HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



Đinh Thị Hương Thảo

NGHIÊN CỨU VÀ ỨNG DỤNG LLM CHO BÀI TOÁN TẠO NHIỆM VỤ (QUEST) TRONG GAME CASUAL

Chuyên ngành : **Khoa học máy tính**

Mã số: **8.48.01.01 (Khoa học máy tính)**

TÓM TẮT ĐỀ ÁN TỐT NGHIỆP THẠC SĨ

HÀ NỘI - 2025

Đề án tốt nghiệp được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: TS. Đặng Hoàng Long

Phản biện 1: ……

Phản biện 2: ……

Đề án tốt nghiệp sẽ được bảo vệ trước Hội đồng chấm đề án tốt nghiệp thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông

Vào lúc: …… giờ …… ngày …… tháng …… năm……

Có thể tìm hiểu đề án tốt nghiệp tại:

- Thư viện của Học viện Công nghệ Bưu chính Viễn thông.

MỞ ĐẦU

1. Lý do chọn đề tài

Bối cảnh thị trường thị trường game casual đang chiếm tỷ trọng lớn trên toàn cầu nhờ phù hợp với hành vi tiêu thụ nhanh của người dùng hiện đại. Đặc biệt tại Việt Nam, ngành game đang bùng nổ với quy mô tăng từ 192 triệu USD lên gần 1.5 tỷ USD (2024), đưa Việt Nam trở thành quốc gia đứng đầu thế giới về lượng tải game mobile.

Thách thức khi duy trì tỷ lệ giữ chân người chơi (retention) trong các game live-service, nhà phát hành buộc phải liên tục cập nhật nội dung (nhiệm vụ, sự kiện). Tuy nhiên, phương pháp thiết kế thủ công đang gặp rào cản lớn: chi phí cao, tốn thời gian, khả năng mở rộng thấp và thiếu tính cá nhân hóa, dẫn đến trải nghiệm người chơi dễ bị nhàm chán.

Đề tài tập trung nghiên cứu ứng dụng LLM để tự động hóa quy trình tạo nhiệm vụ (Quest). Mục tiêu là giải quyết bài toán về sản lượng nội dung, tối ưu chi phí vận hành và tăng cường sự đa dạng trong trải nghiệm game casual qua việc tận dung các mô hình ngôn ngữ lớn.

Từ bối cảnh thực tiễn và nhu cầu của ngành công nghiệp game toàn cầu nói chung và Việt Nam nói riêng, đề tài **"Nghiên cứu và ứng dụng LLM cho bài toán tạo nhiệm vụ (Quest) trong Game Casual"** được lựa chọn nhằm giải quyết bài toán tự động hóa nội dung, giảm chi phí vận hành, tăng tính đa dạng và cá nhân hóa nhiệm vụ cho game.

2. Mục đích nghiên cứu

Mục tiêu tổng quát: nghiên cứu và ứng dụng các kỹ thuật prompt engineering trong việc huấn luyện mô hình tạo các nhiệm vụ trò chơi

Mục tiêu cụ thể:

(i) Nghiên cứu cơ sở lý thuyết về game casual, hành vi người chơi, mô hình ngôn ngữ lớn và các kỹ thuật Prompt Engineering.

(ii) Xây dựng pipeline sinh nhiệm vụ tự động cho game casual dựa trên LLM và lớp kiểm định (validation) đảm bảo nhiệm vụ hợp lệ, đa dạng và khả thi trong gameplay.

(iii) Kiểm thử và đánh giá chất lượng nhiệm vụ tạo sinh thông qua kết quả thực nghiệm và khảo sát người dùng.

3. Ý nghĩa khoa học và thực tiễn

Ý nghĩa khoa học

Đề tài áp dụng các kỹ thuật tiên tiến của lĩnh vực NLP như Transformer Attention [**Error! Reference source not found.**], schema-constrained generation [**Error! Reference source not found.**], in-context learning [**Error! Reference source not found.**] vào một bài toán mới: tạo nhiệm vụ game casual.

Ý nghĩa thực tiễn

Đề tài mang ý nghĩa thực tiễn cao, giúp giảm tải khối lượng công việc lặp lại cho đội ngũ thiết kế game; tạo ra các nhiệm vụ nhanh, đa dạng, hạn chế trùng lặp, phù hợp gameplay, tiềm năng mở rộng ứng dụng sang nhiều thể loại game hoặc lĩnh vực sáng tạo số như giáo dục, mô phỏng, gamification doanh nghiệp, từ đó thúc đẩy việc ứng dụng AI trong ngành công nghiệp game Việt Nam – một ngành đang tăng trưởng mạnh [**Error! Reference source not found.**].

4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu

Mô hình ngôn ngữ lớn (LLM) và khả năng tạo nhiệm vụ game casual.

Phạm vi nghiên cứu

Đề tài có nghiên cứu trong phạm vi:

(i) Game casual trên nền tảng di động.
(ii) Nội dung nhiệm vụ (quests) dưới dạng JSON, không nghiên cứu sinh level, hình ảnh, hội thoại.
(iii) Mô hình dùng phương pháp Prompt Engineering, không fine-tuning mô hình.
(iv) Thời gian thực hiện: năm học 2025.

5. Phương pháp nghiên cứu

Các phương pháp nghiên cứu áp dụng trong đề tài:

Phương pháp lý thuyết: tổng hợp, phân tích tài liệu (thu thập và phân tích lý thuyết về game casual, kiến trúc Transformer, các kỹ thuật Prompt Engineering…)

Phương pháp thực nghiệm: chạy thử nghiệm nhiều lượt tạo nhiệm vụ theo các kịch bản gameplay thực tế, đánh giá các câu hỏi tạo ra bằng validation layer; đánh giá chất lượng nhiệm vụ sinh ra theo tiêu chí cụ thể.

Phương pháp khảo sát – đánh giá: tổ chức khảo sát người dùng (người chơi hoặc nhà phát triển game) để đánh giá tính lợp lý của các nhiệm vụ sinh ra

Các công công cụ sử dụng: Dify, mô hình LLM (GPT/LLaMA API), Unity, Google Forms…

CHƯƠNG 1. CƠ SỞ LÝ LUẬN

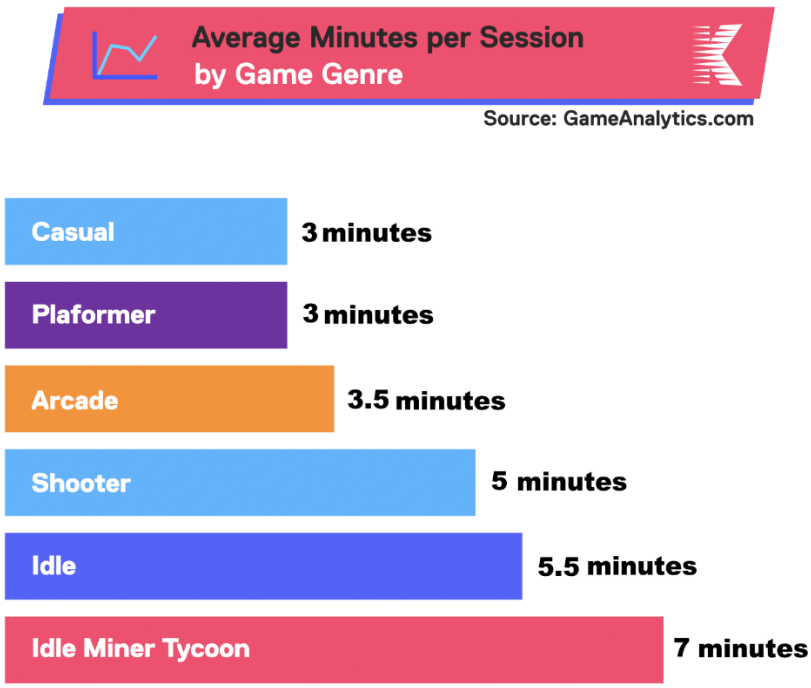
Chương này tập trung trình bày các kiến thức nền tảng cho toàn bộ đề tài. Đầu tiên, nghiên cứu sẽ giới thiệu vắn tắt đặc điểm game casual và hệ thống nhiệm vụ. Kế đến, chương cũng trình bày tổng quan về mô hình ngôn ngữ lớn và các kỹ thuật sinh nội dung. Cuối cùng, chương tiếp tục đưa ra những vấn đề còn tồn tại trong thiết kế nhiệm vụ và xác định nhu cầu ứng dụng LLM. Những nội dung này đặt cơ sở cho việc xây dựng quy trình sinh nhiệm vụ tự động ở CHƯƠNG 2.

1.1 GAME CASUAL VÀ ĐẶC ĐIỂM NHIỆM VỤ (QUEST)

1.1.1 Đặc trưng của game casual

Lý do game casual có thể chiếm lĩnh thị trường game tỷ đô và cạnh tranh với những dòng game khác trên thị trường do chúng có những đặc trưng nổi bật sau đây.

Thứ nhất, game casual được thiết kế với luật chơi đơn giản, thời gian học ngắn và vòng lặp gameplay rõ ràng.



Hình 1-1 Biểu đồ so sánh hời gian chơi trung bình mỗi session theo thể loại game [Nguồn: GameAnaytics.com]

Thứ hai, bên cạnh đó, game casual thường sở hữu luật chơi đơn giản và trực quan[3].

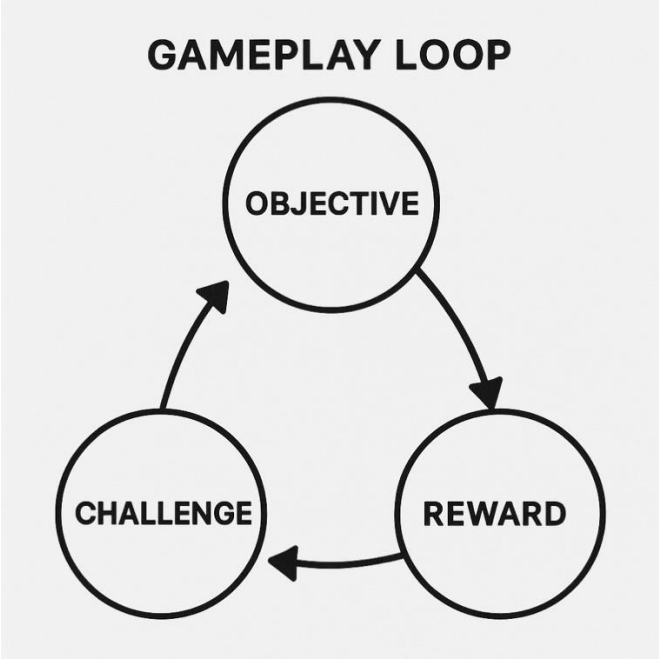
Thứ ba, game casual thường được thiết kế theo hướng linh hoạt và dễ mở rộng nội dung.

Cuối cùng, game casual cũng thường chú trọng vào tính phù hợp đại chúng (mass appeal).

Những đặc điểm này không chỉ giải thích vì sao game casual phổ biến mà còn tạo nền tảng cho quá trình thiết kế nhiệm vụ và ứng dụng LLM vào việc sinh nội dung phù hợp với hành vi người chơi, sao cho đáp ứng và tăng cường thế mạnh của thể loại game này.

1.1.2 Tâm lý học hành vi người chơi trong game casual

Các công trình nghiên cứu về game casual nhấn mạnh core loop có thể mô tả như sau:



Hình 1-2 Sơ đồ vòng lặp game tác động tới hành vi người chơi

Trong đó:

(i) Objective (Mục tiêu): Việc người chơi cần hoàn thành trong mỗi lượt, thường đơn giản và rõ ràng.

(ii) Challenge (Thử thách): Trở ngại hoặc mức độ khó khiến người chơi phải nỗ lực để đạt mục tiêu.

(iii) Reward (Phần thưởng): Kết quả tích cực người chơi nhận được sau khi vượt qua thử thách, thúc đẩy họ tiếp tục vòng lặp tiếp theo.

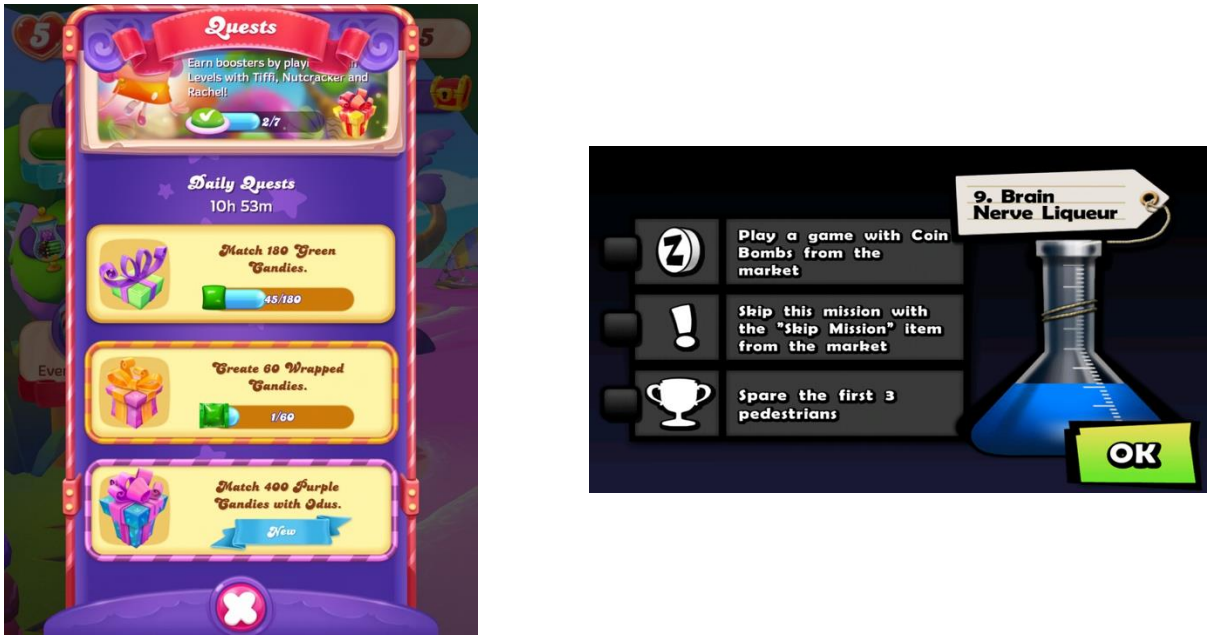
Việc thúc đẩy người chơi thông qua hệ thống nhiệm vụ góp phần tăng sức hấp dẫn cho gameplay cốt lõi bằng cách tạo những vòng lặp dopamine nhỏ nhưng liên tục này là nền tảng của trò chơi casual thành công.

1.1.3 Cơ sở thiết kế và phân loại nhiệm vụ trong game casual

Heej thống nhiệm vụ (Quest System) đóng vai trò là cơ chế then chốt để duy trì vòng đời người chơi (User Lifecycle) và tối ưu hóa chỉ số giữ chân (Retention). Nó tác động trực tiếp đến người chơi thông qua ba cơ chế chính: (1) Định hướng hành vi (Behavior Shaping) – dẫn dắt người chơi thực hiện các thao tác mong muốn; (2) Thiết lập mục tiêu ngắn hạn (Short-term Goal Setting) – tạo động lực tức thời trong các phiên chơi ngắn; và (3) Kích thích sự quay lại (Return Incentive) – tạo thói quen đăng nhập hằng ngày thông qua phần thưởng.

Hệ thống nhiệm vụ thường được phân loại thành ba nhóm chức năng nền tảng, bao quát toàn bộ trải nghiệm của người chơi gồm: nhóm nhiệm vụ định hướng hành động (Action-oriented), nhóm nhiệm vụ ghi nhận tiến trình (Progression-oriented) và nhóm nhiệm vụ điều tiết tài nguyên (Resource-spending).

Cấu trúc phân loại này hoàn toàn tương thích với đặc thù vận hành của dòng game casual nói chung và tựa game Merge Fruit nói riêng và là cơ sở quan trọng để xây dựng dữ liệu huấn luyện cho mô hình ngôn ngữ lớn ở các chương sau.



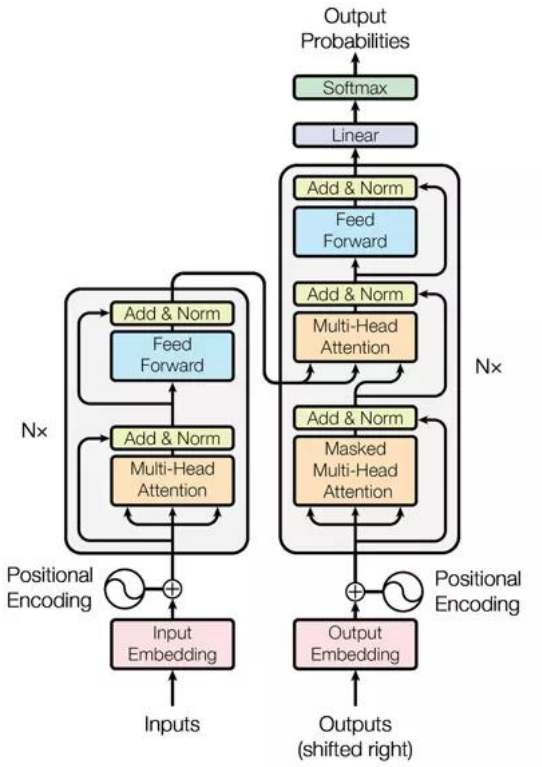
Hình 1-3 Hệ thống nhiệm vụ trong game casual

1.2 MÔ HÌNH NGÔN NGỮ LỚN (LLM)

1.2.1 Kiến trúc Transformer và mô hình Decoder-only

Nền tảng của hầu hết các mô hình ngôn ngữ lớn hiện nay là kiến trúc Transformer, được giới thiệu lần đầu bởi Vaswani et al. (2017) trong công trình "Attention Is All You Need" [5]. Khác với các mạng nơ-ron hồi quy (RNN) hay LSTM truyền thống vốn xử lý dữ

liệu tuần tự, Transformer loại bỏ hoàn toàn các kết nối hồi quy và thay thế bằng cơ chế chú ý (attention mechanism), cho phép xử lý dữ liệu song song và nắm bắt các phụ thuộc xa trong chuỗi văn bản hiệu quả hơn đáng kể [6].



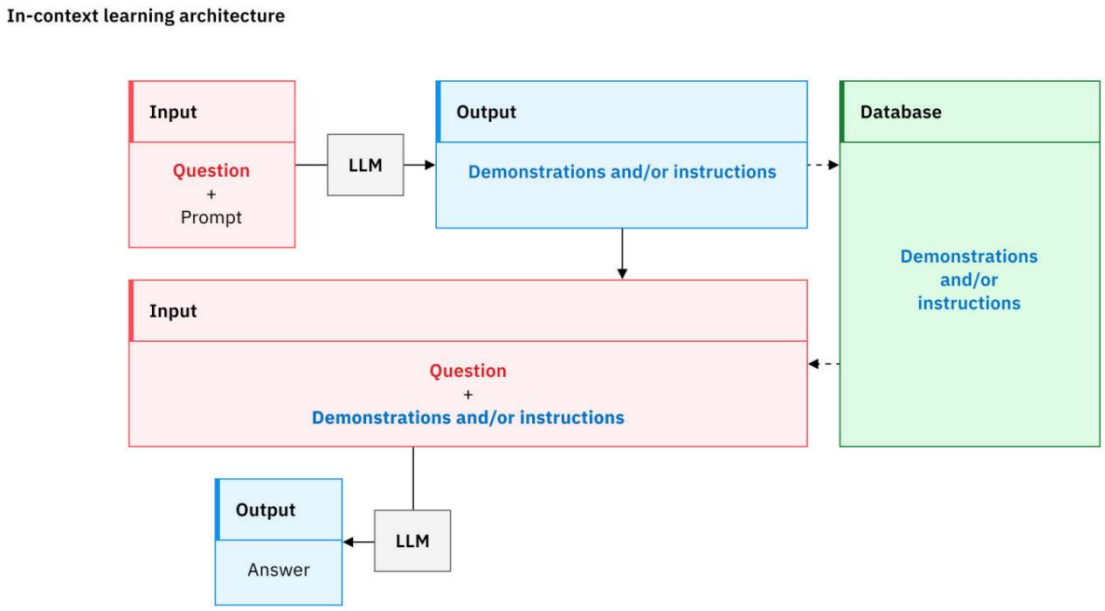
Hình 1-4 Kiến trúc mô hình transformer

1.2.2 Cơ chế Self-Attention và Cửa sổ ngữ cảnh

Sức mạnh cốt lõi giúp LLM duy trì được logic của trò chơi và tuân thủ các cấu trúc dữ liệu phức tạp nằm ở cơ chế Self-Attention. Trong ngữ cảnh của đề tài, cơ chế này đóng vai trò quyết định trong việc xử lý "Cửa sổ ngữ cảnh" (Context Window), cho phép thực hiện chiến lược "Nạp ngữ cảnh" (Context Injection), nơi toàn bộ bộ tham chiếu gameplay và luật chơi được nạp vào bộ nhớ ngắn hạn của mô hình để định hướng quá trình sinh nhiệm vụ.

1.2.3 Cơ chế học trong ngữ cảnh (In-context Learning)

Khả năng thực hiện nhiệm vụ mới mà không cần cập nhật trọng số mạng nơ-ron (gradient updates) của LLM được gọi là Học trong ngữ cảnh (In-context Learning - ICL). Các mô hình như GPT-3 thể hiện năng lực học tập "few-shot" mạnh mẽ: chỉ cần cung cấp một vài ví dụ minh họa hoặc chỉ dẫn tự nhiên trong ngữ cảnh đầu vào, mô hình có thể suy luận và thực hiện tác vụ tương ứng với độ chính xác cao [14].



Hình 1-5 Kiến trúc In-context learning architecture

Cơ chế này đặt nền móng lý thuyết cho việc lựa chọn giải pháp Prompt Engineering thay vì Tinh chỉnh mô hình (Fine-tuning). In-context Learning cho phép hệ thống game tích hợp và điều chỉnh logic sinh nhiệm vụ một cách linh hoạt, tiết kiệm chi phí và phù hợp với quy trình phát triển nhanh (agile) của các dự án game casual, nơi luật chơi và cân bằng số liệu thay đổi thường xuyên.

1.2.4 Cơ chế tuân thủ chỉ dẫn (Instruction Following & Alignment)

Cơ chế tuân thủ chỉ dẫn có vai trò đặc biệt quan trọng trong các ứng dụng yêu cầu đầu ra định dạng nghiêm ngặt, như sinh mã, sinh dữ liệu JSON, hoặc xử lý biểu mẫu. Trong khuôn khổ đề tài này, nó là nền tảng để triển khai các ràng buộc kỹ thuật trong prompt. Cụ thể, nó cho phép mô hình hiểu và thực thi các quy định như: chỉ trả về chuỗi JSON hợp lệ theo đúng schema đã định; không tạo ra giá trị nằm ngoài giới hạn enum; và loại trừ các thành phần không tồn tại trong gameplay (ví dụ: không sinh quái vật vượt quá cấp độ 11).

Nếu thiếu cơ chế này, LLM sẽ có xu hướng tiếp tục sinh văn bản tự do (freeform text) thay vì bám sát cấu trúc dữ liệu JSON, dẫn đến việc không thể tích hợp vào các hệ thống phần mềm tự động hóa.

1.2.5 Kỹ thuật Prompt Engineering

Prompt Engineering được hiểu là một phương pháp tiếp cận kỹ thuật nhằm tối ưu hóa nội dung đầu vào để điều khiển hành vi của mô hình ngôn ngữ lớn (LLM), đóng vai trò như một giải pháp huấn luyện phi tham số trong các hệ thống trí tuệ nhân tạo. Trong phạm vi đề

tài này, một hệ thống các kỹ thuật Prompt Engineering đã được áp dụng để tăng cường khả năng tuân thủ định dạng và logic gameplay của mô hình là Instruction Prompting, Schema-Constrained Prompting, Enum-Constrained, Rule-Based Contextualization liên quan đến việc nạp các bộ quy tắc vận hành (Operational Rulesets), Negative Prompting (Gợi ý phủ định), Few-shot Prompting.

Tổng hợp lại, các cơ sở lý thuyết từ kiến trúc Transformer, cơ chế Attention, In-context Learning đến Instruction Following tạo thành khung tham chiếu vững chắc, chứng minh tính khả thi và hiệu quả của việc sử dụng LLM kết hợp với Prompt Engineering để giải quyết bài toán sinh nội dung tự động trong trò chơi điện tử.

1.3 VẤN ĐỀ NGHIÊN CỨU VÀ NHU CẦU ỨNG DỤNG LLM

Trong các trò chơi thuộc thể loại casual, hệ thống nhiệm vụ (quest) đóng vai trò trung tâm trong việc giữ chân người chơi, khuyến khích hành vi tích cực và tạo động lực quay lại trò chơi mỗi ngày. Tuy nhiên, quá trình thiết kế các nhiệm vụ này hiện vẫn đang được thực hiện chủ yếu bằng phương pháp thủ công, dẫn đến nhiều hạn chế đáng kể như khó mở rộng quy mô, nội dung dễ bị lặp lại, và tốn kém chi phí vận hành do phụ thuộc vào nhân lực sáng tạo nội dung.

Sự xuất hiện của các mô hình ngôn ngữ lớn (LLM) mở ra cơ hội đột phá cho việc tự động hóa quy trình thiết kế nhiệm vụ trong game. Với khả năng sinh văn bản mượt mà, đa dạng và giàu ngữ cảnh, LLM có tiềm năng tạo ra các nhiệm vụ phong phú, linh hoạt mà vẫn đảm bảo phù hợp với logic gameplay.

Toàn bộ phần còn lại của luận văn sẽ tập trung vào việc thiết kế, huấn luyện và kiểm định một pipeline sinh nhiệm vụ tự động dựa trên LLM, sử dụng các kỹ thuật Prompt Engineering và cơ chế xác thực nhiều lớp nhằm đảm bảo chất lượng đầu ra phù hợp với yêu cầu của game mobile hiện đại.

CHƯƠNG 2.CHUẨN BỊ DỮ LIỆU VÀ HUẤN LUYỆN MÔ HÌNH

Chương này đóng vai trò nền tảng kỹ thuật cho toàn bộ đề tài, đặt tiền đề cho việc kiểm thử và đánh giá kết quả ở chương tiếp theo. Các nội dung được trình bày lần lượt từ phân tích dữ liệu đầu vào (mục 2.2), đến lựa chọn mô hình phù hợp (2.3), thiết lập quy trình huấn luyện (2.4), và thiết kế module kiểm định đầu ra (2.5).

2.1 CHUẨN BỊ DỮ LIỆU

Nội dung này trình bày phương pháp phân tích và chuẩn bị dữ liệu cho quá trình Prompt Engineering. Dữ liệu được chuẩn bị đóng vai trò chỉ dẫn mô hình LLM trả về kết quả theo yêu cầu.

2.1.1 Quy trình thu thập dữ liệu

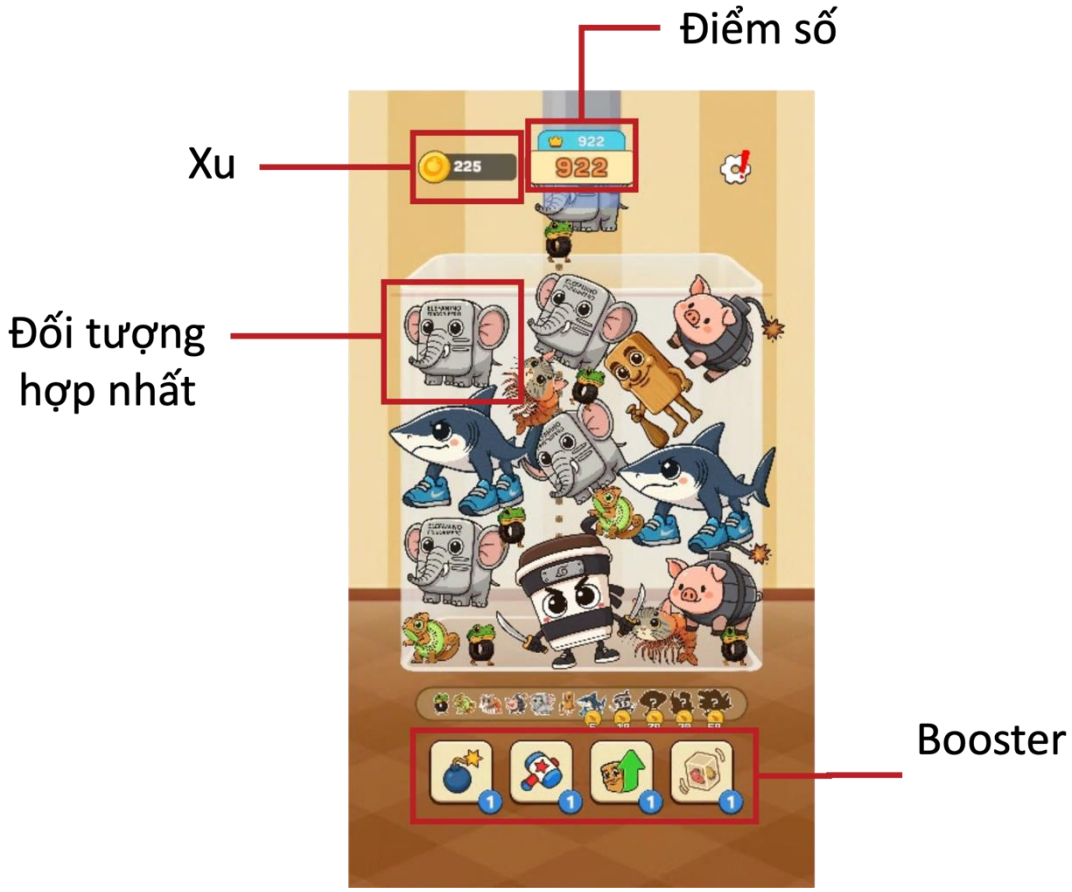
Quy trình thu thập dữ liệu được xây dựng nhằm tạo lập bộ dữ liệu nền thống nhất phục vụ cho giai đoạn mô hình hóa và sinh nhiệm vụ bằng LLM, bảo đảm rằng dữ liệu phản ánh chính xác logic vận hành của trò chơi, thực hiện qua ba pha chính: Thu thập dữ liệu gameplay, Chuẩn hóa thành Từ điển dữ liệu (Data Dictionary), và Thiết lập bộ quy tắc cân bằng (Balancing Ruleset). Kết quả cuối cùng của quy trình là một tập dữ liệu (Gameplay Reference Set) tinh gọn, nhất quán và sẵn sàng nạp vào pipeline tạo sinh.



Hình 2-1 Sơ đồ quy trình thu thập dữ liệu

a) Thu thập dữ liệu

Dữ liệu gameplay được thu thập thông qua phương pháp phân tích tĩnh mã nguồn Unity (Static Code Analysis) – tập trung vào các file định nghĩa Enum và ScriptableObject – kết hợp với quan sát thực nghiệm và tài liệu thiết kế trò chơi (Game Design Document).



Hình 2-2 Ảnh nội dung trò chơi merge

Ở giai đoạn này, dữ liệu tồn tại ở dạng thô, phản ánh các đặc tính trạng thái và hành động của trò chơi, nhưng chưa có cấu trúc chuẩn hóa cho mô hình LLM. Kết quả thu thập thô này đóng vai trò nền tảng để bước chuẩn hóa ở phần tiếp theo có thể diễn ra hiệu quả và chính xác hơn.

b) Chuẩn hóa dữ liệu

Sau giai đoạn thu thập dữ liệu thô, bước chuẩn hóa được thực hiện nhằm chuyển đổi thông tin sang định dạng cấu trúc hóa, đồng thời triệt tiêu các yếu tố nhập nhằng về mặt ngữ nghĩa. Mục tiêu cốt lõi của giai đoạn này là xây dựng sự thống nhất tuyệt đối giữa ngôn ngữ tự nhiên và logic lập trình thông qua các thao tác xử lý cụ thể: Ánh xạ định danh (Identifier Mapping), Chuẩn hóa thuật ngữ (Terminology Normalization), Phân loại chức năng (Functional Classification), Định nghĩa ranh giới cú pháp (Syntax Boundary Definition).

Kết quả của quy trình này là sự hình thành một Từ điển dữ liệu (Data Dictionary) nhất quán để tham chiếu giúp mô hình ngôn ngữ lớn (LLM) thấu hiểu chính xác các khái niệm chuyên ngành, từ đó ngăn chặn hiệu quả tình trạng trùng lặp hoặc sai lệch định nghĩa trong quá trình sinh nhiệm vụ tự động.

c) Thiết lập bộ quy tắc cân bằng

Hệ thống dữ liệu cân bằng (Balancing Data) được thiết lập dựa trên sự tổng hợp giữa các giới hạn vật lý của game (như kích thước bàn chơi, tốc độ sinh quái) và dữ liệu kiểm thử nội bộ (Playtest Data) để đảm bảo tính cân bằng cho game. Bộ quy tắc gồm: Độ khó thực tế (Actual Difficulty), Tính khả thi theo phiên (Session Feasibility), Tỷ lệ Nỗ lực – Phần thưởng (Effort-to-Reward Ratio), Định giá kinh tế (Economic Valuation). Các quy tắc cân bằng này đồng thời cung cấp nền tảng logic thiết yếu cho mô-đun kiểm định (Validation Layer) ở các bước hậu xử lý.

2.1.2 JSON Schema và yêu cầu ràng buộc

Đây là phương pháp giúp chuẩn hóa cấu trúc dữ liệu và tạo ra khung ràng buộc rõ ràng cho mô hình ngôn ngữ lớn. Cách tiếp cận này phù hợp với kỹ thuật "schema-constrained generation" giúp giảm thiểu lỗi cú pháp, mà còn hạn chế hiện tượng sai lệch ngữ nghĩa, từ đó mô hình tập trung vào các trường thông tin bắt buộc thay vì tạo ra nội dung không liên quan, đồng thời hỗ trợ module kiểm định (Validation Layer) hoạt động hiệu quả.

a) Đặc tả cấu trúc JSON

Cấu trúc nhiệm vụ được định nghĩa xoay quanh 5 trường dữ liệu cốt lõi, được tối ưu hóa cho việc xử lý máy tính:

Bảng 2-1 Mô tả dữ liệu ứng với mỗi nhiệm vụ

Tên trường	**Kiểu dữ liệu**	**Mô tả chức**	**Ràng buộc (Constraints)**
taskName	Integer	Mã định danh loại nhiệm vụ (mapping theo enum)	Giá trị từ 0 đến 24 (Tham chiếu bảng…).
taskDes	String	Mô tả nhiệm vụ hiển thị trên UI	Tiếng Anh, ngắn gọn (tối đa 8 từ), văn phong vui tươi casual
targetValue	Integer	Giá trị mục tiêu cần đạt	Phải > 0 và khả thi trong 1 ngày chơi
rewardDaily	Integer	Loại phần thưởng	0 (Coin) hoặc 1 (Booster).
numReward	Integer	Số lượng phần thưởng	Tuân thủ quy tắc cân bằng (Tham chiếu bảng …).

Các trường dữ liệu (taskName, targetValue, rewardDaily, numReward…) đảm bảo tính rõ ràng – tương tự các hệ thống phân loại nhiệm vụ trong thiết kế game học thuật [**Error! Reference source not found.**], [**Error! Reference source not found.**].

b) Minh hoạ JSON Schema

Dạng Schema tổng quát được sử dụng trong đề tài có thể được mô tả như sau:

```
{
  "dailyTasks": [
    {
      "taskName": <giá trị số trong enum DailyTaskName>,
      "taskDes": "1 câu mô tả ngắn gọn, rõ ràng về nhiệm vụ, tối
đa 8 từ",
      "targetValue": <giá trị mục tiêu cần đạt>,
      "rewardDaily": <giá trị số trong enum RewardDaily>,
      "numReward": <số lượng phần thưởng>
    }
  ]
}
```

Hình 2-3 Cấu trúc JSON Schema của nhiệm vụ

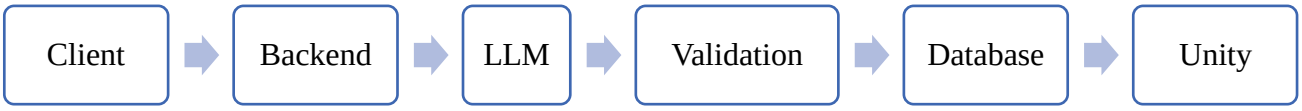
2.1.3 Bộ tham chiếu gameplay (Gameplay Reference Set)

Bộ tham chiếu gameplay (Gameplay Reference Set) là tập hợp tri thức miền (domain knowledge) được trích xuất và chuẩn hóa từ hệ thống trò chơi, dùng làm nền tảng để mô hình ngôn ngữ lớn (LLM) hiểu và sinh ra các nhiệm vụ phù hợp với ngữ cảnh của game Merge Fruit. Trong kỹ thuật Prompt Engineering cho mô hình sinh có ràng buộc, việc cung cấp một tập tri thức tham chiếu rõ ràng giúp tăng độ chính xác, giảm hiện tượng "ảo giác" (hallucination), đồng thời đảm bảo đầu ra tuân thủ các yêu cầu về khả thi, cân bằng và phù hợp với trải nghiệm người chơi. Các bảng tham chiếu trên giúp LLM hiểu đúng môi trường trò chơi và sinh ra các nhiệm vụ hợp lệ cả về cú pháp, ngữ nghĩa lẫn logic gameplay. Đây là lớp dữ liệu quan trọng đảm bảo pipeline sinh nhiệm vụ tự động vận hành ổn định, chính xác và phù hợp với cấu trúc thiết kế của game.

2.2 KIẾN TRÚC HỆ THỐNG & LỰA CHỌN MÔ HÌNH

2.2.1 Sơ đồ kiến trúc tổng thể

Hệ thống được thiết kế theo mô hình Client-Server kết hợp với kiến trúc Pipeline xử lý dữ liệu, nhằm đảm bảo tính ổn định, khả năng mở rộng (Scalability) và dễ dàng tích hợp vào Game Engine Unity. Pipeline hệ thống được xây dựng theo chu trình:



Hình 2-4 Sơ đồ pipeline hệ thống tích hợp LLM vào Unity Engine

Cách thiết kế này phù hợp với các khuyến nghị triển khai hệ thống sinh nội dung bằng AI trong môi trường sản xuất (production-level content generation), được đề cập trong nhiều khảo sát về kiến trúc mô hình Transformer [**Error! Reference source not found.**].

2.2.2 Lựa chọn mô hình ngôn ngữ

Việc lựa chọn mô hình ngôn ngữ dựa trên ba tiêu chí cốt lõi: năng lực sinh nội dung, tuân thủ ràng buộc cấu trúc và hiệu quả triển khai hệ thống. Lựa chọn mô hình Gemini 2.5 Flash làm mô hình thực nghiệm chính. Cách tiếp cận sử dụng few-shot prompting kết hợp với Validation Layer, thay vì fine-tuning mô hình, giúp đảm bảo sự cân bằng giữa chất lượng sinh nhiệm vụ, tính linh hoạt trong triển khai và khả năng kiểm soát chi phí. Điều này cũng phù hợp với các kết luận trong nghiên cứu của Brown et al. (2020) về hiệu quả cao của phương pháp Few-shot và Zero-shot trong các hệ thống ứng dụng thực tế sử dụng LLM [14].

2.2.3 So sánh chi phí–hiệu năng

Trong thị trường game casual cạnh tranh, việc lựa chọn mô hình ngôn ngữ không chỉ dựa trên chất lượng mà còn phụ thuộc vào hiệu quả kinh tế, theo báo cáo thị trường của Newzoo và Sensor Tower [15], [16]. Sử dụng Gemini 2.5 Flash qua API mang lại lợi thế vượt trội so với phương pháp tinh chỉnh (fine-tuning) hoặc tự triển khai mô hình nội bộ nhờ lợi thế của nó. Do đó, đây là điểm cân bằng lý tưởng giữa chi phí và hiệu năng, phù hợp với xu hướng ứng dụng LLM hiện đại được Brown et al. (2020) khuyến nghị.

2.3 QUY TRÌNH HUẤN LUYỆN VÀ TỐI ƯU

2.3.1 Quy trình tối ưu hóa Prompt (Prompt Optimization Pipeline)

Quy trình xây dựng prompt không được thực hiện đơn lẻ mà tuân theo phương pháp tối ưu hóa lặp (Iterative Refinement Process) dựa trên vòng phản hồi thực nghiệm. Chiến lược này được chia thành ba giai đoạn phát triển kế tiếp nhau, tương ứng với mức độ kiểm soát ngày càng chặt chẽ:

1. Giai đoạn Khởi tạo (Baseline - Zero-shot): Đánh giá năng lực tự nhiên của mô hình khi chưa có ràng buộc (Prompt V1).

2. Giai đoạn Cấu trúc hóa (Structural Alignment): Tích hợp JSON Schema và Enum để chuẩn hóa định dạng đầu ra (Prompt V2).
3. Giai đoạn Ngữ cảnh hóa toàn diện (Contextual Integration): Nạp tri thức gameplay và áp dụng các kỹ thuật prompt để hoàn thiện logic thiết kế (Prompt V3).

Mỗi giai đoạn đều được đánh giá dựa trên các lỗi phát sinh thực tế để làm cơ sở cho các cải tiến ở bước tiếp theo, đảm bảo kết quả cuối cùng đạt độ chính xác cao nhất về cả cú pháp lẫn ngữ nghĩa.

2.3.2 Prompt V1: Phiên bản khởi đầu (Baseline Prompt)

Prompt V1 là bước thử nghiệm ban đầu nhằm đánh giá khả năng tạo sinh nhiệm vụ của mô hình khi không có bất kỳ ràng buộc kỹ thuật hoặc ngữ cảnh hỗ trợ nào. Phiên bản này chỉ sử dụng một yêu cầu khái quát.

2.3.3 Prompt V2: Bổ sung ràng buộc cấu trúc (Structure-Constrained Prompt)

Dựa trên phân tích các lỗi cú pháp từ phiên bản khởi đầu, Prompt V2 được phát triển với trọng tâm là kỹ thuật Schema-Constrained Prompting được đề xuất trong Lin et al. (2022) [6]. Mục tiêu của phiên bản này là chuyển đổi đầu ra từ văn bản tự do sang dữ liệu có cấu trúc định sẵn, đảm bảo khả năng tích hợp hệ thống. Các cải tiến cốt lõi giúp cải thiện đầu ra của mô hình.

2.3.4 Prompt V3: Tối ưu hoá ngữ cảnh (Final Prompt)

Prompt V3 đại diện cho bước cải tiến cuối cùng trong chuỗi tối ưu hóa prompt, đánh dấu sự chuyển đổi từ một mô hình ngôn ngữ tổng quát sang một tác tử chuyên biệt cho thiết kế nhiệm vụ trong trò chơi (Game Design Agent). Cấu trúc của Prompt V3 không chỉ là sự cải tiến về mặt cú pháp, mà còn là kết quả của quá trình tích hợp đa tầng các cơ chế lý thuyết đã trình bày tại Chương 1. Thiết kế này kết hợp chặt chẽ giữa kỹ thuật kiểm soát đầu ra, ràng buộc hệ thống và duy trì tri thức nghiệp vụ, hướng đến một hệ thống sinh nhiệm vụ ổn định, chính xác và nhất quán.

2.4 MODULE KIỂM ĐỊNH (VALIDATION LAYER)

Trong kiến trúc hệ thống đề xuất, Module Kiểm định (Validation Layer) đóng vai trò là "hàng rào kiểm soát chất lượng" (Quality Gate) cuối cùng trước khi dữ liệu được chuyển đến người chơi. Dựa trên tính chất xác suất (probabilistic nature) của các mô hình ngôn ngữ lớn, việc tồn tại một tỷ lệ lỗi nhất định trong đầu ra là điều khó tránh khỏi, ngay cả khi đã áp dụng các kỹ thuật Prompt Engineering tối ưu [6], [14]. Do đó, việc thiết lập một lớp kiểm

định độc lập, vận hành theo cơ chế tất định (deterministic), là yêu cầu thiết yếu để đảm bảo sự ổn định và an toàn cho hệ thống trò chơi. Module này được thiết kế dựa trên mô hình kiểm soát ba tầng, tương ứng với ba cấp độ toàn vẹn của dữ liệu: Cấu trúc, Ngữ nghĩa và Nghiệp vụ.

2.4.1 Kiểm định cấu trúc (Structural Validation)

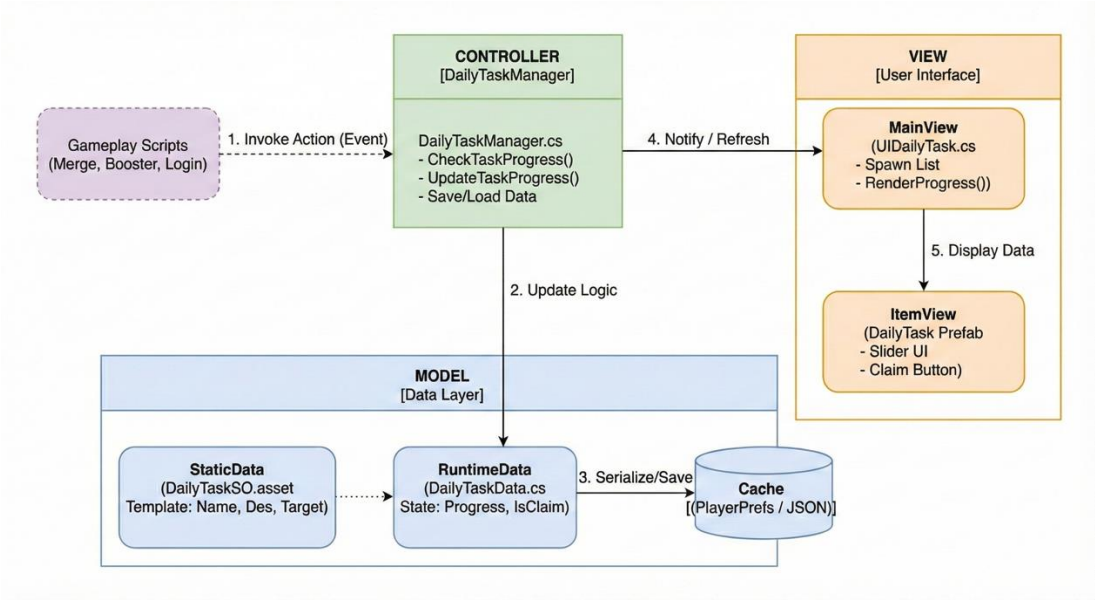
2.4.2 Kiểm định logic gameplay (Gameplay Logic Validation)

2.4.3 Kiểm định khả thi và cân bằng thiết kế (Gameplay Feasibility & Balancing)

2.5 TÍCH HỢP THỰC NGHIỆM VÀO GAME

2.5.1 Thiết kế và triển khai Quest System theo mô hình

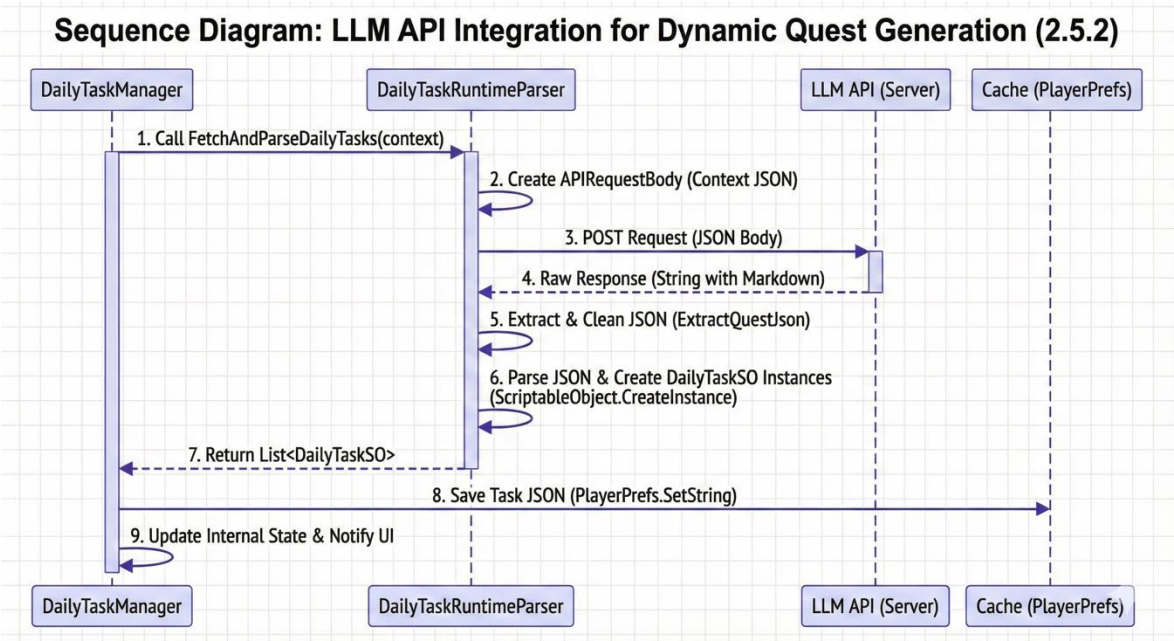
Hệ thống nhiệm vụ (Quest System) được kiến trúc và triển khai dựa trên mô hình Model-View-Controller (MVC) kết hợp với Observer Pattern (Mẫu quan sát viên) nhằm đảm bảo sự tách biệt rõ ràng giữa dữ liệu, giao diện và logic xử lý, đồng thời tối ưu hóa khả năng mở rộng.



Hình 2-5 Sơ đồ mô hình MVC triển khai Quest System

2.5.2 Tích hợp API tạo Quest tự động bằng LLM

Việc tích hợp trí tuệ nhân tạo vào quy trình tạo nhiệm vụ được thực hiện thông qua module trung gian DailyTaskRuntimeParser, hoạt động như một lớp dịch vụ (Service Layer) giúp kết nối Game Client với hệ thống Large Language Model (LLM). Quy trình này giải quyết bài toán tạo nội dung động (Procedural Content Generation) mà không cần can thiệp cứng vào mã nguồn.



Hình 2-6 Biểu đồ luồng hoạt động tích hợp LLM cho trò chơi

2.5.3 Xây dựng giao diện Quest System trong game



Hình 2-7 Giao diện Quest System trong game

Xây dựng giao diện Quest System hiển thị tên câu hỏi, phần thưởng, tiến trình, các nút bấm nhận thưởng, trạng thái đã nhân và nút reset câu hỏi để cập nhật bộ câu hỏi mới qua API.

CHƯƠNG 3.KIỂM THỬ VÀ KHẢO SÁT NGƯỜI DÙNG

Nội dung chương được cấu trúc theo quy trình đánh giá toàn diện từ máy đến người dùng. Đầu tiên, môi trường thực nghiệm được thiết lập để đo lường hiệu năng kỹ thuật của các phiên bản Prompt (V1, V2, V3) dựa trên các chỉ số định lượng về cấu trúc dữ liệu và tính logic. Tiếp theo, nghiên cứu tiến hành khảo sát người dùng (User Acceptance Testing) để đánh giá chất lượng nội dung dưới góc độ tâm lý học hành vi người chơi. Cuối cùng, bài toán hiệu quả kinh tế được phân tích thông qua việc so sánh năng suất và chi phí giữa quy trình tự động hóa bằng AI và quy trình thủ công truyền thống, từ đó khẳng định tính khả thi trong việc triển khai hệ thống vào môi trường sản xuất thực tế.

3.1 MÔI TRƯỜNG THỰC NGHIỆM

3.1.1 Mục tiêu thực nghiệm

Mục tiêu cốt lõi của quá trình thực nghiệm là kiểm chứng tính hiệu quả của quy trình tối ưu hóa Prompt (Prompt Optimization Pipeline) đã đề xuất tại Chương 2.

3.1.2 Cấu hình thực nghiệm

Sử dụng mô hình Gemini 2.5 Flash (Google) thông qua API và hệ thống Client được phát triển trên nền tảng Unity Engine (phiên bản Unity 6000.1.17f1), tích hợp Module Kiểm định (Validation Layer) được viết bằng C#. Thực hiện tổng cộng 150 lượt sinh dữ liệu (Generation Iterations), được chia đều thành 3 tập mẫu tương ứng với 50 lượt cho mỗi phiên bản Prompt (V1, V2, V3).

3.1.3 Công cụ đo lường (Evaluation Metrics)

Hiệu quả của mô hình được đánh giá thông qua hệ thống thước đo kép, bao gồm các chỉ số kỹ thuật định lượng và chỉ số trải nghiệm định tính thông qua Nhóm chỉ số kỹ thuật (Technical Metrics) và Nhóm chỉ số trải nghiệm người dùng (User Experience Metrics)

3.2 KẾT QUẢ SINH NHIỆM VỤ TỪ MÔ HÌNH

Các dữ liệu được phân tích dựa trên sự so sánh đối chiếu giữa năng lực cơ sở (Baseline) và năng lực sau khi tối ưu hóa, nhằm làm rõ tác động của từng kỹ thuật Prompt Engineering đối với chất lượng đầu ra.

3.2.1 Hiệu quả của các phiên bản prompt

Quá trình thực nghiệm được chia thành ba giai đoạn tương ứng với ba cấp độ can thiệp kỹ thuật.

Bảng 3-1 Bảng so sánh 3 cấp độ prompt

Đặc điểm	Prompt V1	Prompt V2	Prompt V3 (Final)
Kỹ thuật	Zero-shot	Schema-Constrained + Enum Mapping	Context Injection + Few-shot + Ruleset
Đầu ra JSON	Lỗi cú pháp thường xuyên	Chính xác cấu trúc 100%	Chính xác cấu trúc 100%
Logic Game	Ảo giác (sinh item lạ)	Sai lệch (Level > 11, thưởng sai)	Chính xác, cân bằng, có Flavor Text
Tính khả thi	Thấp	Trung bình	Cao (Sẵn sàng cho Production)

3.2.2 Thống kê định lượng

Để đánh giá chính xác hiệu năng kỹ thuật, đề tài tiến hành đo lường trên tập mẫu 150 lượt sinh dữ liệu (50 lượt/phiên bản). Kết quả được tổng hợp qua ba chỉ số hiệu suất chính (KPIs):

Bảng 3-2 Bảng tổng hợp hiệu suất kỹ thuật giữa các phiên bản Prompt

Chỉ số đánh giá	Prompt V1 (Zero-shot)	Prompt V2 (Schema-only)	Prompt V3 (Context-aware)
Tỷ lệ JSON hợp lệ	0%	100%	**100%**
Tỷ lệ Enum hợp lệ	0%	100%	**100%**
Tỷ lệ Logic Game hợp lệ	0%	40%	**96%**

3.2.3 Phân tích các dạng lỗi điển hình

Thông qua việc phân tích sâu các trường hợp thất bại (Failure Modes) trong tập dữ liệu thực nghiệm, nghiên cứu đã nhận diện các nhóm lỗi đặc trưng và cơ chế khắc phục tương ứng của hệ thống đề xuất gồm Ảo giác nội dung (Content Hallucination), Mất cân bằng kinh tế (Economic Imbalance) và Bất khả thi vật lý (Physical Infeasibility).

Tổng hợp lại, kết quả phân tích lỗi khẳng định rằng chỉ khi kết hợp đồng bộ cả ba trụ cột kỹ thuật – Cấu trúc, Ngữ nghĩa và Logic – trong Prompt V3, hệ thống mới đạt được độ tin cậy cần thiết để triển khai trong môi trường sản xuất thực tế.

3.3 KHẢO SÁT NGƯỜI DÙNG

Song song với các đánh giá định lượng về mặt kỹ thuật, việc đo lường mức độ chấp nhận của người dùng (User Acceptance) đóng vai trò then chốt để xác định tính khả thi của hệ thống khi triển khai thực tế. Mục tiêu của phần này là kiểm chứng xem các nhiệm vụ do mô hình AI sinh ra có đáp ứng được các tiêu chuẩn về trải nghiệm người chơi (Player Experience - PX) hay không, đặc biệt là trong bối cảnh game casual nơi yếu tố giữ chân người dùng (retention) phụ thuộc lớn vào sự hài lòng tức thời.

3.3.1 Mục tiêu khảo sát

Khảo sát được thiết kế nhằm đánh giá chất lượng nhiệm vụ dựa trên ba tiêu chí tâm lý học hành vi cốt lõi: Mức độ dễ hiểu (Clarity), Sự phù hợp của độ khó (Difficulty Fit) và Khả năng tạo động lực (Motivation).

3.3.2 Phương pháp khảo sát

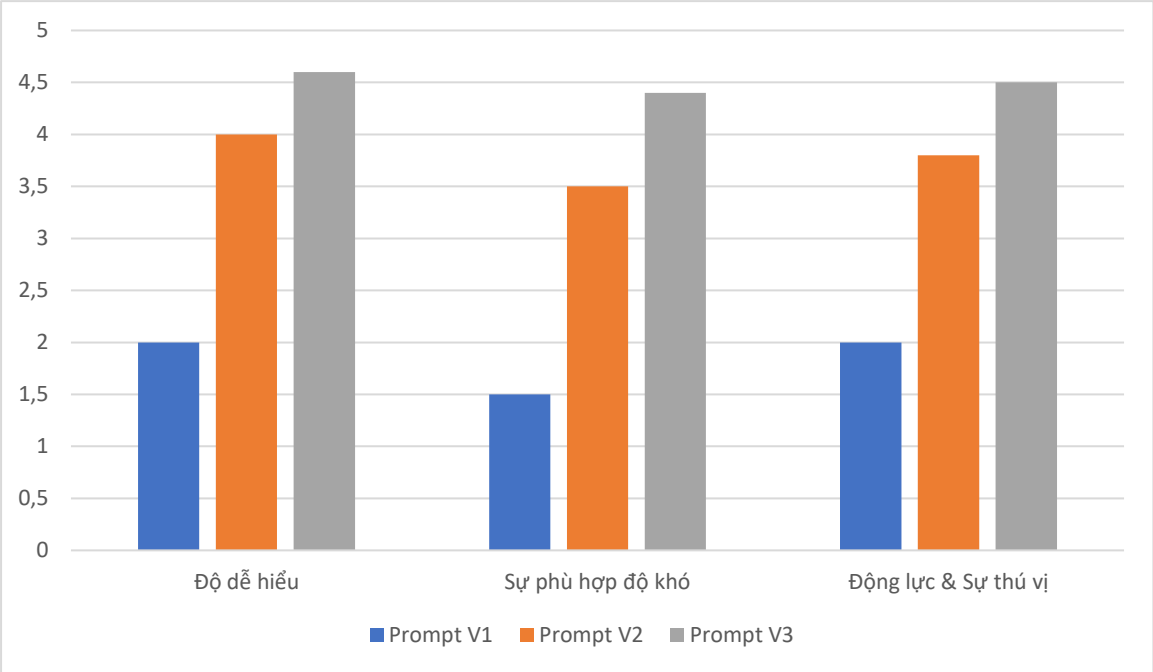
Quá trình khảo sát được thực hiện trên nhóm đối tượng mẫu gồm 20 người tham gia, bao gồm cả người chơi phổ thông và chuyên gia thiết kế game (Game Designers) để đảm bảo góc nhìn đa chiều. Phương pháp đánh giá định lượng được áp dụng thông qua thang đo Likert 5 mức độ (1 - Rất tệ đến 5 - Rất tốt). Các người tham gia được trải nghiệm ngẫu nhiên (Blind Test) các bộ nhiệm vụ được sinh ra bởi Prompt V1, V2 và V3 mà không biết trước nguồn gốc, sau đó chấm điểm dựa trên trải nghiệm thực tế trong môi trường game *Merge Fruit*. Quy trình này tuân thủ các chuẩn mực về Nghiên cứu trải nghiệm người dùng (UX Research) và Phân tích hành vi người chơi thường được áp dụng trong phát triển game casual [1], [2].

3.3.3 Kết quả khảo sát

Kết quả thu thập được cho thấy sự tiến bộ rõ rệt về trải nghiệm người dùng tương ứng với sự hoàn thiện của các phiên bản Prompt.

Bảng 3-3 Bảng tổng hợp kết quả đánh giá trải nghiệm người dùng (Thang đo likert 1-5)

Tiêu chí đánh giá	Prompt V1 (Zero-shot)	Prompt V2 (Schema-only)	Prompt V3 (Final)	Mức độ cải thiện (V1 → V3)
1. Độ dễ hiểu	2.0	4.0	4.6	+130.0%
2. Sự phù hợp độ khó	1.5	3.5	4.4	+193.3%
3. Động lực & Sự thú vị	2.0	3.8	4.5	+125.0%



Hình 3-1 Biểu đồ tương quan kết quả đánh giá trải nghiệm người dùng

3.4 ĐÁNH GIÁ HIỆU QUẢ SẢN XUẤT VÀ CHI PHÍ

Bên cạnh các chỉ số về độ chính xác kỹ thuật và trải nghiệm người dùng, tính hiệu quả trong vận hành (Operational Efficiency) là yếu tố quyết định khả năng ứng dụng của mô hình trong môi trường doanh nghiệp.

3.4.1 So sánh năng suất: Con người vs AI

Thực nghiệm đối chứng được tiến hành trên cùng một khối lượng công việc là tạo ra 10 nhiệm vụ hằng ngày. Kết quả ghi nhận sự chênh lệch đáng kể về thời gian và nguồn lực:

Bảng 3-4 Bảng so sánh hiệu suất giữa con người và AI trong việc tạo nhiệm vụ game (Thang đo likert 1-5)

Tiêu chí so sánh	**Quy trình thủ công** ***(Game Designer)***	**Quy trình tự động hóa** ***(Gemini Flash + Validation)***	**Mức độ tối ưu**
Thời gian hoàn thành	20 - 30 phút	3 – 5 phút	**~ 90%**
Tính đa dạng nội dung	Phụ thuộc vào sức sáng tạo cá nhân (dễ bị lặp lại)	Cao (nhờ tham số Temperature và Over-generation)	**Cao**
Tỷ lệ lỗi kỹ thuật	Thấp (nhưng dễ mắc lỗi nhập liệu typo)	Thấp (sau khi qua lớp Validation Layer)	**Tương đương**
Khả năng mở rộng	Khó (tốn thêm nhân sự nếu cần 1000 task)	Dễ dàng (chỉ cần lặp lại vòng lặp API)	**Vượt trội**

Từ bảng so sánh trên có thể thấy hệ thống AI đề xuất giúp rút ngắn khoảng 85 - 90% tổng thời gian phát triển nội dung. Nếu như con người mất trung bình 2.5 - 3 phút để hoàn thiện một nhiệm vụ từ khâu ý tưởng đến nhập liệu, thì hệ thống tích hợp Gemini Flash chỉ mất chưa đầy 20 giây cho mỗi nhiệm vụ (bao gồm cả thời gian sinh, kiểm định và lọc lỗi). Sự cải thiện năng suất này đặc biệt có ý nghĩa trong bối cảnh thị trường game casual đòi hỏi khối lượng nội dung cập nhật liên tục (Live-ops) để duy trì chỉ số Retention, phù hợp với xu hướng "AI hóa" quy trình sản xuất được ghi nhận trong báo cáo thị trường của Newzoo và Sensor Tower [15], [16].

3.4.2 Phân tích chi phí - hiệu năng (Cost-Performance):

Trong bối cảnh ngành công nghiệp game ngày càng cạnh tranh, việc tối ưu hóa chi phí vận hành (OpEx) là yếu tố sống còn, đặc biệt đối với các studio quy mô vừa và nhỏ (Indie/Mid-core). Việc lựa chọn mô hình Gemini 1.5 Flash cho hệ thống sinh nhiệm vụ không chỉ xuất phát từ yêu cầu kỹ thuật về tốc độ phản hồi mà còn là một quyết định chiến lược về mặt kinh tế. Dòng mô hình này được Google thiết kế tối ưu hóa đặc biệt cho các tác vụ tần suất cao và độ trễ thấp, với mức định giá dựa trên triệu token (Cost per Million Tokens) cạnh tranh bậc nhất trên thị trường hiện nay.

KẾT LUẬN

Kết quả của nghiên cứu

Trong khuôn khổ đề tài ***"Nghiên cứu và ứng dụng LLM cho bài toán tạo nhiệm vụ (Quest) trong game casual"***, tác giả đã xây dựng và triển khai thành công một hệ thống tạo nhiệm vụ tự động, góp phần chuyển đổi quy trình sản xuất nội dung game từ thủ công sang hướng tự động hóa thông minh thông qua việc kết hợp lý thuyết về mô hình ngôn ngữ lớn với các kỹ thuật thiết kế hướng cấu trúc. Qua đây, đề tài đã đạt được những kết quả nổi bật trên cả ba phương diện: lý thuyết, hệ thống và thực nghiệm.

Về mặt lý thuyết và quy trình, đề tài đã đề xuất một quy trình tối ưu hóa Prompt (Prompt Optimization Pipeline) dựa trên cơ chế Attention, In-context Learning và Instruction Following qua việc kết hợp ba trụ cột: Ràng buộc cấu trúc (Schema-Constrained) – Ánh xạ ngữ nghĩa (Semantic Mapping) – Ngữ cảnh hoá dựa trên luật (Rule-based Contextualization). Việc áp dụng các kỹ thuật này giúp kiểm soát được đặc tính ngẫu nhiên của LLM, đồng thời đảm bảo đầu ra luôn tuân thủ định dạng và quy tắc trò chơi.

Về mặt hệ thống, đề tài đã thiết kế kiến trúc tích hợp giữa mô hình Gemini 2.5 Flash và nền tảng Unity thông qua lớp trung gian Dify, mang lại hiệu năng phản hồi nhanh, chi phí thấp và khả năng tích hợp trực tiếp với hệ thống game thực tế. Trong đó, Module Kiểm định (Validation Layer) hoạt động như một hàng rào chất lượng cuối cùng, bảo đảm toàn bộ đầu ra đều hợp lệ về mặt kỹ thuật và logic gameplay.

Về hiệu quả thực nghiệm, kết quả kiểm thử thực tế cho thấy:

(i) Tỷ lệ nhiệm vụ hợp lệ đạt 96% ở phiên bản Prompt V3.
(ii) Năng suất sinh nội dung tăng xấp xỉ 90% so với phương pháp thủ công.
(iii) Mức độ chấp nhận của người chơi đạt trung bình 4.5/5 điểm về độ phù hợp và mức độ động lực.
(iv) Chi phí vận hành được tối ưu đáng kể nhờ sử dụng mô hình Flash có chi phí token thấp.

Những kết quả này không chỉ chứng minh tính khả thi của mô hình đề xuất, mà còn mở ra tiềm năng lớn trong việc ứng dụng LLM vào quy trình thiết kế nội dung trò chơi điện tử.

Kiến nghị đề xuất

Dựa trên quá trình nghiên cứu và thử nghiệm, đề tài đưa ra một số kiến nghị nhằm hỗ trợ các nhà phát triển game – đặc biệt là các studio Indie và Mid-core – trong việc triển khai Generative AI:

Thứ nhất, kết hợp mô hình kiến trúc lai (Hybrid Architecture), không nên hoàn toàn phó mặc cho AI mà cần kết hợp LLM với các lớp kiểm soát logic và cấu trúc (Validation Layer) để bảo đảm độ tin cậy của đầu ra.

Thứ hai, ưu tiên hướng huấn luyện phi tham số, việc đầu tư vào kỹ thuật Prompt Engineering và xây dựng dữ liệu ngữ cảnh có thể tiết kiệm chi phí và tăng tính linh hoạt hơn so với việc tinh chỉnh mô hình riêng (Fine-tuning).

Thứ ba, lựa chọn mô hình phù hợp với tác vụ, đối với các tác vụ có cấu trúc lặp lại, như sinh nhiệm vụ hàng ngày, nên ưu tiên sử dụng các mô hình gọn nhẹ (như Gemini Flash) để tối ưu chi phí và tốc độ xử lý.

Hướng nghiên cứu trong tương lai

Mặc dù hệ thống hiện tại đã xử lý hiệu quả bài toán sinh nhiệm vụ "tĩnh", hướng phát triển tiếp theo sẽ tập trung vào các kỹ thuật "siêu cá nhân hóa" (Hyper-personalization) nhằm nâng cao mức độ thích ứng với người chơi cụ thể. Cụ thể, đề tài có thể phát triển các hướng tiềm năng sau.

Nạp hồ sơ người dùng vào prompt (User Profiling Injection) là một hướng tiềm năng thông qua việc bổ sung thông tin về hành vi người chơi như tần suất chơi, mức độ tiêu dùng, tỷ lệ thắng/thua để sinh nhiệm vụ sát với từng đối tượng. Ví dụ, với người chơi "hardcore", hệ thống có thể sinh các nhiệm vụ có độ khó cao kèm phần thưởng danh hiệu; trong khi đó, người chơi có nguy cơ rời bỏ sẽ nhận nhiệm vụ dễ với phần thưởng hấp dẫn.

Điều chỉnh độ khó động (Dynamic Difficulty Adjustment) bằng cách sử dụng LLM để theo dõi chỉ số kỹ năng người chơi trong thời gian thực cũng là một hướng đem lại ý nghĩa lớn cho nhà phát triển game, từ đó điều chỉnh targetValue phù hợp nhằm duy trì trải nghiệm mượt mà và hưng phấn (flow state).

Tổng kết lại, hệ thống được đề xuất không đơn thuần là một giải pháp kỹ thuật, mà còn đại diện cho tư duy thiết kế game hiện đại: chuyển từ cách làm thủ công sang tiếp cận dựa trên dữ liệu và AI, hướng tới trải nghiệm chơi game được cá nhân hóa cao và được cập nhật liên tục theo hành vi người dùng./.