geosciences

MDPI

Article

LLM-Powered Multi-Agent Framework for Automated PPV Prediction in Tunnel Blasting

Jian Xu [1], Haiping Fan [2] and Danial Jahed Armaghani [3,*]

[1] School of Architectural Equipment, Zhejiang College of Construction, Hangzhou 311231, China; xujian@zjjs.edu.cn
[2] China Railway Engineering Design and Consulting Group Co., Ltd., Beijing 100071, China; fanhaiping@crecg.com
[3] School of Civil and Environmental Engineering, University of Technology Sydney, Ultimo, NSW 2007, Australia
* Correspondence: danial.jahedarmaghani@uts.edu.au

Abstract

Accurate prediction of blasting-induced peak particle velocity (PPV) is critical for assessing structural damage risk and ensuring safe tunnel construction. This study proposes an AI agent-based Evaluator-Optimizer workflow that automates the model-development pipeline from prepared dataset input through model training, performance evaluation, hyperparameter optimization, and ensemble construction, with limited manual intervention after dataset definition. The framework employs a multi-agent architecture comprising three collaborative agents—an Orchestrator, an Evaluator, and an Optimizer—supported by a large language model (LLM) reasoning layer. The Evaluator agent analyzes model performance across multiple metrics and generates diagnostic insights; the Optimizer agent translates these insights into structured optimization plans; and the Orchestrator coordinates the evaluate-optimize loop and stopping logic. The workflow was applied to a dataset of 102 tunnel blasting events. Nine candidate regression models spanning tree-based, kernel-based, neural network, and regularized linear families were trained and evaluated. The results show that the workflow enables three substantive observations: (i) across five tree-based models the powder factor is the dominant predictor (28.7–50.5% relative importance); (ii) under 50 Monte-Carlo repeated 80/20 splits, KNN and the Voting ensemble are statistically indistinguishable and form the most stable performance cluster, while Gradient Boosting lies within the same cluster with larger variance; and (iii) under nested 5×5 cross-validation, the un-leaked R^2 for the top models is about 0.84–0.86, which quantifies the small-sample over-optimism that any future PPV study on single 80/20 splits should expect. The study therefore contributes both a portable agent architecture for tabular geotechnical regression and a concrete cautionary result about single-split benchmarking.

Keywords: peak particle velocity; tunnel blasting; AI agent; large language model; machine learning; gradient boosting

Check for updates

Academic Editors: Hongyuan Liu, Xianhui Feng and Haoyu Han

Received: 23 March 2026
Revised: 19 April 2026
Accepted: 21 April 2026
Published: 28 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Blasting remains an indispensable and economically efficient technique for rock excavation and fragmentation in tunnel engineering. A primary and unavoidable consequence of blasting operations is the generation of blast-induced ground vibrations, which propagate through the rock and can pose a substantial threat to the structural integrity of adjacent structures [1–3]. These vibrations can induce cumulative damage, leading to significant

safety concerns, operational disruptions, and potential economic liabilities [4,5]. Consequently, the accurate prediction and meticulous control of these vibrations are not merely operational objectives but are of paramount importance for ensuring engineering safety, regulatory compliance, and effective risk management [6].

Peak particle velocity (PPV), defined as the maximum instantaneous vibration velocity of a particle during wave propagation, is widely used as a key indicator to characterize the intensity of tunnel blasting vibration [7]. Its prediction is essential in tunnel blasting operations to minimize vibration damage to nearby structures and ensure safe excavation practices. Traditional empirical methods have long been the standard approach, relying on established mathematical relationships between blast parameters and resulting ground vibrations. However, these conventional techniques often struggle with the complex, non-linear relationships inherent in blasting scenarios, where multiple variables interact in unpredictable ways. Conventional empirical formulas typically consider only two factors—charging amount and explosive distance—while completely ignoring other critical parameters that significantly influence ground vibrations [8–10]. This reductionist approach fails to account for essential factors including geological conditions, elevation differences caused by topography, rock properties, rock mass structure, delay timing, and blast geometry [11]. The omission of these significant parameters leads to less accurate results and limits the practical value of empirical predictions.

As a result, researchers have increasingly turned to machine learning approaches, hoping to capture these complex patterns through data-driven models [12–14]. Support vector machines (SVM) have been recognized as versatile tools for PPV prediction, offering high accuracy and fast computation as major advantages [15]. Artificial neural networks (ANN) approach has been successfully implemented using four-layer feed-forward back propagation multi-layer perceptron trained with Levenberg–Marquardt algorithms, achieving coefficient of determination values as high as 0.957 [16]. Assessment studies using root mean squared error and coefficient correlation have consistently shown that artificial neural network-based PPV predictions yield the lowest errors compared to conventional empirical formulas and multivariable regression analyses [17,18].

The evolution from single machine learning models to hybrid systems optimized by metaheuristic algorithms represents a significant advancement in PPV prediction accuracy and reliability [19–22]. Hybrid artificial neural network models incorporating genetic algorithm (GA), particle swarm optimization (PSO), imperialism competitive algorithm (ICA), artificial bee colony (ABC), and firefly algorithm (FA) optimizers have demonstrated progressively improved performance, with the FA-ANN model achieving the highest accuracy levels (R^2 = 0.913 for training and 0.910 for testing) [23]. Research has consistently shown that hybrid models significantly reduce prediction errors compared to single models, with notable examples including ANN combined with FA reducing PPV by 60% and gene expression programming with cuckoo optimization algorithm achieving 55.33% reduction in PPV values [24]. In addition, advanced ensemble methods also have gained prominence through the integration of tree-based algorithms with optimization techniques. Random Forest models enhanced with whale optimization algorithm (WOA) have demonstrated superior performance compared to regular RF models [25]. XGBoost has become particularly popular as a base algorithm for optimization, with studies showing that PSO-optimized XGBoost models achieve higher prediction accuracy than original implementations [26]. Multiple metaheuristic algorithms including WOA, gray wolf optimization (GWO), and Bayesian optimization have been successfully applied to fine-tune XGBoost hyperparameters, with all optimized models showing superior performance to baseline XGBoost [27].

Despite the promise of machine/deep learning methods, most PPV studies still rely on a largely manual workflow: the analyst must pre-select model families, set or search hy-

perparameters, compare metrics, and decide when to stop or try another configuration [28]. Such workflows are time-consuming, difficult to reproduce in full detail, and offer limited structured reasoning about why one model outperforms another or which improvement direction to pursue next.

Representative AutoML platforms such as Auto-Sklearn, TPOT, and H_2O AutoML can also automate model comparison and hyperparameter tuning with limited manual intervention. However, in engineering applications the objective is not only to obtain a competitive score, but also to preserve a traceable workflow that explains why a model was preferred, which variables dominated the response, and why further refinement should or should not continue. The present study is therefore positioned as a domain-oriented agentic workflow, not as a replacement for mature AutoML libraries.

In parallel, the broader artificial intelligence community has advanced "agentic" systems that combine multiple specialized roles—planning, evaluation, and optimization—often orchestrated by large language models (LLMs) to reason over natural-language and numerical evidence. In engineering-oriented AI, such systems are best understood as role-based workflows in which complex tasks are decomposed into inspectable decision stages rather than as unrestricted autonomous agents. Recent LLM applications in geotechnical engineering have mainly focused on retrieval, interpretation support, and text-assisted analysis [29], whereas the present study embeds the LLM within a model-development workflow as a reasoning layer for performance diagnosis and optimization planning. Applications in geoscience and engineering geology are emerging, but the use of an explicit LLM-powered agent framework dedicated to blast–vibration prediction and PPV model development remains, to the best of our knowledge, not yet reported in the peer-reviewed literature.

This paper addresses the above gap by proposing and implementing an AI agent-based Evaluator–Optimizer workflow for tunnel blasting-induced PPV prediction, and by evaluating what the agent layer actually contributes beyond automation. The framework is designed as follows: (i) an Orchestrator coordinates training, evaluation, and optimization rounds; (ii) an Evaluator agent synthesizes multi-metric performance, cross-model diagnostics, and feature-importance summaries; and (iii) an Optimizer agent translates evaluation insights into executable steps—hyperparameter search and ensemble construction—rather than a single fixed optimizer run in isolation. The present study should be viewed as a proof-of-concept application of an LLM-assisted agentic workflow to PPV prediction, with the goal of testing whether such an architecture can provide a transparent model-development pipeline while remaining competitive with established nonlinear regressors on the available dataset. The objectives of this study are therefore:

(1) A portable, role-based LLM-Evaluator–Optimizer–Orchestrator architecture for tabular geotechnical regression, with LLM restricted to a diagnostic role and all scoring, fitting, and searching kept deterministic;
(2) To benchmark the pipeline against a diverse set of regressors (including tree-based, kernel-based, neural, and linear models) on a field tunnel blasting dataset, and to report prediction accuracy, residual diagnostics, and feature importance in a form suitable for engineering interpretation;
(3) To discuss practical limitations—such as small sample size and the trade-off between hyperparameter search and generalization—so that future work can build on the same agent-based design for other blast-related or geotechnical prediction tasks.

2. Materials

2.1. Dataset Description

This study collected data from the construction of a tunnel in Xiushan County, Chongqing, China. The tunnel, built in 2024 as part of the railway system, lies in a region characterized by hilly denudation and karst erosion landforms, with relatively flat terrain. The lithology at the tunnel site primarily consists of strongly to weakly weathered tuff and hard plastic silty clay. These formations typically exhibit good self-stabilization properties, facilitating tunnel construction.

The dataset employed in this study was collected from a tunnel blasting project and comprises 102 blasting events. For each blasting event, six input parameters that are widely recognized as key influencing factors of blast-induced ground vibration were measured, along with the corresponding PPV as the target variable. The input parameters include the maximum charge per delay (Q_{max}), the distance from the blast to the monitoring point (DI), the burden-to-spacing ratio (B/S), the stemming length (St), the powder factor (PF), and the hole depth (HD). A description of each variable is provided below:

Q_{max}: The maximum weight of explosive detonated within a single delay interval. This parameter directly governs the energy released per detonation and is one of the most dominant factors influencing blast-induced vibration intensity.

DI: The distance from the blast source to the vibration monitoring point. According to the wave attenuation theory, ground vibration attenuates with increasing distance from the blast.

B/S: The ratio of the burden to the hole spacing in the blast pattern design. This geometric parameter controls the confinement conditions and the fragmentation efficiency of the blast.

St: The length of inert material (typically drill cuttings) placed above the explosive charge in the blasthole to confine the explosive gases. Adequate stemming improves explosive energy utilization and influences vibration propagation.

PF: The quantity of explosive consumed per unit volume of rock to be broken. A higher powder factor indicates more intensive blasting and generally results in stronger ground vibrations.

HD: The total depth of the blasthole. Longer holes result in larger explosive column lengths and can influence both the vibration magnitude and the spatial distribution of the blast energy.

2.2. Statistical Analysis

Table 1 summarizes the descriptive statistics of the dataset. The 102 samples are complete with no missing values across any of the seven variables. The Q_{max} ranges from 45.804 kg to 305.60 kg with a mean of 179.623 kg and a coefficient of variation (CV) of 36.46%, reflecting the substantial variation in charge weight across different blasting rounds. The DI varies between 285 m and 531 m, with a relatively moderate CV of 18.91%, indicating that the monitoring stations were positioned within a consistent range. The B/S exhibits the lowest variability among all parameters (CV = 7.74%), spanning a narrow range from 0.70 to 0.92, which suggests that the blast pattern geometry was maintained in a relatively controlled manner throughout the project. The St ranges from 1.90 m to 3.60 m with a CV of 15.13%. The PF, ranging from 0.232 to 0.94 kg/m^3 with a CV of 28.34%, shows considerable spread and reflects the diversity of blasting conditions encountered at the site. The HD ranges from 5.23 m to 23.17 m (CV = 28.45%), capturing a wide spectrum of drilling configurations. The target variable, PPV, exhibits the highest variability among all parameters, with a CV of 57.32%, ranging from 0.13 mm/s to 11.05 mm/s. This

wide spread underscores the complexity of blast-induced vibration and the challenge of accurate prediction.

Table 1. Descriptive statistics of the blasting dataset.

Variable	Count	Mean	Std	Min	Median	Max	CV (%)
Q_{max} (kg)	102	179.623	65.487	45.804	188.056	305.6	36.46
DI (m)		379.52	71.769	285	376.5	531	18.91
B/S		0.819	0.063	0.7	0.82	0.92	7.74
St (m)		2.63	0.398	1.9	2.6	3.6	15.13
PF (kg/m^3)		0.655	0.186	0.232	0.680	0.94	28.34
HD (m)		14.115	4.016	5.23	15	23.17	28.45
PPV (mm/s)		5.337	3.059	0.13	5.165	11.05	57.32

The frequency distributions and kernel density estimates (KDE) of all seven variables are presented in Figure 1. The histograms reveal that Q_{max} follows an approximately unimodal distribution centered around 180 kg, while DI displays a bimodal pattern with concentrations near 310 m and 490 m, reflecting two distinct monitoring distances or tunnel sections. The B/S ratio shows a near-uniform distribution across its narrow range, and St exhibits a right-skewed pattern with the majority of values clustered between 2.2 m and 2.8 m. The PF distribution is slightly left-skewed, with higher frequencies at larger powder factors. HD displays a multimodal distribution with a notable concentration around 15 m. The PPV distribution is relatively flat, consistent with its high CV and platykurtic shape.

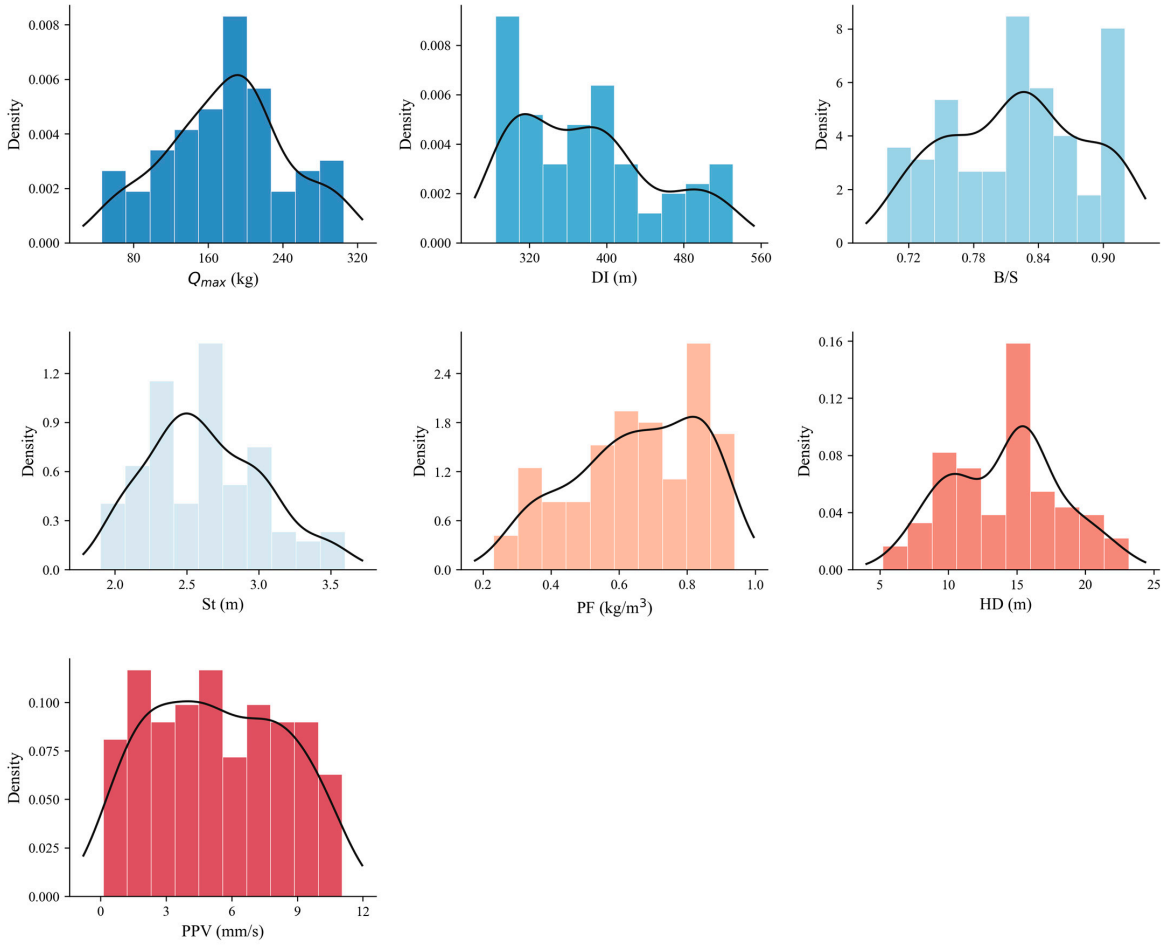

Figure 1. Frequency distributions and kernel density estimates (KDE) of all variables.

Figure 2 shows the box plots of all variables. The interquartile ranges (IQR) are well-defined, and no observations fall beyond 1.5 × IQR from the box boundaries, indicating that the dataset is clean and free from anomalous measurements.

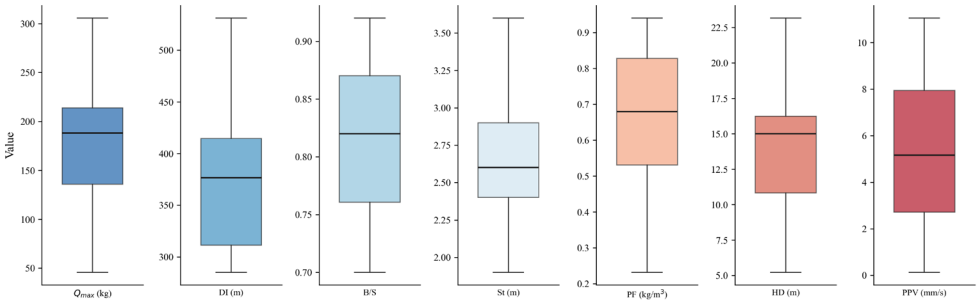

Figure 2. Box plots of all variables.

2.3. Correlation Analysis

A Pearson correlation analysis (r) was conducted to examine the linear relationships between the input parameters and PPV, as well as the inter-parameter dependencies [30]. Figure 3 displays the correlation matrix.

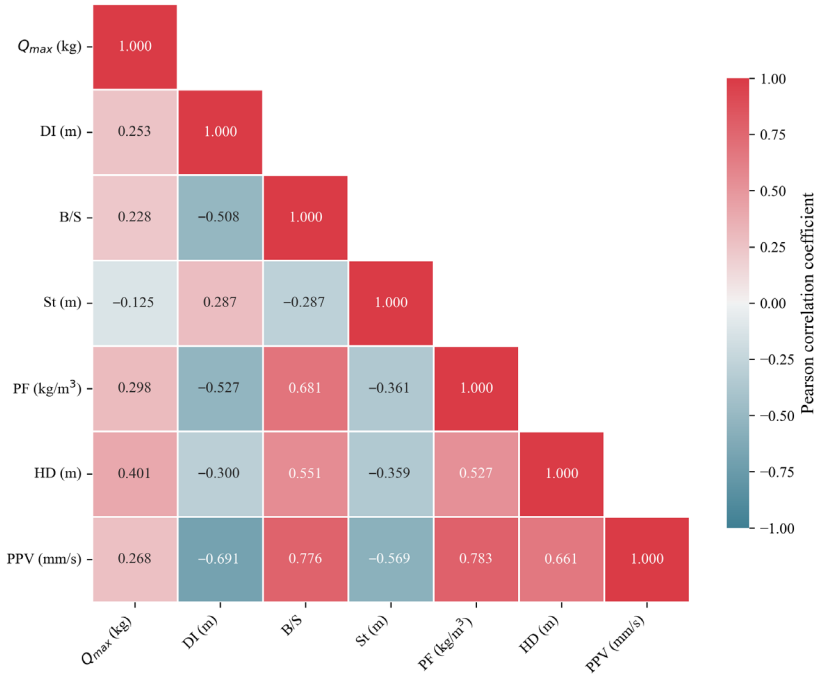

Figure 3. Pearson correlation coefficient matrix of the input parameters and PPV.

Among the six input parameters, PF demonstrates the strongest positive correlation with PPV (r = 0.783), followed by B/S (r = 0.776) and HD (r = 0.661). These results are physically consistent: a higher powder factor indicates a more intensive use of explosives per unit volume, which naturally generates stronger ground vibrations. A larger burden-to-spacing ratio increases the confinement of the explosive energy, potentially leading to higher vibration transmission efficiency. Greater hole depths correspond to longer explosive columns, which release more energy and contribute to higher PPV values [31].

In contrast, DI shows a strong negative correlation with PPV (r = −0.691), which is consistent with the well-established principle of vibration attenuation with distance. The St also exhibits a moderate negative correlation (r = −0.569), as longer stemming improves the confinement of explosive gases and directs the blast energy more efficiently into rock breakage rather than vibration. The Q_{max} shows only a weak positive correlation with

PPV (r = 0.268). Although Q_{max} is traditionally considered a primary controlling factor in empirical PPV prediction equations (e.g., the USBM equation [32]), the relatively low correlation in the present dataset may be attributed to the confounding effects of the other parameters that vary simultaneously.

Regarding inter-parameter correlations, a moderate-to-strong positive correlation is observed between B/S and PF (r = 0.681) and between B/S and HD (r = 0.551), while a moderate negative correlation exists between DI and PF (r = −0.527) and between DI and B/S (r = −0.508). These inter-correlations indicate the presence of multicollinearity among the input parameters, which further supports the adoption of machine learning models that are robust to correlated features, as opposed to traditional multivariate linear regression.

The pairwise scatter plot matrix of the key variables is presented in Figure 4, which provides a comprehensive visual overview of the bivariate relationships and marginal distributions across the dataset. This is a common figure and analysis in many machine learning studies [32–34]. Overall, the scatter patterns confirm that the relationships between the input parameters and PPV are nonlinear in nature, reinforcing the rationale for employing advanced machine learning techniques in the proposed AI agent-based workflow.

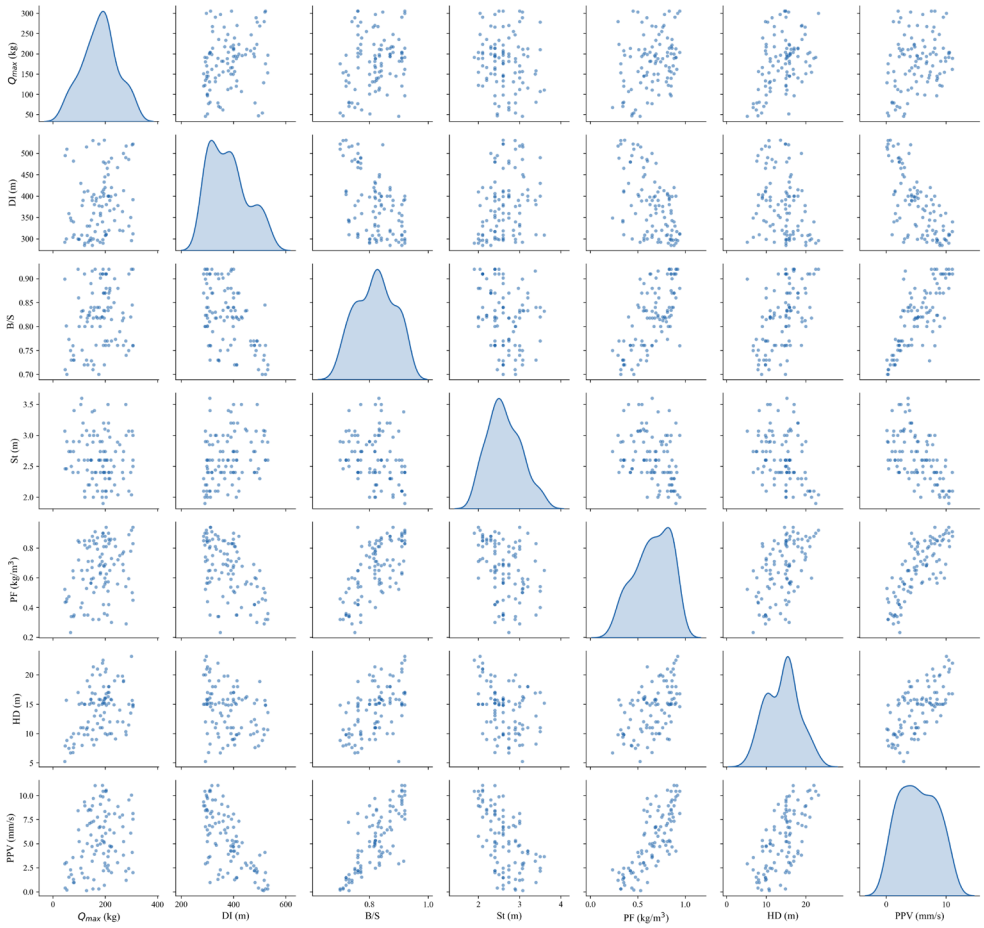

Figure 4. Pairwise scatter plot matrix across all variables.

3. Methods

This study proposes an AI agent-based Evaluator-Optimizer workflow for the automated prediction and optimization of blasting-induced PPV. The overall workflow follows the Evaluator-Optimizer pattern, a well-established design paradigm in agentic AI systems. This pattern decomposes the machine learning optimization problem into two complementary roles: an Evaluator agent that assesses model quality and generates diagnostic insights,

and an Optimizer agent that translates those insights into executable improvement actions. A higher-level Orchestrator agent coordinates the iterative loop between evaluation and optimization, making autonomous decisions about when to continue refining models and when to terminate the process.

The complete workflow, illustrated schematically in Figure 5, proceeds through six phases: (1) Input and preprocessing: the blasting dataset is loaded, split once into training and test subsets (80/20), and standardized; the same partition is reused for all baseline, optimized, and ensemble models. (2) Initial model training: a diverse pool of nine candidate machine learning models is trained on the dataset, spanning tree-based, kernel-based, regularized linear, neural network, and boosting families. (3) LLM-powered evaluation: the Evaluator agent analyzes all model results across multiple performance metrics, ranks models, identifies weaknesses, and generates actionable recommendations via the LLM. (4) LLM-guided optimization: the Optimizer agent interprets the evaluation report and formulates an optimization plan, which is then executed through hyperparameter tuning and ensemble construction. (5) Decision: the Orchestrator assesses whether the optimization target has been achieved (i.e., the target R^2 score is reached, the improvement falls below a predefined threshold, or the maximum number of optimization rounds is exhausted). (6) Output: upon convergence, the framework produces a final report containing the best model, its performance metrics, the R^2 progression across iterations, and feature-importance rankings.

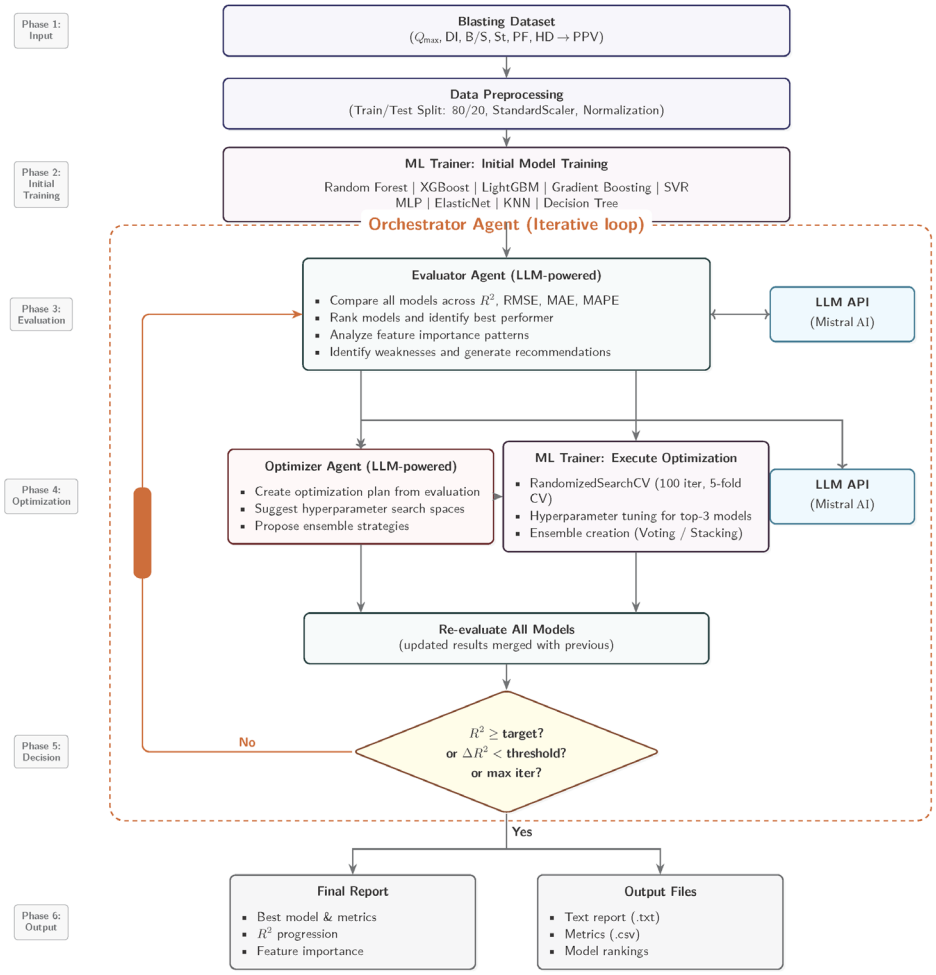

Figure 5. Workflow of the AI Agent-based Evaluator-Optimizer for PPV prediction.

This iterative, closed-loop design ensures that the model selection and optimization process is not a one-shot procedure but a progressive refinement guided by LLM-based reasoning [33]. Each iteration builds upon the accumulated knowledge from all prior iterations, allowing the agents to adaptively focus on the most promising improvement directions [34].

Unlike conventional machine learning pipelines and off-the-shelf AutoML tools (Auto-Sklearn, TPOT, H2O AutoML, FLAML), the intended contribution here is not unrestricted search over all possible model pipelines. Rather, the framework uses a role-based evaluator-optimizer architecture in which the LLM provides traceable diagnostic and planning support, while the underlying model fitting, scoring, and search execution remain deterministic.

AutoML platforms optimize over a broad, framework-defined space of preprocessing and learning algorithms under a single objective of expected-score maximization. The workflow proposed here is deliberately narrower: the model family and its hyperparameter bounds are fixed a priori from model-class knowledge and small-sample considerations, the ML Trainer's model fitting, scoring, and search execution remain deterministic scikit-learn code, and the LLM layer acts only as a planner that (i) prioritizes which models to optimize, (ii) summarizes cross-model diagnostics in natural language, and (iii) rationalizes the Orchestrator's stopping decision. The resulting artifact is closer to an engineering design-review log than to an open hyperparameter search. The present paper therefore does not claim score superiority over AutoML with logging; it claims traceability and inspectability of the decision chain on a dataset scale typical of single-site PPV studies.

3.1. Orchestrator Agent

The Orchestrator agent serves as the top-level controller of the entire workflow. It initializes the ML Trainer, Evaluator, and Optimizer components, and manages the sequential flow of data and control signals between them. After each optimization round, the Orchestrator evaluates whether to proceed with another iteration or to terminate. Three stopping criteria are implemented: (i) the best R^2 score across all models meets or exceeds a user-defined target ($R^2 \geq 0.95$); (ii) the R^2 improvement from the latest iteration falls below a minimum threshold ($\Delta R^2 < 0.01$), indicating convergence; or (iii) the maximum number of optimization rounds is reached (five rounds in the reported experiment).

The Orchestrator maintains a cumulative pool of all model results across iterations. When optimized models or ensembles are produced in a given round, they are merged into the existing result pool, so the Evaluator always has access to the complete history of models. Upon termination, the Orchestrator compiles a comprehensive summary report, including the R^2 progression history, the top-ranked models, and the final set of recommendations.

In the current implementation, the Orchestrator functions primarily as a workflow controller rather than a fully autonomous strategist. Its main responsibilities are sequencing, result aggregation, and application of explicit stopping criteria; accordingly, the present manuscript should not be interpreted as claiming unrestricted high-level LLM decision making at this layer.

3.2. Evaluator Agent

The Evaluator agent is responsible for comprehensive model performance assessment. At each iteration, the Evaluator receives the full set of trained model results and performs multi-metric comparison, feature importance aggregation, LLM-powered analysis, and human-in-the-loop support. All models are evaluated using four regression metrics: root mean square error (RMSE), mean absolute error (MAE), coefficient of determination (R^2), and mean absolute percentage error (MAPE). The models are ranked primarily by R^2, with

the other metrics providing supplementary diagnostic information. Formally, for a set of n test samples with true values y_i and predicted values $\hat{y}_i$:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_i)^2} \quad (1)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (2)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{\eta}|y_i - \hat{y}_i| \quad (3)$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (4)$$

where n is the number of data points, y_i is the actual value of PPV, $\hat{y}_i$ is the predicted value of PPV, and $\overline{y}_i$ is the average of y_i.

For models that expose feature importance attributes (e.g., tree-based models via the Gini impurity or gain-based importance), the Evaluator aggregates the feature importance vectors across all applicable models by averaging. This cross-model aggregation provides a more robust and model-agnostic view of which input parameters are most influential for PPV prediction.

The Evaluator formats all model results—including metrics, cross-validation scores, and feature importance values—into a structured textual summary and submits it to the LLM along with a domain-specific system prompt. The system prompt instructs the LLM to act as an expert ML evaluator specializing in regression tasks and PPV prediction. The LLM then generates a natural-language analysis covering: (i) an overall assessment of model performance, (ii) identification of the best-performing models and the reasons for their success, (iii) a list of weaknesses and failure modes, and (iv) feature-importance insights. The resulting text is then parsed into structured weaknesses and recommendations that are passed to the Optimizer as planning signals. Thus, the LLM acts as a reasoning layer between evaluation and optimization, not as a replacement for deterministic model training.

3.3. Optimizer Agent

The Optimizer agent translates the diagnostic output of the Evaluator into a concrete, executable optimization plan and carries it out. Its operation consists of two stages: planning and execution.

The Optimizer receives the evaluation report (including model rankings, identified weaknesses, and recommendations) and submits it to the LLM with a system prompt instructing the LLM to act as an expert model optimizer. The LLM is asked to return a structured JSON response specifying: (i) which models to target for optimization, (ii) suggested search directions within predefined hyperparameter domains, (iii) feature engineering suggestions, (iv) an ensemble strategy (type, member models, and rationale), and (v) a prioritized list of optimization actions. The search domains are defined a priori from model-family knowledge and small-sample considerations; the LLM prioritizes models and may emphasize portions of those predefined ranges rather than generating unrestricted search spaces from scratch. If the LLM response fails to parse correctly, the Optimizer falls back to a rule-based default plan that targets the top-three models and uses the predefined hyperparameter search spaces.

The Optimizer executes the plan through two primary mechanisms:

- For each target model, the Optimizer invokes the ML Trainer's hyperparameter optimization function, which performs a randomized search over the specified parameter space with 200 sampled configurations and 5-fold cross-validation, scored by R^2. The

"200 iterations" therefore refers to each selected model within a given optimization round, not to the entire workflow. In the reported experiment, only the top three models were tuned in the single optimization round. The best hyperparameter configuration identified by the search is then used to retrain the model, and the resulting performance on the held-out test set is recorded.

- Following the LLM's ensemble strategy recommendation, the Optimizer constructs an ensemble model from the top-performing individual models. Two ensemble paradigms are supported: (i) Voting ensemble, which averages the predictions of the member models, and (ii) Stacking ensemble, which uses a meta-learner to combine the base model predictions through 5-fold cross-validation.

After execution, the Optimizer computes an improvement summary comparing the original and optimized model performances, including both absolute and percentage improvements in R^2.

3.4. ML Trainer

The ML Trainer is the computational engine underlying both the initial training phase and the optimization execution. It is responsible for model instantiation, training, evaluation, and hyperparameter search. The ML Trainer maintains a registry of nine candidate regression models spanning five algorithmic families, as summarized in Table 2.

Table 2. Machine learning models employed in the proposed workflow.

Model	Algorithm Family	Key Characteristics
Random Forest	Bagging ensemble	Parallel decision trees with bootstrap aggregation; robust to overfitting
XGBoost	Gradient boosting	Sequential boosting with regularization; handles sparse data
LightGBM	Gradient boosting	Leaf-wise tree growth; efficient on large datasets
Gradient Boosting	Gradient boosting	Classical stage-wise additive boosting
SVR	Kernel-based	Maps inputs to high-dimensional space via kernel function
MLP	Neural network	Multi-layer perceptron with backpropagation
ElasticNet	Regularized linear	Combines L1 (Lasso) and L2 (Ridge) penalties
KNN	Instance-based	Prediction via weighted average of *k*-nearest training samples
Decision Tree	Tree-based	Single recursive-partitioning tree; high interpretability

The dataset is split into training (80%) and test (20%) sets using a random split with a fixed seed ('andom_state = 42'). The same partition is reused for all baseline, optimized, and ensemble models so that all reported variants are compared on an identical held-out subset. Input features are standardized, which centers each feature to zero mean and unit variance. Each model is trained on the training set and evaluated on the held-out test set using the four metrics defined in Section 3.2. In addition, 5-fold cross-validation is performed on the training set to assess generalization stability. For models that provide feature importance attributes (tree-based and linear models), the importance values are extracted and stored alongside the performance metrics.

Table 3 summarizes the hyperparameter search spaces defined for each model. These spaces were designed to cover a broad range of configurations while remaining computationally tractable for the randomized search strategy with 200 iterations per selected model. Because the dataset is small (n = 102), the bounds were intentionally kept moderate; for example, tree depth and ensemble size were capped within practical ranges to balance model flexibility against overfitting risk.

Table 3. Hyperparameter search spaces for the candidate models.

Model	Hyperparameter	Search Values
Random Forest	'n_estimators'	{50, 100, 200, 300}
	'max_depth'	{3, 5, 7, 10}
	'min_samples_split'	{2, 5, 10}
	'min_samples_leaf'	{1, 2, 4}
	'max_features'	{sqrt, log2, None}
XGBoost	'n_estimators'	{50, 100, 200, 300}
	'learning_rate'	{0.01, 0.05, 0.1, 0.2}
	'max_depth'	{3, 5, 7, 10}
	'subsample'	{0.7, 0.8, 0.9, 1.0}
	'colsample_bytree'	{0.7, 0.8, 0.9, 1.0}
	'reg_alpha'	{0, 0.1, 0.5, 1.0}
	'reg_lambda'	{0, 0.1, 0.5, 1.0}
LightGBM	'n_estimators'	{50, 100, 200, 300}
	'learning_rate'	{0.01, 0.05, 0.1, 0.2}
	'max_depth'	{3, 5, 7, 10}
	'num_leaves'	{3, 6, 9}
	'subsample'	{0.7, 0.8, 0.9, 1.0}
	'colsample_bytree'	{0.7, 0.8, 0.9, 1.0}
Gradient Boosting	'n_estimators'	{50, 100, 200}
	'learning_rate'	{0.01, 0.05, 0.1, 0.2}
	'max_depth'	{3, 5, 7, 10}
	'min_samples_split'	{2, 5, 10}
	'subsample'	{0.8, 0.9, 1.0}
SVR	'C'	{0.1, 1, 10, 100}
	'epsilon'	{0.01, 0.1, 0.2}
	'kernel'	{rbf, linear, poly}
	'gamma'	{scale, auto, 0.1, 0.01}
MLP	'hidden_layer_sizes'	{(8,), (16,), (8, 8), (16, 16)}
	'activation'	{relu, tanh}
	'alpha'	{0.0001, 0.001, 0.01}
	'learning_rate'	{constant, adaptive}
	'learning_rate_init'	{0.001, 0.01, 0.1}
ElasticNet	'alpha'	{0.001, 0.01, 0.1, 1.0, 10.0}
	'l1_ratio'	{0.1, 0.3, 0.5, 0.7, 0.9}
KNN	'n_neighbors'	{3, 5, 7, 9, 11, 15}
	'weights'	{uniform, distance}
	'metric'	{euclidean, manhattan, minkowski}
Decision Tree	'max_depth'	{3, 5, 10, 15}
	'min_samples_split'	{2, 5, 10, 15}
	'min_samples_leaf'	{1, 2, 4, 8}

3.5. Workflow Configuration

The entire workflow is governed by a centralized configuration system organized into three tiers:

- LLM configuration: the provider, model identifier, system prompts, and inference settings used for the reasoning layer. In the implementation, the workflow is configured for Mistral AI ('mistral-large-latest') through a chat-completions interface, with task-specific system prompts for the Evaluator and Optimizer, temperature = 0.7, max_tokens = 4096, no few-shot examples, and no domain fine-tuning.
- ML configuration: The list of models to train, train/test split ratio, number of cross-validation folds, hyperparameter search type, and number of search iterations.

- Agent configuration: The evaluation metrics to compute, the available optimization strategies (hyperparameter tuning, feature engineering, model selection, ensemble creation), the maximum number of optimization iterations, the target R^2 score, and the early stopping patience.

This modular configuration design allows us to easily adapt the workflow to different datasets, performance targets, and computational budgets without modifying the core algorithmic components.

For reproducibility, the workflow can also be executed with a rule-based client when external API access is unavailable; however, such a client should be understood as a fallback implementation for deterministic dry-run testing rather than as a substitute for a production LLM backend.

4. Results

4.1. Workflow Execution Summary

The proposed AI agent-based Evaluator-Optimizer workflow was executed on the blasting dataset comprising 102 samples with six input parameters and PPV as the target variable. The dataset was split into 81 training samples (80%) and 21 test samples (20%) with standardized features, and the same held-out subset was reused for all baseline, optimized, and ensemble variants. The workflow configuration employed the following settings: a target R^2 of 0.95, a maximum of 5 optimization rounds, an improvement threshold of 0.01, and a randomized hyperparameter search with 200 sampled configurations for each selected model.

The Orchestrator agent completed the workflow in one optimization iteration (plus the initial training round). Only the three top-ranked models were tuned, so the randomized-search stage involved $3 \times 200 \times 5 = 3000$ cross-validation fits rather than tuning all nine baseline models. The reported end-to-end runtime was approximately 40 s in an environment with Intel(R) Core(TM) i7-14700 and 32 GB RAM. The workflow was terminated after the first optimization round because the R^2 improvement fell below the threshold ($\Delta R^2 < 0.01$), indicating that the marginal gains from additional search did not justify further iterations on this dataset.

4.2. Initial Model Training Results

In the initial training phase, all nine candidate machine learning models were trained on the dataset. The performance metrics for all models are summarized in Table 4 and visualized in Figure 6.

Among the initial nine models, Gradient Boosting (GB) emerged as the best-performing model with an R^2 of 0.9356, an RMSE of 0.7608 mm/s, and an MAE of 0.5981 mm/s. This was followed by SVR ($R^2 = 0.9219$) and Random Forest ($R^2 = 0.9155$). The three top-performing initial models all achieved R^2 values above 0.91, indicating strong predictive capability for PPV.

The ensemble methods and nonlinear models consistently outperformed the linear baseline. ElasticNet, the only regularized linear model, ranked last among the initial models with an R^2 of 0.8564, confirming the nonlinear nature of the PPV prediction problem identified in the correlation analysis. The MLP neural network achieved a moderate R^2 of 0.8635, suggesting that the relatively small sample size (81 training samples) may limit the effectiveness of neural network approaches compared to tree-based ensemble methods.

The cross-validation results provide insight into model generalization stability. SVR exhibited the most stable cross-validation performance (CV $R^2 = 0.823 \pm 0.053$), while KNN was similarly consistent (0.831 ± 0.070). In contrast, Decision Tree showed the highest variability (0.623 ± 0.219), indicating a propensity for overfitting. Gradient Boosting, despite

achieving the best test-set performance, exhibited moderate cross-validation variability (0.810 ± 0.153), which is typical for boosting methods on small datasets.

Table 4. Performance comparison of all models after the initial training and optimization phases.

Model	R^2	RMSE	MAE	MAPE (%)	CV R^2 (Mean $\pm$ Std)
GB	0.9356	0.7608	0.5981	18.04	0.810 ± 0.153
SVR	0.9219	0.8374	0.6325	37.35	0.823 ± 0.053
Random Forest (Opt.)	0.9208	0.8436	0.7440	39.45	—
Voting Ensemble	0.9178	0.8596	0.6863	16.80	—
Random Forest	0.9155	0.8712	0.7199	31.01	0.804 ± 0.122
GB (Opt.)	0.9045	0.9265	0.7025	19.60	—
XGBoost	0.9007	0.9447	0.7320	18.78	0.764 ± 0.182
KNN	0.8868	1.0087	0.8077	41.76	0.831 ± 0.070
Decision Tree	0.8793	1.0412	0.8148	22.39	0.623 ± 0.219
SVR (Opt.)	0.8740	1.0639	0.8519	49.60	—
MLP	0.8635	1.1075	0.8765	70.02	0.814 ± 0.102
LightGBM	0.8593	1.1243	0.9564	42.54	0.760 ± 0.105
ElasticNet	0.8564	1.1360	0.9269	80.05	0.706 ± 0.073

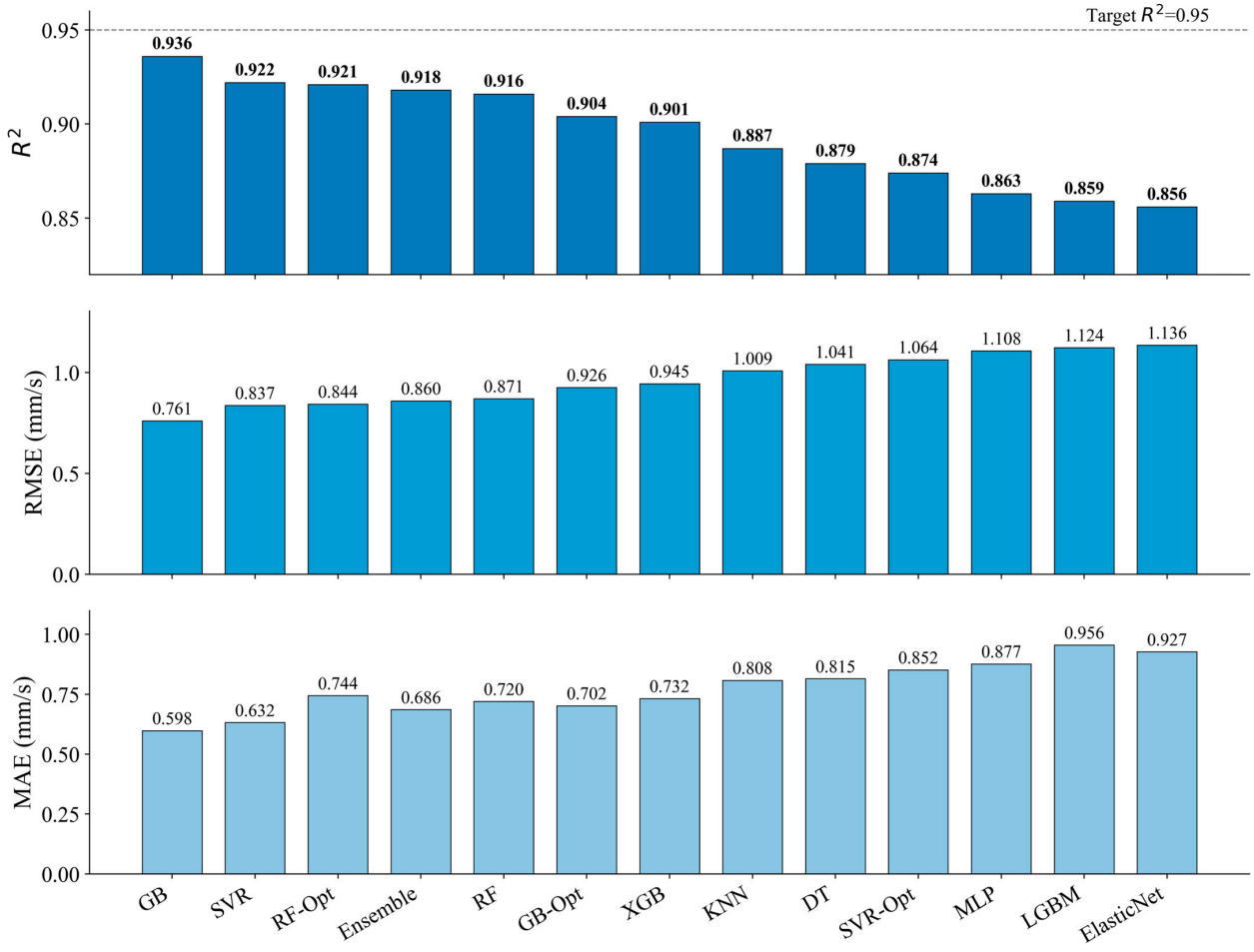

Figure 6. Performance comparison between all trained/optimized models.

4.2.1. Uncertainty Quantification

The single-split ranking in Table 4 was computed with 'random_state = 42'. Because the test set contains only 21 samples, the single-split ranking is sensitive to the specific partition. To quantify this, we repeated the entire 80/20 split, StandardScaler fit, baseline-model fit, and Voting-ensemble fit for 50 independent random seeds. For each seed, the

test-set mean R^2, Std, and 95% CI were stored. Summary statistics are reported in Table 5; full distributions are shown in Figure 7.

Table 5. Monte-Carlo 80/20 test-set R^2 over 50 repetitions (mean ± std; 95% CI).

Model	Mean	Std	95% CI
KNN	0.876	0.047	[0.747, 0.937]
Voting Ensemble	0.867	0.057	[0.743, 0.937]
SVR	0.858	0.044	[0.753, 0.923]
MLP	0.858	0.059	[0.720, 0.934]
Gradient Boosting	0.845	0.078	[0.676, 0.943]
LightGBM	0.827	0.070	[0.649, 0.931]
Random Forest	0.822	0.083	[0.628, 0.925]
XGBoost	0.817	0.093	[0.603, 0.927]
ElasticNet	0.738	0.050	[0.662, 0.843]
Decision Tree	0.734	0.170	[0.408, 0.915]

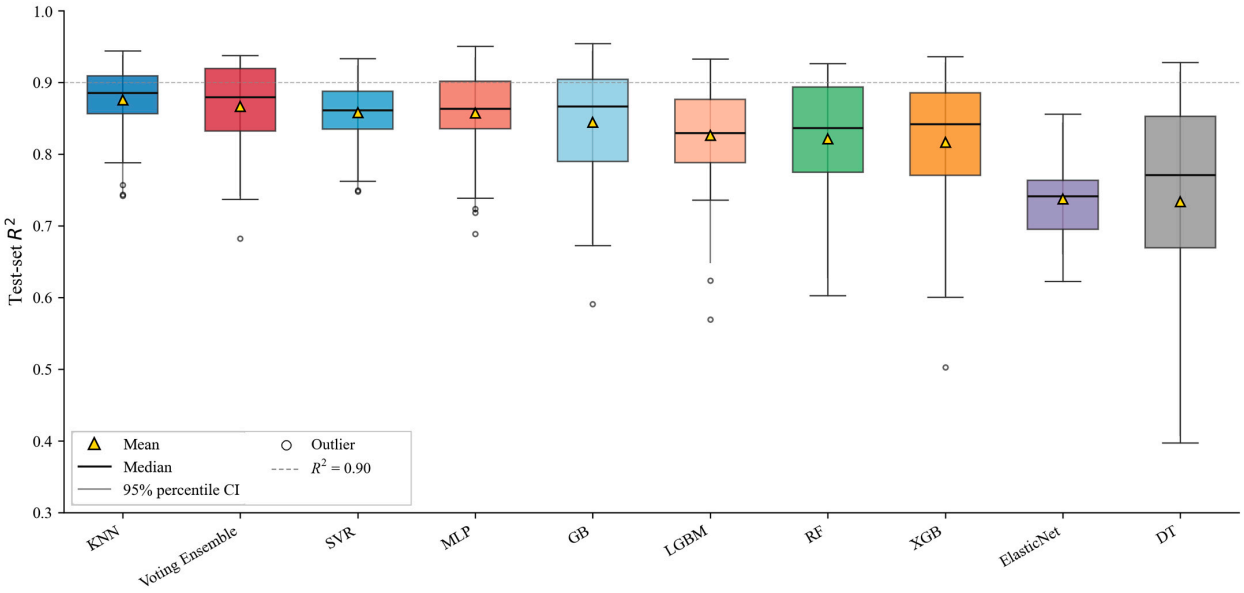

Figure 7. Monte-Carlo test-set R^2 distributions across 50 independent 80/20 splits.

Under repeated splits, KNN (mean R^2 = 0.876, 95% CI [0.747, 0.937]) and the Voting ensemble (0.867, [0.743, 0.937]) are the two most stable top models; SVR (0.858, [0.753, 0.923]) and MLP (0.858, [0.720, 0.934]) are the next cluster; Gradient Boosting (0.845, [0.676, 0.943]) remains in the leading cluster but with distinctly wider variance. The single-split R^2 = 0.9356 originally reported for GB lies at the top of its own distribution and is within the 95% CIs of KNN, SVR, MLP, and the Voting ensemble. This shows that the original ranking is not robust to random-split choice, although the identification of a leading cluster of nonlinear models is robust. ElasticNet and Decision Tree remain clearly separated below this cluster and are therefore unsuitable as stand-alone PPV predictors on this dataset.

4.2.2. Statistical Significance of the Ranking

To test whether the ranking among the top-five Monte-Carlo models is statistically meaningful, paired Wilcoxon signed-rank tests were computed on the 50 paired R^2 values. The two-sided p-values and the rank-biserial correlation (rbc) effect sizes are reported in Table 6. At α = 0.05, KNN and the Voting ensemble are statistically indistinguishable (p = 0.37). Gradient Boosting is significantly below the Voting ensemble ($p < 0.001$, rbc = +0.56, a medium effect) and KNN (p = 0.015), consistent with its larger variance. Differences among SVR, MLP, and GB are not significant. Taken together, this analysis

confirms that no single model can be singled out as the best; instead, a leading cluster of nonlinear models—KNN, Voting ensemble, SVR, MLP, and GB—performs similarly within their confidence intervals, while ElasticNet and Decision Tree are clearly separated below.

Table 6. Paired Wilcoxon signed-rank tests on the 50 Monte-Carlo R^2 values.

Pair	Δ Mean	*p*	rbc
KNN vs. Voting Ensemble	+0.009	0.37	−0.04
KNN vs. SVR	+0.018	0.015	+0.20
KNN vs. MLP	+0.018	0.006	+0.32
KNN vs. GB	+0.031	0.015	+0.16
Voting Ensemble vs. SVR	+0.009	0.07	+0.24
Voting Ensemble vs. MLP	+0.009	0.24	+0.20
Voting Ensemble vs. GB	+0.022	<0.001	+0.56
SVR vs. MLP	+0.000	0.99	−0.08
SVR vs. GB	+0.013	0.32	+0.08
MLP vs. GB	+0.013	0.28	+0.12

4.3. Optimization and Ensemble Results

Following the Evaluator agent's analysis, the Optimizer agent selected the top three models—Gradient Boosting, SVR, and Random Forest—for hyperparameter optimization via the 200 iterations and 5-fold CV. Additionally, a Voting ensemble combining the three optimized models was constructed.

The optimization results, compared to the initial models, are presented in Figure 8. An important finding is that the hyperparameter-optimized versions of Gradient Boosting and SVR did not surpass their initial (default-hyperparameter) counterparts on the test set: GB-Opt achieved R^2 = 0.9045 versus 0.9356 for the initial GB, and SVR-Opt achieved R^2 = 0.8740 versus 0.9219 for the initial SVR. The optimized Random Forest (R^2 = 0.9208) showed a marginal improvement over the initial RF (R^2 = 0.9155), with a gain of ΔR^2 = +0.0053.

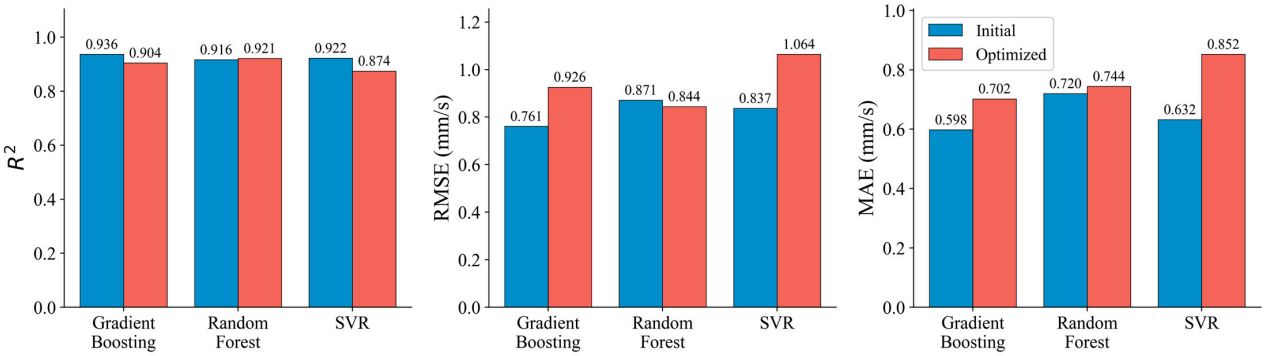

Figure 8. Optimized performance comparison between GB, RF, and SVR.

This counter-intuitive result—where optimization reduced rather than improved performance for two of three models—can be attributed to the small dataset size (n = 102). With only 81 training samples and 5-fold cross-validation, each fold contains approximately 16 samples, which may be insufficient for the randomized search to reliably identify hyperparameter configurations that generalize well to the held-out test set. Supporting evidence for this interpretation is obtained by comparing CV and held-out test performance on the same training/test partition. GB improved from CV R^2 = 0.810 ± 0.153 to 0.848 ± 0.110 after tuning, while its test R^2 decreased from 0.9356 to 0.9045; SVR showed a similar pattern, increasing from 0.823 ± 0.053 to 0.883 ± 0.058 in CV while decreasing from 0.9219 to 0.8740

on the test set. Accordingly, the default configurations are better described as competitive on this dataset rather than universally near-optimal.

The Voting ensemble, which averages the predictions of the three top models, achieved R^2 = 0.9178, RMSE = 0.8596 mm/s, and MAE = 0.6863 mm/s. Notably, the ensemble achieved the lowest MAPE (16.80%) among all 13 models, indicating superior relative error performance. However, the ensemble did not surpass the best individual model (GB, R^2 = 0.9356). This difference points to a metric-dependent trade-off: GB preserves more overall response variance, whereas the ensemble yields better relative-error stability, especially in lower-PPV cases where percentage-based errors are more sensitive.

The R^2 convergence history of the GB model, shown in Figure 9, reveals that the best R^2 score remained at 0.9356 across iterations. The Orchestrator agent correctly identified this plateau and terminated the workflow after one optimization round, demonstrating the effectiveness of the early-stopping mechanism.

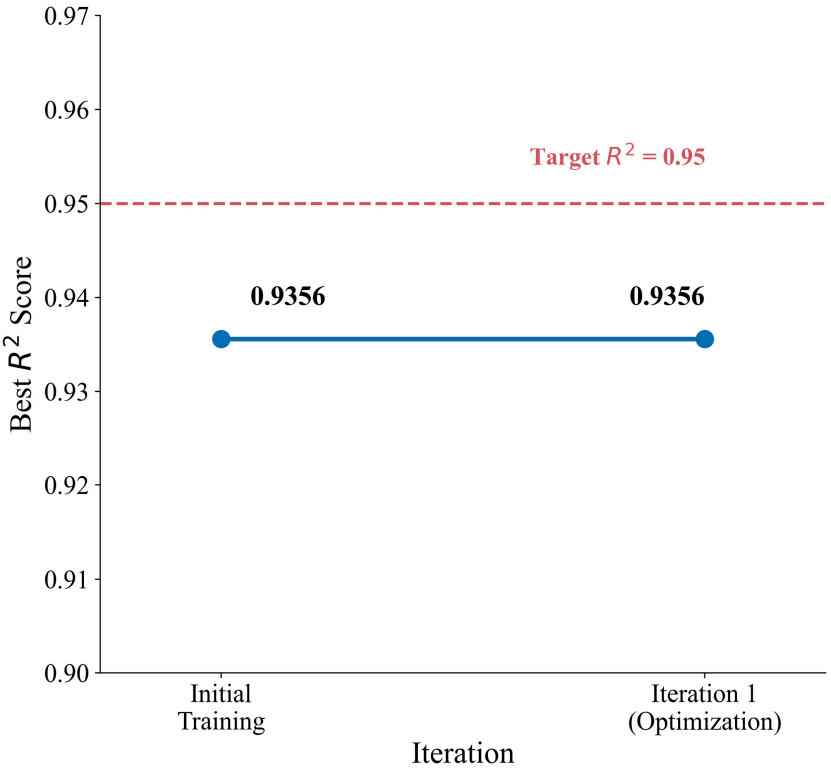

Figure 9. History of R^2 convergence of GB model.

4.3.1. Nested Cross-Validation on the Top-Three Models

The CV–test divergence observed above raises the question of whether the test value itself is reliable at n = 21. We therefore applied a nested 5 × 5 cross-validation on the full standardized dataset, with a RandomizedSearchCV inner loop using the same search spaces as in Table 3 and 50 random samples per inner fit. Figure 10 shows the mean and standard deviation of the outer-fold R^2 for GB, SVR, and RF.

Under nested CV, the three models perform similarly, with overlapping distributions. SVR has the highest mean (0.858) and the smallest variance (0.053), while GB (0.838 ± 0.146) and RF (0.839 ± 0.114) have long lower tails and include one outer fold with R^2 near 0.58 and 0.64 respectively. Two implications follow. First, the un-leaked performance ceiling on this dataset is closer to 0.84–0.86 than to the single-split 0.9356, confirming the Monte-Carlo picture. Second, the lower tail of the tree-based models is real and stems from a subset of folds on which either a high-PPV extreme or a low-PPV cluster is under-represented; this is an artifact of n = 102, not of the algorithm family.

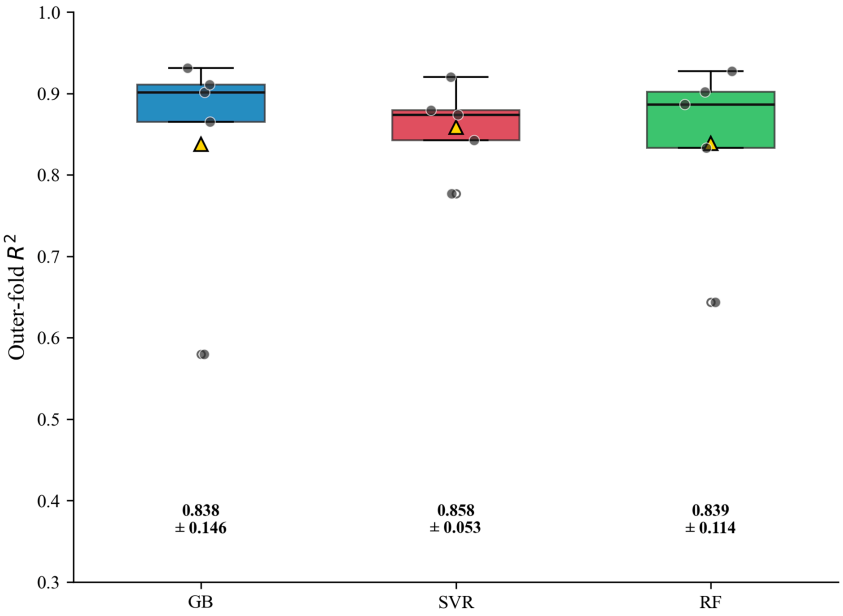

Figure 10. Nested 5 × 5 cross-validation outer-fold R^2 for the top-three models.

4.3.2. Comparison with Grid Search and with Default-Parameter Baselines

To check whether the negative outcome of the randomized search was attributable to a mismatched search space, we added a constrained grid search on Gradient Boosting using only four hyperparameters (n_estimators ∈ {50, 100, 200}, learning_rate ∈ {0.05, 0.1}, max_depth ∈ {3, 5}, subsample ∈ {0.9, 1.0}), i.e., 24 combinations × 5-fold CV = 120 fits. The results are summarized in Figure 11.

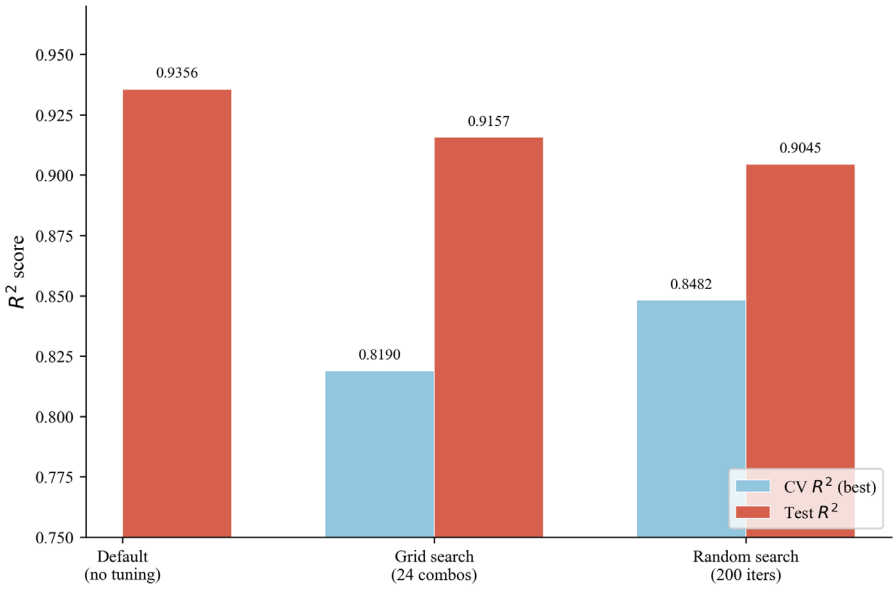

Figure 11. CV-best and held-out test R^2 for Gradient Boosting under three tuning regimes.

All three scores are within the 95% CI of GB in Table 5, i.e., not statistically separable on this split. However, two observations are reproducible. First, the default scikit-learn GB hyperparameters are strongly competitive on this dataset, and in fact were not exceeded by either search. Second, the randomized search has a higher best CV score but a lower test score than the grid search, which is the canonical signature of CV over-optimism on small samples: a broader search finds more configurations that happen to win the inner folds without generalizing.

4.4. Predicted Versus Measured PPV

The predicted versus measured PPV scatter plots for the four best-performing models are presented in Figure 12. In all four subplots, the data points cluster closely along the 1:1 diagonal line, confirming strong agreement between predicted and measured PPV values.

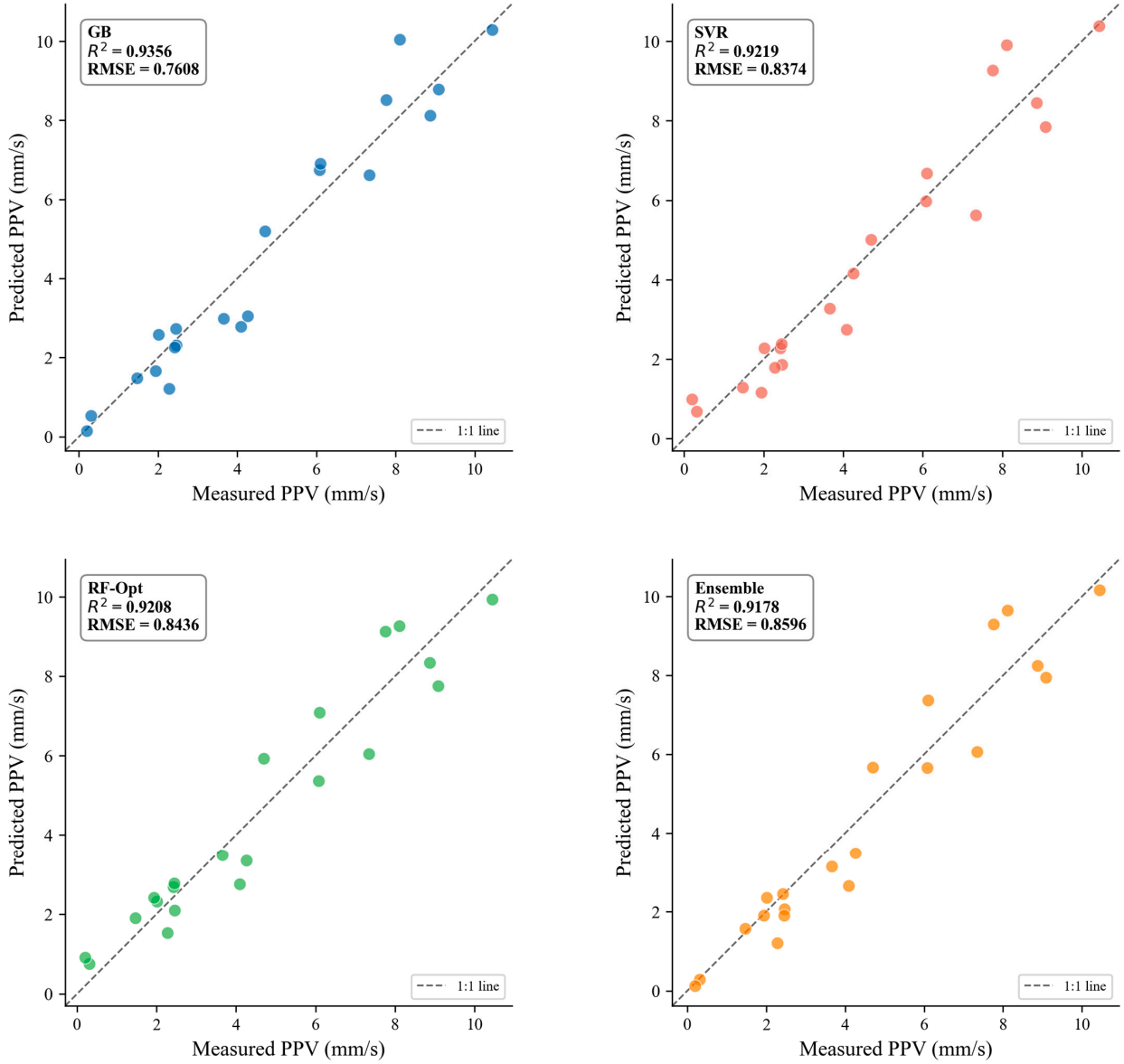

Figure 12. Measured vs. predicted PPV.

Gradient Boosting exhibits the tightest clustering around the 1:1 line, with most predictions deviating by less than 1 mm/s from the measured values. The model performs consistently well across the full range of PPV values (0.13–11.05 mm/s), with no systematic bias toward under- or over-prediction. SVR shows similarly strong alignment, though with slightly more scatter at higher PPV values (>7 mm/s). The optimized Random Forest demonstrates good overall performance but shows a tendency to slightly under-predict in the mid-range (3–6 mm/s). The Voting ensemble achieves balanced predictions, benefiting from the error-averaging effect of combining multiple models.

4.5. Residual Analysis

A residual analysis of the best model (Gradient Boosting) is presented in Figure 13, comprising three diagnostic plots: (a) residuals versus predicted values, (b) the residual distribution with a normal fit, and (c) a normal Q-Q plot.

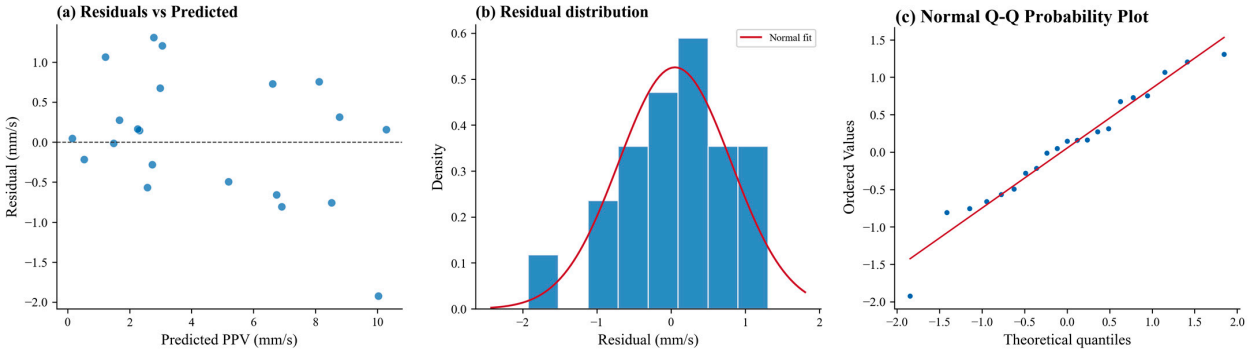

Figure 13. Residual of the GB model.

The residuals-versus-predicted plot shows that the residuals are distributed approximately symmetrically around zero across the full range of predicted values, with no obvious funnel-shaped pattern. The majority of residuals fall within the ±1 mm/s band, with only a few data points exhibiting residuals exceeding ±1.5 mm/s.

The histogram of residuals reveals an approximately normal distribution centered near zero, which indicates an unbiased model. The residual standard deviation is approximately 0.75 mm/s. The Q-Q plot confirms that the residuals largely follow a normal distribution, with minor deviations in the tails. These diagnostic results collectively suggest that the GB model provides reliable and well-calibrated predictions for PPV, with no systematic error patterns that would indicate model misspecification.

4.6. Feature Importance Analysis

The relative feature importance, normalized to percentages, is compared across five tree-based models (GB, RF, XGBoost, RF-Opt, and GB-Opt) in Figure 14. Powder factor (PF) is consistently identified as the most important predictor across all five models, with relative importance ranging from 28.7% (RF-Opt) to 50.5% (XGBoost). This finding aligns with the strong Pearson correlation between PF and PPV (r = 0.783) observed in Section 2.3 and is physically intuitive: a higher explosive concentration per unit volume of rock directly translates to stronger blast-induced vibration [34]. Distance (DI) is the second most important feature, contributing 14.2–19.0% of the total importance across models. This reflects the well-established principle of vibration attenuation with distance from the blast source. Burden-to-spacing ratio (B/S) ranks third, with importance values ranging from 16.3% to 28.9%. The B/S ratio governs the confinement geometry of the blast, influencing the efficiency of energy transfer from the explosive charge to ground vibration. Stemming length (St) contributes 9.1–15.8%, reflecting its moderating role in confining explosive gases and reducing surface vibration. Hole depth (HD) shows relatively low importance (2.7–11.5%), suggesting that its influence on PPV is partially captured by other correlated variables (e.g., PF and B/S). Maximum charge per delay (Q_{max}) is consistently identified as the least important feature (1.4–6.3%), which contrasts with its traditional prominence in empirical PPV prediction equations such as the USBM formula. This finding suggests that in the multivariate machine learning context, the predictive information carried by Q_{max} is largely subsumed by the powder factor and distance, which are more direct physical determinants of ground vibration intensity.

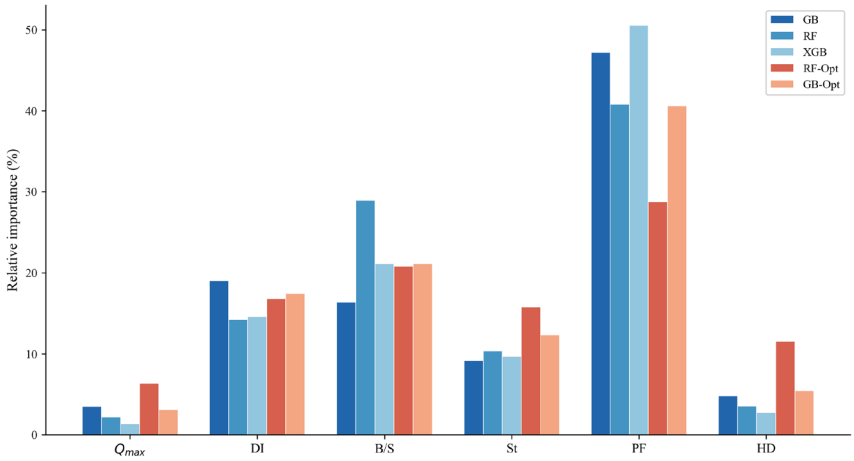

Figure 14. Feature importance identified by different learning models.

5. Discussion

In this study, the proposed framework automated the model-development pipeline from prepared dataset input through training, evaluation, optimization, and reporting. The best model (Gradient Boosting, R^2 = 0.9356) achieved competitive predictive accuracy for the available dataset. The agent workflow produced three findings: (i) a consistent re-ranking of blasting controls that promotes powder factor and demotes maximum charge per delay relative to empirical USBM-style models; (ii) a direct, quantified demonstration on the 102-event dataset that random search improved cross-validated R^2 while reducing held-out test R^2 for both GB and SVR, which is the canonical signature of small-sample CV over-fitting; and (iii) a metric-dependent trade-off in which the Voting ensemble achieves the lowest MAPE (16.80%) even though Gradient Boosting achieves the highest R^2.

The Orchestrator–Evaluator–Optimizer architecture, prompt roles, hyperparameter-space templates, and stopping rules are dataset-agnostic and can be reused for other tabular geotechnical regression tasks such as settlement prediction, slope factor-of-safety estimation, or rock-burst intensity. The fitted PPV models, however, are tied to the Xiushan tunnel site (lithology, blast geometry, and charge range) and therefore should be retrained before deployment elsewhere. Cross-site generalization, transfer learning, and domain adaptation on multi-site PPV datasets remain named open directions.

The small dataset (n = 102) is the dominant limitation of the study. The 80/20 split leaves only 21 test cases, and 5-fold cross-validation on 81 training samples results in folds of roughly 16 cases. This setting helps explain why random search improved CV scores for GB and SVR yet reduced their held-out test performance, indicating fold-specific overfitting. Under the Monte-Carlo split benchmark, the R^2 distributions of the top five nonlinear models overlap substantially, with CI widths of roughly 0.15–0.25. Under nested 5×5 CV, the mean R^2 for the three previously top-ranked models falls to 0.838–0.858 with overlapping variance. Under the constrained grid search, neither grid nor random tuning beat the default GB on the held-out test. Collectively, these results point to the dataset rather than the algorithm family as the binding constraint: with n = 102 and a 80/20 split, the residual variance that a more elaborate search could exploit is comparable to the split-induced variance itself, so additional search trades off only between CV over-optimism and test under-optimism.

A second limitation is the absence of external baselines. This paper is not a head-to-head comparison against Auto-Sklearn, TPOT, H2O AutoML, or FLAML. The value added by the LLM-assisted layer in this study is not an automated pursuit of higher R^2 but the production of a human-readable reasoning trail that connects model diagnostics to the next optimization action and to the stopping decision. A multi-platform, controlled

benchmark against AutoML libraries on a larger multi-site PPV dataset is an explicit follow-up direction.

Thirdly, the scalability of the agent workflow implies that two scalability concerns remain for larger datasets: prompt-length growth with per-model verbose metrics, and the wall-clock cost of nested CV at the scale of a multi-site database. Extending the workflow with distributed CV execution and structured tool calls for the LLM is a named follow-up direction.

6. Conclusions

This study developed an LLM-assisted Evaluator-Optimizer workflow for PPV prediction in tunnel blasting. Applied to a 102-event tunnel-blasting field dataset, the proposed LLM-assisted Evaluator–Optimizer workflow enabled fully automated model comparison, targeted tuning, ensemble construction, and reporting under a deterministic and inspectable search protocol. The principal conclusions are as follows:

(1) Across 50 Monte-Carlo 80/20 splits, the nonlinear models—KNN, Voting ensemble, SVR, MLP, and GB—form a leading cluster with overlapping 95% CIs, rather than a uniquely best model. Paired Wilcoxon tests show that KNN and the Voting ensemble are statistically indistinguishable; GB is significantly lower than the Voting ensemble ($p < 0.001$, rbc = +0.56). The single-split $R^2 = 0.9356$ reported for GB is therefore an upper-tail single-split outcome, not a site-independent estimate.

(2) Under nested 5×5 cross-validation, the mean R^2 for GB, SVR, and RF falls to 0.838, 0.858, and 0.839, respectively. This narrows the dataset-imposed ceiling to about 0.84–0.86 and explains why randomized and grid-based tuning both failed to exceed default-parameter Gradient Boosting on the held-out test: the limiting factor is the single-site sample size, not the algorithm family.

(3) Across five independent tree-family models, the powder factor is the dominant predictor (28.7–50.5% relative importance), while the maximum charge per delay—the primary variable in classical USBM-type empirical models—is consistently the least important (1.4–6.3%). Burden-to-spacing ratio and stemming length carry non-trivial predictive information that USBM-style models do not use.

The study therefore contributes a portable agentic architecture for tabular geotechnical regression, three quantitative scientific findings robust across models and resampling schemes, and a concrete cautionary result about single-split benchmarking. Three directions remain open: (i) multi-site external validation and transfer learning; (ii) controlled benchmarking against mature AutoML frameworks (Auto-Sklearn, TPOT, H2O AutoML, FLAML); and (iii) extension of the LLM layer from a natural-language planner to a structured tool-call interface for large-sample settings.

Author Contributions: Conceptualization, J.X., H.F. and D.J.A.; methodology, J.X. and H.F.; software, J.X.; validation, H.F. and D.J.A.; formal analysis, J.X. and H.F.; investigation, J.X.; resources, D.J.A.; data curation, J.X. and H.F.; writing—original draft preparation, J.X., H.F. and D.J.A.; writing—review and editing, J.X., H.F. and D.J.A.; visualization, J.X.; supervision, D.J.A.; project administration, J.X.; funding acquisition, J.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by 2025 Construction Research Project (Self-Funded), Grant number: 2025K168, and Zhejiang College of Construction Special Cultivation Project for Urban Infrastructure Lifelines, Grant number: Z202510.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: Author Haiping Fan was employed by the company China Railway Engineering Design and Consulting Group Co., Ltd., Beijing, China. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. He, B.; Armaghani, D.J.; Lai, S.H. Assessment of Tunnel Blasting-Induced Overbreak: A Novel Metaheuristic-Based Random Forest Approach. *Tunn. Undergr. Space Technol.* **2023**, *133*, 104979. [CrossRef]
2. Li, H.; Chen, D.; You, G.; Han, H.; Zhang, Q.; Shen, C.; Duan, T.; Jiang, X.; Wang, P. Research on Non-Stationary Blasting Vibration Prediction Models and Analysis of Equal-Interval Delay for Vibration Reduction. *Bull. Eng. Geol. Environ.* **2026**, *85*, 46. [CrossRef]
3. Xia, X.; Li, H.B.; Li, J.C.; Liu, B.; Yu, C. A Case Study on Rock Damage Prediction and Control Method for Underground Tunnels Subjected to Adjacent Excavation Blasting. *Tunn. Undergr. Space Technol.* **2013**, *35*, 1–7. [CrossRef]
4. Chen, D.; Chen, Y.; Ye, W.; Zheng, H.; Lei, G.; Lai, Q. Study on the Propagation Law of Blast-Induced Vibration Waves in Landfill. *Eng. Geol.* **2025**, *351*, 108038. [CrossRef]
5. Guan, X.; Xu, H.; Fu, H.; Zhang, W.; Li, P.; Ding, H.; Yu, K.; Zhang, S. Vibration Characteristics, Attenuation Law and Prediction Method in the near Field of Tunnel Blasting. *Case Stud. Constr. Mater.* **2023**, *19*, e02662. [CrossRef]
6. He, B.; Armaghani, D.J.; Lai, S.H.; He, X.; Asteris, P.G.; Sheng, D. A Deep Dive into Tunnel Blasting Studies between 2000 and 2023—A Systematic Review. *Tunn. Undergr. Space Technol.* **2024**, *147*, 105727. [CrossRef]
7. Longjun, D.; Xibing, L.; Ming, X.; Qiyue, L. Comparisons of Random Forest and Support Vector Machine for Predicting Blasting Vibration Characteristic Parameters. *Procedia Eng.* **2011**, *26*, 1772–1781. [CrossRef]
8. Bhattacharjee, N.; Sur, A.; Manna, B.; Banerjee, A.; Shahu, J.T. ANN-Based Approach for the Prediction of Peak Particle Velocity of Ground Induced by Underground Metro Operations. *Transp. Infrastruct. Geotechnol.* **2025**, *12*, 96. [CrossRef]
9. Asteris, P.G.; Armaghani, D.J. An Empirical-Driven Machine Learning (EDML) Approach to Predict PPV Caused by Quarry Blasting. *Bull. Eng. Geol. Environ.* **2025**, *84*, 200. [CrossRef]
10. Amoako, R.; Jha, A.; Zhong, S. Rock Fragmentation Prediction Using an Artificial Neural Network and Support Vector Regression Hybrid Approach. *Mining* **2022**, *2*, 233–247. [CrossRef]
11. Dindarloo, S.R. Peak Particle Velocity Prediction Using Support Vector Machines: A Surface Blasting Case Study. *J. South. Afr. Inst. Min. Metall.* **2015**, *115*, 637–643. [CrossRef]
12. Saadat, M.; Khandelwal, M.; Monjezi, M. An ANN-Based Approach to Predict Blast-Induced Ground Vibration of Gol-E-Gohar Iron Ore Mine, Iran. *J. Rock Mech. Geotech. Eng.* **2014**, *6*, 67–76. [CrossRef]
13. Choi, Y.-H.; Lee, S.S. Predictive Modelling for Blasting-Induced Vibrations from Open-Pit Excavations. *Appl. Sci.* **2021**, *11*, 7487. [CrossRef]
14. Ma, T.; Chen, C.; Shen, L.; Luo, K.; Jiang, Z.; Liu, H.; Hu, X.; Lin, Y.; Peng, K. A Novel Social Network Search and LightGBM Framework for Accurate Prediction of Blast-Induced Peak Particle Velocity. *Front. Struct. Civ. Eng.* **2025**, *19*, 645–662. [CrossRef]
15. Yu, Z.; Shi, X.; Zhou, J.; Chen, X.; Qiu, X. Effective Assessment of Blast-Induced Ground Vibration Using an Optimized Random Forest Model Based on a Harris Hawks Optimization Algorithm. *Appl. Sci.* **2020**, *10*, 1403. [CrossRef]
16. Bozkurt Keser, S.; Yavuz, M.; Erten, G.E. Advancing the Prediction and Evaluation of Blast-Induced Ground Vibration Using Deep Ensemble Learning with Uncertainty Assessment. *Geosciences* **2025**, *15*, 182. [CrossRef]
17. Gu, J.; Zhang, Y.; Li, C.; Zhou, J.; Onifade, M.; Khandelwal, M. Hybrid Machine Learning with Metaheuristic Optimization for Predicting Peak Particle Velocity in Open-Pit Mines. *Min. Metall. Explor.* **2026**, *43*, 793–818. [CrossRef]
18. Shylaja, G.; Prashanth, R. A Systematic Survey of Hybrid ML Techniques for Predicting Peak Particle Velocity (PPV) in Open-Cast Mine Blasting Operations. *Artif. Intell. Rev.* **2025**, *58*, 203. [CrossRef]
19. Huang, J.; Koopialipoor, M.; Armaghani, D.J. A Combination of Fuzzy Delphi Method and Hybrid ANN-Based Systems to Forecast Ground Vibration Resulting from Blasting. *Sci. Rep.* **2020**, *10*, 19397. [CrossRef]
20. Dumakor-Dupey, N.K.; Arya, S.; Jha, A. Advances in Blast-Induced Impact Prediction—A Review of Machine Learning Applications. *Minerals* **2021**, *11*, 601. [CrossRef]
21. He, B.; Lai, S.H.; Mohammed, A.S.; Muayad, M.; Sabri, S.; Ulrikh, D.V. Estimation of Blast-Induced Peak Particle Velocity through the Improved Weighted Random Forest Technique. *Appl. Sci.* **2022**, *12*, 5019. [CrossRef]
22. Guo, J.; Zhang, C.; Xie, S.; Liu, Y. Research on the Prediction Model of Blasting Vibration Velocity in the Dahuangshan Mine. *Appl. Sci.* **2022**, *12*, 5849. [CrossRef]
23. Xiong, S.; Liu, Z.; Min, C.; Shi, Y.; Zhang, S.; Liu, W. Compressive Strength Prediction of Cemented Backfill Containing Phosphate Tailings Using Extreme Gradient Boosting Optimized by Whale Optimization Algorithm. *Materials* **2022**, *16*, 308. [CrossRef] [PubMed]
24. Daniel, C.; Cheng, S.; Yin, X.; Barrie, Z.M.; Pan, Y.; Liu, Q.; Gao, F.; Li, M.; Huang, X. AI-Aided Short-Term Decision Making of Rockburst Damage Scale in Underground Engineering. *Undergr. Space* **2025**, *23*, 362–378. [CrossRef]

25. Yan, W.; Wu, J.; Chen, G.; Su, S.; Shen, P.; Wong, D.F.; Zhou, W.-H. Standardization and Interpretable Analysis of Geological Database Using Retrieval-Augmented Large Language Model. *Geod. AI* **2025**, *6*, 100047. [CrossRef]
26. Yu, C.; Koopialipoor, M.; Murlidhar, B.R.; Mohammed, A.S.; Armaghani, D.J.; Mohamad, E.T.; Wang, Z. Optimal ELM–Harris Hawks Optimization and ELM–Grasshopper Optimization Models to Forecast Peak Particle Velocity Resulting from Mine Blasting. *Nat. Resour. Res.* **2021**, *30*, 2647–2662. [CrossRef]
27. Fan, H.; Xu, J.; Armaghani, D.J. Addressing Environmental Challenges in Blasting: A Dual-Method Data Generation Framework for Peak Particle Velocity Prediction. *Appl. Soft Comput.* **2026**, *186*, 114263. [CrossRef]
28. Gaopale, K.; Sasaoka, T.; Hamanaka, A.; Shimada, H. Integrated Capuchin Search Algorithm-Optimized Multilayer Perceptron for Robust and Precise Prediction of Blast-Induced Airblast in a Blasting Mining Operation. *Geosciences* **2025**, *15*, 306. [CrossRef]
29. Cui, H.; Zheng, P.; Ren, M.; Yan, Y. An LLM-Based Cross-Domain Knowledge Retrieval Augmented Generation Method for Bio-Inspired Solution Design. *Adv. Eng. Inform.* **2026**, *69*, 104017. [CrossRef]
30. Chen, C.; Li, N.; Qi, J.; Chang, H.; Shi, W.; Xie, J.; Yuan, J.; Yang, H.; Guo, J.; Xu, C.; et al. Leveraging LLMs for Environmental Complexity: Structured Fine-Tuning Data Sets and Deployment Strategies. *Environ. Sci. Technol.* **2025**, *60*, 497–509. [CrossRef] [PubMed]
31. Yusoof, A.; Mohamad, E.T.; Saidin Misnan, M.; Afrazi, M.; Jahed Armaghani, D. Engineering Performance of Data-Driven Powder Factor Optimization in Tunnel Blasting Under Complex Geological Conditions. *Geosciences* **2025**, *15*, 441. [CrossRef]
32. Asteris, P.G.; Tsavdaridis, K.D.; Lemonis, M.E.; Ferreira, F.P.V.; Le, T.T.; Gantes, C.J.; Formisano, A. AI-powered GUI for prediction of axial compression capacity in concrete-filled steel tube columns. *Neural Comput. Appl.* **2024**, *36*, 22429–22459. [CrossRef]
33. Asteris, P.G.; Lemonis, M.E.; Nguyen, T.A.; Van Le, H.; Pham, B.T. Soft computing-based estimation of ultimate axial load of rectangular concrete-filled steel tubes. *Steel Compos. Struct. Int. J.* **2021**, *39*, 471–491.
34. Zhao, J.; Nguyen, H.; Nguyen-Thoi, T.; Asteris, P.G.; Zhou, J. Improved Levenberg–Marquardt backpropagation neural network by particle swarm and whale optimization algorithms to predict the deflection of RC beams. *Eng. Comput.* **2022**, *38*, 3847–3869. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.