AUTOMATIC QUEST AND CHARACTER DIALOGUE GENERATION BASED ON LARGE LANGUAGE MODELS

CHEN-TAO LU[1], JASMINE LIN[2], KUAN-I WU, WEN-KAI TAI[1]

[1]Game Lab, National Taiwan University of Science and Technology, Taipei, R.O.C.
[2] San Marino high school, San Marino, California, USA
E-MAIL: charlie89517@gmail.com, xinyulin148@gmail.com, wktai@gapps.ntust.edu.tw

Abstract:

This paper proposes an automated game quest and dialogue generation system based on a combination of Large Language Models (LLMs) and a Retrieval-Augmented Generation (RAG) architecture. In computer game, the design of quests and NPC dialogues often relies heavily on manual scripting, which is time-consuming and labor-intensive. It also struggles to keep up with real-world events or the evolving needs of players. To address these challenges, our system automatically collects external information sources, filters and processes semantic conditions, and generates dynamic and interactive quest content through LLMs.

Our system integrates modules for data acquisition, semantic filtering, taboo word detection, and quest history tracking, allowing LLMs to produce personalized and contextually relevant task narratives and enabling quests to be updated in response to real-time information. As a result, our method enhances content diversity and player immersion. Also, the findings suggest that incorporating semantic understanding and generation capabilities significantly reduces content creation workload while offering a more adaptive and scalable approach to interactive storytelling in games.

Keywords:

Game Design, Quest Generation, Character Dialogue Generation, RAG, LLM.

1. Introduction

Quest generation in games has long relied primarily on manual authoring by designers. From background narratives and objective setting to dialogue details, every element requires meticulous drafting and iterative revision. While this approach ensures plot integrity and stylistic consistency, it incurs high labor costs and protracted development cycles. Furthermore, manually authored quest content is typically static; once game environments or player demands shift, it is difficult to update or expand them in a timely manner. Repetitive creative work can also result in homogeneous content, thereby diminishing players' sense of novelty and immersion. To resolve these issues of repetition and high costs, some developers have adopted "Canned Quests" as a templated solution, accelerating mass content production by reusing specific quest structures and text. However, despite their efficiency advantages, canned quests are often overly formulaic, lacking the diversity and contextual coherence needed to respond to diverse player behaviors and the dynamic evolution of the game world.

With the rapid growth of Large Language Models (LLMs) in scale and training data volume, their performance on natural language understanding and generation tasks has reached or even surpassed human levels. Through self-supervised pre-training, these models have accumulated vast linguistic knowledge. Under few-shot learning, they demonstrate strong generalization and adaptation capabilities, maintaining the quality and consistency of long-form narratives and multi-turn dialogues. Coupled with prompt engineering and fine-tuning technologies tailored for specific tasks, the accuracy and reliability of these models in specialized application domains have been further enhanced. However, LLMs may still exhibit "hallucinations" or background knowledge inconsistencies, indicating the necessity of coupling external knowledge retrieval with generation mechanisms, e.g., Retrieval-Augmented Generation, RAG. These technological breakthroughs provide a solid foundation for applying the RAG architecture to game quest generation in this paper.

We propose an automated game quest generation system integrated with a Retrieval-Augmented Generation (RAG) architecture. The system utilizes vector embedding technologies applied to taboo word semantic detection, news quest filtering, and plot summary retrieval. Users simply input game configuration parameters and expected goals, and the system automatically generates diverse quest content and NPC dialogues that conform to the game's worldview. There are five phases in our proposed framework: (1) Data Acquisition and Cleaning: Collects real-time news via Google News RSS, removing HTML tags and non-body content. (2) Taboo Word Checking: First performs literal

matching, then calculates similarity using dense and sparse vectors, combining the scores to determine violations. (3) Vectorization and Retrieval: Uses BAAI/bge-large-zh-v1.5 to generate dense and sparse vectors, and BAAI/bge-reranker-v2-m3 for rank enhancement, executing quest condition filtering and plot summary retrieval. (4) Content Generation: Feeds the filtered news snippets and plot summaries as prompts through interchangeable LLM APIs to automatically generate quest descriptions and NPC dialogues. (5) Evaluation and Analysis: Adopts metrics such as novelty, rationality, and coherence, combining human evaluation and automated metrics for qualitative and quantitative verification of the outputs. Although there are many application instances [1][2][3][4], we utilize Stardew Valley [5] as a case study to verify the feasibility and performance of the proposed method in practical game scenarios.

Our main contributions include (1) Automated Generation of Quests and Dialogues: Develop a generation pipeline combining RAG and LLMs to dynamically produce quest titles, descriptions, and NPC dialogues based on external contexts, and update the knowledge base so NPCs can act according to past storylines. (2) Dynamic Knowledge Base Updates: Design a crawling and vectorization process to continuously extract and update news and events, maintaining the timeliness of the content. (3) Multi-tier Safety Mechanisms: Establish a dual-layer taboo word system combining literal matching and semantic filtering to ensure the generated content adheres to policy and ethical guidelines.

2. Related Work

2.1. Automated Game Quest Generation

With the rapid advancements in LLMs and Procedural Content Generation (PCG), the automated generation of side quests and character dialogues has become a critical topic in digital narrative research. Existing studies generally combine Knowledge Graphs with LLMs, leveraging the semantic understanding and reasoning capabilities of the models to achieve dynamic and personalized quest and NPC interactions. These are often evaluated through dual verification (automated metrics and human judges) for narrative coherence and game experience quality [6]. Hybrid methods combining planning-based narratives and genetic algorithms treat story arcs as generative constraints, enabling automated exploration of multi-branch paths while maintaining high planning success rates, proven feasible and novel via blind tests by professional designers [7]. Event-driven architectures emphasize modeling character memories and desires to simulate highly diverse random events without relying on fixed templates [8], further enhancing the realism and diversity of quest flows [9].

Other systems propose automatically parsing game map elements, characters, and objectives from natural language stories, achieving synchronized generation of narrative texts and game levels [10], which showcases the potential of LLMs in narrative-driven level construction. For character simulation and plot collaboration, Yi Wang et al. [11] proposed "Abstract Acts" as a high-level narrative framework combining LLM-driven character simulation and planning technologies. Moreover, van Stegeren and Myśliwiec [12] fine-tuned GPT-2 via dialogue corpora, achieving high-quality creative evaluations in NPC dialogue generation. Davor Hafnar and Jure Demšar et al. [13] introduced a Zero-Shot reasoning paradigm that drives LLMs to generate personalized levels via real-time player behavior summaries, significantly boosting player engagement.

2.2. LLM Creativity

When measuring the creativity of LLMs, academia has evolved from traditional theoretical frameworks to utilizing various scales and task tests. Franceschelli and Musolesi [14] comprehensively analyzed the creative potential and socio-ethical impacts of the GPT series in poetry and story generation from four perspectives based on Boden's dimensions of Value, Novelty, and Surprise [15]. In terms of quantitative assessment, Chakrabarty et al. [16] designed the Torrance Test of Creative Writing (TTCW) to break creativity down into Fluency, Flexibility, Originality, and Elaboration. Yunpu Zhao et al. [17] developed an improved TTCT scale, constructed a 700-item test bank, and explored the impact of different prompting strategies on its accuracy and validity. Bellemare-Pepin et al. [18] also conducted a divergent creativity study, comparing the performance of fourteen mainstream LLM models under large-scale human reference using the Dissonance Associative Test (DAT) and Alternative Use Test (AUT). Some models have approached or surpassed the average human level. Shanahan et al. [19] conducted a case study on multi-voice generation from a literary perspective, verifying the quality and consistency of LLM in the story generation process, and pointing out that a lack of constraints can easily lead to stylistic confusion or logically incoherent paragraphs. In summary, each of these methods has its strengths and weaknesses. Theoretical frameworks provide a unified perspective. Scaled assessments facilitate index-based comparisons. Task-based tests are closer to the creative reality, and case studies can reveal detailed differences.

2.3. Vector Retrieval and Semantic Matching

Vector retrieval and semantic matching form the core of Retrieval-Augmented Generation (RAG) systems. The RAG architecture proposed by Lewis et al. [20] significantly improves performance on knowledge-intensive tasks by encoding queries into vectors to retrieve top-k passages and fusing them into the generator. Liu et al. [21] further combined dense and sparse vector retrieval to propose Dense Hybrid Retrieval, which weights the inner product of both vectors to capture both global semantic coverage and precise keyword matching. Building upon this, Chen et al. [22] proposed the BGE M3-Embedding model which simultaneously realizes multi-lingual, multi-functional (dense, multi-vector, sparse), and multi-granularity inputs within a single encoder.

3. **Methods**

Our proposed framework is formalized in four steps as follows:

Step 1. News Scraping: Pulling raw headlines and content from external news sources.

Step 2. Preprocessing and Retrieval: Cleaning news text, filtering game-relevant headlines via LLM, applying taboo word filtering, and retrieving contextual data relevant to the quest.

Step 3. Quest Generation: Generating quest titles, completion conditions, descriptions, storylines, and NPC dialogues sequentially through LLMs based on the retrieved context.

Step 4. Post-processing and Output: Formatting the final filtered and generated content into JSON format for output.

And, we divide our proposed system into the following modules:.

3.1. Modules

LLM Communication Module: Handles communication with external language model APIs (e.g., GPT).

News Data Scraping Module: Google News RSS feeds [24] are used to capture current events, restricted by timeframe and topic categorization so the news aligns with quest generation contexts. Regular expressions (<title>(.*?)-) extract the headlines, which are subsequently used to populate prompt templates.

Data Retrieval Module: We unify the use of the BAAI/bge [22] model series for semantic and vector matching, converting inputs into dense and sparse vectors for semantic similarity filtering and keyword importance ranking. This module conducts dual vectorization on texts for: (1) Quest Completion Condition Filtering: Retrieves game conditions most relevant to the real-time news text. (2) Story Summary Retrieval: Retrieves an NPC's quest history from the plot pool to maintain character continuity. The system uses BAAI/bge-large-zh-v1.5 [22] to generate dense vectors (semantic context) and sparse vectors (keyword weight/distribution). It calculates similarity distances (L2 for dense, inner product for sparse) via the pgvector extension [25][26], retrieves Top-k candidates, and merges them. It then applies BAAI/bge-reranker-v2-m3 [27] to sort the combined candidates [27] for the final prompt injection. Eventually, our method has to check all input content against taboo vocabulary through both lexical and semantic matching to guarantee generation safety using Forbidden Word Checking Module and track NPC and quest history to support subsequent personalized dialogue generation using Story Summary Module.

Forbidden Word Checking Module: This module prevents LLMs from generating inappropriate content by filtering out risky few-shot prompt examples based on Lexical Matching which uses Jieba tokenizer [29] and a Trie Tree built from the taboo word list to perform exact matches [30] and immediately discard explicitly violating vocabulary and Semantic Matching which calculates a weighted hybrid similarity score (dense + sparse) between the candidate text and taboo words. If the rank score exceeds a threshold (e.g., 0.45), the text is deemed too semantically close to taboo concepts and is appended as Negative Prompts.

Story Summary Module: Dedicated to extracting plot content associated with specific NPCs from the quest history database. This module ensures sequential logic by referencing the previous quests to reconstruct chronological plot dependencies. This module first retrieves all quests an NPC has participated in based on the NPC name recorded in the quest history table, and extracts the plot description text. To maintain chronological order and contextual coherence, the system also refers to the "preQuest" information to reconstruct the chronological relationship between tasks. The reason for using NPC names as search criteria is based on a finding that if an NPC plays a core role in a plot, the vector distance between their plot and their name in the vector space will be closer. For multiple tasks participated in by the same NPC, this module uses semantic vector similarity comparison to select the plots with the highest relevance to the target NPC, sorts them sequentially, and ultimately assembles a coherent narrative with contextual continuity. This content will be part of the prompts, assisting LLM in accurately simulating character memory and tone style during the dialogue generation stage.

3.2. Prompts

In this paper, the Prompt design serves as a crucial bridge connecting the language model and task generation logic. To ensure the accuracy and diversity of generated content, we conduct some design principles and conceptual architecture. The Prompt integration process involves replacing the text within curly braces based on user input and database data, generating the necessary context for different content, and then sending it to the LLM for text generation.

Here shows Prompt template examples:

Prompt Template for Quest Title Generation
Game Description = {GameDescription} Examples of Quest Title = {ExamplesOfQuestTitle} News Title = {NewsTitle} Quest Condition Keys & Values = {qcKeysAndValues} Quest Condition Descriptions = {qcDescriptions} Quest Target = {questTarget}

Prompt Template for Quest Description Generation
Game Description = {GameDescription} Examples of Quest Description = {ExamplesOfQuestDescription} News Title = {NewsTitle} Quest Condition Keys & Values = {qcKeysAndValues} Quest Condition Descriptions = {qcDescriptions}

Prompt Template for Story Generation
Game Description = {GameDescription} Quest Target = {QuestTarget} NPCs = {NPCs} Keywords = {Keywords} News Title = {NewsTitle} Quest Condition Keys & Values = {qcKeysAndValues} Quest Condition Descriptions = {qcDescriptions} Previous Related Task = {relatedTask}

Prompt Template for NPC Dialogue Generation
Game Description = {GameDescription} Quest Target = {QuestTarget} NPC List = {NPCList} News Title = {NewsTitle} Quest Condition Keys & Values = {qcKeysAndValues} Quest Descriptions = {qcDescriptions}

This system uses Markdown syntax to write Prompts, which is designed by the following principles:

• Consistency: All prompts must adhere to a unified format to reduce the probability of misinterpretation and improve the consistency and controllability of responses.

• Clarity: The designed prompts should clearly convey the desired task objectives and constraints, avoiding semantic ambiguity that could lead to model generation bias.

• Context Awareness: The Prompt structure should contain contextual information relevant to the task context [32] to help the model generate content that is closer to the real-world.

• Constraint Injection: For semantic or content constraints that need to be strictly followed (such as prohibited words), Prompt should contain clear prompts or control statements to guide the model to avoid inappropriate generation.

The above design principles ensure that the language model can generate semantically correct, stylistically consistent, and controllable task and dialogue content based on the input conditions, thereby improving the generation quality and game experience of the system.

4. Experimental Results

All experiments have made on a PC with NVIDIA GeForce RTX 4090 GPU and Intel Core i5-13400 CPU with 64 GB RAM, and using software such as Windows 11, Python 3.11.5, PyTorch 2.6.0+cu124, PostgreSQL 16, pgvector 0.7.4., OpenAI o4-mini API, and Google gemini-2.5-flash API.

To evaluate the effectiveness, we adopt performance analysis by measuring the average latency per generation phase, coherence evaluation by automatically scoring long-form story coherence using LLMs, and human evaluation to judge the result based on description consistency, news-quest coherence, and creativity.

Performance Analysis To evaluate the system's performance in the Stardew Valley [5] quest generation process, we used Chat-GPT o4-mini as a base and measured the average execution time for four steps: "News Scraping and Filtering," "Forbidden Words Detection," "Title and Description Generation," and "Dialogue Generation." Generating a single quest takes approximately 3 minutes on average. The Dialogue Generation phase is the primary bottleneck, taking 119.75 seconds (61.6% of total time). Title and Description generation takes 36.45 seconds (18.7%), News Scraping takes 32.16 seconds (16.5%), and Forbidden Word Detection is 6.1 seconds (3.1%).

Coherence Evaluation We compared quests generated with and without news assistance across 15 quest trees (depth of 10). And we show the results as follows:

• gemini-2.5-flash: Maintained the highest narrative completeness, though introducing news slightly lowered its coherence score from 6.00 to 5.93 at a much higher word count cost (~2853 words).

•o4-mini: Introducing news increased its average coherence score from 4.93 to 5.07. While slightly less coherent than Gemini, it demonstrated significantly better text efficiency (~1193 words).

Human Evaluation There are 30 participants who score the quality of generated quests in terms of

• Content: Whether the task text clearly presents the completion conditions.

•Creativity: Evaluated according to "Novelty, Surprise, and

Value" [15], with creativity being the average of the three indicators.

• News Coherence: Evaluates the relevance and rationality between the overall content of the task and the corresponding news event.

Table 1 shows that our proposed methos outperforms the previous model fine-tuned with GPT-2 in all three dimensions of "descriptive consistency", "creativity" and "news consistency" [12]. Among them, o4-mini performs best in terms of content accuracy (descriptive consistency), while gemini-2.5-flash is slightly better in terms of overall creativity; both also maintain a stable and high level in terms of news consistency. This result proves that our method can not only improve the novelty and interest of the text, but also take into account the close connection with news events.

TABLE 1 Average scores and creativity indicators of each model

System/Models	Content	Novelty	Surprise	Value	News Coherence	Creativity
Ours/Ours	5.23	5.18	5.13	5.43	5.32	5.25
Ours/o4-mini	**5.66**	5.02	5.08	5.32	**5.34**	5.14
Ours/gemini-2.5-flash	4.69	**5.38**	**5.19**	**5.56**	5.29	**5.38**

5. Conclusions

We have proposed a game quest and character dialogue generation system with semantic security detection, automated generation capabilities, and the ability to integrate real-time news events. Experimental results show that this system not only effectively filters sensitive or inappropriate words but also accurately converts news events into narratively compelling and playable task text through vector retrieval and a multi-stage RAG architecture. This system balances the clarity of task descriptions, the appeal of innovative text, and the reasonable connection to current events, demonstrating the feasibility and advantages of security detection and automated generation strategies in game task design.

Future work can be expanded in the following directions: (1) Explore and integrate more efficient Large Language Models (LLMs) to improve generation quality and reduce computational and latency costs; (2) Explore the synergistic effect of multimodal inputs (such as speech and video) and RAG to enrich task content; (3) Introduce a separate knowledge base for NPCs to optimize dialogue generation; (4) Apply this framework to more game scenarios and platforms to verify its universality and stability.

References

[1] Latitude. AI Dungeon. https://aidungeon.com/. Accessed: 2025-06-04.

[2] Game Rant. Skyrim Player Uses ChatGPT AI To Generate New Quests. https: //gamerant.com/ skyrim-quests-chatgpt-ai/. Accessed: 2025-06-04.

[3] 342Dave. Speaking Villagers –ChatGPT and TTS. https://www.curseforge.com/minecraft/mc-mods/speaking-villagers. Accessed: 2025-06-08.

[4] Friends & Fables. Friends & Fables. https://fables.gg/. Accessed: 2025-06-08.

[5] ConcernedApe. Stardew Valley. https://www. stardewvalley.net/. Accessed: 2025-06-04.

[6] Trevor Ashby, Braden K Webb, Gregory Knapp, Jackson Searle, and Nancy Fulda. “Personalized Quest and Dialogue Generation in Role-Playing Games: A Knowledge Graph- and Language Model-based Approach”. In: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. Hamburg, Germany: ACM, 2023. ISBN: 9781450394215. DOI: 10.1145/3544548.3581441.

[7] Edirlei Soares de Lima, Bruno Feijó, and Antonio L. Furtado. “Procedural generation of branching quests for games”. Entertainment Computing 43 (2022), p. 100491. ISSN: 1875-9521. DOI: https://doi.org/10.1016/j.entcom. 2022.100491.

[8] Vincent L. Prins, Jelmer Prins, Mike Preuss, and Marcello A. Gómez-Maureira. “StoryWorld: Procedural Quest Generation Rooted in Variety & Believability”. Proceedings of the 18th International Conference on the Foundations of Digital Games. FDG ’23. Lisbon, Portugal: ACM, 2023. ISBN: 9781450398558. DOI: 10.1145/3582437.3587181.

[9] Suzan Al-Nassar, Anthonie Schaap, Michael Van Der Zwart, Mike Preuss, and Marcello A. Gómez-Maureira. “QuestVille: Procedural Quest Generation Using NLP Models”. Proceedings of the 18th International Conference on the Foundations of Digital Games. FDG ’23. Lisbon, Portugal: ACM, 2023. ISBN: 9781450398558. DOI: 10.1145/ 3582437.3587188.

[10] Muhammad U. Nasir, Steven James, and Julian Togelius. Word2World: Generating Stories and Worlds through Large Language Models. 2024. arXiv: 2405.06686 [cs.CL].

[11] Yi Wang, Qian Zhou, and David Ledo. “StoryVerse: Towards Co-authoring Dynamic Plot with LLM-based Character Simulation via Narrative Planning”. In: Proceedings of the 19th International Conference on the Foundations of Digital Games. FDG 2024. ACM, May 2024, pp. 1–4. DOI: 10.1145/3649921.3656987.

[12] Judith van Stegeren and Jakub Myśliwiec. “Fine-tuning GPT-2 on annotated RPG quests for NPC dialogue generation”. In: Proceedings of the 16th International Conference on the Foundations of Digital Games. FDG ’21. Montreal, QC, Canada: Association for Computing Machinery, 2021. ISBN: 9781450384223. DOI: 10.1145/3472538.3472595.

[13] Davor Hafnar and Jure Demšar. “Zero-Shot Reasoning: Personalized Content Generation Without the Cold Start Problem”. In: IEEE Transactions on Games (2024), pp. 1–10. DOI: 10.1109/TG.2024.3421590.

[14] Giorgio Franceschelli and Mirco Musolesi. “On the creativity of large language models”. In: AI amp; SOCIETY (Nov. 2024). ISSN: 1435-5655. DOI: 10.1007/s00146-024-02127-3.

[15] Margaret A. Boden. “Creativity in a nutshell”. In: Think 5.15 (2007), pp. 83–96. DOI: 10.1017/ S147717560000230X.

[16] Tuhin Chakrabarty, Philippe Laban, Divyansh Agarwal, Smaranda Muresan, and Chien-Sheng Wu. “Art or Artifice? Large Language Models and the False Promise of Creativity”. In: Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems. CHI ’24. Honolulu, HI, USA: ACM, 2024. ISBN: 9798400703300. DOI: 10.1145/ 3613904.3642731.

[17] Yunpu Zhao, Rui Zhang, Wenyi Li, and Ling Li. “Assessing and Understanding Creativity in Large Language Models”. In: Machine Intelligence Research 22.3 (Apr. 2025), pp. 417–436. ISSN: 2731-5398. DOI: 10.1007/s11633-025-1546-4.

[18] Antoine Bellemare-Pepin, François Lespinasse, Philipp Thölke, Yann Harel, Kory Mathewson, Jay A. Olson, Yoshua Bengio, and Karim Jerbi. Divergent Creativity in Humans and Large Language Models. 2024. arXiv: 2405.13012 [cs.CL].

[19] Murray Shanahan and Catherine Clarke. Evaluating Large Language Model Creativity from a Literary Perspective. 2023. arXiv: 2312 . 03746 [cs.CL].

[20] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. “Retrieval-augmented generation for knowledge intensive NLP tasks”. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS ’20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.

[21] Ye Liu, Kazuma Hashimoto, Yingbo Zhou, Semih Yavuz, Caiming Xiong, and Philip Yu. “Dense Hierarchical Retrieval for Open-domain Question Answering”. In: Findings of the Association for Computational Linguistics: EMNLP 2021. Nov. 2021, pp. 188–200. DOI: 10.18653/v1/2021.findings - emnlp . 19.

[22] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. 2024. arXiv: 2402.03216 [cs.CL].

[23] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. “Retrieval - Augmented Generation for Knowledge- Intensive NLP Tasks”. In: Advances in Neural Information Processing Systems. Vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474.

[24] Google News RSS Feed https:// news.google.com/ rss?hl=zh-TW&gl=TW&ceid=TW:zh- Hant. Accessed: 2025-04-20.

[25] TensorChord. pgvecto.rs: pgvector on Rust steroids. https://github.com/ tensorchord/ pgvecto.rs. Accessed: 2025-04-20.

[26] pgvector: Open source vector similarity search for Postgres. https://github.com/pgvector/pgvector. Version v0.8.0; PostgreSQL License. 2025.27

[27] Beijing Academy of Artificial Intelligence. BAAI/bge-reranker-v2-m3: A Multilingual Reranker Model Based on BGE-M3 Embeddings. https:// huggingface.co/BAAI/bge-reranker-v2-m3. Apache-2.0 License; 568M parameters. 2025.

[28] FlagOpen contributors. FlagOpen/FlagEmbedding: Retrieval and Retrieval-augmented LLMs. https:// github.com/FlagOpen/FlagEmbedding. Version v1.3.4 (Feb 7, 2025); MIT License. 2025.

[29] f”xsjy” and contributors. jieba: Chinese Text Segmentation for Python. https://github.com/fxsjy/ jieba. Version v0.42.1 (Jan 20, 2020); MIT License. 2025.

[30] Maha Maabar. Trie Data Structure. https:// bioinformatics.cvr.ac.uk/trie-data-structure/. Accessed: 2025-05-09.

[31] Google Cloud. Prompt Engineering: overview and guideline, https://cloud.google.com/discover/ what-is-prompt-engineering?hl=zh_tw. Accessed 2025-04-22.

[32] Amazon Web Services. Prompt engineering concepts. https://docs.aws.amazon.com/bedrock/latest/userguide /prompt-engineering-guidelines.html. Accessed 2025-04-22.