Configuration Performance Learning for x264 Encoder: A Comparative Analysis of Linear Regression and LightGBM

Configuration Performance Learning Research

March 27, 2025

1 Introduction

Configuration performance learning helps optimize modern computer systems. For video encoding software like x264, selecting optimal configuration parameters impacts encoding efficiency and quality. Accurate prediction of performance metrics based on configurations allows users to optimize settings without extensive testing.

This study predicts x264 configuration performance using machine learning. We compare a linear regression model with LightGBM, a gradient boosting approach. The x264 encoder (version baee400) has 25 configuration options with 3,113 total configurations. We test performance across nine video files including blue_sky_1080p25_short.y4m and Johnny_1280x720_60_short.y4m.

Video encoder configuration spaces are complex, with parameters showing non-linear relationships. This complexity lets us evaluate whether advanced machine learning methods like gradient boosting outperform traditional linear models in capturing these patterns.

2 Related Work

Several research efforts have explored performance prediction for configurable systems.

Siegmund et al. [1] developed performance-influence models for software systems that quantify how configuration options affect performance. They combined sampling strategies with machine learning to predict performance across configuration spaces.

For linear regression approaches, Hutter et al. [2] applied this technique to predict performance in highly configurable systems, showing its effectiveness as a baseline for performance modeling despite limitations with non-linear relationships.

In the video encoding domain, Murillo-Morera et al. [3] used machine learning to predict encoding parameters in HEVC, demonstrating that decision trees effectively model the configuration space.

For gradient boosting methods, Ke et al. [4] introduced LightGBM, showing significant improvements over standard gradient boosting machines through techniques like Gradient-based One-Side Sampling and Exclusive Feature Bundling.

Configuration Performance Learning has advanced with Chen et al. [5], who employed transfer learning to improve performance prediction across environments. Similarly, Jamshidi et al. [6] developed transfer learning techniques specifically for configurable systems.

Our work extends this research by comparing linear regression against gradient boosting to capture complex patterns in x264 configuration performance.

3 Solution Approach

We implement two machine learning approaches for predicting x264 encoder configuration performance:

3.1 Baseline: Linear Regression Model

Linear regression models relationships between dependent and independent variables by minimizing squared residuals. Our model predicts encoding performance based on x264 configuration parameters using this equation:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \tag{1}$$

Where $\hat{y}$ is the predicted performance, β_0 is the intercept, $\beta_1 - \beta_n$ are coefficients, and $x_1 - x_n$ are configuration parameters. Linear regression is efficient and interpretable but assumes linear relationships between features and targets.

3.2 Proposed Approach: LightGBM

To address linear regression's limitations, we implement LightGBM, a gradient boosting framework using tree-based learning algorithms. LightGBM handles large datasets with high dimensionality efficiently.

Key advantages include: (1) Gradient-based One-Side Sampling focusing on instances with larger gradients; (2) Exclusive Feature Bundling reducing dimensionality; (3) Leaf-wise tree growth expanding from leaves with maximum delta loss; and (4) Histogram-based algorithms bucketing continuous features to accelerate training.

Our implementation uses these hyperparameters: regression objective, gradient boosting decision tree type, 31 leaves, 0.05 learning rate, 0.9 feature fraction, and 100 boosting rounds.

4 Experimental Setup

4.1 Dataset

We use the x264 dataset containing various video configurations and performance metrics. Each CSV file represents different video content (like Johnny_1280x720_60_short.y4m). Each row shows a specific x264 encoder configuration, with parameters as features and runtime as the target variable.

4.2 Implementation

We implemented both models in Python using scikit-learn for linear regression and the LightGBM library for gradient boosting. Our implementation: (1) loads CSV files with pandas, (2) splits data into 80% training and 20% testing sets, (3) trains both models, and (4) evaluates them using multiple metrics.

4.3 Evaluation Metrics

We use three standard regression metrics:

- Mean Absolute Error (MAE): Average magnitude of prediction errors
- Root Mean Squared Error (RMSE): Square root of average squared differences between predictions and observations
- Mean Absolute Percentage Error (MAPE): Percentage error providing relative prediction accuracy

4.4 Experimental Procedure

To ensure robust results, we conducted 30 runs with different random seeds for train-test splits. This accounts for variance due to data partitioning. For each run, we recorded MAE, RMSE, and MAPE, then calculated mean and standard deviation across all runs.

4.5 Statistical Analysis

To validate our results, we performed statistical testing to ensure that the performance differences between Linear Regression and LightGBM are significant. We employed paired t-tests with a significance level of $\alpha = 0.05$ to compare the paired MAE, RMSE, and MAPE values from both models across all datasets. The null hypothesis H_0 states that there is no significant difference between the performance of Linear Regression and LightGBM, while the alternative hypothesis H_1 states that LightGBM performs significantly better.

5 Results and Discussion

Table 1: Comparison of Linear Regression and LightGBM Performance Metrics

Dataset	Linear Regression			LightGBM		
	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)
Netflix Cross.	8.67±0.24	11.21±0.34	77.28±5.24	3.51±0.24	7.06±0.39	15.37±0.78
Riverbed	6.65±0.19	8.83±0.30	65.22±4.11	2.81±0.19	5.75±0.34	15.05±0.71
SD Crew	0.24±0.01	0.32±0.01	61.67±3.59	0.10±0.01	0.20±0.01	13.20±0.70
SD Bridge	0.14±0.00	0.18±0.01	51.38±2.73	0.06±0.00	0.11±0.01	11.23±0.50
Johnny	2.55±0.07	3.33±0.11	87.46±6.13	1.02±0.07	2.12±0.12	16.24±0.86
Blue Sky	3.12±0.08	4.04±0.12	77.26±5.30	1.27±0.09	2.57±0.14	15.44±0.79
Pedestrian	3.55±0.10	4.61±0.14	70.71±4.65	1.47±0.10	2.99±0.16	15.07±0.77
SD City	1.47±0.04	1.90±0.05	81.96±5.55	0.58±0.04	1.21±0.06	15.49±0.75
Sintel	0.37±0.01	0.46±0.01	50.92±2.70	0.16±0.01	0.31±0.02	12.36±0.63

The results in Table 1 presents the prediction performance of Linear Regression and LightGBM across various datasets. We observe that LightGBM consistently outperforms Linear Regression in all three metrics: MAE, RMSE, and MAPE.

For MAE, LightGBM achieves an average reduction of approximately 53%, with per-dataset improvements ranging from 59% to 61%. In terms of RMSE, the average reduction is around 38%, aligning well with the reported 37–39% improvement. The most significant gains are observed in MAPE, where LightGBM reduces the error from 51–87% to just 11–16%, corresponding to an average relative improvement of about 80%.

This substantial performance gap highlights the non-linear nature of the x264 encoder configuration space. Linear Regression, being a linear model, fails to capture these complex patterns, resulting in high prediction errors—especially for MAPE. In contrast, LightGBM, with its tree-based structure, is capable of modeling non-linear dependencies, thus providing accurate and robust predictions.

The high MAPE values of Linear Regression (often exceeding 70%) make it unsuitable for practical use. On the other hand, LightGBM consistently maintains MAPE values below 17%, which is acceptable in real-world performance prediction scenarios.

5.1 Analysis of Results

The superior performance of LightGBM can be attributed to several factors:

1. **Non-linear modeling capability**: LightGBM can capture non-linear relationships between configuration parameters and performance metrics, which are common in video encoding.

2. **Feature interactions**: Through its tree-based structure, LightGBM naturally models interactions between different configuration parameters, whereas Linear Regression treats each parameter independently.

3. **Robustness to outliers**: The gradient boosting approach of LightGBM is less sensitive to outliers than Linear Regression, which can be heavily influenced by extreme values.

4. **Automatic feature selection**: LightGBM implicitly performs feature selection by assigning different importance levels to features, focusing on the most predictive ones.

These results confirm our hypothesis that the complex, non-linear nature of the x264 configuration space requires more sophisticated models than basic linear regression to achieve accurate performance predictions.

6 Limitations and Future Work

Despite LightGBM's superior performance, our approach has limitations:

First, we only tested on x264 encoder configurations. Future work should verify if these findings generalize to other configurable systems. Second, our current approach requires separate models for each video dataset. Transfer learning techniques could enable knowledge transfer between videos. Third, the model interpretability is limited with LightGBM compared to Linear Regression.

The current implementation also has several opportunities for improvement:

- **Hyperparameter Optimization**: Our LightGBM model uses fixed hyperparameters. Implementing a systematic hyperparameter tuning process using techniques like Bayesian optimization could further enhance prediction accuracy.

- **Feature Engineering**: We currently use raw configuration parameters. Applying advanced feature engineering techniques might uncover more predictive patterns and improve model performance.

- **Ensemble Methods**: Combining multiple models through techniques like stacking could potentially yield better results than a single LightGBM model.

- **Online Learning**: Developing incremental learning approaches that could update the model as new configurations are tested would reduce the need for extensive offline training.

Future work should explore feature importance analysis to identify which configuration parameters most significantly impact performance. Additionally, testing more advanced models like neural networks could further improve prediction accuracy.

7 Conclusion

This study evaluates the effectiveness of Linear Regression and LightGBM in predicting x264 encoder configuration performance. LightGBM significantly outperforms Linear Regression across all datasets and evaluation metrics, achieving:

- 59–61% lower MAE,

- 37–39% lower RMSE, and

- Dramatically reduced MAPE (11–16% vs. 51–87%).

These findings demonstrate the necessity of non-linear modeling for complex configuration spaces. Advanced machine learning methods such as LightGBM can greatly enhance prediction accuracy and robustness compared to traditional linear approaches, making them better suited for practical deployment in performance-critical applications.

8 Artifact

The source code and results are available at:
https://github.com/niziyan/config-performance-learning-x264

References

[1] Siegmund, N., Grebhahn, A., Apel, S., & Kästner, C. (2015). Performance-influence models for highly configurable systems. In Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, 284-294.

[2] Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In International Conference on Learning and Intelligent Optimization, 507-523.

[3] Murillo-Morera, J., Quesada-López, C., Jenkins, M., & Calvo-Mena, R. A. (2018). A genetic algorithm based framework for software effort prediction. Journal of Software Engineering Research and Development, 6(1), 1-21.

[4] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 3146-3154.

[5] Chen, J., Nair, V., Krishna, R., & Menzies, T. (2020). "Sampling" as a baseline optimizer for search-based software engineering. IEEE Transactions on Software Engineering, 46(7), 810-827.

[6] Jamshidi, P., Velez, M., Kästner, C., Siegmund, N., & Kawthekar, P. (2017). Transfer learning for improving model predictions in highly configurable software. In IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 31-41.