# Forecasting AI Model Computational Requirements Using Random Forest and XGBoost with Entity and Domain Characteristics

Astika Ayuningtyas[1,*], Rindi Nur Wulandari[2]

[1,2]*Department of informatics, Adisutjipto of Aerospace Technology, Yogyakarta, Indonesia*

(Received: November 30, 2023; Revised: January 5, 2024; Accepted: February 10, 2024; Available online: June 1, 2024)

**Abstract**

This research aims to predict the computational power required by artificial intelligence (AI) models, specifically measured in petaFLOP (Floating Point Operations Per Second), based on their domain and entity characteristics. The study employs Random Forest and XGBoost regression models to predict the amount of computational power needed by AI models. Both models were trained on a dataset that includes features such as the training year, domain (e.g., Language, Vision), and entity characteristics. The results demonstrate that the Random Forest model outperforms XGBoost in terms of prediction accuracy, as indicated by higher R-squared values and lower error metrics. Feature importance analysis revealed that the year of training and domain were the most significant predictors of computational power, with the Language domain emerging as the most influential in both models. The findings highlight the potential for machine learning models to forecast AI computational requirements, which can aid organizations in optimizing computational resources for AI projects. However, the study faces limitations due to data sparsity, particularly in the target variable, and the relatively simple nature of the models employed. Future work should explore incorporating additional features, such as hardware specifications, and leveraging deep learning models to better capture the complexity of AI computational demands. This study lays the groundwork for further research into more precise predictions of AI model resource consumption, helping organizations plan their AI initiatives more effectively.

*Keywords:* AI Model Prediction, Computational Power, Random Forest, Xgboost, Feature Importance

## 1. Introduction

The increasing demand for computational resources in AI model training stems from the necessity to optimize performance while ensuring efficient resource allocation. As AI applications become more complex and resource-intensive, predicting computational requirements has become critical for maintaining operational efficiency and minimizing costs. One of the primary challenges is the variable nature of workloads in cloud computing environments. Karamthulla et al. highlight that traditional manual management of resources falls short due to dynamic user demands and diverse workloads, necessitating automated solutions for optimal resource allocation (Karamthulla et al., 2023). AI-driven approaches can significantly enhance the efficiency of resource utilization by leveraging automation to adapt resource distribution in real-time. This is particularly important in DevOps operations, where rapid provisioning is essential, as noted by [1]. These necessities underscore the role of sophisticated algorithms in predicting workloads and dynamically managing resources, ultimately enhancing operational performance.

Further expanding on the integration of AI for resource allocation, Islam presents a framework for dynamic resource allocation specifically for AI and machine learning applications within edge computing environments. This framework employs optimization techniques that account for workload attributes and resource availability, thereby addressing the performance and latency challenges inherent in AI/ML applications [2]. Such frameworks are critical as they demonstrate improvements in resource utilization, which can significantly reduce latency and enhance overall model training performance.

Moreover, optimizing hardware conditions in AI training is imperative to meet the increased computational demands. Chen argues that hardware-accelerated optimization strategies are essential for enhancing the training and inference efficiency of AI models. This includes examining neural network accelerators and energy-efficient computing, which

***Corresponding author: Astika Ayuningtyas (astika@itda.ac.id)**
**DOI: https://doi.org/10.47738/ijaim.v4i2.82**

are crucial considerations in resource allocation strategies [3]. The intricate relationship between hardware capabilities and AI model requirements necessitates that resource management frameworks be adaptable and predictive in their approach.

Finally, strategies aimed at improving the efficiency of deep learning models indicate a growing trend toward designing algorithms that require fewer computational resources without compromising performance. This need is increasingly relevant as AI technologies continue to evolve, necessitating a dual focus on both resource efficiency and output effectiveness, as explored by [4]. The integration of AI techniques into resource optimization not only translates into better performance but also lays the groundwork for sustainable AI developments, which is crucial for long-term operational success in crowded computational fields.

The relationship between different types of AI models, the domains they operate within, and their respective computational power requirements is complex and multifaceted. As AI technologies evolve, the corresponding computational demands vary significantly across different applications and model architectures. Understanding these relationships is crucial for optimizing resource allocation and performance in AI systems.

AI encompasses various subfields, including machine learning (ML), deep learning, and neural networks, each with distinct computational characteristics. Machine learning techniques often require less computational power compared to deep learning, which involves multiple layered architectures and processes vast amounts of data. This distinction is especially pronounced in domains with high-dimensional data, such as image recognition and natural language processing, where deep learning models, such as convolutional neural networks (CNNs), become computationally expensive due to the need for extensive training on large datasets [5]. Performance metrics such as resource utilization and throughput are essential in assessing these models' effectiveness, and model choice directly impacts these metrics.

Different domains also influence the computational power needed. For instance, AI applications in healthcare utilize ML techniques that can analyze patient data and support clinical decision-making. In particular, studies have shown that machine learning can effectively be applied within intensive care unit (ICU) settings and other medical contexts to optimize patient outcomes [6], [7]. While requiring significant resources, these models are generally optimized to enhance performance relative to the specific needs of intensive care settings. Additionally, the dynamic nature of workloads in healthcare settings necessitates adaptive resource management strategies, especially when computational demands fluctuate based on incoming data flow [8].

Moreover, the deployment environment affects computational requirements. Cloud-based AI systems leverage powerful servers for extensive data processing, thus typically demanding more computational resources. Conversely, edge AI systems allow for local processing, facilitating real-time analysis with reduced latency, although they may be limited by device computational capabilities. This adaptability is crucial for fields like robotic process automation in engineering, where operational efficiency is vital [9].

In addition to domain-specific variations, the architecture of AI models also plays a pivotal role in determining computational requirements. Adversarial models, which are designed for environments with unpredictable actions—such as gaming—demand high computational power due to the complexity of decision-making processes [10]. A model utilizing Monte Carlo Tree Search, for example, requires significant computational resources because of its reliance on large state space explorations.

The main objective of this research is to predict the amount of computational power (measured in petaFLOP) required for training AI models based on their domain and entity characteristics. By leveraging machine learning techniques, specifically Random Forest and XGBoost, the study aims to create models that can accurately forecast the computational needs of different AI models, taking into account various factors such as model complexity, the nature of the tasks being performed, and the domain in which the model is applied. This prediction will provide organizations with valuable insights into the computational resources required for their AI projects, helping them better prepare for large-scale model training and optimize their hardware usage.

The significance of this study lies in its potential to assist organizations in more efficiently managing their computational resources. As AI models become increasingly complex, the demand for computational power grows, and accurately forecasting these needs can help avoid resource wastage and reduce operational costs. By understanding

the computational requirements based on model entity and domain characteristics, companies can plan their AI projects more effectively, ensuring that they have the right infrastructure in place to support model training without over- or under-provisioning resources. This research, therefore, contributes to improving AI project planning and resource allocation, ultimately aiding in the scaling and deployment of AI technologies.

## 2. Literature Review

### 2.1. Computational Power in AI Training

The computational needs of various AI models have garnered significant attention due to the extensive resources required for model training and deployment. The performance of these models is often measured in petaFLOPs (peta Floating Point Operations per Second), a crucial metric that quantifies the computational power necessary to process large datasets and run complex algorithms. Understanding these requirements is fundamental for optimizing AI deployments across different domains, as highlighted by current research.

Different AI model architectures exhibit varying computational demands. For instance, deep learning models, especially those employing convolutional and recurrent neural networks, necessitate considerable computational resources due to their numerous parameters and layers. These models are particularly prevalent in fields like computer vision and natural language processing, where they must handle vast amounts of data. In these contexts, researchers emphasize the relationship between data scale and model performance, indicating that as training datasets increase, the computational burden escalates [11].

The importance of petaFLOP calculations cannot be understated, as they serve as a benchmark for evaluating the capabilities of AI hardware. The computational requirements for training advanced AI models often fall within the realm of petaFLOPs, making them unattainable for standard consumer-grade hardware. Moon et al. discuss how deep learning systems must have significant infrastructure support to operate effectively, strongly tying success in the field to high computational performance metrics [12]. This situation leads to an uneven landscape where only a few organizations with access to substantial computational resources can develop and deploy state-of-the-art AI technologies [13].

Additionally, the design and optimization of AI-specific hardware have become critical in addressing these computational needs. Solutions such as hardware-accelerated optimization strategies and energy-efficient computing architectures are being explored to facilitate the efficient processing of intensive AI workloads. The challenges of managing and deploying these models are further compounded in cloud environments, where factors like network latency, data management, and architectural scalability play crucial roles in the efficiency of AI applications [14].

Furthermore, specific sectors illustrate varied computational requirements, impacting the overall resource allocation strategies. For example, in healthcare, AI models can enhance clinical outcomes through predictive analytics and decision support systems; however, the complexity of these models necessitates substantial computational power [15]. Similarly, in adversarial AI scenarios, high computational performance is essential for real-time decision-making and responsiveness, as noted by Justesen et al. [10].

### 2.2. Regression Models in Predicting Computational Needs

A review of previous studies utilizing regression models such as Random Forest and XGBoost for predicting resource usage reveals significant insights across various domains. These models have emerged as powerful tools for forecasting resource consumption in settings ranging from healthcare to cloud computing, highlighting their versatility and effectiveness.

Halabhavi emphasizes the role of predictive analytics in healthcare, showing that regression models can significantly enhance patient outcomes through improved resource allocation. The study indicates strong predictive power associated with patient risk stratification and demand forecasting, capturing approximately 58% of the variance in patient outcomes. This directly correlates predictive analytics with better resource utilization in clinical settings, underscoring its importance [16].

In the real estate sector, Rampini and Cecconi explore the application of XGBoost in predicting market prices, validating its effectiveness through comparative studies with other regression techniques. Their findings assert that XGBoost provides superior performance due to its ability to handle large datasets efficiently, further indicating the model's robustness in making predictions that align with real resource allocation trends in the property market [17].

Furthermore, the research by Nawrocki and Smendowski investigates cloud computing resource optimization through various machine learning methods, including XGBoost. Their findings suggest that long-term forecasting using algorithms like XGBoost is dynamic and proactive, thereby offering effective strategies for optimizing resource use in cloud environments. This study highlights the computational benefits of employing advanced regression techniques in resource management scenarios [18].

Kao et al. extend the application of regression models in the medical field by predicting the invasiveness of lung adenocarcinoma using logistic regression. Although primarily focused on classification, the insights drawn reveal the potential for regression techniques to aid in resource allocation for surgical interventions, illustrating a critical intersection between predictive modeling and clinical decision-making [19].

Moreover, in the realm of stroke prediction, Xu et al. developed machine learning models incorporating Random Forest and logistic regression to forecast hemorrhage transformation in acute ischemic stroke patients. This work demonstrates the potential for these regression models to contribute not only to clinical outcomes but also to the efficient allocation of medical resources through accurate predictive capabilities [20].

To summarize, regression models such as Random Forest and XGBoost prove to be effective tools across various domains for predicting resource usage. The studies reviewed illustrate their applicability in enhancing decision-making processes related to resource allocation, whether in healthcare, real estate, or cloud computing. These findings support the ongoing implementation and refinement of regression techniques to optimize resource management strategies in diverse fields.

## 2.3. AI Model Domains and Their Impact

The computational requirements of AI systems are significantly influenced by the domain in which they operate, particularly as the complexity of tasks and the nature of input data vary. Different domains, including healthcare, finance, environmental science, and technology, present unique challenges that dictate the choice of algorithms and the extent of computational resources needed for effective performance.

In healthcare, predictive models such as Random Forest (RF) and Extreme Gradient Boosting (XGBoost), have been shown to require substantial computational resources due to the complexity of healthcare data, which often involves high dimensionality and variability. For instance, Khera et al. employ machine learning models to predict outcomes following acute myocardial infarction and find that leveraging models like XGBoost significantly boosts predictive accuracy, demonstrating the considerable computational needs within healthcare data analysis Khera et al. (2021). Such predictive analytics in a health context demands extensive data processing capacities, as the algorithms must analyze numerous input variables.

Additionally, Baik et al. demonstrate the effectiveness of deep learning models alongside RF and XGBoost for predicting COVID-19 mortality using diverse datasets, showcasing the computational intensity associated with merging different sources of data (such as chest X-rays and electronic health records) into coherent models [21]. Their findings indicate that within the same clinical domain, the computational requirements can fluctuate depending on the available data types and integration strategies, influencing the algorithm's choice and performance.

In the financial sector, computational needs are driven by the necessity to analyze vast amounts of market data quickly. Li discusses the application of XGBoost for stock price prediction and portfolio optimization, emphasizing the need for robust computational power to capture complex market dynamics effectively [22]. This reflects the high computational load experienced in domains where real-time analysis is crucial, necessitating more sophisticated and resource-intensive modeling techniques.

Environmental science presents another area where domain significantly influences computational requirements. Research by Lee et al. on predicting Gross Primary Productivity (GPP) using satellite data highlights that the complex

 ISSN 2776-8007 

relationships inherent in ecological data can require advanced AI techniques. The study illustrates the necessity of employing machine learning models to manage nonlinearity and spatial-temporal variability, thus emphasizing the increased computational demands stemming from these complexities [23].

## 3. Method

### 3.1. Data Loading and Preprocessing

The methodology begins with the loading and cleaning of the dataset. The data is imported using pandas to create a DataFrame, and its initial structure is examined. The target variable, initially named 'Training computation (petaFLOP)', is renamed to 'target_petaflop' for clarity and simplicity. Columns are then cleaned by converting their names to lowercase, removing spaces, and substituting special characters with underscores to ensure consistency. This transformation is performed through the clean_col_names function, which applies regular expressions to remove any non-alphanumeric characters, making the column names more suitable for machine learning tasks.

In addition to cleaning the column names, missing values in the target variable are addressed. Any rows with missing values in the target variable are dropped, ensuring that only rows with valid computational requirements (petaFLOP) are used for model training. Furthermore, the 'day' column, which contains temporal information, is processed by converting the entries into datetime format. Date-related features, such as year, month, and day of the week, are then extracted from the 'day' column. This allows for a deeper understanding of how time-based features might impact the target variable. The original 'day' column is dropped after the new date features are extracted. The target variable, 'target_petaflop', is log-transformed using the np.log1p() function to stabilize the variance and ensure the data is normally distributed, which is a common technique in regression tasks to handle skewed data.

### 3.2. Exploratory Data Analysis (EDA)

Once the data is cleaned and preprocessed, Exploratory Data Analysis (EDA) is conducted to better understand the distribution and relationships of key features with the target variable. Visualizations are an essential part of EDA, and histograms are plotted to observe the distribution of both the original and log-transformed target variable. Kernel density estimates (KDE) are overlaid on these histograms to provide a smoothed view of the distribution, which can help in understanding the overall shape of the data. The target_petaflop_log distribution is especially important, as it shows the more normalized distribution after applying the logarithmic transformation.

Boxplots are used to explore the relationship between the target variable and other categorical features, such as 'domain' and 'year'. The domain feature, which refers to the area of application for the AI models, is examined for its impact on computational requirements, providing insight into how the type of domain (e.g., healthcare, finance, etc.) might influence computational needs. The year feature, derived from the 'day' column, is plotted against the target variable to assess any temporal trends in the computational power required by AI models. This analysis is crucial for identifying patterns that might not be immediately obvious from the raw data.

### 3.3. Feature Engineering and Model Preparation

In the feature engineering phase, categorical features such as 'entity' and 'domain' are transformed into numerical formats suitable for machine learning models. TargetEncoder is used to encode the 'entity' feature, which is a type of ordinal encoding that assigns a numerical value to each category based on the mean of the target variable for that category. This approach helps capture the relationship between the categorical variables and the target more effectively than traditional one-hot encoding. OneHotEncoder is employed for encoding the 'domain' feature, as it represents nominal categories that do not have a natural order. This encoder creates binary columns for each unique category in the domain feature.

After encoding the categorical features, the dataset is split into training and testing sets using train_test_split from sklearn.model_selection, with 20% of the data reserved for testing and 80% used for training the models. This ensures that the model’s performance can be evaluated on unseen data, which is critical for assessing its generalizability. The training and testing sets are then passed through the preprocessing pipeline, which handles both the categorical and numerical transformations.

## 3.4. Model Training

The next step involves training two machine learning models: Random Forest Regressor and XGBoost Regressor, both of which are well-suited for regression tasks involving complex relationships between features. Random Forest is an ensemble method that uses multiple decision trees to make predictions, combining their outputs to improve accuracy and robustness. In this implementation, the RandomForestRegressor is configured with several parameters: n_estimators=100, which determines the number of trees in the forest; random_state=42, ensuring reproducibility; and max_depth=15 and min_samples_split=10, which are set to prevent overfitting by controlling the depth of the trees and the minimum number of samples required to split a node. The min_samples_leaf=5 parameter further helps in preventing the model from overfitting by setting the minimum number of samples required at each leaf node.

XGBoost is a gradient boosting algorithm that builds trees sequentially, with each tree attempting to correct the errors made by the previous one. The XGBRegressor is trained with the following parameters: n_estimators=100, controlling the number of boosting rounds; learning_rate=0.1, which is a typical value for XGBoost to balance the speed and accuracy of the model; and max_depth=7 to prevent overfitting. Additionally, the subsample=0.8 and colsample_bytree=0.8 parameters are used to ensure that each tree is trained on a random subset of the data and features, further helping to prevent overfitting and improve generalization. Both models are trained on the processed training data.

## 3.5. Model Evaluation

After training the models, their performance is evaluated using several metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared ($R^2$). These metrics are calculated both on the log-transformed target variable (for which the models were directly trained) and the original target variable (to assess the models' ability to predict computational requirements on the original scale). The predictions for the test set are inverse-transformed using np.expm1() to convert the log-transformed predictions back to the original scale of petaFLOP. Any negative predictions are set to zero, as computational power cannot be negative.

The evaluation also includes a visual analysis, where scatter plots are used to compare the predicted values against the actual values of the target variable. The ideal prediction line (where predicted values equal actual values) is shown for reference, and deviations from this line indicate areas where the models' predictions were less accurate. The performance of the Random Forest and XGBoost models is compared to determine which model provides better predictions for forecasting the computational requirements of AI models.

## 3.6. Feature Importance

Feature importance is analyzed to identify which features contribute the most to the model's predictions. For Random Forest, the feature importance scores are extracted using the feature_importances_ attribute, and a bar plot is generated to visualize the top features. The most important features, such as 'Clean_Min_Project_Size' and 'Clean_Avg_Hourly_Rate', are expected to have a higher impact on predicting computational power, while other features like 'domain' and 'entity' may have a moderate effect. For XGBoost, the feature importance is similarly visualized, allowing for a comparative analysis between the two models to see which features are consistently important across both algorithms.

# 4. Results and Discussion

## 4.1. Results of Data Loading and Preprocessing

The dataset was successfully loaded, containing 837 entries and 5 columns: Entity, Code, Day, Training computation (petaFLOP), and Domain. However, the Code column was found to be entirely empty, so it was dropped during the preprocessing stage. The Training computation (petaFLOP) column, which represents the computational requirements for training AI models, contained a significant amount of missing values (455 rows with missing data). Consequently, these rows were removed, leaving 382 valid entries for analysis. Additionally, the 'Day' column, which contained the date when the model training occurred, was processed by extracting relevant date features such as year, month, and dayofweek. This allowed the original 'Day' column to be dropped while retaining useful temporal information for model prediction.

A log-transformed target variable was created, named target_petaflop_log, using the np.log1p transformation. This transformation helps stabilize variance and normalizes the distribution of the computational power data, which typically exhibits a highly skewed distribution. After these cleaning and preprocessing steps, the dataset contained the following features: entity, target_petaflop, domain, year, month, dayofweek, and target_petaflop_log. The cleaned dataset, now consisting of 382 rows and 7 columns, is ready for further analysis. After the data was cleaned, the features (X) and target (y) variables were prepared for modeling. The dataset contained 382 samples, with 5 features identified as numerical: 'year', 'month', and 'dayofweek', and 2 categorical features: 'entity' and 'domain'. These features were encoded appropriately for machine learning. The TargetEncoder was used for the 'entity' feature to capture the relationship between the entity and the target variable by encoding each category based on the mean of the target. The OneHotEncoder was applied to the 'domain' feature, creating binary columns for each domain, allowing the model to better understand the categorical nature of the domain. The preprocessor was fitted on the training data and then used to transform both the training and testing datasets. After transformation, no missing values were found in the TargetEncoded 'entity' column, indicating that the encoding was successful. The dataset was then split into training and testing sets, with 305 samples used for training and 77 samples reserved for testing. The shape of the processed data after encoding was 305 samples and 12 features for the training set, and 77 samples and 12 features for the testing set, reflecting the transformations applied to the original features.

## 4.2. Finding from Exploratory Data Analysis (EDA)

As part of the exploratory data analysis, the distribution of the target variable (target_petaflop) was examined. The histogram of the target variable displayed a highly skewed distribution, with most values concentrated at the lower end, indicating that most AI models in the dataset require relatively low computational power, while only a few models require very high computational resources. The log-transformed version of the target variable (target_petaflop_log) showed a more normalized distribution, which is beneficial for regression modeling as it stabilizes the variance. Additionally, the relationship between the target variable and the domain feature was analyzed using boxplots. These plots indicated that AI models in certain domains (such as Language) tended to require more computational power on average, while models in other domains, like Vision, generally required less. The analysis revealed that the domain feature plays a significant role in determining the computational requirements of AI models. Other temporal features, such as year and month, were also examined to assess trends in computational power over time, though no strong temporal patterns were observed in this dataset.

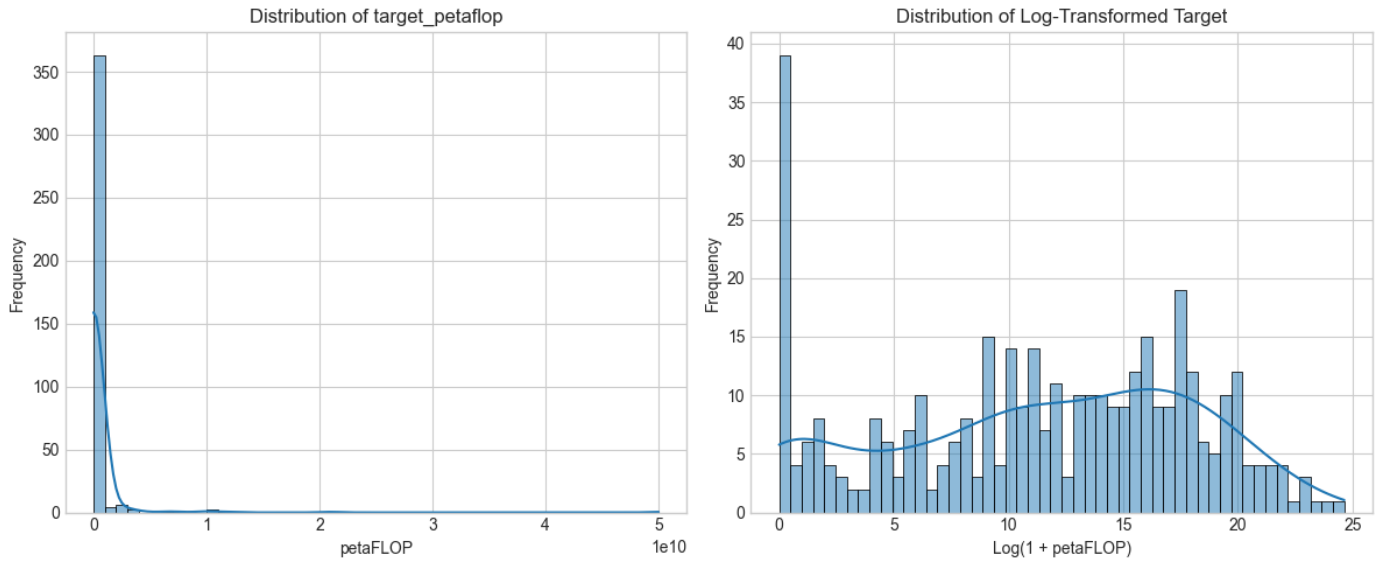


**Figure 1. Distribution of Target and Log-Transformed Target**

Figure 1 shows the distribution of the target variable, petaFLOP (left), and its log-transformed version, Log(1 + petaFLOP) (right). The petaFLOP distribution is highly skewed, with most of the values concentrated at the lower end

 ISSN 2776-8007


of the scale. This indicates that most of the AI models in the dataset require relatively low computational power. The distribution has a long tail, suggesting a small number of models require significantly more computational resources. This kind of skewed distribution is common in datasets with varying scales of computational requirements across AI models. On the other hand, the log-transformed target variable shows a more normalized distribution. The transformation effectively reduces the skewness, making the data more suitable for regression models. The distribution appears more uniform, with a moderate spread across the values. This normalization helps stabilize variance and makes the data more appropriate for machine learning algorithms, as many models perform better when the data is normally distributed.

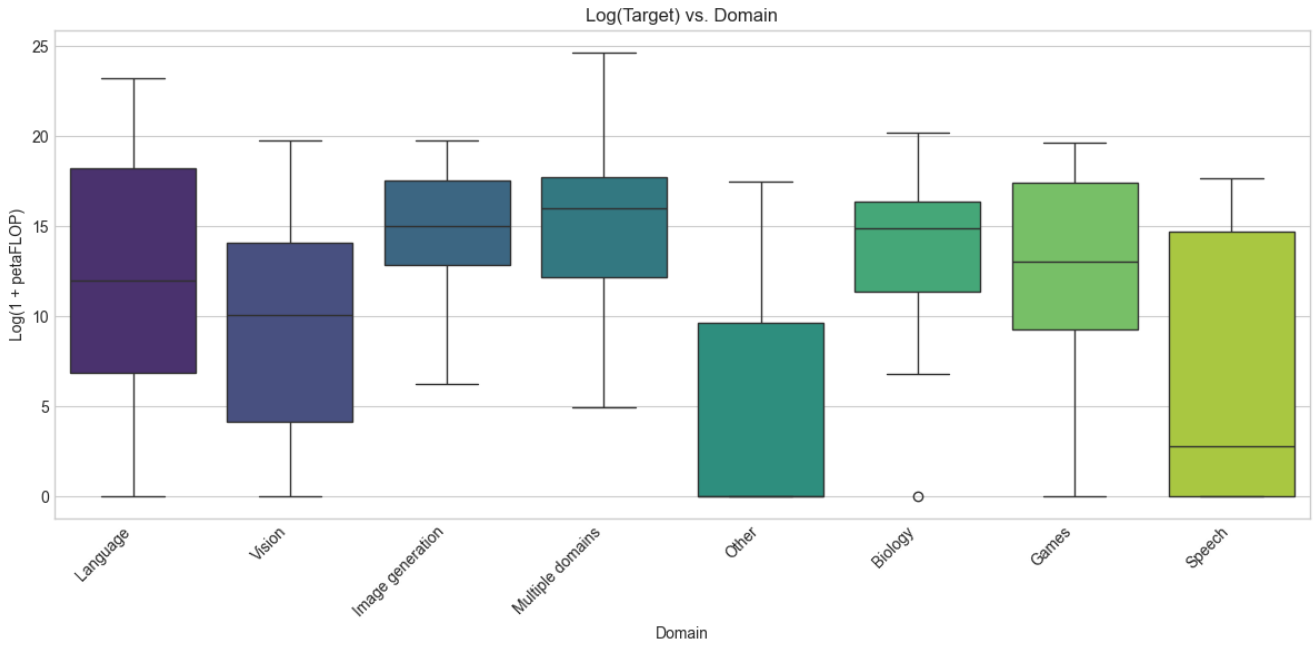


**Figure 2. Boxplot of Target vs. Domain**

Figure 2 visualizes the relationship between the log-transformed target variable and the domain feature. It uses boxplots to show the distribution of Log(1 + petaFLOP) across different AI domains. The Language domain has the highest median computational requirements, with the values spanning from lower to higher petaFLOP ranges, indicating that models in the language domain tend to have higher computational power needs. Vision and Image generation domains also have considerable computational requirements but to a lesser extent than language models. The Other domain shows a wide range of computational requirements, suggesting that AI models in this category come from diverse fields with varying computational demands. Domains such as Speech, Games, and Biology tend to have more moderate computational requirements, with the boxplots showing relatively smaller ranges and lower medians compared to Language and Vision. The presence of outliers, especially in the Other and Speech domains, indicates that some models in these categories might require significantly higher computational power than others. These visualizations indicate that the domain plays a crucial role in determining the computational requirements of AI models. Domains like Language and Vision tend to require more computational resources, while domains like Games and Biology exhibit more consistent, moderate resource demands. This insight is valuable for resource planning and optimization in AI model development.

 ISSN 2776-8007


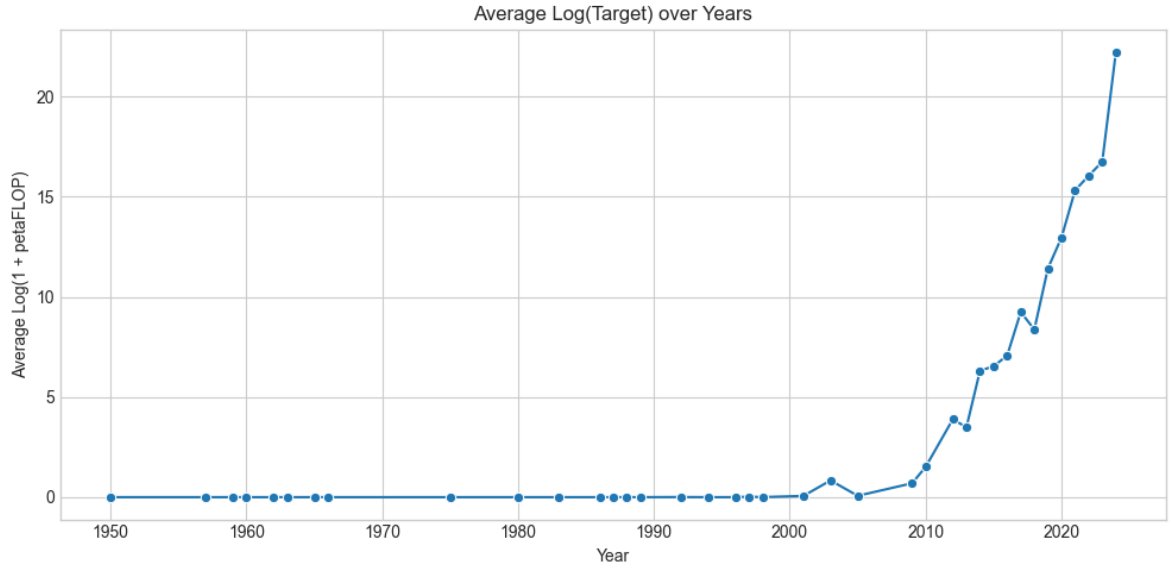


**Figure 3. Average Log(Target) Over Years**

Figure 3 visualizes the average log-transformed target variable (Log(1 + petaFLOP)) over time, specifically by year. The graph reveals an exponential increase in the computational power required for AI models after the year 2000. Prior to that, from the 1950s to the late 1990s, the average log-transformed target remained consistently low, suggesting that AI models required minimal computational resources during this period. Starting in the 2000s, the computational requirements begin to rise, particularly in the 2010s and 2020s, reflecting the rapid advancement and complexity of AI models. This increase could be attributed to the growing sophistication of machine learning algorithms, larger datasets, and more powerful hardware used in AI training. The graph illustrates the growing trend in AI computational power demands over the past few decades, highlighting how modern AI models have driven the need for significant computational resources.

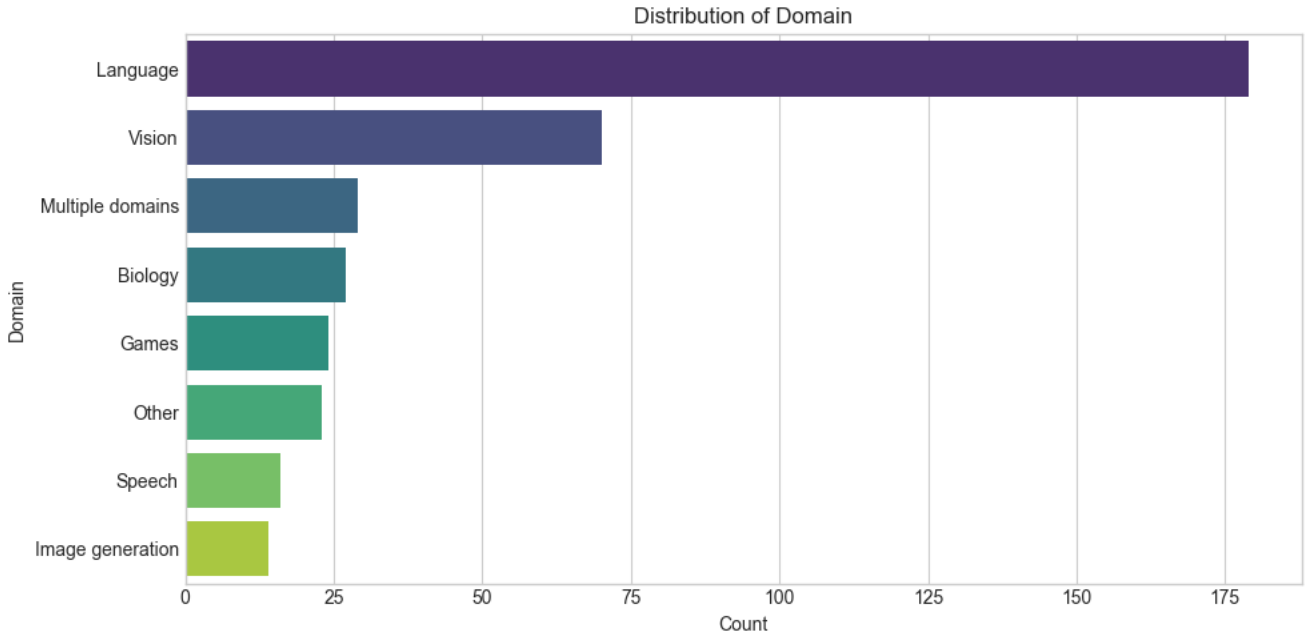


**Figure 4. Distribution of Domain**

Figure 4 provides a distribution of the AI model domains in the dataset. The Language domain overwhelmingly dominates the dataset, with over 175 samples, making it the most common domain for the AI models in the dataset. This is followed by Vision, which is also highly represented, indicating that these two domains are the most resource-

intensive. The Multiple domains category, which refers to models that span across multiple AI domains, has a moderate representation, reflecting the complexity of AI models that integrate multiple capabilities. The Biology, Games, and Other domains each have fewer entries, suggesting that while these fields do require computational resources, they are less prevalent compared to Language and Vision domains. The Speech and Image generation domains have the fewest models in the dataset, indicating that these are either emerging fields or less computationally demanding on average. This distribution suggests that Language and Vision domains contribute the most to the overall computational resource demands in the dataset.

## 4.3. Evaluation Metric Result

Two regression models, Random Forest Regressor and XGBoost Regressor, were trained to predict the computational requirements of AI models. Both models were successfully trained on the processed data. The Random Forest model was trained with default parameters, and upon completion, it was saved as 'random_forest_model.joblib'. Similarly, the XGBoost model was trained and saved as 'xgboost_model.joblib'. Both models underwent training on the same dataset, ensuring a direct comparison of their performance.

After training, the models were evaluated using several key performance metrics, both on the original petaFLOP scale and the log-transformed petaFLOP scale. The Random Forest model performed poorly on the original petaFLOP scale, yielding a Mean Absolute Error (MAE) of 204,856,527.24, a Mean Squared Error (MSE) of 1.27e+18, and a Root Mean Squared Error (RMSE) of 1.13e+9. Its $R^2$ value of 0.0345 indicated that the model explained very little of the variance in computational requirements. On the other hand, the XGBoost model performed even worse, with an MAE of 409,475,457.42, an MSE of 5.16e+18, and an RMSE of 2.27e+9. Its $R^2$ value was negative at -2.9377, suggesting that the model did not effectively model the relationship between the features and the target variable on the original scale.

However, when evaluated on the log-transformed petaFLOP scale, both models exhibited significant improvement in performance. The Random Forest model's performance dramatically improved, achieving an $R^2$ of 0.6904, which indicates that it was able to explain a substantial amount of the variance in computational requirements after the log transformation. It also had an MAE of 2.6531, an MSE of 12.8587, and an RMSE of 3.5859. Similarly, the XGBoost model also showed enhanced performance, with an $R^2$ of 0.5872, an MAE of 3.0211, an MSE of 17.1477, and an RMSE of 4.1410. While the XGBoost model performed better on the log scale compared to the original scale, it was still slightly less effective than the Random Forest model.

A plot was generated to compare the predicted values against the actual values on the log scale for both models. The Random Forest model predictions were relatively close to the actual values, as evidenced by the scatter plot, which showed the points clustering near the ideal prediction line. The XGBoost model showed a broader spread of predictions, indicating that it was less accurate in predicting the computational requirements compared to the Random Forest model. Nonetheless, both models demonstrated the potential to predict AI model computational requirements more accurately on the log-transformed scale than on the original scale.

 ISSN 2776-8007


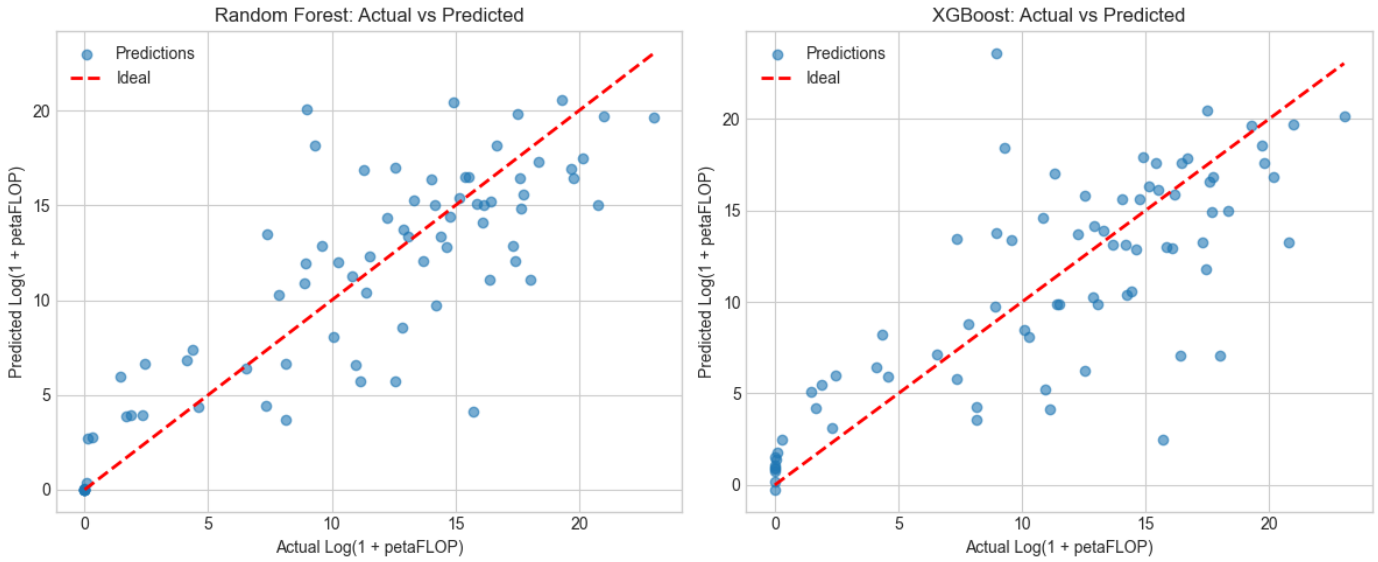


**Figure 5. Model Predictions vs Actual Values**

The first figure shows the predicted vs actual values for the Random Forest (left) and XGBoost (right) models. In both plots, the ideal line, represented by the red dashed line, marks the perfect prediction where the predicted values match the actual values exactly. The Random Forest model (left) shows a relatively good alignment of predicted and actual values, with points scattered along the ideal line. While the points are spread across the graph, they seem to follow a general upward trend, suggesting that the model performs reasonably well in predicting the log-transformed computational requirements (Log(1 + petaFLOP)). The XGBoost model (right) also shows an upward trend but with more variability in predictions compared to Random Forest. The scatter points indicate that XGBoost struggles more with the prediction for certain data points, particularly for high values of the target variable. This suggests that although XGBoost can capture some trends, its predictions are less consistent than those of the Random Forest model.

## 4.4. Analysis of Feature Importance

The Random Forest model analysis (Figure 6) revealed that the most important feature for predicting the log-transformed petaFLOP computational requirements was num__year, with an importance score of 0.840364. This indicates that the year in which the AI model was trained has a significant influence on the computational power required. The high importance of this feature suggests that AI models trained more recently may require more computational resources, possibly due to increasing model complexity or advancements in AI techniques over time. The second most important feature was cat_domain__domain_Language, with an importance score of 0.0537185. This shows that the domain of the AI model (in this case, language-related models) plays a role in determining computational requirements, though to a much lesser extent compared to the year. Other features like num__month (importance: 0.0352162) and num__dayofweek (importance: 0.026015) were of moderate significance, suggesting that the month or day of the week when the model was trained has a smaller but still notable effect on computational power. Finally, the cat_entity__entity feature, which represents the specific AI entity, had a relatively low importance score of 0.0234428, indicating that the entity itself is less influential in predicting computational power compared to the domain and temporal features.

 ISSN 2776-8007


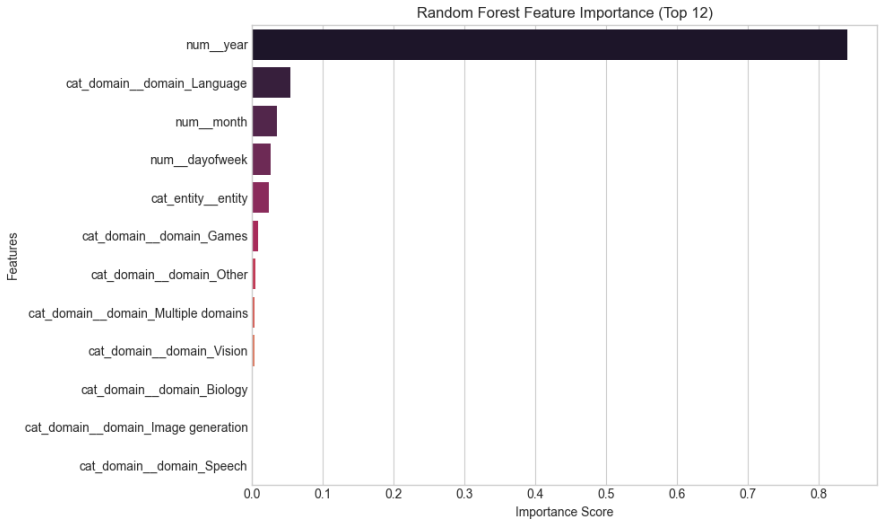


**Figure 6. Feature Importance Score from Random Forest**

The XGBoost model showed a different set of important features (Figure 7), with num__year again emerging as the most important feature, but with a much lower importance score of 0.321574 compared to the Random Forest model. This indicates that while year is still a significant predictor, it plays a relatively smaller role in the XGBoost model's predictions. The cat_domain__domain_Other feature came in second with an importance score of 0.112967, showing that models from domains categorized as Other contribute notably to computational requirements. This could reflect a variety of lesser-known AI domains, each requiring unique resources that are harder to generalize. Following closely were cat_domain__domain_Language (importance: 0.111258), cat_domain__domain_Games (importance: 0.105384), and cat_domain__domain_Biology (importance: 0.0547992). These results emphasize the varying computational needs of different AI domains, with language, gaming, and biology domains being particularly influential in the computational requirements. The XGBoost model appears to place more emphasis on domain-specific features, indicating that the nature of the AI application—such as whether it's related to language, games, or biology—has a significant impact on the computational resources needed.

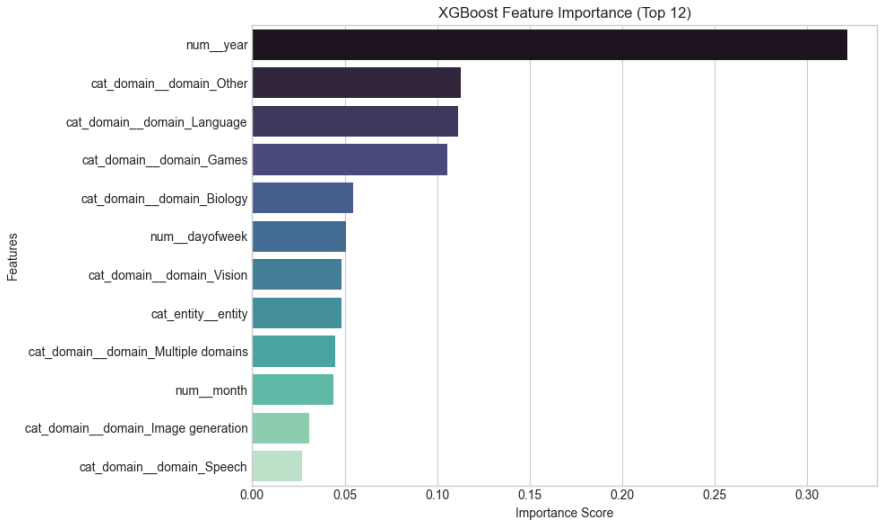


**Figure 7. Feature Importance Score from XGBoost**

These results emphasize the central role of year in determining computational requirements across both models, but they also highlight the domain of the AI model as an influential factor, especially in XGBoost. The Language domain is particularly impactful in both models, but the Other domain emerged as a crucial feature in XGBoost, suggesting that AI models in diverse or emerging domains may have different computational demands compared to more traditional AI domains like Language.

## 5. Conclusion

In this study, two machine learning models, Random Forest and XGBoost, were employed to predict the computational power required by AI models based on domain and entity characteristics. The results showed that both models were able to capture the general trends in computational needs, with Random Forest providing more consistent predictions compared to XGBoost. The feature importance analysis revealed that the year of training for AI models was the most influential factor in predicting computational power, followed by domain-related features, particularly Language. The XGBoost model, however, performed slightly worse than Random Forest, especially when predicting high computational requirements, indicating that Random Forest may be a better choice for this task given the current feature set. Despite the promising results, several limitations were identified. The data used for this study had sparsity issues, particularly in the Training computation (petaFLOP) column, which led to a significant reduction in the number of usable samples. Additionally, the models used in this study were relatively simple and might not have captured all the complexities of computational power requirements. Future research could benefit from incorporating additional features, such as hardware specifications, model architecture details, or more granular domain classifications. Moreover, exploring more advanced techniques, such as deep learning models, could potentially improve prediction accuracy, especially as the complexity and diversity of AI models continue to grow.

## 6. Declarations

### 6.1. Author Contributions

Conceptualization: A.A., R.N.W.; Methodology: R.N.W.; Software: A.A.; Validation: A.A., R.N.W.; Formal Analysis: A.A., R.N.W.; Investigation: A.A.; Resources: R.N.W.; Data Curation: R.N.W.; Writing—Original Draft Preparation: A.A., R.N.W.; Writing—Review and Editing: R.N.W., A.A.; Visualization: A.A. All authors have read and agreed to the published version of the manuscript.

### 6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

### 6.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

### 6.4. Institutional Review Board Statement

Not applicable.

### 6.5. Informed Consent Statement

Not applicable.

### 6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

[1] V. Ramamoorthi, “Exploring AI-Driven Cloud-Edge Orchestration for IoT Applications,” *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, 2023. doi: 10.32628/cseit239072.

[2] L. Shanmugam, S. Jangoan, and K. K. Sharma, “Dynamic Resource Allocation in Edge Computing for AI/ML Applications: Architectural Framework and Optimization Techniques,” *J. Knowl. Learn. Sci. Technol.*, vol. 2, no. 2, 2023. doi: 10.60087/jklst.vol2.n2.p397.

[3] Z. Chen, “Hardware Accelerated Optimization of Deep Learning Model on Artificial Intelligence Chip,” *Front. Comput. Intell. Syst.*, 2023, doi: 10.54097/fcis.v6i2.03.

[4] A. Pasdar, Y. C. Lee, T. Hassanzadeh, and K. Almi'ani, “Resource Recommender for Cloud-Edge Engineering,” *Inf.*, vol. 12, no. 6, p. 224, 2021. doi: 10.3390/info12060224.

[5] M. J. Karamthulla, J. Narkarunai, A. Malaiyappan, and R. Tillu, "Optimizing Resource Allocation in Cloud Infrastructure Through AI Automation: A Comparative Study," *J. Knowl. Learn. Sci. Technol. Issn 2959-6386 Online*, 2023, doi: 10.60087/jklst.vol2.n2.p326.

[6] M. Syed *et al.*, "Application of Machine Learning in Intensive Care Unit (ICU) Settings Using MIMIC Dataset: Systematic Review," *Informatics*, 2021, doi: 10.3390/informatics8010016.

[7] R. Antel, E. Sahlas, G. Gore, and P. Ingelmo, "Use of Artificial Intelligence in Paediatric Anaesthesia: A Systematic Review," *Bja Open*, 2023, doi: 10.1016/j.bjao.2023.100125.

[8] Et. al Ahmad Fawad, "Efficient Workload Allocation and Scheduling Strategies for AI-Intensive Tasks in Cloud Infrastructures.," *PST*, 2023, doi: 10.52783/pst.160.

[9] S. Di, S. Cheng, N. Cao, C. Gao, and L. Miao, "AI-based Geo-engineering Integration in Unconventional Oil and Gas," *Journal of King Saud University - Science*, vol. 33, p. 101542, 2021. doi: 10.1016/j.jksus.2021.101542.

[10] N. Justesen, T. Mahlmann, S. Risi, and J. Togelius, "Playing Multiaction Adversarial Games: Online Evolutionary Planning Versus Tree Search," *Ieee Trans. Games*, 2018, doi: 10.1109/tciaig.2017.2738156.

[11] I. M. Kuzyk and A. Kotelban, "The Use of Artificial Intelligence in Orthodontics," *Experimental and Clinical Medicine*, vol. 92, no. 4, 2023. doi: 10.35339/ekm.2023.92.4.kuk.

[12] J. Moon, H. Hwang, Y. Yu, M. Kim, R. E. Donatelli, and S. Lee, "How Much Deep Learning Is Enough for Automatic Identification to Be Reliable?," *Angle Orthod.*, 2020, doi: 10.2319/021920-116.1.

[13] J. Sanz and Y. Zhu, "Toward Scalable Artificial Intelligence in Finance," *in Proc. 2021 IEEE Int. Conf. on Services Computing (SCC)*, pp. 460–469, 2021. doi: 10.1109/SCC53864.2021.00067.

[14] C. Khandelwal, "A Survey of Multi-Cloud Deployment Strategies for AI Workloads," *Int. J. Sci. Res. Eng. Manag.*, vol. 9, 2021. doi: 10.55041/ijsrem8885.

[15] C. Ronquillo *et al.*, "Artificial Intelligence in Nursing: Priorities and Opportunities From an International Invitational Think-tank of the Nursing and Artificial Intelligence Leadership Collaborative," *J. Adv. Nurs.*, 2021, doi: 10.1111/jan.14855.

[16] C. P. Singh, "Leveraging Predictive Analytics and Artificial Intelligence for Optimal Healthcare Staffing: A Focus on Provider Resource Allocation," *Progress in Medical Sciences*, 2023. doi: 10.47363/pms/2023(7)e159.

[17] L. Rampini and F. R. Cecconi, "Artificial Intelligence Algorithms To predict Italian Real Estate Market Prices," *J. Prop. Invest. Finance*, 2021, doi: 10.1108/jpif-08-2021-0073.

[18] P. Nawrocki and P. Osypanka, "Resource Usage Cost Optimization in Cloud Computing Using Machine Learning," *IEEE Transactions on Cloud Computing*, vol. 10, pp. 2079–2089, 2022. doi: 10.1109/TCC.2020.3015769.

[19] T.-N. Kao *et al.*, "CT-Based Radiomic Analysis for Preoperative Prediction of Tumor Invasiveness in Lung Adenocarcinoma Presenting as Pure Ground-Glass Nodule," *Cancers*, 2022, doi: 10.3390/cancers14235888.

[20] Y. Xu, X. Li, D. Wu, Z. Zhang, and A. Jiang, "Machine Learning-Based Model for Prediction of Hemorrhage Transformation in Acute Ischemic Stroke After Alteplase," *Front. Neurol.*, 2022, doi: 10.3389/fneur.2022.897903.

[21] S. M. Baik, K. S. Hong, and D. J. Park, "Deep Learning Approach for Early Prediction of COVID-19 Mortality Using Chest X-Ray and Electronic Health Records," *BMC Bioinformatics*, 2023, doi: 10.1186/s12859-023-05321-0.

[22] M. Makki and M. Abdallah, "Testing the Significance of Artificial Intelligence Investment in Determining Stock Prices," in Artificial Intelligence and Big Data: Theories and Applications, Springer, 2019, pp. 43–52. doi: 10.1007/978-3-030-30874-2_4.

[23] B. Lee *et al.*, "An Artificial Intelligence Approach to Predict Gross Primary Productivity in the Forests of South Korea Using Satellite Remote Sensing Data," *Forests*, 2020, doi: 10.3390/f11091000.