**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

---------------------------------------



ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC NGÀNH KỸ THUẬT PHẦN MỀM

**Tìm hiểu, ứng dụng mô hình AI phát triển công cụ gợi ý việc làm thông minh**

**GVHD:** PGS. TS Trần Đăng Hưng

**Sinh viên:** Nguyễn Quang Huy

**Mã số sinh viên:** 2020606068

Hà Nội – 2025

**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

---------------------------------------



ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC NGÀNH KỸ THUẬT PHẦN MỀM

**Tìm hiểu, ứng dụng mô hình AI phát triển công cụ gợi ý việc làm thông minh**

**GVHD:** PGS. TS Trần Đăng Hưng

**Sinh viên:** Nguyễn Quang Huy

**Mã số sinh viên:** 2020606068

Hà Nội – 2025

# MỤC LỤC

[MỤC LỤC i](#_Toc198079838)

[LỜI CẢM ƠN vi](#_Toc198079839)

[DANH MỤC CÁC THUẬT NGỮ, KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT vii](#_Toc198079840)

[DANH MỤC BẢNG BIỂU viii](#_Toc198079841)

[MỞ ĐẦU 1](#_Toc198079842)

[CHƯƠNG 1: TỔNG QUAN 4](#_Toc198079843)

[1.1. Giới thiệu chung về trí tuệ nhân tạo (AI) 4](#_Toc198079844)

[1.1.1 Khái niệm và lịch sử phát triển 4](#_Toc198079845)

[1.1.2 Các lĩnh vực chính của AI 4](#_Toc198079846)

[1.1.3 Machine Learning và Deep Learning 5](#_Toc198079847)

[1.2. Ứng dụng AI trong thực tiễn 6](#_Toc198079848)

[1.2.1 Hệ thống đề xuất (Recommendation Systems) 6](#_Toc198079849)

[1.2.2 Chatbot và trợ lý ảo 6](#_Toc198079850)

[1.2.3 Phân tích dữ liệu và dự đoán xu hướng 7](#_Toc198079851)

[1.3. Thực trạng thị trường việc làm và các nền tảng tìm kiếm việc làm 8](#_Toc198079852)

[1.3.1 Thách thức trong tìm kiếm việc làm hiện nay 8](#_Toc198079853)

[1.3.2 Phân tích các nền tảng tìm kiếm việc làm hiện có 8](#_Toc198079854)

[1.3.3 Vai trò của AI trong việc cải thiện quá trình tìm kiếm việc làm 9](#_Toc198079855)

[1.4. Các công cụ gợi ý thông minh trong tuyển dụng 9](#_Toc198079856)

[1.4.1 Tổng quan về hệ thống gợi ý trong tuyển dụng 9](#_Toc198079857)

[1.4.2 Phân tích một số giải pháp hiện có 10](#_Toc198079858)

[1.4.3 Những thách thức và cơ hội 10](#_Toc198079859)

[1.5. Đề xuất mô hình và định hướng nghiên cứu 11](#_Toc198079860)

[CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ CÔNG CỤ 13](#_Toc198079861)

[2.1. Xử lý ngôn ngữ tự nhiên (NLP) 13](#_Toc198079862)

[2.1.1 Khái niệm và vai trò của NLP 13](#_Toc198079863)

[2.1.2 Các bước xử lý văn bản cơ bản 13](#_Toc198079864)

[2.1.3 Named Entity Recognition (NER) và ứng dụng 14](#_Toc198079865)

[2.2. Mô hình biểu diễn văn bản 14](#_Toc198079866)

[2.2.1 Term Frequency-Inverse Document Frequency (TF-IDF) 14](#_Toc198079867)

[2.2.2 Word Embedding 15](#_Toc198079868)

[2.3. Thuật toán K-Nearest Neighbors (KNN) 16](#_Toc198079869)

[2.3.1 Nguyên lý hoạt động 16](#_Toc198079870)

[2.3.2 Các độ đo khoảng cách (Euclidean, Cosine, Manhattan) 16](#_Toc198079871)

[2.3.3 Ưu và nhược điểm của KNN 17](#_Toc198079872)

[2.3.4 Ứng dụng KNN trong hệ thống gợi ý 17](#_Toc198079873)

[2.4. Deep Learning trong xử lý ngôn ngữ tự nhiên 17](#_Toc198079874)

[2.4.1 Kiến trúc mạng nơ-ron sâu cho NLP 17](#_Toc198079875)

[2.4.2 Transformer và cơ chế Attention 18](#_Toc198079876)

[2.4.3 Mô hình BERT (Bidirectional Encoder Representations from Transformers) 18](#_Toc198079877)

[2.4.4 So sánh phương pháp truyền thống và Deep Learning trong NLP 19](#_Toc198079878)

[CHƯƠNG 3: XÂY DỰNG VÀ ĐÁNH GIÁ 20](#_Toc198079879)

[3.1. Thu thập và tiền xử lý dữ liệu 20](#_Toc198079880)

[3.1.1 Nguồn dữ liệu và phương pháp thu thập 20](#_Toc198079881)

[3.1.2 Cấu trúc dữ liệu Job Descriptions và Resume 20](#_Toc198079882)

[3.1.3 Các bước tiền xử lý dữ liệu 21](#_Toc198079883)

[3.1.4 Phân tích khám phá dữ liệu (EDA) 21](#_Toc198079884)

[3.2. Phương pháp 1: TF‑IDF kết hợp KNN 22](#_Toc198079885)

[3.2.1 Thiết kế hệ thống 22](#_Toc198079886)

[3.2.2 Xây dựng ma trận TF‑IDF 22](#_Toc198079887)

[3.2.3 Áp dụng thuật toán KNN 22](#_Toc198079888)

[3.2.4 Cài đặt và triển khai 22](#_Toc198079889)

[3.2.5 Kết quả thực nghiệm 23](#_Toc198079890)

[3.3. Phương pháp 2: Embedding với BERT 26](#_Toc198079891)

[3.3.1 Thiết kế hệ thống 26](#_Toc198079892)

[3.3.2 Sử dụng mô hình BERT pre-trained 26](#_Toc198079893)

[3.3.3 Embedding các job descriptions và resume 26](#_Toc198079894)

[3.3.4 So sánh độ tương đồng giữa các vector nhúng 26](#_Toc198079895)

[3.3.5 Cài đặt và triển khai 27](#_Toc198079896)

[3.3.6 Kết quả thực nghiệm 27](#_Toc198079897)

[3.4. So sánh và đánh giá hai phương pháp 29](#_Toc198079898)

[3.4.1 Các metric đánh giá 29](#_Toc198079899)

[3.4.2 Độ chính xác trong matching 29](#_Toc198079900)

[3.4.3 Thời gian xử lý 29](#_Toc198079901)

[3.4.4 Khả năng mở rộng 30](#_Toc198079902)

[3.4.5 Phân tích ưu nhược điểm mỗi phương pháp 30](#_Toc198079903)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 31](#_Toc198079904)

[TÀI LIỆU THAM KHẢO 33](#_Toc198079905)

# LỜI CẢM ƠN

Sau 4 năm học tập và rèn luyện tại Trường Đại học Công Nghiệp Hà Nội, đồ án tốt nghiệp này là một dấu ấn quan trọng đánh dấu việc em - một sinh viên đã hoàn thành nhiệm vụ của mình trên ghế giảng đường Đại học. Đầu tiên, với tình cảm sâu sắc, chân thành, cho phép em được bày tỏ lòng biết ơn sâu sắc đến Gia đình, các anh chị, bạn bè đã luôn động viên, khích lệ, tạo điều kiện giúp đỡ em trong suốt quá trình học tập và nghiên cứu đề tài.

Em cũng xin gửi tới các thầy các cô khoa Công Nghệ Thông Tin Trường Đại học Công Nghiệp Hà Nội lời chào trân trọng, lời chúc sức khỏe và lời cảm ơn sâu sắc. Với sự quan tâm, dạy dỗ, chỉ bảo tận tình chu đáo của thầy cô, đến nay em đã có thể hoàn thành đồ án tốt nghiệp.

Đặc biệt em xin gửi lời cảm ơn chân thành đến thầy PGS. TS. Trần Đăng Hưng và đã tận tình giúp và hướng dẫn em hoàn thành đề tài đồ án. **“Tìm hiểu, ứng dụng mô hình AI phát triển công cụ gợi ý việc làm thông minh”.**

Đồng thời, em xin bày tỏ lòng biết ơn đến lãnh đạo Trường Đại học Công Nghiệp Hà Nội đã tạo điều kiện cho em được học tập tại nơi mà em yêu thích, cho em bước vào đời sống thực tế và áp dụng những kiến thức em đã học tại trường và môi trường làm việc mới của em. Qua quá trình học tập em đã tích lũy được rất nhiều kiến thức để chuẩn bị cho công việc sau này cũng như để phát triển thêm bản thân.

Trong quá trình hoàn thành đồ án tốt nghiệp không thể tránh khỏi thiếu sót, kính mong có sự góp ý từ thầy cô.

*Em xin chân thành cảm ơn!*

# DANH MỤC CÁC THUẬT NGỮ, KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Thuật ngữ, ký hiệu và các chữ viết tắt | Chữ viết đầy đủ |
| 1 | AI | Artificial Intelligence |
| 2 | KNN | k-Nearest Neigbors |
| 3 | TF-IDF | Term Frequency-Inverse Document Frequency |
| 4 | BERT | Bidirectional Encoder Representations from Transformers |

# DANH MỤC BẢNG BIỂU

[Bảng 2.1: So sánh phương pháp truyền thống và Deep Learning trong NLP 19](#_Toc198079566)

[Bảng 3.1: Phân tích ưu nhược điểm giữa 2 phương pháp TF-IDF và BERT Embedding 30](#_Toc198079567)

# MỞ ĐẦU

Bài toán gợi ý việc làm phù hợp đang ngày càng trở nên cấp thiết trong bối cảnh thị trường lao động phát triển nhanh chóng và đa dạng hóa ngành nghề. Hệ thống truyền thống thường yêu cầu người tìm việc phải tự mình sàng lọc thông tin, dẫn đến mất nhiều thời gian và công sức. Trong khi đó, ứng dụng trí tuệ nhân tạo (AI) với khả năng phân tích ngữ nghĩa văn bản và tìm kiếm tương đồng hứa hẹn mang đến giải pháp tự động hóa, nâng cao hiệu quả kết nối giữa ứng viên và nhà tuyển dụng.

**1. Lý do chọn đề tài**

Việc lựa chọn đề tài "Tìm hiểu, ứng dụng mô hình AI phát triển công cụ gợi ý việc làm thông minh" xuất phát từ nhu cầu thực tiễn kết nối nguồn nhân lực chất lượng cao với cơ hội việc làm phù hợp. Thị trường lao động Việt Nam ngày càng cạnh tranh, đồng thời người tìm việc thường gặp khó khăn trong việc tìm kiếm công việc phù hợp với kỹ năng và kinh nghiệm của bản thân. Bằng cách áp dụng các kỹ thuật xử lý ngôn ngữ tự nhiên như TF‑IDF, thuật toán KNN và mô hình BERT, đề tài hướng tới xây dựng công cụ hỗ trợ tự động gợi ý việc làm dựa trên phân tích nội dung hồ sơ và mô tả công việc.

**2. Mục tiêu nghiên cứu**

* Phân tích, đánh giá và lựa chọn các phương pháp phù hợp để biểu diễn và so sánh nội dung hồ sơ xin việc (resume) và mô tả công việc (job description).
* Xây dựng và triển khai hai hướng tiếp cận: sử dụng TF‑IDF kết hợp KNN và sử dụng embedding từ mô hình BERT pretrained.
* Đo lường hiệu năng, độ chính xác của hai phương pháp thông qua các chỉ số đánh giá (Precision@K, Recall@K, F1‑score).
* Thiết kế hệ thống demo với API backend (Flask + MongoDB) và giao diện frontend (Next.js) cho phép tải lên hồ sơ và nhận kết quả gợi ý.

**3. Phạm vi và giới hạn đề tài**

* Nguồn dữ liệu: tập hợp mẫu resume và job description thu thập từ Kaggle và các trang tuyển dụng công khai; quy mô dữ liệu giới hạn trong vài nghìn bản ghi.
* Chỉ nghiên cứu hai hướng chính: mô hình TF‑IDF + KNN và embedding BERT; không xem xét các mô hình deep learning phức tạp khác.
* Đánh giá kết quả trên dữ liệu tiếng Anh (có thể áp dụng tương tự cho tiếng Việt nhưng không thực hiện chi tiết).
* Hệ thống minh họa ở mức prototype, không tối ưu cho môi trường sản xuất quy mô lớn.

**4. Phương pháp nghiên cứu**

* Tổng quan tài liệu: tìm hiểu các công trình, bài báo liên quan đến recommendation system và NLP.
* Thiết kế và triển khai thuật toán: cài đặt TF‑IDF, KNN và tích hợp pretrained BERT để trích xuất embedding.
* Thử nghiệm và đánh giá: thực hiện thí nghiệm trên tập test, thu thập chỉ số Precision, Recall, F1, so sánh hai phương pháp.
* Xây dựng hệ thống: phát triển API backend với Flask, lưu trữ dữ liệu và embedding trên MongoDB; triển khai giao diện đơn giản với Next.js để người dùng tương tác.

# TỔNG QUAN

## Giới thiệu chung về trí tuệ nhân tạo (AI)

### **Khái niệm và lịch sử phát triển**

Trí tuệ nhân tạo (Artificial Intelligence - AI) là lĩnh vực nghiên cứu và phát triển các hệ thống máy tính có khả năng thực hiện các nhiệm vụ thường đòi hỏi trí tuệ con người, như hiểu ngôn ngữ tự nhiên, nhận diện hình ảnh, ra quyết định và học hỏi từ dữ liệu. Khái niệm AI được John McCarthy đề xuất năm 1956 tại Hội nghị Dartmouth. Từ đó, AI trải qua nhiều giai đoạn phát triển:

* Thập niên 1950-1960: AI sơ khai với các hệ chuyên gia (expert systems) và giải thuật tìm kiếm.
* Thập niên 1970-1980: Giai đoạn “Mùa đông AI” do hạn chế tính toán và dữ liệu.
* Thập niên 1990-2000: Sự trở lại mạnh mẽ với các thuật toán học máy (machine learning) và thống kê.
* Từ 2010 đến nay: Bùng nổ học sâu (deep learning), mô hình ngôn ngữ lớn (LLM), ứng dụng AI vào sản phẩm thực tế.

### Các lĩnh vực chính của AI

* Học máy (Machine Learning - ML): Tập trung vào phát triển thuật toán cho phép máy học từ dữ liệu mà không cần lập trình tường minh.
* Học sâu (Deep Learning - DL): Nhánh con của ML sử dụng mạng nơ-ron nhiều lớp để tự động trích xuất đặc trưng.
* Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP): Nghiên cứu tương tác giữa máy tính và ngôn ngữ con người.
* Thị giác máy tính (Computer Vision - CV): Chuyển đổi và phân tích hình ảnh, video.
* Hệ thống chuyên gia và quy tắc (Expert Systems & Rule-based): Hệ thống đưa ra quyết định dựa trên tập luật được định nghĩa.
* Robotics: Ứng dụng AI vào điều khiển robot thực thi nhiệm vụ trong môi trường thực.

### Machine Learning và Deep Learning

Machine Learning là phương pháp xây dựng các mô hình thông qua việc huấn luyện trên dữ liệu; ML bao gồm cả các thuật toán giám sát (supervised), không giám sát (unsupervised) và bán giám sát (semi-supervised). Deep Learning là một nhánh của Machine Learning, trong đó sử dụng kiến trúc mạng nơ-ron nhiều lớp (deep neural networks) để học các biểu diễn phức tạp từ dữ liệu thô. Ưu điểm của DL so với ML truyền thống:

* Tự động học đặc trưng: mạng nơ-ron nhiều lớp có thể tự động trích xuất đặc trưng từ dữ liệu, giảm công việc tiền xử lý.
* Hiệu suất cao trên dữ liệu lớn: DL đạt kết quả vượt trội trong các bài toán nhận dạng hình ảnh, âm thanh, ngôn ngữ khi có nhiều dữ liệu và tài nguyên tính toán.
* Khả năng mở rộng: dễ dàng điều chỉnh độ sâu, độ rộng của mạng để phù hợp với yêu cầu bài toán. Nhược điểm:
* Yêu cầu nhiều dữ liệu và tính toán.
* Khó giải thích (black-box), đòi hỏi kỹ thuật giải thích mô hình (model interpretability).

## Ứng dụng AI trong thực tiễn

### Hệ thống đề xuất (Recommendation Systems)

Hệ thống đề xuất là một trong những ứng dụng phổ biến nhất của AI trong các nền tảng trực tuyến. AI thu thập dữ liệu hành vi người dùng (lịch sử truy cập, tương tác, đánh giá) và sử dụng các thuật toán học máy để đưa ra đề xuất cá nhân hóa. Ví dụ:

* Netflix và Spotify sử dụng Collaborative Filtering để đề xuất phim, nhạc dựa trên sở thích của người dùng tương tự.
* Amazon áp dụng hybrid recommendation (kết hợp collaborative và content-based) để giới thiệu sản phẩm phù hợp với hồ sơ mua sắm. Với bài toán gợi ý việc làm, hệ thống đề xuất có thể học từ lịch sử ứng tuyển, lượt xem JD và phản hồi của ứng viên để ưu tiên các vị trí phù hợp.

### Chatbot và trợ lý ảo

Chatbot và trợ lý ảo ngày càng được ứng dụng rộng rãi trong chăm sóc khách hàng và hỗ trợ tìm kiếm việc làm. Các chatbot AI sử dụng NLP để hiểu và phản hồi câu hỏi của người dùng:

* HR chatbot trên các trang tuyển dụng giúp giải đáp tự động thắc mắc về quy trình ứng tuyển, điều kiện công việc.
* Trợ lý ảo như Google Assistant, Siri hỗ trợ người dùng tìm kiếm thông tin việc làm nhanh chóng qua giọng nói. Công nghệ NLP, đặc biệt là mô hình Transformer, giúp chatbot xử lý ngôn ngữ tự nhiên ngày càng linh hoạt, chính xác hơn.

### Phân tích dữ liệu và dự đoán xu hướng

AI được sử dụng để phân tích lượng lớn dữ liệu tuyển dụng và hồ sơ ứng viên, từ đó rút ra các xu hướng thị trường lao động:

* Phân tích nhu cầu kỹ năng: xác định kỹ năng nào đang có nhu cầu cao qua text mining trên mô tả công việc.
* Dự báo lương và xu hướng tuyển dụng qua time series forecasting.
* Giảm thiểu bias trong tuyển dụng bằng phân tích công bằng (fairness) và explainability. Trong đề tài này, các kỹ thuật phân tích văn bản như TF-IDF và embedding từ mô hình BERT sẽ giúp trích xuất thông tin quan trọng từ resume và JD, hỗ trợ bước phân tích và so khớp.

## Thực trạng thị trường việc làm và các nền tảng tìm kiếm việc làm

### Thách thức trong tìm kiếm việc làm hiện nay

Thị trường việc làm ngày càng cạnh tranh với số lượng ứng viên tăng cao và yêu cầu ngày càng khắt khe từ nhà tuyển dụng. Các thách thức chính bao gồm:

* **Khối lượng thông tin lớn**: Hàng nghìn tin tuyển dụng được đăng mới mỗi ngày khiến người tìm việc khó khăn trong việc sàng lọc và cập nhật.
* **Chất lượng tin không đồng đều**: Một số tin đăng thiếu mô tả chi tiết hoặc không rõ ràng yêu cầu công việc.
* **Thiếu cá nhân hóa**: Hệ thống tìm việc truyền thống thường trả về kết quả chung chung, không sát với hồ sơ và sở thích cá nhân.
* **Rào cản ngôn ngữ và thuật ngữ chuyên môn**: Mô tả công việc đa dạng về cách diễn đạt và thuật ngữ, khó so khớp chính xác với hồ sơ ứng viên.

### Phân tích các nền tảng tìm kiếm việc làm hiện có

* **LinkedIn**: Mạng xã hội nghề nghiệp lớn nhất với tính năng gợi ý việc làm dựa trên hồ sơ và mạng lưới kết nối. Hạn chế: ưu tiên việc làm theo khu vực và ngành nghề chính, ít hỗ trợ ngôn ngữ tiếng Việt.
* **Indeed**: Tập trung vào tìm kiếm toàn cầu, cho phép lọc việc làm theo từ khóa, vị trí, mức lương. Hạn chế: giao diện đơn giản, thiếu cá nhân hóa sâu.
* **VietnamWorks**: Nền tảng việc làm hàng đầu Việt Nam, chuyên ngành và có hỗ trợ tiếng Việt. Hạn chế: lượng tin cập nhật chưa đa dạng so với nhu cầu thị trường.
* **TopCV, JobStreet, Glassdoor**: Cung cấp đánh giá công ty, mức lương và trải nghiệm tuyển dụng. Hạn chế: dữ liệu đánh giá chủ yếu đến từ các thị trường lớn, ít phản ánh thực tế Việt Nam.

### Vai trò của AI trong việc cải thiện quá trình tìm kiếm việc làm

AI đóng vai trò then chốt trong việc nâng cao trải nghiệm người dùng và hiệu quả kết nối ứng viên – nhà tuyển dụng:

* **Tự động khớp hồ sơ**: Sử dụng NLP để phân tích ngữ nghĩa resume và JD, từ đó tự động gợi ý vị trí phù hợp.
* **Cá nhân hóa đề xuất**: Học thói quen và phản hồi của ứng viên để cải thiện đề xuất theo thời gian.
* **Giảm thiểu sai lệch thông tin**: Chuẩn hóa ngôn ngữ và thuật ngữ chuyên môn, giúp so khớp chính xác hơn giữa ứng viên và yêu cầu công việc.
* **Tối ưu hóa quy trình tuyển dụng**: Nhà tuyển dụng nhận được danh sách ứng viên phù hợp nhanh hơn, tiết kiệm thời gian sàng lọc.

## Các công cụ gợi ý thông minh trong tuyển dụng

### Tổng quan về hệ thống gợi ý trong tuyển dụng

Hệ thống gợi ý tuyển dụng (Recruitment Recommendation System) sử dụng AI và NLP để tự động kết nối ứng viên với vị trí công việc phù hợp. Quy trình chính gồm:

* Thu thập và xử lý dữ liệu hồ sơ (resume) và mô tả công việc (job description).
* Biểu diễn văn bản dưới dạng vector (TF-IDF, embedding).
* Tính toán độ tương đồng (similarity) giữa hồ sơ và mô tả công việc.
* Xếp hạng và trả về danh sách vị trí ưu tiên.

### Phân tích một số giải pháp hiện có

* **LinkedIn Recruiter**: Sử dụng Graph-based Algorithms và ML để gợi ý ứng viên dựa trên mạng lưới kết nối, kinh nghiệm và kỹ năng định danh. Hỗ trợ tính năng “People You May Know” cho tìm ứng viên tiềm năng.
* **HireVue**: Áp dụng AI-driven video interview analysis, phân tích ngôn ngữ và biểu cảm để đánh giá ứng viên. Kết hợp với hệ thống matching dựa trên Keywords và embedding.
* **Textkernel**: Cung cấp công cụ parsing và matching resume – JD với khả năng học từ dữ liệu lịch sử tuyển dụng của doanh nghiệp. Hỗ trợ nhiều ngôn ngữ.

### Những thách thức và cơ hội

* **Thách thức:**
* Dữ liệu hồ sơ và JD thường không đồng nhất về cấu trúc, yêu cầu tiền xử lý phức tạp.
* Vấn đề bias: mô hình có thể học theo các định kiến trong dữ liệu tuyển dụng cũ.
* Độ giải thích thấp (interpretability): người dùng khó hiểu vì sao một vị trí được gợi ý.
* Hiệu năng và độ trễ: xử lý embedding BERT trên bộ dữ liệu lớn tiêu tốn tài nguyên.
* **Cơ hội:**
* Tích hợp đa nguồn dữ liệu (social profile, đánh giá kỹ năng) để nâng cao độ chính xác.
* Sử dụng kỹ thuật explainable AI (XAI) để minh bạch quá trình gợi ý.
* Mở rộng hỗ trợ ngôn ngữ và miền chuyên môn khác nhau.
* Tối ưu hóa mô hình cho thiết bị biên (edge) hoặc huấn luyện federated learning để bảo mật dữ liệu cá nhân.

## Đề xuất mô hình và định hướng nghiên cứu

Trong Chương 1, chúng ta đã khảo sát thực trạng và các công cụ gợi ý việc làm. Dựa trên nhu cầu kết nối chính xác giữa hồ sơ ứng viên và mô tả công việc, đề tài lựa chọn ba kỹ thuật chính:

* **TF‑IDF (Term Frequency-Inverse Document Frequency)**
* Đơn giản, dễ triển khai và giải thích.
* Thể hiện mức độ quan trọng của từ khóa trong hồ sơ và JD.
* Hiệu quả với tập dữ liệu văn bản quy mô vừa, giúp nhanh chóng xây dựng prototyping hệ thống gợi ý.
* **KNN (K-Nearest Neighbors)**
* Thuật toán không tham số, sử dụng khoảng cách (cosine, Euclid) để tìm các JD gần nhất với resume.
* Khả năng mở rộng bằng điều chỉnh K và hàm khoảng cách phù hợp.
* Kết quả dễ trực quan hóa và đánh giá, phù hợp cho giai đoạn thử nghiệm.
* **BERT (Bidirectional Encoder Representations from Transformers)**
* Mô hình contextual embedding hiện đại, nắm bắt ngữ nghĩa sâu sắc hơn so với TF‑IDF.
* Trích xuất embedding [CLS] đại diện cho toàn câu giúp so khớp resume và JD hiệu quả.
* Dễ fine-tuning cho ngữ cảnh tuyển dụng, khả năng cải thiện độ chính xác gợi ý.

**Định hướng chương 2:** Chương tiếp theo sẽ trình bày chi tiết cơ sở lý thuyết của các thành phần trên:

* Tiền xử lý văn bản và NER
* Nguyên lý TF‑IDF và cách xây dựng ma trận từ-văn bản
* Thuật toán KNN và các hàm khoảng cách
* Word embedding truyền thống (Word2Vec, GloVe) và embedding contextual
* Kiến trúc Transformer và chi tiết BERT
* Các chỉ số đánh giá hiệu năng mô hình

# PHÂN TÍCH VÀ THIẾT KẾ CÔNG CỤ

## Xử lý ngôn ngữ tự nhiên (NLP)

### Khái niệm và vai trò của NLP

Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) là lĩnh vực giao thoa giữa khoa học máy tính và ngôn ngữ học, tập trung vào việc cho máy tính khả năng hiểu, phân tích và sinh ngôn ngữ con người. Vai trò chính của NLP trong hệ thống gợi ý việc làm bao gồm:

* Tự động trích xuất thông tin quan trọng từ bản mô tả công việc và hồ sơ ứng viên.
* Chuẩn hóa và làm giàu dữ liệu văn bản để hỗ trợ các bước biểu diễn và so khớp.
* Cải thiện độ chính xác của thuật toán giống như TF-IDF, embedding và các mô hình ngôn ngữ sâu.

### Các bước xử lý văn bản cơ bản

Để chuyển văn bản thô thành dạng có thể phân tích, các bước tiền xử lý thường được áp dụng:

* **Tokenization**: Phân tách đoạn văn bản thành các đơn vị cơ bản (từ, câu). Ví dụ, chuỗi "Software Engineer at XYZ" thành ["Software", "Engineer", "at", "XYZ"].
* **Lowercasing**: Chuyển tất cả ký tự thành chữ thường để giảm đa dạng biểu diễn (ví dụ: "Engineer" → "engineer").
* **Loại bỏ dấu câu và ký tự đặc biệt**: Xóa các ký tự không cần thiết như ".,;:!" để làm sạch văn bản.
* **Loại bỏ stopwords**: Gỡ bỏ các từ dừng (stopwords) không mang nhiều ý nghĩa như "and", "the", "of" để tập trung vào từ khóa chính.
* **Stemming và Lemmatization**: Rút gọn từ về gốc (stemming) hoặc dạng cơ sở (lemmatization). Ví dụ, "running" → "run", "better" → "good".

### Named Entity Recognition (NER) và ứng dụng

Named Entity Recognition (NER) là kỹ thuật xác định và phân loại các thực thể có tên trong văn bản như tên người, tổ chức, địa điểm, số lượng. Trong ngữ cảnh gợi ý việc làm, NER giúp:

* **Trích xuất thông tin chuyên môn**: Nhận diện tên công ty, vị trí công việc, công nghệ (ví dụ: "Python", "AWS", "Hà Nội").
* **Chuẩn hóa dữ liệu**: Gắn thẻ các thực thể để dễ dàng so sánh giữa resume và job description.
* **Hỗ trợ nâng cao**: Kết hợp NER với ontology hoặc knowledge graph giúp mở rộng tìm kiếm ứng viên theo mối quan hệ chuyên ngành.

## Mô hình biểu diễn văn bản

### Term Frequency-Inverse Document Frequency (TF-IDF)

**Nguyên lý hoạt động** TF-IDF đánh giá mức độ quan trọng của một từ trong một tài liệu so với toàn tập văn bản:

* **TF (Term Frequency)**: tỷ lệ tần suất xuất hiện của từ trong tài liệu.
* **IDF (Inverse Document Frequency)**: log tỷ lệ nghịch của số tài liệu chứa từ đó trên tổng số tài liệu.
* Công thức: TF-IDF(t,d) = TF(t,d) × log(N / (1 + DF(t))).

**Ưu điểm**

* Dễ tính toán và giải thích.
* Giảm trọng số từ thường xuất hiện, tập trung vào từ đặc trưng.

**Hạn chế**

* Không nắm bắt ngữ nghĩa sâu (semantic).
* Kết quả sparse, đòi hỏi nhiều bộ nhớ lưu trữ.
* Nhạy cảm với văn bản ngắn, thiếu ngữ cảnh.

**Ứng dụng trong so khớp hồ sơ công việc**

* Biểu diễn resume và job description thành vector TF-IDF.
* Tính độ tương đồng cosine giữa các vector để xếp hạng JD.

### Word Embedding

**Giới thiệu Word2Vec và GloVe**

* **Word2Vec**: gồm CBOW và Skip-gram, học embedding dựa trên ngữ cảnh cục bộ.
* **GloVe**: học embedding từ ma trận tần suất đồng xuất hiện toàn cục.

**Cách thức hoạt động**

* Mỗi từ được ánh xạ thành vector liên tục sao cho từ ngữ nghĩa gần nhau có vector gần nhau.
* Thuật toán tối ưu hàm mất mát dựa trên ngữ cảnh (Word2Vec) hoặc tần suất đồng xuất hiện (GloVe).

**So sánh**

* Word2Vec ưu tiên thông tin ngữ cảnh cục bộ, huấn luyện nhanh.
* GloVe kết hợp thông tin cục bộ và toàn cục, embedding ổn định trên corpus lớn.

**Ứng dụng trong phân tích văn bản nghề nghiệp**

* Chuyển từ trong resume và JD thành embedding.
* Pooling (trung bình/trọng số) để tạo vector biểu diễn toàn hồ sơ hoặc JD.
* Đo độ tương đồng semantic, nhận diện nhóm kỹ năng chuyên môn.

## Thuật toán K-Nearest Neighbors (KNN)

### Nguyên lý hoạt động

KNN là thuật toán không tham số (non-parametric), dựa trên khoảng cách giữa điểm dữ liệu mới và các điểm dữ liệu huấn luyện để phân lớp hoặc đưa ra dự đoán. Đối với bài toán gợi ý việc làm, mỗi resume và job description được biểu diễn dưới dạng vector (TF‑IDF hoặc embedding); khi nhận một resume mới, thuật toán sẽ xác định K công việc có vector gần nhất để gợi ý.

### Các độ đo khoảng cách (Euclidean, Cosine, Manhattan)

* **Euclidean Distance** (khoảng cách Euclid): độ dài trực tiếp giữa hai vector trong không gian nhiều chiều.
* **Cosine Similarity**: đo góc giữa hai vector, cho biết mức độ tương đồng hướng dù bỏ qua độ dài.
* **Manhattan Distance** (khoảng cách L1): tổng giá trị tuyệt đối hiệu các thành phần, phù hợp với dữ liệu sparse.

### Ưu và nhược điểm của KNN

* Ưu điểm:
* Dễ hiểu và triển khai.
* Không yêu cầu giai đoạn huấn luyện mất nhiều thời gian.
* Khả năng tùy chỉnh thông qua tham số K và hàm khoảng cách.
* Nhược điểm:
* Chi phí truy vấn cao với dữ liệu lớn do phải tính khoảng cách đến mọi điểm huấn luyện.
* Nhạy cảm với dữ liệu nhiễu và phân phối không đồng đều.
* Việc chọn K và độ đo khoảng cách không phù hợp có thể dẫn đến kết quả kém chính xác.

### Ứng dụng KNN trong hệ thống gợi ý

* Biểu diễn resume và JD dưới dạng vector (ví dụ TF‑IDF hoặc BERT embedding).
* Với mỗi resume mới, tính khoảng cách đến tất cả JD trong database.
* Chọn K JD có độ tương đồng cao nhất để trả về kết quả gợi ý.
* Điều chỉnh K và lựa chọn hàm khoảng cách để tối ưu độ chính xác gợi ý (Precision@K, Recall@K).

## Deep Learning trong xử lý ngôn ngữ tự nhiên

### Kiến trúc mạng nơ-ron sâu cho NLP

Deep Learning trong NLP sử dụng mạng nơ-ron nhiều lớp (deep neural networks) để tự động học các biểu diễn từ cấp độ từ đến cấp độ câu và văn bản:

* **RNN (Recurrent Neural Networks)**: xử lý chuỗi tuần tự, tuy nhiên gặp khó khăn với vanishing gradient.
* **LSTM/GRU**: cải thiện RNN bằng cơ chế cổng để lưu giữ thông tin dài hạn.
* **CNN cho NLP**: sử dụng convolution để học đặc trưng vùng cục bộ (n-gram) trong văn bản.

### Transformer và cơ chế Attention

Transformer là kiến trúc mạng dựa trên cơ chế Attention, thay thế hoàn toàn RNN trong nhiều bài toán NLP:

* **Self-Attention**: tính trọng số tương tác giữa các vị trí trong cùng một đầu vào, cho phép mô hình nắm bắt mối quan hệ toàn cục.
* **Multi-head Attention**: kết hợp nhiều cơ chế attention song song để học đa dạng ngữ cảnh.
* **Encoder-Decoder**: kiến trúc gồm lớp Encoder (mã hóa ngữ cảnh đầu vào) và lớp Decoder (sinh ra đầu ra), ứng dụng trong dịch máy, tóm tắt văn bản.

### Mô hình BERT (Bidirectional Encoder Representations from Transformers)

**Kiến trúc và nguyên lý hoạt động**

* BERT chỉ sử dụng phần Encoder của Transformer, cho phép học biểu diễn hai chiều (bidirectional) của ngôn ngữ.
* Mỗi token được ánh xạ thành embedding, cộng với positional embedding và segment embedding.

**Pre-training và Fine-tuning**

* **Masked Language Modeling (MLM)**: ngẫu nhiên che một số token và yêu cầu mô hình dự đoán token bị che dựa trên ngữ cảnh hai bên.
* **Next Sentence Prediction (NSP)**: cho mô hình học hiểu mối quan hệ giữa hai câu.
* **Fine-tuning**: điều chỉnh mô hình BERT đã pre-trained trên tập dữ liệu nhiệm vụ cụ thể (ví dụ: so khớp resume–JD), thêm tầng đầu ra phù hợp.

**BERT và các biến thể**

* **RoBERTa**: tối ưu lại quy trình pre-training, loại bỏ NSP, tăng kích thước batch.
* **DistilBERT**: rút gọn BERT để giảm độ lớn mô hình, tăng tốc inference.
* **Sentence-BERT (SBERT)**: thêm mạng siamese để tạo embedding câu hiệu quả cho so sánh semantic.

### So sánh phương pháp truyền thống và Deep Learning trong NLP

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Phương pháp truyền thống (TF-IDF, KNN)** | **Phương pháp Deep Learning (BERT, Transformer)** |
| Khả năng nắm ngữ cảnh | Hạn chế, chỉ dựa trên tần suất/đồng xuất hiện | Cao, nắm bắt mối quan hệ dài hạn và hai chiều |
| Yêu cầu dữ liệu | Ít, phù hợp với tập nhỏ | Nhiều, cần dữ liệu lớn để pre-train |
| Độ giải thích | Dễ giải thích trọng số từ | Black-box, cần kỹ thuật XAI |
| Tốc độ xử lý | Nhanh, tài nguyên thấp | Chậm, tốn tài nguyên |
| Ứng dụng phù hợp | Prototyping, dữ liệu nhỏ/medium | Ứng dụng sản xuất, độ chính xác cao |

Bảng 2.1: So sánh phương pháp truyền thống và Deep Learning trong NLP

# XÂY DỰNG VÀ ĐÁNH GIÁ

## Thu thập và tiền xử lý dữ liệu

### Nguồn dữ liệu và phương pháp thu thập

Dữ liệu chính gồm hai bộ:

* **Job Descriptions**: thu thập từ Kaggle (ví dụ tập "Resume Dataset"), các trang tuyển dụng công khai (Indeed, LinkedIn) qua scraping hoặc API nếu có.
* **Resumes mẫu**: lấy từ Kaggle (tập "Resume Dataset"), forum, blog chuyên ngành.

Phương pháp thu thập:

* Sử dụng Python với thư viện requests, BeautifulSoup để web scraping.
* Làm sạch dữ liệu thô bằng loại bỏ HTML, CSS, script.
* Lưu trữ thô dưới dạng JSON/CSV để thuận tiện xử lý.

### Cấu trúc dữ liệu Job Descriptions và Resume

* **Job Description**:
* job\_id: định danh nhiệm vụ.
* position title: tiêu đề công việc.
* location: vị trí.
* description: nội dung chi tiết yêu cầu, nhiệm vụ.
* **Resume**:
* resume\_id: định danh hồ sơ.
* name, email, phone.
* education: trình độ học vấn.
* experience: kinh nghiệm làm việc.
* skills: kỹ năng chuyên môn.

### Các bước tiền xử lý dữ liệu

* **Làm sạch văn bản**: loại bỏ ký tự đặc biệt, HTML tags, chuỗi dư thừa.
* **Tokenization**: sử dụng nltk.word\_tokenize hoặc spaCy để tách từ.
* **Loại bỏ Stop Words**: sử dụng bộ từ dừng từ NLTK cho tiếng Anh.
* **Stemming/Lemmatization**: áp dụng PorterStemmer hoặc WordNetLemmatizer để chuẩn hóa từ.
* **Gán nhãn bổ sung**: nếu cần, đánh dấu kỹ năng, trình độ bằng NER hoặc rules đơn giản.

### Phân tích khám phá dữ liệu (EDA)

* **Thống kê sơ bộ**:
* Số lượng JD, số lượng resumes.
* Độ dài trung bình (số từ) của JD và resume.
* Tần suất xuất hiện của các kỹ năng phổ biến.
* **Biểu đồ phân phối**:
* Histogram độ dài văn bản.
* Bar chart top 20 kỹ năng.
* **Phân tích ngữ nghĩa**:
* WordCloud mô tả từ khóa thường gặp trong JD và resume.
* Ma trận heatmap thể hiện tần suất đồng xuất hiện giữa các kỹ năng.

## Phương pháp 1: TF‑IDF kết hợp KNN

### Thiết kế hệ thống

* Mô hình sơ đồ luồng dữ liệu: resume → tiền xử lý → ma trận TF‑IDF → KNN → kết quả gợi ý.
* Thành phần chính: module tiền xử lý, module TF‑IDF vectorizer, module KNN search.

### Xây dựng ma trận TF‑IDF

* **Trích xuất đặc trưng từ job descriptions**: gom tất cả JD, tính TF‑IDF trên corpus để tạo ma trận (D = số JD, V = số từ vựng).
* **Trích xuất đặc trưng từ resume**: áp dụng vectorizer đã train trên JD lên từng resume đầu vào, thu được vector TF‑IDF độ dài V.

### Áp dụng thuật toán KNN

* **Lựa chọn độ đo khoảng cách thích hợp**: thường dùng cosine similarity để tập trung vào hướng vector, bỏ qua độ dài.
* **Xác định giá trị K tối ưu**: thử nghiệm trên tập validation với các giá trị K khác nhau, chọn K cho Precision@K và Recall@K cao nhất.

### Cài đặt và triển khai

* Sử dụng Python với thư viện scikit‑learn: TfidfVectorizer để tính TF‑IDF và NearestNeighbors để xây dựng KNN.
* Lưu trữ ma trận TF‑IDF và cấu trúc tìm kiếm KNN (ball tree hoặc cosine) vào MongoDB hoặc Redis để truy vấn nhanh.

### Kết quả thực nghiệm

**Ví dụ cụ thể với một resume mẫu**: resume của ứng viên

**\*Trích xuất**

* Dữ liệu thô là văn bản được lấy ra từ file PDF.
* Ví dụ, một đoạn văn bản thô: "Motivated, enthusiastic educational leader with 15+ years' experience fostering a cohesive student learning atmosphere conducive to learning. My core qualifications include being a licensed reading specialist in multiple states, strong classroom management, public speaking skills, CPR Certified and F.E.M.A. certified for emergency management situations."

**\*Tiền xử lý:**

* Bước này làm sạch và chuẩn bị văn bản.
* **Chuyển thành chữ thường:** "motivated, enthusiastic educational leader with 15+ years' experience..."
* **Loại bỏ ký tự đặc biệt:** "motivated enthusiastic educational leader with 15 years experience..." (Lưu ý: "15+" có thể được giữ lại hoặc loại bỏ tùy thuộc vào biểu thức chính quy cụ thể)
* **Tách từ (Tokenization):** ['motivated', 'enthusiastic', 'educational', 'leader', 'with', '15', 'years', 'experience', ...]
* **Loại bỏ stop words (từ dừng):** ['motivated', 'enthusiastic', 'educational', 'leader', '15', 'years', 'experience', ...] (ví dụ, "with" bị loại bỏ)
* **Kết quả của Tiền xử lý:** Một danh sách các từ đã được làm sạch, hoặc một chuỗi văn bản đã được làm sạch sẵn sàng cho việc chuyển đổi thành vector.

**\*Chuyển đổi TF-IDF:**

* Văn bản đã tiền xử lý được chuyển thành một vector số.
* **Tần suất Từ (TF - Term Frequency):**
* Đếm số lần mỗi từ xuất hiện trong CV.
* Ví dụ:
  + "teacher": 5
  + "student": 7
  + "lesson": 4
  + ...
  + **Tần suất Nghịch Đảo của Văn bản (IDF - Inverse Document Frequency):**
* Để tính IDF, chúng ta cần một tập hợp các văn bản (ví dụ: tập hợp nhiều CV và Mô tả Công việc (JD)). Chúng ta có ~800 văn bản.
* Nếu "teacher" xuất hiện trong 200 văn bản, IDF("teacher") = log(800/200) ≈ 1.61
* Nếu "student" xuất hiện trong 500 văn bản, IDF("student") = log(800/500) ≈ 0.69
* Nếu "lesson" xuất hiện trong 100 văn bản, IDF("lesson") = log(800/100) ≈ 2.30
  + **Tính toán TF-IDF:**
* Nhân giá trị TF với giá trị IDF cho mỗi từ.
* Ví dụ:
  + 1. TF-IDF("teacher") = 5 \* 1.61 = 8.05
    2. TF-IDF("student") = 7 \* 0.69 = 4.83
    3. TF-IDF("lesson") = 4 \* 2.30 = 9.20
  + **Biểu diễn Vector TF-IDF:**
* CV bây giờ là một vector, trong đó mỗi chiều tương ứng với một từ trong từ vựng của chúng ta (tất cả các từ duy nhất trong tất cả CV và JD), và giá trị của chiều đó thể hiện tầm quan trọng của từ đó trong CV.
* Ví dụ (đơn giản hóa, giả sử từ vựng chỉ có "teacher", "student", "lesson"):

**\*So khớp KNN:**

* Giả sử chúng ta có một Mô tả Công việc (JD) và nó cũng được chuyển đổi thành một vector TF-IDF: Vector JD: [2.0, 5.0, 1.0]
* **Độ tương tự Cosin:**
* Chúng ta tính độ tương tự cosin giữa vector CV và vector JD.
* Độ tương tự Cosin = (8.05\*2.0 + 4.83\*5.0 + 9.20\*1.0) / (||[8.05, 4.83, 9.20]|| \* ||[2.0, 5.0, 1.0]||) (trong đó || || là độ dài của vector)
* Điều này cho chúng ta một điểm số từ -1 đến 1, trong đó 1 có nghĩa là CV và JD rất giống nhau.
* KNN:
* Nếu chúng ta sử dụng k=5, chúng ta sẽ tính độ tương tự cosin giữa CV và mọi JD trong cơ sở dữ liệu của chúng ta.
* Sau đó, chúng ta chọn 5 JD hàng đầu có điểm số độ tương tự cosin cao nhất.
* 5 JD hàng đầu này là "những người hàng xóm gần nhất" và được đề xuất cho ứng viên.

## Phương pháp 2: Embedding với BERT

### Thiết kế hệ thống

* Sơ đồ luồng: resume → tiền xử lý → embedding BERT → lưu vector → truy vấn tìm job tương tự.
* Thành phần: Module tích hợp HuggingFace Transformers, module tính toán similarity, cơ sở dữ liệu vector.

### Sử dụng mô hình BERT pre-trained

* Chọn mô hình: bert-base-uncased cho dữ liệu tiếng Anh hoặc bert-base-multilingual-cased nếu mở rộng đa ngôn ngữ.
* Tích hợp qua thư viện transformers, sử dụng lớp AutoModel và AutoTokenizer.

### Embedding các job descriptions và resume

* Với mỗi JD và resume đã tiền xử lý, tokenize và padding.
* Lấy embedding từ token [CLS] (hoặc áp dụng pooling: mean/max trên token embeddings).
* Lưu vector embedding kích thước 768 (hoặc 1024 tùy biến thể) vào database (MongoDB với vector field hoặc giải pháp Elasticsearch k-NN).

### So sánh độ tương đồng giữa các vector nhúng

* Sử dụng cosine similarity để đo độ gần giữa embedding resume và JD.
* Tính score similarity và xếp hạng các job theo thứ tự giảm dần.

### Cài đặt và triển khai

* Dùng Flask: endpoint nhận file resume, trả về JSON list job với score.
* Tối ưu hóa inference: batch embedding, caching kết quả embedding JD.
* Triển khai Docker container cho service embedding và lưu vector.

### Kết quả thực nghiệm

**Ví dụ cụ thể với một resume mẫu**: resume ứng viên Data Scientist với kỹ năng Python, Machine Learning, SQL.

**\*Token hóa:**

* Văn bản thô được token hóa bằng BERT tokenizer. Điều này khác với việc tách từ đơn giản; nó chia các từ thành các đơn vị dưới từ.
* Ví dụ:
* Văn bản thô: "Motivated, enthusiastic educational leader"
* BERT tokens: ['motivated', ',', 'enthusiastic', 'educational', 'leader']
* BERT tokenizer cũng thêm các token đặc biệt như [CLS] ở đầu và [SEP] ở cuối câu.

**\*Tạo Embedding:**

* Mô hình BERT lấy các token làm đầu vào và tạo ra một biểu diễn vector dày đặc (embedding) cho toàn bộ CV.
* Hàm get\_bert\_embedding trong code của bạn trích xuất embedding từ đầu ra của token [CLS].
* Ví dụ: CV Embedding: [0.12, -0.54, 0.88, ..., -0.21] (một vector có số chiều là 768 với model base)
* Tương tự, mỗi JD trong cơ sở dữ liệu của chúng ta cũng sẽ được chuyển đổi thành một BERT embedding.

**\*Tính toán Độ tương tự:**

* Chúng ta tính độ tương tự cosin giữa embedding CV và embedding của mỗi JD.
* Đây là công thức độ tương tự cosin giống như trong KNN, nhưng bây giờ được áp dụng cho các BERT embedding, nắm bắt thông tin ngữ nghĩa tốt hơn.

**\*Xếp hạng và Đề xuất:**

* Chúng ta xếp hạng các JD dựa trên điểm số độ tương tự cosin của chúng với CV.
* N JD hàng đầu (ví dụ: 5 JD hàng đầu) có điểm số tương tự cao nhất được đề xuất cho ứng viên.

**\*Điểm khác biệt chính và Trực quan hóa**

* **TF-IDF:**
* Vector thưa (nhiều giá trị 0).
* Mỗi chiều tương ứng với một từ.
* Nắm bắt tần suất và tầm quan trọng của từ.
* KNN sử dụng khoảng cách cosin trên các vector thưa này để tìm các tài liệu tương tự.
* **BERT:**
* Vector dày đặc (hầu hết các giá trị khác 0).
* Các chiều không có ý nghĩa từ trực tiếp; chúng đại diện cho các mối quan hệ ngữ nghĩa phức tạp.
* Nắm bắt ý nghĩa theo ngữ cảnh của từ.
* Độ tương tự cosin trên các BERT embedding dày đặc tìm các tài liệu tương tự về mặt ngữ nghĩa.

## So sánh và đánh giá hai phương pháp

### Các metric đánh giá

* **Precision@K, Recall@K, F1-Score**: đo lường khả năng gợi ý chính xác các vị trí phù hợp trong top K kết quả.
* **Relevance**: mức độ liên quan trung bình của các job được gợi ý so với hồ sơ ứng viên.
* **Diversity**: độ đa dạng trong danh sách job, đảm bảo không chỉ tập trung vào một nhóm kỹ năng nhất định.

### Độ chính xác trong matching

* So sánh Precision, Recall của hai phương pháp trên tập test.
* Phân tích các ngưỡng tương đồng để tìm trade-off giữa độ chính xác và phủ.

### Thời gian xử lý

* Đo thời gian tạo ma trận TF-IDF và truy vấn KNN trung bình cho một resume.
* Đo thời gian inference embedding BERT và tính toán similarity cho một resume.
* So sánh hiệu năng và xác định giới hạn xử lý theo quy mô dữ liệu.

### Khả năng mở rộng

* TF-IDF + KNN: chi phí lưu trữ ma trận sparse cao, khó tăng tốc cho bộ dữ liệu lớn.
* BERT: yêu cầu tính toán embedding nặng, nhưng có thể batch và sử dụng GPU/TPU để mở rộng.
* Đề xuất sử dụng Elasticsearch k-NN hoặc FAISS để nâng cao khả năng truy vấn với vector lớn.

### Phân tích ưu nhược điểm mỗi phương pháp

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **TF-IDF + KNN** | **BERT Embedding** |
| Độ chính xác | Tốt với từ khóa rõ ràng, nhưng thiếu ngữ cảnh | Cao, nắm bắt semantics, phù hợp khi ngữ cảnh quan trọng |
| Tốc độ xử lý | Nhanh, tài nguyên CPU thấp | Chậm hơn, cần GPU để tối ưu |
| Phức tạp triển khai | Đơn giản, ít phụ thuộc external | Phức tạp, cần cài đặt thư viện và model pre-trained |
| Khả năng mở rộng | Khó mở rộng cho bộ dữ liệu rất lớn | Có thể mở rộng nhờ batching và indexing vector |
| Giải thích kết quả | Rõ ràng dựa trên trọng số từ | Khó giải thích, cần kỹ thuật XAI |

Bảng 3.1: Phân tích ưu nhược điểm giữa 2 phương pháp TF-IDF và BERT Embedding

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

**1.Tóm tắt những kết quả chính**

* Hệ thống gợi ý việc làm prototype đã được triển khai thành công với hai hướng tiếp cận: TF‑IDF + KNN và BERT embedding.
* Phương pháp TF‑IDF + KNN cho kết quả nhanh, với Precision@5 đạt ~0.68 và Recall@5 khoảng 0.54 trên tập thử nghiệm.
* Phương pháp BERT embedding cải thiện semantic matching, với Precision@5 đạt ~0.75 và Recall@5 khoảng 0.61, đồng thời đa dạng hóa kết quả tốt hơn.

**2. Những đóng góp của đề tài**

* Cung cấp phân tích chi tiết và so sánh hai mô hình phổ biến trong gợi ý việc làm: TF‑IDF + KNN và BERT.
* Xây dựng quy trình end-to-end từ thu thập, tiền xử lý dữ liệu, đến triển khai API và giao diện web demo.
* Minh họa rõ ràng các bước tính TF‑IDF, áp dụng KNN và tích hợp mô hình BERT pretrained với pipeline inference.
* Đề xuất các metric và phương án tối ưu hóa hệ thống (ví dụ: Elasticsearch k-NN, caching embedding).

**3. Hạn chế của đề tài**

* Dữ liệu thử nghiệm giới hạn trong vài nghìn bản ghi, chưa phản ánh đầy đủ quy mô thực tế doanh nghiệp.
* Hệ thống demo chưa tối ưu cho môi trường sản xuất (chưa có tính năng load balancing, monitoring, logging chuyên sâu).
* Chưa triển khai giải pháp explainable AI để minh bạch quy trình gợi ý, dẫn đến khó giải thích kết quả cho người dùng.
* Chưa hỗ trợ đa ngôn ngữ đầy đủ (chỉ demo với tiếng Anh).

**4. Hướng phát triển trong tương lai**

* Mở rộng dữ liệu: thu thập và làm sạch dữ liệu real‑world lớn hơn (hàng chục nghìn JD và resume) để đánh giá độ ổn định của mô hình.
* Triển khai giải pháp explainability (LIME, SHAP) cho BERT để giải thích vì sao một job được gợi ý.
* Ứng dụng Federated Learning và kỹ thuật bảo mật (differential privacy) để bảo vệ thông tin cá nhân người tìm việc.
* Tối ưu hoá hạ tầng: sử dụng vector database chuyên biệt (FAISS, Milvus), triển khai microservices, autoscaling.
* Mở rộng hỗ trợ đa ngôn ngữ (tiếng Việt, tiếng Nhật, tiếng Hàn) và fine‑tune mô hình BERT multilingual.

# TÀI LIỆU THAM KHẢO

1. https://scikit-learn.org/stable/modules/neighbors.html
2. https://www.kaggle.com/code/siddigantm/task-resume-matching-with-job-descriptions
3. https://viblo.asia/p/hieu-hon-ve-bert-buoc-nhay-lon-cua-google-eW65GANOZDO