Optuna vs Code Llama: Are LLMs a New Paradigm for Hyperparameter Tuning?

Supplementary Material

6. RMSE Comparison

6.1. Models from train dataset

Computer vision models. Table 2 presents a detailed comparison of RMSE values for each model and epoch under three conditions: baseline Optuna, Code Llama after the first fine-tuning cycle (FT Cycle 1), and Code Llama after the second fine-tuning cycle (FT Cycle 2). The final two columns (*Difference 1* and *Difference 2*) show how much RMSE changes relative to the baseline, with positive values indicating that Code Llama yields a *lower* RMSE than Optuna (i.e., an improvement), and negative values implying the baseline outperforms Code Llama.

A number of architectures exhibit immediate gains in the first fine-tuning cycle. For example, DenseNet, MobileNetV2, RegNet, ShuffleNet, SqueezeNet, SwinTransformer, and VGG consistently show positive differences across their epochs, indicating a clear reduction in RMSE relative to Optuna. ConvNeXt also demonstrates improvements in both epochs, confirming Code Llama's ability to quickly adapt to a variety of high-performance models. AlexNet shows a notable positive difference in epoch 2, reflecting early and meaningful gains in hyperparameter recommendations. In contrast, some models — such as EfficientNet and GoogLeNet (epoch 1) — register negative differences, implying that for certain configurations, Optuna remains more effective in finding lower-RMSE settings.

The second fine-tuning cycle further improves RMSE in several models. AlexNet continues to perform well, with both epochs showing additional gains over the first cycle. Similarly, GoogLeNet and SqueezeNet exhibit further improvements across both epochs, highlighting the benefits of iterative tuning. In models such as ConvNeXt and MobileNetV3 (epoch 2), the second cycle also yields enhanced performance compared to the first, mitigating earlier shortcomings.

However, not all models benefit equally from iterative LLM-based fine-tuning. For instance, MNASNet shows a drop in performance in the second epoch, indicating a potential limitation in its adaptability to the fine-tuning process. Similarly, ResNet and RegNet show a decline in RMSE improvements in the second cycle, pointing to possible instability in the learned hyperparameter patterns. These observations highlight the importance of carefully assessing trade-offs and potential limitations when applying fine-tuning to specific model types.

When comparing the fine-tuning results to the Optuna baseline, it becomes clear that fine-tuning effectively reduces RMSE for the majority of models and epochs — particularly for computationally demanding architectures, such as ConvNeXt, SwinTransformer, and VGG. These results affirm that fine-tuning is a robust optimization strategy capable of surpassing or complementing traditional hyperparameter search methods like Optuna for many computer vision tasks. At the same time, the variability across models and epochs underscores the need for architecture-specific tuning strategies to maximize the benefits of fine-tuning.

Text generation models. The analysis of RMSE differences for the first fine-tuning cycle, as shown in Table 3, provides detailed insight into the impact of fine-tuning on text generation models. For the RNN model, fine-tuning results in a small but consistent improvement in RMSE—approximately 0.0185 lower than the Optuna baseline at epoch 2 — indicating stable and beneficial adjustments from the LLM.

The LSTM model also shows improvement, reducing RMSE by about 0.0127 at epoch 1 compared to Optuna, although a minor performance drop is observed at epoch 2. This variation suggests that LSTM responds positively to fine-tuning overall, though tuning depth may affect consistency.

In contrast, the Llama model shows a slight decline in performance across both epochs, with RMSE values marginally worse than Optuna. This may point to the challenges of adapting large, complex architectures like Llama using the current tuning strategy.

6.2. Models from test dataset

The analysis of the RMSE differences between Optuna and the second fine-tuning cycle for models from the test dataset, as shown in Table 4, evaluates the impact of fine-tuning on previously unseen computer vision models.

Among the evaluated models, MaxVit and VisionTransformer show substantial improvements after fine-tuning. MaxVit achieves positive RMSE differences of 0.1919 and 0.2941 for epochs 1 and 2, respectively, while VisionTransformer improves by 0.1884 and 0.2406. These results indicate that the fine-tuning process significantly enhances the accuracy of these architectures, even though they were not part of the original training set. Such outcomes suggest that Code Llama generalizes well to new models, successfully adapting its hyperparameter recommendations to unfamiliar architectures.

Model	Epochs	RMSE Optuna	RMSE FT Cycle 1	Difference Cycle 1	RMSE FT Cycle 2	Difference Cycle 2
AlexNet	1	0.8258	0.8421	-0.0163	0.5779	0.2479
AlexNet	2	0.8236	0.6912	0.1324	0.4221	0.4015
ConvNeXt	1	0.7924	0.7460	0.0464	0.7493	0.0431
ConvNeXt	2	0.7426	0.6861	0.0565	0.6863	0.0563
DenseNet	1	0.4930	0.4526	0.0404	0.5899	-0.0969
DenseNet	2	0.3525	0.3325	0.0201	0.4307	-0.0782
EfficientNet	1	0.4835	0.6250	-0.1415	0.6498	-0.1663
EfficientNet	2	0.3370	0.5486	-0.2116	0.5563	-0.2193
GoogLeNet	1	0.5158	0.5865	-0.0707	0.5093	0.0065
GoogLeNet	2	0.3956	0.3857	0.0099	0.3044	0.0912
MNASNet	1	0.4613	0.5309	-0.0696	0.5132	-0.0519
MNASNet	2	0.3682	0.4308	-0.0626	0.4620	-0.0938
MobileNetV2	1	0.4654	0.4155	0.0499	0.4316	0.0338
MobileNetV2	2	0.3617	0.2854	0.0763	0.4279	-0.0662
MobileNetV3	1	0.4852	0.7328	-0.2476	0.8186	-0.3334
MobileNetV3	2	0.3368	0.6797	-0.3429	0.6567	-0.3199
RegNet	1	0.5677	0.5392	0.0285	0.6254	-0.0577
RegNet	2	0.4368	0.3865	0.0503	0.5484	-0.1116
ResNet	1	0.4692	0.5153	-0.0461	0.5847	-0.1155
ResNet	2	0.3371	0.4325	-0.0954	0.5601	-0.2230
ShuffleNet	1	0.4832	0.4282	0.0550	0.5222	-0.0390
ShuffleNet	2	0.3471	0.3236	0.0235	0.4638	-0.1167
SqueezeNet	1	0.8597	0.7200	0.1397	0.6888	0.1709
SqueezeNet	2	0.8532	0.6676	0.1856	0.5829	0.2703
SwinTransformer	1	0.7500	0.6527	0.0973	0.6746	0.0754
SwinTransformer	2	0.6726	0.5676	0.1050	0.6237	0.0489
VGG	1	0.8192	0.5898	0.2295	0.5364	0.2829
VGG	2	0.7961	0.3741	0.4220	0.3743	0.4218

Table 2. Comparison of RMSE differences for fine-tuning cycles 1 and 2 for each computer vision model and epoch from the training dataset. The table highlights the RMSE values obtained using Optuna as a baseline (RMSE Optuna) and compares them with the RMSE values achieved during the first and second fine-tuning cycles of Code Llama (RMSE FT Cycle 1 and RMSE FT 2, respectively). The Difference Cycle 1 and Difference Cycle 2 columns indicate the changes in RMSE relative to Optuna for the first and second fine-tuning cycles, respectively. Positive values in the difference columns represent improvements, while negative values indicate an increase in RMSE.

In contrast, InceptionV3 demonstrates a notable decline in performance following the second fine-tuning cycle. RMSE values increase by –0.2363 and -0.3364 for epochs 1 and 2, respectively, compared to the Optuna baseline. While these results indicate that the current fine-tuning setup may not fully optimize InceptionV3, they also point to valuable opportunities for future enhancement. With refined prompt design, adjusted parameter initialization, or extended fine-tuning schedules, there is strong potential to better align the model's architecture with the optimization process and achieve improved outcomes.

7. Best Accuracy Comparison

7.1. Models from train dataset

Table 5 presents a detailed result of the best accuracy achieved by Optuna and fine-tuned Code Llama across two fine-tuning cycles for a diverse range of computer vision models and epochs.

The fine-tuning process demonstrates its ability to enhance accuracy for many models, with notable improvements observed during the second cycle. For instance, MNASNet and MobileNetV2 achieve significant gains in the first epoch of the second fine-tuning cycle, underscoring the effectiveness of fine-tuning in refining hyperparameters to better align with the models' architectures. Similarly, RegNet and SqueezeNet show consistent accuracy improve-

Model	Epochs	RMSE Optuna	RMSE FT 1	Difference
RNN	1	0.7959	0.7880	0.0079
RNN	2	0.7888	0.7703	0.0185
LSTM	1	0.7691	0.7564	0.0127
LSTM	2	0.7468	0.7478	-0.0010
Llama	1	0.9988	0.9990	-0.0002
Llama	2	0.9987	0.9991	-0.0004

Table 3. Comparison of RMSE values for text generation models from the training dataset during fine-tuning cycle 1. The table highlights the differences in RMSE between the baseline results (Optuna) and the fine-tuned results (FT 1). Positive differences indicate an improvement in RMSE after fine-tuning, while negative differences suggest a potential decline in performance.

ments across cycles, particularly excelling in the second cycle, showcasing the iterative benefits of Code Llama's fine-tuning framework.

The results also highlight the stability achieved through fine-tuning, as models such as DenseNet and ResNet demonstrate consistent accuracy across multiple epochs and cycles. This stability reflects the robustness of Code Llama's fine-tuning methodology, positioning it as a reliable tool for hyperparameter optimization. In many cases, the fine-tuning process achieves accuracy results comparable to or exceeding those of Optuna, reinforcing its utility as a competitive alternative.

While certain models, such as MobileNetV3 and SwinTransformer, exhibit moderate or variable improvements across cycles, these findings underscore the importance of tailoring fine-tuning strategies to the specific requirements of each architecture. For models that are well-optimized in the initial cycle, additional fine-tuning may yield diminishing returns, suggesting the need for a balanced approach that considers computational costs alongside expected performance gains. This variability highlights the importance of model-specific strategies in leveraging the full potential of fine-tuning.

7.2. Models from test dataset

The results presented in Table 6 provide a detailed evaluation of the accuracy differences between the baseline Optuna approach and the second fine-tuning cycle (FT Cycle 2) for models in the test dataset.

The fine-tuning process introduces significant changes in model accuracy, with varying results depending on the architecture. For InceptionV3, fine-tuning in FT Cycle 2 results in a notable reduction in accuracy compared to Optuna, with differences of -0.3001 for one epoch and -0.2423 for two epochs. These findings suggest that while fine-tuning significantly adjusts the model's parameters, it may not always align with the specific optimization needs of complex architectures like InceptionV3. This highlights the potential necessity of more customized hyperparameter tuning strategies for such models to better leverage fine-tuning.

For VisionTransformer, the differences between Optuna and FT Cycle 2 are less pronounced, with values of -0.0523 for one epoch and -0.0427 for two epochs. This stability in performance suggests that the second cycle of fine-tuning effectively preserves the model's baseline accuracy while introducing modest refinements. The relatively small variations indicate that VisionTransformer may require fewer fine-tuning adjustments to reach or maintain optimal performance, reflecting its robustness to the fine-tuning process.

MaxVit demonstrates the most consistent and positive results among the evaluated models. The difference is minimal for one epoch (-0.0018), and for two epochs, FT Cycle 2 outperforms Optuna with a positive difference of 0.0249. This improvement underscores MaxVit's adaptability to fine-tuning, highlighting its potential for further optimization.

8. Prediction Dynamics Across Epochs

The results of one-shot predictions across 13 computer vision models, which are not covered in Section 4.4 of this paper, for different numbers of epochs (1, 2, 5, 10, 15, and 20), shown in Figures 6-7, demonstrate a clear trend of increasing accuracy with additional training epochs for these models. This improvement reflects the expected behavior, as extended training allows models to refine their predictions and optimize performance. For many architectures, including ConvNeXt, ShuffleNet, and ResNet, accuracy stabilizes after 10 epochs, indicating that these models reach their optimal performance within this training range.

Interestingly, models such as GoogLeNet, ShuffleNet, and DenseNet exhibit high accuracy even at early epochs, showcasing their ability to converge quickly and effectively. These results highlight their potential for efficient training in scenarios where computational resources or time are limited. Additionally, one-shot predictions for several models, such as MobileNetV3 and EfficientNet, reveal that their learning dynamics can vary across epochs, reflecting the unique characteristics of their architectures and the data.

However, not all models exhibit consistent improvements in accuracy. For instance, MobileNetV3 shows a significant drop in performance after 15 epochs before recovering at 20 epochs, suggesting potential overfitting or instability during certain training phases. Similarly, InceptionV3, which originates from the test dataset and was not included in the fine-tuning process of Code Llama, struggles to maintain accuracy gains as training progresses, with a notable decline in performance by the 20th epoch. These results indicate that certain architectures, particularly those not exposed to fine-tuning, may face challenges in sustaining accuracy improvements across extended training, poten-

Model	Epochs	RMSE Optuna	RMSE FT 2	Difference
InceptionV3	1	0.5906	0.8269	-0.2363
InceptionV3	2	0.4323	0.7687	-0.3364
MaxVit	1	0.8349	0.6430	0.1919
MaxVit	2	0.8297	0.5356	0.2941
VisionTransformer	1	0.7768	0.5884	0.1884
VisionTransformer	2	0.7659	0.5253	0.2406

Table 4. Comparison of RMSE differences between Optuna and Fine-tuning Cycle 2 for models from the test dataset. Positive differences indicate an improvement in RMSE after fine-tuning, while negative differences suggest a potential decline in performance.

Model	Epochs	Best Accuracy Optuna	Best Accuracy FT Cycle 1	Difference Cycle 1	Best Accuracy FT Cycle 2	Difference Cycle 2
MNASNet	1	0.6449	0.6390	-0.0059	0.7184	0.0735
MNASNet	2	0.7480	0.7476	-0.0004	0.7420	-0.0060
MobileNetV2	1	0.6452	0.6424	-0.0028	0.7365	0.0913
MobileNetV2	2	0.7493	0.7457	-0.0036	0.7446	-0.0047
AlexNet	1	0.5364	0.5392	0.0028	0.6282	0.0918
AlexNet	2	0.6640	0.6623	-0.0017	0.6680	0.0040
MobileNetV3	1	0.6322	0.6228	-0.0094	0.6174	-0.0148
MobileNetV3	2	0.7488	0.7314	-0.0174	0.7149	-0.0339
ConvNeXt	1	0.3454	0.3393	-0.0061	0.3020	-0.0434
ConvNeXt	2	0.4085	0.3962	-0.0123	0.4119	0.0034
VGG	1	0.5829	0.5859	0.0030	0.5804	-0.0025
VGG	2	0.7031	0.6828	-0.0203	0.6973	-0.0058
SwinTransformer	1	0.4548	0.4365	-0.0183	0.4345	-0.0203
SwinTransformer	2	0.5311	0.5180	-0.0131	0.5029	-0.0282
DenseNet	1	0.6316	0.6177	-0.0139	0.6216	-0.0100
DenseNet	2	0.7438	0.7420	-0.0018	0.7461	0.0023
SqueezeNet	1	0.4017	0.3848	-0.0169	0.4006	-0.0011
SqueezeNet	2	0.4742	0.4662	-0.0080	0.4820	0.0078
EfficientNet	1	0.6170	0.6198	0.0028	0.5935	-0.0235
EfficientNet	2	0.7374	0.7387	0.0013	0.7037	-0.0337
GoogLeNet	1	0.6405	0.6746	0.0341	0.6538	0.0133
GoogLeNet	2	0.7494	0.7402	-0.0092	0.7630	0.0136
ShuffleNet	1	0.6346	0.6369	0.0023	0.6321	-0.0025
ShuffleNet	2	0.7185	0.7246	0.0061	0.7177	-0.0008
RegNet	1	0.5289	0.5331	0.0042	0.5466	0.0177
RegNet	2	0.6498	0.6523	0.0025	0.6629	0.0131
ResNet	1	0.6255	0.6241	-0.0014	0.6282	0.0027
ResNet	2	0.7378	0.7399	0.0021	0.7399	0.0021

Table 5. Comparison of the best accuracy differences for fine-tuning cycles 1 and 2 across various computer vision models and epochs, alongside the best accuracy achieved using Optuna. Each row represents a model evaluated for 1 or 2 epochs, comparing the performance of fine-tuned models (FT Cycle 1 and FT Cycle 2) with the baseline Optuna results. Positive values in the Difference columns (Difference Cycle 1 and Difference Cycle 2) indicate an improvement in accuracy after fine-tuning, while negative values suggest a decrease in performance.

tially requiring more tailored hyperparameter adjustments or regularization strategies.

Across all models, it is evident that fine-tuned hyperparameters provided by Code Llama contribute to strong prediction performance. Notably, GoogLeNet, ShuffleNet, and ResNet display consistently high stability and accuracy

Model	Epochs	Best Accuracy Optuna	Best Accuracy FT Cycle 2	Difference
InceptionV3	1	0.5137	0.2136	-0.3001
InceptionV3	2	0.6818	0.4395	-0.2423
VisionTransformer	1	0.4715	0.4192	-0.0523
VisionTransformer	2	0.533	0.4903	-0.0427
MaxVit	1	0.4323	0.4305	-0.0018
MaxVit	2	0.5096	0.5345	0.0249

Table 6. Comparison of the best accuracy between Optuna and Fine-tuned (FT Cycle 2) models for test dataset architectures. Each row represents a specific model evaluated after 1 or 2 epochs, comparing the best accuracy achieved using Optuna with the accuracy obtained after the second cycle of fine-tuning (FT Cycle 2). The Difference column highlights the variation in performance, where positive values indicate an improvement in accuracy after fine-tuning, and negative values suggest a decline.

across the evaluated epochs, emphasizing their robustness in various training settings. Even for architectures that show more variability, such as MobileNetV3 and MNASNet, the overall trend demonstrates the ability of Code Llama to enhance model performance efficiently.

These findings demonstrate the value of one-shot predictions and emphasize the need to account for the specific characteristics of each model when designing training strategies. By using fine-tuned hyperparameters and adjusting training durations to suit the unique requirements of each architecture, it becomes possible to achieve high accuracy while efficiently utilizing computational resources.

9. Training Trends and Performance Comparison

The analysis of the accuracy performance of 14 computer vision models from the train dataset, each trained on 1 and 2 epochs (resulting in 28 graphs), excludes six graphs already covered in Section 4.5 of this paper. The remaining 22 graphs, which are not presented in the main text, are illustrated in Figures 8-10, showcasing key performance dynamics across models trained with Optuna over 100 trials, fine-tuned Code Llama, and one-shot predictions.

Fine-tuned Code Llama consistently delivers superior accuracy compared to Optuna's results, as reflected by the blue line surpassing the scattered green points for almost all models. This highlights the fine-tuned model's capability to optimize hyperparameters effectively. Pronounced improvements are observed in models like ResNet, ShuffleNet, and DenseNet, where the fine-tuning achieves significantly higher accuracy than Optuna's best results.

Fine-tuned Code Llama also offers enhanced stability. Models like ConvNeXt, EfficientNet, and SwinTransformer, which show considerable variability in Optuna's accuracy values, achieve more consistent performance with Code Llama. This consistency is particularly valuable in applications requiring dependable predictions across different conditions.

Increasing the number of epochs from one to two leads to expected accuracy improvements across all models. This trend is especially evident in models like EfficientNet and MobileNetV3, where additional training allows the parameters to refine further, resulting in higher accuracy.

One-shot predictions (purple line) demonstrate strong accuracy potential with minimal computational effort. In EfficientNet and MobileNetV3, the one-shot accuracy is nearly on par with the best results achieved through fine-tuning, emphasizing its viability as a quick and efficient method for generating robust predictions.

Certain models, such as SwinTransformer and SqueezeNet, show marked benefits from fine-tuning, overcoming the variability in their performance observed with Optuna. Code Llama fine-tuning reduces performance fluctuations and reliably enhances accuracy, even for architectures that are initially more challenging to optimize.

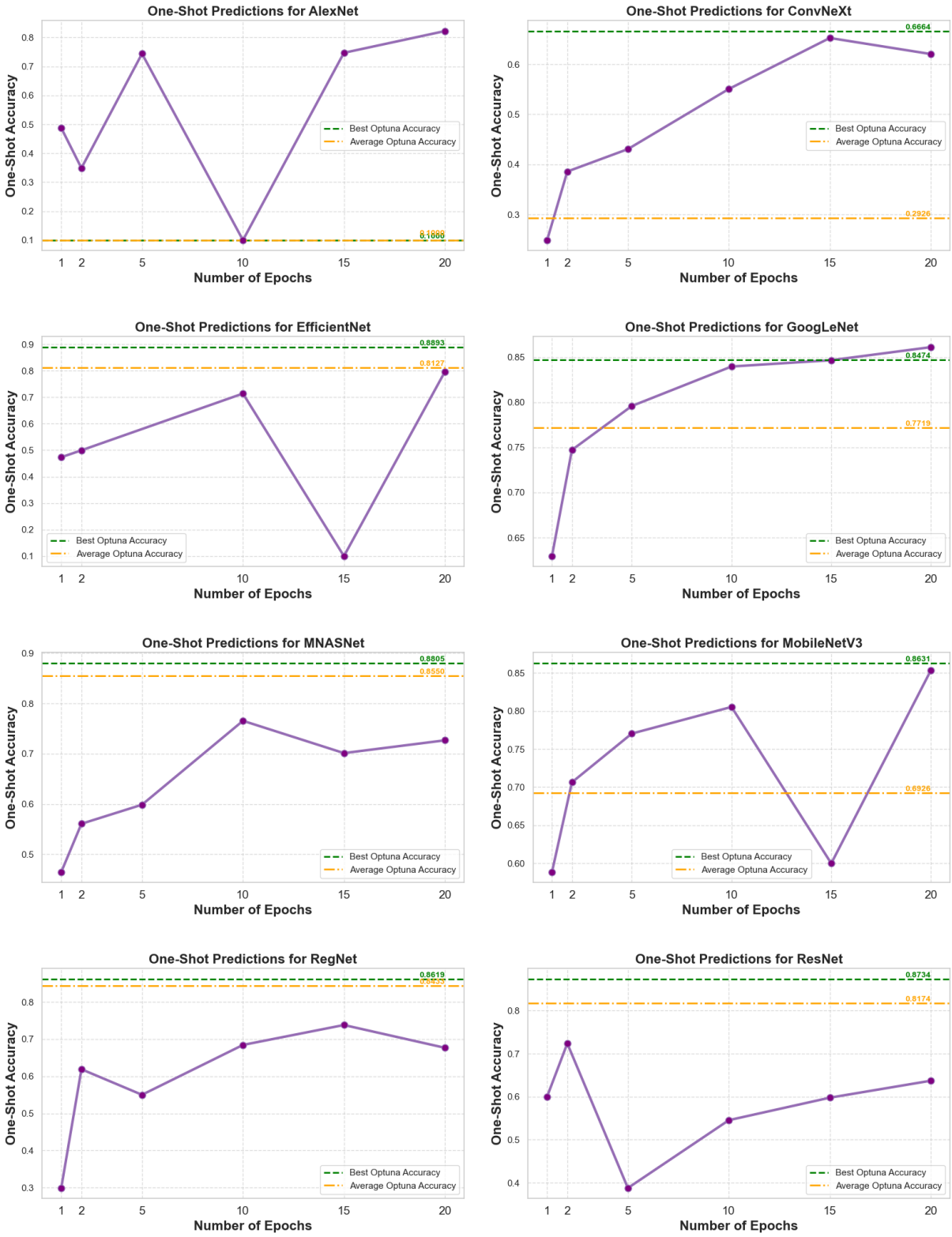

Figure 6. Part 1: Accuracy trends observed in one-shot predictions for 13 computer vision models over varying epochs (1, 2, 5, 10, 15, and 20). Each plot represents the accuracy performance of a specific model using the one-shot prediction approach. All models in this part are derived from the train dataset, demonstrating the trends within the training data context.

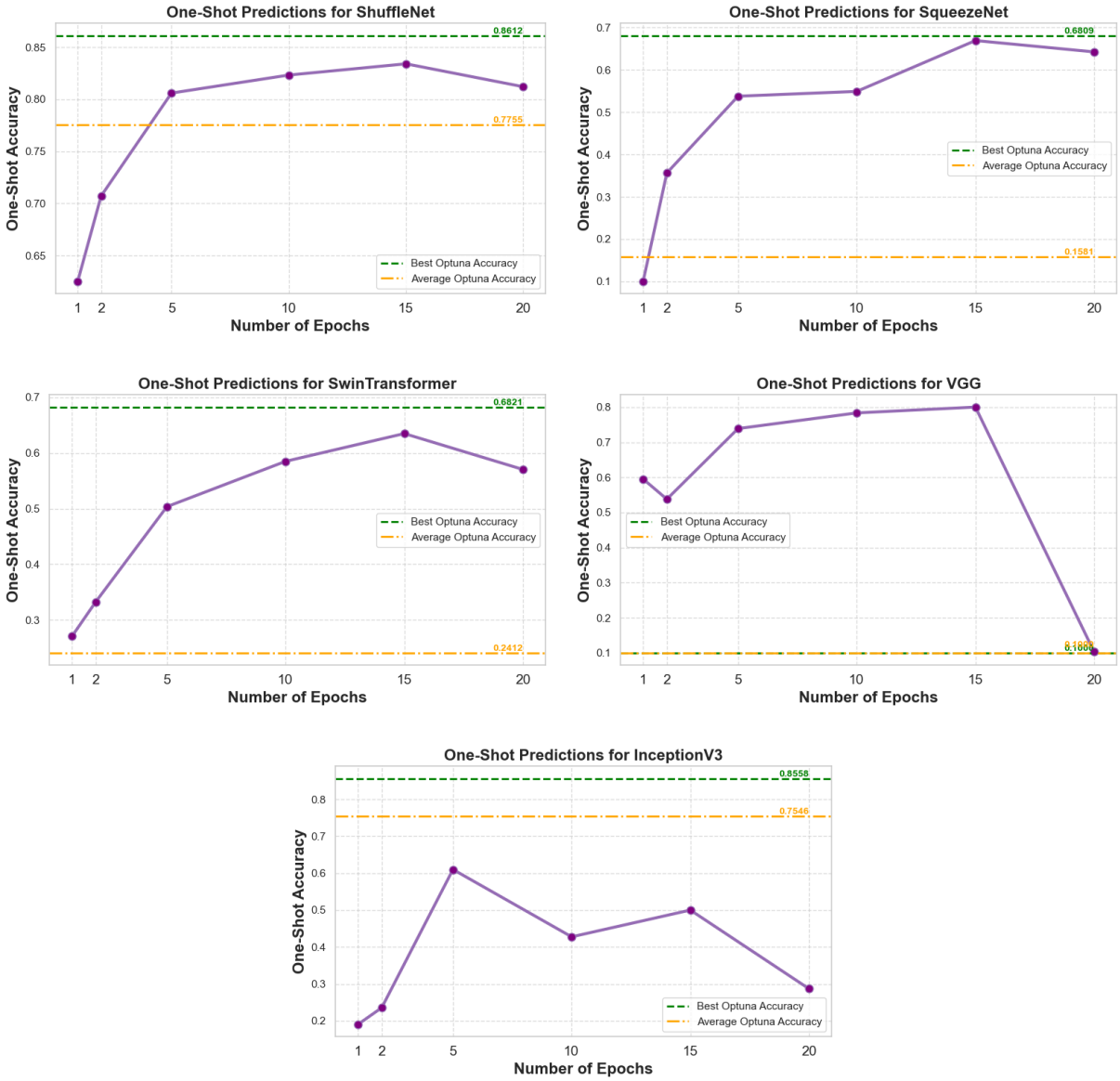

Figure 7. Part 2: Accuracy trends observed in one-shot predictions for 13 computer vision models over varying epochs (1, 2, 5, 10, 15, and 20). Each plot represents the accuracy performance of a specific model using the one-shot prediction approach. All models in this part are derived from the train dataset, except for InceptionV3, which is taken from the test dataset and was not involved in the fine-tuning of Code Llama.

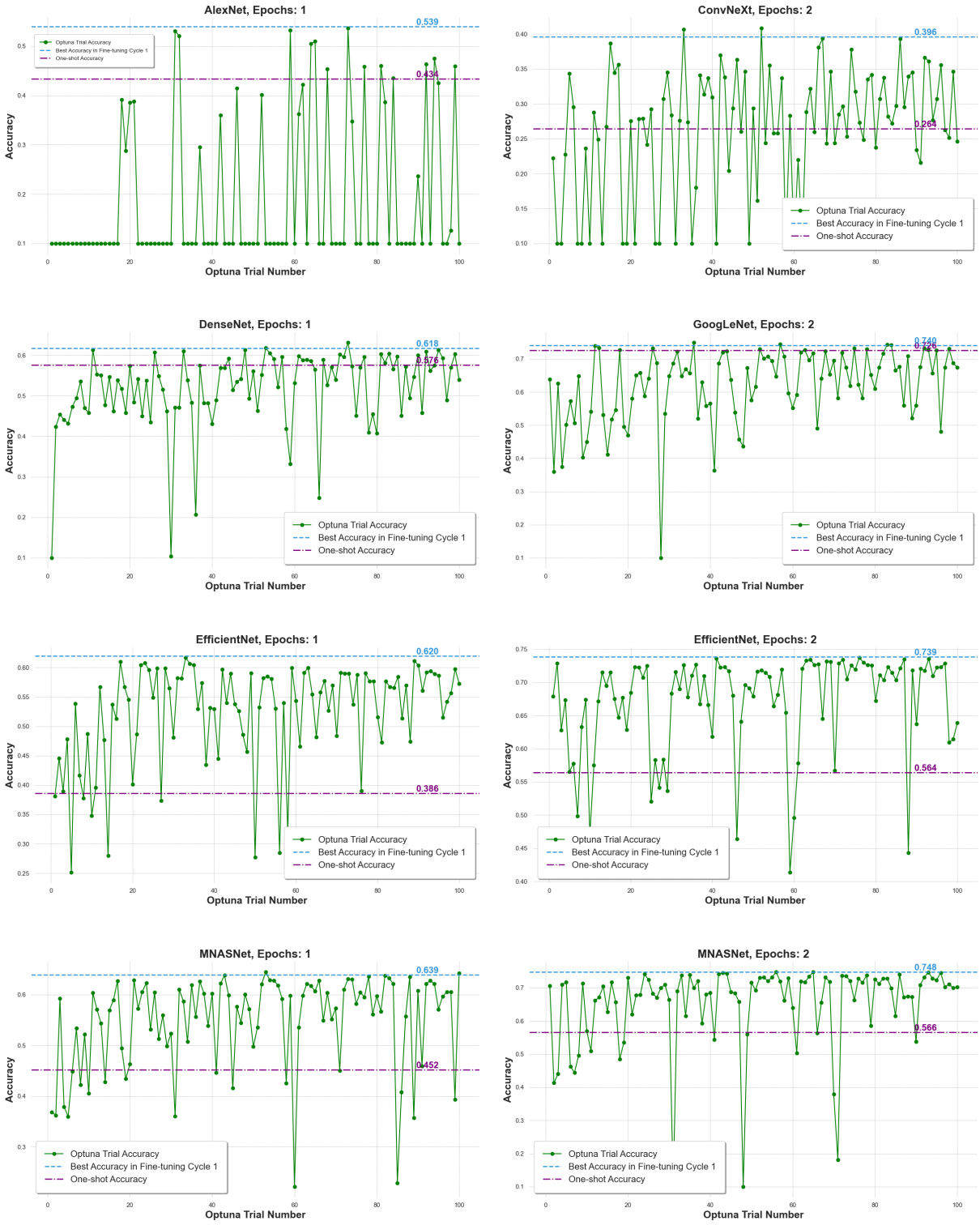

Figure 8. Part 1: Comparative analysis of accuracy performance for computer vision models evaluated over 100 trials using three distinct approaches: Optuna (green lines), fine-tuning with hyperparameters derived from Code Llama in Cycle 1 (blue lines), and one-shot predictions based on Code Llama (purple dashed lines). Each subplot represents a specific model and epoch configuration, illustrating variations in accuracy metrics and highlighting comparative trends.

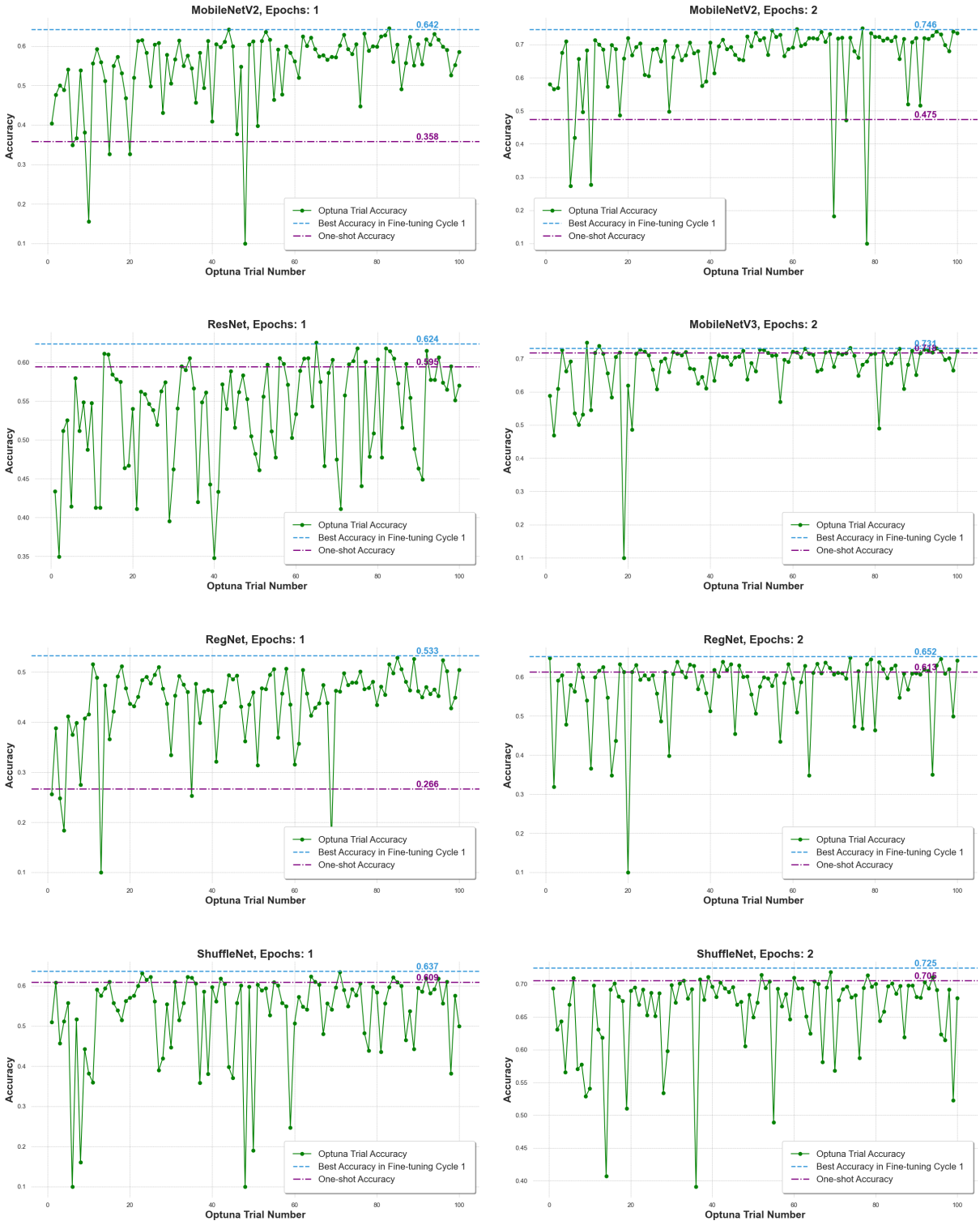

Figure 9. Part 2: Comparative analysis of the accuracy performance for the second set of computer vision models evaluated over 100 trials using three distinct approaches: Optuna (green lines), fine-tuning with hyperparameters derived from Code Llama in Cycle 1 (blue lines), and one-shot predictions based on Code Llama (purple dashed lines). Each subplot represents a specific model and epoch configuration, illustrating the variations in accuracy metrics and the relative performance across methodologies.

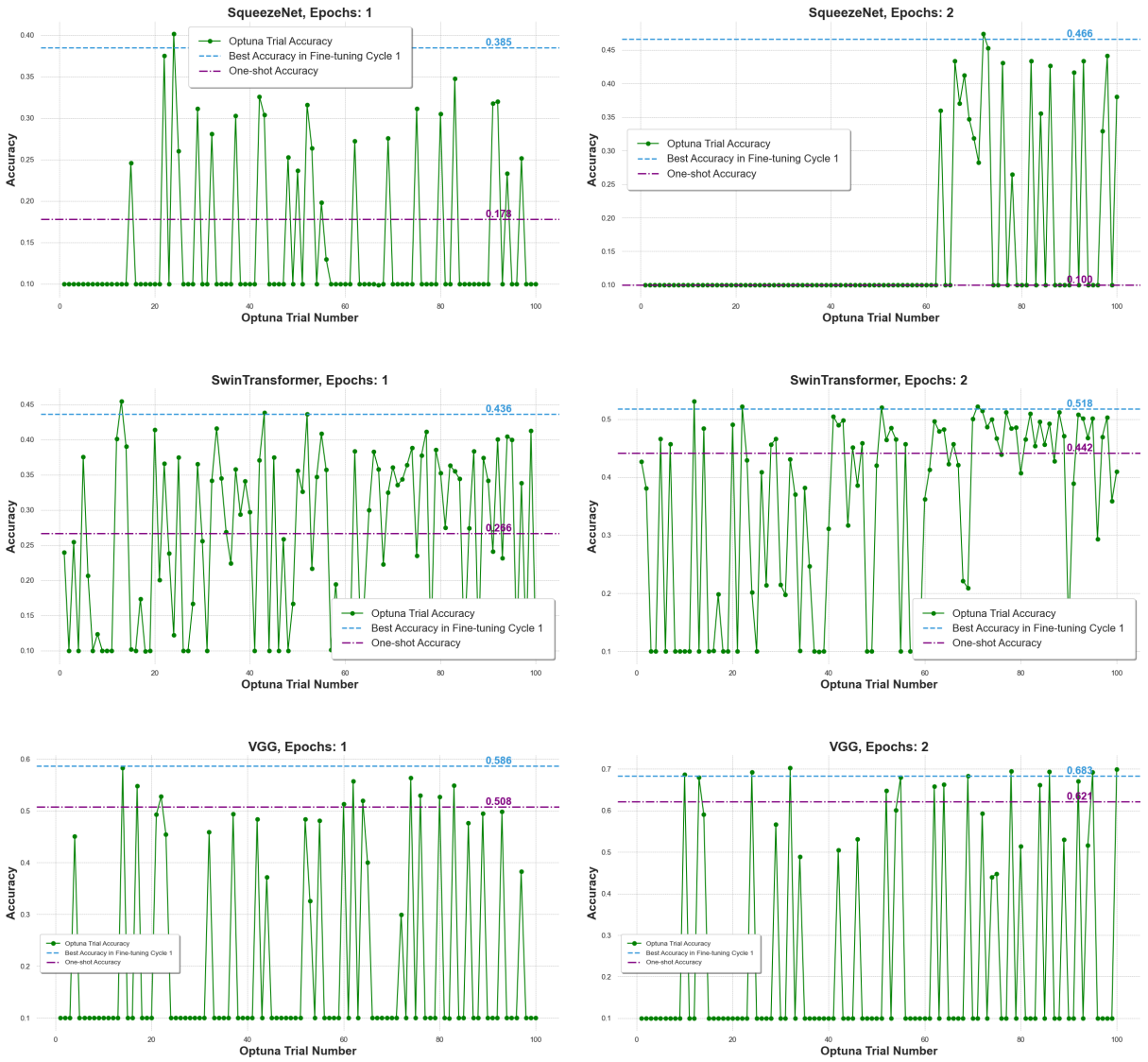

Figure 10. Part 3: Final analysis of accuracy dynamics for remaining computer vision models evaluated across 100 trials using three different approaches: Optuna (green lines), fine-tuning with hyperparameters obtained from Code Llama in Cycle 1 (blue lines), and one-shot predictions based on Code Llama (purple dashed lines). Each subplot corresponds to a specific model and epoch configuration, highlighting the variations in accuracy metrics.