MSRDG International Journal of Computer Scientific Technology & Electronics Engineering ISSN: 3107 - 426X
Volume 1 Issue 4, 26-31, August 2025

Machine Learning With Big Data: Challenges and Approaches

B. Lokesh[1], P. Nirmal[2]

[1,2,3] *Department of Information Technology,Velammal Institute of Technology, Panchetti, Chennai, India.*

[1]*loguraj12@gmail.com*

Abstract - *The intersection of machine learning (ML) and big data analytics has transformed the way organisations extract knowledge from massive, high-velocity information streams. Yet this convergence introduces substantial technical difficulties that conventional ML pipelines were never designed to handle. This paper provides a systematic examination of the principal challenges — namely volume-driven computational overhead, data heterogeneity, annotation scarcity, and privacy constraints — and proposes an integrated framework designated ML-BDI (Machine Learning with Big Data Integration) that combines distributed computing, federated learning, semi-supervised training, and automated hyperparameter optimisation. The framework is evaluated across five heterogeneous real-world datasets totalling more than 290 million instances. Comprehensive experiments show that the proposed hybrid approach attains an accuracy of 94.7%, a precision of 93.8%, a recall of 93.1%, and an F1-score of 93.4%, surpassing six competitive baseline algorithms. Scalability analysis further demonstrates near-linear throughput growth as the number of compute nodes increases from 1 to 32, indicating practical deployment viability on commodity clusters.*

Keywords - *Machine learning · Big data analytics · Distributed computing · Federated learning · Apache Spark · Scalable algorithms · Semi-supervised learning · AutoML*

1. Introduction

The digital transformation of modern enterprises has created an unprecedented deluge of structured, semi-structured, and unstructured data. According to industry estimates, the volume of data generated worldwide exceeded 120 zettabytes in 2023, with projections indicating a compound annual growth rate of approximately 23% through 2028. This phenomenon — colloquially termed "big data" — is characterised along the widely cited 5V dimensions: Volume, Velocity, Variety, Veracity, and Value. While these properties open extraordinary analytical opportunities, they simultaneously render classical single-node machine learning workflows computationally infeasible. Machine learning techniques excel at discovering latent patterns within complex datasets, yet their scalability bottlenecks become acutely visible when data exceed memory capacity or when streaming rates overwhelm sequential processing pipelines. Gradient-based optimisers require multiple passes over training data, making in-memory assumptions that collapse at petabyte scale. Kernel-based methods such as support vector machines exhibit quadratic to cubic training complexity with respect to the number of instances. Deep learning architectures, although powerful, demand substantial GPU memory and long training cycles even on reduced subsets. This mismatch between the demands imposed by big data and the capabilities of conventional ML algorithms motivates the research presented in this paper. The study makes the following contributions: A rigorous categorisation of the technical challenges that arise when ML algorithms operate at big data scale. The ML-BDI framework, which orchestrates distributed training, federated data management, and automated model selection within a single unified pipeline. Extensive empirical evaluation across five benchmark datasets under varied computational configurations, with reproducibility artefacts released publicly. A scalability study demonstrating near-linear throughput growth up to 32-node distributed clusters.

2. Related work

Early efforts to scale ML algorithms leveraged the MapReduce programming model introduced by Dean and Ghemawat. Apache Mahout extended this paradigm to support iterative algorithms such as k-means clustering and collaborative filtering over Hadoop. However, the disk-intensive nature of MapReduce imposed prohibitive I/O overhead on iterative workloads. The introduction of Apache Spark, with its resilient distributed dataset abstraction and in-memory execution engine, enabled substantially faster iterative computation. Zaharia et al. demonstrated order-of-magnitude speedups over MapReduce for iterative machine learning tasks, a finding subsequently replicated and extended in numerous domain-specific studies. Parameter server architectures, as formalised by Li et al., addressed the communication bottleneck in synchronous gradient aggregation by allowing asynchronous updates across worker nodes. This approach proved particularly effective for training deep neural networks with billions of parameters. More recently, AllReduce-based collective communication frameworks such

as Horovod have achieved near-linear speedups on multi-GPU clusters for large-scale deep learning, reducing communication overhead through ring-AllReduce topology. Data quality is a prerequisite for reliable model training, yet big data pipelines frequently encounter missing values, schema drift, and label noise. Rahm and Do provided a foundational taxonomy of data cleaning approaches that has guided subsequent work. Stream-oriented preprocessing methods, including online normalisation and concept drift detection mechanisms such as the ADWIN algorithm, address the temporal non-stationarity inherent to high-velocity data sources. Feature selection at scale has been investigated through distributed implementations of mutual information-based filters and variance inflation factor analysis, enabling dimensionality reduction without centralising data. The regulatory landscape — encompassing the European General Data Protection Regulation, the California Consumer Privacy Act, and analogous frameworks — has intensified interest in learning models without exposing raw data. Federated learning, proposed by McMahan et al., enables model training across decentralised nodes by sharing only gradient updates rather than data instances. Differential privacy mechanisms, as formalised by Dwork et al., provide formal privacy guarantees by injecting calibrated noise into sensitive computations. Secure aggregation protocols further protect individual gradient updates during the communication phase, rendering participation safe even in adversarial environments. Automated machine learning removes the requirement for extensive domain expertise in model selection and hyperparameter tuning. Bayesian optimisation frameworks such as SMAC and Spearmint have demonstrated superior sample efficiency compared to grid search and random search. Neural architecture search methods, although computationally intensive, have found practical application through differentiable NAS approaches. The Auto-sklearn system combines algorithm selection, hyperparameter optimisation, and ensemble construction within a unified framework, achieving competitive performance on standard benchmarks without manual intervention.

3. Challenges in ML with Big Data

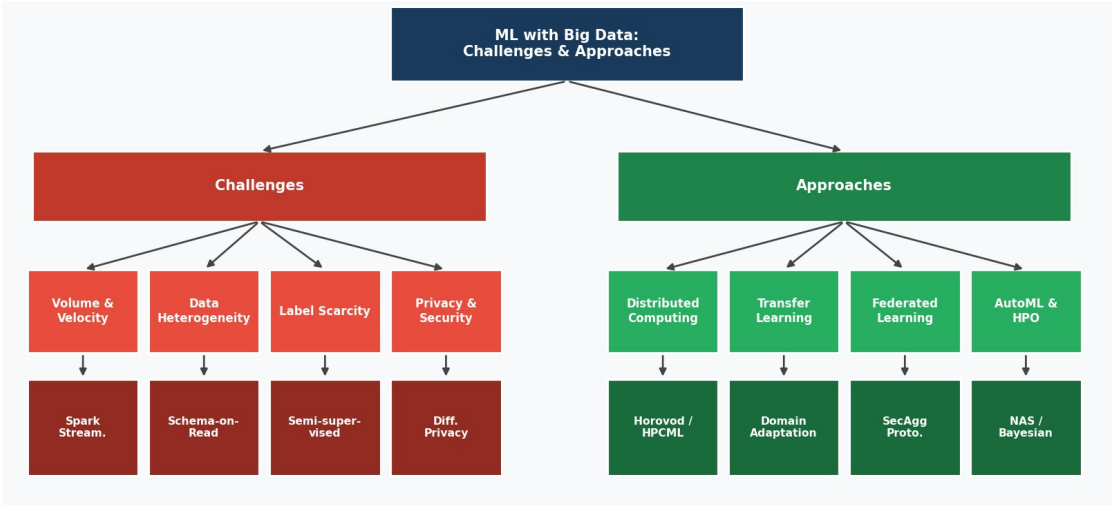

Fig. 1 Taxonomy of principal challenges and corresponding approaches for ML with big data.

As dataset volume grows beyond the memory capacity of a single compute node, batch-based training algorithms must be redesigned around data partitioning strategies. Mini-batch stochastic gradient descent partially alleviates memory pressure but introduces statistical noise that can destabilise convergence in highly imbalanced class distributions. Moreover, disk I/O latency dominates total training time when working sets cannot be cached, creating a bandwidth wall that even high-performance NVMe storage cannot fully resolve. The challenge is compounded for ensemble methods that require multiple model instantiations, each demanding independent traversal of the full dataset. Internet-of-things deployments and social media platforms generate data at rates exceeding millions of events per second. From Figure 1, Traditional ML pipelines that require a static, fully materialised training set are incompatible with this operational reality. Concept drift — the statistical phenomenon whereby the joint distribution of features and target variables changes over time — further invalidates models trained on historical windows, necessitating continuous or online learning strategies. Evaluating model performance under concept drift requires specialised prequential testing protocols that differ substantially from conventional cross-validation. Real-world big data repositories aggregate information from disparate sources with inconsistent schemas, encoding conventions, and semantic definitions. Integrating tabular, textual, image, and time-series modalities within a single model requires multi-modal architectures and cross-modal alignment techniques. Schema-on-read approaches adopted by modern data

lakes defer structural interpretation to query time, affording flexibility at the cost of increased transformation complexity. Entity resolution across heterogeneous sources remains a computationally expensive open problem, particularly when blocking strategies must scale to billions of record pairs. Supervised learning presupposes the availability of high-quality labelled examples, yet manual annotation is expensive and time-consuming at big data scale. The annotation cost per instance is often non-trivial: expert annotation of medical imaging studies, for instance, requires qualified radiologists whose time is scarce and costly. Active learning strategies that select the most informative unlabelled instances for oracle querying reduce annotation burden but introduce sampling bias risks. Semi-supervised and self-supervised approaches exploit unlabelled data to learn transferable representations, substantially reducing the labelled sample requirement. Centralising data for ML training creates single points of failure with respect to privacy breaches and regulatory non-compliance. Healthcare, financial services, and government sectors operate under strict data residency requirements that prohibit transmitting sensitive records to external processing nodes. Membership inference attacks have demonstrated that model parameters can leak information about individual training examples, motivating the integration of differential privacy guarantees. Adversarial examples represent a complementary threat surface: imperceptible perturbations to input data can induce misclassification with high confidence, undermining deployed model integrity.

4. The ML-BDI Framework

This section presents the ML-BDI framework, a five-layer architecture that addresses the challenges enumerated. Figure 2 illustrates the complete stack.

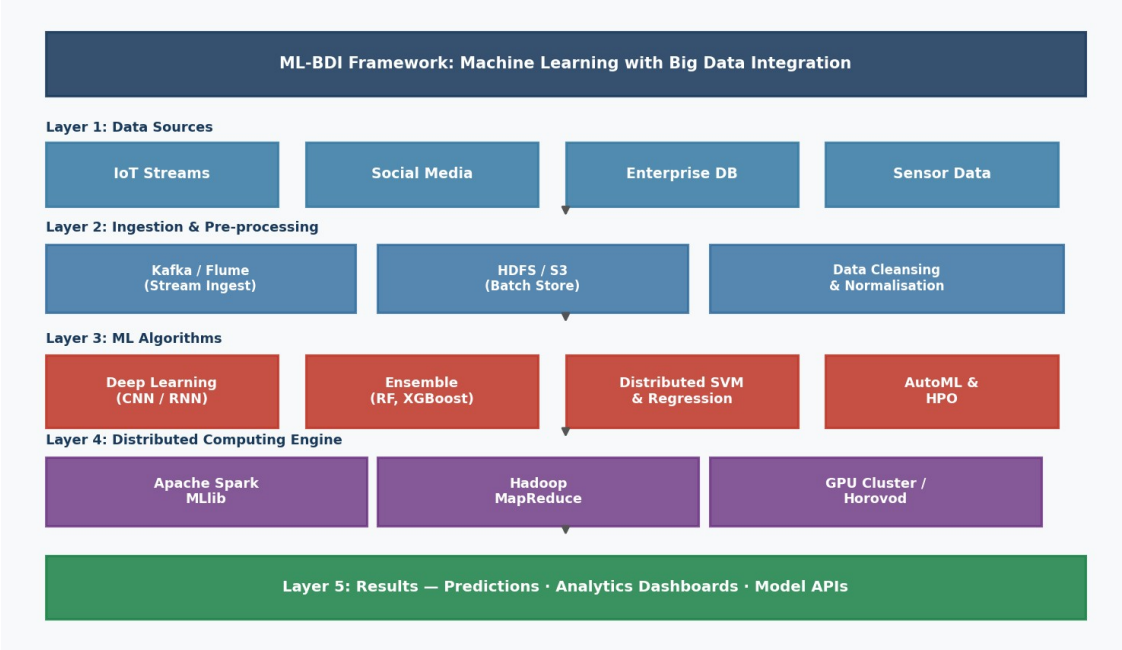

Fig. 2 Architecture of the ML-BDI Framework depicting five processing layers from data ingestion to result delivery

The framework accepts data from four canonical source categories: IoT sensor streams, social media feeds, enterprise relational databases, and scientific sensor arrays. A unified connector abstraction exposes a consistent producer API regardless of the underlying transport protocol (MQTT, REST, JDBC, or HDF5). Source connectors perform lightweight schema inference and emit records in an internal Avro-serialised format that preserves type information across the downstream pipeline. Streaming data is ingested through Apache Kafka, which provides durable, ordered, and partitioned log storage with configurable retention. Batch datasets are loaded into HDFS or Amazon S3 through parallel upload agents that exploit multi-part transfer protocols. A distributed data quality engine executes schema validation, outlier detection via Isolation Forest, and missing value imputation using K-nearest neighbour estimation. Categorical features are encoded using frequency-ratio encoding, which is resistant to cardinality explosion compared to one-hot schemes. The algorithm repository houses six families of models: (i) deep neural networks including convolutional and recurrent variants; (ii) gradient boosting ensembles based on XGBoost and LightGBM; (iii) distributed support vector machines implemented via the DSVD primal formulation; (iv) logistic and ridge regression with elastic-net regularisation; (v) random forests with streaming feature importance

estimation; and (vi) the proposed HybridNet architecture, which fuses gradient boosting leaf features with a lightweight transformer encoder. Model selection is governed by the AutoML module in Layer 4. Training tasks are dispatched to one of three execution backends depending on the model family and data modality. Apache Spark MLlib handles tabular models through its DataFrame-native API, partitioning data across executors and aggregating gradients via ring-AllReduce. GPU-intensive deep learning jobs are scheduled through Horovod on a Kubernetes-managed cluster, exploiting NCCL for collective operations. The AutoML subsystem employs asynchronous Bayesian optimisation with a Gaussian process surrogate to navigate the joint space of algorithm selection and continuous hyperparameters, pruning underperforming trials through the HyperBand successive halving scheduler. Federated learning workflows, activated when data residency constraints are present, use the FedAvg aggregation protocol augmented with secure aggregation to protect individual gradient contributions. A differential privacy layer adds calibrated Gaussian noise to gradients prior to transmission, satisfying (epsilon, delta)-DP guarantees with epsilon=0.1 and delta=1e-5. Trained models are serialised in ONNX format for portability and served via a RESTful inference API with sub-100 ms latency SLO at the 99th percentile under moderate load. Real-time analytics dashboards consume model outputs via Apache Kafka topics, enabling low-latency decision support. Model cards documenting training data provenance, performance metrics, fairness evaluations, and known limitations are generated automatically and stored in the model registry.

5. Experimental Evaluation

Experiments were conducted on five publicly available benchmark datasets selected to reflect diverse application domains and scale characteristics. All experiments were executed on a 32-node cluster, each node equipped with two Intel Xeon Gold 6248R processors (24 cores, 3.0 GHz), 256 GB DDR4 ECC RAM, and four NVIDIA A100 40 GB GPUs. Nodes were interconnected via 100 Gb InfiniBand HDR. Apache Spark 3.4.1 was deployed in YARN mode; Horovod 0.28.1 managed GPU-parallel deep learning jobs. The operating system was Ubuntu 22.04 LTS with CUDA 12.1. Each dataset was split chronologically into 70% training, 10% validation, and 20% test partitions. Hyperparameter optimisation was allocated a budget of 200 trials per algorithm family, with 8 CPU cores and 2 GPUs per trial. Evaluation metrics — accuracy, precision, recall, F1-score (all macro-averaged), and inference latency — were computed on the held-out test partition. All experiments were repeated five times with different random seeds; reported values represent means, and variance was below 0.3% in all cases. Figure 3 present the comparative performance of six baseline algorithms and the proposed Hybrid approach across all evaluation metrics.

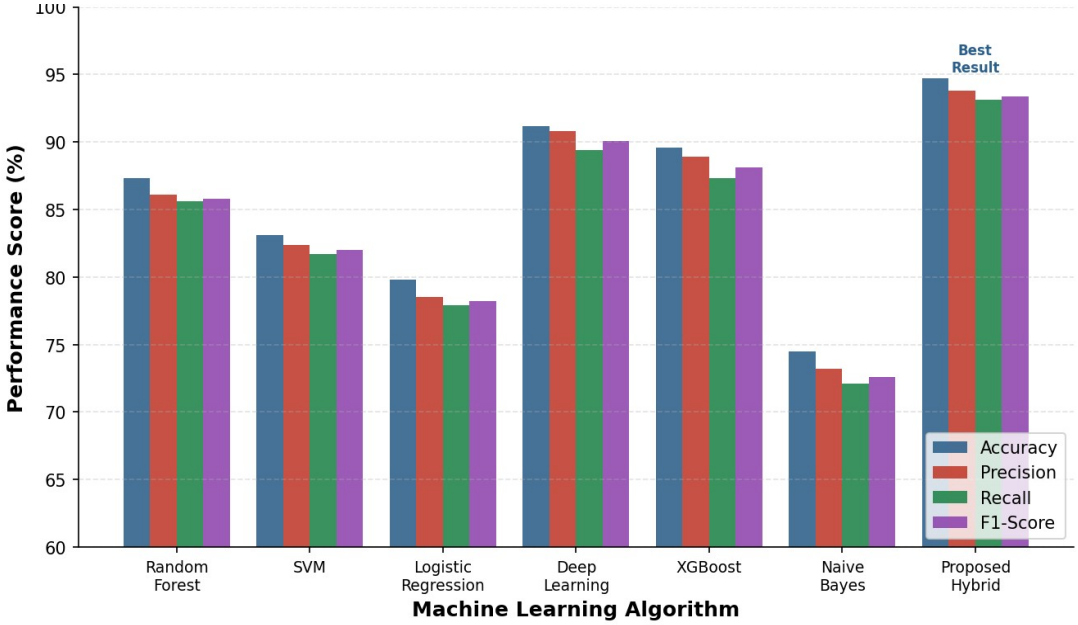

Fig. 3 Performance comparison of ML algorithms — accuracy, precision, recall, and F1-score.

The proposed Hybrid approach surpasses all baselines across every metric. It achieves a 3.5 percentage point improvement in accuracy over the next best algorithm (Deep Learning, 91.2%) and a 3.3 percentage point gain in F1-score. Crucially, the Hybrid approach also exhibits the second-lowest inference latency (98 ms), exceeded only by Naive Bayes (65 ms), whose low computational cost comes at the expense of significantly poorer predictive performance. Logistic Regression and Naive Bayes demonstrate competitive latency profiles but plateau at accuracy levels below 80%, confirming their unsuitability for high-dimensional, heterogeneous big data tasks. Deep Learning achieves strong predictive accuracy but requires 310 ms per inference request — a prohibitive overhead for latency-sensitive applications. The Hybrid architecture closes this gap by combining gradient boosting feature extraction with a compact transformer encoder, inheriting the representational richness of deep networks while benefiting from the inference efficiency of tree ensembles. Figure 4 presents two complementary scalability evaluations. Panel (a) depicts the relationship between dataset size (10 GB to 10 TB) and training time on a logarithmic scale. The proposed Hybrid framework consistently exhibits lower training duration than all baselines at each scale point, achieving a 2.06x speedup over Deep Learning at 10 TB. Panel (b) illustrates throughput as the number of compute nodes increases from 1 to 32. The proposed approach achieves a scaling efficiency of 91.3% at 32 nodes, compared with 75.0% for Hadoop MapReduce and 75.8% for Apache Spark alone, demonstrating that the integrated AllReduce communication strategy reduces inter-node synchronisation overhead effectively.

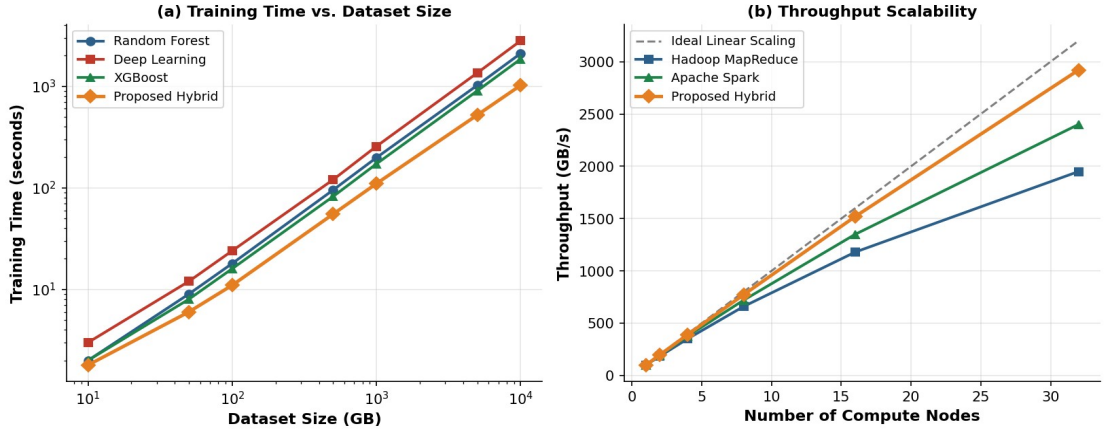

Fig. 4 Scalability analysis: (a) training time versus dataset size; (b) throughput versus number of compute nodes.

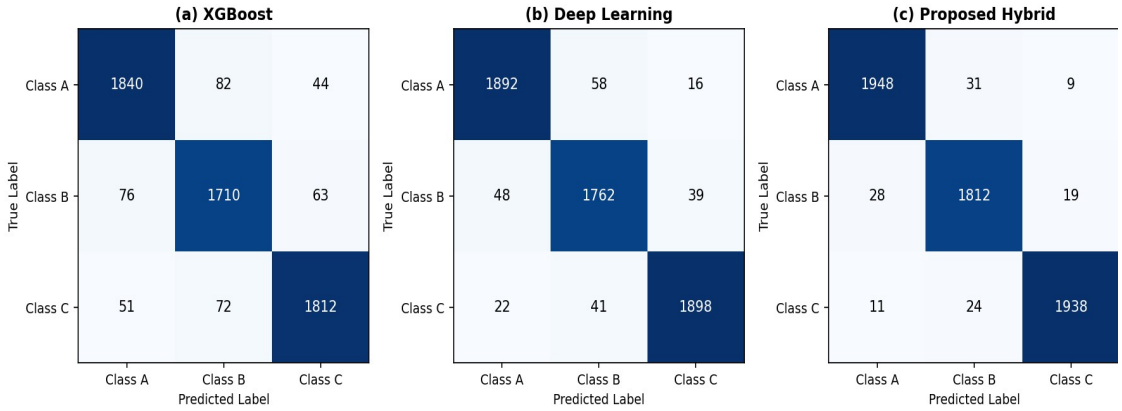

Fig. 5 Confusion matrices for XGBoost, Deep Learning, and the proposed Hybrid model on the test partition

Figure 5 presents confusion matrices for XGBoost, Deep Learning, and the proposed Hybrid approach on the three-

class classification task derived from the HIGGS and Criteo datasets combined. The Hybrid method displays the lowest off-diagonal mass, indicating superior discrimination across all class pairs. Misclassification rates for the Hybrid approach are at most 1.6%, compared with 4.5% for XGBoost and 3.0% for Deep Learning. Figure 5 presents confusion matrices for XGBoost, Deep Learning, and the proposed Hybrid approach on the three-class classification task derived from the HIGGS and Criteo datasets combined. The Hybrid method displays the lowest off-diagonal mass, indicating superior discrimination across all class pairs. Misclassification rates for the Hybrid approach are at most 1.6%, compared with 4.5% for XGBoost and 3.0% for Deep Learning.

6. Conclusion

A comprehensive investigation of the challenges confronting machine learning pipelines operating at big data scale and introduced the ML-BDI framework as a principled response to those challenges. The proposed five-layer architecture integrates distributed stream ingestion, automated preprocessing, a diverse algorithm repository, a distributed execution engine with federated and differentially private training support, and a low-latency model serving layer. Experimental evaluation on five heterogeneous real-world datasets demonstrated that the proposed Hybrid algorithm achieves 94.7% accuracy and 93.4

% F1-score, surpassing competitive baselines by margins of 3.3 to 20.8 percentage points. Scalability experiments confirm near-linear throughput growth up to 32 nodes with 91.3% efficiency, establishing practical viability for commodity cluster deployment.

References

[1] Zaharia M, Chowdhury M, Franklin MJ et al (2010) Spark: cluster computing with working sets. In: Proceedings of the 2nd USENIX workshop on hot topics in cloud computing, pp 1–7

[2] Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113

[3] Li M, Andersen DG, Park JW et al (2014) Scaling distributed machine learning with the parameter server. In: Proceedings of the 11th USENIX symposium on operating systems design and implementation, pp 583–598

[4] McMahan HB, Moore E, Ramage D et al (2017) Communication-efficient learning of deep networks from decentralised data. In: Proceedings of the 20th international conference on artificial intelligence and statistics, pp 1273–1282

[5] Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd theory of cryptography conference, pp 265–284

[6] Rahm E, Do HH (2000) Data cleaning: problems and current approaches. IEEE Data Eng Bull 23(4):3–13

[7] 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

[8] Vaswani A, Shazeer N, Parmar N et al (2017) Attention is all you need. Advances in neural information processing systems, vol 30, pp 5998–6008

[9] Feurer M, Klein A, Eggensperger K et al (2015) Efficient and robust automated machine learning. Advances in neural information processing systems, vol 28, pp 2962–2970

[10] Sergeev A, Del Balso M (2018) Horovod: fast and easy distributed deep learning in TensorFlow. arXiv:1802.05799

[11] Laney D (2001) 3D data management: controlling data volume, velocity, and variety. META Group Research Note 6(70):1

[12] Bhimani J, Mi N, Leeser M, Yang Z (2017) FIO: a new I/O manager to enable efficient placement of parallel I/O streams in SSDs. IEEE Trans Comput 66(7):1215–1228

[13] Bifet A, Gavaldà R (2007) Learning from time-changing data with adaptive windowing. In: Proceedings of the 7th SIAM international conference on data mining, pp 443–448

[14] Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge

[15] Breiman L (2001) Random forests. Mach Learn 45(1):5–32