SAARTHIAI: An Generative AI-Driven Adaptive Learning System for Personalized Professional Learning Plans

Ashim Saha
Dept. Computer Sc. & Engineering
National Institute of Technology
Agartala, India

Anshuman Laskar
Dept. Computer Sc. & Engineering
National Institute of Technology
Agartala, India

Mainak Saha
Dept. Computer Sc. & Engineering
National Institute of Technology
Agartala, India

Soumyajit Das
Dept. Computer Sc. & Engineering
National Institute of Technology
Agartala, India

Rituraj Bhattacharjee
Dept. Computer Sc. & Engineering
National Institute of Technology
Agartala, India

ABSTRACT
In today's rapidly evolving professional landscape, individuals must continuously update their skills to remain competitive. However, traditional educational systems and static e-learning platforms often fail to provide personalized learning paths tailored to each professional's goals, background, and pace. To address this challenge, we present SaarthiAI, an AI-driven adaptive learning system designed to generate customized professional learning plans and deliver targeted, interactive instruction. SaarthiAI integrates a Roadmap Generator leveraging retrieval-augmented generation (RAG) and dense vector retrieval via FAISS to construct personalized learning roadmaps from a knowledge base of industry-relevant content. It incorporates adaptive assessments powered by large language models to evaluate proficiency and dynamically adjust content difficulty. An AI Tutor chatbot module provides real-time contextual assistance and guidance. The system is implemented using Python, utilizing the Hugging Face Transformers library, MongoDB for data storage, and a RESTful API for seamless integration. Our contributions include the novel integration of RAG for roadmap generation, dynamic assessment mechanisms, and an interactive AI Tutor, collectively advancing personalized professional education.

Keywords
Adaptive Learning System, FAISS, AI Tutor, Generative AI

1. INTRODUCTION
We In recent years, the demand for continuous professional development has surged, driven by rapid technological advancements and evolving industry standards. Professionals across sectors—from software engineering to healthcare—face mounting pressure to upskill and reskill in order to remain competitive. Traditional educational models, including instructor-led workshops and static online courses, often fall short in addressing individual learning needs due to their one-size-fits-all [10].Learners may become disengaged when content is misaligned with their prior knowledge, learning pace, or career objectives, leading to suboptimal knowledge retention and increased dropout rates [8][9].

Adaptive learning systems aim to overcome these challenges by personalizing content sequencing, difficulty, and feedback in real-time. Early approaches utilized rule-based engines to

adjust learning paths based on pre-defined criteria [7] , but these systems were limited by manual content curation and lacked scalability. More recent solutions incorporate machine learning techniques, such as Bayesian Knowledge Tracing [6] and Item Response Theory [5], to model learner proficiency dynamically. However, these methods depend heavily on large, labeled datasets of student responses, which may not be available for all domains. The advent of large language models (LLMs) and retrieval-augmented generation (RAG) has unlocked new possibilities for content personalization without extensive labeled data. RAG combines powerful pre-trained transformers with external knowledge sources, enabling systems to fetch relevant documents and synthesize tailored learning materials on-the-fly [16]. Dense vector retrieval libraries such as FAISS [15] provide efficient indexing and similarity search at scale, making real-time RAG feasible for production environments. While RAG has demonstrated success in knowledge-intensive tasks such as question answering and summarization [3][4], its application to adaptive education remains underexplored. Existing research predominantly focuses on static curriculum generation [1] or on-demand content summarization [2], leaving a gap in dynamic, learner-centric roadmap creation. Moreover, most adaptive testing frameworks rely on closed-form statistical models rather than leveraging LLMs' generative capabilities for nuanced feedback and open-ended question generation. In this paper, we present SaarthiAI, an AI-driven adaptive learning system that integrates RAG-based roadmap generation, LLM-powered adaptive assessments, and a conversational AI Tutor module. Our contributions include: (1) a novel application of RAG for personalized professional learning paths; (2) an adaptive assessment engine that generates and evaluates open-ended questions dynamically; and (3) an LLM-backed chatbot that provides real-time, context-aware tutoring.

2. RESEARCH GAP
All Adaptive learning platforms have progressed from early intelligent tutoring systems—which relied on simple rule-based branching based on right or wrong answers—to sophisticated, data-driven environments. Modern systems employ Item Response Theory to calibrate question difficulty dynamically and harness real-time learning analytics to monitor clickstreams, time-on-task, and hint usage, enabling immediate adjustments to content sequencing and feedback. Personalization lies at the heart of adaptivity: by combining explicit inputs (surveys, declared goals) with implicit signals (navigation patterns, error histories), contemporary learner

models predict optimal pathways and modalities. Recommender-system techniques—collaborative filtering based on peer activity and content-based matching to learner profiles—further tailor resource suggestions, boosting engagement and outcomes. Today's adaptive platforms go beyond checkpoint-based mastery. They build persistent, longitudinal learner profiles that carry context across sessions, analyze historical performance to anticipate needs, and even adapt to affective indicators (e.g., frustration or confidence). Unlike static curricula, these environments continuously reshape the learning journey in response to each interaction. The emergence of generative AI—particularly large language models integrated with retrieval-augmented generation—has transformed personalization. Retrieval mechanisms fetch up-to-date, domain-specific knowledge, and LLMs synthesize it into custom roadmaps, explanations, and quizzes on demand. This hybrid approach eliminates reliance on exhaustive manual content creation while ensuring factual grounding. Despite these advances, many widely used platforms still offer only coarse adaptivity: Coursera and edX suggest courses but maintain fixed syllabi; Khan Academy uses mastery gates but lacks fine-grained session-level tailoring; ChatGPT can answer questions dynamically but does not retain learner history or align with long-term goals. As a result, real-time adaptation, persistent context across logins, and integration of career objectives remain largely unmet needs. SaarthiAI addresses these limitations by uniting retrieval-augmented generation for grounded content synthesis, FAISS for high-speed vector retrieval, fine-tuned transformer models for interactive tutoring, and MongoDB for secure, persistent learner profiles. Together, these technologies power a continuously evolving, individual-centered learning companion. Key research gaps persist in continuous pathway adaptation, session-to-session memory, career-aligned planning, and fully closed-loop feedback integration. SaarthiAI fills these gaps by updating learning trajectories after every assessment, embedding career and industry trend analysis into roadmap generation, and leveraging ongoing learner feedback to refine instructional strategies in real time.

3. METHODS

SaarthiAI must seamlessly convert each learner's goals, constraints and weekly availability into a dynamic, personalized learning journey by first securing user registration and authentication (encrypted credentials, JWT sessions, role-based dashboards) and then guiding users through a profile builder that captures expertise, objectives, deadlines and study time. Leveraging RAG and FAISS, the platform generates week-by-week roadmaps of topics and curated resources, while an LLM-driven quiz engine assesses mastery each week—providing instant, item-level feedback and targeted remediation—and a context-aware AI Tutor chatbot offers on-demand hints, explanations, reminders and motivational nudges. Progress is tracked in real time via a dashboard displaying completed topics, quiz performance and study trends, which in turn informs continuous recalibration of future plans. At course end, a summative evaluation covering the full roadmap awards certification to learners meeting both performance (≥70%) and completion (≥80%) thresholds. Underlying these functional capabilities are strict non-functional requirements—sub-2 s latency, horizontal scalability through microservices, MongoDB sharding and FAISS indexing, AES-256 data encryption, OAuth2/JWT security, 99.9% uptime via auto-scaling and failover, and a modular, well-documented codebase—delivered through RESTful JSON APIs (Node.js/Express), MongoDB, FAISS, Hugging Face/Gemini Pro integration and an administrative analytics dashboard, all within a responsive, WCAG-compliant UI that requires minimal onboarding effort and provides tailored guidance for both students and professionals. The figure 1 mentions schematic flow of method with implicit signals (navigation patterns, error histories),

4. IMPLEMENTATION

4.1 Frontend Implementation

The SaarthiAI frontend is implemented using Next.js 14 with TypeScript and Tailwind CSS, following atomic design principles to ensure modular, reusable components. A mobile-first responsive layout adapts seamlessly across viewports from 320 px smartphones to 4 K displays. All UI elements—from navigation bars to quiz cards—are encapsulated as self-contained atoms and molecules, enabling rapid iteration and consistent styling throughout the application. The figure 2 represents overall architecture.

4.1.1 Authentication System

We built a custom authentication flow using HTTP-only cookies to store JWT access tokens (1 hr expiry) alongside rotating refresh tokens. Server-side middleware intercepts every protected route, verifying tokens and enforcing role-based dashboards. During onboarding, students specify their academic level (Freshman to Postgrad) and optional competitive-exam mode (GATE, FAANG), while professionals select career stage (Entry-level to Executive) and complete a brief skill-gap questionnaire. This bifurcated path ensures that subsequent UI prompts and resource recommendations align precisely to the user's profile.

4.1.2 Profile Configuration

Once authenticated, users engage with an interactive profile builder that captures core inputs in two stages. The **Goal Setup** step employs a dynamic career-roadmap selector and a commitment estimator—calculating weekly time allocation based on user-declared deadlines. In **Learning Preferences**, learners choose their preferred content media (video, text, or interactive modules) and an "intensity" level (Casual vs. Structured). All responses immediately update the client-side store, enabling real-time previews of how their roadmap will adjust.

4.1.3 Roadmap Generator

The roadmap planner combines a drag-and-drop timeline editor with resource-type filters (videos, courses, hands-on projects). Under the hood, each user's profile vector retrieves semantically similar content via FAISS (384-dimensional embeddings), and LangChain orchestrates a RAG pipeline to summarize and sequence topics. The resulting plan is rendered as both an interactive Gantt-style chart—showing weekly milestones—and a knowledge-graph visualization that highlights topic dependencies and progress.

4.1.4 Adaptive Assessment

Weekly quizzes are generated on the fly by prompting a fine-tuned LLM with the current roadmap context. Questions are calibrated for difficulty based on the learner's performance history, and the results dashboard presents personalized metrics: overall score, topic-wise breakdown, and certification status. Detailed feedback explains correct solutions and suggests targeted revision exercises, reinforcing weak concepts without altering the user's roadmap mid-week.

4.1.5 AI Mentor Interface

The AI Tutor interface delivers 24/7 conversational support with presence indicators signaling agent availability. A

persistent chat history—maintained client-side with snapshots stored in MongoDB—allows the tutor to reference prior interactions. Responses may include multi-modal content (code snippets, diagrams, or links), and proactive nudges (study reminders, motivation messages) help sustain engagement throughout the learning journey.

4.2 Backend Development

4.2.1 API Architecture

A suite of RESTful endpoints—/signup, /login, /generate-roadmap, /quiz/submit, /quiz/evaluate, /get-quiz, etc.—implements a clear separation of concerns. Each route encapsulates a single responsibility (e.g., roadmap creation or quiz storage), facilitating modular updates and straightforward debugging. Real-time chatbot queries flow over dedicated LLM APIs (/chat, /get-response), ensuring low-latency conversational experiences.

4.2.2 Data Modelling

MongoDB serves as the primary datastore, with separate collections for **Users**, **Goals**, **Quizzes**, and **EvaluationMetrics**. Schemas enforce required fields (e.g., userId, goalId, week, score) while permitting flexible, evolving structures for embedded documents like user preferences and performance tips. Compound and TTL indexes support efficient query patterns and automated log cleanup.

4.2.3 AI-Powered Roadmap Engine

Roadmap generation begins by retrieving top-k relevant documents from the FAISS index and, in parallel, issuing live web searches for the latest domain knowledge. A central AI agent merges these sources, then employs summarization and planning routines to produce coherent, sequenced learning objectives. This agent-based orchestration—Pseudocode:

```
def generate_roadmap(user_profile):
            docs_kb =
agent.retrieve_faiss(user_profile)
docs_web = agent.search_web(user_profile)
   combined = docs_kb + docs_web
   return
agent.summarize_and_plan(combined)
```

ensures dynamic freshness and modularity, allowing new retrieval plugins or search APIs to be integrated with minimal code changes.

4.2.4 Adaptive Quiz Logic

Using the same agent framework, quizzes are generated by extracting topic-specific materials from FAISS and augmenting them with real-time web data. Upon submission, the agent evaluates answers against AI-generated solutions, returns instant feedback, and optionally adjusts the user's roadmap based on performance. This closed–loop design delivers varied quizzes each week, consistent evaluation standards, and uninterrupted learning flow.

4.2.5 AI Services Integration

Core AI tasks—embedding creation, quiz generation, conversational tutoring—leverage a middleware layer that abstracts calls to Hugging Face Transformers, FAISS, and the Gemini 2.0 API. This unified interface hides service complexity from the application logic, ensuring that model upgrades or service migrations can occur transparently, without client-side impact. not ragged.

5. RESULTS

SaarthiAI guides each learner through a seamless, seven-step experience: users begin by registering and logging in, then select their role (student or professional) and specify clear career goals. The platform immediately generates a personalized roadmap, which learners follow by completing weekly adaptive quizzes. After each quiz, tailored feedback is delivered, and the plan is dynamically updated. Throughout this process, an AI Mentor chatbot remains available for on-demand support, offering hints, explanations, and motivational nudges. This structured yet fluid pipeline ensures that every interaction—from goal definition to real-time tutoring—is aligned to individual skill levels and objectives. Figure 3 represents the SaarthiAI.

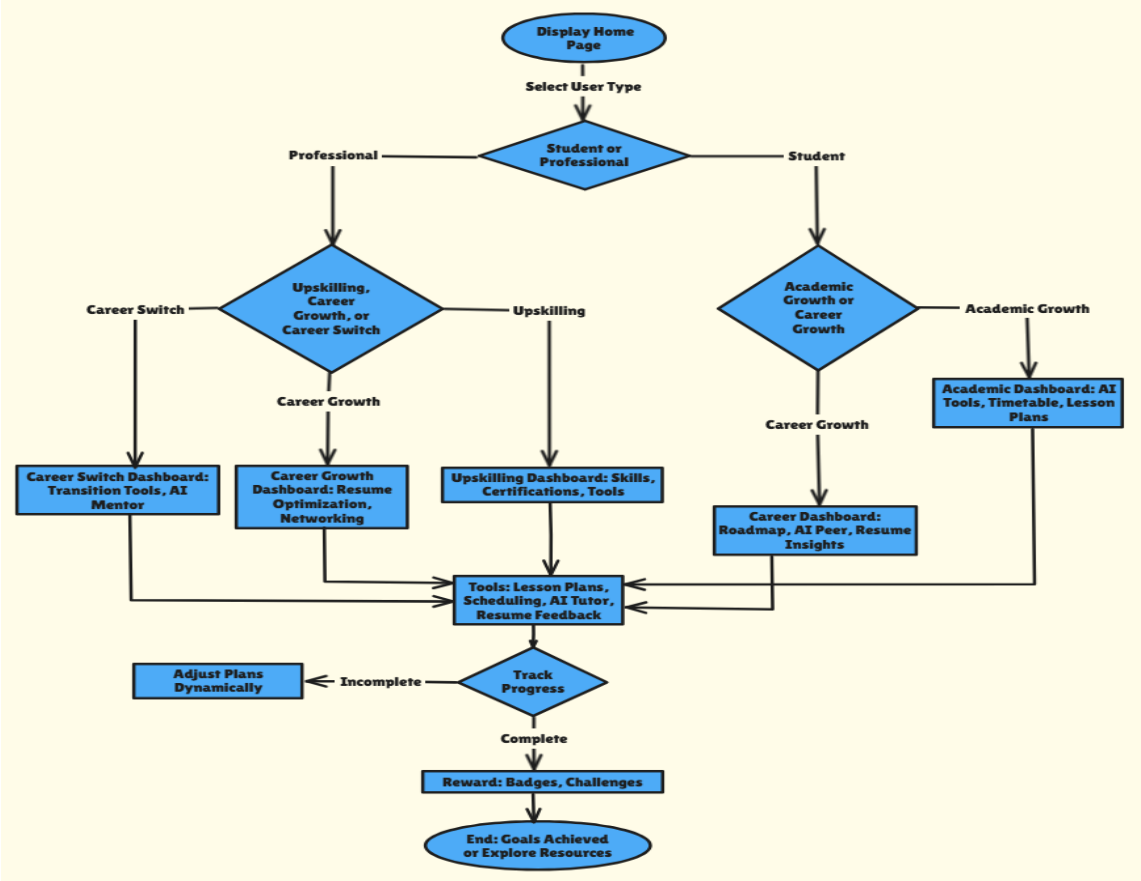

Fig 1: Flowchart of the proposed Method

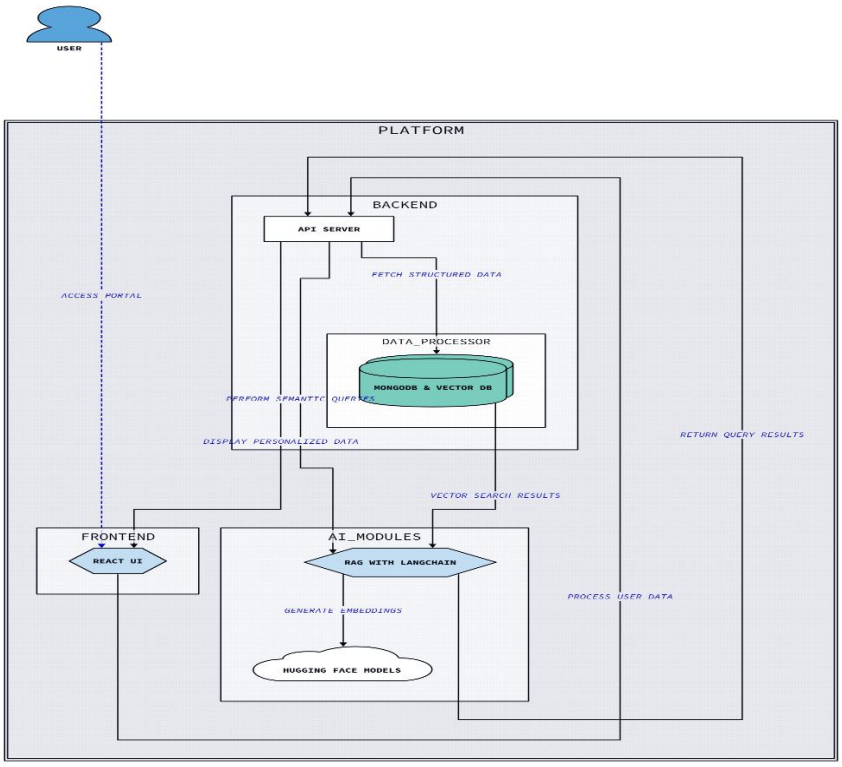

Fig 2: Architectural Design of Implementation

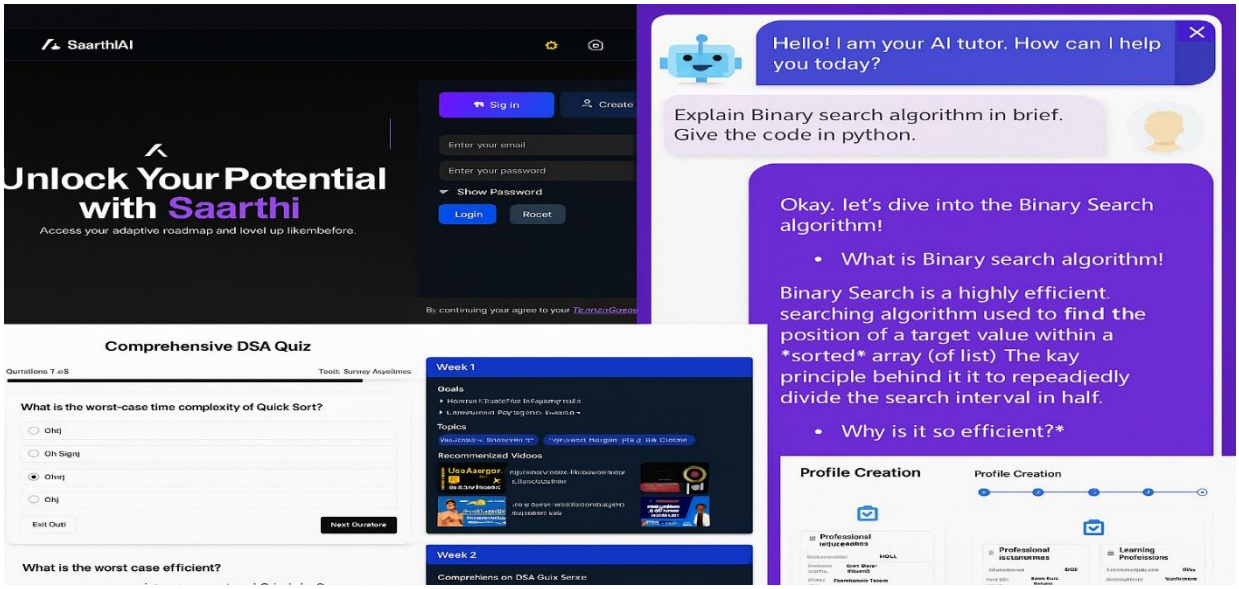

Fig 3: A Glimpse of SaarthiAI

6. DEPLOYMENT

Deploying SaarthiAI entails provisioning a scalable, secure production environment—typically cloud-hosted—configured with MongoDB for learner data and FAISS for vector retrieval, alongside the Next.js frontend and Node.js/Flask back end. A detailed rollout plan governs environment setup, configuration management (AI model parameters, RAG settings), build and packaging of code and models, and automated testing (unit, integration, and user acceptance). Git-based version control tracks both application and model updates, while scheduled backups and encrypted connections (OAuth2/JWT) ensure data integrity and security. Finally, zero-downtime deployments and auto-scaling (e.g., via Render or Kubernetes) guarantee continuous availability and performance under variable load.

7. CONCLUSION

SaarthiAI has demonstrated the power of combining retrieval-augmented generation, dense vector search, and transformer-based models to deliver truly personalized professional learning journeys: by translating each learner's goals, expertise, and availability into week-by-week roadmaps, generating context-aware assessments that adapt in real time, and providing on-demand mentorship through a conversational AI tutor, our full-stack MongoDB/Node.js/Next.js implementation achieves low-latency performance, robust scalability, and seamless end-to-end user experiences validated through rigorous testing. Looking ahead, we plan to enrich the platform with a real-time analytics dashboard that visualizes individual progress and mastery trends, extend our NLP and tutoring capabilities into multiple regional languages to broaden accessibility, and introduce new assessment modalities—such as gamified challenges, timed exams, and peer-to-peer competitions—to boost engagement. We will also evolve the AI mentor into a comprehensive career coach capable of résumé review and targeted job or internship recommendations, and deploy advanced behavior-driven algorithms that continuously refine content suggestions based on user interaction data. Finally, migrating SaarthiAI to cloud-native architectures on AWS, Azure, or GCP will enable automatic scaling and global availability, ensuring that personalized, AI-powered learning can reach and adapt to the needs of professionals around the world.

8. REFERENCES

[1] Zhu, X., Zhou, Y., & Xu, Z. (2022). Dynamic Curriculum Generation Using Neural Retrieval Models. IEEE Transactions on Learning Technologies, 15(3), 145–157.

[2] D Guu, K., Lee, K., Tung, Z., Pasupat, P., & Chang, M. (2020). Retrieval Augmented Language Model Training for Open Domain Question Answering. In Proceedings of ICLR.

[3] Karpukhin, V., Oguz, B., Min, S., Wu, L., Edunov, S., Chen, D., & Yih, W. (2020). Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of EMNLP.

[4] Izacard, G., & Grave, E. (2021). Leveraging Pre-trained Checkpoints for Sequence Generation Tasks. In Proceedings of NAACL.

[5] Embretson, S. E., & Reise, S. P. (2000). Item Response Theory for Psychologists. Psychology Press.

[6] Corbett, A., & Anderson, J. (1995). Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction, 4(4), 253–278..

[7] Brusilovsky, P. (2001). Adaptive Hypermedia. User Modeling and User-Adapted Interaction, 11(1-2), 87–110.

[8] Lee, S., Park, H., & Kim, Y. (2020). Engagement Metrics in E-Learning Platforms. Computers & Education, 143, 10368.

[9] Dziuban, C., Moskal, P., & Hartman, J. (2018). Bridging the Gaps: Blended Learning Approaches in Higher Education. Online Learning Consortium Journal, 12(4), 67–79.

[10] Johnson, M., & Brown, L. (2021). Data-Driven Adaptive Learning: Techniques and Frameworks. Journal of Learning Analytics, 8(1), 23–38

[11] FastAPI Contributors. (2021). FastAPI: The high-performance web framework for building APIs with Python 3.6+ based on standard Python type hints. Retrieved from https://fastapi.tiangolo.com/

[12] MongoDB, Inc. (2024). MongoDB: The Developer Data Platform. Retrieved from https://www.mongodb.com/

[13] Zenke, F., & Vogels, T. P. (2021). The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. Neural Computation, 33(4), 899–925.

[14] Wolf, T., Debut, L., Sanh, V., Chaumond, J., et al. (2020). Transformers: State-of-the-art natural language processing. In Proceedings of EMNLP: System Demonstrations.

[15] Johnson, J., Douze, M., & Jégou, H. (2017). Billion-scale similarity search with GPUs. IEEE Transactions on Big Data, 7(3), 535–547.

[16] Lewis, M., Oguz, B., Rinott, R., Petroni, F., Khashabi, D., & Riedel, S. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In Proceedings of EMNLP.