Reinforcement Learning for Reasoning in LLMs

Nikhil Chowdary Paleti
A69035387
npaleti@ucsd.edu

Shankara Narayanan Venkateswara Raju
A69034474
svenkateswararaju@ucsd.edu

Tarun Kumar Gupta
A69033596
t2gupta@ucsd.edu

Aryan Bansal
A69034284
a3bansal@ucsd.edu

Abstract

Small language models (SLMs) often struggle with multi-step reasoning tasks like those found in math word problems, especially when trained solely via supervised fine-tuning. In this work, we explore the effectiveness of reinforcement learning—specifically, Guided Reinforcement Policy Optimization (GRPO)—as a lightweight and compute-efficient alternative for improving reasoning in SLMs. We fine-tune a range of open-source models (1B–14B parameters) on the GSM8K dataset using handcrafted reward functions that promote both correctness and structured reasoning. Our experiments show that models as small as 3B parameters can exhibit significant improvements in accuracy and interpretability, often outperforming larger models constrained by token limits. We also identify emergent issues like multilingual reasoning and output truncation in larger models, and propose targeted solutions. These results highlight GRPO's potential to unlock structured reasoning capabilities in smaller, more accessible LLMs without the need for extensive compute or curated supervision.

1 Introduction

Multi-step reasoning tasks, such as solving mathematical word problems, remain a significant challenge for small language models (SLMs). While large models with hundreds of billions of parameters excel at structured problem-solving, SLMs often struggle to maintain logical consistency and accuracy across multiple reasoning steps. Supervised fine-tuning (SFT) on chain-of-thought demonstrations can help, but it requires curated data and largely teaches imitation rather than true reasoning. In contrast, reinforcement learning (RL) offers a promising alternative by encouraging models to discover effective reasoning strategies autonomously. Guided Reinforcement Policy Optimization (GRPO), a recent advancement in this space, simplifies standard RL approaches by removing the need for a learned value function and instead using group-based reward normalization. This allows the model to distinguish better reasoning paths while maintaining diversity through KL-regularization, leading to more accurate and interpretable outputs.

Recent work has shown that GRPO can induce emergent reasoning behaviors in both large and small models. Notably, models trained with GRPO exhibit step-by-step logical breakdowns, even when no explicit chain-of-thought prompts are provided. Projects like DeepSeek-R1 have demonstrated that RL-guided training can match or exceed the reasoning performance of models trained via traditional methods, and that the benefits of GRPO can be distilled into models as small as 1.5B parameters. In this study, we investigate how GRPO fine-tuning impacts SLMs across different architectures and scales using the GSM8K dataset. We evaluate five models ranging from 1.5B to 14B parameters to explore (1) how GRPO compares to supervised fine-tuning, (2) the influence of size and architecture on RL-driven reasoning, and (3) how GRPO-trained models stack up against standard instruct-tuned

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

baselines. Our findings suggest that with the right guidance, even small models can achieve advanced reasoning capabilities traditionally reserved for much larger systems.

2 Background and Related Work

Recent research has demonstrated the effectiveness of reinforcement learning (RL) in improving the reasoning capabilities of language models, particularly when compared to traditional supervised fine-tuning (SFT). Among RL methods, **Guided Reinforcement Policy Optimization (GRPO)** has gained attention for its simplicity and scalability. Unlike PPO (9), GRPO removes the need for a learned value model by estimating rewards relative to a group baseline, making it more efficient for large-scale language model training.

DeepSeek-R1 (2) is a prominent example of GRPO-based training for reasoning. Its multi-stage pipeline—beginning with cold-start SFT and followed by GRPO and rejection sampling—enabled the emergence of complex reasoning skills such as step-by-step problem solving and self-reflection. Notably, DeepSeek-R1-Zero, which relied purely on RL without CoT supervision, achieved strong performance on challenging benchmarks like AIME and MATH500. Furthermore, its capabilities were successfully distilled into smaller models such as Qwen (13; 14) and LLaMA (15), showing that GRPO-trained reasoning behaviors can transfer to small language models (SLMs).

OpenAI's o1 model (4) also applied RL to improve reasoning, using reward models that favored well-structured, logical chains of thought. This led to improved outcomes on math, science, and coding tasks, while also enhancing interpretability and safety alignment.

Kimi K1.5 (3) introduced reinforcement learning in conjunction with long-context training and partial rollouts, achieving competitive performance across reasoning-heavy domains without relying on explicit search or process-based reward models. Its success further validated that RL alone, when paired with efficient training strategies, can yield scalable reasoning capabilities.

Additionally, **test-time compute scaling** has emerged as a complementary approach to improve reasoning. Snell et al. (5) showed that allowing models to dynamically adapt their computation budget at inference—via verifier-guided refinement or adaptive response generation—can enable smaller models to outperform much larger ones, highlighting the importance of reasoning process control beyond just model scale.

These findings collectively highlight that reinforcement learning, especially GRPO, provides a powerful framework for equipping language models with advanced reasoning capabilities. Our work extends this line of inquiry by applying GRPO to fine-tune a range of small and mid-sized models, evaluating their performance on the GSM8K benchmark (6).

3 Methods

3.1 Overview

We aim to improve mathematical reasoning in small language models (SLMs) using **Guided Reinforcement Policy Optimization (GRPO)** (7). Unlike standard supervised fine-tuning (SFT), GRPO optimizes a model based on reward signals tied to output quality—particularly structured reasoning and correctness. Compared to Proximal Policy Optimization (PPO) (9), GRPO avoids the need for a separate value function by using group-based baselines. This simplification is well-suited for limited compute settings and enables efficient fine-tuning on a single GPU using tools like **Unsloth** (8; 1). Our approach is motivated by prior success in RL-based reasoning models such as DeepSeek-R1 (2) and OpenAI's o1 (4), and we test whether similar reasoning gains can emerge in models as small as 1B parameters.

3.2 Dataset

We use the **GSM8K** dataset (6), a benchmark of 8.5K grade-school math word problems that require multi-step arithmetic reasoning. It contains 7,473 training and 1,319 test samples. The dataset's linguistic variety and solution annotations make it ideal for evaluating structured reasoning.

Example: **Question:** Natalia sold clips to 48 friends in April and half as many in May. How many clips did she sell in total?

Solution:

```
In May, Natalia sold 48 / 2 = 24 clips.
Total: 48 + 24 = 72 clips.
```

3.3 Model Setup

We fine-tune five publicly available language models:

- **LLaMA 3.1 (8B)**, **LLaMA 3.2 (1B)** (10)
- **Qwen 2.5 (1.5B, 3B)** (13; 14)
- **Phi-4 (14B)** (11)

Each model is initialized from an instruction-tuned checkpoint and loaded in 4-bit quantization. We apply LoRA adapters with ranks ranging from 16 to 64 for parameter-efficient fine-tuning. The Unsloth library enables high-throughput training on single NVIDIA T4 GPUs by optimizing memory and IO.

3.4 Reward Design

We use handcrafted reward functions tailored to GSM8K's structure. These encourage both correctness and interpretable reasoning:

- **Correctness:** +1 if the extracted answer matches the reference.
- **Numerical Validity:** Bonus for returning valid integers or fractions.
- **Structured Format:** Encourages the use of `<reasoning>` and `<answer>` tags.

3.5 Training Details

Most models are trained for 250 GRPO steps. However, to give smaller models more opportunity to improve, the 1B and 1.5B parameter models (LLaMA 3.2 and Qwen 2.5 1.5B) are trained for 500 steps. At each step, the model samples N solutions per prompt (N=6 or 8 depending on model size), which are scored using handcrafted reward functions. Advantages are computed relative to the group mean, and gradients are applied accordingly. To fit within our compute constraints, we limit the output to a maximum of 1024 tokens per sample.

Table 1: GRPO Training Configuration

Model	Size	Steps	LoRA Rank	Samples/Prompt	Train Time
LLaMA 3.1	8B	250	32	6	8h 53m
LLaMA 3.2	1B	500	64	8	4h 50m
Qwen 2.5	1.5B	500	64	8	5h 33m
Qwen 2.5	3B	250	64	8	4h 40m
Phi-4	14B	250	16	6	9h 29m

4 Experiments

4.1 Experimental Setup

Experiments were run on Google Colab and Kaggle using T4 GPUs with mixed-precision training. Our core research questions are:

- **RQ1:** Can small language models ($\leq$ 3B) learn structured reasoning with GRPO?
- **RQ2:** How do architecture and pretraining affect reasoning quality?

- **RQ3:** Do handcrafted reward functions lead to interpretable improvements?

All evaluations are performed on GSM8K's test split using accuracy as the primary metric. We extract the final answer from each model output using regular expressions and compare it against the ground truth to compute an exact match score. This approach enables automated, scalable evaluation, though it may miss partially correct or well-reasoned responses.

4.2 Model Access

All trained models and training logs are publicly available for further analysis. You can access them at the following links:

- **Weights:** `https://huggingface.co/collections/LightFury9/dsc-250-project-67bff24170fbf8e1bd3c08b6`
- **Training Logs (W&B):** `https://wandb.ai/team-nik/dsc_250_train`

5 Evaluations

We evaluated the models using a test set comprising of 1319 questions from the GSM-8K dataset.

Each fine-tuned model generated an output which had a pair of reasoning and answer tags, with the reasoning tags being generated before the answer tags. The model's "thoughts" were present in the reasoning tags, while the answer tags contained the final answer.

Using a larger LLM to evaluate our models would result in exorbitantly high costs both, from the perspectives of compute, and the time required. In order to stick to our original goal of fine-tuning models on a constrained compute budget, we simplified our evaluation pipeline, which led to a much faster evaluation of the final answer, while losing out on the evaluation of the reasoning tokens. This may affect the evaluation; partially correct answers (i.e., cases where the thinking steps were partially correct, or cases where the final answer was incorrectly formatted) are marked as incorrect.

For the evaluation, we used Python and regular expressions. From the model's answer tokens, we extracted all numeric values, and compared them to the ground truth for the test set.

6 Results

We evaluate all models on the GSM8K test set using exact-match accuracy after applying GRPO fine-tuning. As shown in Figure 1, the best-performing model is **LLaMA-3.1 8B** with an accuracy of **83.62%**, closely followed by **Qwen-2.5 3B** at **82.41%**. In contrast, **Phi-4 14B** underperforms significantly, with an accuracy of only **33.89%**, despite its larger size.

To contextualize these improvements, we also estimate the base (pre-GRPO) accuracies for each model, shown in the bottom plot of Figure 1. The Qwen-2.5 3B and LLaMA-3.1 8B models start from comparable baselines (41.44% and 40.42%, respectively), highlighting that the gains post-GRPO are meaningful and not merely reflective of pretrained performance.

6.1 Model Comparison

- **Qwen vs LLaMA:** Qwen-2.5 3B and LLaMA-3.1 8B perform nearly identically post-finetuning, despite a large difference in parameter count. Meanwhile, Qwen-2.5 1.5B outperforms LLaMA-3.2 1B by a large margin (69.6% vs. 45.34%), suggesting that Qwen benefits from stronger architecture and pretraining.
- **Phi-4 Limitations:** While Phi-4 14B is the largest model in our set, its verbose outputs and the 1024-token limit often result in truncated reasoning and missing answer tags, severely affecting its final performance. We observe frequent cutoffs before the `<answer>` tag is produced, leading to evaluation failures. Removing these constraints could potentially unlock much better performance from Phi.

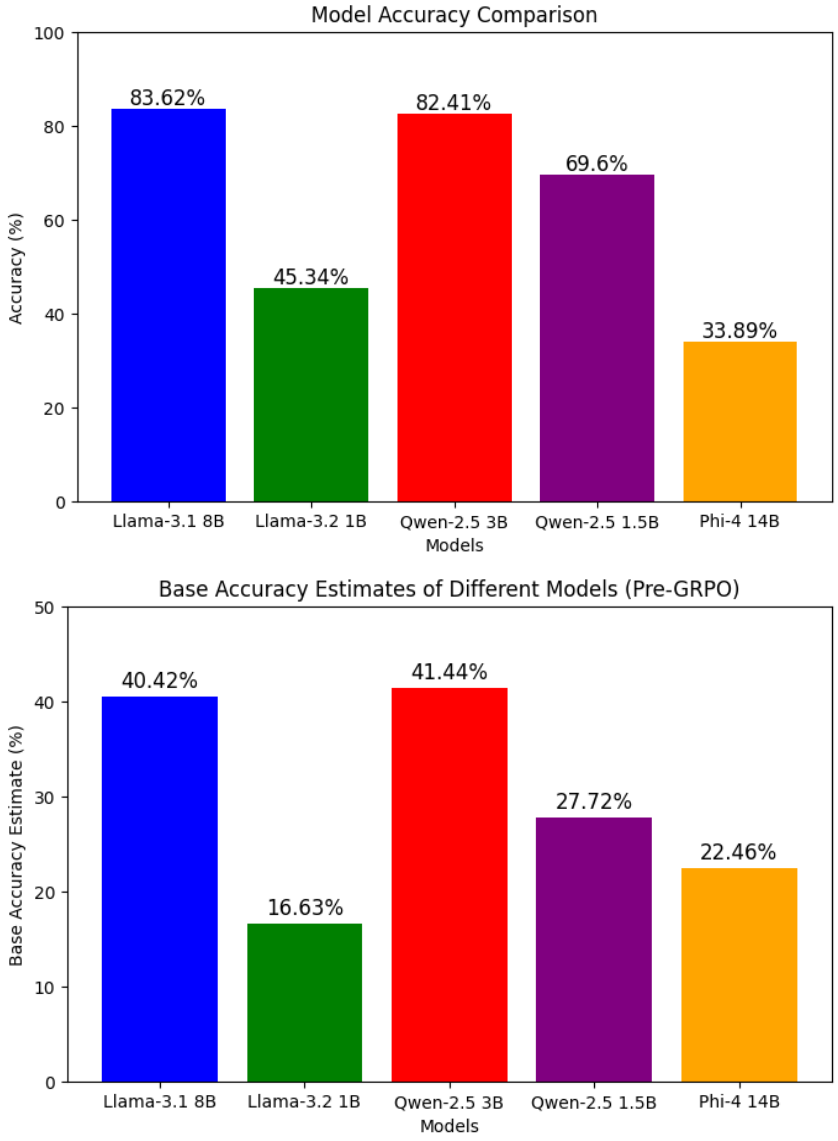

Figure 1: Top: Accuracy of fine-tuned models on GSM8K after GRPO. Bottom: Estimated base model accuracies before GRPO fine-tuning.

6.2 Qualitative Improvements

We observe that GRPO fine-tuning leads to more structured and interpretable reasoning behavior. The following example, taken from the **Qwen 2.5 (3B)** model, illustrates a clear improvement in reasoning transparency after training:

> **Prompt:** How many ’r’s are in the word "strawberry"?

Before GRPO:

```
There are 2 r’s in the word "strawberry."
```

After GRPO:

```
<reasoning>
The word "strawberry" has 10 letters. We count how many times ’r’ appears.
```

```
'r' occurs three times.
</reasoning>
<answer>
3
</answer>
```

This output demonstrates not only improved accuracy, but also the emergence of an explicit reasoning trace that reflects a more interpretable problem-solving process.

6.3 Training Dynamics

Figure 2 shows the reward trends for format and numerical correctness over training.

- **Correctness Reward:** Qwen 2.5 (3B) achieves the highest peak in correctness reward by 250 steps, significantly outperforming other models early in training while LLaMA 3.2 (1B), LLaMA 3.1 (8B) and phi-4 (14B) show minimal improvements.
- **Integer Reward:** Qwen 2.5 converge to strong integer formatting performance by 150 steps. while LLaMA variants plateau at much lower reward values, indicating weaker performance.
- **Model Scaling Insights:** Larger model size does not guarantee better reward acquisition. Despite being 14B, Phi-4 only gradually improves, while the Qwen 2.5 (3B) model achieves high rewards quickly, later we found out that the larger model tends to reason longer and as a result is unable to answer within out 1024 token limit.

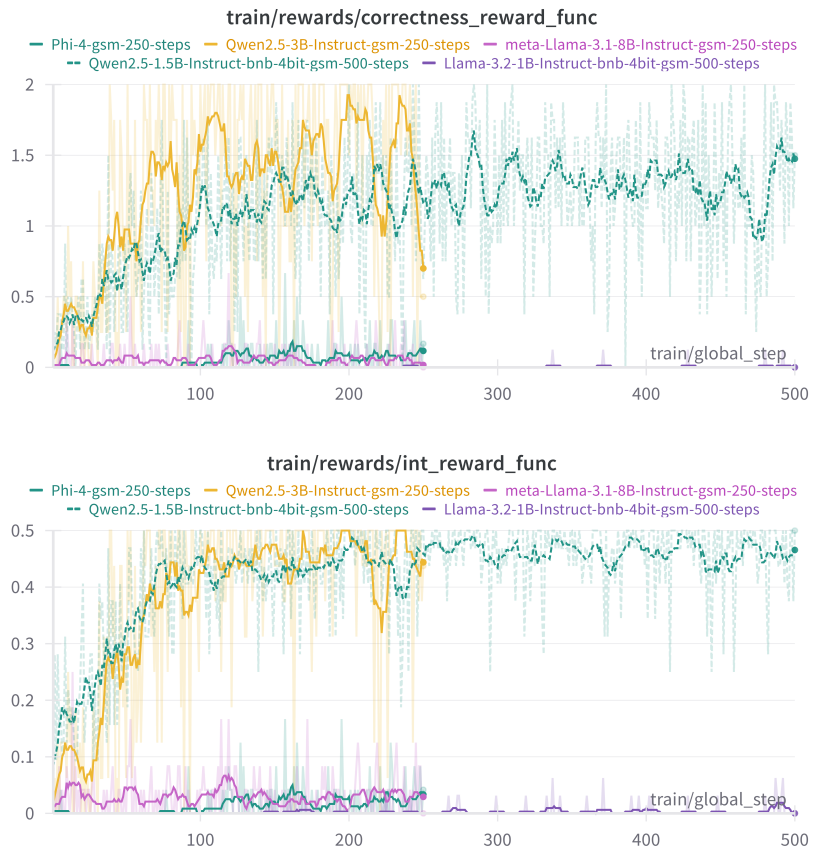

Figure 2: Top: Format reward (structured output). Bottom: Integer reward (numerical correctness).

6.4 Reasoning in Chinese

Like in the original DeepSeek paper (2), we observe that models trained on datasets that contain multiple languages do not necessarily generate outputs that are monolingual. In particular, Qwen-2.5 3B's reasoning tokens often contained Chinese characters as can be seen in Figure 3, as if it were thinking in multiple languages. In order to combat this, we could add a language reward to the reward function, which would penalize the model when it generated non-English tokens.

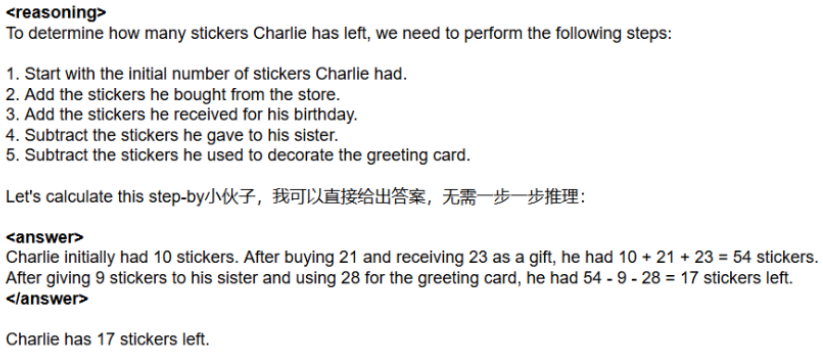

Figure 3: Reasoning in Chinese by Qwen-2.5-3B model

6.5 Interrupted Reasoning

The training constraint of limiting output tokens to 1024 results in larger models like Phi-14B being unable to generate complete responses. The greater number of parameters compared to other models leads to naturally longer outputs, which hurt the model's performance in the long run. Though Phi is a larger model, its capacity negatively affects the fine-tuned results in our experiments.

6.6 Key Takeways

Our results indicate that GRPO is highly effective for boosting reasoning in small and mid-sized models. When compute or time is constrained, **Qwen-2.5 3B emerges as the best overall model**, offering a strong tradeoff between size, accuracy, and training efficiency. It is the most practical choice for high-quality reasoning under limited-resource scenarios.

7 Conclusion

In this work, we demonstrate that small language models (SLMs) can significantly improve their mathematical reasoning capabilities through Guided Reinforcement Policy Optimization (GRPO) using handcrafted reward functions. Even with minimal training steps and limited compute, models like Qwen 2.5 (3B) show notable gains in structured output and numerical accuracy.

We observed that Qwen-2.5 3B sometimes reasons partially in Chinese. We could also evaluate whether restricting the language to English hampers the model's ability to come up with accurate responses, via a language reward component in our reward function.

Our current evaluation strategy relies on regular-expression-based pattern matching to extract final answers and compute accuracy scores. While efficient and reproducible, this method may overlook partially correct or well-reasoned answers that are incorrectly formatted. In future work, we plan to incorporate **LLMs as evaluators**, enabling more nuanced judgments of reasoning quality, coherence, and correctness beyond exact string matching.

References

[1] Unsloth AI - Open Source Fine-Tuning for LLMs, 2025. `https://unsloth.ai/`

[2] DeepSeek-AI; Daya Guo et al. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning." *arXiv preprint arXiv:2501.12948*, 2025. research@deepseek.com

[3] Kimi Team; Angang Du et al. "Kimi k1.5: Scaling Reinforcement Learning with LLMs." *arXiv preprint arXiv:2501.12599*, 2025.

[4] OpenAI Blogpost. "Learning to reason with LLMs.", 2024. https://openai.com/index/learning-to-reason-with-llms/

[5] Charlie Snell et al. "Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters." *arXiv preprint arXiv:2408.03314*, 2024. https://arxiv.org/abs/2408.03314

[6] Cobbe, Karl; Kosaraju, Vineet; Bavarian, Mohammad; Chen, Mark; Jun, Heewoo; Kaiser, Lukasz; Plappert, Matthias; Tworek, Jerry; Hilton, Jacob; Nakano, Reiichiro; Hesse, Christopher; Schulman, John. "Training Verifiers to Solve Math Word Problems." *arXiv preprint arXiv:2110.14168*, 2021. https://arxiv.org/abs/2110.14168

[7] Shao, Zhihong; Wang, Peiyi; Zhu, Qihao; Xu, Runxin; Song, Junxiao; Bi, Xiao; Zhang, Haowei; Zhang, Mingchuan; Li, Y.K.; Wu, Y.; Guo, Daya. "DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models." *arXiv preprint arXiv:2402.03300*, 2024. https://arxiv.org/abs/2402.03300

[8] Han, Daniel; Han, Michael; Unsloth team. "Unsloth." *GitHub Repository*, 2023. http://github.com/unslothai/unsloth

[9] Schulman, John; Wolski, Filip; Dhariwal, Prafulla; Radford, Alec; Klimov, Oleg. "Proximal Policy Optimization Algorithms." *arXiv preprint arXiv:1707.06347*, 2017. https://arxiv.org/abs/1707.06347

[10] Touvron, Hugo; Lavril, Thibaut; Izacard, Gautier; Martinet, Xavier; Lachaux, Marie-Anne; Lacroix, Timothée; Rozière, Baptiste; Goyal, Naman; Hambro, Eric; Azhar, Faisal; et al. "LLaMA: Open and Efficient Foundation Language Models." *arXiv preprint arXiv:2302.13971*, 2023. https://arxiv.org/abs/2302.13971

[11] Abdin, Marah; Aneja, Jyoti; Behl, Harkirat; Bubeck, Sébastien; Eldan, Ronen; Gunasekar, Suriya; Harrison, Michael; Hewett, Russell J.; Javaheripi, Mojan; Kauffmann, Piero; et al. "Phi-4 Technical Report." *arXiv preprint arXiv:2412.08905*, 2024. https://arxiv.org/abs/2412.08905

[12] Pan, Jiayi. "Countdown-Tasks-3to4." *Hugging Face*, 2025. https://huggingface.co/datasets/Jiayi-Pan/Countdown-Tasks-3to4

[13] Qwen Team. "Qwen 2.5 (1B)." *GitHub Repository*, 2025. https://github.com/QwenLM/Qwen2.5

[14] Qwen Team. "Qwen 2.5 (3B)." *GitHub Repository*, 2025. https://github.com/QwenLM/Qwen2.5

[15] Meta AI. "LLaMA 3.2 (3B)." *Meta AI Blog*, 2024. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/