

Machine Learning Strategies for Automated Performance Optimization

Stefano Coniglio

Associate Professor, University of Bergamo, Italy

Abstract

The increasing complexity of digital systems has created a growing need for automated approaches capable of continuously monitoring and improving system performance. Conventional performance optimization techniques generally depend on manually defined rules, periodic evaluations, and expert intervention, which can become inefficient in dynamic computational environments. Machine learning (ML) provides an advanced alternative by enabling systems to learn from historical and real-time performance data and automatically identify optimization opportunities.

This study examines machine learning strategies for automated performance optimization and proposes an integrated framework combining predictive analytics, adaptive learning, performance monitoring, automated decision-making, and continuous model evaluation. The study adopts a conceptual and descriptive research design based on secondary academic literature related to machine learning, automated optimization, predictive analytics, intelligent systems, and performance management. The proposed framework emphasizes data collection, preprocessing, feature engineering, model development, performance prediction, optimization strategy selection, automated implementation, and feedback-based adaptation.

The analysis indicates that machine learning can improve computational efficiency, resource utilization, system responsiveness, and operational reliability when integrated with appropriate monitoring mechanisms. However, challenges involving data quality, model drift, computational overhead, interpretability, security, and optimization stability remain important. The study concludes that automated performance optimization through machine learning can provide a scalable foundation for intelligent digital systems and advanced multidisciplinary applications.

Keywords: Machine Learning, Automated Optimization, Performance Optimization, Predictive Analytics, Intelligent Systems, Adaptive Learning, Resource Management, Artificial Intelligence.

1. Introduction

The rapid expansion of digital technologies has resulted in increasingly complex computational systems that must operate efficiently under changing workloads and environmental conditions. Cloud platforms, enterprise applications, intelligent devices, distributed infrastructures, and data-intensive services continuously generate performance-related information. Maintaining optimal performance in such environments requires continuous monitoring and timely intervention. Traditional optimization approaches often depend on predefined rules, manual configuration, and periodic performance

assessments. Although these methods can be effective in stable environments, they may become less suitable when workloads fluctuate rapidly or system conditions change unpredictably. Machine learning provides an alternative approach by allowing systems to learn from performance data and support automated optimization decisions.

Machine learning has become an important component of intelligent computing because it can identify complex relationships within large datasets and generate predictions without requiring every operational rule to be explicitly programmed. In performance management, ML algorithms can analyze variables such as response time, processing load, memory consumption, network traffic, resource utilization, and application behavior. These observations can be used to predict potential performance degradation and determine appropriate optimization actions. As a result, organizations can move from reactive performance management toward proactive and predictive optimization. This transition is particularly important in environments where delays, resource inefficiency, or system failures can significantly affect operational outcomes.

Automated performance optimization involves more than predicting whether a system will experience degradation. An effective optimization mechanism must also determine what action should be taken, implement that action, and evaluate whether the intervention produces the expected improvement. Machine learning can support this process through predictive models, reinforcement learning, anomaly detection, classification, regression, and ensemble-based approaches. Different algorithms can be selected according to the nature of the optimization problem. For example, regression models can predict continuous performance measures, while reinforcement learning can support sequential decisions in dynamic environments. The integration of these approaches creates opportunities for more adaptive performance-management systems.

The increasing use of cloud computing and distributed architectures has further strengthened the importance of automated optimization. Modern systems may dynamically allocate computational resources according to changing demand. Manual resource allocation can lead to over-provisioning, under-utilization, increased operational costs, or performance bottlenecks. Machine learning can analyze historical workload patterns and predict future resource requirements. Automated systems can then adjust computing resources according to predicted demand. Such mechanisms may improve both system performance and resource efficiency, particularly when optimization decisions must be made continuously.

However, the adoption of ML-based performance optimization introduces several technical and organizational challenges. Machine learning models depend heavily on the quality and representativeness of training data. Changes in workload patterns can also cause model performance to deteriorate over time, creating the problem of model drift. In addition, complex optimization systems may introduce computational overhead, making it necessary to balance the cost of prediction with the performance benefits obtained. Automated interventions must also be carefully controlled because an incorrect optimization decision can negatively affect system stability. Therefore, robust monitoring, validation, and feedback mechanisms are essential.

This study examines machine learning strategies for automated performance optimization and develops an integrated conceptual framework for intelligent performance management. The framework connects data collection, predictive modeling, optimization strategy selection, automated intervention,

and continuous feedback. The study also evaluates major benefits, challenges, and multidisciplinary applications of ML-based optimization. By integrating these dimensions, the research provides a foundation for future empirical investigations into intelligent systems capable of continuously improving their own operational performance.

2. Objectives of the Study

2.1 Primary Objective

The primary objective of this study is to examine how machine learning strategies can support automated performance optimization by enabling systems to predict performance conditions, identify optimization opportunities, automate resource-management decisions, and continuously adapt to changing operational environments.

2.2 Examination of Machine Learning-Based Optimization Strategies

The study examines the application of supervised learning, unsupervised learning, ensemble methods, reinforcement learning, and predictive analytics for automated performance optimization. It focuses on how different ML approaches can identify performance patterns and generate optimization decisions in dynamic computational environments.

2.3 Development of an Automated Optimization Framework

The study aims to develop a conceptual framework integrating performance monitoring, data preprocessing, feature extraction, machine learning prediction, optimization strategy selection, automated implementation, and feedback-based learning. The framework is designed to support continuous performance improvement.

2.4 Identification of Implementation Challenges

The study identifies major challenges associated with automated ML-based optimization, including data quality, model drift, computational overhead, explainability, security, optimization instability, and integration with existing infrastructures. It also examines mechanisms for improving reliability and sustainability.

3. Review of Literature

3.1 Machine Learning for Performance Management

Machine learning has increasingly been applied to performance-management problems because of its ability to identify patterns in complex operational data. Traditional performance-management systems frequently rely on threshold-based rules that trigger predefined responses when specific metrics exceed acceptable limits. Machine learning extends this approach by learning relationships among multiple variables and identifying conditions that may precede performance degradation.

Predictive models can estimate future resource requirements, response times, workload levels, or failure probabilities. This predictive capability enables organizations to address potential problems before they become critical. Consequently, ML-based performance management represents a shift from static monitoring toward adaptive and data-driven optimization.

3.2 Predictive Performance Optimization

Predictive optimization uses historical and real-time information to estimate future performance conditions and determine appropriate interventions. Regression, time-series models, neural networks, and ensemble learning techniques can be used to forecast performance indicators. Predictions may include expected workload, system latency, resource utilization, or probability of failure.

These predictions provide an analytical basis for automated resource allocation and preventive intervention. The effectiveness of predictive optimization depends on model accuracy and the stability of the underlying data distribution. Continuous validation is therefore required to ensure that predictions remain useful when operating conditions change.

3.3 Reinforcement Learning for Automated Optimization

Reinforcement learning offers a particularly relevant approach to automated optimization because it allows an agent to learn through interaction with an environment. Instead of simply predicting performance, a reinforcement-learning system evaluates possible actions and learns which actions generate favorable outcomes. In dynamic computing environments, an agent may learn when to allocate resources, modify configurations, or adjust system parameters.

The approach is valuable when optimization involves a sequence of decisions rather than a single intervention. However, reinforcement learning can require substantial training and careful reward design. Poorly defined objectives may cause the system to optimize one performance dimension while negatively affecting another.

3.4 Ensemble Learning and Optimization Accuracy

Ensemble learning combines predictions from multiple models to improve robustness and predictive accuracy. Techniques such as random forests, gradient boosting, and stacking can capture different aspects of complex performance data. In automated optimization, ensemble models may reduce the weaknesses associated with individual algorithms and provide more stable predictions.

This can be particularly useful when system performance depends on multiple interacting variables. Nevertheless, ensemble models may increase computational requirements and reduce interpretability. Therefore, model selection should consider the balance between predictive performance, computational efficiency, and transparency.

3.5 Adaptive Learning and Model Drift

Dynamic systems frequently experience changes in workload patterns, user behavior, infrastructure configuration, and operational conditions. These changes can cause machine learning models trained on historical data to become less accurate. This phenomenon is commonly associated with model or concept drift. Adaptive learning mechanisms can address this challenge by updating models when new information becomes available.

Continuous monitoring can identify declining predictive performance and trigger retraining procedures. However, uncontrolled adaptation may introduce instability or cause models to learn temporary anomalies. Effective automated optimization therefore requires carefully designed update policies and validation mechanisms.

4. Research Methodology

4.1 Research Design

The study adopts a conceptual and descriptive research design to examine the application of machine learning strategies in automated performance optimization. This approach is appropriate because the research focuses on integrating established concepts from machine learning, predictive analytics, performance engineering, and intelligent systems.

Rather than testing a single algorithm experimentally, the study synthesizes existing knowledge to construct a general optimization framework. The framework considers both technical and operational dimensions of automated performance management. It can subsequently be validated through experimental or empirical studies using real-world performance datasets.

4.2 Data Sources

The research is based on secondary academic literature concerning machine learning, automated optimization, predictive analytics, intelligent resource management, reinforcement learning, and performance engineering. Scholarly research and established technical literature are considered to identify major optimization strategies and their applications.

The reviewed literature provides information regarding algorithmic capabilities, performance-monitoring techniques, adaptive learning, and automated decision-making. Particular emphasis is placed on research addressing dynamic computational environments. The collected evidence forms the conceptual foundation for developing the proposed framework.

4.3 Data Analysis

A thematic and descriptive analytical approach is used to organize the reviewed literature into major research dimensions. The analysis identifies recurring relationships between machine learning capabilities and performance optimization outcomes. The major analytical categories include predictive performance modeling, anomaly detection, resource allocation, adaptive optimization, automated intervention, and continuous feedback.

These categories are compared to determine how different ML strategies contribute to automated performance management. The resulting themes are then integrated into a unified conceptual framework.

4.4 Framework Development

The proposed framework is developed by connecting the major stages of automated performance optimization into a continuous learning cycle. Performance data are first collected from relevant systems and subjected to preprocessing and feature extraction. Machine learning models then analyze the processed information to predict performance conditions or identify optimization opportunities.

An optimization engine selects appropriate actions based on model outputs and predefined objectives. The results of these actions are continuously monitored and returned to the learning system as feedback. This structure enables iterative improvement while maintaining performance oversight.

5. Proposed Machine Learning Framework for Automated Performance Optimization

5.1 Performance Data Collection

The first component of the proposed framework involves continuous collection of performance-related information from the target system. Relevant variables may include processor utilization, memory consumption, network throughput, response time, workload intensity, transaction volume, and application-level performance indicators.

The quality and granularity of collected data directly influence the ability of machine learning models to identify meaningful patterns. Data should therefore be collected consistently and securely across relevant operational components. Real-time monitoring can provide immediate information, while historical records can support predictive modeling. A comprehensive data-collection mechanism provides the foundation for reliable automated optimization.

5.2 Data Preprocessing and Feature Engineering

Raw performance data often contain missing values, measurement errors, outliers, redundant variables, and inconsistent observations. Preprocessing is therefore necessary before machine learning models are applied. Appropriate procedures may include data cleaning, normalization, missing-value treatment, outlier detection, and temporal aggregation.

Feature engineering can further identify variables that provide stronger predictive information about system performance. Carefully designed features may improve model accuracy while reducing unnecessary computational complexity. The preprocessing stage should also be automated where possible so that the optimization pipeline can operate continuously without excessive manual intervention.

5.3 Machine Learning Model Selection

Different optimization problems require different machine learning approaches. Supervised learning can be applied when historical performance outcomes are available, while unsupervised learning can identify previously unknown performance patterns or anomalies. Reinforcement learning can support sequential optimization decisions in dynamic environments.

Ensemble methods can improve predictive robustness by combining multiple models. Model selection should consider predictive accuracy, latency, computational requirements, interpretability, and adaptability. Rather than assuming that one algorithm is universally superior, the proposed framework recommends selecting models according to the specific characteristics and objectives of the target system.

5.4 Automated Optimization Engine

The optimization engine translates model predictions into operational actions. Depending on the application, these actions may involve resource allocation, workload scheduling, configuration adjustment, task prioritization, or capacity scaling. The engine should evaluate potential interventions according to multiple performance objectives rather than optimizing a single metric.

For example, improving processing speed should not create excessive resource consumption or reduce reliability. Decision policies can therefore incorporate weighted objectives related to speed, cost,

reliability, and resource efficiency. Automated actions should also operate within predefined safety constraints to prevent undesirable system behavior.

5.5 Continuous Feedback and Model Adaptation

Continuous feedback is essential because optimization outcomes provide information about whether an intervention was successful. After an action is implemented, the system should measure subsequent performance and compare the observed outcome with the predicted outcome. This information can be used to update model parameters, refine optimization policies, or identify changes in the operating environment.

Feedback mechanisms allow the system to improve gradually rather than depending entirely on its initial training. However, adaptation should be controlled through validation and performance thresholds. This ensures that continuous learning improves system behavior without introducing unstable or unnecessary changes.

6. Analysis

6.1 Automated Resource Optimization

Machine learning can improve resource optimization by predicting future demand and identifying patterns associated with inefficient resource allocation. Instead of allocating resources according to fixed thresholds, intelligent systems can estimate expected workloads and adjust capacity dynamically. Such optimization can reduce both under-provisioning and over-provisioning. Under-provisioning may result in poor responsiveness, while excessive resources can increase operational costs.

ML-based prediction provides an opportunity to balance these competing requirements. The effectiveness of automated resource optimization depends on prediction accuracy, system responsiveness, and the ability to execute resource changes without disrupting ongoing operations.

6.2 Predictive Detection of Performance Degradation

Predictive performance monitoring enables systems to identify early signals associated with degradation. Machine learning models can examine historical patterns in response times, resource utilization, error rates, and workload characteristics to estimate the likelihood of future performance problems. Early detection allows optimization actions to occur before a critical failure develops.

This proactive approach can improve system availability and reduce the operational impact of unexpected disruptions. However, false alarms may cause unnecessary interventions. Models must therefore be evaluated using appropriate performance criteria that balance detection sensitivity with operational stability.

6.3 Automated Workload Optimization

Workload optimization involves determining how computational tasks should be distributed and scheduled to achieve efficient system performance. Machine learning can analyze task characteristics, resource availability, historical execution patterns, and current system conditions. Predictive models can estimate task duration or resource requirements, while reinforcement learning can learn scheduling policies through repeated interaction. Automated workload optimization can reduce bottlenecks and

improve overall resource utilization. Nevertheless, optimization decisions must account for dependencies among tasks and changing system conditions. Continuous monitoring is therefore necessary to ensure that scheduling strategies remain effective.

6.4 Multi-Objective Performance Optimization

Real-world performance optimization rarely involves a single objective. Organizations may simultaneously seek to minimize latency, reduce resource costs, maximize throughput, improve reliability, and maintain energy efficiency. Machine learning can support multi-objective optimization by generating predictions across multiple performance dimensions and evaluating alternative actions. Optimization policies can assign appropriate priorities to competing objectives according to organizational requirements.

This approach provides greater flexibility than single-metric optimization. However, multi-objective optimization increases model and decision complexity, making transparent objective definitions and performance monitoring essential.

7. Discussion

7.1 Efficiency and Operational Responsiveness

Automated ML-based optimization can improve operational responsiveness by reducing the time between performance detection and corrective action. Conventional approaches may require administrators to interpret monitoring information and manually modify configurations. Machine learning can automate portions of this process and provide predictive insights before performance problems become severe.

Faster response can be especially valuable in large-scale digital systems where small inefficiencies can accumulate rapidly. Nevertheless, automation should not eliminate monitoring and human oversight entirely. High-impact optimization decisions should remain subject to appropriate safeguards and validation mechanisms.

7.2 Accuracy and Reliability

The reliability of automated optimization depends heavily on the accuracy of machine learning predictions. Accurate models can identify performance trends and recommend effective interventions, whereas inaccurate predictions may trigger unnecessary or harmful actions. Model evaluation should therefore consider both predictive accuracy and operational outcomes.

A model that performs well statistically may not necessarily generate the best optimization decisions in a real system. Performance evaluation should include practical indicators such as latency, resource efficiency, stability, and failure frequency. This broader evaluation perspective can improve the reliability of automated optimization frameworks.

7.3 Adaptability in Dynamic Environments

Dynamic computational environments require optimization strategies capable of responding to changing conditions. Fixed rules may become outdated when workloads, user behavior, or system configurations change. Machine learning can improve adaptability by learning from newly observed data and updating

predictions. Adaptive mechanisms are particularly valuable when system behavior cannot be accurately represented through static assumptions.

However, continuous adaptation introduces the risk of learning from temporary or abnormal conditions. Appropriate validation, retraining policies, and performance thresholds are therefore required to maintain stability.

7.4 Human Oversight and Explainability

Although automated optimization can reduce manual workload, human oversight remains important when decisions have significant operational consequences. Complex machine learning models may produce recommendations that are difficult to interpret, creating challenges for system administrators and decision-makers.

Explainability mechanisms can help users understand why particular optimization decisions were generated. Human oversight can also identify situations in which model recommendations conflict with broader operational objectives. A balanced approach should therefore combine automated decision-making with transparent monitoring and clearly defined intervention controls.

8. Practical Applications

Machine learning-based automated performance optimization can be applied across numerous technological and organizational domains. In cloud computing, ML models can predict workload demand and support dynamic resource allocation. In manufacturing, predictive models can optimize machine utilization and maintenance scheduling. In telecommunications, intelligent systems can analyze network conditions and optimize capacity distribution.

In financial technology, ML can support transaction-processing efficiency and automated workload management. In software engineering, intelligent systems can analyze application performance and identify optimization opportunities. These applications demonstrate that automated ML optimization can provide value across multidisciplinary environments where performance conditions are dynamic and large quantities of operational data are continuously generated.

9. Challenges and Limitations

9.1 Data Quality and Availability

Machine learning models require reliable and representative performance data. Incomplete, inconsistent, noisy, or biased data can reduce model accuracy and lead to ineffective optimization decisions. Data collection can also become difficult when systems contain heterogeneous platforms or legacy components.

Automated pipelines should therefore include mechanisms for data validation and quality assessment. Historical data may also fail to represent future operating conditions, creating additional prediction challenges. Consequently, organizations should continuously evaluate whether available data remain appropriate for the optimization task.

9.2 Model Drift and Adaptation

Performance environments can change because of software updates, infrastructure modifications, workload changes, user behavior, or external events. Such changes can reduce the predictive accuracy of models trained under previous conditions. Continuous monitoring is necessary to detect declining performance and identify potential model drift.

Retraining strategies can restore model accuracy, but excessive retraining may increase computational costs and operational complexity. Automated adaptation should therefore be governed by measurable criteria. Carefully designed update mechanisms can balance responsiveness with model stability.

9.3 Computational Overhead

Machine learning itself requires computational resources, particularly when complex models are used for real-time optimization. If model execution consumes excessive resources, the optimization system may partially offset the performance benefits it is intended to create. This problem is particularly relevant in latency-sensitive environments.

Efficient model architectures, feature selection, edge processing, and model compression can help reduce computational overhead. Organizations should evaluate the total cost of the optimization pipeline rather than considering only the performance improvements produced by the model.

9.4 Security and Operational Risk

Automated optimization systems may create security and operational risks if unauthorized users manipulate performance data or optimization policies. Incorrect or malicious inputs could cause the system to make inappropriate decisions. Security mechanisms should therefore protect data collection, model execution, and automated intervention processes.

Access controls, authentication, monitoring, and audit mechanisms can reduce these risks. Safety constraints should also prevent automated systems from making actions that could destabilize critical infrastructure. Security should consequently be considered an integral part of intelligent optimization design.

10. Future Research

Future research should empirically compare different machine learning strategies for automated performance optimization across real-world environments. Comparative experiments could evaluate supervised learning, ensemble methods, reinforcement learning, deep learning, and hybrid approaches under different workload conditions.

Researchers should examine not only prediction accuracy but also actual improvements in latency, resource utilization, cost efficiency, reliability, and system stability. Longitudinal studies would be particularly valuable for determining whether adaptive ML systems maintain performance advantages over extended operational periods. Such research could provide stronger evidence regarding the conditions under which particular optimization strategies are most effective.

Another important direction involves developing explainable, energy-efficient, and self-adaptive optimization systems. Future research should investigate how explainability can be integrated into

automated optimization without significantly increasing computational overhead. Researchers should also examine the relationship between optimization performance and energy consumption to support sustainable computing.

Hybrid approaches combining predictive modeling with reinforcement learning may provide opportunities for more flexible optimization. Federated learning and privacy-preserving approaches could further support optimization across distributed environments without requiring centralized access to sensitive operational data.

11. Conclusion

Machine learning provides a powerful foundation for developing automated performance optimization systems capable of learning from operational data and adapting to changing computational conditions. By integrating predictive modeling, anomaly detection, intelligent resource allocation, automated decision-making, and continuous feedback, ML-based systems can move performance management from reactive intervention toward proactive optimization. The proposed framework emphasizes a continuous cycle in which data are collected, processed, analyzed, transformed into optimization decisions, and evaluated through subsequent system feedback.

The study also demonstrates that successful automated optimization requires more than selecting a highly accurate machine learning algorithm. Data quality, model stability, computational efficiency, system security, explainability, and operational constraints must be considered throughout the optimization lifecycle. Automated actions should operate within clearly defined boundaries and remain subject to appropriate monitoring and human oversight. These requirements become increasingly important as optimization systems are deployed in complex and mission-critical environments.

Overall, machine learning strategies for automated performance optimization represent an important area of advanced research and development. The integration of intelligent prediction with adaptive decision-making can support efficient resource utilization, improved responsiveness, and greater operational reliability across multiple domains. Future research should focus on empirical validation, hybrid learning architectures, explainable optimization, energy efficiency, and long-term adaptive performance. Such developments can contribute to the creation of intelligent systems capable of continuously improving their performance while maintaining stability, transparency, and responsible operation.

References

1. Amari, S.-I. (1998). Natural gradient works efficiently in learning. *Neural Computation*, 10(2), 251–276. <https://doi.org/10.1162/089976698300017746>
2. Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, 24 (pp. 2546–2554).
3. Bottou, L., Curtis, F. E., & Nocedal, J. (2018). Optimization methods for large-scale machine learning. *SIAM Review*, 60(2), 223–311. <https://doi.org/10.1137/16M1080173>
4. Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>

5. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). ACM. <https://doi.org/10.1145/2939672.2939785>
6. Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12, 2121–2159.
7. Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), 1189–1232. <https://doi.org/10.1214/aos/1013203451>
8. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
9. Hutter, F., Hoos, H. H., Leyton-Brown, K., & Murphy, K. P. (2011). Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization Conference* (pp. 507–523). Springer.
10. Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245), 255–260. <https://doi.org/10.1126/science.aaa8415>
11. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *International Conference on Learning Representations*. <https://arxiv.org/abs/1412.6980>
12. Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F., & Leyton-Brown, K. (2017). Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *Journal of Machine Learning Research*, 18(25), 1–5.
13. Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2018). Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185), 1–52.
14. Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D. B., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M. J., Zadeh, R., Zaharia, M., & Talwalkar, A. (2016). MLlib: Machine learning in Apache Spark. *Journal of Machine Learning Research*, 17(34), 1–7.
15. Sharp, M., Ak, R., & Hedberg, T. D., Jr. (2018). A survey of the advancing use and development of machine learning in smart manufacturing. *Journal of Manufacturing Systems*, 48, 170–179. <https://doi.org/10.1016/j.jmsy.2018.02.004>
16. Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 25 (pp. 2951–2959).
17. Tao, F., Qi, Q., Liu, A., & Kusiak, A. (2018). Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48, 157–169. <https://doi.org/10.1016/j.jmsy.2018.01.006>
18. Wang, J., Ma, Y., Zhang, L., Gao, R. X., & Wu, D. (2018). Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, 48, 144–156. <https://doi.org/10.1016/j.jmsy.2018.01.003>
19. Zhang, Q., Yang, L. T., Chen, Z., & Li, P. (2018). A survey on deep learning for big data. *Information Fusion*, 42, 146–157. <https://doi.org/10.1016/j.inffus.2017.10.006>
20. Zheng, P., Wang, H., Sang, Z., Zhong, R. Y., Liu, Y., Liu, C., Mubarak, K., Yu, S., & Xu, X. (2018). Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives. *Frontiers of Mechanical Engineering*, 13, 137–150. <https://doi.org/10.1007/s11465-018-0499-5>