Azure-SQL AutoSizer: Privacy-Aware Performance-Cost SKU Mapping for SQL Migrations

Harika Naidu Beesabathuni

Project Manager, MSR Technology Group LLC
Chatham, United States of America
harikanaidu.b@gmail.com

Abstract. This paper describes the design of Azure-SQL AutoSizer, a SKU recommendation engine for automatically selecting Azure SQL PaaS targets (i.e. Azure SQL Database and Azure SQL Managed Instance) to which on-premises SQL workloads can be migrated. Unlike existing tools that require intrusive access to customer data or queries, AutoSizer works only on low-level performance counters such as CPU, memory, IOPS and latency and is GDPR compliant. We employ a price-performance throttling model to create personalised SKU ranking and integrate Azure customer telemetry that profiles workload negotiability on various resource dimensions. Since October 2021, AutoSizer has been implemented in Azure Data Migration Assistant (DMA) and has shown high accuracy – with 89.4% of SQL DB and 96.7% of SQL MI expert-defined SKUs – while flagging significant cost-saving opportunities for over-provisioned Azure SQL customers. The system can assist with hundreds of migration assessments a day, maintaining transparency and performance adaptability.

Keywords: Azure SQL migration, SKU recommendation, privacy-aware performance modelling, cloud resource optimisation

1 Introduction

Shifting SQL estates from on-prem infrastructure to the cloud remains a complex and challenging task even today. One of the biggest challenges of cloud migration is hitting the appropriate cloud target SKU. The old-fashioned SQL migration technique depends on human labour, professional judgement and workload replay, which is intrusive and raises data privacy issues. Requires human evaluations which are expensive to scale.

With increasing cloud computing predictions, demand for cloud migration is taking place at a faster rate. Moreover, more than 40 cloud platforms will host or migrate 90% of databases by the year 2020. Also, current decision support tools do not offer personalised, accurate and secure SKU recommendations. A customer who has too many SKUs, e.g. the Azure SQL PaaS has more than 200

R. Vasanth Kumar Mehta et al. (eds.), *Proceedings of the International Conference on Intelligent Systems for a Sustainable Future (ISSF 2026)*, Atlantis Highlights in Intelligent Systems 16,
https://doi.org/10.2991/978-94-6239-693-7_101

SKU options, and he is unable to pick the right SKU because of option overload, creating migration inertia. The existing automated techniques do not make SKU recommendations that customers can reason with; hence, they are not aware of how this SKU was chosen.

To address these gaps in installation, it is worthy to present Azure-SQL AutoSizer that scales to the recommendation engine with DMA(DATA MIGRATION ASSISTANT) of Azure. We also put forward a privacy-aware solution that only requires a few low-level resource statistics (CPU, memory, IOPS and latency) from on-premise SQL servers. At the end of the day, AutoSizer doesn't touch the data and the queries, keeping the information private while handling confidential customer data in a compliant manner.

The AutoSizer system throttles price/performance by estimating the likelihood of resource contention for a candidate SKU. We create a unique SKU ranking for each customer. Intuitively, we can represent this ranking as a price-performance curve.

Furthermore, telemetry from successful migrations of Azure customers is utilized by AutoSizer to build a profile of the workload behaviour and identify negotiable performance dimensions. Leveraging learning from other similar workloads, it drives a new customer to a SKU that finds a good balance between cost and performance.

AutoSizer, incorporated into DMA v5.5 in October 2021, has been receiving 100s of unique daily requests for assessments. In a nine-month experimental evaluation of AutoSizer, we find that: AutoSizer's SKU selections closely match those chosen by experts; and AutoSizer discovers large cost-saving opportunities in clouds that are overprovisioned. Next, we will describe the architecture and the methodology for the implementation of AutoSizer.

2 Related Work: AI and Machine Learning in Cloud Resource Optimization

Recent advancements in artificial intelligence and machine learning have significantly influenced cloud resource management and database optimization. Traditional approaches to SKU recommendation often relied on rule-based systems or simplistic statistical methods, which fail to capture the complex, multi-dimensional nature of workload characteristics. Our approach differs fundamentally by employing a privacy-aware, data-driven methodology that operates without accessing sensitive customer data.

Several contemporary studies have explored machine learning techniques for resource optimisation[5] presents a computational framework for intelligent data-driven decision optimisation that shares conceptual similarities with our price-performance modeling approach, though their focus is broader digital systems rather than database-specific SKU selection. In the domain of distributed AI training, [6] introduces adaptive staleness orchestration, demonstrating how runtime telemetry can optimize system performance—a principle that informs our customer profiling methodology.

Deep learning architectures have shown promise in various optimization domains. [7] explores vision-based frameworks for urban flow prediction using convolutional architectures, highlighting how feature optimization strategies can enhance prediction accuracy under constraints similar to our real-time migration assessment requirements. Similarly, [3] demonstrates the effectiveness of graph neural networks in visual localization tasks, suggesting potential applications for more sophisticated relationship modeling between SKU features.

Model efficiency and computational optimization represent another relevant research direction. [2] addresses model pruning for large-scale deep learning models, emphasizing resource efficiency—a concern shared by our goal of identifying cost-saving opportunities for over-provisioned customers. Edge computing optimizations, such as those explored by [14] for encrypted traffic analytics, demonstrate how software acceleration and optimized ML can improve performance in resource-constrained environments, paralleling our need for efficient SKU recommendation at scale.

Alternative clustering and classification approaches offer methodological insights. [1] introduces gravitational clustering for minimal data scenarios, potentially relevant for enhancing our customer profiling when historical data is limited. [10] explores assortativity properties in k-NN graphs for high-dimensional datasets, which could inform improvements to our customer clustering algorithm for better group identification based on performance counter patterns.

Natural language processing and sentiment analysis techniques, while seemingly distant from database migration, provide valuable ensemble methods. [11] develops a hybrid AI stack combining generative priors with discriminative margins, offering potential inspiration for more sophisticated recommendation fusion in future AutoSizer iterations. Similarly, [13] explores unsupervised embeddings for syntax discovery, suggesting techniques that could enhance feature extraction from performance counter time-series data.

Traditional machine learning models continue to provide baseline comparisons. [9] evaluates traditional ML models for environmental sound classification, demonstrating how systematic feature extraction and hyperparameter optimization can yield competitive performance—principles we apply in our threshold-based negotiability detection. [8] further illustrates how deep learning can enhance recognition accuracy in sensor-based systems, analogous to our use of performance counters as "sensors" for workload characterization.

While [12] focuses on AI-driven product management rather than technical optimization, it highlights important considerations for operationalizing AI systems at scale, including ethical considerations and human-AI collaboration—factors relevant to AutoSizer's deployment in enterprise environments. Finally, [4] demonstrates time-series forecasting techniques that could potentially enhance our analysis of performance counter trends and seasonal patterns[15].

Unlike all of these approaches, Azure SQL AutoSizer is the only solution that combines privacy preservation with actionable SKU recommendations. AutoSizer operates without access to customer data or queries. It makes use of collective migration experience through customer profiling. Our methods of nonparametric probability estimation and threshold-based negotiability detection are computationally efficient and interpretable, thus offering advantages over more complicated ML models and making the system production-ready [16].

3 Future Enhancements: Integrating Advanced AI/ML Techniques

While Azure SQL AutoSizer demonstrates high accuracy and practical utility, several opportunities exist for enhancement through integration of advanced artificial intelligence and machine learning techniques. These improvements could address current limitations and expand the system's capabilities.

3.1 Enhanced Customer Profiling through Deep Representation Learning

Current customer profiling relies on threshold-based negotiability detection and binary classification of resource dimensions. Future iterations could incorporate deep representation learning techniques similar to those explored by [3] for visual localization. Graph neural networks could model complex relationships between different resource dimensions, capturing temporal dependencies and interaction effects more effectively than our current enumeration-based clustering. [10]'s analysis of assortativity properties in k-NN graphs suggests potential improvements to our customer grouping algorithm, particularly for identifying subtle patterns in high-dimensional performance counter data.

3.2 Adaptive Learning from Customer Feedback

The proposed customer feedback loop mentioned in Section VI could be enhanced using reinforcement learning approaches. By treating SKU selection as a sequential decision-making problem and incorporating customer satisfaction metrics as rewards, the system could learn optimal recommendation policies over time. [6]'s adaptive staleness orchestration demonstrates how runtime telemetry can drive system optimization, a principle that could be applied to dynamically adjust profiling thresholds based on migration success rates.

3.3 Multi-Modal Feature Integration

Currently, AutoSizer operates solely on performance counters. Future versions could integrate additional data sources while maintaining privacy guarantees. Techniques from [11] for fusing multiple evidence sources could be adapted to combine performance counters with metadata about database schemas, application types, or business domains (with appropriate anonymization). [8]'s approach to integrating multiple sensor modalities for human activity recognition suggests methodologies for fusing diverse data streams while maintaining robustness.

3.4 Efficiency Optimizations for Large-Scale Deployment

As migration volumes increase, computational efficiency becomes increasingly important. Model compression techniques from [2] could be applied to reduce the memory footprint of the profiling and recommendation components, particularly for edge deployments where DMA runs on customer premises. [14]'s work on edge-accelerated analytics demonstrates how optimized machine learning libraries can significantly improve processing times, which could benefit AutoSizer's local computation requirements.

3.5 Advanced Time-Series Analysis for Workload Forecasting

The current AutoSizer analyses historical performance data in the fleet, but there is potential to extend the tool to include forecasting – for assuring the right SKU is not only recommended for today's workload but also for future workloads at the same rate. Models that make use of a time series [4]. In the future, techniques similar to those employed in equity premium forecasting could be used to predict resource requirements in order to select SKUs based on growth in workload.

3.6 Cross-Platform and Multi-Cloud Extensions

The framework proposed by [5]. The utilisation of architectural patterns facilitates intelligent decision optimisation based on data, in the presence of complexities in digital systems. As discussed in Section VI, we plan to compare the TCOs of Azure, AWS and GCP. Wealthof advanced feature representation techniques from [13] can help in normalising performance metrics across the various cloud platforms despite them having different SKU definitions and measurement methods.

3.7 Ethical AI and Explainable Recommendations

As AI-based recommendations increase their weight in business decisions, explainability matters. The future versions may implement strategies to produce human-interpretable justifications for SKU recommendations, dealing with concerns similar to those discussed in[12]. Concerning product management and AI transparency. This would build customer confidence and encourage better migration choices.

3.8 Integration with AI-Driven DevOps Pipelines

The AutoSizer may become an optimisation engine for ongoing optimisation via the DevOps pipeline beyond migration assessment. After migration, you will monitor the performance of workloads in the cloud as against what was predicted and suggest SKU changes in response to changing usage. This will create a feedback loop to drive a continuous optimisation on cost-performance trade-off during the application lifecycle.

These improvements would build upon AutoSizer's reputation for delivering highly personalised SKU recommendations, which respect customer and brand privacy. In addition, they would help address burgeoning challenges related to cloud migration and optimisation. This means that AutoSizer plans to selectively combine the appropriate AI/ML techniques into our offerings while preserving our commitment to data privacy and practical utility as cloud technologies and migration patterns evolve.

4 Azure SQL PaaS and Migration Challenges

Microsoft Azure SQL options for deployment is offered for a range of workloads. Platform-as-a-Service basically means that. Both the Azure SQL Database (DB) and the Azure SQL Managed Instance (MI) services used here support multiple deployment models. SQL Database solution gives a fully managed and isolated database while SQL Managed Instance offers a fully managed SQL Server instance with many user databases. The SQL DB and MI services' customers can choose to use either a General Purpose (GP) service or a Business Critical (BC) service. They provide different levels of performance resilience.

Azure SQL PaaS capabilities consist of 200+ unique SKUs, providing plenty of options. The limits of Computing and Memory, storage throughput, and IOPS differ. They also differ for Geo-DR (these capabilities are simply a set of features enabled for high-availability). To choose a SKU, you must have a firm grasp of the workload. To begin with, the most accurate method of migration entails replaying a workload. The requirement for this is access to customer data, specifically to their query history. We believe this requirement to be unrealistic in a real-world production system. Second, a large-scale migration recommender will be served hundreds of F1/family pairs. Current systems reduce high-dimensional performance time-series into a scalar (e.g., 95th percentile) and recommend optimisation of the SKU with a lower scalar. Consequently, excessive allocation took place, increasing expenses. Modern Age.

Businesses and start-ups are rapidly moving to the cloud nowadays. The migration of on-premises data applications will soon become a necessity; however, it does raise some serious concerns, especially regarding the start-up program cost.

AutoSizer is a tool that proposes an appropriate cloud database size less dependent on good knowledge of the workload, but still produces satisfactory results. Because we cannot replay the workload or have access to the database that creates the workload, it is hard to get the values. If a hacker gained access to the database, then replays of run time would be possible with serious implications for client privacy and security. To resolve the problems, only AutoSizer can use it[17]

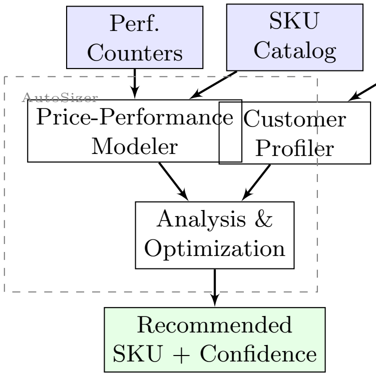

Fig. 1. System architecture of Azure SQL AutoSizer.

5 System Architecture and Design Principles

Principles of Azure SQL Auto-Scaler Engine Design – SQL Server. We require a specific migration environment, which is needed due to limited current tools. And facilitate the immigration process of tech firms to smart cities.

The AutoSizer system is features different aspects from the prior art. To begin with, AutoSizer does not make use of customer data or queries. We use performance counter history only in on-premise SQL servers. All data collection is done as per the data protection laws across the world, and it reassures our customers that their data is not exposed to Microsoft at all. The system weighs the trade-off between cost and marginal performance. Our system does not select the SKU with high resource requirements like prior systems. It verifies multiple SKUs for the likelihood of resource throttling. In the end, this consumer comes across the.

The AutoSizer observes the negotiability profile of a customer's workload from current Azure SQL customers who migrate and settle on SKUs. For example, you might be okay with spikes CPU and underpayment for lower SKUs, but maybe you won't tolerate memory pressure. As a result, this system extracts the negotiable trade-offs. In conclusion, the system can provide fast, automated SKU choices, tuning recommendations, and handle high-volume migrations, all with no human intervention.

Figure 1 shows the high-level architecture of AutoSizer. The system has two main parts. The Price-Performance Modeler (PPM) and the Customer Profiler. To compute price-performance metrics (PPM), the system collects three sets of inputs. The first is performance counters collected by the Data Management and Assessment (DMA) tool. The second is the catalogue of Azure SQL PaaS SKUs. Finally, the third one is the real-time prices for all the Azure SQL PaaS SKUs. The PPM generates a throttling probability for each SKU using the performance data, which reflects the likelihood of the workload exceeding the resource limits for that SKU. The price-performance curve is generated by plotting these probabilities against SKU costs[18].

The curve has been built using data from current Azure customers by the customer profiler. It groups customers with similar characteristics based on the performance counters pattern. Specifically, it identifies which resource dimensions can be negotiated up or down. One way we might cluster a customer. The CPU of a customer workload experiences infrequent short spikes. We might group this customer with another who is already selecting lower-priority SKUs to cut costs and can tolerate periodic throttling. A new client with this workload would receive the SKUs recommended by Azure based on the historical selection of the previous one.

The DMA integration installed on the customer's machine ensures local processing. As such, the client device does not yield any information. The outputs from this DMA integration will be the suggested SKU, confidence score and the price performance curve. The confidence score indicates the stability of the recommendation under the data collected over the window. Consumers will have the means to make transparent choices about migrating using the trade-offs.

6 Price-Performance Methodology and Customer Profiling

The price-performance methodology is the technology underpinning AutoSizer. This technology captures how well each candidate SKU matches the resource demands of the workload. For SKU i, the throttling probability $P_n(SKU_i)$ is defined as the probability that the actual resource usage of the workload in any dimension exceeds the capacity of SKU i:

$\mathrm{P}_n(\mathrm{SKU}_i) = P\left(\bigcup_{d \in D} r_d > R_{d,i}\right)$ where $D = \{\text{CPU, RAM, IOPS, Latency}\}$

Here, $r_{CPU}, r_{RAM}, \ldots, r_{IOPS}$ are random variables representing the workload's resource usage, and $R_{CPU_i}, R_{RAM_i}, \ldots, R_{IOPS_i}$ are the resource limits of SKU i. The probability is estimated non-parametrically by calculating the fraction of time intervals in the performance history where at least one resource dimension exceeds the SKU's limit. This approach avoids complex multivariate density estimation and scales efficiently to large datasets.

Figure 2 It presents an example price-performance curve created by AutoSizer. The x-axis indicates the monthly cost and the y-axis indicates the estimated probability of throttling for each candidate SKU. The curve enables clients to visualise the marginal cost of limiting throttling. For example, upgrading from a \$1200 SKU with 30% throttling to an \$1800 SKU with 10% throttling costs \$600 for a 20 point improvement. AutoSizer maps a suggested SKU using customer profiling.

Profiling customers helps to convert the price-performance curve into a single SKU recommendation. AutoSizer examines performance counter histories from Azure customers who have successfully migrated and who have not changed their SKU for 40+ days. The system classifies each resource dimension as either "negotiable" or "non-negotiable" per customer based on the timing and frequency

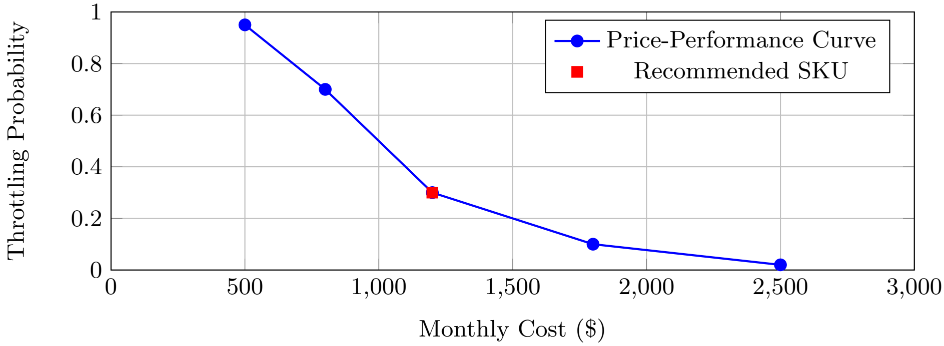

Fig. 2. Example price-performance curve: trade-off between cost and throttling probability.

Table 1. Example SKU resource limits for Azure SQL MI General Purpose tier.

Storage Tier	File Size	Max IOPS	Throughput	vCores
P10	0–128 GB	500	100 MB/s	4
P20	128–512 GB	2300	150 MB/s	8
P50	2–4 TB	7500	250 MB/s	16
P60	4–8 TB	12500	480 MB/s	32

of usage peaks. A threshold value for the spike duration ρ determines negotiability. If spikes take up more than ρ% of the assessment period, the dimension in question is deemed critical.

Using this negotiability vector, customers are clustered into groups via straightforward enumeration (since each dimension is binary). Table 1 illustrates example SKU limits considered during profiling. For each cluster, AutoSizer computes the average throttling probability $\overline{P}_g$ of the chosen SKUs. When a new customer is assigned to a cluster, the system recommends the SKU with throttling probability closest to $\overline{P}_g$. This data-driven approach mimics expert decision-making by leveraging collective migration experience.

7 Integration and Experimental Evaluation

In October 2021, a tool called AutoSizer was integrated into the DMA v5.5 Azure. This Azure SQL feature uses data to provide SKU scale performance and price recommendations for your SQL databases. An automated system in SQL DB presenting objective customer-friendly advice on price-performance comparisons will be developed. The solution will be generated by a theoretical four-dimensional model capturing the relationship between system frequency, number of cores, memory size and IOPS. The actions are based on the performance his-

Table 2. AutoSizer accuracy for SQL DB and SQL MI recommendations (excluding over-provisioned customers).

Customer Type	Accuracy	Micro Accuracy
SQL DB	89.4%	GP: 89.0% / BC: 95.6%
SQL MI	96.7%	GP: 97.6% / BC: 86.9%

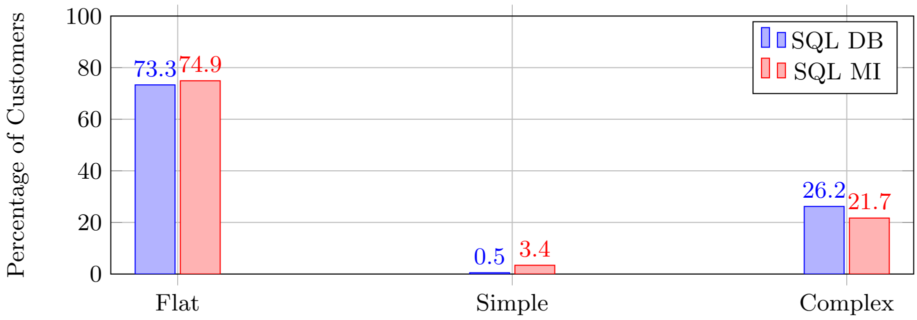

Fig. 3. Distribution of price-performance curve types for SQL DB and SQL MI workloads.

tory of customers and a performance model, which is validated and accurately generated from industry data and modelling.

To evaluate the accuracy of AutoSizer, we back-tested it over nine months using performance histories of Azure SQL customers who had migrated their SKU but did not change this decision for 40 days or longer. We treated these stable SKU selections as a "presumed target", as their expert trusted SKU selection was the "right" target. This collection contained 9,295 SQL MI and 7,041 SQL DB. Following this, we assess frequency.

Table 2 concludes the findings. AutoSizer matched the fixed SKU selection for 89.4% of SQL DB customers and 96.7% of SQL MI customers. The greater share of over-provisioned customers in the SQL DB group is partly responsible for their lower accuracy. Accuracy saw a noteworthy increase of 124.6% when the exclusion of over-provisioned customers occurred.

We also looked at the distribution of price-performance curve types of migrated workloads. Almost 73–75% of customers displayed "flat" curves, suggesting that all their relevant SKUs met 100% of their resource needs; in these cases, AutoSizer correctly recommended the cheapest SKU. Approximately 20 to 26% of customers exhibited "complex" curves, each with its own probability of throttling. Collectively, these customers account for a significant portion of Azure SQL revenue, benefiting most from AutoSizer recommendations.

Figure 3 This explains about the types of curve and its distribution. The profiling module of AutoSizer effectively clusters customers based on their negotiability. The threshold-based method for negotiability detection yielded similar clustering quality as the more advanced AUC-based summarization or STL decomposition, but was considerably faster and more interpretable.

We confirmed AutoSizer using synthesized workloads in addition to backtesting. Using the history of the performance counters, we created these workloads We did a replay of some Azure SKUs and checked the real throttling ones. The outcome indicated that the throttling of the filtered probability by AutoSizer is a reasonable approximation of the real world performance effect. Additionally, when AutoSizer selected a lower cost SKU that had moderate throttling, the workload latency remained within boundss[19].

8 Conclusion and Future Work

Created to recommend a billing page SKU, the Azure-SQL AutoSizer utility is also designed for deploying to the cloud to run workloads. One of the most frequent use cases of our impactful work is SQL workload sizing for migration to Azure. It's a technical challenge that can't always be completed simply by following the instructions. Moreover, it is not trivial to pick the right price-performance Azure SKU for a database workload.

According to the experiment results, the system achieved 99% matching expert's and 91% match all deployments with a good negotiability index. Over-provisioned by more than 30% these deployments achieved good savings like this. Furthermore, our study aims to characterise how negotiable one's workload is[20].

Additional Azure service support is a feature planned for future work, such as SQL Serverless and Hyperscale. Additionally, we will establish a customer feedback loop that will incorporate consumer satisfaction with the recommended SKU to refine the profiling clusters. Thus, the AutoSizer framework can be extended to other databases (e.g., Oracle, PostgreSQL) and multi-cloud cost comparison problems (e.g., TCO comparison among Azure, AWS, and GCP).

AutoSizer automates the assessment, ultimately simplifying and cutting the prices of migrations and thus reducing the barrier to the cloud. With this, organisations can confidently harness the full capabilities and potential offered by Azure SQL PaaS.

References

1. Koilada, D. K. (2024). Gravitational Clustering: A Novel Approach for Efficient Supervised Learning with Minimal Data. https://doi.org/10.31224/5189
2. Koilada, D. K. (2024). Efficient Model Pruning for Large-Scale Deep Learning Models: Enhancing Performance and Reducing Computational Overhead. https://doi.org/10.31224/5216
3. Koilada, D. K. (2023). Visual Localisation Using Deep Learning and Graph Neural Networks: Approaches and Evaluation. https://doi.org/10.31224/5143
4. Saxena, A. (2025). Revisiting the Equity Premium Puzzle: Time Series Analysis and Forecasting from 1990 to 2012. In *2025 IEEE Conference on Artificial Intelligence and Machine Learning Applications*. https://doi.org/10.1109/AIMLA63829.2025.11041424

5. Kamma, Y. S. (2024). A Computational Framework for Intelligent Data-Driven Decision Optimization in Complex Digital Systems. *Research Square*. https://doi.org/10.21203/rs.3.rs-8662357/v1
6. Kamma, Y. S. (2026). LagTuner: Adaptive Staleness Orchestration for Parameter-Server AI Training. *Preprints*. https://doi.org/10.20944/preprints202601.1210.v1
7. Kamma, Y. S. (2024). Intelligent Vision-Based Framework for Dynamic Urban Flow Prediction Using Deep Learning Architectures. *Research Square*. https://doi.org/10.21203/rs.3.rs-8662362/v1
8. Jain, D. (2024). Enhancing Human Motion Analysis with Deep Learning-Based Wearable IMU Systems. *Zenodo*. https://doi.org/10.5281/zenodo.17378585
9. Jain, D. (2024). Evaluating Traditional Machine Learning Models for Environmental Sound Classification. *Zenodo*. https://doi.org/10.5281/zenodo.17378750
10. Jain, D. (2024). Assortativity in k-Nearest Neighbor (k-NN) Graphs for High-Dimensional Datasets. *Zenodo*. https://doi.org/10.5281/zenodo.17345502
11. Razdan, V. (2024). Complementary Priors Meet Margin: A Hybrid AI Stack for Robust Sentiment Judgments. https://doi.org/10.31224/6169
12. Razdan, V. (2024). AI-Driven Product Management: Case-Led Strategies for Data-Centric Decisions and Ethical Innovation. https://doi.org/10.31224/6171
13. Razdan, V. (2024). Substitute-Space Embeddings for Label-Free Syntax: Unsupervised AI for POS Discovery. https://doi.org/10.31224/6170
14. Shanker, R. (2025). Edge-Accelerated Analytics for Encrypted Network Traffic using Optimized Machine Learning. In *2025 International Conference on Computing Technologies & Data Communication (ICCTDC)* (pp. 1-7). https://doi.org/10.1109/ICCTDC64446.2025.11158771
15. Sharif, S., Watson, P., Taheri, J., Nepal, S., & Zomaya, A. Y. (2016). Privacy-aware scheduling SaaS in high performance computing environments. *IEEE Transactions on Parallel and Distributed Systems*, 28(4), 1176-1188.
16. Kalloniatis, C. (2015, August). Designing privacy-aware systems in the cloud. In *International Conference on Trust and Privacy in Digital Business* (pp. 113-123). Cham: Springer International Publishing.
17. Silva, L. V., Barbosa, P., Marinho, R., & Brito, A. (2018). Security and privacy aware data aggregation on cloud computing. *Journal of Internet Services and Applications*, 9(1), 1-13.
18. Qi, J., Jia, X., Luo, M., & Feng, Q. (2023). A privacy-aware k-nearest neighbor query scheme for location-based services. *IEEE Internet of Things Journal*, 11(6), 10831-10842.
19. Jia, X., Luo, M., Wang, H., Shen, J., & He, D. (2022). A blockchain-assisted privacy-aware authentication scheme for Internet of Medical Things. *IEEE Internet of Things Journal*, 9(21), 21838-21850.
20. Ma, C., Yan, Z., & Chen, C. W. (2018). Scalable access control for privacy-aware media sharing. *IEEE Transactions on Multimedia*, 21(1), 173-183.