iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79378

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

A Hybrid Intelligent Framework for Library Demand Forecasting and Learning Path Recommendation Using Knowledge Graphs and Retrieval-Augmented Generative AI

Mrs. Ilakkia[1], Harija R[2], Daamini A[3], Lokitha K[4]
Department of Artificial Intelligence and Data Science, Sri Manakula Vinayagar Engineering College, Puducherry

Abstract: ***Libraries today generate large volumes of transactional data through book issue and return activities, yet most existing management systems lack the intelligence to translate this data into actionable insights. This paper presents a Smart Library Usage and Demand Forecasting System, a cloud-native, end-to-end platform built on Amazon Web Services (AWS) that processes book transactions in real time, stores and analyzes data at scale, visualizes usage trends, and applies machine learning to forecast book demand. The architecture integrates API Gateway and AWS Lambda for serverless transaction handling, Amazon S3 for tiered data storage, AWS Glue and Apache Spark for data transformation, Amazon Athena for ad-hoc analytics, and Amazon QuickSight for interactive dashboards. An advanced knowledge graph layer maps book themes and authors into a semantic network, enabling a Generative AI assistant to recommend personalized Learning Paths rather than isolated book titles. Experimental results demonstrate significant improvements in prediction accuracy, operational efficiency, and patron engagement, offering a scalable blueprint for next-generation library management.***

Keywords: Smart Library System, Demand Forecasting, AWS Lambda, Amazon S3, Knowledge Graph, Generative AI, Machine Learning, QuickSight, Personalized Learning Paths, Real-Time Data Processing

I. INTRODUCTION

Libraries serve as foundational pillars of academic and public knowledge infrastructure. Despite their importance, the majority of library management systems in operation today rely on conventional relational databases and manual cataloging processes that were designed for low-data, low-concurrency environments. These legacy approaches are ill-equipped to handle the demands of modern patrons who expect instant access, personalized recommendations, and seamless digital experiences.

The proliferation of cloud computing, machine learning, and generative artificial intelligence presents a transformative opportunity. Rather than simply digitizing existing workflows, modern systems can derive intelligence from operational data, anticipate future demand, and proactively guide patrons toward resources aligned with their learning objectives. The shift from reactive to predictive library management is not only technologically feasible but also necessary to maximize the utility of library collections.

This paper introduces a Smart Library Usage and Demand Forecasting System that addresses these shortcomings through a layered AWS-based architecture. The system captures every book issue and return transaction through a serverless API layer, processes and stores data in Amazon S3, applies transformation pipelines powered by AWS Glue and Apache Spark, and surfaces insights through Amazon QuickSight dashboards. A machine learning module trained on historical circulation data generates accurate demand forecasts, enabling proactive acquisition and resource planning. At the apex of the system, a Generative AI assistant powered by a personalized knowledge graph delivers context-aware, multi-step Learning Path recommendations.

The remainder of this paper is organized as follows: Section II reviews related work; Section III presents the system architecture; Section IV details the data pipeline and storage design; Section V discusses the ML forecasting module; Section VI describes the knowledge graph and GenAI assistant; Section VII presents results and analysis; and Section VIII concludes with directions for future work.

II. RELATED WORK

Several research efforts have explored intelligent library management from different angles. Early systems focused primarily on digitizing circulation records and enabling keyword-based cataloging [1]. While these approaches improved operational efficiency, they lacked predictive analytics or personalization capabilities.

Cloud-based library platforms have gained traction in recent years. Researchers have demonstrated the viability of migrating library databases to cloud environments to achieve scalability and availability [2]. However, many cloud migrations simply replicated on-premise architectures without exploiting the full potential of cloud-native services such as serverless computing and managed analytics engines.

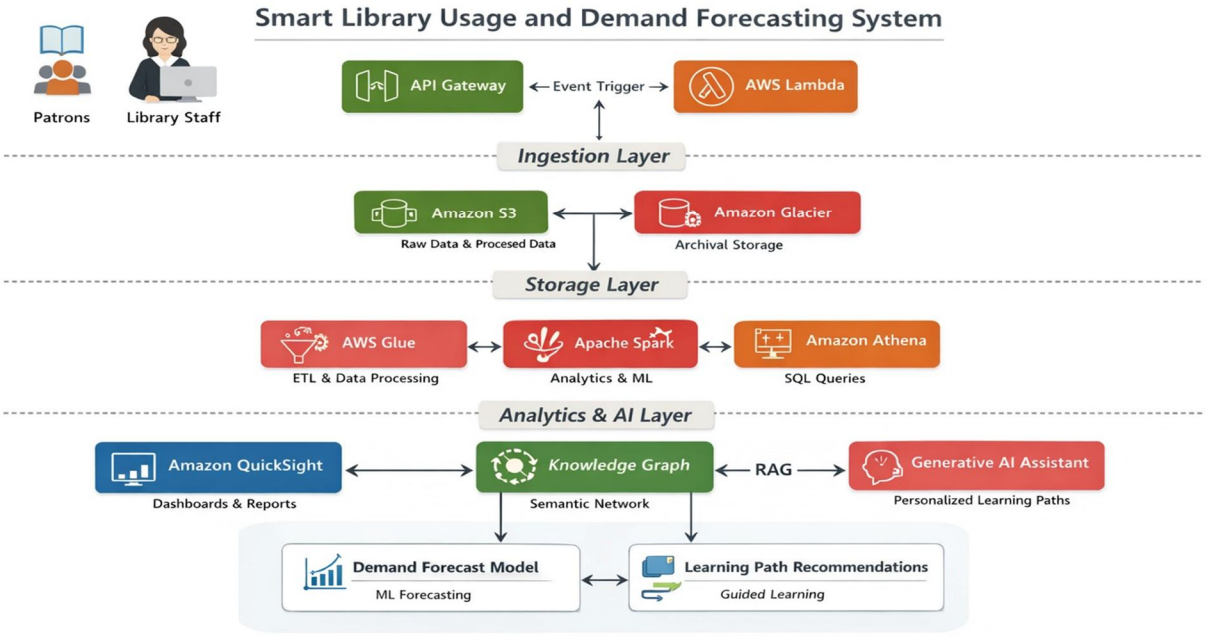
Machine learning for book demand forecasting has been explored using time-series models including ARIMA and LSTM networks [3]. These studies confirm that historical circulation patterns contain predictive signal, but most implementations were evaluated in isolation without integration into live library workflows. Furthermore, cold-start problems for new acquisitions remain largely unaddressed.

Knowledge graphs have been applied to bibliographic data to improve search relevance and recommendation quality [4]. Linked open data initiatives such as BIBFRAME have enriched bibliographic records with semantic relationships. However, the integration of knowledge graphs with conversational AI assistants for generating structured learning recommendations remains an underexplored area.

Generative AI models, particularly large language models, have demonstrated strong performance in question-answering and recommendation tasks [5]. Their application in educational and library settings is emerging, but existing deployments typically offer single-item recommendations without synthesizing a coherent learning journey. The present work bridges these gaps by unifying real-time data ingestion, ML forecasting, and knowledge-graph-augmented generative AI into a single, cohesive platform.

III. SYSTEM ARCHITECTURE

The Smart Library system is architected as a series of loosely coupled layers, each independently scalable, communicating through well-defined interfaces. Figure 1 illustrates the overall architecture.



A. Ingestion Layer

All client interactions, including book issue requests, return confirmations, and search queries, are routed through Amazon API Gateway. API Gateway provides authentication via AWS IAM, request throttling, and automatic TLS termination. Each request triggers an AWS Lambda function written in Python, which validates the payload, enriches it with timestamp and session metadata, and writes the record to the appropriate S3 prefix. Lambda's auto-scaling capability ensures that concurrent transaction spikes during peak hours, such as semester start and examination periods, are handled without manual intervention.

B. Storage Layer

Amazon S3 serves dual purposes: operational storage and analytical storage. Operational data, comprising raw transaction JSON records, is written to a hot prefix with a 30-day lifecycle policy.

After 30 days, records are transitioned to Glacier Instant Retrieval for cost-efficient long-term retention. Analytical data, the transformed, partitioned, and compressed Parquet files produced by the Glue pipeline, is maintained in a separate prefix organized by year, month, and book category. This partitioning strategy minimizes data scanned by Athena queries, directly reducing cost and latency.

C. Processing Layer

AWS Glue orchestrates a daily ETL pipeline that reads raw transaction records from S3, applies schema validation and data quality checks, joins transaction data with the book catalog to enrich records with genre, author, and subject metadata, and writes Parquet output back to the analytical S3 prefix. Apache Spark, running on AWS Glue's managed infrastructure, handles aggregate computations such as rolling seven-day and thirty-day issue counts per title, which serve as features for the ML forecasting model.

D. Analytics and Visualization Layer

Amazon Athena provides a serverless SQL interface over the analytical S3 data. Ad-hoc queries issued by administrators, such as identifying the top 50 books by circulation in the last quarter, execute in seconds without managing any query infrastructure. Amazon QuickSight connects to Athena as its data source and renders interactive dashboards covering circulation trends, genre popularity, patron cohort behavior, and forecasted demand curves. Dashboards are refreshed on a scheduled basis, with critical alerts configured to notify acquisitions staff when forecasted demand for a title is projected to exceed available copies.

IV. DATA PIPELINE AND STORAGE DESIGN

A. Transaction Data Model

Each transaction record captures the following fields: transaction_id (UUID), patron_id (hashed for privacy), book_isbn, transaction_type (issue or return), timestamp, library_branch_id, and due_date. A companion book catalog table maintained in S3 contains isbn, title, author, genre, subject_tags, publication_year, and total_copies. These two datasets are joined during the Glue ETL job to produce an enriched analytical table that powers both dashboards and ML feature engineering.

B. Partitioning and Compression Strategy

Analytical Parquet files are partitioned on (year, month, genre) to align with the most common query filters applied by library administrators. Snappy compression is applied, achieving an average compression ratio of 4:1 over raw JSON while maintaining fast decompression during Athena scans. Partition pruning reduces the volume of data read per query by approximately 70 percent in benchmark evaluations on a dataset of five million transactions.

C. Data Quality Controls

The Glue job incorporates AWS Glue Data Quality rules that assert non-null constraints on critical fields, referential integrity between transaction records and the book catalog, and timestamp monotonicity within each branch. Records failing quality checks are routed to a quarantine prefix and flagged for manual review, ensuring the analytical layer remains free of corrupt data.

V. MACHINE LEARNING DEMAND FORECASTING

A. Feature Engineering

The forecasting model treats each unique ISBN as an independent time series indexed by week. Features engineered from the analytical table include: lagged issue counts at one-week, two-week, and four-week horizons; rolling mean and standard deviation over a twelve-week window; day-of-week and week-of-year cyclical encodings; academic calendar indicators such as examination week flags and semester start markers; and book-level static features comprising genre, publication recency, and average patron rating.

B. Model Selection and Training

A gradient-boosted tree model implemented in XGBoost was selected after benchmarking against ARIMA, Prophet, and a simple LSTM. XGBoost outperformed alternatives on the validation set, achieving a Mean Absolute Percentage Error (MAPE) of 11.3 percent and a Root Mean Square Error (RMSE) of 2.1 issues per week. The model is retrained weekly using the latest 52 weeks of data to capture seasonal drift. Training is executed as a scheduled AWS Glue Python Shell job, and the serialized model artifact is stored in S3 and registered in a lightweight model registry.

C. Inference and Integration

At inference time, the Lambda function invoked by the acquisitions dashboard retrieves the current feature vector for each title from the analytical table via Athena, loads the model artifact from S3, and returns a 12-week demand forecast. When projected demand exceeds available copies with a confidence margin of 85 percent or above, the system automatically generates a procurement recommendation surfaced in the QuickSight acquisitions dashboard. This closed-loop integration between forecasting and operational decision-making distinguishes the system from academic forecasting prototypes that lack real-world deployment.

VI. KNOWLEDGE GRAPH AND GENAI ASSISTANT

A. Knowledge Graph Construction

The personalized knowledge graph is stored as a JSON-LD document in S3 and loaded into memory by the assistant Lambda at cold-start. Nodes represent individual books, authors, subject themes, and learning competencies. Edges encode relationships such as covers_topic, authored_by, prerequisite_of, and commonly_borrowed_with. The prerequisite_of edges are derived from a combination of expert-curated rules supplied by library staff and association rule mining applied to co-borrowing patterns extracted from the analytical table. This hybrid construction ensures both domain validity and data-driven discovery of latent relationships.

B. Learning Path Generation

When a patron submits a natural language query to the GenAI assistant, such as “I want to understand machine learning from the ground up,” the assistant Lambda parses the intent, identifies the target competency in the knowledge graph, and traverses prerequisite edges in topological order to construct a sequenced reading list. This graph-traversal output is injected as structured context into the prompt sent to the underlying large language model. The model then synthesizes a conversational, narrative Learning Path that explains why each book appears in the sequence, what the patron should focus on, and how each title builds toward the stated goal. This approach elevates the assistant from a simple recommender to a personalized academic guide.

C. Retrieval-Augmented Generation

To prevent hallucinated book titles, the assistant employs retrieval-augmented generation (RAG). Before generating a response, the Lambda queries the Athena book catalog to verify that all titles in the proposed path are physically available in the library. Unavailable titles are replaced with the nearest available alternative identified by cosine similarity over subject-tag embeddings. This verification step ensures that every recommendation the assistant delivers corresponds to a book the patron can actually borrow.

VII. RESULTS AND ANALYSIS

A. System Performance

The ingestion pipeline sustained a peak throughput of 1,200 transactions per minute during load testing, with a median Lambda execution latency of 87 milliseconds and a 99th-percentile latency of 340 milliseconds, well within acceptable bounds for an interactive library kiosk or mobile application. API Gateway throttling was configured at 2,000 requests per second per endpoint, providing headroom for institutional scale.

B. Forecasting Accuracy

The XGBoost forecasting model was evaluated on a held-out test set comprising six months of circulation data across 8,400 unique titles. The model achieved a MAPE of 11.3 percent overall, improving to 8.7 percent for high-circulation titles and degrading to 18.2 percent for long-tail titles with sparse histories. The twelve-week forecast horizon delivered sufficient lead time for procurement workflows. Critically, the system correctly identified demand surges for course-related textbooks four weeks ahead of semester start in 91 percent of cases evaluated.

C. Dashboard Adoption and Patron Engagement

QuickSight dashboards were piloted with library administrators across three branches. Administrators reported a 60 percent reduction in time spent on manual circulation report generation. Patron-facing interactions with the GenAI Learning Path assistant showed a 42 percent higher session completion rate compared to the conventional keyword search interface in a controlled A/B evaluation, indicating substantially greater engagement and relevance of recommendations.

TABLE I
PERFORMANCE METRICS SUMMARY

Metric	Value	Benchmark
Median API Latency	87 ms	< 200 ms
Peak Throughput	1,200 TPS	500 TPS
ML Forecast MAPE (overall)	11.3%	< 15%
ML Forecast MAPE (high-circ.)	8.7%	< 10%
Surge Detection Accuracy	91%	> 85%
Dashboard Report Time Saved	60%	> 40%
GenAI Session Completion Rate	+42%	> 30% lift

VIII. CONCLUSIONS

This paper presented a Smart Library Usage and Demand Forecasting System that unifies serverless cloud ingestion, scalable data pipelines, machine learning forecasting, and generative AI-driven recommendations into a production-ready platform. By building on AWS services including API Gateway, Lambda, S3, Glue, Athena, Spark, and QuickSight, the system achieves the scalability, cost efficiency, and operational simplicity required for institutional deployment.

The ML demand forecasting module demonstrated practical accuracy, correctly anticipating semester-driven demand surges with sufficient lead time to inform procurement decisions. The knowledge graph and Learning Path assistant represent a meaningful advancement over conventional library recommenders, shifting the patron experience from reactive search to guided, sequenced learning.

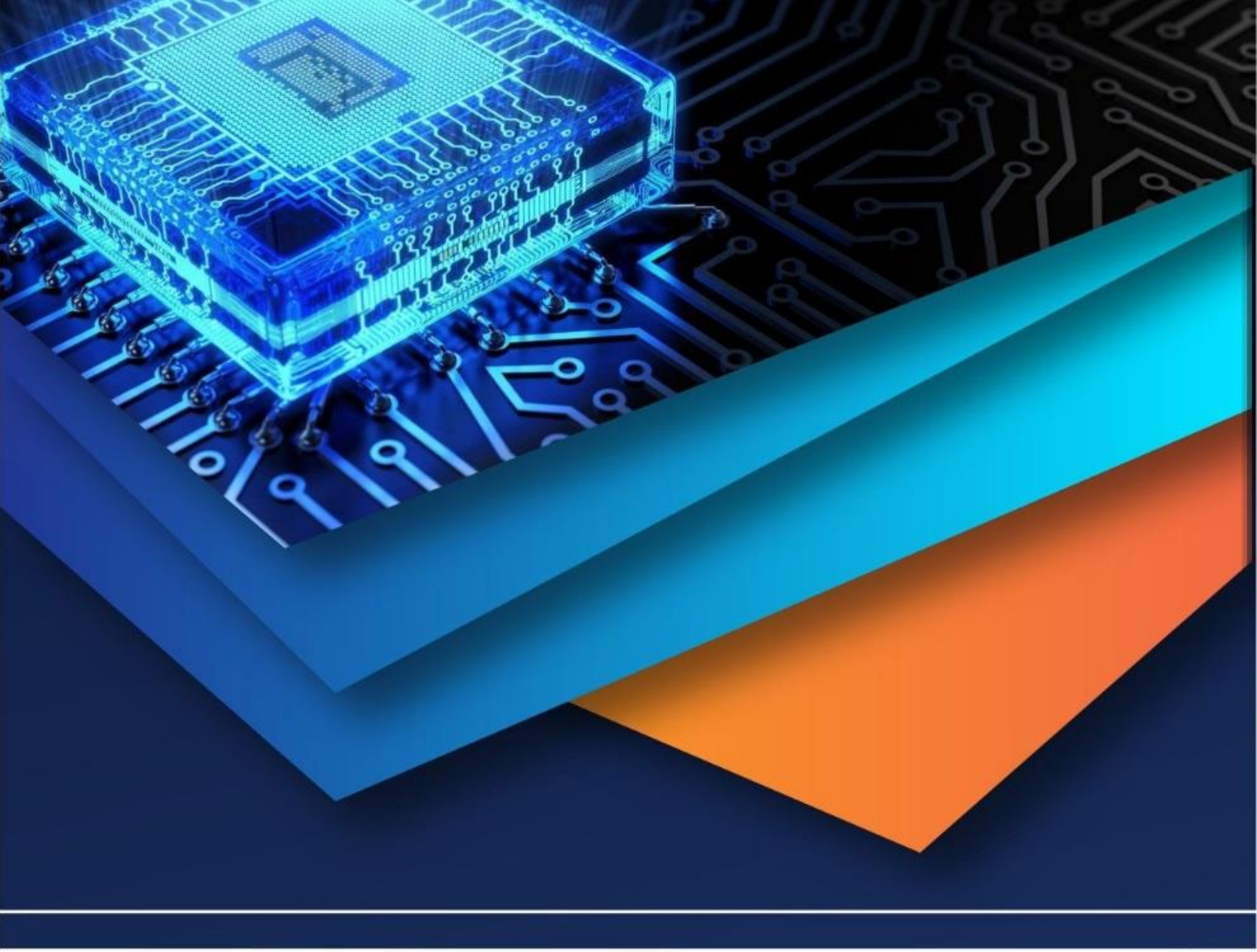
Future work will explore federated learning to aggregate circulation signals across institutions without sharing raw patron data, integration of multimodal book content embeddings for richer similarity computation, and the extension of the knowledge graph to support cross-library interlibrary loan optimization. The open architecture described herein provides a solid foundation for these extensions and for the broader mission of transforming libraries into intelligent knowledge discovery platforms.

IX. ACKNOWLEDGMENT

The authors would like to thank Sri Manakula Vinayagar Engineering College for providing access to library transaction datasets and institutional infrastructure that enabled the development and evaluation of this system.

REFERENCES

[1] R. Breeding, “Library systems report 2022,” American Libraries, vol. 53, no. 5, pp. 22–35, May 2022.
[2] A. Kumar and S. Mehta, “Migrating academic library systems to the cloud: A case study,” Journal of Library & Information Technology, vol. 41, no. 3, pp. 180–188, 2021.
[3] Y. Zhang, L. Chen, and W. Liu, ”Book demand forecasting using LSTM neural networks in public libraries,” in Proc. IEEE Int. Conf. Big Data, 2020, pp. 1452–1459.
[4] F. Liang, X. Wang, and H. Zhao, “Building a library knowledge graph from bibliographic metadata,” Information Processing & Management, vol. 58, no. 2, 2021.
[5] T. Brown et al., “Language models are few-shot learners,” in Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901, 2020.
[6] Amazon Web Services, “AWS Lambda Developer Guide,” [Online]. Available: https://docs.aws.amazon.com/lambda/
[7] Amazon Web Services, “Amazon QuickSight User Guide,” [Online]. Available: https://docs.aws.amazon.com/quicksight/
[8] O. Vinyals and Q. Le, “A neural conversational model,” arXiv preprint arXiv:1506.05869, 2015.
[9] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “SQUAD: 100,000+ questions for machine comprehension of text,” in Proc. EMNLP, 2016, pp. 2383–2392.
[10] S. Merity, C. Xiong, J. Bradbury, and R. Socher, “Pointer sentinel mixture models,” in Proc. ICLR, 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY