LEVEL UP: DEFINING AND EXPLOITING TRANSITIONAL PROBLEMS FOR CURRICULUM LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Curriculum learning—ordering training examples in a sequence to aid machine learning—takes inspiration from human learning, but has not gained widespread acceptance. Static strategies for scoring item difficulty rely on indirect proxy scores of varying quality and produce curricula that are not specific to the learner at hand. Dynamic approaches base difficulty estimates on gradient information, requiring considerable extra computation during training. We introduce a novel method for measuring the difficulty of individual problem instances that is calibrated to a series of models of increasing competence, and identify *transitional problems* that are consistently easier as model ability increases. Applying this method to diverse model series constructed from sets of models that are readily available on many tasks, we find that training on a curriculum that *levels up* from easier to harder transitional problems most efficiently improves a model to the next tier of competence. These problems induce a natural progression from easier to harder items, which outperforms other training strategies. By measuring difficulty directly relative to model competence, our method yields interpretable problems, learner-specific curricula, and a principled basis for step-by-step improvement.

1 INTRODUCTION

Machine learning (ML) differs significantly from human learning in practice, both in terms of the learner and the learning method. Human learning is generally curriculum-driven, with a training diet of examples that increase in complexity as the learner improves in ability. On the other hand, an ML model optimized on data drawn i.i.d. from the training distribution (i.e., empirical risk minimization, or ERM) typically learns to perform a task at least as well as when optimized on the same examples presented in a structured manner. Curriculum learning (Elman, 1993; Sanger, 2002; Bengio et al., 2009)—the process of training ML models on a structured data regimen, as opposed to random samples—has shown benefits in some settings, particularly for models that are trained on noisy or limited data (Wu et al., 2020; Wang et al., 2023), in models for sequential decision making such as in reinforcement learning (Tao et al., 2024; Zhao et al., 2022), and in the post-training of foundation models for mathematical reasoning (Zeng et al., 2025). However, several negative results for curriculum learning temper these successes. Curricula have shown no benefit in end performance over ERM on high-quality or very large multi-task datasets (Wu et al., 2020), with overparameterized networks (Mannelli et al., 2024), and even when training on data that emulates human learning (Warstadt et al., 2023).

A primary consideration in curriculum learning is the definition of an appropriate measure of the difficulty of a training example. The structure of a curriculum for human learners depends not only on the relations between the concepts being learned, but also on the learning outcomes and experiences of past learners. Difficulty measures for ML models typically rely on domain-specific and human-centric information that may not translate well to ML models (e.g., in BabyLM submissions (Oba et al., 2023)), or on the evolving training dynamics of the learner (such as the gradient norm (Graves et al., 2017)), which can add considerable expense to training. We observe that very few difficulty measures for curriculum learning on ML tasks leverage the diverse collection of existing models for that task. Such a collection is readily available: training leaves behind checkpoints, foundation models come in many sizes, and a single LLM can be made stronger or weaker simply by prompting. Analogous to the range of performance within and across human learners, these models vary in task proficiency due to *inter-model* differences (size, architecture, complexity of training data, etc.) and *intra-model*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

differences (task-specific context, input, time/compute resources, etc.). Inspired by this observation, as well as the stratification of many human curricula into multiple 'levels' (such as the scholastic grades in the K-12 education system), we ask the following questions: (1) Are there specific problems that consistently mark the boundary between the competence levels of an existing collection of ML models? (2) Can we identify these specific problems at each level, and their common characteristics? (3) Does training on such problems provide an efficient way for the learner to make progress towards the next level (e.g., $4^{\text{th}} \rightarrow 5^{\text{th}}$ grade in K-12), and inform a strategy for curriculum learning?

We answer these questions in the affirmative by defining and identifying *transitional problems*—problems that exhibit a sharp transition in solvability across increasing competence levels. Given a series of models ordered by increasing performance, a transitional problem at a given level can be solved by all models at or above that level, but not by those below. These problems mark competence boundaries and yield an empirically grounded easy-to-hard partitioning of the training data distribution. We develop methods to characterize the levels of competence of an ML model, identify transitional problems, and build a curriculum inspired by human learning. Through experiments across a variety of model series, we show that a LEVELUP curriculum on transitional problems leads to better learning outcomes than other data orderings (including i.i.d.) on both transitional problems and held-out test problems. LEVELUP also outperforms domain-specific curricula (including those based on ground-truth human difficulty measures) and the consistency score of Jiang et al. (2020), which scores each problem with the same models but discards their ordering. Finally, we study how the levels of transitional problems we collect vary along human measures of difficulty. We find in each setting that they align with some key measures but not other, and thus cannot be directly reconstructed from first principles without the use of a model series.

2 RELATED WORK

Efficacy of Curriculum Learning. Despite the ubiquity of curricula in human learning, curriculum learning remains niche in ML practice, with multiple negative results in the literature. The BabyLM challenge (Oba et al., 2023) for developmentally-inspired LLM pretraining saw a variety of domain-specific (Martinez et al., 2023; Edman & Bylinina, 2023; Oba et al., 2023), model-dependent (Opper et al., 2023), and student-teacher curricula (Chobey et al., 2023; Zhang et al., 2023) fail to beat the baseline. Other works show that curricula provide no benefit with large training budgets and clean data (Wu et al., 2020) or over-parameterized models (Mannelli et al., 2024). Fine-grained studies of curriculum learning primarily show benefits in low-data or high-noise regimes (Wu et al., 2020). However, recent work has shown wins in structured domains like reinforcement learning (for long-horizon tasks (Narvekar et al., 2020; Patel et al., 2024; Li et al., 2025b; Zhao et al., 2022) and faster convergence (Tao et al., 2024)) and LLM post-training for instruction-following (Ge et al., 2025) and complex reasoning (Zeng et al., 2025; Polu et al., 2022). The training of large-scale foundation models is increasingly divided into pre-, mid-, and post-training phases (Ouyang et al., 2022; Wang et al., 2025; OLMo et al., 2024), with different training tasks, datasets, and learning objectives in each phase (Liu et al., 2024; Zeng et al., 2025). While these phased approaches operate at a coarser granularity than example-level curriculum learning, they reflect a similar principle of structuring the training process. At a fine-grained level, the GPT-3 and T5 LLMs had non-uniform mixing strategies (Brown et al., 2020; Raffel et al., 2020) in pretraining, and recent work shows additional promising results for curricula in pretraining (Zhang et al., 2026; Liu et al., 2026).

Static and Dynamic Curricula. Curriculum learning methods are typically designed as *static* strategies for ordering and pacing the training data input to a model (Bengio et al., 2009; Weinshall et al., 2018; Jiang et al., 2020), focusing on long-term improvement. The initial cost of curriculum design is amortized over a long training process or by applying to multiple learners, making static curricula suitable for large-scale training such as in structured pre- and post-training (Brown et al., 2020; Wake et al., 2024) and length generalization (Liu et al., 2024) in LLMs. An alternative approach is to *dynamically* adapt the data diet based on a model's learning performance during training (Bellemare et al., 2016; Graves et al., 2017; Zhang et al., 2021). While adaptivity can lead to a better learning signal in some settings, dynamic curricula require periodic re-evaluation and recomputation of difficulty during every training run. While both dynamic and static approaches have their advantages, in this work, we focus on developing and evaluating a *static* curriculum. We present additional details on these strategies in Appendix A.

Difficulty, Ordering, and Pacing. The design of a static curriculum typically involves three key decisions (Wu et al., 2020). The first (and perhaps defining) choice is that of selecting an appropriate *item difficulty strategy*, which is a measure of the utility of a training example for learning. Static curricula typically focus on *model-independent* difficulty measures, such as the quality of an image (Wang et al., 2023; Sheybani et al., 2023), complexity of a reasoning problem (Polu et al., 2022), or the size and quality of textual data (Zhang et al., 2018; Warstadt et al., 2023). These leverage the innate complexity of the training data, but do not account for the specific characteristics of a learning model. While model-derived approaches are rare, we highlight two key works in this area. The first scores item difficulty by training an SVM (Cortes & Vapnik, 1995) on the representations of a teacher model for image classification (Weinshall et al., 2018; Hacohen & Weinshall, 2019). This approach relies on an external model of the performance and dynamics of a model (the SVM), and may not easily generalize to other domains. The second aggregates the performance of models trained on diverse pretraining sets as the *consistency score* (C-score) of an example (Jiang et al., 2020). Like our method, the C-score requires a collection of models and robustly evaluates learning difficulty across them, but it only counts how many models solve an example, discarding which ones do. In this work, we order the same models by their measured performance on the target task and identify the *transitional problems* that separate adjacent levels. With identical models and correctness labels, this ordering yields a curriculum that outperforms one built on the C-score.

Once a strategy for item difficulty is set, the *ordering* of the samples by difficulty (easy-to-hard, hard-to-easy, random) as well as the *pacing* (training effort per difficulty level) must be set. However, finding the optimal pacing (constant, linear, exponential, etc.) requires considerable effort, and in some cases provides only a marginal benefit (Wu et al., 2020). Thus, in this work, we set a simple *constant* pacing strategy for every curriculum, with each difficulty level receiving equal attention. This ensures that performance differs only based on the difficulty measure and ordering.

Human Learning. Our strategy for 'leveling up' ML models with transitional problems is motivated by the structure of curriculum-driven human learning. A learning regimen for humans typically emphasizes sub-tasks that are just out of reach of a learner's current capabilities. In psychology, the technique of *scaffolding* (Wood et al., 1976) is used to teach infants new skills (such as object detection and manipulation) by focusing on tasks in their *Zone of Proximal Development* (ZPD) (Vygotsky, 1978), i.e., tasks that can be accomplished with some teacher assistance. This is analogous to the concept of training on the transitional problems of the next competence level. Neural networks trained on egocentric videos from infants perform best when trained on a developmental (young-to-old) ordering (Sheybani et al., 2023). Even beyond infancy, ordering concepts by complexity is the standard in human learning, even for physical tasks such as motor skills (Sungeelee et al., 2024).

3 Transitional Problems

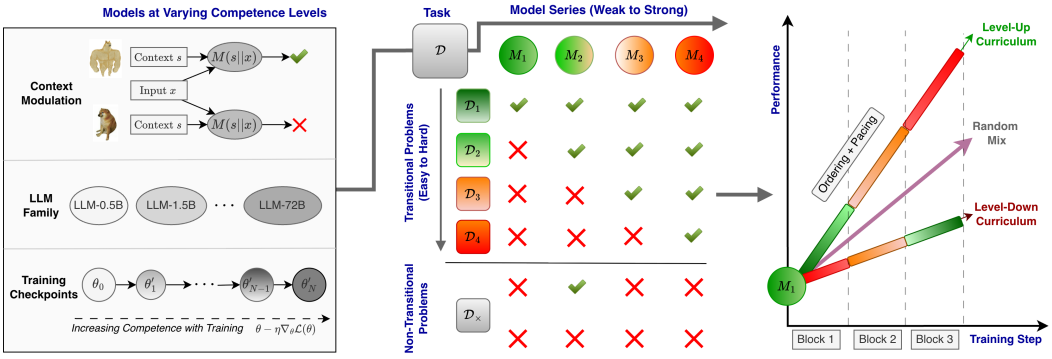

Figure 1: Transitional Problems at a level i can only and consistently be solved by models at a competence level $j \geq i$. We find that with modest budgets, training on an ascending-level curriculum on transitional problems produces models that perform better than those trained with other strategies.

3.1 Models at Varying Levels of Competence

Given a collection of ML models $\mathbb{M}$ on a task with data distribution $\mathcal{D}$, our goal is to design a curriculum for the rapid improvement of a base model M_0. Formally, we select a subset of these models that forms a *model series* of monotonically increasing *strength* (performance) s.

Definition 3.1 (Model Series)**.** Let $\mathcal{M} = \{M_0, M_1, \ldots, M_n\} \subseteq \mathbb{M}$ be a finite set of $n+1$ models. $\mathcal{M}$ forms a *model series* w.r.t. data distribution $\mathcal{D}$ and performance measure $s : \mathcal{M} \to \mathbb{R}^+$ iff the models are indexed by strictly increasing strength: $s(M_i) < s(M_j)$ for all $0 \leq i < j \leq n$.

The variation in performance of models within such a series may be due to *inter-model* differences or *intra-model* differences (Section 1). For example, a family of Large Language Models (LLMs) shows an inter-model progression in size (from 0.5–100+ billion parameters)—and correspondingly, competence—on a broad swathe of ML tasks. Another example is the series of *training checkpoints* collected for error recovery and to study training dynamics, which form a natural progression in strength as their parameters adapt to a task. In contrast, a single LLM can exhibit intra-model variation through *few-shot in-context learning* (Brown et al., 2020), which updates the context with task-specific examples to improve model performance. Models like the Maia-2 Tang et al. (2024) can adapt their performance based on strength-modulating contextual inputs. We depict these groups of models in Figure 1 (left). It is not necessary that higher-level models in these series are generally stronger than lower level ones — only that their performance on the specific task chosen is higher. Appendix B summarizes the four kinds of series used in this paper, emphasizing their ease of construction.

3.2 Defining Transitional Problems

Given a series of models that differ in performance on a learning task, a natural definition for the learning difficulty of an example is then the level of the weakest model that is 'correct' on that example.[1] However, a weak model may get a hard problem right due to shortcut learning, or just by chance. Thus, we add a *monotonicity* constraint to this definition: for a given problem p to be at level i, any model that is below level i must fail to answer p, and any at or above level i must answer p correctly. We designate examples that satisfy this constraint as the *transitional problems* (Figure 1) corresponding to that model series, to indicate the smooth transition in easiness as models grow in competence. In concrete terms, we define the *transitional problems* $\mathcal{T} \subseteq \mathcal{D}$ as follows.

Definition 3.2 (Transitional Problem)**.** Let $\phi_p(M_i) = 1$ if model M_i correctly solves problem p, and 0 otherwise. A problem p is transitional w.r.t. $\mathcal{M}$ at level $1 \leq k \leq n$ if:

- $\forall i \in \{0, \ldots, k-1\} : \phi_p(M_i) = 0 \ \wedge \ \forall i \in \{k, \ldots, n\} : \phi_p(M_i) = 1$

We call $\tau_p = k$ the *transition point* and M_k the *transition model* for problem p. This characterization is based purely on the *performance of the model series* on that problem, and is not directly dependent on the nature of the problem domain or the existence of domain-specific difficulty measures. Simply counting the models that solve p, $\sum_i \phi_p(M_i)$ unconstrained, recovers the C-score (Jiang et al., 2020).

Definition 3.3 (Transitional Problems at τ)**.** For a data distribution $\mathcal{D}$ and a transition point $\tau \in [n]$, the transitional data distribution $\mathcal{D}_\tau$ at point τ is defined as:

$$\mathcal{D}_\tau = \{p \in \mathcal{D} : \phi_p(M_i)_{i<\tau} = 0 \wedge \phi_p(M_i)_{i\geq\tau} = 1\} \tag{1}$$

Restricting a dataset to transitional problems produces a *partially ordered subset*—problems at a given level are equivalent, but are strictly harder (easier) than ones at a lower (higher) level. Notably, this difficulty is with respect to the model series, not based on a human-centric metric. We construct curricula on three *orderings* of transitional problems at constant *pacing*. The LevelUp curriculum trains a model M_i at level i on transitional problems at levels $\mathcal{D}_{i+1} \to \mathcal{D}_{i+2} \to \cdots \to \mathcal{D}_n$. LevelDown reverses this order, training M_i on $\mathcal{D}_n \to \cdots \to \mathcal{D}_{i+1}$. Finally, Random trains on a uniformly random sample from the set $\mathcal{T} = \bigcup_{j=i+1}^{n} \mathcal{D}_j$. In the following sections, we show that transitional problems represent an easy-to-hard ordering of training data—both in terms of learning difficulty and along some human-interpretable features—through experiments with these and baseline curricula across a range of inter-model and intra-model series that are typical in ML settings.

[1] We focus on problem solving (with binary correctness) here, but appropriate thresholding can generalize this approach to other evaluation settings.

4 INTER-MODEL TRANSITIONAL PROBLEMS: MATH

As outlined in Section 3.1, a model series can be constructed in many ways. In this section, we primarily study transitional problems derived from various types of *inter-model* series. In many domains, practitioners develop a series of models that trade off raw performance for other factors (efficiency, cost, runtime, etc.). This is especially the case in the vision and language domains, particularly on reasoning tasks. Models that can reason well in natural language (e.g., LLMs) are typically large and expensive, leading to the development of smaller models (SLMs with ≤ 4 billion parameters). LLM reasoning can be improved permanently with supervised training or temporarily with in-context learning Brown et al. (2020). Given this diversity, we select *mathematical reasoning with SLMs* as our first testbed for transitional problems. We conduct a comprehensive evaluation of curriculum learning across 3 types of model series—distillation, checkpointing, few-shot adaptation—and find that transitional problems are an effective instrument for curriculum learning. We also show that transitional problems identified on a small dataset can be transferred to a larger related one and that LEVELUP on these problems is effective on the target dataset, so the series need only be run on the small set. Finally, we study the characteristics of transitional problems across levels.

Dataset and Models. Our choice of models and dataset is determined by two factors. First, the weaker models in the series must show a low base performance on the dataset, and improve considerably with training. This enables us to study the effects of curriculum learning in settings where the model performance is not too high (i.e., saturated) or too low to be able to observe differences. Second, the dataset must be reasonably interpretable and admit curricula on domain-specific difficulty measures. Based on these criteria, we choose the Qwen2.5 family of LLMs—with $\{0.5, 1.5, 7, 14, 32\}$ billion parameters—as our series (Qwen et al., 2025), and the GSM8k dataset Cobbe et al. (2021) as our reasoning task. The smaller Qwen2.5 models show a low 0-shot performance on GSM8k but improve considerably with training, and the larger ones perform well. Unlike datasets for competition-level math (Hendrycks et al., 2021) or formal reasoning (Zheng et al., 2021), GSM8k solutions are well-formatted (with 2–10 annotated steps of grade-school arithmetic each) and are easy to interpret, allowing for comparisons with domain-specific and dataset-dependent curricula.

Baselines. Aside from training on randomly sampled problems ('IID'), we also benchmark our transitional curricula on two dataset-specific measures of difficulty: the *length of the example* (longer problems are more difficult to interpret and solve) and the *number of reasoning steps in the solution* (annotated by GSM8k). We also evaluate a curriculum based on the C-score (Jiang et al., 2020), which uses exactly the same models and correctness labels as our method but ignores which model solved each problem, isolating the effect of ordering the series. In each experiment, we train the model on each curriculum that is implementable on the given dataset, and evaluate its average accuracy on the held-out test split of that dataset. We evaluate the average success rate over 8 attempts per problem (i.e., *Avg@8*) to account for the variance in model generations (more details in Appendix E.2).

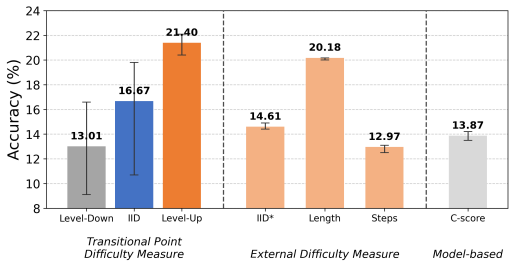

Figure 2: **Math:** Distillation from the Qwen2.5 model family into Qwen2.5-0.5B-Base. LEVELUP, corresponding to progressive distillation, outperforms other orderings and baselines.

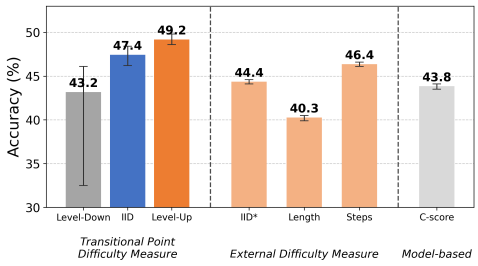

Figure 3: **Math:** Curricula on transitional problems collected from Qwen2.5-1.5B training checkpoints on GSM8k. LEVELUP outperforms RANDOM, LEVELDOWN, and all baselines.

Series A: Distillation. SLMs are typically trained by distilling information—usually output logits—from their larger counterparts Hinton et al. (2015); Xu et al. (2024). Prior work shows improved results from *progressive distillation* (Mirzadeh et al., 2020; Panigrahi et al., 2024), where knowledge is distilled from increasingly stronger models. This is a natural setting for our method, which also leverages a model series ordered by strength. Crucially, our method does not require the full outputs

of a model, only correctness, enabling learning from closed-weight LLMs like ChatGPT[2] and Claude[3]. The series costs one inference pass over the training set per member, computed once and reused by every curriculum and learner (Appendix E.2), whereas dynamic curricula re-score data throughout training. Here, we distill reasoning ability into the smallest model of the Qwen2.5 model family (Qwen2.5-0.5B-Base) via curriculum learning, using information from larger counterparts (1.5B–32B parameters). Figure 2 shows that LEVELUP outperforms all other curricula, including the standard practice of training i.i.d. and the C-score baseline. Contrary to this, the LEVELDOWN curriculum under-performs RANDOM on transitional problems. This indicates that transitional problems are ordered in terms of the *learning difficulty* w.r.t. the model series, which parallels observations in the literature on progressive distillation (Mirzadeh et al., 2020; Panigrahi et al., 2024).

Series B: Checkpointing. Frequent checkpointing is another characteristic of training ML models that leads to a model series with a natural progression in strength. To collect our model series, we train the Qwen2.5-1.5B-Base model M_0 with 28% 0-shot test accuracy on the GSM8k dataset $\mathcal{D}$, periodically collecting training checkpoints $C'_1, C'_2, \ldots, C'_s$ that range from 35% to 59% validation accuracy. To ensure a meaningful separation between levels, we select a 6-checkpoint subset $\{C_1, C_2, \ldots, C_6\}$ such that there is a notable increase ($\sim 5\%$) in accuracy between each (C_i, C_{i+1}) pair on a held-out validation dataset. We then re-train and evaluate copies of Qwen2.5-1.5B-Base on each transitional and baseline curriculum. Figure 3 shows that positive results in the distillation setting also hold for this model series. LEVELUP curriculum outperforms RANDOM and LEVELDOWN, as well as each baseline curriculum. On transitional problems, LEVELDOWN underperforms RANDOM. We observe a similar trend across multiple experimental settings (Appendix D).

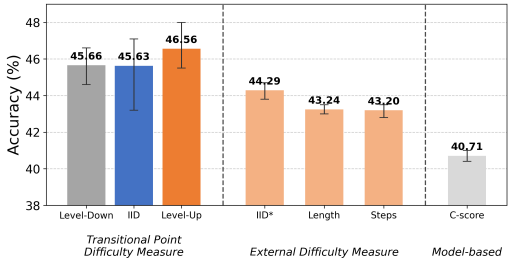

Figure 4: **Math:** Qwen2.5-1.5B trained on transitional problems from k-shot adaptation.

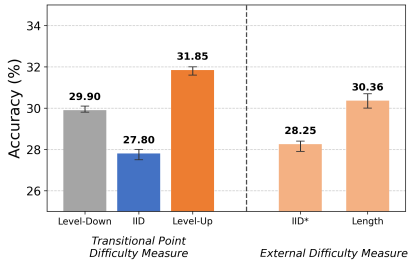

Figure 5: **Math:** Cross-dataset transfer from GSM8k to Orca-math, Qwen2.5-0.5B.

Series C: In-context Adaptation. The strength of an LLM also depends on its task-specific input. Through techniques such as *few-shot in-context adaptation* Brown et al. (2020), an LLM can leverage context to temporarily improve in strength on a task. Here, we conduct a novel analysis on distilling k-shot improvement into a *permanent* increase in task performance with curriculum learning. We collect transitional problems from a 6-model series corresponding to Qwen2.5-1.5B-Base adapted with 0–16 in-context GSM8k examples (with validation accuracies from 35%–73% respectively). This is the cheapest series we study: it needs only the base model and a few labeled examples, so a series can be built even when *no checkpoints or stronger models are available.* We then fine-tune the model with each of our curricula, and find again that LEVELUP outperforms every other curriculum (Figure 4). We see a similar result when training the Qwen2.5-0.5B-Base model on these transitional problems: a case in which the learner is not part of the series, showing that levels measured on one model can guide the training of another, supporting our distillation observations (Appendix D).

Cross-dataset transfer. Because every member must be run on every problem, the one-time cost of a series grows linearly with the size of the dataset. To amortize this cost on a large dataset $\mathcal{D}'$, we study a setting where the transitional problems are instead collected on a related but smaller dataset $\mathcal{D}$ and matched level-by-level to identify *neo-transitional* problems on $\mathcal{D}'$. Here, we study the transfer from our GSM8k transitional problems to a formatted version of Orca-200k (Mitra et al., 2024), a large-scale pretraining dataset of grade-school mathematics word problems. Using a powerful contextual embedding model (Qwen3-Embedding-8B (Zhang et al., 2025b)) we embed problems from both datasets into vector space and identify *neo-transitional* Orca problems at each level based

[2] https://chatgpt.com
[3] https://claude.com/product/overview

on their cosine similarity to GSM8k transitional problems at that level. A detailed description of this procedure is presented in Appendix E.2. We find that training Qwen2.5-0.5B-Base with the LEVELUP curriculum on *neo-transitional* problems outperforms RANDOM and LEVELDOWN, as well as two baselines on the Orca training set[4], showing effective transfer learning (Figure 5). This result boosts the practicality of our method, enabling model-calibrated curricula on large datasets.

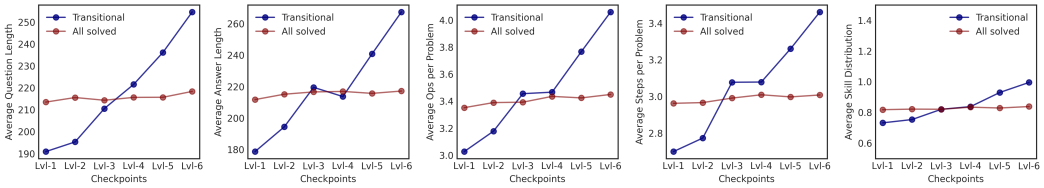

Figure 6: **Math:** Variation in average question length, answer length, and arithmetic operations for transitional problems vs. all solved problems for the 6-model Qwen2.5-1.5B-Base checkpoint series.

Interpreting Transitional GSM8k Problems. Figure 6 shows that transitional problems at higher levels are on average longer and require more operations and reasoning steps than those at lower levels. These trends are absent from the sets of all problems solved by the model at each level i, indicating that the ordering comes from our consistency constraint rather than from structure already present in the dataset. The alignment with human difficulty is also partial (e.g., the number of legal moves in chess is flat across levels; Fig. 12), and curricula ordered directly by human measures underperform LEVELUP in every setting we study. We observe these patterns across every model series (Appendix D); problem counts are in Table 4.

5 INTRA-MODEL TRANSITIONAL PROBLEMS: CHESS

In addition to a variety of inter-model series, the previous section evaluates an *intra-model* series constructed by in-context learning. However, in-context learning is an emergent property of pretrained foundation models and gives only coarse control over the gap between levels. Some models, in contrast, are explicitly trained to modulate their strength from a contextual input, giving fine-grained control over the series. One domain where this is the case is *game-playing*, especially chess, where ML models match the performance of humans at various tiers of competence to emulate competitive sparring partners. To study curricula with transitional problems with such an *intra-model* series, we thus conduct experiments in the *chess* domain.

Chess as a Model Domain. Traditional chess engines such as Stockfish (Romstad et al., 2023) and AlphaZero (Silver et al., 2017) can vary their 'thinking' (search depth) to weaken play from superhuman levels, but this reduction is not well-calibrated and the models cannot then improve by training on chess games. Instead, we use Maia-2 (Tang et al., 2024), a state-of-the-art chess foundation model that can mimic human-level chess at various levels of competence. Maia-2 uses a *skill-aware attention* mechanism to coherently capture the spectrum of human ability, making it well-suited to our study of transitional problems and curriculum learning. Taking as input a chess position and the player/opponent Elo ratings (Elo, 1978), Maia-2 outputs a next-move probability distribution at the player's rating. Our strength function lower bounds this modeling as $s(M_0) = 1000, s(M_1) = 1100, \ldots, s(M_{10}) = 1900$.

Finding Transitional Chess Problems. First, we define the set of **transitional positions** from regular chess games at transition point τ as:

$$\mathcal{D}^{pos}_{\tau} = \{p \in \mathcal{D}^{pos} : \phi_p(M_i)_{i<\tau} = 0 \wedge \phi_p(M_i)_{i\geq\tau} = 1\} \tag{2}$$

where $\mathcal{D}^{pos}$ is a dataset of positions from human chess games in the Lichess Database[5] annotated with the best next move according to Stockfish (Romstad et al., 2023), and $\phi_p(M_i)$ indicates whether the model at strength i (e.g., 1600 Elo) correctly identifies the strongest move. While game positions provide diverse training data, *chess puzzles* are particularly high-quality learning material for skill

[4]Other baselines do not apply: Orca solutions do not annotate reasoning steps; and we assume no access to model predictions for the C-score.

[5]https://lichess.org

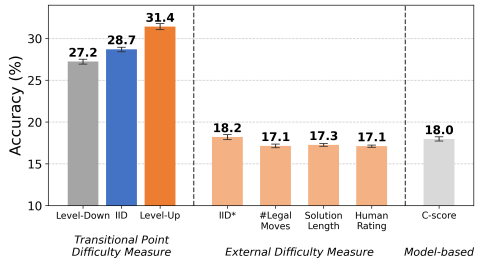

Figure 7: **Chess:** Maia2-1000 trained on puzzles and evaluated on puzzles using next-move accuracy. LEVELUP > IID (RANDOM) > LEVELDOWN, and all of these outperform baselines.

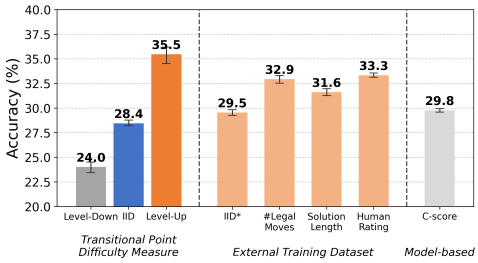

Figure 8: **Chess:** Transitional curricula trained on game positions and evaluated on puzzles using next-move accuracy. LEVELUP outperforms other transitional curricula and baselines.

acquisition. Chess puzzles are a subset of regular positions with a unique best move that typically leads to a decisive advantage, specifically designed to isolate and teach critical tactical patterns and strategic concepts. Therefore, we also define the set of **transitional puzzles** at level τ:

$$\mathcal{D}^{puz}_{\tau} = \{p \in \mathcal{D}^{puz} : \phi_p(M_i)_{i<\tau} = 0 \wedge \phi_p(M_i)_{i\geq\tau} = 1\} \tag{3}$$

where $\mathcal{D}^{puz}$ is a set of randomly selected puzzles from Lichess. We report next-move accuracy on puzzles, scoring only the first move played by the solver. A prediction is correct if the model's highest-probability move matches the target move. Puzzles are a standard way to train human learners on chess tactics, and these transitional puzzle sets enable targeted training on the exact skills and concepts needed by the Maia-2 models to progress from $s_{\tau-1}$ to s_{τ}.

1. Learning from transitional chess puzzles. We identify large sets of training and evaluation transitional problems from the Lichess databases using the Maia-2 model series (1100–1900 Elo). We train the weakest setting of the Maia-2 chess foundation model (at 1000 Elo) on the LEVELUP, RANDOM, and LEVELDOWN curricula. In Figures 7 and 8, IID denotes RANDOM: random sampling from the same transitional training set used by LEVELUP and LEVELDOWN. IID* samples randomly from the training data pool without filtering for transitional problems. We also compare with curricula ordered by the C-score (avg. correct) or by three domain-specific difficulty measures: the *number of legal moves*; *length of the principal variation*; and *player/puzzle Elo* rating in a position. The domain-specific baselines reflect the increase in branching factor, the search depth, and the reasoning difficulty of a chess problem respectively. We first train and evaluate on datasets of chess puzzles, typically used by human learners to improve specific skills. We observe that the LEVELUP transitional curriculum outperforms RANDOM, which in turn outperforms LEVELDOWN (Figure 7) and all baselines, including i.i.d. training. These patterns hold across a variety of pacing stages and training budgets (Appendix C). Additional details of our experiments are presented in Appendix E.

2. Transfer from general positions to chess puzzles. To study transfer from general game play to tactical move prediction, we train models on game-positions ($\mathcal{D}^{pos}$) and evaluate on chess puzzles ($\mathcal{D}^{puz}$)—a structurally similar setting but with differing search characteristics. As shown in Figure 8, LEVELUP achieves 35.5% next-move accuracy, outperforming both RANDOM and LEVELDOWN on transitional data as well as the baselines that use external difficulty measures. This demonstrates that transitional problems can generalize beyond specific datasets in similarly-structured domains.

Atomic Leveling Up in Chess. The characteristics of transitional and all-solved chess puzzles/positions are similar to the math setting, so we present these results in Figure 12 (Appendix C). Since we have a very large number of transitional chess puzzles at every competence level (Table 4 in the Appendix), we can split them into training and testing subsets and study the effects of curriculum learning in the limiting case. We fine-tune Maia-2 at each level ($M_i = M_{1000} - M_{1800}$) on transitional puzzles at every level, and measure next-move accuracy on transitional puzzles at the immediate next level $\mathcal{D}^{puz}_{i+1}$. As shown in Figure 9, training on the transitional problems at the immediate next level $(i+1)$ almost always produces the maximal improvement in the model at level i. For example, the third subplot shows that M_{1400} performs best on 1500-level testing problems ($\mathcal{D}^{puz}_5$) by training with 1500-level problems ($\mathcal{D}^{puz}_5$). Figure 15 (Appendix C) shows similar results for $\mathcal{D}^{pos}$–$\mathcal{D}^{puz}$ transfer learning.

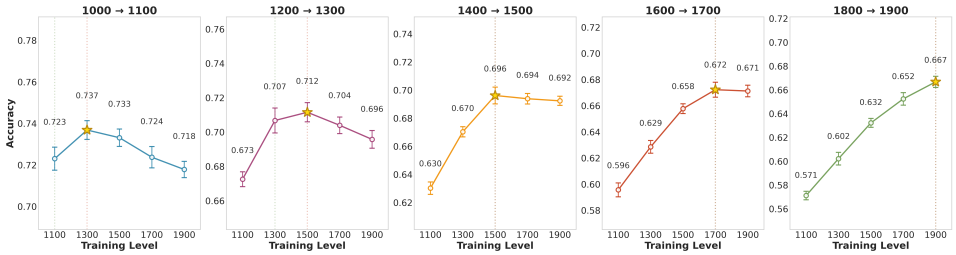

Figure 9: **Chess**: Each subplot depicts results when a chess model at a particular competence level i is trained on puzzles from a single level (along the x-axis), and evaluated by next-move accuracy on puzzles at level $i + 1$ (y-axis). The vertical line in each indicates our hypothesized level to achieve the best performance. The star denotes the actual level that achieves the best performance, which largely aligns with our hypothesis. Error bars represent *std* across 10 runs.

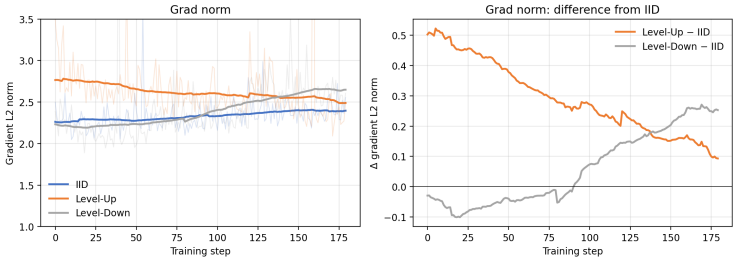

Figure 10: Gradient L^2 norm of LEVELUP, LEVELDOWN, and IID curricula during fine-tuning on chess positions. **Left:** per-step gradient norm (faint lines) overlaid with a centered rolling mean (solid lines). **Right:** smoothed difference from the IID baseline.

Analyzing the Learning Signal. Inspired by an observation in curriculum learning with infant egocentric videos Sheybani et al. (2023), we evaluate the magnitude (L^2 norm) of Maia-2 gradients during training with LEVELUP, LEVELDOWN, and IID curricula. Figure 10 shows that LEVELUP starts with a noticeably larger gradient than IID and decays towards it, while LEVELDOWN begins close to IID and grows above it as training progresses. This effect is also observed in the former work, indicating that an effective curriculum exposes the learner to a qualitatively different optimization regime. We note that the learners in both works use a convolutional neural network LeCun et al. (1989) backbone—we do not observe a similar result in our math setting with LLM learners, likely due to differences in network architecture and output structure.

6 DISCUSSION

In this work, we introduce the notion of *transitional problems* by identifying the training problems that uniquely define the level of competence of an ML model, and are also well-calibrated to models across every level of competence. Through experiments in chess and mathematics, we observe that: (1) *Level-up* curricula that progress through transitional problems of increasing difficulty outperform other curricula on transitional problems and on general training data; and (2) Transitional problems naturally vary along some human measures of problem difficulty, despite not being explicitly optimized to do so. Thus, transitional problems are a *model-specific easy-to-hard* ordering of a subset of training problems based on learning difficulty.

Limitations. Our results show that LEVELUP curriculum learning on transitional problems is potentially a strong and widely applicable training method. However, we only evaluate a constant pacing strategy for each curriculum, and do not mix data between blocks of training. Our method also requires a series whose performance on the target task increases, and closely spaced levels yield fewer transitional problems. Cross-dataset transfer mitigates the latter, but non-monotonic or multi-objective settings remain future work.

AI USE STATEMENT

In this work,

- We **have used** generative AI tools for the following tasks: designing and providing feedback on research methodology or experiments, (assisting with) implementing methods, cleaning and reformatting datasets, supporting qualitative and thematic data analysis.
- We **have not used** generative AI tools for the following tasks: proposing or refining hypotheses, developing theoretical models or conceptual frameworks, interpreting results.
- The following AI tasks are **not applicable** to our work: Generating synthetic data sets, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs.

We have not used generative AI tools for the initial ideation, domain and model selection, and major parts of the implementation of the method. Additionally, we used generative AI tools for the following: creating and modifying scientific figures or images, suggesting experimental parameters, creating and editing software code, drafting parts of a research paper, brainstorming, sourcing/searching for information, editing a research paper to improve readability, and identifying relevant literature.

We have reviewed all AI-assisted work in the following manner.

- All AI-generated code was reviewed by the authors, and AI was not used to directly write the code for the core methodology.
- Code used in plotting was manually verified to use the recorded results of experiments to prevent the hallucination of results or information.
- LLM-suggested relevant literature was verified by a manual look up and review of the suggested work, and bibtex entries were retrieved from official sources.
- All LLM-generated writing and editing in this manuscript was subjected to a thorough review and collective editing by the authors.

We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We present extensive details on our research methodology and experimental setup in Sections 4 and 5, and in Appendix E.

REFERENCES

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Aryaman Chobey, Oliver Smith, Anzi Wang, and Grusha Prasad. Can training neural language models on a curriculum with developmentally plausible data improve alignment with human reading behavior? *arXiv preprint arXiv:2311.18761*, 2023.

Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34:11933–11944, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Lukas Edman and Lisa Bylinina. Too much information: Keeping training simple for babylms. *arXiv preprint arXiv:2311.01955*, 2023.

Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

Arpad E Elo. *The rating of chessplayers, past and present*. Arco Pub., 1978.

Chendi Ge, Xin Wang, Zeyang Zhang, Hong Chen, Jiapei Fan, Longtao Huang, Hui Xue, and Wenwu Zhu. Dynamic mixture of curriculum lora experts for continual multimodal instruction tuning. *arXiv preprint arXiv:2506.11672*, 2025.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *international conference on machine learning*, pp. 1311–1320. Pmlr, 2017.

Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International conference on machine learning*, pp. 2535–2544. PMLR, 2019.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Guochao Jiang, Wenfeng Feng, Guofeng Quan, Chuzhan Hao, Yuewei Zhang, Guohua Liu, and Hao Wang. Vcrl: Variance-based curriculum reinforcement learning for large language models, 2025. URL https://arxiv.org/abs/2509.19803.

Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. Characterizing structural regularities of labeled data in overparameterized models. *arXiv preprint arXiv:2002.03206*, 2020.

Sina Khajehabdollahi, Roxana Zeraati, Emmanouil Giannakakis, Tim Jakob Schäfer, Georg Martius, and Anna Levina. Emergent mechanisms for long timescales depend on training curriculum and affect performance in memory tasks. *arXiv preprint arXiv:2309.12927*, 2023.

Jin-Young Kim, Hyojun Go, Soonwoo Kwon, and Hyun-Gyoon Kim. Denoising task difficulty-based curriculum for training diffusion models. *arXiv preprint arXiv:2403.10348*, 2024.

Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, R. Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. In D. Touretzky (ed.), *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989. URL https://proceedings.neurips.cc/paper_files/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

Haoyang Li, Xin Wang, Zeyang Zhang, Zongyuan Wu, Linxin Xiao, Yaofei Wu, and Wenwu Zhu. Self-supervised masked graph autoencoder via structure-aware curriculum. *international conference on machine learning*, 2025a.

Mingxuan Li, Junzhe Zhang, and Elias Bareinboim. Causally aligned curriculum learning. *arXiv preprint arXiv:2503.16799*, 2025b.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Emmy Liu, Kaiser Sun, Millicent Li, Isabelle Lee, Lindia Tjuatja, Jen tse Huang, and Graham Neubig. What do language models learn and when? the implicit curriculum hypothesis, 2026. URL https://arxiv.org/abs/2604.08510.

Zhuoming Liu, Hao Ding, Huaping Zhong, Weijia Li, Jifeng Dai, and Conghui He. Influence selection for active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9274–9283, October 2021.

Stefano Sarao Mannelli, Yaraslau Ivashynka, Andrew Saxe, and Luca Saglietti. Tilting the odds at the lottery: the interplay of overparameterisation and curricula in neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2024(11):114001, 2024.

Richard Diehl Martinez, Zebulon Goriely, Hope McGovern, Christopher Davis, Andrew Caines, Paula Buttery, and Lisa Beinborn. Climb: Curriculum learning for infant-inspired model building. *arXiv preprint arXiv:2311.08886*, 2023.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher–student curriculum learning. *IEEE transactions on neural networks and learning systems*, 31(9):3732–3740, 2019.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models, 2025. URL https://arxiv.org/abs/2410.05229.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5191–5198, 2020.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. Orca-math: Unlocking the potential of slms in grade school math, 2024. URL https://arxiv.org/abs/2402.14830.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020.

Miyu Oba, Akari Haga, Akiyo Fukatsu, and Yohei Oseki. Babylm challenge: Curriculum learning based on sentence complexity approximating language acquisition. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pp. 290–297, 2023.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Mattia Opper, J Morrison, and N Siddharth. On the effect of curriculum learning with developmental data for grammar acquisition. *arXiv preprint arXiv:2311.00128*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Abhishek Panigrahi, Bingbin Liu, Sadhika Malladi, Andrej Risteski, and Surbhi Goel. Progressive distillation induces an implicit curriculum. *arXiv preprint arXiv:2410.05464*, 2024.

YJ Patel, A Kundu, M Ostaszewski, X Bonet-Monroig, V Dunjko, and O Danaci. Curriculum reinforcement learning for quantum architecture search under hardware errors. *arXiv preprint arXiv:2402.03500*, 2024.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning. *arXiv preprint arXiv:2202.01344*, 2022.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Tord Romstad, Marco Costalba, Joona Kiiski, and et al. Stockfish. stockfishchess.org, 2023. Accessed: 2024-01-05.

Terence D Sanger. Neural network learning control of robot manipulators using gradually increasing task difficulty. *IEEE transactions on Robotics and Automation*, 10(3):323–333, 2002.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

Saber Sheybani, Himanshu Hansaria, Justin Wood, Linda Smith, and Zoran Tiganj. Curriculum learning with infant egocentric videos. *Advances in Neural Information Processing Systems*, 36: 54199–54212, 2023.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv*, 2017.

Vaynee Sungeelee, Antoine Loriette, Olivier Sigaud, and Baptiste Caramiaux. Interactive curriculum learning increases and homogenizes motor smoothness. *Scientific Reports*, 14(1):2843, 2024.

Zhenwei Tang, Difan Jiao, Reid McIlroy-Young, Jon Kleinberg, Siddhartha Sen, and Ashton Anderson. Maia-2: A unified model for human-ai alignment in chess. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2409.20553.

Stone Tao, Arth Shukla, Tse-kai Chan, and Hao Su. Reverse forward curriculum learning for extreme sample and demonstration efficiency in reinforcement learning. *arXiv preprint arXiv:2405.03379*, 2024.

Lev S Vygotsky. *Mind in society: The development of higher psychological processes*, volume 86. Harvard university press, 1978.

Alan Wake, Bei Chen, CX Lv, Chao Li, Chengen Huang, Chenglin Cai, Chujie Zheng, Daniel Cooper, Fan Zhou, Feng Hu, et al. Yi-lightning technical report. *arXiv preprint arXiv:2412.01253*, 2024.

Yulin Wang, Yang Yue, Rui Lu, Tianjiao Liu, Zhao Zhong, Shiji Song, and Gao Huang. Efficienttrain: Exploring generalized curriculum learning for training visual backbones. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5852–5864, 2023.

Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. Octothinker: Mid-training incentivizes reinforcement learning scaling. *arXiv preprint arXiv:2506.20512*, 2025.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell (eds.), *Proceedings*

of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning, pp. 1–34, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.conll-babylm.1. URL https://aclanthology.org/2023.conll-babylm.1/.

Daphna Weinshall, Gad Cohen, and Dan Amir. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *International conference on machine learning*, pp. 5238–5246. PMLR, 2018.

David Wood, Jerome S Bruner, and Gail Ross. The role of tutoring in problem solving. *Journal of child psychology and psychiatry*, 17(2):89–100, 1976.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? *arXiv preprint arXiv:2012.03107*, 2020.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*, 2024.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in neural information processing systems*, 34:18408–18419, 2021.

Xiang Zhang, Jiaqi Wei, Zijie Qiu, Sheng Xu, Nanqing Dong, Zhiqiang Gao, and Siqi Sun. Curriculum learning for biological sequence prediction: The case of de novo peptide sequencing. *arXiv preprint arXiv:2506.13485*, 2025a.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat. An empirical exploration of curriculum learning for neural machine translation. *arXiv preprint arXiv:1811.00739*, 2018.

Yang Zhang, Amr Mohamed, Hadi Abdine, Guokan Shang, and Michalis Vazirgiannis. Beyond random sampling: Efficient language model pretraining via curriculum learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5776–5794, 2026.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025b. URL https://arxiv.org/abs/2506.05176.

Zheyu Zhang, Han Yang, Bolei Ma, David Rügamer, and Ercong Nie. Baby's cothought: Leveraging large language models for enhanced reasoning in compact models. *arXiv preprint arXiv:2308.01684*, 2023.

Wenshuai Zhao, Zhiyuan Li, and Joni Pajarinen. Learning progress driven multi-agent curriculum. *arXiv preprint arXiv:2205.10016*, 2022.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. Minif2f: a cross-system benchmark for formal olympiad-level mathematics. *arXiv preprint arXiv:2109.00110*, 2021.

A Additional Related Work

Structured Training The two major approaches to structured training are *active learning* (Cohn et al., 1994) and *curriculum learning*. Active learning optimizes for the *short-term* goal of finding the best next problem(s) to train a learning model on. Strategies for active learning generally rely on 'teacher' models to select batches that minimize the uncertainty over the training distribution, via data clustering (Citovsky et al., 2021), influence functions (Liu et al., 2021), and coresets (Sener & Savarese, 2017). These methods tend to add considerable overhead and require strong estimates of learner uncertainty, and are thus not popular for training massive-scale foundation models. Contrary to this approach, curriculum learning optimizes for the *long-term* of maximizing the performance at the end of a training phase instead of per-step uncertainty. These approaches can be combined to develop curricula with dynamic batch selection using teacher-student models (Matiisen et al., 2019) or multi-armed bandits (Graves et al., 2017), but this adds considerable overhead.

Curriculum Learning in the Literature Though ERM remains the de facto recommendation in ML, we observe that training with various forms of non-uniform data ordering is standard in modern massive-scale ML pipelines. LLMs are frequently pre-trained on sequences that increase in length and context windows that grow in size over training (Liu et al., 2024). Similarly, curricula are also useful in learning other long sequences of data with autoregressive models, at scales ranging from simple parity tasks for RNNs (Khajehabdollahi et al., 2023) to peptide sequencing (Zhang et al., 2025a), graph learning (Li et al., 2025a), and diffusion models (Kim et al., 2024). Reinforcement learning methods often use curricula to learn increasingly long action sequences (Narvekar et al., 2020; Patel et al., 2024; Li et al., 2025b; Zhao et al., 2022) and for faster convergence (Tao et al., 2024). The training of many autoregressive foundation models has a consistent, curriculum-like structure, with large-scale pre-training phase on unstructured data followed by a structured post-training phase Brown et al. (2020). Recent training schemes include a 'mid-training' phase (Wake et al., 2024; OLMo et al., 2024) that focuses on enhancing capabilities with higher-quality data than in pre-training. The highly structured post-training phase has shown wins for curricula in training for instruction-following (Ge et al., 2025) and complex reasoning Zeng et al. (2025); Polu et al. (2022). In contrast, efforts to explicitly incorporate curriculum learning into pre-training have been largely unsuccessful, even with data inspired by human development. The BabyLM challenge (Oba et al., 2023) saw a myriad of curriculum-based approaches fail to beat the baseline for pre-training a language model on 10-100 million tokens of data. These included domain-specific (Martinez et al., 2023; Edman & Bylinina, 2023; Oba et al., 2023), model-dependent (Opper et al., 2023), and student-teacher (Chobey et al., 2023; Zhang et al., 2023) setups, but notably did not evaluate curricula that dynamically assess example difficulty to determine the next inputs during training. Other negative results for curriculum learning stem from settings where models are overparameterized (Mannelli et al., 2024) or can be trained to convergence with relatively noise-free datasets (Wu et al., 2020).

B Constructing a Model Series

Table 1 summarizes the four kinds of model series used in this paper and what each requires beyond the base model M_0. In every case, the input to our method is a matrix of per-problem correctness with one row per member of the series, obtained by a single pass of inference over the training set with each member; weights, logits, and gradients are never needed. Two of the four series (training checkpoints and k-shot prompting) involve no model other than M_0 itself, so a series can be constructed for any model one intends to train. Because the series is evaluated once and the resulting transitional problems are a static partition of the training set, its cost is amortized over every curriculum run and every learner that uses it (Section 4 trains both Qwen2.5-0.5B and Qwen2.5-1.5B on problems identified with the same series).

C Additional Chess Results

Evaluating Maia-2 models, in detail.

In Section 3, we provide an overview of our process for collecting transitional problems with Maia-2. Since this model is uniquely suited to the chess domain, we outline our exact methodology for this process below.

Series	Members	Needed beyond M_0	Cost in this paper
Checkpoints (Sec. 4, Series B)	checkpoints of a short training run of M_0	nothing; a by-product of training	25 min on 4×A100 (Table 5)
Model family (Sec. 4, Series A)	size variants of one family	inference with larger models; correctness only, so closed-weight models suffice	inference on the training set
k-shot prompting (Sec. 4, Series C)	M_0 with $k = 0, \ldots, 16$ in-context examples	nothing	inference on the training set
Skill-conditioned model (Sec. 5)	one model with a strength input (Maia-2)	nothing	inference on the training set

Table 1: The four kinds of model series used in this paper and what each requires. Two can be built from the base model alone, so the method never presupposes models that do not already exist.

We set both the active and opponent strengths to s_i to ensure a consistent skill representation. For a given chess position b conditioned on s_i, the Maia-2 policy head output $f_\theta(b, s_i) = \pi(\mathbf{a}|b, s_i)$, estimates the probability distribution over legal moves (i.e., actions) $\mathbf{a}$. Each M_i can be fine-tuned with the following loss function:

$$\mathcal{L}(\theta) = -\mathbb{E}_{b\sim\mathcal{D}}[\log \pi(a^*|b, s_i)] \tag{4}$$

where $\mathbb{E}_{b\sim\mathcal{D}}[\cdot]$ denotes the expectation over chess position b sampled from the data distribution $\mathcal{D}$, where a^* is the target move under b, which we set to be the highest-ranked move according to Stockfish. This serves as a proxy for the best move, and in positions where multiple moves are equally strong, this choice is arbitrary. For puzzles, b is the board state before the solver's first move, and a^* is the target move. We score only this move, not the complete solution sequence. Throughout the chess experiments, accuracy is the fraction of evaluated examples for which the model's highest-probability move matches a^*. We determine move correctness as:

$$\phi_p(M_i) = \begin{cases} 1 & \text{if } \operatorname{argmax}_{\mathbf{a}} \pi(\mathbf{a}|b, s_i) = a^* \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

This allows us to map each problem $p = \{b, a^*\}$ to its transition point $\tau_p \in \{1, 2, \ldots, 10\}$ in the model series, where positions solvable or unsolvable by all models are excluded.

Transitional Problems beget Curriculum learning. As shown in Figure 11, only training on transitional problems yields positive results for easy-to-hard (i.e., level-up) curricula: Transitional Position achieves +9.9% and Transitional Puzzle achieves +6.0% improvement over IID. In contrast, external difficulty measures fail to benefit from a curriculum/anti-curriculum, with #Legal Moves showing substantial degradation (-7.3%) and Solution Length and Human Rating remaining roughly on par with IID. This indicates that transitional problems capture a notion of difficulty that is particularly amenable to curriculum learning, whereas external heuristics do not induce a useful easy-to-hard structure for the learner.

Characteristics of Chess Problems. Like in the math setting, we observe that transitional problems are weakly correlated with some human measures of difficulty. There is virtually no variation in the average difficulty (to humans) of chess puzzles whose first solver move is predicted correctly at each level of the Maia-2 model (Figure 12, in maroon). The Average puzzle rating stays between 1352–1360 Elo, solution length between 3.43–3.45 moves, and the number of legal moves per position at ∼30.65. In contrast, the sets of transitional problems (in navy) show a larger increase in the average puzzle rating ($1525 \rightarrow 1646$) and solution length ($3.6 \rightarrow 3.8$), indicating that our consistency constraint leads to better alignment with human difficulty. The average number of legal moves remains static in both settings, reinforcing that transitional problems cannot be identified simply by composing intuitive difficulty measures.

Effect of Training Budgets. As shown in Figure 13, the advantage of Level-Up curricula holds consistently across training budgets. Level-Up outperforms both IID and Level-Down at all scales (Tiny, Low, Mid), with the relative ordering Level-Up > IID > Level-Down preserved throughout. The absolute gap between Level-Up and IID remains stable (∼3–4%), indicating that the benefits of transitional point-based curricula are robust across different computational budgets.

Effect of Curriculum Steps. As shown in Figure 14, we did not find notable patterns in the effect of curriculum steps in chess. We would like to clarify that our goal to include multiple settings in terms of steps is to show the *consistency* of the advantage of level-up curricula over baselines, instead of finding the best hyperparameter setting.

Transfer Learning in Atomic Level-Up. We have shown the results with $\mathcal{D}^{puz}$-$\mathcal{D}^{puz}$ pair ($\mathcal{D}^{puz}$ for training, $\mathcal{D}^{puz}$ for testing) in the main paper, while we also observe consistent patterns with the $\mathcal{D}^{pos}$ - $\mathcal{D}^{puz}$ pair, which is under such a *transfer learning* setting. We fine-tune models $\{M_0, M_2, M_4, M_6, M_8\}$ on their 'level-up' transitional game positions $\mathcal{D}^{pos}_{\tau}, \tau \in \{1, 3, 5, 7, 9\}$ and test on the equivalent transitional puzzles $\mathcal{D}^{puz}_{i+1}$ for model M_i. As the results show in Figure 15, training on transitional problems one level up from the model's competence lead to the largest improvement in most settings.

D Additional Math Results

In Section 4, we examine the effect of training the Qwen2.5-1.5B-Base model on GSM8k transitional problems in an *in-distribution* setting. We also evaluate a progressive distillation setting, where the Qwen2.5-0.5B model is trained on larger models from its model family. We find that in these settings, the 'level-up' (easy-to-hard via increasing transitional problem level) curriculum outperforms the random and level-down curricula, and also outperforms curricula on the original training distribution for the same amount of compute. In this section, we evaluate curriculum learning with transitional problems in additional settings, and find that our results generalize beyond the specific model and collection method combinations presented in the main text.

D.1 Transitional Problems from the Trained Qwen2.5-0.5B Series

Section 4 showed the effect of training Qwen2.5-1.5B-Base on a model series that is a collection of selected checkpoints from the training and evaluation of a separate copy of Qwen2.5-1.5B-Base. Figure 16 shows a similar result for this experiment conducted with the Qwen2.5-0.5B-Base model. In this case, the level-up curriculum outperforms other curricula, but only matches the performance of i.i.d. sampling from the full training distribution. Nonetheless, this reinforces our result that transitional problems represent an easy-to-hard ordering of training problems based on learning difficulty, as the level-down curriculum performs worse than a random mixture of transitional problems, and the level-up curriculum performs significantly better than random. Figure 17 shows that the level-up curriculum beats the random and level-down curricula across training budgets.

Interpretability. As with the the Qwen2.5-1.5B checkpoint series, we observe that the transitional problems derived from the 5-model series on Qwen2.5-0.5B varies by level along human intuition in the length and the number of steps in the answer, but not in the type of skills required to solve a problem on average (Figure 18). This trend is weaker than in the series from 1.5 billion parameter model, which may explain the relatively weaker performance relative to i.i.d. training on the full data distribution that is observed in Figure 17.

Additionally, we show that this variation in difficulty is not unique to any given set of checkpoints (i.e., one model series) from training a base model. Figure 18 shows that a 5-model series (with 10% improvement in accuracy between levels) shows the same trends as a 6-model series (where the improvement drops to 5%, Figure 6), indicating that our curriculum results are likely to hold for any definition of a model series with a reasonable difference in performance between levels.

D.2 Additional Results on Progressive Distillation

In Section 4, we show that transitional problems can be used as an instrument of progressive distillation—transferring knowledge from more competent (here, larger) models to less competent (smaller) models by training the Qwen2.5-0.5B base model on transitional problems from a model series corresponding to increasing model sizes (1.5B, 7B, 14B, 32B) in the Qwen2.5 family of models. Figure 19 shows the effect of training Qwen2.5-1.5B on the subsequently stronger models, with the level-up curriculum corresponding to the transitional problems from the $7 \rightarrow 14 \rightarrow 32$ billion levels, respectively. As in all other settings, the level-up curriculum outperforms the random and level-down curricula as well as the training-set baselines. However, the margin of improvement is smaller than

with Qwen2.5-0.5B, particularly with a few more training steps (Figure 20), likely due to the reduced number of levels (3 as opposed to 4) due to the unavailability of a larger model in the same family.

Interpretability. The performance of each model in the Qwen2.5 model family (i.e., model series) is significantly higher ($> 10\%$) than its smaller predecessor. Thus, we find that even the sets of all solved problems by a model at level i vary mildly along human-interpretable criteria. However, as seen in Figure 21, the transitional problems still show a much stronger increase in difficulty (by, e.g., question or answer length) than the set of all solved problems, showing that the consistency constraint provides a useful separation even when the gap between levels is large.

D.3 An Alternate Formulation: the Few-shot Model Series

For a given series of models ordered by their competence on a task, the set of transitional problems at level i is the subset of problems from that task distribution that are solved by all models that have a competence level $\geq i$ (Section 3). So far, we have explored two ways to define the model series: via checkpoints from training on the target task distribution, and via a family of models that are naturally increasingly competent due to, e.g., increasing model size. In both of these settings, it is *intuitive* that a stronger model represents a higher tier of competence than a weaker model, and that the weaker model can be improved either by task-specific training or by distillation from stronger models. Now, we examine yet another setup for defining a model series—a model at level i is simply the base (level 0) model to which a test-time few-shot adaptation (such as in-context learning) is applied, with increasing adaptation as the level i increases. Though this setup is less intuitive than the previous settings, it is by far the most efficient to implement, as identifying higher levels does not require the training of the base model or inference with larger models.

To test whether transitional problems defined over the k-shot model series are useful to curriculum learning, we use the Qwen2.5-1.5B base model with $k = \{1, 2, 4, 8, 16\}$ shots of adaptation. We choose the 1.5 billion parameter model as it shows a consistent increase in performance as additional in-context examples are added, going from 35.8% 0-shot validation accuracy to 73.4% with 16 in-context examples. We then identify the transitional problems from this model series, and train the base Qwen2.5-0.5B and Qwen2.5-1.5B models on the level-up, random, and level-down curricula, in addition to our baseline curricula for the same number of training steps. As seen in Figures 22 and 23, the easy-to-hard curriculum on transitional problems outperforms all other curricula, especially when transferring to the weaker 0.5 billion parameter model.

Interpretability. Figure 24 shows that transitional problems corresponding to the k-shot transitional series display the same characteristics as the corresponding problems defined in other ways, with some limited jitter. In particular, the differentiation in the average number of operations or the proportion of skills varies more sharply than with other methods, indicating that *in-context examples are useful in recapitulating arithmetic skill* in the Qwen2.5-1.5B model.

E Reproducibility

In this section, we describe our experimental settings in full. We aim to provide enough detail to independently reproduce our results from this paper.

E.1 Chess Experiments

Table 2: Hyperparameter Settings in Chess Experiments.

Budget	Learning Rate	Weight Decay	#Train	#Test	#Runs	Max Steps	Batch Size
Tiny	10^{-3}	10^{-6}	10,000	10,000	10	90	64
Low	10^{-3}	10^{-6}	10,000	10,000	10	90	128
Mid	10^{-3}	10^{-6}	10,000	10,000	10	180	256

E.2 MATH EXPERIMENTS

Training checkpoints for the model series. To collect the checkpoints for the model series, we train our base learners (0.5/1.5 billion parameter Qwen2.5 models) on the GSM8k training dataset for 100 steps with an effective batch size of 32 (1.5B) or 64 (0.5B) training samples distributed across 4 NVIDIA A100 GPUs[6] and a learning rate of 1×10^{-5}, and collect checkpoints every 5 training steps. We then evaluate the validation accuracy for each checkpoint, and select checkpoints that differ in validation performance by an evenly-spaced interval corresponding to the desired number of models in the series. In this and every other evaluation, we compute the performance of a model as its expected 0-shot 'exact match' accuracy to the GSM8k final answer over 8 attempts for every problem (Avg@8), averaged over the evaluated dataset. Following the recommendations of the Qwen team in existing work (Jiang et al., 2025), we use a temperature of 0.75 and a top-p value of 0.95. We use a generation prompt that is similar to the evaluation prompt in GSM-Symbolic (Mirzadeh et al., 2025).

Qwen2.5 model family series. We evaluate each model in the Qwen2.5 model family as above, but in a 1-shot in-context learning setting (i.e., with 1 training problem and solution as an example in the prompt). This is to ensure that each model formats the answers correctly—the larger Qwen models tend to put the answer between the `\boxed` LaTeXblock, which is incompatible with our answer extraction process. The validation accuracy of the model family ranges from 38% for the smallest model to over 93% for the largest.

Transfer learning. Matching Orca problems to GSM8k problems based on learning difficulty is a complex process that we detail here. The Orca dataset is large, but meant for pretraining prior to fine-tuning on GSM8k, and this contains poorly formatted LLM-generated solutions. We use the DeepSeek-R1 distilled Qwen3-8B model (DeepSeek-AI, 2025) in thinking mode to re-format the solutions of 50,000 Orca problems into the style of GSM8k for better problem matching. We use a prompt that encourages mathematical consistency with the original answer, but formatting consistency with 16 provided GSM8k examples. After filtering to ensure the above consistency creiteria are met, our `orca-gsm8k-formatted` dataset consists of roughly 6000 training problems, of which 3800 are training problems, and 2554 are test problems. We then use the Qwen3-Embedding-8B contextual embedding model (Zhang et al., 2025b) to embed each GSM8k transitional problem and Orca example into a 256-dimensional vector, with a prompt to the embedding model that encourages the problems to have a high cosine similarity if they are equally difficult. Using manually defined thresholds on cosine similarity, (mean similarity ≥ 0.65, max similarity ≥ 0.8), we identify the Orca problems that best correspond to each set of GSM8k transitional problems to produce the *neo-transitional* problems on `orca-gsm8k-formatted`. We then conduct curriculum learning experiments on `orca-gsm8k-formatted` with our model series as in Section 4.

Curriculum Comparisons.

To collect data for each baseline curriculum (len, steps), we sort the dataset by the answer/question length or steps, respectively, and use a *stratified sampling* strategy to find a subset that spans the full range of lengths (or steps) in the dataset. We repeat each experiment 5 times with varying random seeds for sampling i.i.d. and between each curriculum level, and present averaged results over experiments. We train with the same hyperparameters detailed above. The number of steps in each curriculum varies depending on the number of transitional problems we identify in the GSM8k training set, and is listed in Table 3 for each setting. Note that each step involves training on 32 (1.5 billion) or 64 (0.5 billion) GSM8k training examples, and our experiments are primarily limited by the training set size of GSM8k and the transitional problems we identify.

Table 3: Hyperparameter Settings in the Math Experiments.

Size	0.5B-ckpts	1.5B-ckpts	0.5B-family	1.5B-family	0.5B-k-shot	1.5B-k-shot	0.5B-Orca
Tiny	8	8	–	–	–	–	–
Low	10	10	10	10	10	10	10
Mid	15	12	–	12	–	–	–

[6] `https://www.nvidia.com/en-us/data-center/a100/`

Table 4: Dataset statistics. #Source is the total pool used to identify transitional problems; #Transitional is the number kept for training (identified for Math, capped for Chess); #Per level lists the per-stage split.

Setting	Method	Model	Dataset	#Source	#Transitional	#Per level
Math	In-distribution	Qwen2.5-1.5B-Base	GSM8k	6,500	2,092	675 / 462 / 272 / 289 / 394
Math	Distillation	Qwen2.5-0.5B-Base	GSM8k	6,500	5,442	2,181 / 2,113 / 693 / 255
Math	Neo-transitional	Qwen2.5-0.5B-Base	Orca-formatted	5,000	539	174 / 139 / 114 / 56 / 56
Chess	In-distribution	Maia-2	Puzzles	∼5.8M	180,000	20,000
Chess	Transfer	Maia-2	Game Positions	∼300B	180,000	20,000

Table 5: Per-phase runtime for the math experiments.

Dataset	Phase	Model	Runtime
GSM8k	Collect checkpoints	Qwen2.5-1.5B-Base	25 min 22 sec
Transitional/GSM8k	Curriculum (per method/baseline)	Qwen2.5-1.5B-Base	8 min 5 sec

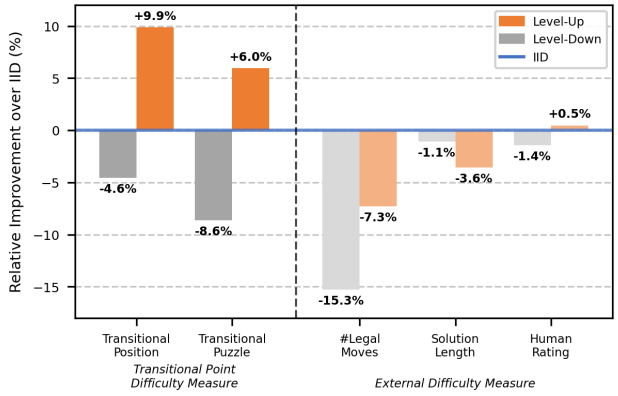

Figure 11: Relative improvement over IID on chess, evaluated on test sets matched to each difficulty measure. Only transitional point-based measures (left) benefit from Level-Up curriculum ordering, while external difficulty measures (right) show marginal improvement or degradation.

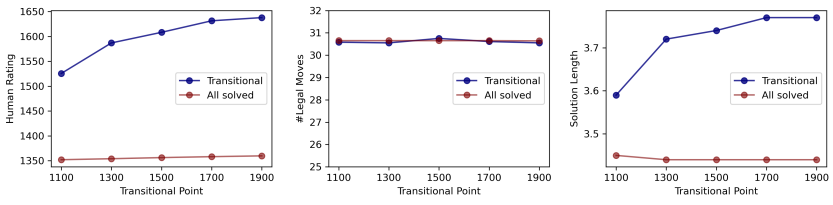

Figure 12: **Chess**: Average human rating, number of legal moves, and solution length for transitional problems versus all solved problems at each transitional point. For transitional problems, the human rating and solution length correlate with the competence level, while the number of legal moves does not. None of these measures correlate for the full set of puzzles across levels.

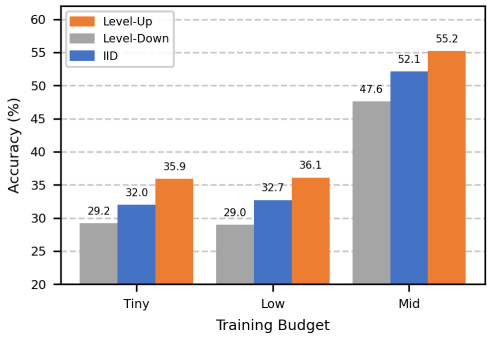

Figure 13: Performance comparison of curriculum learning strategies for chess models across different computational budgets. IID, Level-Up, and Level-Down denote the i.i.d baseline, easy-to-hard curriculum, and hard-to-easy curriculum, respectively, with the Tiny, Low, or Mid training budgets defined in Table 2.

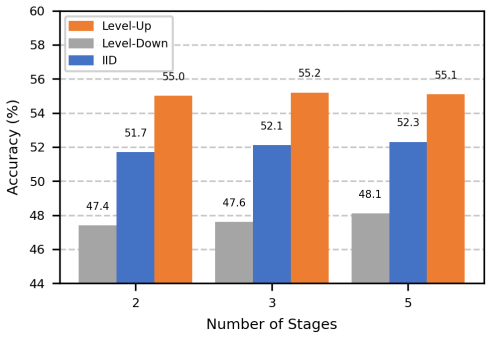

Figure 14: Performance comparison of curriculum learning strategies for chess models across the number of curriculum steps. IID, Level-Up, and Level-Down. denote the i.i.d baseline, ascending curriculum from easy to hard, and descending curriculum from hard to easy, respectively, including 2, 3, or 5 curriculum steps.

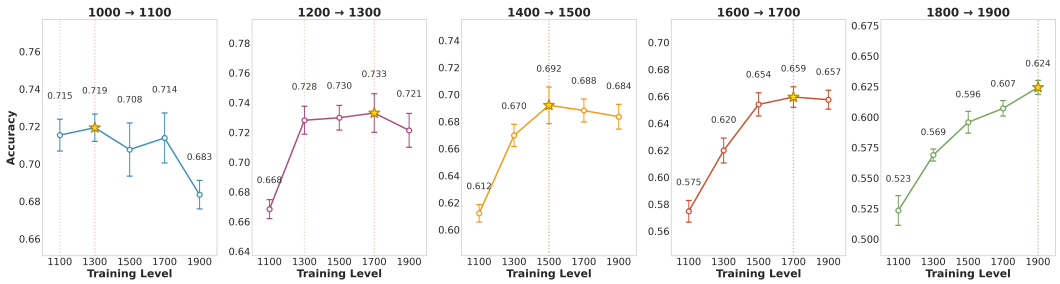

Figure 15: Each subplot depicts results when a **chess** model at a particular competence level i is trained on game-positions from a single level (along the x-axis), and evaluated by next-move accuracy on puzzles at level $i+1$ (y-axis). The vertical line in each indicates our hypothesized level to achieve the best performance. The star denotes the actual level that achieves the best performance, which largely aligns with our hypothesis. Error bars represent *std* across 10 runs.

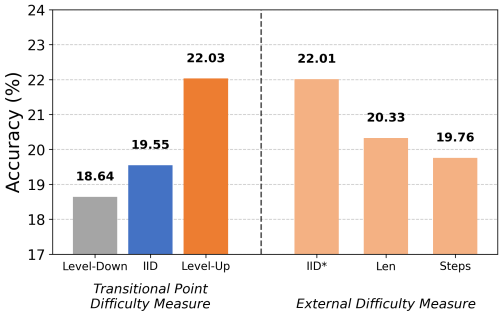

Figure 16: Qwen2.5-0.5B-Base trained on transitional problems from the few-shot adapted Qwen2.5-1.5B model series. The level-up curriculum outperforms every other curriculum, but only shows a slim win over training i.i.d. on the full dataset.

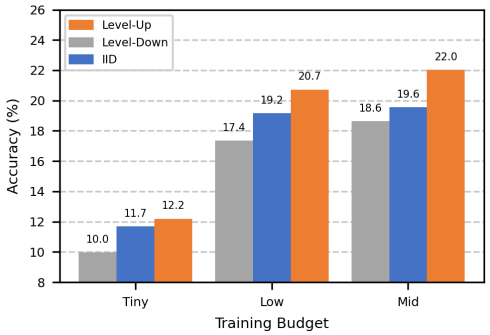

Figure 17: As the number of training steps increases, the level-up curriculum maintains an edge over other curricula on transitional problems, up until the saturation of test set performance.

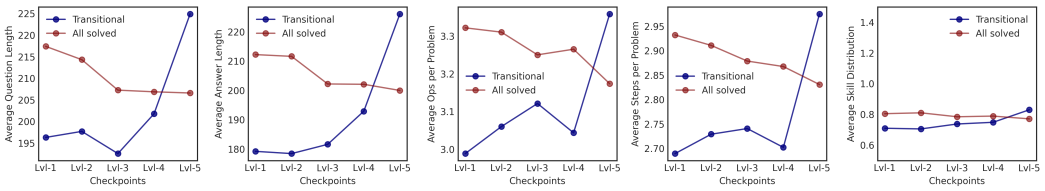

Figure 18: Variation in average question length, answer length, and arithmetic operations for transitional problems vs. all solved problems for a 5-model series obtained from the training checkpoints of **Qwen2.5-0.5B** on GSM8k.

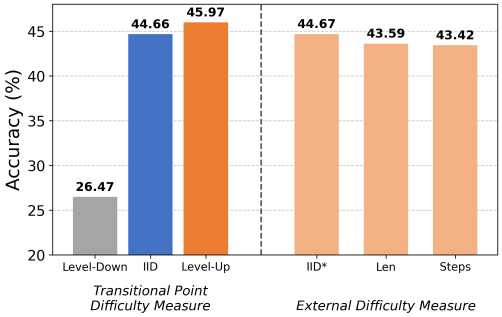

Figure 19: Qwen2.5-1.5B trained on transitional problems from a model series consisting of the 7, 14, and 32 billion parameter Qwen2.5 models. We observe the same ordering of results, but with significantly worse LEVELDOWN performance.

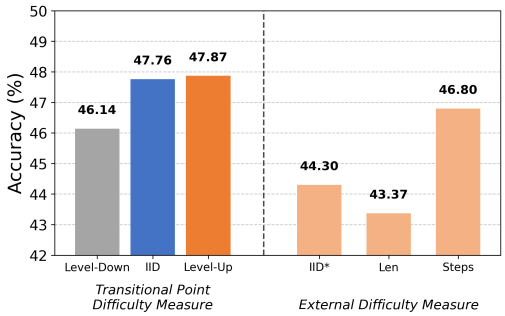

Figure 20: As the number of training steps increases, the lack of additional levels (and thus, transitional problems) reduces the advantage of the level-up curriculum maintains an edge over other curricula.

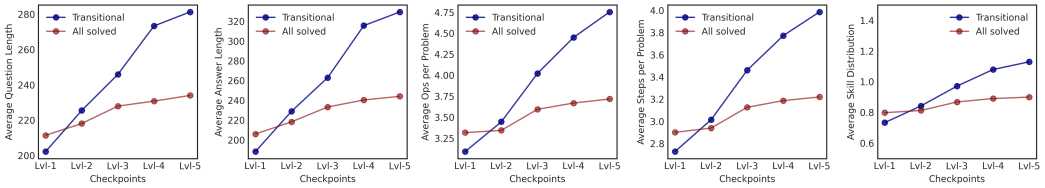

Figure 21: Variation in human measures of difficulty for a 5-model series obtained from the Qwen2.5 model family. Due to the large improvement in performance between levels, the set of all solved problems also shows an increase in difficulty. However, the transitional problems show a much sharper change between levels, indicating better differentiation in learning difficulty.

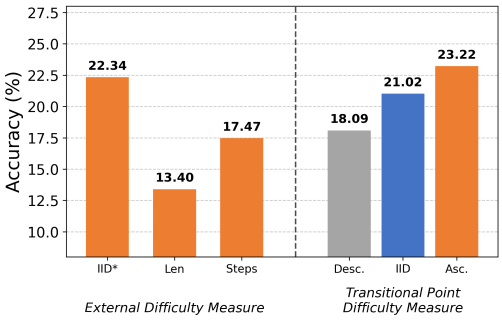

Figure 22: Qwen2.5-0.5B trained on k-shot transitional problems from Qwen2.5-1.5B

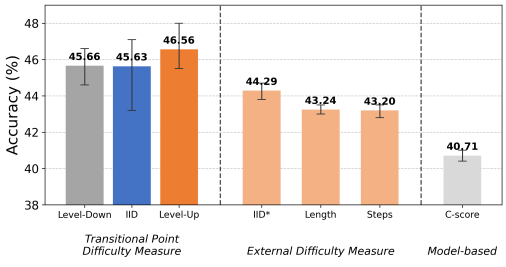

Figure 23: Qwen2.5-1.5B trained on k-shot transitional problems from Qwen2.5-1.5B

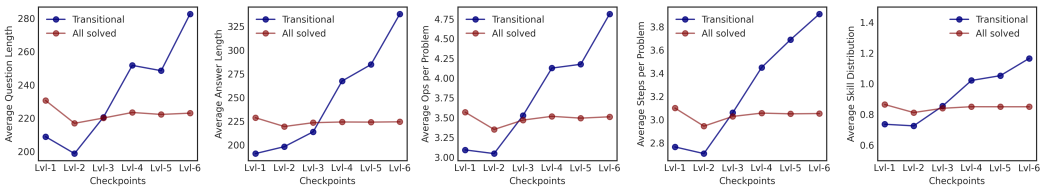

Figure 24: Variation in human measures of difficulty for a 6-model series obtained from increasing k-shot adaptation on the Qwen2.5-1.5B model. Even with the reduced differentiation in performance between model levels, the transitional problems exhibit a human-interpretable difficulty increase.