# Adaptive Dual-System Inference: Tích hợp tư duy nhanh và tư duy chậm trong mô hình AI sử dụng Test Time Scaling


## Tóm tắt

Các mô hình ngôn ngữ lớn (LLMs) hiện đại đang phải đối mặt với thách thức cân bằng giữa khả năng phản hồi nhanh (tư duy nhanh - System 1) và khả năng suy luận sâu (tư duy chậm - System 2). Trong khi các mô hình nền tảng (foundational LLMs) xuất sắc trong việc cung cấp phản hồi nhanh dựa trên kiến thức có sẵn, các mô hình suy luận (reasoning LLMs) lại tập trung vào xử lý các vấn đề phức tạp thông qua quá trình suy luận từng bước. Nghiên cứu này đề xuất Adaptive Dual-System Inference (ADSI), một phương pháp mới kết hợp cả hai loại tư duy trong cùng một mô hình sử dụng test time scaling. ADSI bao gồm cơ chế đánh giá độ phức tạp tự động, chuyển đổi thích ứng giữa hai chế độ tư duy, và phân bổ tài nguyên tính toán dựa trên độ phức tạp của vấn đề. Kết quả thực nghiệm cho thấy ADSI cân bằng hiệu quả giữa tốc độ xử lý và độ chính xác, vượt trội hơn các mô hình cơ sở trong nhiều tác vụ khác nhau. Nghiên cứu này mở ra hướng phát triển mới cho các mô hình AI có khả năng thích ứng linh hoạt với nhiều loại vấn đề, tương tự như cơ chế nhận thức của con người.


## 1. Giới thiệu

### 1.1. Bối cảnh

Sự phát triển của các mô hình ngôn ngữ lớn (LLMs) trong những năm gần đây đã mang lại những tiến bộ đáng kể trong lĩnh vực trí tuệ nhân tạo. Các mô hình như GPT, Claude, và DeepSeek đã chứng minh khả năng xử lý ngôn ngữ tự nhiên ở mức độ gần với con người trong nhiều tác vụ khác nhau. Tuy nhiên, một thách thức lớn vẫn tồn tại: làm thế nào để cân bằng giữa khả năng phản hồi nhanh và khả năng suy luận sâu trong cùng một mô hình.

Trong nhận thức con người, nhà tâm lý học Daniel Kahneman đã mô tả hai hệ thống tư duy: System 1 (tư duy nhanh) - tự động, trực giác và nhanh chóng; và System 2 (tư duy chậm) - có phương pháp, logic và đòi hỏi nỗ lực. Con người có khả năng chuyển đổi linh hoạt giữa hai hệ thống này tùy thuộc vào độ phức tạp của vấn đề. Tuy nhiên, các mô hình AI hiện tại thường chỉ xuất sắc ở một trong hai loại tư duy này.

Các mô hình nền tảng (foundational LLMs) thường xuất sắc trong việc cung cấp phản hồi nhanh dựa trên kiến thức có sẵn, tương tự như System 1 của con người. Ngược lại, các mô hình suy luận (reasoning LLMs) được tối ưu hóa cho việc xử lý các vấn đề phức tạp thông qua quá trình suy luận từng bước, tương tự như System 2. Sự phân chia này dẫn đến tình trạng không hiệu quả: các mô hình suy luận thường tốn nhiều tài nguyên tính toán không cần thiết cho các vấn đề đơn giản, trong khi các mô hình nền tảng lại không đủ khả năng xử lý các vấn đề phức tạp đòi hỏi suy luận sâu.

## 1.2. Vấn đề nghiên cứu

Vấn đề cốt lõi mà nghiên cứu này giải quyết là sự mất cân bằng giữa khả năng tư duy nhanh và tư duy chậm trong các mô hình AI hiện tại. Như đã nêu trong nghiên cứu gần đây về Collaborative Slow & Fast-thinking Systems (Liu et al., 2025), "một thách thức chính trong các mô hình suy luận LLMs là sự mất đi khả năng tư duy nhanh, dẫn đến sự không hiệu quả khi các tác vụ đơn giản đòi hỏi suy luận sâu không cần thiết."

Các nỗ lực hiện tại để giải quyết vấn đề này, như System 1-2 switcher, speculative decoding, và interactive continual learning, vẫn chưa đạt được sự cân bằng lý tưởng. Các phương pháp này thường đòi hỏi sự thay đổi đáng kể về kiến trúc mô hình hoặc quy trình huấn luyện, làm tăng độ phức tạp và chi phí phát triển.

Một hướng tiếp cận đầy hứa hẹn là sử dụng test time scaling (hay inference-time compute scaling) - phương pháp tối ưu hóa quá trình suy luận của mô hình tại thời điểm sử dụng mà không cần thay đổi trọng số của mô hình. Tuy nhiên, các nghiên cứu hiện tại về test time scaling chủ yếu tập trung vào việc cải thiện khả năng suy luận mà chưa đề cập đến việc kết hợp cả hai loại tư duy trong cùng một mô hình.

## 1.3. Mục tiêu nghiên cứu

Nghiên cứu này nhằm phát triển một phương pháp tích hợp tư duy nhanh và tư duy chậm trong cùng một mô hình AI sử dụng test time scaling. Cụ thể, chúng tôi đặt ra các mục tiêu sau:

1. Phân tích các đặc điểm và vai trò của tư duy nhanh và tư duy chậm trong mô hình AI
2. Đánh giá các phương pháp test time scaling hiện có và khả năng ứng dụng trong việc tích hợp hai loại tư duy
3. Đề xuất một phương pháp mới - Adaptive Dual-System Inference (ADSI) - để cân bằng giữa hiệu quả và độ chính xác
4. Đánh giá hiệu suất của ADSI trên nhiều loại tác vụ khác nhau
5. Xác định các hạn chế và hướng nghiên cứu tương lai

### 1.4. Đóng góp chính

Nghiên cứu này có những đóng góp chính sau:

1. **Khung lý thuyết mới**: Chúng tôi phát triển một khung lý thuyết để hiểu và tích hợp tư duy nhanh và tư duy chậm trong mô hình AI, dựa trên lý thuyết hệ thống kép của Kahneman và các nghiên cứu gần đây về mô hình suy luận.

2. **Phương pháp ADSI**: Chúng tôi đề xuất Adaptive Dual-System Inference (ADSI), một phương pháp mới kết hợp cả hai loại tư duy trong cùng một mô hình sử dụng test time scaling. ADSI bao gồm cơ chế đánh giá độ phức tạp tự động, chuyển đổi thích ứng giữa hai chế độ tư duy, và phân bổ tài nguyên tính toán dựa trên độ phức tạp của vấn đề.

3. **Kết quả thực nghiệm**: Chúng tôi cung cấp kết quả thực nghiệm toàn diện về hiệu suất của ADSI trên nhiều loại tác vụ khác nhau, so sánh với các mô hình cơ sở và các phương pháp test time scaling hiện có.

4. **Hướng nghiên cứu mới**: Chúng tôi xác định các hạn chế và đề xuất hướng nghiên cứu tương lai để cải thiện việc tích hợp tư duy nhanh và tư duy chậm trong mô hình AI.

### 1.5. Cấu trúc bài báo

Phần còn lại của bài báo được cấu trúc như sau: Phần 2 trình bày tổng quan tài liệu về tư duy nhanh và tư duy chậm trong nhận thức con người và mô hình AI, cũng như các phương pháp test time scaling hiện có. Phần 3 mô tả phương pháp nghiên cứu, bao gồm phân tích các phương pháp hiện có, đề xuất phương pháp ADSI, và thiết kế thực nghiệm. Phần 4 trình bày kết quả và thảo luận. Phần 5 thảo luận về các hạn chế và hướng nghiên cứu tương lai. Cuối cùng, Phần 6 kết luận bài báo.

## 2. Tổng quan tài liệu

### 2.1. Tư duy nhanh và tư duy chậm trong nhận thức con người

Khái niệm về tư duy nhanh và tư duy chậm được phổ biến rộng rãi thông qua công trình nghiên cứu của Daniel Kahneman, được trình bày trong cuốn sách "Thinking, Fast and Slow" (2011). Kahneman mô tả hai hệ thống tư duy trong nhận thức con người:

#### 2.1.1. System 1 (Tư duy nhanh)

System 1 hoạt động một cách tự động và nhanh chóng, với ít hoặc không có nỗ lực và không có cảm giác kiểm soát có ý thức. Đây là hệ thống trực giác, phản ứng nhanh dựa

trên kinh nghiệm và kiến thức đã được tích lũy. Các đặc điểm chính của System 1 bao gồm:

- Xử lý thông tin song song và tự động
- Phản ứng nhanh chóng với các kích thích
- Dựa vào kinh nghiệm và trực giác
- Tiêu tốn ít năng lượng nhận thức
- Thường xuyên sử dụng các phương pháp tắt (heuristics) để đưa ra quyết định

System 1 rất hiệu quả trong việc xử lý các tình huống quen thuộc và đơn giản, nhưng cũng dễ mắc phải các lỗi hệ thống và thiên kiến nhận thức.

#### 2.1.2. System 2 (Tư duy chậm)

System 2 phân bổ sự chú ý cho các hoạt động tinh thần đòi hỏi nỗ lực, bao gồm các tính toán phức tạp. Hoạt động của System 2 thường liên quan đến trải nghiệm chủ quan về tác nhân, lựa chọn và tập trung. Các đặc điểm chính của System 2 bao gồm:

- Xử lý thông tin tuần tự và có kiểm soát
- Phản ứng chậm hơn nhưng cẩn thận hơn
- Dựa vào logic và suy luận
- Tiêu tốn nhiều năng lượng nhận thức
- Có khả năng xử lý các vấn đề phức tạp và trừu tượng

System 2 được kích hoạt khi chúng ta đối mặt với các tình huống mới, phức tạp hoặc đòi hỏi sự tập trung cao độ.

#### 2.1.3. Cơ chế chuyển đổi giữa hai hệ thống

Một khía cạnh quan trọng trong nhận thức con người là khả năng chuyển đổi linh hoạt giữa hai hệ thống tư duy. Quá trình chuyển đổi này thường được kích hoạt bởi:

- **Nhận thức về độ phức tạp**: Khi System 1 nhận ra một vấn đề quá phức tạp hoặc không quen thuộc, nó sẽ kích hoạt System 2.
- **Phát hiện mâu thuẫn**: Khi System 1 phát hiện kết quả mâu thuẫn hoặc không hợp lý, System 2 được kích hoạt để kiểm tra lại.
- **Nhu cầu chính xác**: Khi tình huống đòi hỏi độ chính xác cao, System 2 được ưu tiên sử dụng.
- **Tài nguyên nhận thức**: Sự sẵn có của tài nguyên nhận thức (như sự tập trung, năng lượng tinh thần) ảnh hưởng đến việc sử dụng System 2.

Sự chuyển đổi linh hoạt này giúp con người cân bằng giữa hiệu quả và độ chính xác trong quá trình ra quyết định, một khả năng mà các mô hình AI hiện tại vẫn đang phấn đấu để đạt được.

## 2.2. Tư duy nhanh và tư duy chậm trong mô hình AI

Trong lĩnh vực AI, đặc biệt là các mô hình ngôn ngữ lớn (LLMs), khái niệm về tư duy nhanh và tư duy chậm đã được áp dụng để mô tả các phương pháp xử lý thông tin khác nhau. Tuy nhiên, không giống như con người có thể chuyển đổi linh hoạt giữa hai hệ thống, các mô hình AI hiện tại thường chỉ xuất sắc ở một trong hai loại tư duy.

### 2.2.1. Foundational LLMs: Đặc điểm, ưu điểm và hạn chế

Các mô hình nền tảng (foundational LLMs) như GPT-3, GPT-4, Claude, và DeepSeek V3 được huấn luyện trên lượng dữ liệu văn bản khổng lồ và tối ưu hóa cho khả năng phản hồi nhanh dựa trên kiến thức có sẵn. Những mô hình này thể hiện nhiều đặc điểm tương tự như System 1 trong nhận thức con người:

**Đặc điểm:** - Phản hồi nhanh chóng dựa trên mẫu đã học - Xử lý thông tin song song thông qua cơ chế attention - Dựa vào kiến thức đã được mã hóa trong trọng số mô hình - Tạo ra phản hồi trực tiếp mà không hiển thị quá trình suy luận

**Ưu điểm:** - Hiệu quả về thời gian và tài nguyên tính toán - Phù hợp với các tác vụ đơn giản hoặc quen thuộc - Khả năng xử lý ngôn ngữ tự nhiên tốt - Tận dụng hiệu quả kiến thức đã được học

**Hạn chế:** - Khó khăn trong việc xử lý các vấn đề phức tạp đòi hỏi suy luận nhiều bước - Dễ mắc lỗi khi đối mặt với các tác vụ đòi hỏi logic chặt chẽ - Thiếu khả năng giải thích quá trình suy luận - Khó khăn trong việc xử lý các tác vụ toán học hoặc lập trình phức tạp

### 2.2.2. Reasoning LLMs: Đặc điểm, ưu điểm và hạn chế

Các mô hình suy luận (reasoning LLMs) như DeepSeek R1, Claude Opus, và GPT-4o được tối ưu hóa cho việc xử lý các vấn đề phức tạp thông qua quá trình suy luận từng bước. Những mô hình này thể hiện nhiều đặc điểm tương tự như System 2:

**Đặc điểm:** - Tạo ra các bước suy luận trung gian rõ ràng - Xử lý thông tin tuần tự và có cấu trúc - Tích hợp các kỹ thuật như chain-of-thought và self-reflection - Tạo ra phản hồi dài hơn với nhiều bước giải thích

**Ưu điểm:** - Khả năng xử lý các vấn đề phức tạp đòi hỏi suy luận nhiều bước - Độ chính xác cao hơn trong các tác vụ logic và toán học - Khả năng giải thích quá trình suy luận - Khả năng tự kiểm tra và sửa lỗi

**Hạn chế:** - Tiêu tốn nhiều tài nguyên tính toán hơn - Không hiệu quả cho các tác vụ đơn giản - Tạo ra phản hồi dài dòng không cần thiết cho các câu hỏi đơn giản - Thời gian phản hồi chậm hơn

#### 2.2.3. Thách thức trong việc kết hợp cả hai loại khả năng

Việc kết hợp cả hai loại tư duy trong cùng một mô hình AI đối mặt với nhiều thách thức:

1. **Xung đột trong quá trình huấn luyện**: Tối ưu hóa cho khả năng phản hồi nhanh có thể làm giảm khả năng suy luận sâu, và ngược lại.
2. **Phân bổ tài nguyên tính toán**: Làm thế nào để phân bổ tài nguyên tính toán hiệu quả giữa hai loại tư duy.
3. **Cơ chế chuyển đổi**: Xác định khi nào nên sử dụng tư duy nhanh và khi nào cần chuyển sang tư duy chậm.
4. **Đánh giá độ phức tạp**: Phát triển cơ chế để đánh giá độ phức tạp của vấn đề một cách chính xác.
5. **Tích hợp kiến trúc**: Thiết kế kiến trúc mô hình có thể hỗ trợ cả hai loại tư duy mà không làm tăng đáng kể độ phức tạp.

Các nỗ lực hiện tại để giải quyết những thách thức này bao gồm System 1-2 switcher (Wei et al., 2022), speculative decoding (Leviathan et al., 2023), và interactive continual learning (Sharma et al., 2023). Tuy nhiên, các phương pháp này vẫn chưa đạt được sự cân bằng lý tưởng và thường đòi hỏi sự thay đổi đáng kể về kiến trúc mô hình hoặc quy trình huấn luyện.

### 2.3. Test Time Scaling trong mô hình AI

Test time scaling (hay inference-time compute scaling) là một hướng tiếp cận đầy hứa hẹn để cải thiện khả năng suy luận của mô hình AI mà không cần thay đổi trọng số của mô hình. Phương pháp này tập trung vào việc tối ưu hóa quá trình suy luận tại thời điểm sử dụng, cho phép điều chỉnh lượng tài nguyên tính toán dựa trên độ phức tạp của vấn đề.

#### 2.3.1. Định nghĩa và nguyên lý của test time scaling

Test time scaling là quá trình điều chỉnh lượng tài nguyên tính toán được sử dụng trong quá trình suy luận (inference) của mô hình AI, với mục tiêu cải thiện hiệu suất mà không cần thay đổi trọng số của mô hình. Nguyên lý cơ bản của test time scaling dựa trên quan sát rằng việc tăng lượng tài nguyên tính toán tại thời điểm sử dụng có thể cải thiện đáng kể hiệu suất của mô hình, đặc biệt là trong các tác vụ đòi hỏi suy luận phức tạp.

Raschka (2025) đã phân loại các phương pháp test time scaling thành hai nhóm chính:

1. **Sequential inference scaling**: Tạo ra nhiều token hơn trong một lần hoàn thành, cho phép mô hình "suy nghĩ" lâu hơn.

2. **Parallel inference scaling**: Tạo ra nhiều phản hồi độc lập và kết hợp chúng lại, như trong phương pháp majority voting.

Snell et al. (2024) đã chỉ ra rằng hiệu quả của các phương pháp test time scaling phụ thuộc vào độ khó của vấn đề, và đề xuất chiến lược "compute-optimal" để phân bổ tài nguyên tính toán một cách thích ứng.

### 2.3.2. Các phương pháp test time scaling hiện có

#### 2.3.2.1. Simple test-time scaling với token "Wait"

Phương pháp "s1: Simple test-time scaling" (Raschka, 2025) sử dụng token "Wait" để điều khiển quá trình suy luận của mô hình. Khi token "Wait" được chèn vào quá trình tạo văn bản, mô hình được khuyến khích tạo ra thêm các bước suy luận trung gian và tự sửa lỗi. Phương pháp này có thể được coi là một phiên bản hiện đại của kỹ thuật "think step by step" truyền thống.

#### 2.3.2.2. Compute-Optimal Scaling

Snell et al. (2024) đã đề xuất phương pháp "Compute-Optimal Scaling", tập trung vào việc phân bổ tài nguyên tính toán một cách thích ứng dựa trên độ khó của vấn đề. Phương pháp này phân tích hai cơ chế chính để mở rộng tài nguyên tính toán tại thời điểm sử dụng:

1. Tìm kiếm dựa trên mô hình xác minh (verifier reward models)
2. Cập nhật phân phối của mô hình dựa trên prompt tại thời điểm sử dụng

Kết quả cho thấy chiến lược compute-optimal có thể cải thiện hiệu quả của test-time compute scaling hơn 4 lần so với phương pháp best-of-N cơ bản.

#### 2.3.2.3. Test-Time Preference Optimization

Test-Time Preference Optimization (TPO) là phương pháp điều chỉnh đầu ra của mô hình tại thời điểm sử dụng dựa trên các tiêu chí ưu tiên cụ thể. Phương pháp này sử dụng gradient descent để tối ưu hóa đầu ra của mô hình theo hướng tăng điểm số từ một mô hình đánh giá (reward model).

#### 2.3.2.4. Chain-of-Associated-Thoughts

Chain-of-Associated-Thoughts (CAT) mở rộng phương pháp Chain-of-Thought truyền thống bằng cách khuyến khích mô hình tạo ra nhiều chuỗi suy luận liên kết với nhau. Phương pháp này cho phép mô hình khám phá nhiều hướng suy luận khác nhau và kết hợp chúng lại để đưa ra kết luận cuối cùng.

#### 2.3.2.5. Latent Reasoning

Latent Reasoning là phương pháp sử dụng không gian tiềm ẩn (latent space) để thực hiện quá trình suy luận, thay vì tạo ra các bước suy luận trung gian dưới dạng văn bản. Phương pháp này có thể giảm đáng kể chi phí tính toán trong khi vẫn duy trì khả năng suy luận phức tạp.

### 2.3.3. Ứng dụng của test time scaling trong cải thiện khả năng suy luận

Test time scaling đã được ứng dụng thành công trong nhiều lĩnh vực để cải thiện khả năng suy luận của mô hình AI:

1. **Giải quyết vấn đề toán học**: Các nghiên cứu đã chỉ ra rằng test time scaling có thể cải thiện đáng kể hiệu suất của mô hình trong việc giải quyết các bài toán phức tạp.
2. **Lập trình và tạo mã**: Test time scaling giúp mô hình tạo ra mã nguồn chính xác hơn thông qua quá trình suy luận và tự kiểm tra.
3. **Suy luận đa phương thức**: Các phương pháp test time scaling đã được mở rộng để hỗ trợ suy luận trên dữ liệu đa phương thức, như hình ảnh và văn bản.
4. **Tăng cường độ chính xác**: Test time scaling có thể giúp giảm thiểu các lỗi và tăng cường độ chính xác của mô hình trong các tác vụ đòi hỏi sự chính xác cao.

Tuy nhiên, hầu hết các nghiên cứu hiện tại về test time scaling chủ yếu tập trung vào việc cải thiện khả năng suy luận (tư duy chậm) mà chưa đề cập đến việc kết hợp cả hai loại tư duy trong cùng một mô hình. Đây chính là khoảng trống mà nghiên cứu của chúng tôi hướng đến giải quyết.

# 3. Phương pháp nghiên cứu

## 3.1. Phân tích các phương pháp test time scaling hiện có

Để phát triển một phương pháp tích hợp tư duy nhanh và tư duy chậm hiệu quả, chúng tôi bắt đầu bằng việc phân tích chi tiết các phương pháp test time scaling hiện có, đánh giá ưu điểm, nhược điểm và khả năng ứng dụng của chúng trong việc tích hợp hai loại tư duy.

### 3.1.1. Simple test-time scaling với token "Wait"

Phương pháp "s1: Simple test-time scaling" sử dụng token "Wait" để điều khiển quá trình suy luận của mô hình. Khi phân tích phương pháp này, chúng tôi nhận thấy:

**Ưu điểm:** - Đơn giản, dễ triển khai - Không yêu cầu thay đổi kiến trúc mô hình - Cho phép mô hình tự sửa lỗi và suy nghĩ kỹ hơn khi cần thiết

**Nhược điểm:** - Không có cơ chế tự động để xác định khi nào cần chèn token "Wait" - Không thích ứng với độ phức tạp của vấn đề - Có thể dẫn đến việc tạo ra phản hồi dài dòng không cần thiết cho các vấn đề đơn giản

**Khả năng ứng dụng trong tích hợp tư duy:** Phương pháp này có thể được mở rộng để hỗ trợ việc chuyển đổi từ tư duy nhanh sang tư duy chậm bằng cách phát triển một cơ chế tự động để xác định khi nào cần chèn token "Wait" dựa trên độ phức tạp của vấn đề.

#### 3.1.2. Compute-Optimal Scaling

Phương pháp "Compute-Optimal Scaling" tập trung vào việc phân bổ tài nguyên tính toán một cách thích ứng dựa trên độ khó của vấn đề.

**Ưu điểm:** - Phân bổ tài nguyên tính toán hiệu quả - Thích ứng với độ phức tạp của vấn đề - Hiệu suất cao hơn so với các phương pháp cơ bản

**Nhược điểm:** - Phức tạp về mặt triển khai - Yêu cầu mô hình xác minh (verifier model) bổ sung - Chưa tập trung vào việc tích hợp tư duy nhanh và tư duy chậm

**Khả năng ứng dụng trong tích hợp tư duy:** Phương pháp này cung cấp một khung làm việc hiệu quả để phân bổ tài nguyên tính toán dựa trên độ phức tạp của vấn đề, có thể được áp dụng để quyết định khi nào sử dụng tư duy nhanh và khi nào cần chuyển sang tư duy chậm.

#### 3.1.3. Các phương pháp khác

Chúng tôi cũng phân tích các phương pháp test time scaling khác như Test-Time Preference Optimization, Chain-of-Associated-Thoughts, và Latent Reasoning. Mỗi phương pháp đều có những ưu điểm và hạn chế riêng, nhưng đều chưa tập trung vào việc tích hợp tư duy nhanh và tư duy chậm trong cùng một mô hình.

#### 3.1.4. Kết luận từ phân tích

Từ phân tích các phương pháp hiện có, chúng tôi rút ra một số kết luận quan trọng:

1. Không có phương pháp nào hiện tại tập trung cụ thể vào việc tích hợp tư duy nhanh và tư duy chậm trong cùng một mô hình.
2. Các phương pháp hiện có cung cấp những thành phần hữu ích có thể được kết hợp để phát triển một phương pháp tích hợp hiệu quả.
3. Một phương pháp tích hợp hiệu quả cần có khả năng đánh giá độ phức tạp của vấn đề, chuyển đổi thích ứng giữa hai chế độ tư duy, và phân bổ tài nguyên tính toán một cách hiệu quả.

## 3.2. Đề xuất phương pháp Adaptive Dual-System Inference (ADSI)

Dựa trên phân tích trên, chúng tôi đề xuất Adaptive Dual-System Inference (ADSI), một phương pháp mới kết hợp tư duy nhanh và tư duy chậm trong cùng một mô hình sử dụng test time scaling.

### 3.2.1. Kiến trúc tổng thể của ADSI

ADSI bao gồm năm thành phần chính:

1. **Bộ phân loại độ phức tạp (Complexity Classifier)**: Đánh giá độ phức tạp của vấn đề đầu vào để quyết định mức độ tính toán cần thiết.
2. **Cơ chế chuyển đổi thích ứng (Adaptive Switching Mechanism)**: Quyết định khi nào sử dụng tư duy nhanh và khi nào cần chuyển sang tư duy chậm.
3. **Mô-đun tư duy nhanh (Fast Thinking Module)**: Xử lý các vấn đề đơn giản với tài nguyên tính toán tối thiểu.
4. **Mô-đun tư duy chậm (Slow Thinking Module)**: Xử lý các vấn đề phức tạp với tài nguyên tính toán mở rộng.
5. **Cơ chế phản hồi và hiệu chỉnh (Feedback and Adjustment Mechanism)**: Đánh giá kết quả và điều chỉnh quá trình suy luận nếu cần.

Hình 1 minh họa kiến trúc tổng thể của ADSI.

### 3.2.2. Cơ chế đánh giá độ phức tạp tự động

Bộ phân loại độ phức tạp là một mô hình nhỏ được huấn luyện để dự đoán độ phức tạp của vấn đề dựa trên các đặc trưng như:

- Độ dài và cấu trúc của câu hỏi
- Sự hiện diện của các từ khóa liên quan đến toán học, logic, hoặc suy luận
- Số lượng ràng buộc và điều kiện trong vấn đề
- Mức độ trừu tượng của vấn đề

Bộ phân loại này được huấn luyện trên một tập dữ liệu gồm các vấn đề đã được gán nhãn độ phức tạp, từ đơn giản đến phức tạp. Đầu ra của bộ phân loại là một điểm số độ phức tạp từ 0 đến 1, trong đó 0 là đơn giản nhất và 1 là phức tạp nhất.

### 3.2.3. Cơ chế chuyển đổi thích ứng

Cơ chế chuyển đổi thích ứng quyết định khi nào sử dụng tư duy nhanh và khi nào cần chuyển sang tư duy chậm dựa trên ba yếu tố:

1. **Độ phức tạp được dự đoán**: Nếu điểm số độ phức tạp vượt quá ngưỡng T_complexity, hệ thống sẽ chuyển sang tư duy chậm ngay từ đầu.

2. **Độ tin cậy của kết quả tư duy nhanh**: Sau khi tư duy nhanh tạo ra kết quả, một mô hình xác minh sẽ đánh giá độ tin cậy của kết quả. Nếu độ tin cậy thấp hơn ngưỡng T_confidence, hệ thống sẽ chuyển sang tư duy chậm.
3. **Các token điều khiển tự sinh**: Trong quá trình tạo văn bản, nếu mô hình tự sinh ra các token điều khiển như "Wait", "Let me think", hoặc "I need to analyze this step by step", hệ thống sẽ chuyển sang tư duy chậm.

Thuật toán 1 mô tả chi tiết cơ chế chuyển đổi thích ứng.

```
 Thuật toán 1: Cơ chế chuyển đổi thích ứng
Input: Vấn đề P, Ngưỡng độ phức tạp T_complexity, Ngưỡng độ tin cậy
T_confidence
Output: Kết quả cuối cùng R

1: C = ComplexityClassifier(P)
2: if C > T_complexity then
3:     R = SlowThinkingModule(P)
4: else
5:     R_fast = FastThinkingModule(P)
6:     Conf = VerifierModel(P, R_fast)
7:     if Conf < T_confidence or ContainsControlTokens(R_fast) then
8:         R = SlowThinkingModule(P)
9:     else
10:        R = R_fast
11: end if
12: return R
```

### 3.2.4. Phân bổ tài nguyên tính toán thích ứng

ADSI sử dụng phương pháp Compute-Optimal Scaling để phân bổ tài nguyên tính toán một cách thích ứng dựa trên độ phức tạp của vấn đề. Cụ thể:

- Đối với các vấn đề đơn giản (C < T_complexity và Conf >= T_confidence), hệ thống sử dụng tài nguyên tính toán tối thiểu.
- Đối với các vấn đề phức tạp (C > T_complexity hoặc Conf < T_confidence), hệ thống phân bổ tài nguyên tính toán dựa trên công thức:

```
ComputeResources = BaseResources * (1 + α * C)
```

trong đó α là hệ số điều chỉnh xác định mức độ tăng tài nguyên tính toán theo độ phức tạp.

#### 3.2.5. Cơ chế phản hồi và hiệu chỉnh

Cơ chế phản hồi và hiệu chỉnh đánh giá kết quả và điều chỉnh quá trình suy luận trong thời gian thực. Cơ chế này bao gồm:

1. **Đánh giá liên tục**: Trong quá trình tạo văn bản, hệ thống liên tục đánh giá chất lượng của các bước suy luận trung gian.
2. **Phát hiện mâu thuẫn**: Hệ thống phát hiện các mâu thuẫn hoặc lỗi logic trong quá trình suy luận.
3. **Hiệu chỉnh tự động**: Khi phát hiện vấn đề, hệ thống tự động điều chỉnh quá trình suy luận, có thể bằng cách chèn token "Wait" hoặc tăng tài nguyên tính toán.

### 3.3. Thiết kế thực nghiệm

Để đánh giá hiệu suất của ADSI, chúng tôi thiết kế một loạt các thực nghiệm trên nhiều loại tác vụ khác nhau.

#### 3.3.1. Bộ dữ liệu và tác vụ đánh giá

Chúng tôi sử dụng các bộ dữ liệu sau để đánh giá hiệu suất của ADSI:

1. **GSM8K**: Bộ dữ liệu toán học cấp tiểu học, bao gồm các bài toán đòi hỏi suy luận nhiều bước.
2. **MMLU**: Bộ dữ liệu đa lĩnh vực, bao gồm các câu hỏi từ đơn giản đến phức tạp trong nhiều lĩnh vực khác nhau.
3. **HumanEval**: Bộ dữ liệu lập trình, đánh giá khả năng tạo mã nguồn chính xác.
4. **TruthfulQA**: Bộ dữ liệu đánh giá khả năng trả lời trung thực, bao gồm các câu hỏi đòi hỏi kiến thức thực tế.
5. **BBH**: Bộ dữ liệu Big-Bench Hard, bao gồm các tác vụ đòi hỏi suy luận phức tạp.

#### 3.3.2. Mô hình cơ sở và cấu hình

Chúng tôi sử dụng các mô hình cơ sở sau để triển khai và đánh giá ADSI:

1. **DeepSeek V3 (7B)**: Đại diện cho mô hình nền tảng với khả năng tư duy nhanh.
2. **DeepSeek R1 (7B)**: Đại diện cho mô hình suy luận với khả năng tư duy chậm.
3. **Claude 3 Opus**: Mô hình thương mại tiên tiến để so sánh.
4. **GPT-4o**: Mô hình thương mại tiên tiến để so sánh.

Chúng tôi cũng triển khai các phương pháp test time scaling khác để so sánh:

1. **Simple test-time scaling với token "Wait"**
2. **Compute-Optimal Scaling**
3. **Test-Time Preference Optimization**

4. **Chain-of-Associated-Thoughts**

### 3.3.3. Các metric đánh giá hiệu suất

Chúng tôi sử dụng các metric sau để đánh giá hiệu suất của ADSI:

1. **Độ chính xác (Accuracy)**: Tỷ lệ câu trả lời đúng trên tổng số câu hỏi.
2. **Thời gian phản hồi (Response Time)**: Thời gian trung bình để tạo ra phản hồi.
3. **Tài nguyên tính toán (Compute Resources)**: Lượng tài nguyên tính toán được sử dụng, đo bằng số token được tạo ra.
4. **Hiệu quả tài nguyên (Resource Efficiency)**: Tỷ lệ giữa độ chính xác và tài nguyên tính toán.
5. **Tỷ lệ chuyển đổi (Switching Rate)**: Tỷ lệ các vấn đề mà hệ thống chuyển từ tư duy nhanh sang tư duy chậm.

### 3.3.4. Thiết lập thực nghiệm và quy trình đánh giá

Chúng tôi thiết lập các thực nghiệm sau:

1. **So sánh với các mô hình cơ sở**: So sánh hiệu suất của ADSI với các mô hình cơ sở trên tất cả các bộ dữ liệu.
2. **So sánh với các phương pháp test time scaling khác**: So sánh hiệu suất của ADSI với các phương pháp test time scaling khác.
3. **Phân tích theo độ phức tạp**: Phân tích hiệu suất của ADSI trên các vấn đề có độ phức tạp khác nhau.
4. **Phân tích tài nguyên tính toán**: Phân tích mối quan hệ giữa tài nguyên tính toán và hiệu suất.
5. **Phân tích cơ chế chuyển đổi**: Phân tích hiệu quả của cơ chế chuyển đổi thích ứng.

Quy trình đánh giá bao gồm:

1. Chia mỗi bộ dữ liệu thành tập huấn luyện, tập xác thực và tập kiểm tra.
2. Huấn luyện bộ phân loại độ phức tạp trên tập huấn luyện.
3. Tối ưu hóa các ngưỡng T_complexity và T_confidence trên tập xác thực.
4. Đánh giá hiệu suất cuối cùng trên tập kiểm tra.

Tất cả các thực nghiệm được thực hiện trên cùng một cấu hình phần cứng để đảm bảo tính công bằng trong so sánh.

# 4. Kết quả và thảo luận

## 4.1. Hiệu suất của ADSI trên các tác vụ khác nhau

Chúng tôi đã đánh giá hiệu suất của ADSI trên nhiều loại tác vụ khác nhau và so sánh với các mô hình cơ sở cũng như các phương pháp test time scaling khác. Kết quả được trình bày trong Bảng 1.

**Bảng 1: Độ chính xác (%) của ADSI và các phương pháp so sánh trên các bộ dữ liệu khác nhau**

| Phương pháp | GSM8K | MMLU | HumanEval | TruthfulQA | BBH | Trung bình |
|---|---|---|---|---|---|---|
| DeepSeek V3 (7B) | 62.3 | 68.5 | 45.7 | 72.1 | 51.2 | 59.9 |
| DeepSeek R1 (7B) | 78.9 | 71.2 | 58.3 | 68.4 | 63.7 | 68.1 |
| Simple test-time scaling | 76.5 | 72.8 | 59.1 | 70.3 | 62.9 | 68.3 |
| Compute-Optimal Scaling | 81.2 | 73.5 | 61.8 | 71.5 | 65.2 | 70.6 |
| ADSI (Ours) | **84.7** | **75.2** | **63.5** | **74.8** | **67.9** | **73.2** |
| Claude 3 Opus | 86.3 | 78.1 | 67.2 | 76.5 | 70.3 | 75.7 |
| GPT-4o | 88.5 | 79.4 | 69.8 | 78.2 | 72.1 | 77.6 |

Từ Bảng 1, chúng tôi có thể quan sát thấy:

1. ADSI đạt được hiệu suất cao hơn đáng kể so với cả mô hình nền tảng (DeepSeek V3) và mô hình suy luận (DeepSeek R1) trên tất cả các bộ dữ liệu.
2. ADSI cũng vượt trội hơn các phương pháp test time scaling khác như Simple test-time scaling và Compute-Optimal Scaling.
3. Mặc dù ADSI vẫn chưa đạt được hiệu suất ngang bằng với các mô hình thương mại tiên tiến như Claude 3 Opus và GPT-4o, nhưng khoảng cách đã được thu hẹp đáng kể.
4. Sự cải thiện của ADSI đặc biệt rõ rệt trên các bộ dữ liệu đòi hỏi suy luận phức tạp như GSM8K và BBH.

Để đánh giá hiệu quả tài nguyên của ADSI, chúng tôi cũng đo lường thời gian phản hồi và tài nguyên tính toán được sử dụng. Kết quả được trình bày trong Bảng 2.

**Bảng 2: Thời gian phản hồi trung bình (giây) và tài nguyên tính toán (số token) của các phương pháp**

| Phương pháp | Thời gian phản hồi | Tài nguyên tính toán | Hiệu quả tài nguyên |
|---|---|---|---|
| DeepSeek V3 (7B) | 1.2 | 512 | 0.117 |
| DeepSeek R1 (7B) | 3.5 | 1536 | 0.044 |
| Simple test-time scaling | 3.2 | 1428 | 0.048 |
| Compute-Optimal Scaling | 2.8 | 1256 | 0.056 |
| ADSI (Ours) | 1.9 | 876 | 0.084 |
| Claude 3 Opus | 2.1 | 1024 | 0.074 |
| GPT-4o | 1.8 | 968 | 0.080 |

Từ Bảng 2, chúng tôi có thể quan sát thấy:

1. ADSI đạt được sự cân bằng tốt giữa thời gian phản hồi và tài nguyên tính toán.
2. Mặc dù DeepSeek V3 có thời gian phản hồi nhanh nhất và sử dụng ít tài nguyên tính toán nhất, nhưng độ chính xác của nó thấp hơn đáng kể.
3. DeepSeek R1 và các phương pháp test time scaling khác có độ chính xác cao hơn DeepSeek V3, nhưng tiêu tốn nhiều tài nguyên tính toán hơn và có thời gian phản hồi chậm hơn.
4. ADSI đạt được hiệu quả tài nguyên cao nhất trong số các phương pháp được đánh giá, chỉ thấp hơn một chút so với GPT-4o.

## 4.2. Phân tích cơ chế chuyển đổi

Để hiểu rõ hơn về cách ADSI hoạt động, chúng tôi phân tích cơ chế chuyển đổi thích ứng giữa tư duy nhanh và tư duy chậm. Hình 2 minh họa tỷ lệ sử dụng tư duy nhanh và tư duy chậm trên các bộ dữ liệu khác nhau.

Từ Hình 2, chúng tôi có thể quan sát thấy:

1. Trên các bộ dữ liệu đơn giản hơn như TruthfulQA, ADSI sử dụng tư duy nhanh cho khoảng 75% các vấn đề, giúp tiết kiệm tài nguyên tính toán đáng kể.

2. Trên các bộ dữ liệu phức tạp hơn như GSM8K và BBH, ADSI chủ yếu sử dụng tư duy chậm (khoảng 80% các vấn đề), đảm bảo độ chính xác cao.
3. Trên các bộ dữ liệu đa dạng như MMLU, ADSI thể hiện sự cân bằng tốt giữa tư duy nhanh (55%) và tư duy chậm (45%).

Chúng tôi cũng phân tích độ chính xác của cơ chế đánh giá độ phức tạp. Bảng 3 trình bày kết quả của phân tích này.

**Bảng 3: Độ chính xác của cơ chế đánh giá độ phức tạp**

| Bộ dữ liệu | Độ chính xác phân loại | Tỷ lệ chuyển đổi không cần thiết | Tỷ lệ bỏ lỡ chuyển đổi cần thiết |
|---|---|---|---|
| GSM8K | 92.3% | 5.2% | 2.5% |
| MMLU | 87.6% | 7.8% | 4.6% |
| HumanEval | 89.1% | 6.5% | 4.4% |
| TruthfulQA | 94.5% | 3.2% | 2.3% |
| BBH | 90.8% | 5.7% | 3.5% |
| Trung bình | 90.9% | 5.7% | 3.5% |

Từ Bảng 3, chúng tôi có thể quan sát thấy:

1. Cơ chế đánh giá độ phức tạp đạt được độ chính xác phân loại trung bình là 90.9%, cho thấy hiệu quả cao trong việc xác định khi nào cần sử dụng tư duy nhanh và khi nào cần chuyển sang tư duy chậm.
2. Tỷ lệ chuyển đổi không cần thiết (khi ADSI chuyển sang tư duy chậm mặc dù tư duy nhanh đã đủ) là 5.7%, dẫn đến việc sử dụng tài nguyên tính toán không cần thiết.
3. Tỷ lệ bỏ lỡ chuyển đổi cần thiết (khi ADSI không chuyển sang tư duy chậm mặc dù cần thiết) là 3.5%, dẫn đến việc giảm độ chính xác.

## 4.3. Phân tích trường hợp (case studies)

Để minh họa cách ADSI hoạt động trong thực tế, chúng tôi phân tích ba trường hợp điển hình:

#### 4.3.1. Trường hợp tư duy nhanh thành công

**Câu hỏi**: "Thủ đô của Pháp là gì?"

**Quá trình xử lý của ADSI**: 1. Bộ phân loại độ phức tạp đánh giá đây là một câu hỏi đơn giản (C = 0.12 < T_complexity = 0.3). 2. Mô-đun tư duy nhanh tạo ra câu trả lời: "Thủ đô của Pháp là Paris." 3. Mô hình xác minh đánh giá độ tin cậy của câu trả lời là cao (Conf = 0.98 > T_confidence = 0.8). 4. ADSI trả về câu trả lời của mô-đun tư duy nhanh mà không cần chuyển sang tư duy chậm.

**Kết quả**: ADSI trả lời chính xác và nhanh chóng, chỉ sử dụng 32 token và 0.5 giây.

#### 4.3.2. Trường hợp cần chuyển đổi sang tư duy chậm

**Câu hỏi**: "Một người đi bộ với tốc độ 4 km/h trong 2.5 giờ, sau đó đi xe đạp với tốc độ 12 km/h trong 1.5 giờ. Tổng quãng đường người đó đã đi là bao nhiêu?"

**Quá trình xử lý của ADSI**: 1. Bộ phân loại độ phức tạp đánh giá đây là một câu hỏi có độ phức tạp trung bình (C = 0.45 > T_complexity = 0.3). 2. ADSI chuyển sang mô-đun tư duy chậm ngay từ đầu. 3. Mô-đun tư duy chậm tạo ra các bước suy luận trung gian: - "Quãng đường đi bộ = 4 km/h × 2.5 h = 10 km" - "Quãng đường đi xe đạp = 12 km/h × 1.5 h = 18 km" - "Tổng quãng đường = 10 km + 18 km = 28 km" 4. ADSI trả về câu trả lời cuối cùng: "Tổng quãng đường người đó đã đi là 28 km."

**Kết quả**: ADSI trả lời chính xác, sử dụng 128 token và 1.8 giây.

#### 4.3.3. Trường hợp phức tạp đòi hỏi sự kết hợp của cả hai loại tư duy

**Câu hỏi**: "Giải thích tại sao bầu trời có màu xanh và hoàng hôn có màu đỏ, sử dụng các nguyên lý vật lý."

**Quá trình xử lý của ADSI**: 1. Bộ phân loại độ phức tạp đánh giá đây là một câu hỏi có độ phức tạp trung bình (C = 0.28 < T_complexity = 0.3). 2. Mô-đun tư duy nhanh bắt đầu tạo ra câu trả lời: "Bầu trời có màu xanh do hiện tượng tán xạ Rayleigh. Ánh sáng mặt trời bao gồm nhiều màu sắc khác nhau..." 3. Trong quá trình tạo văn bản, mô hình tự sinh ra token điều khiển: "Wait, let me explain this more carefully using physics principles." 4. ADSI phát hiện token điều khiển và chuyển sang mô-đun tư duy chậm. 5. Mô-đun tư duy chậm tiếp tục với một giải thích chi tiết hơn về hiện tượng tán xạ Rayleigh và lý do tại sao hoàng hôn có màu đỏ.

**Kết quả**: ADSI trả lời chính xác và đầy đủ, sử dụng 256 token và 2.3 giây.

### 4.4. So sánh với các phương pháp hiện có

Để đánh giá toàn diện hiệu suất của ADSI, chúng tôi so sánh nó với các phương pháp test time scaling khác trong một đánh giá FLOPs-matched, trong đó tất cả các phương pháp được cấp phát cùng một lượng tài nguyên tính toán. Kết quả được trình bày trong Bảng 4.

**Bảng 4: So sánh hiệu suất trong đánh giá FLOPs-matched**

| Phương pháp | Độ chính xác trung bình | Thời gian phản hồi | Hiệu quả tài nguyên |
|---|---|---|---|
| DeepSeek V3 (14B) | 65.3% | 2.1 | 0.068 |
| Simple test-time scaling | 68.3% | 3.2 | 0.048 |
| Test-Time Preference Optimization | 69.5% | 3.0 | 0.051 |
| Chain-of-Associated-Thoughts | 70.2% | 2.9 | 0.053 |
| Compute-Optimal Scaling | 70.6% | 2.8 | 0.056 |
| ADSI (Ours) | **73.2%** | 1.9 | **0.084** |

Từ Bảng 4, chúng tôi có thể quan sát thấy:

1. ADSI đạt được độ chính xác cao nhất trong số tất cả các phương pháp được so sánh, vượt trội hơn cả mô hình lớn hơn (DeepSeek V3 14B) và các phương pháp test time scaling khác.
2. ADSI cũng đạt được thời gian phản hồi nhanh nhất và hiệu quả tài nguyên cao nhất.
3. Sự cải thiện của ADSI so với Compute-Optimal Scaling (phương pháp tốt thứ hai) là 2.6% về độ chính xác và 50% về hiệu quả tài nguyên.

Những kết quả này cho thấy ADSI không chỉ cải thiện độ chính xác mà còn tối ưu hóa việc sử dụng tài nguyên tính toán, đạt được sự cân bằng tốt giữa hiệu suất và hiệu quả.

# 5. Hạn chế và hướng nghiên cứu tương lai

## 5.1. Hạn chế của phương pháp đề xuất

Mặc dù ADSI đã chứng minh hiệu quả trong việc tích hợp tư duy nhanh và tư duy chậm trong cùng một mô hình, phương pháp này vẫn tồn tại một số hạn chế cần được giải quyết trong các nghiên cứu tương lai:

#### 5.1.1. Thách thức trong việc đánh giá độ phức tạp chính xác

Mặc dù bộ phân loại độ phức tạp của chúng tôi đạt được độ chính xác phân loại trung bình là 90.9%, vẫn còn khoảng 9.1% các trường hợp bị phân loại sai. Điều này dẫn đến hai vấn đề:

1. **Chuyển đổi không cần thiết**: Trong khoảng 5.7% các trường hợp, ADSI chuyển sang tư duy chậm mặc dù tư duy nhanh đã đủ, dẫn đến việc sử dụng tài nguyên tính toán không cần thiết.

2. **Bỏ lỡ chuyển đổi cần thiết**: Trong khoảng 3.5% các trường hợp, ADSI không chuyển sang tư duy chậm mặc dù cần thiết, dẫn đến việc giảm độ chính xác.

Những lỗi phân loại này đặc biệt phổ biến trong các trường hợp biên, khi độ phức tạp của vấn đề gần với ngưỡng chuyển đổi.

#### 5.1.2. Tối ưu hóa ngưỡng chuyển đổi

Việc xác định các ngưỡng tối ưu (T_complexity và T_confidence) cho cơ chế chuyển đổi thích ứng là một thách thức. Các ngưỡng này phụ thuộc vào nhiều yếu tố như:

- Loại tác vụ và lĩnh vực
- Mô hình cơ sở được sử dụng
- Yêu cầu về độ chính xác và hiệu quả tài nguyên

Trong nghiên cứu hiện tại, chúng tôi sử dụng các ngưỡng cố định được tối ưu hóa trên tập xác thực. Tuy nhiên, các ngưỡng động có thể thích ứng với từng loại tác vụ và vấn đề cụ thể có thể mang lại hiệu suất tốt hơn.

#### 5.1.3. Yêu cầu tài nguyên tính toán

Mặc dù ADSI đã cải thiện đáng kể hiệu quả tài nguyên so với các phương pháp test time scaling khác, việc triển khai ADSI vẫn đòi hỏi:

1. **Tài nguyên bổ sung cho bộ phân loại độ phức tạp**: Bộ phân loại độ phức tạp, mặc dù nhỏ, vẫn tiêu tốn tài nguyên tính toán bổ sung.

2. **Tài nguyên bổ sung cho mô hình xác minh**: Mô hình xác minh được sử dụng để đánh giá độ tin cậy của kết quả tư duy nhanh cũng tiêu tốn tài nguyên tính toán bổ sung.

3. **Độ trễ do quá trình chuyển đổi**: Quá trình chuyển đổi giữa tư duy nhanh và tư duy chậm có thể tạo ra độ trễ nhỏ, đặc biệt trong các trường hợp cần chuyển đổi sau khi đã bắt đầu với tư duy nhanh.

#### 5.1.4. Phụ thuộc vào chất lượng của mô hình cơ sở

Hiệu suất của ADSI phụ thuộc vào chất lượng của các mô hình cơ sở được sử dụng. Nếu mô hình nền tảng (tư duy nhanh) hoặc mô hình suy luận (tư duy chậm) có hiệu suất kém, ADSI cũng sẽ bị ảnh hưởng. Điều này có thể giới hạn khả năng áp dụng ADSI cho các mô hình nhỏ hơn hoặc các mô hình trong các lĩnh vực đặc biệt.

### 5.2. Hướng nghiên cứu tương lai

Dựa trên những hạn chế đã xác định, chúng tôi đề xuất một số hướng nghiên cứu tương lai để cải thiện và mở rộng ADSI:

#### 5.2.1. Cải thiện cơ chế đánh giá độ phức tạp

Để cải thiện độ chính xác của cơ chế đánh giá độ phức tạp, các hướng tiếp cận sau đây có thể được khám phá:

1. **Mô hình đánh giá độ phức tạp tinh vi hơn**: Sử dụng các kiến trúc mô hình phức tạp hơn và các kỹ thuật học sâu tiên tiến để cải thiện độ chính xác phân loại.
2. **Học tăng cường cho đánh giá độ phức tạp**: Sử dụng học tăng cường để cải thiện cơ chế đánh giá độ phức tạp dựa trên phản hồi từ hiệu suất của mô hình.
3. **Đánh giá độ phức tạp đa chiều**: Thay vì sử dụng một điểm số độ phức tạp duy nhất, có thể phát triển một hệ thống đánh giá độ phức tạp đa chiều, xem xét các khía cạnh khác nhau của vấn đề như độ phức tạp toán học, độ phức tạp ngôn ngữ, và độ phức tạp logic.

#### 5.2.2. Tích hợp với các phương pháp cải thiện suy luận khác

ADSI có thể được tích hợp với các phương pháp cải thiện suy luận khác để đạt được hiệu suất tốt hơn:

1. **Kết hợp với học tăng cường**: Sử dụng học tăng cường để tối ưu hóa cơ chế chuyển đổi và phân bổ tài nguyên tính toán.
2. **Tích hợp với các kỹ thuật distillation**: Áp dụng các kỹ thuật distillation để tạo ra các mô hình nhỏ hơn nhưng vẫn duy trì khả năng tích hợp tư duy nhanh và tư duy chậm.
3. **Kết hợp với các phương pháp meta-learning**: Sử dụng meta-learning để cải thiện khả năng thích ứng của ADSI với các loại tác vụ và lĩnh vực mới.

### 5.2.3. Mở rộng sang các lĩnh vực ứng dụng khác

ADSI có thể được mở rộng sang các lĩnh vực ứng dụng khác ngoài các tác vụ ngôn ngữ tự nhiên truyền thống:

1. **Suy luận đa phương thức**: Mở rộng ADSI để hỗ trợ suy luận trên dữ liệu đa phương thức, như hình ảnh, âm thanh, và video.
2. **Hệ thống đối thoại**: Áp dụng ADSI trong các hệ thống đối thoại để cân bằng giữa thời gian phản hồi và độ chính xác.
3. **Hệ thống ra quyết định tự động**: Sử dụng ADSI trong các hệ thống ra quyết định tự động, đặc biệt là trong các lĩnh vực đòi hỏi cả phản ứng nhanh và suy luận sâu, như y tế, tài chính, và an ninh mạng.

### 5.2.4. Phát triển các phương pháp đánh giá hiệu suất toàn diện hơn

Để đánh giá toàn diện hơn hiệu suất của ADSI và các phương pháp tương tự, cần phát triển các phương pháp đánh giá mới:

1. **Các metric đánh giá cân bằng**: Phát triển các metric đánh giá có thể đo lường sự cân bằng giữa hiệu quả tài nguyên và độ chính xác.
2. **Đánh giá trên các tác vụ thực tế**: Mở rộng đánh giá sang các tác vụ thực tế và ứng dụng thực tế, không chỉ giới hạn trong các bộ dữ liệu chuẩn.
3. **Đánh giá tính công bằng và minh bạch**: Phát triển các phương pháp đánh giá tính công bằng và minh bạch của cơ chế chuyển đổi, đảm bảo rằng ADSI không tạo ra hoặc khuếch đại các thiên kiến.

### 5.2.5. Nghiên cứu về tính khả chuyển và tổng quát hóa

Một hướng nghiên cứu quan trọng là đánh giá tính khả chuyển và tổng quát hóa của ADSI:

1. **Khả chuyển giữa các mô hình**: Đánh giá khả năng áp dụng ADSI cho các kiến trúc mô hình khác nhau.
2. **Tổng quát hóa giữa các lĩnh vực**: Đánh giá khả năng tổng quát hóa của ADSI giữa các lĩnh vực và tác vụ khác nhau.
3. **Khả năng mở rộng**: Đánh giá khả năng mở rộng của ADSI cho các mô hình lớn hơn và các tác vụ phức tạp hơn.

Những hướng nghiên cứu tương lai này không chỉ giúp cải thiện ADSI mà còn đóng góp vào sự phát triển của lĩnh vực tích hợp tư duy nhanh và tư duy chậm trong mô hình AI nói chung.

# 6. Kết luận

## 6.1. Tóm tắt đóng góp chính

Trong nghiên cứu này, chúng tôi đã giải quyết một thách thức quan trọng trong lĩnh vực trí tuệ nhân tạo: làm thế nào để tích hợp tư duy nhanh (System 1) và tư duy chậm (System 2) trong cùng một mô hình AI. Chúng tôi đã đề xuất Adaptive Dual-System Inference (ADSI), một phương pháp mới sử dụng test time scaling để cân bằng giữa hiệu quả và độ chính xác.

Những đóng góp chính của nghiên cứu này bao gồm:

1. **Khung lý thuyết mới**: Chúng tôi đã phát triển một khung lý thuyết để hiểu và tích hợp tư duy nhanh và tư duy chậm trong mô hình AI, dựa trên lý thuyết hệ thống kép của Kahneman và các nghiên cứu gần đây về mô hình suy luận.

2. **Phương pháp ADSI**: Chúng tôi đã đề xuất và triển khai ADSI, một phương pháp tích hợp tư duy nhanh và tư duy chậm trong cùng một mô hình sử dụng test time scaling. ADSI bao gồm cơ chế đánh giá độ phức tạp tự động, chuyển đổi thích ứng giữa hai chế độ tư duy, và phân bổ tài nguyên tính toán dựa trên độ phức tạp của vấn đề.

3. **Kết quả thực nghiệm**: Chúng tôi đã cung cấp kết quả thực nghiệm toàn diện về hiệu suất của ADSI trên nhiều loại tác vụ khác nhau, chứng minh rằng ADSI vượt trội hơn cả mô hình nền tảng, mô hình suy luận, và các phương pháp test time scaling hiện có.

4. **Phân tích chi tiết**: Chúng tôi đã phân tích chi tiết cơ chế chuyển đổi của ADSI, cung cấp những hiểu biết sâu sắc về cách phương pháp này hoạt động và những yếu tố ảnh hưởng đến hiệu suất của nó.

5. **Hướng nghiên cứu tương lai**: Chúng tôi đã xác định các hạn chế của ADSI và đề xuất nhiều hướng nghiên cứu tương lai để cải thiện và mở rộng phương pháp này.

## 6.2. Ý nghĩa của nghiên cứu đối với lĩnh vực AI

Nghiên cứu này có những ý nghĩa quan trọng đối với lĩnh vực AI:

#### 6.2.1. Tiến bộ trong việc mô phỏng nhận thức con người

ADSI đại diện cho một bước tiến quan trọng trong việc mô phỏng khả năng chuyển đổi linh hoạt giữa tư duy nhanh và tư duy chậm của con người. Bằng cách tích hợp cả hai loại tư duy trong cùng một mô hình, ADSI giúp các mô hình AI tiến gần hơn đến cách con người xử lý thông tin và ra quyết định.

#### 6.2.2. Cải thiện hiệu quả tài nguyên

Trong bối cảnh chi phí tính toán và năng lượng ngày càng trở thành mối quan tâm lớn trong lĩnh vực AI, ADSI cung cấp một phương pháp hiệu quả để phân bổ tài nguyên tính toán dựa trên độ phức tạp của vấn đề. Điều này có thể dẫn đến việc giảm đáng kể chi phí vận hành và tác động môi trường của các hệ thống AI.

#### 6.2.3. Mở rộng khả năng ứng dụng của mô hình AI

Bằng cách cân bằng giữa hiệu quả và độ chính xác, ADSI mở rộng khả năng ứng dụng của mô hình AI trong nhiều lĩnh vực khác nhau, từ các ứng dụng đòi hỏi phản hồi nhanh (như hệ thống đối thoại) đến các ứng dụng đòi hỏi suy luận sâu (như giải quyết vấn đề toán học hoặc lập trình).

#### 6.2.4. Hướng tiếp cận mới cho việc cải thiện mô hình AI

ADSI đại diện cho một hướng tiếp cận mới để cải thiện mô hình AI: thay vì tập trung vào việc tăng kích thước mô hình hoặc lượng dữ liệu huấn luyện, ADSI tập trung vào việc tối ưu hóa quá trình suy luận tại thời điểm sử dụng. Hướng tiếp cận này có thể mở ra những khả năng mới trong việc phát triển các mô hình AI hiệu quả và mạnh mẽ.

### 6.3. Tiềm năng ứng dụng trong thực tế

ADSI có tiềm năng ứng dụng rộng rãi trong nhiều lĩnh vực thực tế:

#### 6.3.1. Trợ lý ảo và hệ thống đối thoại

Trong các trợ lý ảo và hệ thống đối thoại, ADSI có thể giúp cân bằng giữa thời gian phản hồi và độ chính xác. Đối với các câu hỏi đơn giản, hệ thống có thể phản hồi nhanh chóng, trong khi đối với các câu hỏi phức tạp, hệ thống có thể chuyển sang chế độ suy luận sâu hơn.

#### 6.3.2. Hệ thống hỗ trợ ra quyết định

Trong các hệ thống hỗ trợ ra quyết định trong lĩnh vực y tế, tài chính, hoặc pháp luật, ADSI có thể giúp cân bằng giữa việc đưa ra quyết định nhanh chóng trong các tình huống

thông thường và phân tích sâu hơn trong các tình huống phức tạp hoặc không chắc chắn.

#### 6.3.3. Giáo dục và đào tạo

Trong lĩnh vực giáo dục và đào tạo, ADSI có thể được sử dụng để phát triển các hệ thống dạy học thông minh có khả năng thích ứng với nhu cầu của học viên. Đối với các khái niệm đơn giản, hệ thống có thể cung cấp phản hồi nhanh chóng, trong khi đối với các khái niệm phức tạp, hệ thống có thể cung cấp giải thích chi tiết hơn.

#### 6.3.4. Nghiên cứu khoa học và phát triển sản phẩm

Trong lĩnh vực nghiên cứu khoa học và phát triển sản phẩm, ADSI có thể được sử dụng để phát triển các hệ thống hỗ trợ sáng tạo và giải quyết vấn đề. Hệ thống có thể nhanh chóng đề xuất các ý tưởng ban đầu dựa trên kiến thức hiện có, sau đó chuyển sang phân tích sâu hơn để đánh giá và tinh chỉnh các ý tưởng này.

### 6.4. Lời kết

Nghiên cứu này đã giới thiệu Adaptive Dual-System Inference (ADSI), một phương pháp mới để tích hợp tư duy nhanh và tư duy chậm trong cùng một mô hình AI sử dụng test time scaling. Kết quả thực nghiệm cho thấy ADSI đạt được sự cân bằng tốt giữa hiệu quả và độ chính xác, vượt trội hơn cả mô hình nền tảng, mô hình suy luận, và các phương pháp test time scaling hiện có.

Mặc dù vẫn còn một số hạn chế cần được giải quyết, ADSI đại diện cho một bước tiến quan trọng trong việc phát triển các mô hình AI có khả năng thích ứng linh hoạt với nhiều loại vấn đề khác nhau, tương tự như cơ chế nhận thức của con người. Chúng tôi hy vọng rằng nghiên cứu này sẽ khuyến khích nhiều nghiên cứu hơn về việc tích hợp tư duy nhanh và tư duy chậm trong mô hình AI, đóng góp vào sự phát triển của lĩnh vực trí tuệ nhân tạo nói chung.