational similarity patterns.

G Licenses

We adopt the Qwen3 architecture but initialize all model parameters from scratch; no pretrained Qwen3 weights are used. The Mistral-7B-Instruct v0.3 tokenizer is released under the Apache License 2.0. FineWeb-Edu is released under the Open Data Commons Attribution License (ODC-By) v1.0, and its use is also subject to the Common Crawl Terms of Use. The elemental tasks repository is released under the MIT License, while BLiMP is distributed under the Creative Commons Attribution 4.0 License.

H Logic schemata

We organize our formal-derivation schema library into three broad classes: propositional logic, term logic, and first-order logic. Within each class, schemata are further grouped into rule families according to the logical operations or inference patterns they instantiate. As summarized in Table 5, the library contains 49 propositional, 77 term-logic, and 121 first-order schemata, for a total of 247 inference-rule schemata.

Table 5: Counts of derivation schemata in the library by logic type and rule family.

Logic type	Rule family	Count
Propositional	De Morgan	4
	Dilemma	4
	Distribution	4
	Resolution	3
	Absorption	2
	Association	2
	Biconditional elimination	2
	Classical laws	2
	Commutation	2
	Conjunction elimination	2
	Disjunctive syllogism	2
	Double negation	2
	Idempotence	2
	Material equivalence	2
	Material implication	2
	Transposition	2
	Addition	1
	Biconditional introduction	1
	Conjunction introduction	1
	Excluded middle	1
	Exportation	1
	Hypothetical syllogism	1
	Importation	1
	Modus ponens	1
	Modus tollens	1
	Proof by cases	1
	Total	49
Term	Categorical syllogisms	24
	Complemented-term variants	16
	Syllogistic reductions	15
	Square of opposition	9
	Sorites chains	8
	Derived syllogisms	5
	Total	77
First-order	Syllogism	57
	Scope	14
	Derived patterns	12
	Equality	9
	Distribution	6
	Classical quantifier laws	4
	Quantifier negation	4
	Uniqueness	4
	Vacuous quantification	4
	Quantifier rules	3
	Quantifier commutation	2
	Renaming	2
	Total	121
Overall		247

I Logic Data Samples

I.1 Propositional Logic

Sample 1

$$\begin{aligned}
\Theta &= (\neg X \vee A) \wedge (\neg X \vee B \vee C), \\
\mathcal{P}^{(1)} &= \{G \vee W; W \rightarrow X; G \rightarrow X; \neg\Theta \rightarrow (\neg\Theta \wedge \neg Y); \\
&\quad K \rightarrow \Theta; Y\}, \quad g^{(1)} = A \wedge (B \vee C). \\
x_1^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_1^{(1)} &= W \vee G. \\
x_2^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_2^{(1)} &= X. \\
x_3^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; y_2^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_3^{(1)} &= \neg\Theta \rightarrow \neg Y. \\
x_4^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_3^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_4^{(1)} &= Y \rightarrow \Theta. \\
x_5^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_4^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_5^{(1)} &= Y \vee K. \\
x_6^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_5^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_6^{(1)} &= \Theta. \\
x_7^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_6^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_7^{(1)} &= \neg X \vee (A \wedge (B \vee C)). \\
x_8^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_7^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_8^{(1)} &= A \wedge (B \vee C). \\
x_9^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_8^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_9^{(1)} &= (A \wedge B) \vee (A \wedge C). \\
x_{10}^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_9^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_{10}^{(1)} &= A \wedge (B \vee C).
\end{aligned}$$

Sample 2

$$\begin{aligned}
\mathcal{P}^{(2)} &= \{O; A \rightarrow L; L \rightarrow M; (M \rightarrow B \vee C) \rightarrow D; \\
&\quad D \rightarrow (M \rightarrow B \vee C); D\}, \quad g^{(2)} = A \rightarrow B \vee C. \\
x_1^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_1^{(2)} &= \neg D \rightarrow \neg(M \rightarrow B \vee C).
\end{aligned}$$

$$\begin{aligned}
x_2^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_2^{(2)} &= (M \rightarrow B \vee C) \rightarrow D. \\
x_3^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; y_2^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_3^{(2)} &= D \wedge M \rightarrow B \vee C. \\
x_4^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_3^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_4^{(2)} &= D \rightarrow (M \rightarrow B \vee C). \\
x_5^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_4^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_5^{(2)} &= (M \rightarrow B \vee C) \leftrightarrow D. \\
x_6^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_5^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_6^{(2)} &= M \rightarrow B \vee C. \\
x_7^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_6^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_7^{(2)} &= L \rightarrow B \vee C. \\
x_8^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_7^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_8^{(2)} &= A \rightarrow B \vee C. \\
x_9^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_8^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_9^{(2)} &= O \wedge (A \rightarrow B \vee C). \\
x_{10}^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_9^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_{10}^{(2)} &= A \rightarrow B \vee C.
\end{aligned}$$

Sample 3

$$\mathcal{P}^{(3)} = \{L \wedge (\neg B \vee C); A\}, \quad g^{(3)} = A \wedge (B \rightarrow C).$$

$$\begin{aligned}
x_1^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_1^{(3)} &= (L \wedge \neg B) \vee (L \wedge C). \\
x_2^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_2^{(3)} &= L \wedge (\neg B \vee C). \\
x_3^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; y_2^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_3^{(3)} &= (L \wedge \neg B) \vee (L \wedge C). \\
x_4^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_3^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_4^{(3)} &= ((L \wedge \neg B) \vee L) \wedge ((L \wedge \neg B) \vee C). \\
x_5^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_4^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_5^{(3)} &= (L \wedge \neg B) \vee (L \wedge C).
\end{aligned}$$

$$\begin{aligned}
x_6^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_5^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_6^{(3)} &= L \wedge (\neg B \vee C). \\
x_7^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_6^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_7^{(3)} &= \neg B \vee C. \\
x_8^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_7^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_8^{(3)} &= B \rightarrow C. \\
x_9^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_8^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_9^{(3)} &= (B \rightarrow C) \wedge A. \\
x_{10}^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_9^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_{10}^{(3)} &= A \wedge (B \rightarrow C).
\end{aligned}$$

I.2 Term Logic Sample 1

$$\begin{aligned}
\mathcal{P}^{(1)} &= \{A \models T; T \models V; V \models J; J \models F; F \models \neg O; \neg O \models X; \\
&\quad X \models \neg N; \neg N \models H; H \models Z; Z \models \neg Q; \neg Q \models M; M \models B\}, \\
g^{(1)} &= A \models B.
\end{aligned}$$

$$\begin{aligned}
x_1^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_1^{(1)} &= J \models \neg O. \\
x_2^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_2^{(1)} &= \neg O \models \neg N. \\
x_3^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; y_2^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_3^{(1)} &= \neg O \models H. \\
x_4^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_3^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_4^{(1)} &= Q \models \neg H. \\
x_5^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_4^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_5^{(1)} &= \neg O \models \neg Q. \\
x_6^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_5^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_6^{(1)} &= J \models \neg Q. \\
x_7^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_6^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_7^{(1)} &= J \models M. \\
x_8^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_7^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_8^{(1)} &= V \models M.
\end{aligned}$$

$$\begin{aligned}
x_9^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_8^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_9^{(1)} &= T \models M. \\
x_{10}^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_9^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_{10}^{(1)} &= A \models B.
\end{aligned}$$

Sample 2

$$\begin{aligned}
\mathcal{P}^{(2)} &= \{A \models G; G \models \neg M; \neg M \models R; R \models U; \\
&\quad \neg C \models E; E \models \neg Q; \neg Q \models X; X \models \neg W; \neg W \models \neg U\}, \\
g^{(2)} &= A \not\models \neg C. \\
x_1^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_1^{(2)} &= G \models R. \\
x_2^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_2^{(2)} &= \neg C \models \neg Q. \\
x_3^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; y_2^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_3^{(2)} &= \neg C \models X. \\
x_4^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_3^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_4^{(2)} &= X \models \neg U. \\
x_5^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_4^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_5^{(2)} &= \neg C \models \neg U. \\
x_6^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_5^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_6^{(2)} &= R \models \neg \neg C. \\
x_7^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_6^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_7^{(2)} &= G \models \neg \neg C. \\
x_8^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_7^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_8^{(2)} &= A \models \neg \neg C. \\
x_9^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_8^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_9^{(2)} &= \neg(A \not\models \neg \neg C). \\
x_{10}^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_9^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_{10}^{(2)} &= A \not\models \neg C.
\end{aligned}$$

Sample 3

$$\begin{aligned}
\mathcal{P}^{(3)} &= \{Y \models F; F \models L; A \models \neg L; \neg Y \models D; \\
&\quad D \models K; K \models \neg T; \neg T \models W; W \models B\}, \\
g^{(3)} &= \neg(A \models \neg B).
\end{aligned}$$

$$\begin{aligned}
x_1^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_1^{(3)} &= Y \models L. \\
x_2^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_2^{(3)} &= A \models \neg Y. \\
x_3^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; y_2^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_3^{(3)} &= \neg Y \models K. \\
x_4^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_3^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_4^{(3)} &= \neg Y \models \neg T. \\
x_5^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_4^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_5^{(3)} &= \neg Y \models W. \\
x_6^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_5^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_6^{(3)} &= \neg Y \models B. \\
x_7^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_6^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_7^{(3)} &= A \models B. \\
x_8^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_7^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_8^{(3)} &= \neg(A \models B). \\
x_9^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_8^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_9^{(3)} &= A \models \neg B. \\
x_{10}^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_9^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_{10}^{(3)} &= \neg(A \models \neg B).
\end{aligned}$$

I.3 First-Order Logic

Sample 1

$$\begin{aligned}
\mathcal{P}^{(1)} &= \{\forall x (N(x) \rightarrow K(x)); \forall x (K(x) \rightarrow O(x)); \forall x (O(x) \rightarrow \Psi(x)); \\
&\quad \forall x I(x) \vee \forall x N(x); \forall x \neg I(x)\}, \quad g^{(1)} = \forall x \Psi(x).
\end{aligned}$$

$$\begin{aligned}
x_1^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_1^{(1)} &= \forall x (K(x) \rightarrow O(x)). \\
x_2^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_2^{(1)} &= \forall x (N(x) \rightarrow \Psi(x)). \\
x_3^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; y_2^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_3^{(1)} &= \forall x I(x) \vee \forall x N(x).
\end{aligned}$$

$$\begin{aligned}
x_4^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_3^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_4^{(1)} &= \forall x (I(x) \vee N(x)). \\
x_5^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_4^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_5^{(1)} &= \neg \exists x I(x). \\
x_6^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_5^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_6^{(1)} &= \forall x \neg I(x). \\
x_7^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_6^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_7^{(1)} &= \forall x N(x). \\
x_8^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_7^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_8^{(1)} &= \neg \exists x \neg N(x). \\
x_9^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_8^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_9^{(1)} &= \forall x N(x). \\
x_{10}^{(1)} &= \langle \text{premises} \rangle \mathcal{P}^{(1)} \quad \langle \text{derived} \rangle y_1^{(1)}; \dots; y_9^{(1)} \\
&\quad \langle \text{goal} \rangle g^{(1)} \quad \langle \text{next} \rangle, \\
y_{10}^{(1)} &= \forall x \Psi(x).
\end{aligned}$$

Sample 2

$$\begin{aligned}
\mathcal{P}^{(2)} &= \{\forall x X(x) \vee \forall x \Phi(x); \neg \exists x X(x)\}, \quad g^{(2)} = \neg \exists x \neg \Phi(x). \\
x_1^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_1^{(2)} &= \forall x (X(x) \vee \Phi(x)). \\
x_2^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_2^{(2)} &= \forall x X(x) \vee \Phi(x). \\
x_3^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; y_2^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_3^{(2)} &= \forall x (X(x) \vee \Phi(x)). \\
x_4^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_3^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_4^{(2)} &= \forall x X(x) \vee \Phi(x). \\
x_5^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_4^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_5^{(2)} &= \forall x (X(x) \vee \Phi(x)). \\
x_6^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_5^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_6^{(2)} &= \forall x \neg X(x). \\
x_7^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_6^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_7^{(2)} &= \neg \exists x X(x).
\end{aligned}$$

$$\begin{aligned}
x_8^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_7^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_8^{(2)} &= \forall x \neg X(x). \\
x_9^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_8^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_9^{(2)} &= \forall x \Phi(x). \\
x_{10}^{(2)} &= \langle \text{premises} \rangle \mathcal{P}^{(2)} \quad \langle \text{derived} \rangle y_1^{(2)}; \dots; y_9^{(2)} \\
&\quad \langle \text{goal} \rangle g^{(2)} \quad \langle \text{next} \rangle, \\
y_{10}^{(2)} &= \neg \exists x \neg \Phi(x).
\end{aligned}$$

Sample 3

$$\begin{aligned}
\mathcal{P}^{(3)} &= \{ \forall z (I(z) \rightarrow C(z)); \forall z (C(z) \rightarrow E(z)); \forall z (L(z) \rightarrow I(z)); \\
&\quad \exists z L(z); \forall z \forall r ((E(z) \wedge E(r)) \rightarrow z = r); \\
&\quad c = w; s = w; \forall l (A(l) \rightarrow E(l)); A(c); E(t) \}, \quad g^{(3)} = s = t.
\end{aligned}$$

$$\begin{aligned}
x_1^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle \text{none} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_1^{(3)} &= \exists z I(z). \\
x_2^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_2^{(3)} &= \exists z E(z). \\
x_3^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; y_2^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_3^{(3)} &= \exists! z E(z). \\
x_4^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_3^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_4^{(3)} &= \exists z (E(z) \wedge \forall p (E(p) \rightarrow p = z)). \\
x_5^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_4^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_5^{(3)} &= \exists! z E(z). \\
x_6^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_5^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_6^{(3)} &= w = s. \\
x_7^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_6^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_7^{(3)} &= c = s. \\
x_8^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_7^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_8^{(3)} &= E(c). \\
x_9^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_8^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_9^{(3)} &= E(s). \\
x_{10}^{(3)} &= \langle \text{premises} \rangle \mathcal{P}^{(3)} \quad \langle \text{derived} \rangle y_1^{(3)}; \dots; y_9^{(3)} \\
&\quad \langle \text{goal} \rangle g^{(3)} \quad \langle \text{next} \rangle, \\
y_{10}^{(3)} &= s = t.
\end{aligned}$$

J Pre-training Details

Table 6: Hyperparameters for the three symbolic PPT conditions. All models are initialized from scratch with the same seed.

Hyperparameter	Logic	Algorithmic	Formal Language
Training tasks	Prop./FOL/term logic	Set, sort, union	Dyck, Shuffle-Dyck
Task sampling	Mixed dataset	Balanced round-robin	Balanced round-robin
Tokenizer	Character-level	Task-specific discrete	Task-specific discrete
Vocabulary size	74	231	359
Training examples	6.0M total	6.0M per task	20.5M per task
Maximum sequence length	2,048	128	128
Batch size	256	256	256
Gradient accumulation	1	1	1
Selected checkpoint step	32,000	84,000	80,000
Input tokens observed	2.600B	2.753B	2.621B
Maximum learning rate	1×10^{-4}	5×10^{-5}	5×10^{-5}
Learning-rate schedule	Constant	Constant	Constant
Warmup steps	100	100	100
Optimizer	AdamW	AdamW	AdamW
Adam ϵ	10^{-8}	10^{-8}	10^{-8}
Adam β_1	0.9	0.9	0.9
Adam β_2	0.999	0.999	0.999
Weight decay	0.01	0.1	0.1
Gradient clipping	1.0	1.0	1.0
Training precision	BF16	BF16	BF16
Loss positions	Target only	Target only	All non-final tokens
Initialization seed	42	42	42
GPU	1 $\times$ NVIDIA GH200 (96 GB)		

Pair	ρ	p-value	n
PT vs Formal-PPT	0.800	0.00058	14
PT vs Alg-PPT	0.768	0.00954	10
PT vs Logic-PPT	0.801	0.00057	14
Formal-PPT vs Alg-PPT	0.768	0.00579	11
Formal-PPT vs Logic-PPT	0.557	0.03084	15
Algo-PPT vs Logic-PPT	0.904	0.000022	13

Table 7: Pairwise Spearman correlations between task-emergence orders under different PPT conditions.

K Task Performance

We report full accuracy trajectories on the ElementalTask suite and BLiMP accuracy broken down by linguistic field, to complement the aggregate results discussed above with per-task and per-field detail. Figure 10 shows accuracy versus cumulative language training tokens for the 17 elemental tasks used in our main emergence analysis, grouped into the four skill categories introduced earlier—String Operations, Morphology, Reading Comprehension, and Compositional—for all four model variants (PT, Formal-PPT, Alg-PPT, Logic-PPT). To check that this curated subset is representative rather than cherry-picked, Figures 11–15 report the same trajectories for the full 130-task elemental task suite, split across five panels for legibility; the 17 tasks in Figure 10 are the subset without a clean emergence point, used for the emergence-order analysis. Figure 16 further decomposes the overall BLiMP trajectory by field (Morphology, Syntax, Semantics), showing that the aggregate curves reported earlier mask substantial field-level variation. Table 7 reports the pairwise Spearman rank correlations between task-emergence orders for every pair of models, providing the data underlying the implicit-curriculum discussion.



Figure 10: Accuracy trajectories for the 17 elemental tasks across language training tokens.

Category	Task	# Samples	Example
String Operations (72)	copying	20	Input: gTpigTHK → Output: gTpigTHK
	simple_icl:lowercase	26	Input: B → Output: b
	simple_icl:uppercase	26	Input: b → Output: B
Morphology (344)	simple_icl:present_to_gerund	179	Input: run → Output: running
	simple_icl:singular_to_plural	165	Input: child → Output: children
Reading Comprehension (1081)	coreference:pronoun_hard	20	Input: “The trophy didn’t fit in the suitcase because it was too big.” What was too big? → Output: the trophy
	coreference:pronoun_simple	20	Input: “Alice told Bob that she would be late.” Who does “she” refer to? → Output: Alice
	fact_extraction:extract_entity	20	Input: Alice gave five apples to Bob at the park. Who received the apples? → Output: Bob
	fact_extraction:extract_number	20	Input: John gave 5 apples to Mary on Tuesday. How many apples? → Output: 5
	ignoring_context	1	Input: Some text here. $X = 5$. More text. What is X ? → Output: 5
	ioi_task	1000	Input: Henry and Phil had a lot of fun at the harbor. Henry gave a basket to → Output: Phil
Compositional (3090)	blended_compositions:extract_verify	6	Input: Nora gave 3 apples to Ben; Ben gave 1 apple to Li. Claim: Ben received apples before giving any away. → Output: True
	compositional:extract_verify	6	Input: Ravi arrived after Mina, but before Joel. Claim: Joel arrived before Mina. → Output: False
	compositional:lower_first	971	Input: AFGHANISTAN → Output: a
	compositional:plural_first	165	Input: child → Output: c
	compositional:reverse_last	971	Input: Afghanistan → Output: A
	compositional:upper_first	971	Input: afghanistan → Output: A

Table 8: Elemental tasks grouped by skill category, with the number of evaluation examples and one representative example for each task.

Full Elemental Tasks Accuracy — part 1/5

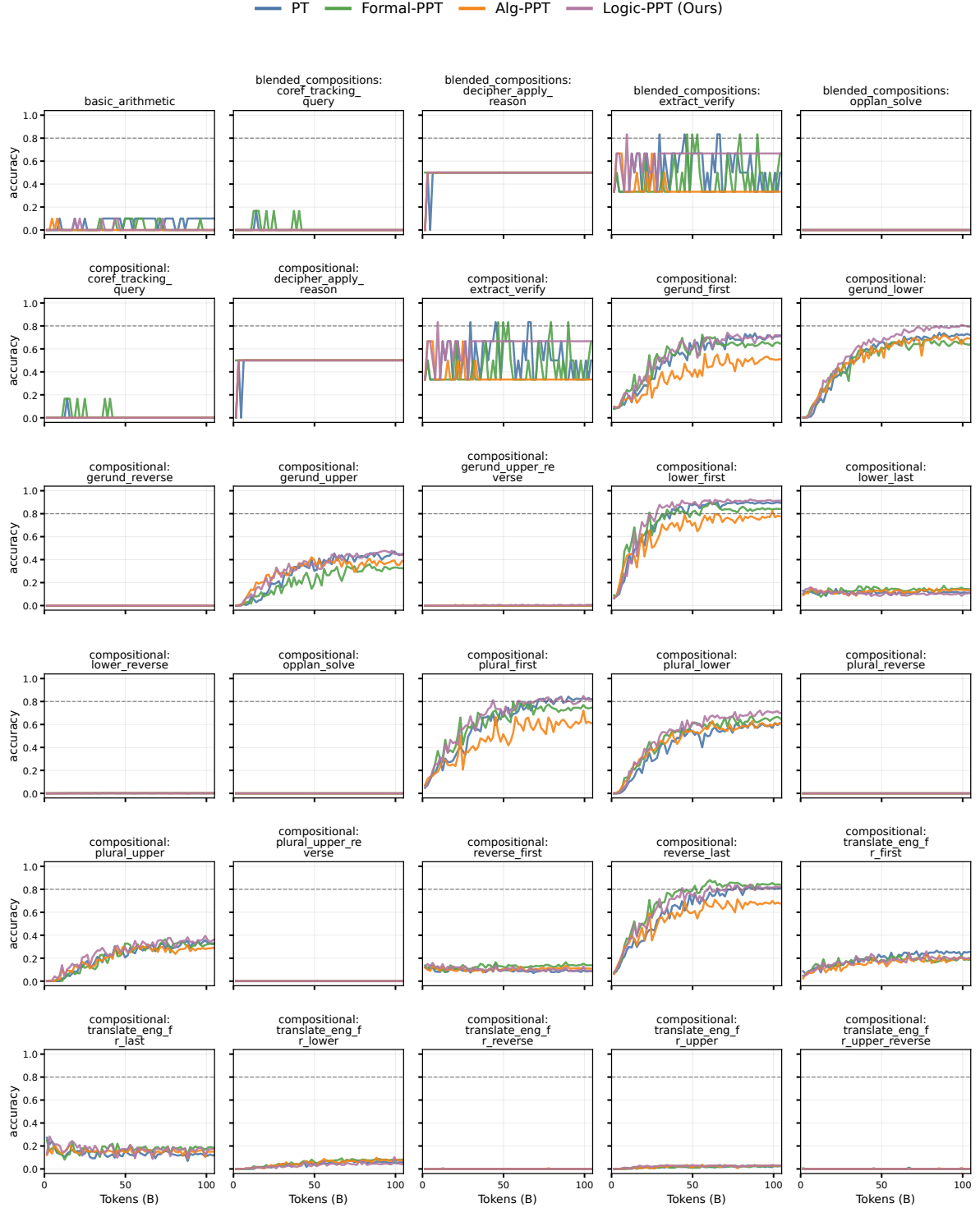


Figure 11: Accuracy trajectories for the first subset of tasks in the full 130-task ElementalTask suite.

Full Elemental Tasks Accuracy — part 2/5

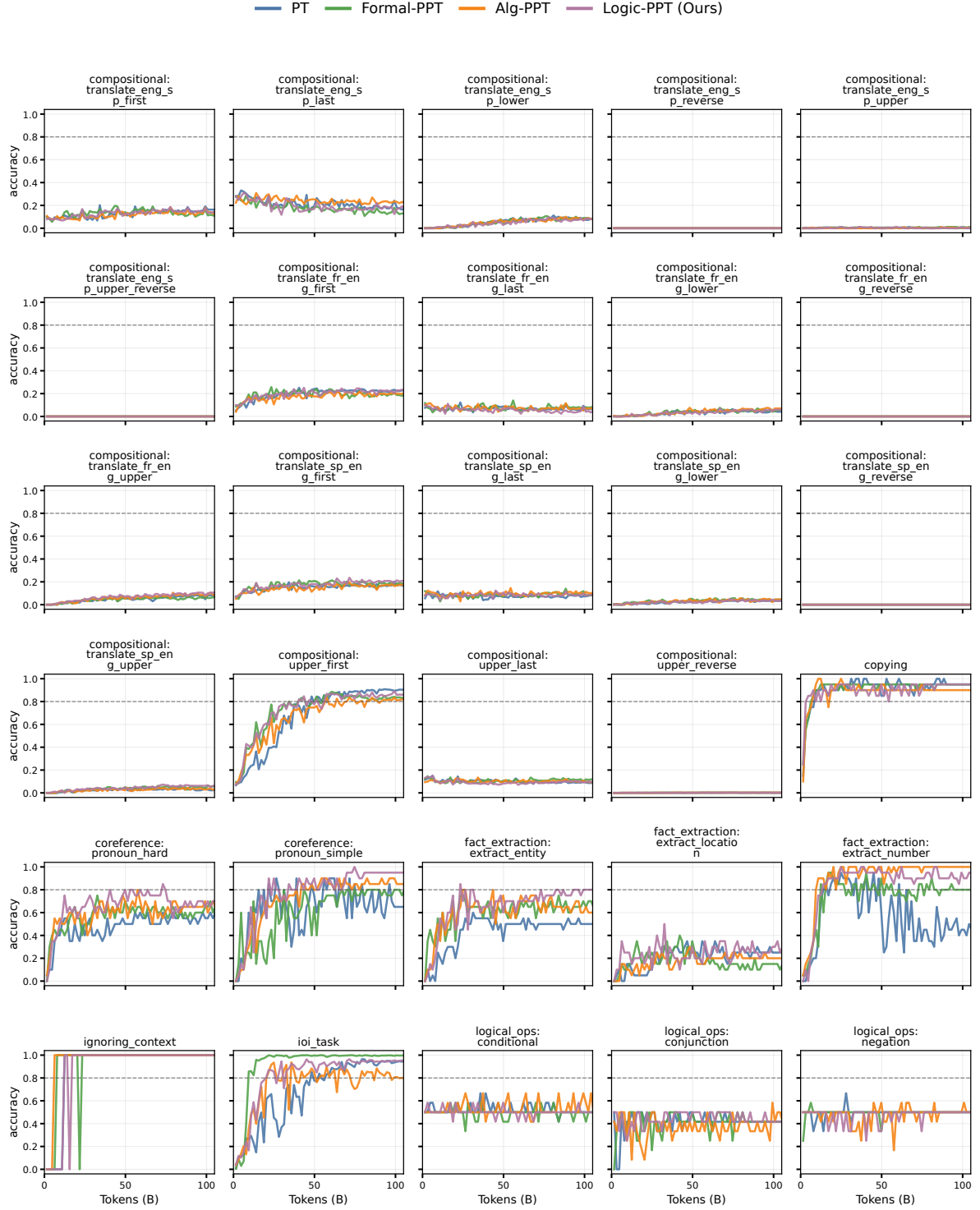


Figure 12: Accuracy trajectories for the second subset of tasks in the full 130-task ElementalTask suite.

Full Elemental Tasks Accuracy — part 3/5

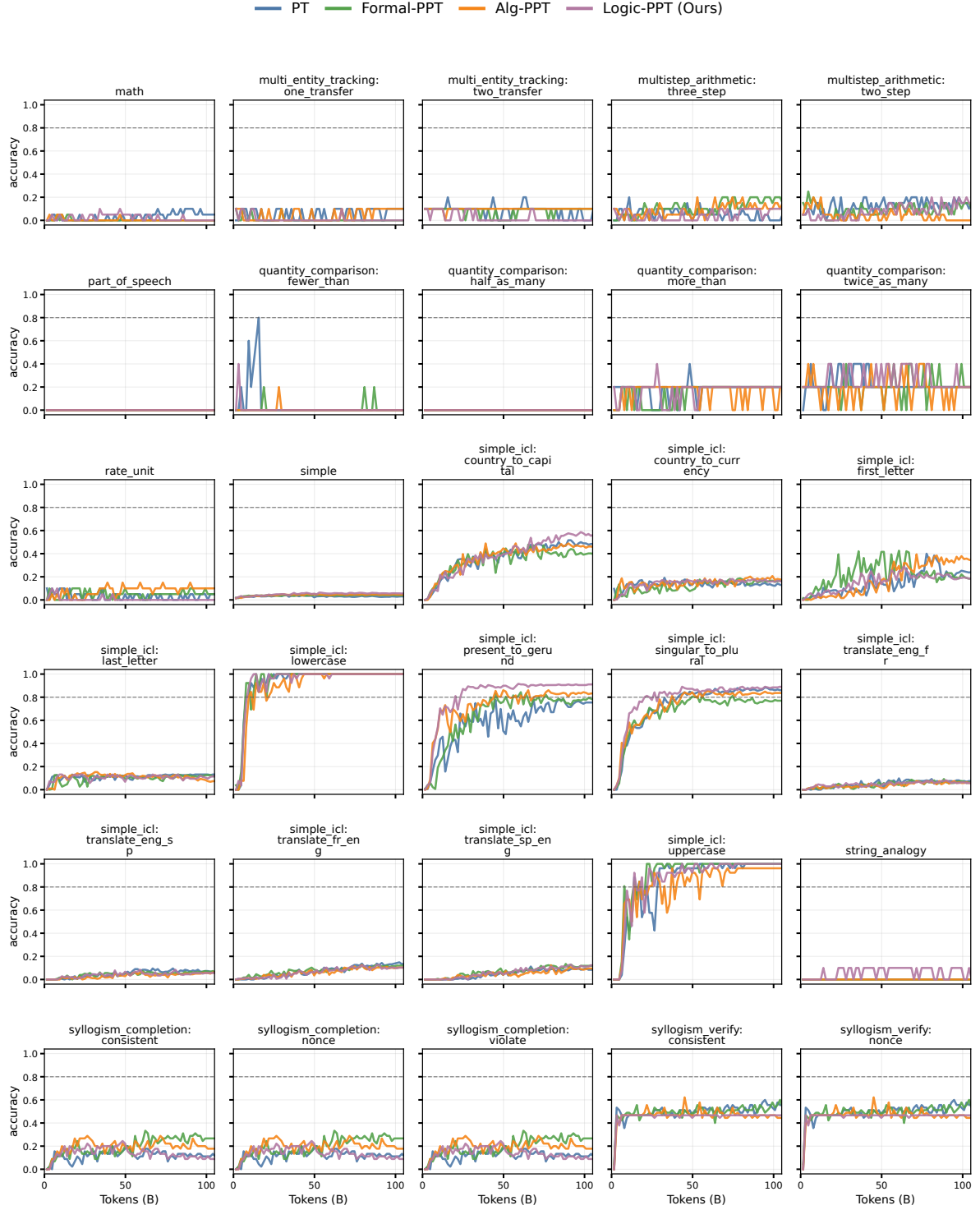


Figure 13: Accuracy trajectories for the third subset of tasks in the full 130-task ElementalTask suite.

Full Elemental Tasks Accuracy — part 4/5

— PT — Formal-PPT — Alg-PPT — Logic-PPT (Ours)

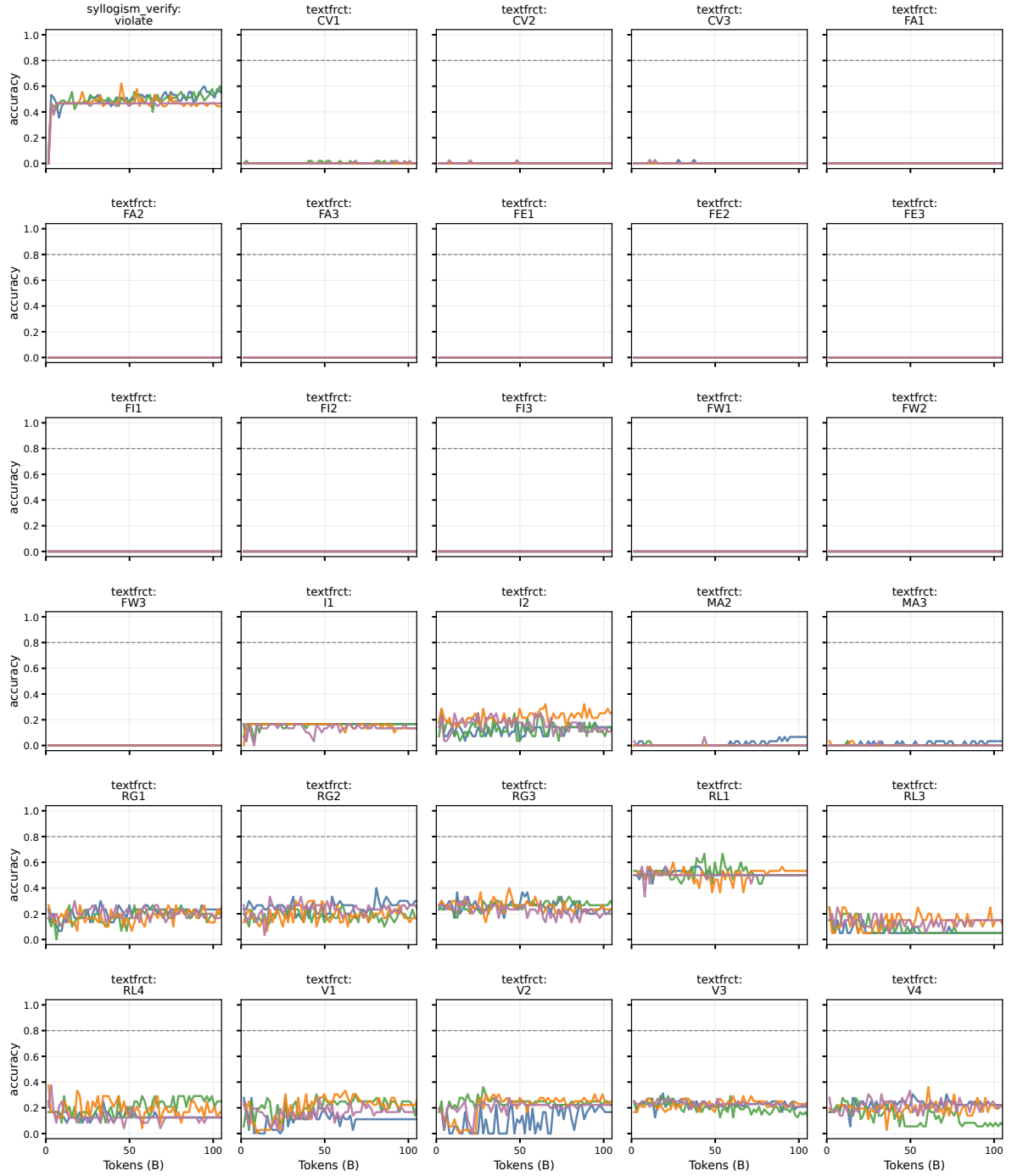


Figure 14: Accuracy trajectories for the fourth subset of tasks in the full 130-task ElementalTask suite.

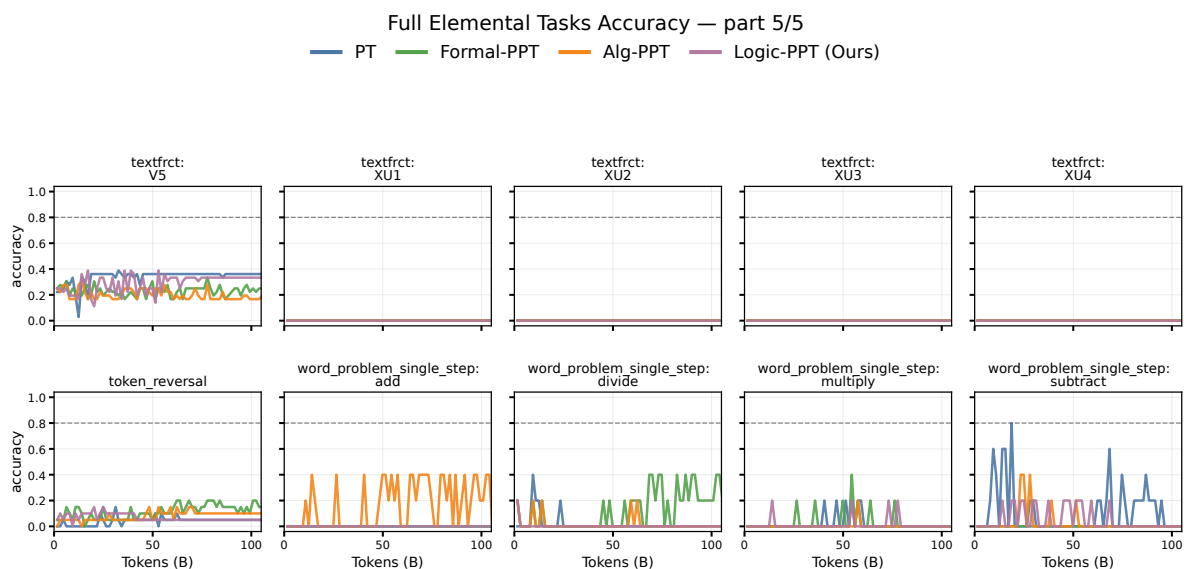


Figure 15: Accuracy trajectories for the fifth subset of tasks in the full 130-task ElementalTask suite.

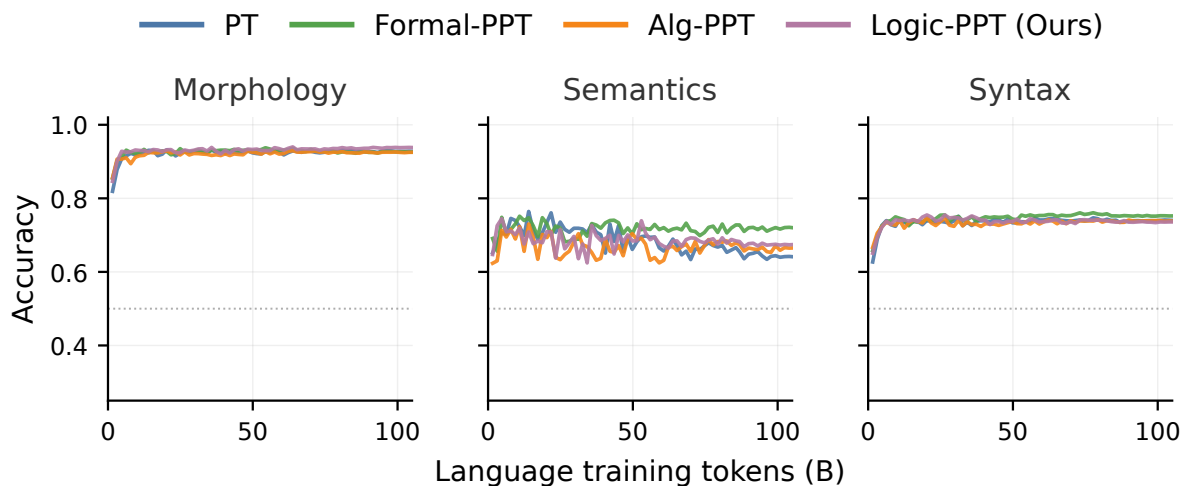


Figure 16: BLiMP accuracy trajectories by linguistic field.