

ĐẠI HỌC QUỐC GIA TP.HCM  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



**Môn học:** KHAI PHÁ DỮ LIỆU TRONG DOANH NGHIỆP

**LỚP:** DS317.P11

## **Báo cáo đề tài**

**GVHD:** ThS. Nguyễn Thị Anh Thư

*Nhóm sinh viên thực hiện:*

Nguyễn Hữu Nam	MSSV: 22520917
Nguyễn Khánh	MSSV: 22520641
Võ Đình Khánh	MSSV: 22520659
Nguyễn Minh Sơn	MSSV: 22521254
Bùi Hồng Sơn	MSSV: 22521246



# Mục lục

<b>1</b>	<b>Tổng quan</b>	<b>3</b>
1.1	Định nghĩa và ngữ cảnh bài toán	3
1.2	Ứng dụng	4
1.3	Khó khăn và thách thức	4
1.4	Các nghiên cứu liên quan	5
<b>2</b>	<b>Các công trình nghiên cứu liên quan</b>	<b>5</b>
2.1	Matrix Factorization	5
2.2	Collaborative Filtering	6
2.3	Content-Based Filtering	6
2.4	Graph-Based Recommender Systems	6
2.5	Các hệ thống khuyến nghị khóa học dựa trên dữ liệu lớn	7
2.6	Ứng dụng của các mô hình học sâu	7
2.7	Tổng quan các công trình nghiên cứu	7
<b>3</b>	<b>Cơ sở lý thuyết</b>	<b>7</b>
3.1	Phương pháp áp dụng - KGAT	7
3.1.1	Các kiến thức cần nắm	8
3.1.2	Tổng quan về KGAT	9
3.1.3	Các kĩ thuật sử dụng của KGAT	10
3.2	Phương pháp khác	13
3.2.1	Content-based Filtering	13
3.2.2	Matrix Factorization - Bayesian Personalized Ranking	13
3.2.3	Factorization Machine	14
3.2.4	Neutral Factorization Machine	15
3.3	Phương pháp tiếp cận cổ điển	16
3.3.1	Content-based filtering (Utility Matrix)	16
<b>4</b>	<b>Phương pháp đề xuất</b>	<b>19</b>
4.1	Mô hình	19



<b>5</b>	<b>Thực nghiệm</b>	<b>21</b>
5.1	Miêu tả bộ dữ liệu . . . . .	21
5.2	Phương pháp tổ chức dữ liệu thực nghiệm . . . . .	21
5.2.1	Dịch bảng . . . . .	22
5.2.2	Khám phá dữ liệu . . . . .	24
5.2.3	Làm sạch dữ liệu . . . . .	42
5.2.4	Chuyển đổi dữ liệu . . . . .	48
5.2.5	Chia tập dữ liệu . . . . .	50
5.3	Độ đo đánh giá . . . . .	50
5.4	Kịch bản thực nghiệm . . . . .	50
5.5	Đánh giá kết quả thực nghiệm . . . . .	50
<b>6</b>	<b>Kết luận và hướng phát triển</b>	<b>50</b>
6.1	Đánh giá các phương pháp . . . . .	50
6.2	Hướng phát triển tiềm năng . . . . .	54



## 1. Tổng quan

Khai phá dữ liệu, đặc biệt là dữ liệu lớn, đã trở thành một lĩnh vực nghiên cứu quan trọng và thu hút sự quan tâm của các nhà khoa học trong những năm gần đây. Các ứng dụng của khai phá dữ liệu rất đa dạng, được triển khai trong nhiều lĩnh vực như kinh doanh, giáo dục, y tế, tài chính, và ngân hàng. Đặc biệt, khai phá dữ liệu trong giáo dục, cụ thể là khai phá dữ liệu lớn, đang là chủ đề thu hút nhiều nghiên cứu nhờ vào tính ứng dụng cao và tiềm năng cải thiện chất lượng giáo dục.

Trong bối cảnh giáo dục trực tuyến hiện nay, người học cần phải tự chủ động và có tinh thần tự giác cao do số lượng môn học đa dạng thuộc nhiều lĩnh vực khác nhau. Họ cần phân bổ thời gian học tập hợp lý cho từng nhóm môn học nhằm bổ sung và nâng cao kiến thức chuyên ngành cần thiết. Tuy nhiên, các nền tảng học tập trực tuyến thường không có ràng buộc cụ thể về thời gian và điểm số, dẫn đến tình trạng nhiều khóa học không được hoàn thành đúng thời hạn, thậm chí bị bỏ dở do người học mất hứng thú.

Vì vậy, công tác cố vấn học tập trên các nền tảng trực tuyến trở nên vô cùng quan trọng để giúp người học cải thiện hiệu suất học tập và gợi ý các khóa học phù hợp với nhu cầu cá nhân. Đây là một bài toán thuộc lĩnh vực khai phá dữ liệu, đặc biệt khi xử lý với số lượng lớn dữ liệu liên quan đến người học và hành vi học tập của họ. Việc nghiên cứu và xây dựng hệ thống khuyến nghị khóa học góp phần quan trọng vào việc cá nhân hóa trải nghiệm học tập, hỗ trợ người dùng lựa chọn các khóa học phù hợp với mục tiêu và nhu cầu học tập.

### 1.1. Định nghĩa và ngữ cảnh bài toán

Trong bối cảnh các nền tảng học tập trực tuyến, người học thường gặp khó khăn trong việc lựa chọn khóa học phù hợp. Điều này đặt ra nhu cầu xây dựng một hệ thống khuyến nghị giúp cá nhân hóa quá trình học tập của từng người. Bài toán được định nghĩa với đầu vào và đầu ra như sau:

- **Input:** Dữ liệu lớn từ các nền tảng học tập trực tuyến, bao gồm thông tin về người học, thông tin khóa học, và dữ liệu về các hoạt động học tập của người dùng.



- **Output:** Đề xuất top- $k$  khóa học phù hợp nhất với người dùng (trong đó  $k \in \mathbb{N}^*$ , ví dụ trong nghiên cứu này  $k = 10$ ).

## 1.2. Ứng dụng

Bài toán khuyến nghị khóa học cho các nền tảng học tập trực tuyến có nhiều ứng dụng thực tiễn, bao gồm:

- **Cá nhân hóa quá trình học tập:** Hệ thống giúp người học lựa chọn các khóa học phù hợp với nhu cầu và trình độ, từ đó cá nhân hóa lộ trình học tập.
- **Tăng tỷ lệ hoàn thành khóa học:** Đề xuất khóa học phù hợp giúp người học duy trì động lực học tập, từ đó tăng tỷ lệ hoàn thành khóa học.
- **Tối ưu hóa lộ trình học tập:** Gợi ý các khóa học tiếp theo dựa trên kỹ năng hiện tại và các khóa học đã hoàn thành.
- **Ứng dụng trong đào tạo doanh nghiệp:** Hỗ trợ doanh nghiệp xây dựng chương trình đào tạo nhân viên hiệu quả, phù hợp với mục tiêu phát triển nghề nghiệp.
- **Nâng cao hiệu quả sử dụng tài nguyên:** Giúp người học tiết kiệm thời gian và tập trung vào các khóa học có giá trị cao.

## 1.3. Khó khăn và thách thức

Mặc dù có nhiều tiềm năng, bài toán khuyến nghị khóa học vẫn gặp phải các khó khăn và thách thức như:

- **Chất lượng và sự đa dạng của dữ liệu:** Dữ liệu không đồng nhất hoặc không đầy đủ, gây khó khăn trong phân tích hành vi người học.
- **Xử lý dữ liệu lớn:** Khối lượng dữ liệu lớn đòi hỏi khả năng tính toán mạnh mẽ và các thuật toán tối ưu.
- **Lựa chọn đặc trưng quan trọng:** Việc chọn lọc các đặc trưng phù hợp từ bộ dữ liệu lớn đòi hỏi sự cân nhắc về tài nguyên và thời gian.



- **Đánh giá mô hình:** Thiếu dữ liệu rõ ràng về mức độ hài lòng của người học, khiến việc đánh giá hệ thống trở nên khó khăn.

## 1.4. Các nghiên cứu liên quan

Các phương pháp khuyến nghị chính bao gồm:

- **Matrix Factorization:** Phân rã ma trận để tìm các yếu tố tiềm ẩn.
- **Collaborative Filtering:** Sử dụng thông tin tương đồng giữa người dùng hoặc khóa học.
- **Content-Based Filtering:** Gợi ý dựa trên nội dung và đặc trưng của khóa học.
- **Hybrid Systems:** Kết hợp nhiều phương pháp để tăng hiệu quả.
- **Graph-Based Methods:** Sử dụng đồ thị để biểu diễn mối quan hệ giữa người dùng và khóa học.
- **Neural Collaborative Filtering:** Ứng dụng Deep Learning để học các tương tác phi tuyến.

## 2. Các công trình nghiên cứu liên quan

Trong lĩnh vực khuyến nghị khóa học, nhiều công trình nghiên cứu đã được thực hiện nhằm cải thiện hiệu quả và độ chính xác của các hệ thống khuyến nghị. Một số phương pháp nổi bật được đề xuất như sau:

### 2.1. Matrix Factorization

Matrix Factorization (MF) là một kỹ thuật phổ biến trong hệ thống khuyến nghị, được sử dụng để phân rã ma trận user-item nhằm phát hiện các yếu tố tiềm ẩn ảnh hưởng đến hành vi của người dùng. Koren và cộng sự (2009) đã giới thiệu phương pháp này và áp dụng vào hệ thống khuyến nghị, đặc biệt trong bài toán đề xuất phim. Phương pháp này cho phép mô hình hóa sự tương tác giữa người



dùng và các khóa học dựa trên các đặc trưng tiềm ẩn, mang lại kết quả tốt trong nhiều bài toán thực tế.

## 2.2. Collaborative Filtering

Collaborative Filtering (CF) là một trong những phương pháp lâu đời nhất và hiệu quả nhất trong khuyến nghị. CF được chia thành hai nhánh chính:

- **User-User Collaborative Filtering:** Dựa trên sự tương đồng giữa các người dùng để gợi ý các khóa học mà người dùng có thể quan tâm.
- **Item-Item Collaborative Filtering:** Dựa trên sự tương đồng giữa các khóa học để gợi ý các khóa học mới cho người dùng.

Su và Khoshgoftaar (2009) đã thực hiện một khảo sát toàn diện về CF và các phương pháp cải tiến nhằm khắc phục những hạn chế của nó, bao gồm vấn đề thưa thớt dữ liệu và mở rộng quy mô.

## 2.3. Content-Based Filtering

Content-Based Filtering (CBF) là phương pháp tập trung vào các đặc trưng của khóa học, dựa trên thông tin về nội dung của các khóa học mà người dùng đã tham gia để đưa ra gợi ý. Burke (2002) đã tổng hợp các phương pháp CBF và chỉ ra rằng việc kết hợp các đặc trưng nội dung của khóa học và hành vi người dùng có thể nâng cao hiệu quả gợi ý.

## 2.4. Graph-Based Recommender Systems

Phương pháp dựa trên đồ thị (Graph-Based Recommender Systems) sử dụng cấu trúc đồ thị để biểu diễn mối quan hệ giữa người dùng và khóa học. Các thuật toán như Random Walk hoặc PageRank được áp dụng để tìm kiếm và khai thác các kết nối trong dữ liệu. Phương pháp này đặc biệt hiệu quả khi xử lý dữ liệu phức tạp với nhiều mối quan hệ đa chiều.



## 2.5. Các hệ thống khuyến nghị khóa học dựa trên dữ liệu lớn

Trong bối cảnh dữ liệu lớn, các nền tảng như MOOCCubeX cung cấp một lượng dữ liệu khổng lồ về hành vi học tập của người dùng. Zhang và cộng sự (2022) đã phát triển một hệ thống khuyến nghị dựa trên dữ liệu từ MOOCCubeX, sử dụng các mô hình đồ thị để cá nhân hóa lộ trình học tập. Kết quả cho thấy hệ thống có khả năng gợi ý các khóa học phù hợp và tăng tỷ lệ hoàn thành khóa học.

## 2.6. Ứng dụng của các mô hình học sâu

Học sâu (Deep Learning) được áp dụng rộng rãi trong các hệ thống khuyến nghị khóa học hiện đại. Các mạng nơ-ron tích chập (Convolutional Neural Networks - CNNs) và mạng nơ-ron hồi tiếp (Recurrent Neural Networks - RNNs) được sử dụng để phân tích các chuỗi hành vi học tập của người dùng. Ngoài ra, Transformer và mô hình Attention cũng được triển khai để tối ưu hóa việc gợi ý dựa trên các đặc trưng tuần tự và ngữ cảnh.

## 2.7. Tổng quan các công trình nghiên cứu

Tóm lại, các phương pháp khuyến nghị khóa học đã có nhiều bước tiến đáng kể nhờ vào việc kết hợp các kỹ thuật truyền thống và hiện đại, từ Matrix Factorization đến Neural Collaborative Filtering. Các nghiên cứu không chỉ tập trung vào việc cải thiện độ chính xác mà còn nhấn mạnh đến khả năng mở rộng, xử lý dữ liệu lớn, và cá nhân hóa trải nghiệm học tập cho từng người dùng.

# 3. Cơ sở lý thuyết

## 3.1. Phương pháp áp dụng - KGAT

Phương pháp tiếp cận gần đây - KGAT: Knowledge Graph Attention Network for Recommendation





Bài báo đề xuất một phương pháp để cải thiện độ chính xác, đa dạng và khả năng giải thích của hệ thống gợi ý, cần phải xem xét thêm thông tin bổ sung thay vì chỉ dựa vào tương tác người dùng-sản phẩm. Thường thì các phương pháp truyền thống như học máy không đủ để nắm bắt các mối quan hệ phức tạp giữa các mục, từ đó cho ra kết quả dự đoán kết quả không được cao. Trong nghiên cứu này, nhóm tác giả sử dụng đồ thị tri thức (Knowledge Graph) để liên kết các mục với thuộc tính của chúng, cho rằng các quan hệ bậc cao là chìa khóa cho việc gợi ý hiệu quả. Tổng quan phương pháp này được gọi là Knowledge Graph Attention Network for Recommendation (KGAT) [3], mô hình hóa rõ ràng các kết nối bậc cao và sử dụng cơ chế chú ý để đánh giá tầm quan trọng của các liên kết. Kết quả thực nghiệm cho thấy KGAT vượt trội hơn các phương pháp tiên tiến hiện có và cung cấp khả năng giải thích tốt hơn.

KGAT là một trong những hệ thống đề xuất tập trung nhiều hơn về những thông tin ẩn về người dùng. Bằng cách kết hợp những nhiều trường thông tin ẩn vào mô hình, nhóm tác giả đã tạo được một mô hình không chỉ hiểu về những thông tin rõ ràng đã có của người dùng mà còn các tương tác ẩn giữa người dùng với các đối tượng đó.

### 3.1.1. Các kiến thức cần nắm

Bài báo KGAT đề xuất một phương pháp xử lý dữ liệu tốt hơn bằng cách tận dụng trường thông tin bổ sung (Item side information) để tăng hiệu suất của mô hình.

Giải quyết bài toán đề xuất với cách tiếp cận là tạo ra một mô hình có mối quan hệ bậc cao rõ ràng giữa các trường thông tin và toàn diện với cách xử lý của một mạng thần kinh đồ thị (Graph Neural Network [8] [9] – GNN).

Tiến hành mở rộng và so sánh với các phương pháp kinh điển trước đó để chứng minh được hiệu suất rõ ràng của mô hình.

Về tổng quát, có thể thấy KGAT tập trung phát triển và cải thiện các kĩ thuật xử lý dữ liệu dựa trên GNN. Vậy GNN là gì? Cốt lõi của GNN là gì? Đáp án của những câu hỏi trên nằm ở một câu trả lời duy nhất, đó là đồ thị tri thức (Knowledge Graph - KG). KG có thể được hiểu là một mạng phức tạp liên kết các thực thể - chẳng hạn như giữa người dùng (users), vật phẩm (items) và các

đặc trưng (features) của chúng thông qua mối quan hệ phức tạp của chúng.

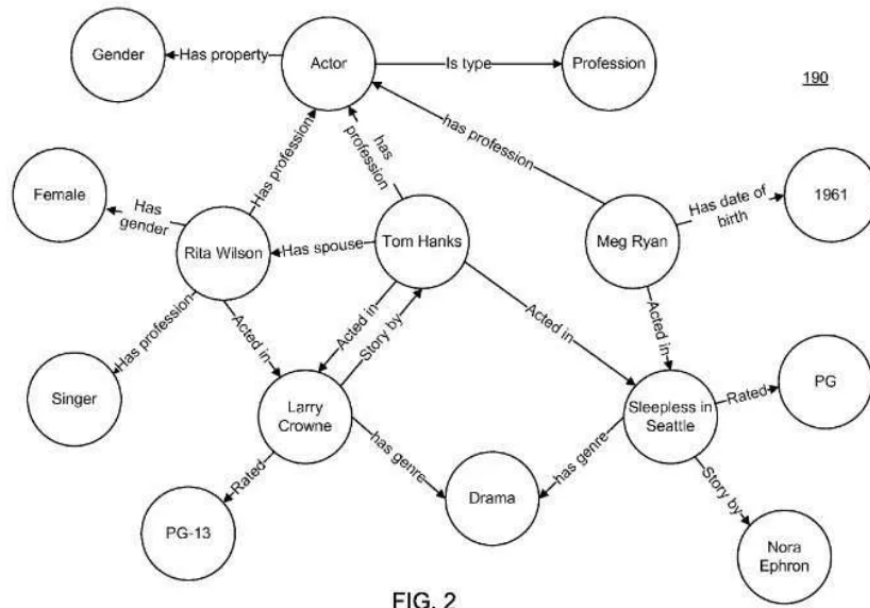


FIG. 2

Hình 1: Biểu đồ tri thức KG trong thực tế

Thường thì, các thành phần cấu tạo nên một KG sẽ bao gồm:

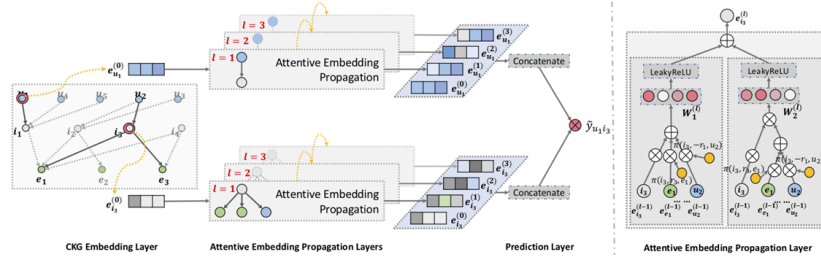
- Nút: đại diện cho các thực thể (ví dụ: người dùng, sản phẩm, danh mục, ...)
- Cạnh: mô tả mối quan hệ của các nút (ví dụ: đã mua, đã xem, ...). Và ngược lại)

Có thể thấy đây là một kiến trúc mạng linh hoạt có nhiều lớp thông tin có thể được bổ sung và tích hợp với nhau, điều này cho phép một mạng KG có thể hiểu được toàn diện về ngữ cảnh và môi trường của người dùng nếu được tổ chức tốt.

### 3.1.2. Tổng quan về KGAT

Mô hình KGAT sử dụng thông tin bổ sung để xây dựng một biểu đồ tri thức nhằm nắm bắt mối quan hệ giữa người dùng, các mục (items) và các tương tác người dùng (user interaction) với nhau. Mô hình KGAT sử dụng cơ chế học chú ý (Attention Mechanism) với đồ thị nhằm cho phép mô hình có thể hiểu được tầm

quan trọng về sự khác nhau giữa các vật phẩm và những đặc trưng của chúng. Điểm khác biệt giữa mô hình KGAT so với các mô hình trước đây là nó được thiết kế với sự thay đổi nhằm làm phong phú thêm về sự hiểu biết của mô hình về hành vi và sở thích của người dùng. Điều này dẫn đến những dự đoán không chỉ dựa trên những thuộc tính của các mục (items) mà còn được định hình dự trên đặc điểm của từng người dùng, giúp cải thiện đáng kể độ phù hợp và liên quan giữa các mục được mô hình đề xuất.



Hình 2: Kiến trúc tổng quan mô hình khuyến nghị học sâu KGAT

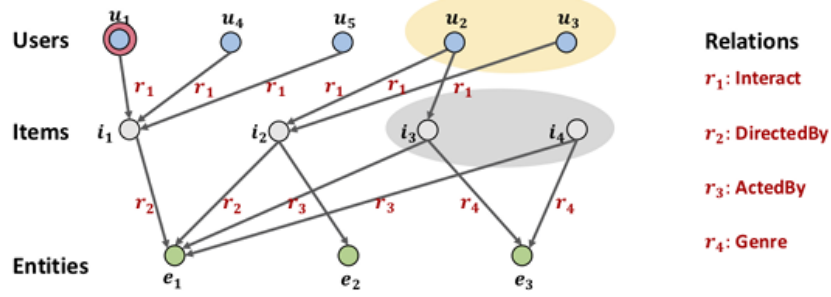
### 3.1.3. Các kĩ thuật sử dụng của KGAT

**Embedding layer.** Về mặt lý thuyết, tác giả tạo một collaborative knowledge graph – CKG. CKG là sự kết hợp giữa một KG và một user-item bipartite graph. CKG có thể được hiểu là một dạng mở rộng của một KG, trong đó thông tin không chỉ đến từ thông tin của User và Item mà còn đến từ các thông tin bổ trợ khác (ở hình sau là các Entities). Mục tiêu của một CKG là sử dụng được thông tin phong phú từ nhiều nguồn để cải thiện độ chính xác và tăng khả năng dự đoán của mô hình.

Điểm hay của KGAT là khai thác được các quan hệ bậc cao trong CKG, ví dụ như các kết nối dài hạn sau:

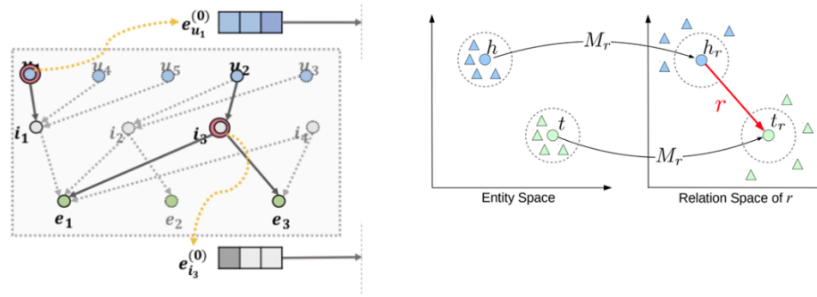
- $u_1 \ r_1 \rightarrow i_1 \ -r_2 \rightarrow e_1 \ r_2 \rightarrow i_2 \ -r_1 \rightarrow \{u_2, u_3\}$
- $u_1 \ r_1 \rightarrow i_1 \ -r_2 \rightarrow e_1 \ r_3 \rightarrow \{i_3, i_4\}$

Về cách thức thực hiện, embedding layer tham số hóa các thực thể và quan hệ dưới dạng các vectors trong khi vẫn bảo toàn cấu trúc đồ thị. Trong bài báo,



Hình 3: Một CKG biểu diễn liên kết cho các loại thực thể

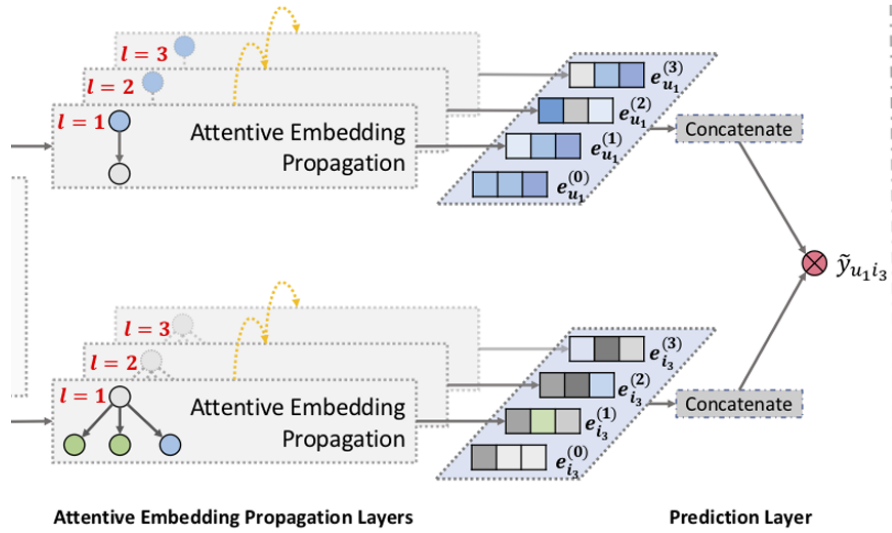
nhóm tác giả sử dụng TransR [10] để tham số hóa các thực thể và mối quan hệ trong CKG thành các biểu diễn vector, xem xét sự biểu diễn của chúng với kết nối trực tiếp của mỗi bộ ba phần tử ( $h, r, t$ ):



Hình 4: Cơ chế embeddings (bên trái) và TranR (bên phải) trong KGAT

**Attentive Embedding propagation layers.** Về lý thuyết, đây được xem như giai đoạn biểu diễn mỗi nút dưới dạng một vector bằng cấu trúc bảo toàn cấu trúc của CKG, đệ quy lan truyền để truyền các lớp nhúng học được từ các nút lân cận để cập nhật biểu diễn cho nó, sử dụng cơ chế học có chú ý (Attention) để học chi tiết từng trọng số cho các nút lân cận trong quá trình lan truyền.

Cách tiếp cận có hệ thống này biến mô hình KGAT thành một hệ thống kết hợp nắm bắt cả các thông tin items trực tiếp của người dùng và bối cảnh rộng hơn của các tùy chọn đó, tạo ra một hệ thống đề xuất rõ ràng hơn có khả năng mang được nhiều thông tin để dự đoán hơn.



Hình 5: Cơ chế Attention và cơ chế truyền thông tin giữa các lớp

**Model prediction.** Sau khi thực hiện lan truyền thông tin qua  $L$  lớp, ta thu được nhiều đại diện của người dùng node  $u$ , bao gồm:  $\{e_u(1), \dots, e_u(L)\}$ ; tương tự với sản phẩm node  $i$ , ta cũng thu được  $e_i(1), \dots, e_i(L)$ . Tiếp đến, ta sẽ ghép nối embedding ban đầu với các embeddings mới thu được theo công thức sau:

$$eu^* = eu(0) || \dots || eu(L), \quad ei^* = ei(0) || \dots || ei(L)$$

Cuối cùng, điểm số phù hợp (matching score) giữa người dùng và sản phẩm sẽ được tính bằng inner product:

$$y_{u,i} = eu^* Tei^*$$

Về cách thức tối ưu, KGAT sử dụng loss function chính xác đối với thói quen tương tác người dùng.



## 3.2. Phương pháp khác

### 3.2.1. Content-based Filtering

### 3.2.2. Matrix Factorization - Bayesian Personalized Ranking

Bayesian Personalized Ranking (BPR) là một phương pháp học máy được thiết kế đặc biệt để cải thiện hệ thống gợi ý, đặc biệt là trong bối cảnh dữ liệu phản hồi ngầm (implicit feedback) như lượt xem, lượt mua, hay lượt thích.

Khi kết hợp với Matrix Factorization (MF), BPR đã chứng minh hiệu quả vượt trội trong việc cá nhân hóa đề xuất sản phẩm cho người dùng. Trong bối cảnh BPR, dữ liệu thường là dạng phản hồi ngầm, tức là chỉ biết được các tương tác tích cực (ví dụ: người dùng đã mua hoặc xem sản phẩm nào đó) mà không có thông tin về các tương tác tiêu cực. BPR giả định rằng những mục không được tương tác là không được ưu tiên hoặc ít được ưu tiên hơn so với những mục đã được tương tác.

Matrix Factorization là kỹ thuật phổ biến để phân rã ma trận người dùng mục thành các ma trận tiềm ẩn nhỏ hơn, đại diện cho đặc trưng của người dùng và mục. Trong ngữ cảnh BPR

- Ma trận người dùng (U): Đại diện cho các đặc trưng tiềm ẩn của người dùng.
- Ma trận mục (I): Đại diện cho các đặc trưng tiềm ẩn của mục.

BPR dựa trên giả định rằng, với mỗi người dùng  $u$ , một mục tích cực  $i$  (đã được tương tác) nên được ưu tiên hơn một mục tiêu cực  $j$  (chưa được tương tác). Mục tiêu của BPR là tối ưu hóa hàm mục tiêu:

$$\sum_{(u,i,j) \in D_s} \ln \sigma(\hat{x}_{uij}) - \lambda_{\theta} \|\theta\|^2$$

Với  $\hat{x}_{uij}$  là điểm khác biệt giữa điểm số dự đoán  $\hat{x}_{ui}$  và  $\hat{x}_{uj}$

BPR thường sử dụng Stochastic Gradient Descent để tối ưu



### 3.2.3. Factorization Machine

Một trong những nhược điểm chính của mô hình BPRMF là không có khả năng mô hình hóa những thông tin bổ trợ giữa người dùng và sản phẩm. Do đó, một phương pháp mở rộng FM đã ra đời. Đây cũng là phương pháp nền móng cho các kĩ thuật DL được ra đời cho bài toán khuyến nghị.

Thuật toán khuyến nghị FM có thể mở rộng ra với các thông tin bổ trợ của người dùng và sản phẩm. Ví dụ như về khuyến nghị cho người dùng một bộ phim, ta có thể xét mức độ ảnh hưởng của các thông tin bổ trợ như: giới tính, tuổi, nghề nghiệp, ... Những thành phần này sẽ được mã hóa thành các vector one-hot hoặc multi-hot vector. Nếu có thêm các dữ liệu dạng số khác, ta có thể thêm vào  $\mathbf{x}$  các thành phần tương ứng. Với mỗi thành phần được thêm vào  $\mathbf{x}$ , ta thêm một cột vector embedding vào  $\mathbf{V}$  như hình bên dưới đây. Khi đó, độ quan tâm của người dùng có thể được dựng lên như sau:

$$\hat{y}_{ij} = w_0 + \mathbf{x}\mathbf{w} + \sum_{i=1}^d \sum_{j=i+1}^d \mathbf{v}_i^T \mathbf{v}_j x_i x_j$$

Trong đó:

- $\mathbf{w}_0$  đóng vai trò như một hệ số bias trong mô hình hồi quy tuyến tính, nó có thể được xem như là một hệ số vô hướng cố định thêm vào kết quả dự đoán cuối cùng để điều chỉnh sự lệch trung bình.
- $\mathbf{x}\mathbf{w}$ : đây là tích vô hướng vector đặc trưng đầu vào (input feature vector) và một vector trọng số  $\mathbf{w}$  tương ứng với các đặc trưng của  $\mathbf{x}$ .
- $\sum_{i=1}^d \sum_{j=i+1}^d \mathbf{v}_i^T \mathbf{v}_j x_i x_j$ : Đây là thành phần tương tác bậc hai giữa các đặc trưng.
  - $x_i, x_j$  lần lượt là phần tử thứ  $i$ , thứ  $j$  trong feature vector  $\mathbf{x}$ .
  - $\mathbf{v}_i^T \mathbf{v}_j$  là tích vô hướng giữa các vector embeddings tương ứng với từng đặc trưng đầu vào  $x_i$  và  $x_j$ .
  - $\sum_{i=1}^d \sum_{j=i+1}^d \mathbf{v}_i^T \mathbf{v}_j x_i x_j$  là biểu diễn tổng tất cả các cặp tương tác giữa các đặc trưng có trong tập dữ liệu.



Đây chính là ý tưởng chính của FM. Đồng thời, nhờ vào việc  $\mathbf{x}$  thường là một vector rất thưa (rất ít thành phần khác 0), việc huấn luyện và dự đoán trở nên rất nhanh ngay cả khi số lượng người dùng và sản phẩm lớn.

### 3.2.4. Neutral Factorization Machine

Mặc dù có khả năng mô hình hóa các thông tin bổ trợ của người dùng và sản phẩm, performance của FM vẫn bị hạn chế bởi tính tuyến tính của nó cũng như việc chỉ mô hình các tương tác đặc trưng (ví dụ như bậc 2) theo cặp. Với dữ liệu thực tế có cấu trúc cơ bản phức tạp và phi tuyến, FM sẽ không đủ khả năng biểu diễn. Tuy FM bậc cao hơn đã được đề xuất, chúng vẫn thuộc họ mô hình tuyến tính và được cho là khó ước tính.

NFM ra đời nhằm cải tiến FM bằng cách mô hình hóa các tương tác đặc trưng cao và phi tuyến. Bằng cách sử dụng một phép toán mới trong mô hình neural network - Bilinear Interaction (Bi-interaction) pooling, NFM được xem như là sự kết hợp của FM với neural network framework. Thông qua việc xếp chồng các lớp phi tuyến trên Bi-interaction pooling layer, NFM đã làm sâu hơn mô hình FM tuyến tính nông, từ đó mô hình hóa các tương tác đặc trưng phi tuyến và bậc cao một cách hiệu quả, cải thiện performance của FM.

Với một feature vector thưa  $\mathbf{x} \in \mathbb{R}^n$  làm đầu vào, trong đó  $x_i = 0$  nghĩa là đặc trưng thứ  $i$  không tồn tại trong đối tượng, NFM dự đoán mục tiêu như sau:

$$\hat{y}(\mathbf{x}) = w_0 + \mathbf{x}\mathbf{w} + f(\mathbf{x})$$

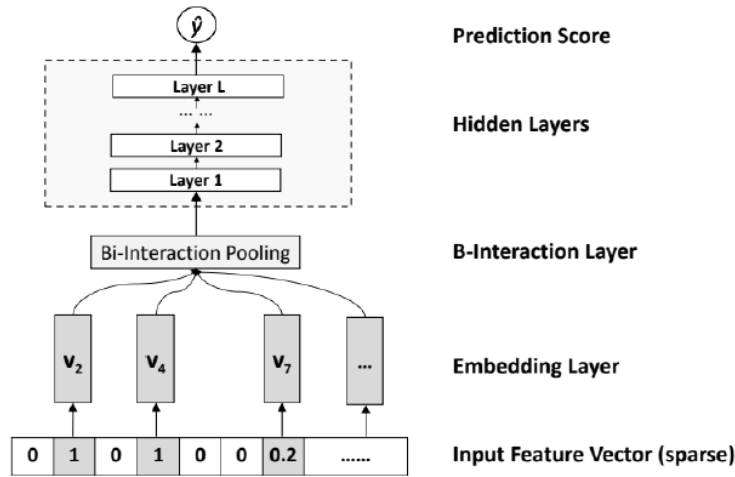
Trong đó, term đầu tiên và thứ 2 là phần linear regression giống với FM, thứ mô hình bias toàn cục và trọng số của các đặc trưng. Còn term thứ 3  $f(\mathbf{x})$  là thành phần cốt lõi trong NFM để mô hình tương tác đặc trưng.  $f(\mathbf{x})$  chính là một multi-layer feed-forward neural network như bên dưới:

Do Embedding layer tương tự như FM nên ta sẽ tiếp tục tìm hiểu về Bi-Interaction pooling, Hidden layer, Prediction layer.

**Bi-interaction pooling:** chuyển đổi tất các embedding vectors  $V_x$  thành 1 vector như sau:

$$f_{BI}(V_x) = \sum_{i=1}^n \sum_{j=i+1}^n x_i v_i \odot x_j v_j$$





Trong đó,  $\odot$  là ký hiệu của element-wise product của 2 vectors;  $x_i, x_j$  lần lượt là phần tử thứ  $i$  và  $j$  trong feature vector  $\mathbf{x}$ .  $\mathbf{v}_i, \mathbf{v}_j$  lần lượt là embedding của feature thứ  $i$  và  $j$ . **Hidden layer:** Trên Bi-interaction pooling là 1 chồng các lớp fully connected, thứ có khả năng mô hình các tương tác bậc cao giữa các đặc trưng. Mỗi hidden layer sẽ có một non-linear activation function như tanh, sigmoid, ReLU. **Prediction layer:** Cuối cùng, vector đầu ra của hidden layer cuối sẽ được chuyển đổi thành prediction score:

$$f(\mathbf{x}) = \mathbf{h}^T \mathbf{z}_L$$

Trong đó,  $\mathbf{h}$  là weight của prediction layer,  $\mathbf{z}_L$  là output vector của hidden layer cuối.

### 3.3. Phương pháp tiếp cận cổ điển

#### 3.3.1. Content-based filtering (Utility Matrix)

##### Giới thiệu sơ thuật toán

Ở đề tài này, ta sử dụng mô hình **Content-based Filtering** dựa trên các đặc điểm của khóa học để đề xuất những khóa học có nội dung tương tự cho người dùng, dựa trên sở thích và lịch sử học tập của họ. Ví dụ, nếu người dùng đã tham gia các khóa học thuộc một chủ đề nhất định, hệ thống sẽ đề xuất các khóa học có chủ đề tương tự.



Để hiểu rõ hơn về khái niệm của mô hình này, ta sẽ đồng thời tìm hiểu về mô hình **Collaborative Filtering** để đối chiếu và đưa ra các điểm khác biệt giữa chúng.

Trước hết, **Content-based Filtering** và **Collaborative Filtering** là hai phương pháp khuyến nghị phổ biến, và điểm khác nhau giữa chúng là cách sử dụng dữ liệu và cách xây dựng đề xuất. Dưới đây là các điểm khác biệt chính:

**a. Cơ sở dữ liệu sử dụng**

- **Content-based Filtering:** Dựa trên các đặc điểm và nội dung của mục tiêu cần khuyến nghị (items), chẳng hạn như mô tả khóa học, chủ đề, hay lĩnh vực học tập. Hệ thống đề xuất những khóa học có nội dung tương tự với các khóa học mà người dùng đã tham gia trước đó. Các thuộc tính của khóa học hoặc mục tiêu sẽ được mã hóa dưới dạng vector đặc trưng, và hệ thống sử dụng các vector này để tìm ra các khóa học tương tự.
- **Collaborative Filtering:** Dựa trên sở thích và hành vi của người dùng như xếp hạng, đánh giá, hoặc lượt xem. Thay vì dựa vào nội dung của các mục tiêu, Collaborative Filtering sử dụng thông tin về cách người dùng tương tác với các mục tiêu, và dựa vào đó để đưa ra các gợi ý dựa trên sự tương đồng giữa người dùng hoặc giữa các khóa học.

**b. Phương pháp tiếp cận**

- **Content-based Filtering:** Dựa vào hồ sơ sở thích của từng người dùng để cá nhân hóa đề xuất. Nếu một người dùng đã xem nhiều khóa học thuộc lĩnh vực A, hệ thống sẽ đề xuất các khóa học khác trong cùng lĩnh vực, dựa trên các đặc điểm giống nhau.
- **Collaborative Filtering:** Có hai hướng tiếp cận chính:
  - User-based Collaborative Filtering: Đề xuất dựa trên những người dùng có sở thích tương tự. Ví dụ, nếu người dùng A và B có cùng sở thích trong một số khóa học, hệ thống có thể gợi ý cho A các khóa học mà B đã tham gia, và ngược lại.



- Item-based Collaborative Filtering: Đề xuất dựa trên các khóa học tương tự mà nhiều người dùng cùng yêu thích. Ví dụ, nếu nhiều người dùng học cả khóa X và Y, hệ thống sẽ đề xuất khóa Y cho những người dùng chỉ mới học khóa X.

### c. Yêu cầu về dữ liệu

- **Content-based Filtering:** Cần có thông tin chi tiết và mô tả về từng mục tiêu (khóa học, phim, sản phẩm), chẳng hạn như từ khóa, chủ đề, hoặc bất kỳ đặc điểm nào liên quan đến nội dung của mục tiêu. Cách tiếp cận này ít phụ thuộc vào dữ liệu người dùng nhưng yêu cầu dữ liệu chất lượng về nội dung.
- **Collaborative Filtering:** Phụ thuộc vào dữ liệu về tương tác của người dùng, nên yêu cầu lượng người dùng lớn và tương tác phong phú. Phương pháp này có thể gặp khó khăn khi dữ liệu thưa (khi người dùng mới hoặc khóa học mới chưa có nhiều tương tác), dẫn đến vấn đề cold-start.

### d. Ưu và nhược điểm

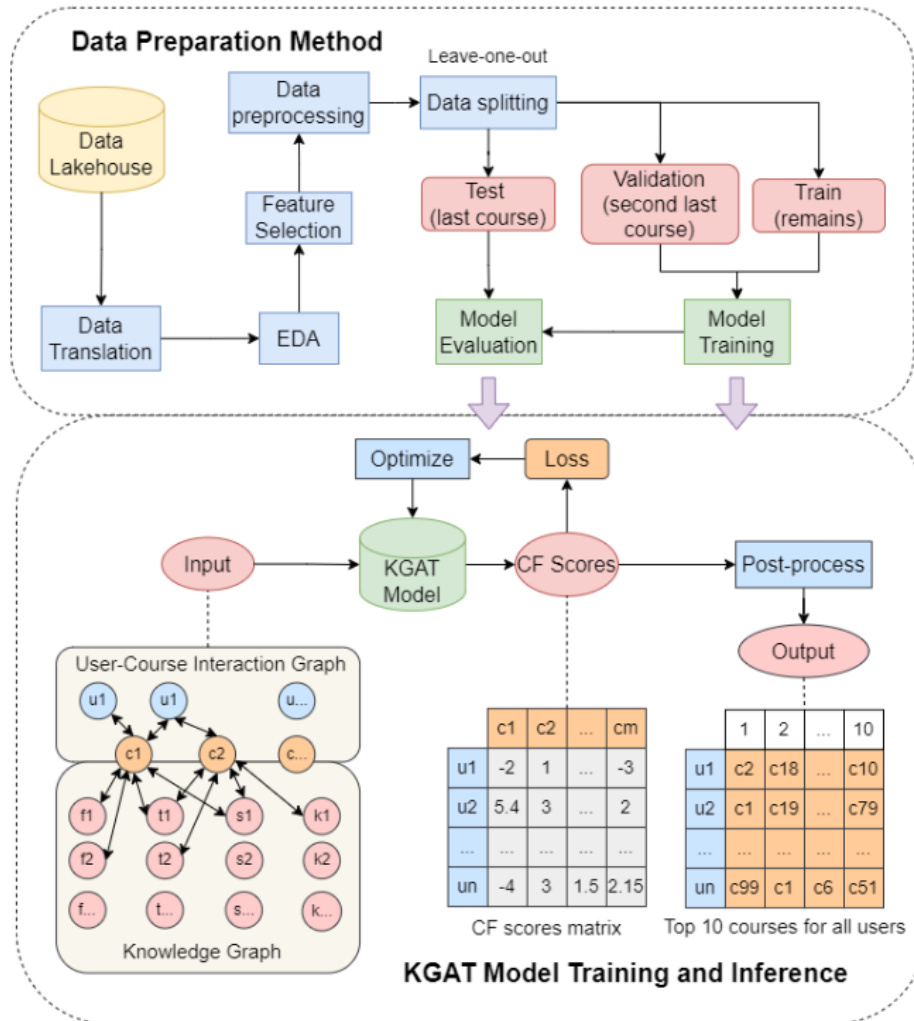
- **Content-based Filtering:**
  - Ưu điểm: Có thể gợi ý khóa học ngay cả khi người dùng không có nhiều tương tác; phù hợp cho trường hợp cold-start ở mức độ nhất định.
  - Nhược điểm: Dễ dẫn đến sự lặp lại và thiếu đa dạng vì chỉ đề xuất nội dung tương tự; không dễ phát hiện những sở thích tiềm ẩn ngoài những gì người dùng đã học.
- **Collaborative Filtering:**
  - Ưu điểm: Khả năng khám phá các khóa học ngoài sở thích hiện tại của người dùng; thường đưa ra các gợi ý đa dạng hơn vì dựa trên hành vi tập thể.
  - Nhược điểm: Không hoạt động hiệu quả trong tình huống cold-start; cần lượng dữ liệu lớn về tương tác giữa người dùng và mục tiêu để có hiệu quả cao.



## 4. Phương pháp đề xuất

### 4.1. Mô hình

Sau khi chạy thực nghiệm và đánh giá các phương pháp gợi ý trên dữ liệu MOOCCubeX [?], nhóm đã chọn ra được phương pháp tốt nhất là mô hình KGAT (chi tiết kết quả trong Mục ??). Vì vậy, trong phần này, nhóm sẽ trình bày cách chuyển đổi từ input sang output của mô hình KGAT, còn phương pháp chuẩn bị dữ liệu sẽ được trình bày ở phần ??.



Hình 6: Hình minh họa quy trình thực nghiệm bài toán với cách tiếp cận sử dụng mô hình KGAT.

Input của mô hình là collaborative knowledge graph chứa hai đồ thị thành phần như Hình ??: user-item bipartite graph và knowledge graph.

- **User-item bipartite graph:** Biểu diễn mối quan hệ giữa người dùng và các khóa học đã đăng ký.
- **Knowledge graph:** Biểu diễn mối quan hệ giữa khóa học và các thuộc tính của nó. Với node xuất phát là khóa học, đồ thị này bao gồm 4 loại quan hệ chính:



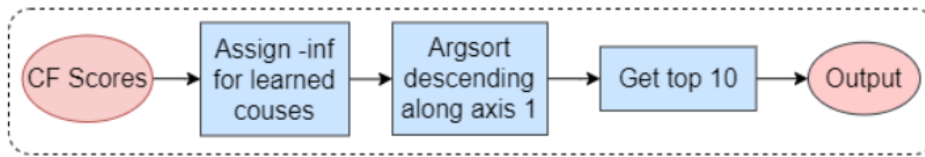
- Khóa học có khái niệm (concept) nào.
- Khóa học thuộc lĩnh vực (field) nào.
- Khóa học được dạy bởi giáo viên (teacher) nào.
- Khóa học được tổ chức bởi trường (school) nào.
- Khóa học có phần mô tả tương đồng với (course) nào.

Tính cả chiều ngược lại, tổng cộng có 8 loại quan hệ giữa khóa học và các thuộc tính.

Khi nhận được input, mô hình trả về một ma trận collaborative scores  $cf\_scores \in \mathbb{R}^{n_{users} \times n_{courses}}$ , với mỗi phần tử  $cf\_scores_{i,j} \in \mathbb{R}$  ( $i, j \in \mathbb{N}; 0 \leq i < n_{users}; 0 \leq j < n_{courses}$ ) cho biết mức độ phù hợp giữa user  $i$  và khóa học  $j$ . Giá trị càng lớn thì user  $i$  càng phù hợp với khóa học  $j$ .

Giai đoạn hậu xử lý (Hình ??) thực hiện các bước sau:

- Gán giá trị âm vô cùng cho  $cf\_scores[i, j]$  nếu user  $i$  đã đăng ký khóa học  $j$ .
- Thực hiện **argsort** giảm dần theo chiều axis=1 để loại bỏ các khóa học đã được đăng ký.
- Lấy top-10 khóa học có điểm cao nhất ứng với mỗi user để làm output.



Hình 7: Hình minh họa giai đoạn hậu xử lý để lọc ra top-k khóa học được đề xuất.

## 5. Thực nghiệm

### 5.1. Miêu tả bộ dữ liệu

### 5.2. Phương pháp tổ chức dữ liệu thực nghiệm



### 5.2.1. Dịch bảng

Trong quá trình chuyển ngữ từ Trung sang Việt, chúng em đã tận dụng thư viện "googletrans một công cụ Python không mất phí và không giới hạn số lần dịch. Thư viện này vận hành thông qua API Google Translate Ajax để thực hiện các tác vụ như nhận diện ngôn ngữ và dịch thuật.

Do khối lượng dữ liệu lớn, quá trình dịch gặp phải một số thách thức về thời gian và kết nối. Để khắc phục, chúng em đã triển khai các giải pháp sau:

- Lưu lại tiến trình dịch để tránh mất dữ liệu
- Thiết lập cơ chế tự động gửi lại yêu cầu khi mất kết nối
- Ứng dụng thư viện "asyncio" cho phép gửi đồng thời nhiều API, giúp tối ưu tốc độ xử lý

Đây là một phần code mẫu đã sử dụng phương pháp đã nêu trên:

```
async def translate(df, batch_start, batch_end):
    tasks = []
    for i in range(batch_start, batch_end):
        tasks.append(async_translate(df.loc[i, COL], i))

    df.loc[batch_start: batch_end - 1, COL] = await asyncio.gather(*tasks)

df = pd.read_json('teacher.json', lines=True)
batch_size = 1000
for i in range(0, len(df), batch_size):
    batch_end = min(len(df), i + batch_size)
    asyncio.run(translate(df, i, batch_end))

df[COL].to_csv(f"translated_{COL}.csv", index=False)
```



Ngoài ra, chúng em nhận thấy không cần thiết phải dịch toàn bộ các trường dữ liệu lớn để huấn luyện mô hình vì một số trường dữ liệu không hỗ trợ cho việc huấn luyện mô hình. Thay vào đó, chúng em chỉ tập trung dịch 1 số trường sau đây:

**-course.json:** dịch cột “name”, “field”, “prerequisites” và “about”

```
async def translate(df, batch_start, batch_end):
    tasks = []
    for i in range(batch_start, batch_end):
        tasks.append(async_translate(df.loc[i, COL], i))

    df.loc[batch_start: batch_end - 1, COL] = await asyncio.gather(*tasks)

df = pd.read_json('teacher.json', lines=True)
batch_size = 1000
for i in range(0, len(df), batch_size):
    batch_end = min(len(df), i + batch_size)
    asyncio.run(translate(df, i, batch_end))

df[COL].to_csv(f"translated_{COL}.csv", index=False)
```

**-user.json:** dịch cột “school”

```
user_df = pd.DataFrame(data_list)
user_df.head()
```

✓ 44.0s Python

	id	name	gender	school	year_of_birth	course_order	enroll_time
0	U_22	我	0.0	None	2015.0	[682129, 2294668]	[2019-10-12 10:28:02, 2020-11-21 14:03:28]
1	U_24	王坤国	1.0	Tsinghua University	6558.0	[597214, 605512, 597211, 597314, 597208, 62950...	[2019-05-20 16:06:48, 2019-05-24 19:34:43, 201...
2	U_25	王坤国	0.0	Tsinghua University	NaN	[1903985]	[2020-08-07 18:59:13]
3	U_53	于歆杰	1.0	Tsinghua University	1973.0	[696679, 1704639, 943255, 1729417, 682164, 177...	[2020-03-01 21:24:30, 2020-03-12 16:17:02, 202...
4	U_54	马昱睿	2.0	Tsinghua University	NaN	[682442, 682164, 1748240, 1778890, 1829031, 17...	[2019-10-09 02:17:49, 2019-11-08 00:49:03, 202...





-**teacher.json**: Tiến hành dịch tất cả (trừ “id” và “name”)

```
teacher_df.head()
```

	id	name	name_en	about	job_title	org_name
0	T_1	刘燕妮	Yanni Liu	Graduated from the Philosophy Department of Pe...	lecturer	Tsinghua University
1	T_2	陈怡	Yi Chen	Born in Chongqing in 1945, he graduated from H...	professor	Tsinghua University
2	T_3	程钢	Gang Cheng	Cheng Gang is the course leader of "Introducti...	Associate Professor	Tsinghua University
3	T_4	谢维和	xie wei he	Xie Weihe, PhD, professor, doctoral supervisor...	professor	Tsinghua University
4	T_5	史静寰	Jing-huan Shi	Shi Jinghuan, female, professor and doctoral s...	professor	Tsinghua University

-**concept.json**: Dịch tất cả các cột của bảng này vì toàn bộ đều ở dạng chuỗi

```
df = pd.read_json("../translated/concept_translated.json", lines=True)
df.head()
```

✓ 3.1s

	id	name	context
0	K_Nervous system_Histology and Embryology	Nervous system	[]
1	K_TSH cells_Histology and Embryology	TSH cells	['The pituitary gland consists of two parts: t...
2	K_Chromophilic cells_Histology and Embryology	Chromophilic cells	[]
3	K_Growth hormone cells_Histology and Embryology	Growth hormone cells	['Answer: B\n13. Adenohypophysis eosinophils c...
4	K_Limonite_Materials Science and Engineering	Limonite	['\nLimonite is a common iron ore, often forme...

-**course-field.json**: Tiến hành dịch cột course\_name và field mang các thông tin dưới dạng chuỗi của bảng.

```
df = pd.read_json("../original_translated/course-field-translated.json", lines=True)
df.head()
```

✓ 0.0s Python

	course_id	course_name	field
0	584313	Introduction to "Zi Zhi Tong Jian"	[Chinese language and literature, History]
1	681932	"Learning by doing" Java programming	[Computer Science and Technology]
2	674962	The spatial art of "Dream of Red Mansions"	[Chinese language and literature]
3	682709	Introduction to the Critique of Pure Reason	[philosophy]
4	682635	Introduction to "Tongwancheng"	[History]

## 5.2.2. Khám phá dữ liệu

a) Bảng **course.json** Ta xem qua bảng **course.json**:



```
course_df = pd.DataFrame(data_list)
course_df.head()
```

Python

	id	name_trans	field	prerequisites_trans	about_trans	resource
0	C_584313	introduction to "zi zhi tong jian"	[history, chinese language and literature]		through the teacher's guidance, students can g...	[{'titles': ['第一课 导论与三家分晋', '导论', '导论'], 'reso...
1	C_584329	calculus - limit theory and functions of one v...	[applied economics, math, physics, theoretical...		this course is a basic mathematics course in s...	[{'titles': ['序言', '序言', '序言'], 'resource_id':...
2	C_584381	photojournalism	[art, journalism and communication]		master basic photography skills, understand ho...	[{'titles': ['第一章 绪论', '第一讲 引言1', '引言1'], 'reso...
3	C_597208	data mining: theory and algorithms	[computer science and technology]		the most interesting theory + the most useful ...	[{'titles': ['走进数据科学: 博大精深, 美不胜收', '整装待发', 'Vide...
4	C_597225	university computer	[]		university computer courses will be guided by ...	[{'titles': ['第1周: 基于计算机的问题求解', '课程介绍', '开篇']...

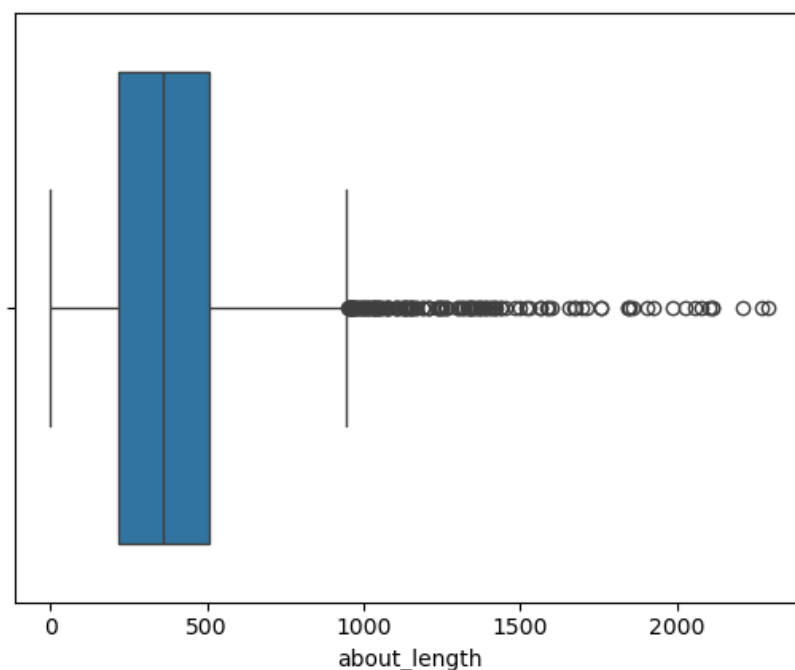
Ta xét độ dài của 3 cột “about”, “name\_trans” và “resource”:

	about_length	name_length	resource_length
count	3781.000000	3781.000000	3781.000000
mean	393.445120	36.942343	71.685533
std	267.904934	21.575065	74.802345
min	0.000000	2.000000	1.000000
25%	217.000000	22.000000	38.000000
50%	361.000000	32.000000	59.000000
75%	509.000000	46.000000	88.000000
max	2293.000000	193.000000	2728.000000



Ta có thể thấy được 1 số thông tin từ dữ liệu trên:

- Có những dòng dữ liệu không tồn tại cột “about”, tồn tại giá trị ngoại lệ ở cột “about” vì mean là 393 mà max lên đến 2293. Ta thể hiện trên boxplot độ dài của cột “about”:



- Có thể thấy thật sự nhiều giá trị ngoại lệ cần được xử lý.
- Có những dòng dữ liệu không có resource\_length, mean cũng rất ngắn (71) chứng tỏ ít thông tin về khoá học.

Ta phân tích sâu cột “resource”:

```
course_df['resource'][0][0]
```

```
{'titles': ['第一课 导论与三家分晋', '导论', '导论'],  
'resource_id': 'V_849',  
'chapter': '1.1.1'}
```



Mỗi resource trong bảng 2 là 1 tập hợp các video hay một tập các exercise. Mỗi resource sẽ có thêm 1 resource\_id là id của resource, chapter là chương chứa resource trong khóa học, titles gồm các tiêu đề như tiêu đề chương, video chương. Thông tin của resource có thể tìm thấy trong file course.json. Một resource có 2 loại: Video và Exercise. Nếu loại tài nguyên là video, nó được xác định bằng ID video bắt đầu bằng ký tự V\_. Nhiều video\_id khác nhau tương ứng với một ccid, và ccid xác định duy nhất một video. Các video\_id này tương ứng với việc hiển thị cùng một video ccid tại các thời gian bắt đầu khác nhau. Mỗi liên hệ giữa video\_id và ccid được lưu trong relations/video\_id-ccid.txt. Phụ đề video có thể được tìm thấy trong tệp entities/video.json thông qua ccid. Ta sẽ kiểm tra xem có bao nhiêu ID video không hợp lệ để phục vụ cho quá trình xử lý dữ liệu sau này:

```
videoID = ccid_df['video_id'].unique()

valid_videoID = set(videoID)

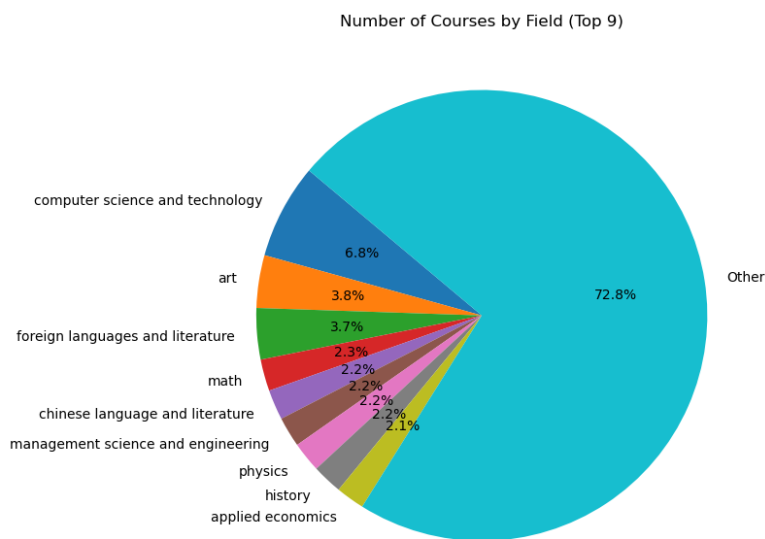
non_existent_ids = unique_video_ids - valid_videoID

# Hiển thị kết quả
print(f"Tổng số lượng các video ID không tồn tại: {len(non_existent_ids)}")
print(f"Các video ID không tồn tại: {non_existent_ids}")

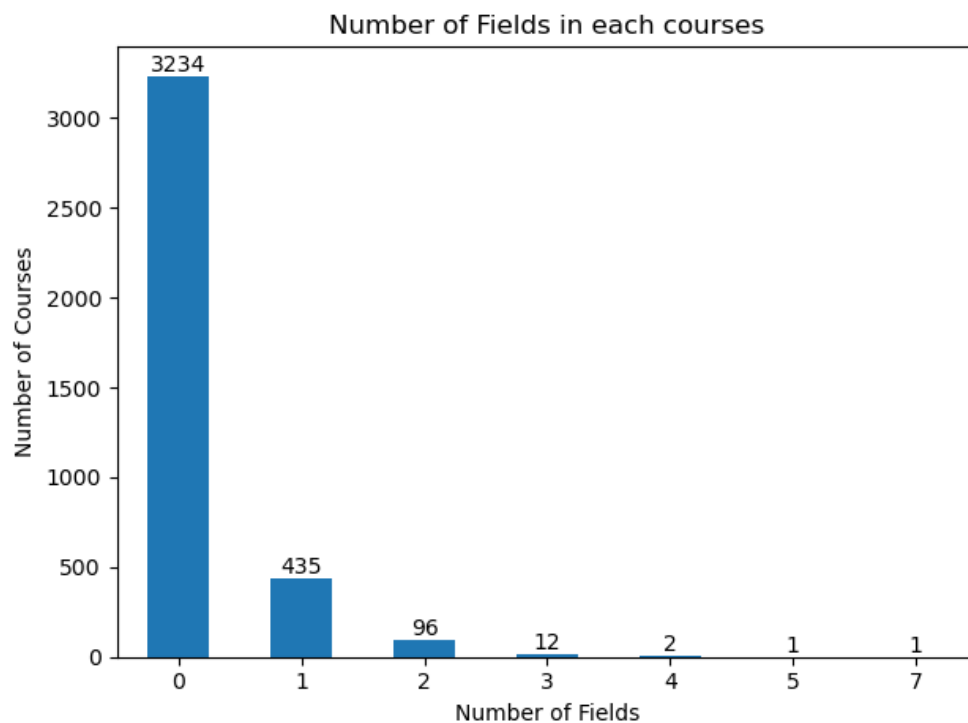
7]
Tổng số lượng các video ID không tồn tại: 2397
Các video ID không tồn tại: {'V_543429', 'V_543378', 'V_543519', 'V_1056006', 'V_3749
```

Có 2397 video ID không tồn tại, ta sẽ lọc đi hỗ trợ cho hiển thị thông tin trong tương lai. Ta bắt đầu tiến hành đếm số khoá học trong cột “name\_trans”, chia bởi lĩnh vực (cột “field”):

```
2) Number of courses by field:
field
computer science and technology    63
art                                35
foreign languages and literature    34
math                                21
history                             20
..
marine science                      1
ship and marine engineering          1
army command science                 1
metallurgical engineering            1
basic chinese medicine               1
Name: count, Length: 81, dtype: int64
```



Ta thấy có tổng 3781 khoá học và 81 lĩnh vực, với “computer science and technology” đứng đầu với 63 khoá học, chiếm 6.8% trên tổng khoá học. Ta cũng kiểm tra với mỗi khoá học được xếp bao nhiêu lĩnh vực (cột “field”):





Ta có thể thấy có rất nhiều khoá học không thuộc lĩnh vực nào, có rất nhiều khóa học không có field nào, có thể cột “field” sẽ không đóng góp nhiều trong xây dựng thuật toán hoặc cần xử lý.

#### b) Bảng user.json

Đầu tiên, ta đọc dữ liệu và quan sát dữ liệu thông qua dạng bảng (DataFrame):

	id	name	gender	school	year_of_birth	course_order	enroll_time
0	U_22	我	0.0	None	2015.0	[682129, 2294668]	[2019-10-12 10:28:02, 2020-11-21 14:03:28]
1	U_24	王帅国	1.0	Tsinghua University	6558.0	[597214, 605512, 597211, 597314, 597208, 62950...	[2019-05-20 16:06:48, 2019-05-24 19:34:43, 201...
2	U_25	王帅国	0.0	Tsinghua University	NaN	[1903985]	[2020-08-07 18:59:13]
3	U_53	于歆杰	1.0	Tsinghua University	1973.0	[696679, 1704639, 943255, 1729417, 682164, 177...	[2020-03-01 21:24:30, 2020-03-12 16:17:02, 202...
4	U_54	马昱春	2.0	Tsinghua University	NaN	[682442, 682164, 1748240, 1778890, 1829031, 17...	[2019-10-09 02:17:49, 2019-11-08 00:49:03, 202...



Ta tiến hành thống kê đặc điểm từng cột có trong bảng:

```
len(user_df)
```

✓ 0.0s Python

1128390

Hình 8: Số lượng users

```
user_df['gender'].describe()
```

✓ 1.7s Python

count	3.330240e+06
mean	9.455748e-01
std	8.321099e-01
min	0.000000e+00
25%	0.000000e+00
50%	1.000000e+00
75%	2.000000e+00
max	2.320000e+02

Name: gender, dtype: float64

Hình 9: Cột “gender”

```
user_df['gender'].value_counts()
```

✓ 0.0s Python

gender	
0.0	1221931
1.0	1067858
2.0	1040449
232.0	1
3.0	1

Name: count, dtype: int64

Hình 10: Phân bố các giá trị trong cột “gender”:



```
user_df['school'].describe()
✓ 0.3s Python
```

count	1128399
unique	25848
top	Tsinghua University
freq	18318

Name: school, dtype: object

Hình 11: Thông tin cột “school”

```
len(user_df["school"].unique())
✓ 0.0s Python
```

25849

Hình 12: Số lượng trường học trong bảng

```
user_df.info()
✓ 0.0s Python
```

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 3330294 entries, 0 to 3330293  
Data columns (total 7 columns):  
# Column Dtype  
--- ---  
0 id object  
1 name object  
2 gender float64  
3 school object  
4 year\_of\_birth float64  
5 course\_order object  
6 enroll\_time object  
dtypes: float64(2), object(5)  
memory usage: 177.9+ MB

Hình 13: Kiểm tra thông tin tổng quan sau cùng





```
user_df.info()
✓ 0.0s Python

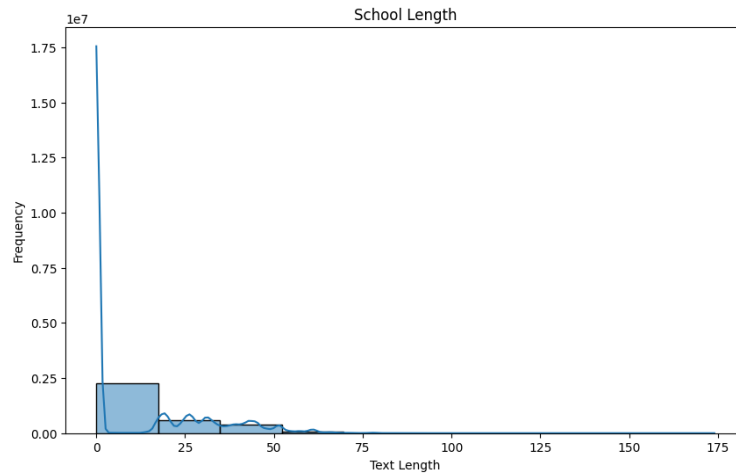
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3330294 entries, 0 to 3330293
Data columns (total 7 columns):
#   Column      Dtype
---  -
0   id           object
1   name         object
2   gender       float64
3   school       object
4   year_of_birth float64
5   course_order object
6   enroll_time  object
dtypes: float64(2), object(5)
memory usage: 177.9+ MB
```

Hình 14: Số lượng sample (users) có trong bảng và số lượng users thuộc về mỗi trường học

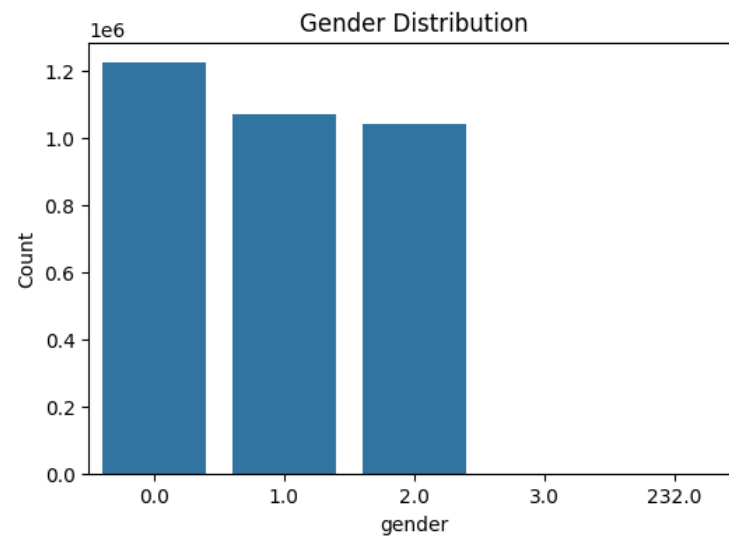
```
user_df['school_length'] = user_df['school'].apply(lambda x: len(x) if x is not None else 0)
user_df['school_length'].describe()
✓ 1.3s Python

count    3.330294e+06
mean     1.137576e+01
std      1.756154e+01
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      2.400000e+01
max      1.740000e+02
Name: school_length, dtype: float64
```

Hình 15: Tạo một cột “school\_length” để phân tích độ dài mỗi sample của cột



Hình 16: Trực quan hóa độ dài của sample cột “school”



Hình 17: Trực quan hóa phân bố các giá trị của cột “gender”

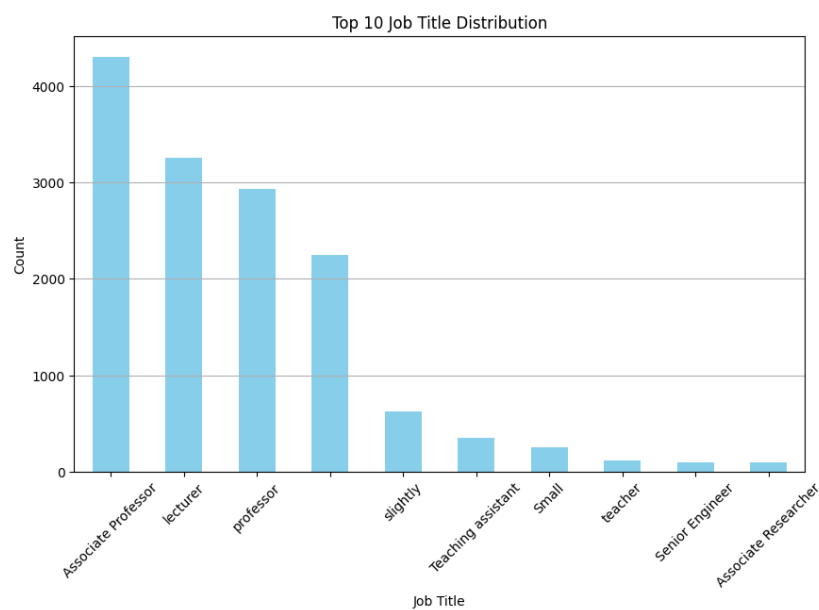


### c) Bảng teacher.json

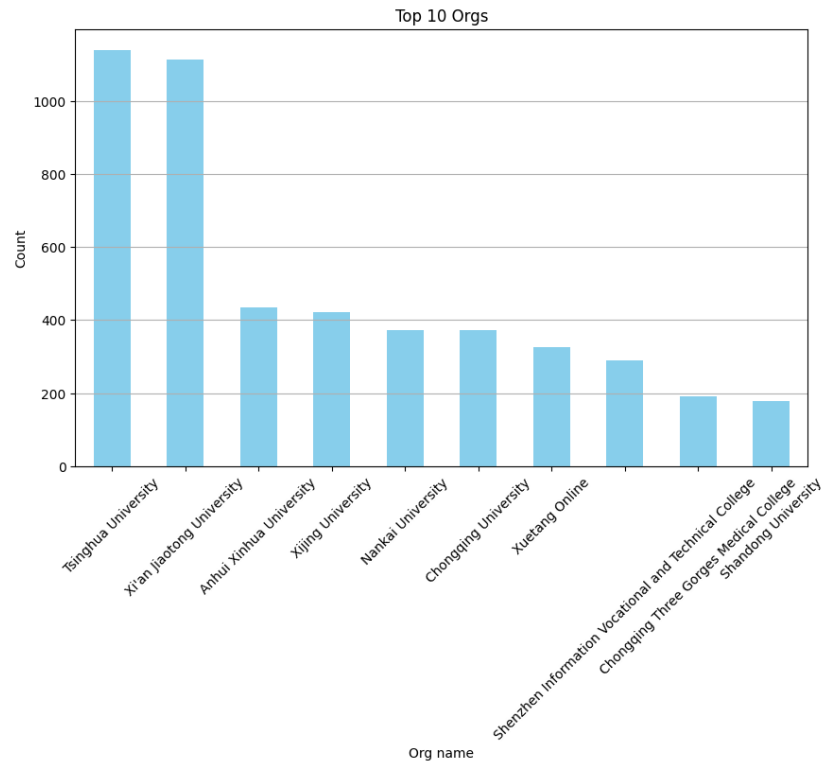
Sau đây là các thống số cơ bản của bảng

	id	name	name_en	about	job_title	org_name
count	17018	17018	17018	13893	14768	17018
unique	17018	13967	11061	12536	1323	998
top	T_1	顾礼平		slightly	Associate Professor	Tsinghua University
freq	1	20	4142	626	4305	1140
	0					
id	object					
name	object					
name_en	object					
about	object					
job_title	object					
org_name	object					
dtype:	object					

Tham khảo phân phối của top 10 tên việc xuất hiện nhiều nhất trong bảng



Tham khảo phân phối của top 10 tổ chức xuất hiện nhiều nhất trong bảng

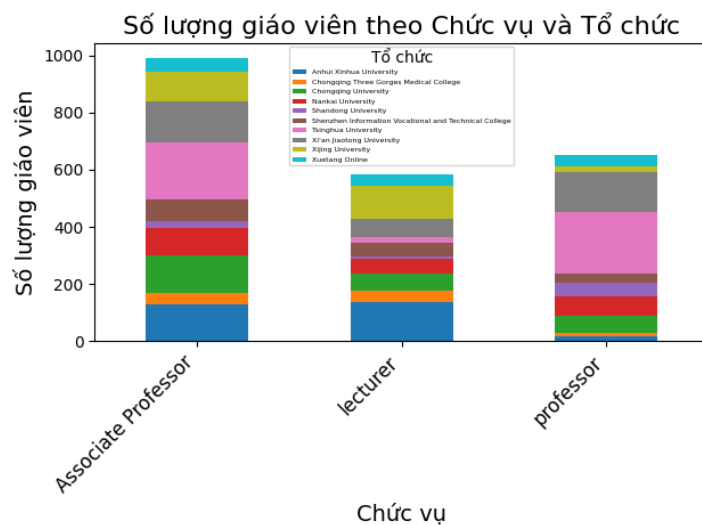


Ta thực hiện phân tích mối quan hệ giữa ba chức vụ (job titles) có số lượng giáo viên nhiều nhất và mười tổ chức (organizations) có số lượng giáo viên cao nhất

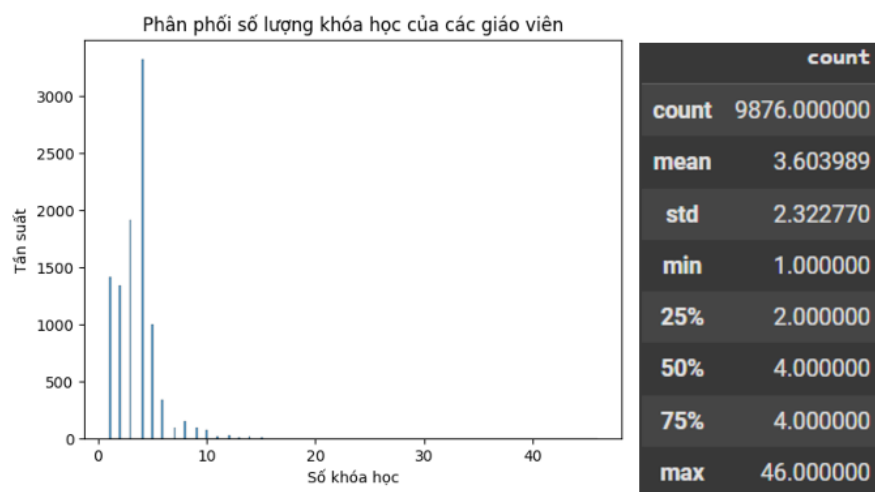
```
print("Bảng tần suất giữa job_title và org_name:")
contingency_table
```

Bảng tần suất giữa job\_title và org\_name:

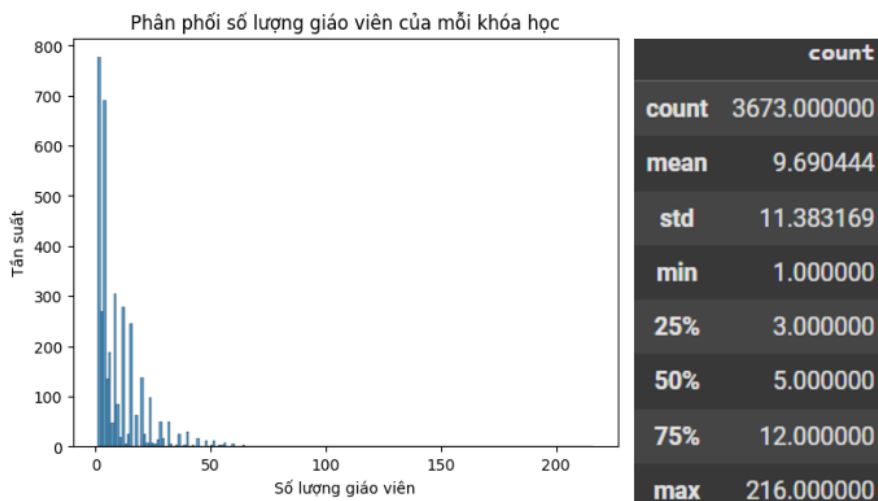
org_name	Anhui Xinhua University	Chongqing Three Gorges Medical College	Chongqing University	Nankai University	Shandong University	Shenzhen Information Vocational and Technical College	Tsinghua University	Xi'an Jiaotong University	Xijing University	Xuetang Online
job_title										
Associate Professor	130	39	130	97	23	78	199	144	105	47
lecturer	136	42	58	52	10	48	19	62	117	41
professor	16	14	57	68	49	33	217	138	19	40



Sau khi lọc bỏ các liên kết có khóa học hoặc teacher không tồn tại dựa vào file course-teacher.txt, số hàng còn lại là 35593. Các thông tin được trực quan hóa như sau



Hình 18: Histogram thể hiện số lượng khóa học của mỗi teacher và bảng thống kê mô tả tương ứng



Hình 19: Histogram thể hiện số lượng teacher của mỗi khóa học và bảng thống kê mô tả tương ứng

d) **Bảng school.json** Ta đếm dữ liệu ở từng cột, đếm các giá trị đặc biệt, giá trị xuất hiện nhiều nhất với tần số của nó:

```
df.describe(include='all')
```

	id	name	name_en	sign	about	motto
count	428	428	428	428	428	428
unique	428	421	423	420	420	138
top	S_1	长安大学	Dalian University of Technology	hzic	Hebei Normal University Of Science & Technolog...	
freq	1	2	2	2	2	282

Kiểm tra kiểu dữ liệu của từng cột:



```

▶ ▾ pd.DataFrame(df.info())
[99]
... <class 'pandas.core.frame.DataFrame'>
RangeIndex: 428 entries, 0 to 427
Data columns (total 6 columns):
#   Column      Non-Null Count  Dtype
---  -
0    id          428 non-null    object
1    name        428 non-null    object
2    name_en     428 non-null    object
3    sign        428 non-null    object
4    about       428 non-null    object
5    motto       428 non-null    object
dtypes: object(6)
memory usage: 20.2+ KB

```

Ta tạo 2 cột mới là “about\_length” và “motto\_length” để lần lượt thể hiện độ dài của giá trị dữ liệu ở 2 cột “about” và “motto”:

```

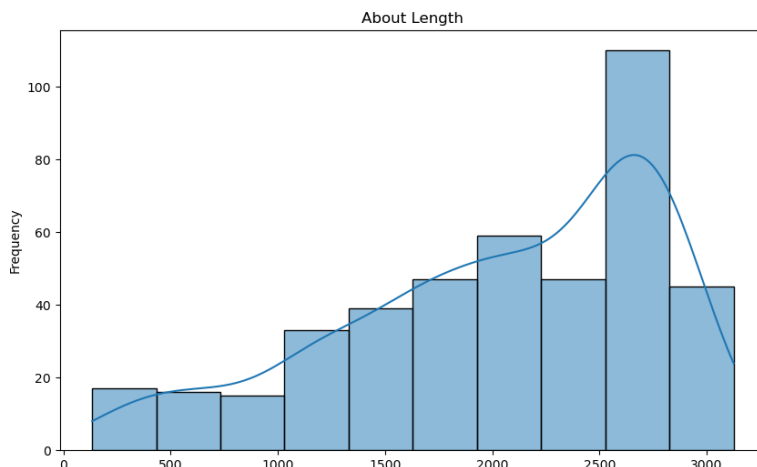
▾
lengths_df = pd.DataFrame({
    'about_length': df['about'].apply(len),
    'motto_length': df['motto'].apply(len)
})
# Display summary
lengths_df[['about_length', 'motto_length']].describe()
[99]

```

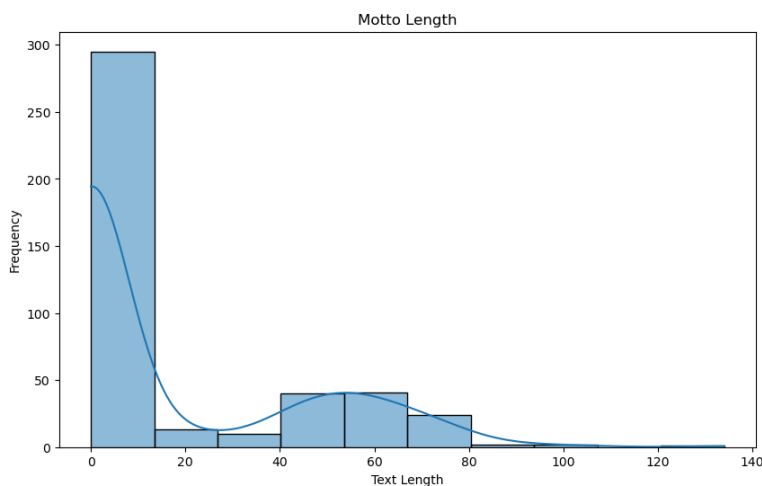
	about_length	motto_length
count	428.000000	428.000000
mean	2023.530374	16.955607
std	737.761007	26.937787
min	134.000000	0.000000
25%	1552.750000	0.000000
50%	2157.000000	0.000000
75%	2647.750000	42.000000
max	3126.000000	134.000000



Có 2 cột ta cần là “about\_length” và “motto\_length” để ta tìm phân bố độ dài của giá trị lên đồ thị:



Dựa vào biểu đồ ta có thể nhận xét rằng mô tả của các trường đều rất chi tiết, số lượng trường với số lượng từ phần mô tả  $> 2000$  chiếm phần lớn. Tuy nhiên thông tin này có vẻ không hữu ích với hệ thống khuyến nghị.



Hầu hết các trường đại học đều có một khẩu hiệu ngắn gọn dưới 20 từ vì chủ yếu khẩu hiệu sẽ đơn giản nhất có thể để truyền đạt tầm nhìn và mục tiêu của trường một cách trực tiếp ngắn gọn, đọng lại trong trí nhớ người xem. Một phần nhỏ hơn các trường có khẩu hiệu tương đối dài với 40 đến 88 chữ.





### e) Bảng course-field.json

```
# 1. Số lượng khóa học
num_courses = df['course_id'].nunique()
print(f"Số lượng khóa học: {num_courses}")

# 2. Số lượng các lĩnh vực khác nhau
unique_fields = set(field for fields_list in df['field'] for field in fields_list)
num_unique_fields = len(unique_fields)
print(f"Số lượng các lĩnh vực khác nhau: {num_unique_fields}")
```

✓ 0.0s Python

Số lượng khóa học: 632  
Số lượng các lĩnh vực khác nhau: 82

Hình 20: Tổng số lượng khóa học và tổng số lượng các lĩnh vực khác nhau

```
# 3. Phân bố số lượng khóa học theo từng lĩnh vực
field_distribution = df.explode('field')['field'].value_counts()
print("\nPhân bố số lượng khóa học theo từng lĩnh vực:")
print(field_distribution)
```

✓ 0.0s Python

Phân bố số lượng khóa học theo từng lĩnh vực:

field	
Computer Science and Technology	75
foreign languages and literature	43
Art	38
Chinese language and literature	26
Management Science and Engineering	25
..	..
Battle Science	1
Military Logistics and Military Equipment Science	1
Weapons Science and Technology	1
Army Command Science	1
Mining Engineering	1

Name: count, Length: 82, dtype: int64

Hình 21: Phân bố số lượng khóa học theo từng lĩnh vực



```
# 4. Phân bố độ dài tên khóa học (số ký tự)
course_name_length = df['course_name'].apply(len)
print("\nThống kê độ dài tên khóa học:")
print(course_name_length.describe())
```

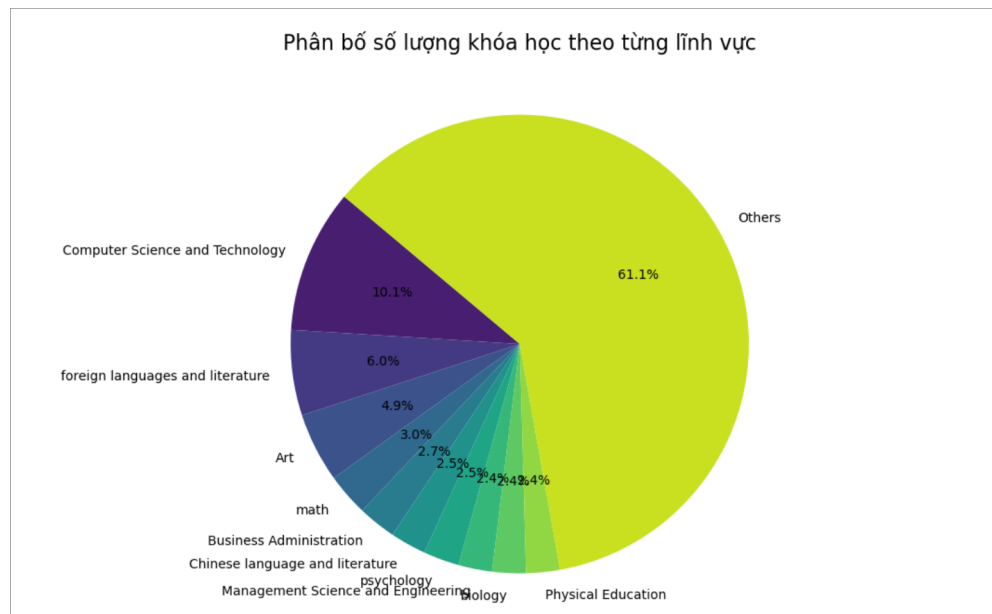
✓ 0.0s Python

Thống kê độ dài tên khóa học:

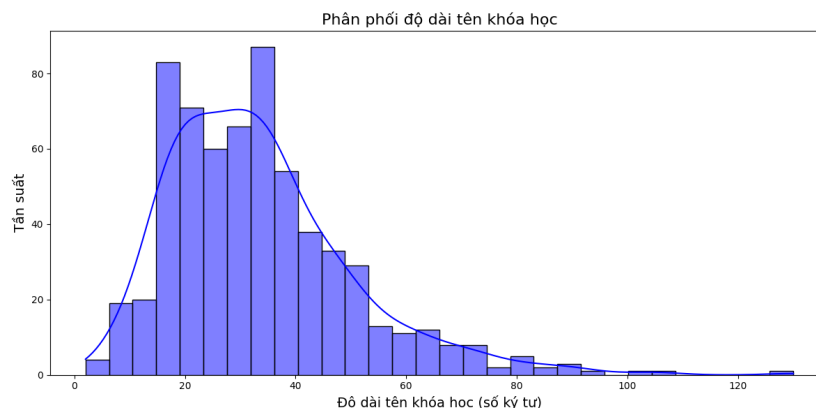
Statistic	Value
count	632.000000
mean	33.507911
std	16.882082
min	2.000000
25%	21.000000
50%	31.000000
75%	41.000000
max	130.000000

Name: course\_name, dtype: float64

Hình 22: Phân bố độ dài tên khóa học



Hình 23: Biểu đồ thanh thể hiện sự phân bố số lượng khóa học theo từng lĩnh vực



Hình 24: Biểu đồ phân phối cho độ dài tên khóa học

### 5.2.3. Làm sạch dữ liệu

#### a) Bảng course.json

Ta kiểm tra dữ liệu thiếu, dữ liệu không nhất quán, dữ liệu trùng lặp và dữ liệu trống:

Đầu tiên ta thấy được có 647 giá trị ở cột “name\_trans” bị trùng lặp cho dù id không bị trùng, chứng tỏ có sự lỗi nhất định trong bộ dữ liệu, cũng như này đã thống kê ta thấy được có rất nhiều giá trị trống ở cột “field\_trans”.

Ta kiểm tra kĩ hơn về các dòng có giá trị trong cột “name” bị trùng lặp:

	NaN values	NA values	Duplicated rows	Empty values
id	0	0	0	0
name_trans	0	0	224	0
field_trans	0	0	618	603
prerequisites_trans	0	0	459	413
about_trans	0	0	52	13
resource	0	0	0	0
course_name	603	603	607	0
field	603	603	618	0



	id	name_trans	field_trans	prerequisites_trans	about_trans	resource	course_name	field
1490	769273	introduction to the basic principles of marxism	(political science.)		"basic principles of marxism" educates college...	{{'titles': ['导论', '1.1 马克思, 何许人也?', 'Video'], ...	马克思主义基本原理概论 (2019春)	[政治学]
1598	837985	introduction to the basic principles of marxism	0	"ideological and moral cultivation and legal b...	why is marx right? what are "universal values"...	{{'titles': ['专题一: 为什么是马克思?', '1.1 为什么是马克思主义?', ...	NaN	NaN
2826	1891061	introduction to the basic principles of marxism	0		the basic principles of marxism are the basic ...	{{'titles': ['绪论', '1.青年马克思', '青年马克思'], 'resou...	NaN	NaN
3757	2342515	introduction to the basic principles of marxism	0	ideological and moral cultivation and legal fo...	the course "introduction to the basic principl...	{{'titles': ['绪论', None, '序言'], 'resource_id': ...	NaN	NaN

Ta thấy được đa số dữ liệu trong này cột “field” đa số bị trống và trùng lặp, cũng như các cột khác không có ý nghĩa hoặc trùng với các cột khác, thực hiện chi square test, ta có được kết quả với P-value rất thấp, chứng tỏ các giá trị phụ thuộc với nhau chứ không hề có giá trị mới. Chứng tỏ ta có thể xóa được các dòng dữ liệu này, cũng như các khoá học không tồn tại trong “course-field.json”.

### b) Bảng user.json

Ta thấy cột “year\_of\_birth” bị thiếu dữ liệu hơn 97% trong khi các cột còn lại tỉ lệ % thiếu là rất thấp. Ta tiến hành loại bỏ cột này, sau đó ta sẽ tiến hành xử lý dữ liệu nhiễu trên cột gender với 2 giá trị nhiễu là 232 và 3

### c) Bảng concept.json

Ta thấy cột “id” bị thiếu 207 giá trị, ta tiến hành bỏ các hàng này.

```
print("Số lượng giá trị thiếu trong từng cột:")
print(df.isnull().sum())
df = df.dropna()
print("Số lượng thiếu sau khi xử lý:", df.isna().sum().sum())
```

✓ 0.3s

```
Số lượng giá trị thiếu trong từng cột:
id      207
name      0
context   0
dtype: int64
Số lượng thiếu sau khi xử lý: 0
```



Ta kiểm tra kĩ hơn về các bản ghi có giá trị trùng lặp và xử lý chúng:

### 3.2 Xử lý dữ liệu trùng lặp

```
print("Số lượng bản ghi trùng lặp:", df.duplicated().sum())
duplicates = df[df.duplicated(keep=False)]
display(duplicates)
df = df.drop_duplicates()
print("Số lượng trùng lặp sau khi xử lý:", df.duplicated().sum())
```

✓ 4.7s

Số lượng bản ghi trùng lặp: 11543

	id	name	context
16	K_Dorsal digital vein_Human anatomy	Dorsal digital vein	[]
240	K_Hilar lymph node tuberculosis_Tuberculosis	Hilar lymph node tuberculosis	[]
246	K_Dynamic stability_Ship engineering	Dynamic stability	[]
287	K_Vesicouterine fistula_Urology	Vesicouterine fistula	[]
289	K_Vesicouterine fistula_Urology	Vesicouterine fistula	[]
...	...	...	...
637335	K_Genetically modified corn_Food Science and E...	Genetically modified corn	[]
637374	K_Meringue_Food Science and Engineering	Meringue	[]
637376	K_Hip fracture_Food Science and Engineering	Hip fracture	[]
637527	K_Esters_Food Science and Engineering	Esters	[]
637570	K_Genetically modified corn_Food Science and E...	Genetically modified corn	[]

21106 rows × 3 columns

Số lượng trùng lặp sau khi xử lý: 0

#### d) Bảng course-field.json

Sử dụng `isnull().sum()` để tính số lượng giá trị thiếu trong từng cột. Sau đó loại bỏ hàng chứa giá trị thiếu bằng cách sử dụng `dropna()`



```
Số lượng giá trị thiếu trong từng cột:  
course_id      0  
course_name    0  
field          0  
dtype: int64  
  
Tỷ lệ dữ liệu thiếu trong từng cột (%):  
course_id      0.0  
course_name    0.0  
field          0.0  
dtype: float64  
  
Số lượng giá trị thiếu sau khi xử lý:  
course_id      0  
course_name    0  
field          0  
dtype: int64
```

Dữ liệu văn bản thường chứa nhiều thông tin nhiễu chẳng hạn như các ký tự không mong muốn: Các ký tự đặc biệt, dấu câu, hoặc ký tự không phải chữ cái có thể làm giảm chất lượng phân tích. Ở đây chúng ta sẽ tiến hành loại bỏ các ký tự không cần thiết, các khoảng trắng dư thừa và thường hóa các ký tự viết hoa

Để kiểm tra dữ liệu trùng lặp, chúng ta sử dụng phương thức `uplicated()` trong `pandas`. Đầu tiên xác định các bản ghi trùng lặp, sau đó đếm số lượng và hiển thị các bản ghi trùng lặp đó. Sau đó tiến hành xóa bản ghi trùng lặp bằng cách sử dụng `drop_duplicates()`



```
# Chuyển đổi cột 'field' thành chuỗi
df['field'] = df['field'].apply(lambda x: ', '.join(x))

# Kiểm tra dữ liệu trùng lặp
duplicate_rows = df.duplicated()

# Đếm số lượng bản ghi trùng lặp
num_duplicates = duplicate_rows.sum()
print(f"Số lượng bản ghi trùng lặp: {num_duplicates}")

# Hiển thị các bản ghi trùng lặp
if num_duplicates > 0:
    print("Các bản ghi trùng lặp:")
    print(df[duplicate_rows])

# Xóa bản ghi trùng lặp (nếu cần)
df_cleaned = df.drop_duplicates()

# Kiểm tra lại số lượng bản ghi sau khi xóa trùng lặp
print(f"Số lượng bản ghi sau khi xóa trùng lặp: {len(df_cleaned)}")
```

✓ 0.0s Python

Số lượng bản ghi trùng lặp: 0  
Số lượng bản ghi sau khi xóa trùng lặp: 632

### e) Bảng school.json

Ta xoá cột “name” đi vì trùng với ý nghĩa với cột “name\_en” (tên nhưng trong Tiếng Anh)

Ta thống nhất cột “sign” (kí hiệu đại diện cho trường) đều là tất cả in hoa:



```
df['sign'] = df['sign'].str.upper() ## convert to uppercase for consistent
df['sign'].value_counts()

[113]
...
sign
XJTU      2
HZIC      2
TJU       2
QDU       2
ECUST      2
..
PASTEURX   1
DELFTX    1
RICE       1
BURGUNDYX  1
HLJUCM     1
Name: count, Length: 418, dtype: int64
```

Vì ở đây tên trường (“name\_en”) cũng như kí hiệu (“sign”) là chìa khoá chính, hay nói cách khác là giá trị duy nhất nên không thể có dòng trùng với nhau, ta tiến hành xoá các dòng trùng giá trị:

Xử lí dữ liệu trùng lặp

```
df.drop_duplicates(subset=['name_en'], keep='first', inplace=True)
df.drop_duplicates(subset=['sign'], keep='first', inplace=True)

4]

name_en_counts = df['name_en'].value_counts()
name_en_counts[name_en_counts > 1]

9]

Series([], Name: count, dtype: int64)

sign_counts = df['sign'].value_counts()
sign_counts[sign_counts > 1]

6]

Series([], Name: count, dtype: int64)
```

### g) Bảng teacher.json

Ở đây có cột name\_en bị thiếu nên điền vào cột đó bằng cách lấy phiên âm của cột name là được. Để làm việc này có thể sử dụng thư viện pypinyin để lấy phát âm dùng cho tên tiếng anh.





```
from pypinyin import pinyin, Style
from functools import reduce

def get_reading(name):
    return ' '.join(r[0] for r in pinyin(name, style=Style.NORMAL))

print(f'Example name translation: "陈怡" to "{get_reading("陈怡")}"')

print("Before filling missing data:")
missing_data = df.isna().any(axis=1)
display(df[missing_data].head())

print("After filling missing data:")
df['name_en'] = df['name-en'].fillna(df['name'].apply(get_reading))
df[missing_data].head()

✓ 1.3s
Example name translation: "陈怡" to "chen yi"
Before filling missing data:
```

	id	name	name_en	about	job_title	org_name
3	T_4	谢维和	None	谢维和, 博士、教授、博士生导师、教育研究院院长。研究方向: 教育学原理、教育社会学、高等教育和...	教授	清华大学
5	T_6	王孙禹	None	王孙禹, 汉族, 教授、博士生导师, 出生于1947年10月, 曾任清华大学人文社会科学学院党委书记...	教授	清华大学
6	T_7	袁本涛	None	袁本涛, 博士、教授、博士生导师, 现任教育研究院副院长, 主要研究领域为高等教育政策, 高等教育管...	教授	清华大学
7	T_8	林健	None	林健, 福建福州人, 英国Lancaster大学管理学博士、博士后, 清华大学公共管理学教授、博士...	教授	清华大学
8	T_9	程建钢	None	程建钢, 博士、教授、博士生导师, 教育技术学科负责人暨学术带头人, 中国教育技术协会学术委员...	教授	清华大学

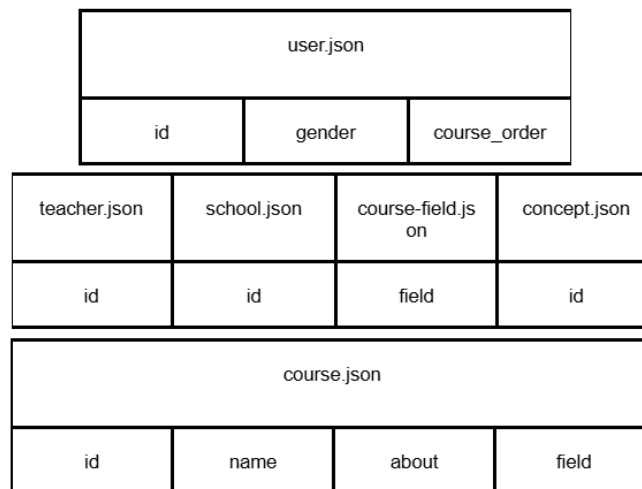
```
After filling missing data:
```

	id	name	name_en	about	job_title	org_name
3	T_4	谢维和	xie wei he	谢维和, 博士、教授、博士生导师、教育研究院院长。研究方向: 教育学原理、教育社会学、高等教育和...	教授	清华大学
5	T_6	王孙禹	wang sun yu	王孙禹, 汉族, 教授、博士生导师, 出生于1947年10月, 曾任清华大学人文社会科学学院党委书记...	教授	清华大学
6	T_7	袁本涛	yuan ben tao	袁本涛, 博士、教授、博士生导师, 现任教育研究院副院长, 主要研究领域为高等教育政策, 高等教育管...	教授	清华大学
7	T_8	林健	lin jian	林健, 福建福州人, 英国Lancaster大学管理学博士、博士后, 清华大学公共管理学教授、博士...	教授	清华大学
8	T_9	程建钢	cheng jian gang	程建钢, 博士、教授、博士生导师, 教育技术学科负责人暨学术带头人, 中国教育技术协会学术委员...	教授	清华大学

### 5.2.4. Chuyển đổi dữ liệu

**Feature Engineering:** Nhóm sẽ chọn các bảng và thuộc tính có thể sử dụng để tạo ra feature các mô hình khuyến nghị dựa trên bộ dữ liệu đã xử lý và làm sạch trước đó:

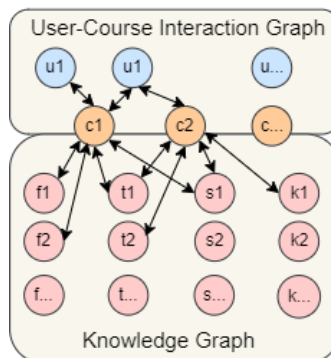
Các bảng được chọn và thuộc tính sử dụng:





- Với ‘user.json’: ‘course\_order’ gồm các khóa học mà user đã đăng ký với khóa học sau cùng là khóa học gần đây nhất, dùng để tạo liên kết giữa ‘user.json’ và ‘course.json’.
- Với ‘course.json’: Đây là table quan trọng chứa thông tin về các khóa học như ‘name’, ‘about’ và ‘field’.
- Với ‘teacher.json’, ‘school.json’: dùng để tạo relation với ‘course.json’ chứa thông tin về trường tổ chức khóa học và giáo viên giảng dạy.
- Với ‘course-field.json’: chứa các field của mỗi khóa học, dùng để kiểm tra với trường ‘field’ trong ‘course.json’.
- Với ‘concept.json’: id theo quy ước ‘K\_concept namefield’, tạo thêm feature concept-name\_field với mỗi khoá học.

Tạo knowledge graph:



Tạo interaction giữa người dùng với khóa học: sử dụng 5-core filtering, lọc người dùng với ít hơn 5 khóa học và những khóa học có số lượng đăng ký dưới 5. Kết quả: Vì data đã được xử lý trước đó nên ta thấy không có thay đổi đáng kể

Trước khi filter	Sau khi filter
1.183.774 interactions	1.182.745 interactions



Tạo relation giữa các entities: course-relation-attribute. Sau đó ta tiến hành lọc theo tiêu chí, số lần course xuất hiện tối thiểu là 5 và số lần xuất hiện tối thiểu của một relation là 25.

Kết quả:

Trước khi filter	Sau khi filter
376.093 interactions	71.787 interactions

### 5.2.5. Chia tập dữ liệu

- Dữ liệu cuối cùng được chia theo chiến lược leave-one-out: Với mỗi user, nhóm giữ khoá học cuối cùng làm test, các khoá học còn lại làm train.
- Dữ liệu cuối cùng để thực nghiệm có 4 loại dữ liệu chính: dữ liệu thô (chưa xử lý), 10% dữ liệu thô, dữ liệu đã xử lý (chiến lược leave-one-out) và 10% dữ liệu đã xử lý.

## 5.3. Độ đo đánh giá

## 5.4. Kịch bản thực nghiệm

## 5.5. Đánh giá kết quả thực nghiệm

# 6. Kết luận và hướng phát triển

## 6.1. Đánh giá các phương pháp

Mỗi phương pháp đều có ưu và khuyết điểm:

### Content-based filtering:

- Ưu điểm:
  - Không cần huấn luyện, suy luận nhanh.



- Không cần dữ liệu từ người dùng khác. Khi xuất hiện một khóa học mới chưa được đăng ký, ta hoàn toàn có thể dựa vào nội dung, thông tin của nó để gợi ý.
- Khả năng cá nhân hóa cao: Do phân tích sở thích cá nhân của người dùng, bộ lọc dựa trên nội dung có thể đưa ra những đề xuất phù hợp và chính xác hơn.
- Nhược điểm:
  - Performance thấp nếu hand-made feature không hiệu quả
  - Không cần dữ liệu từ người dùng khác cũng chính là khuyết điểm của phương pháp này, vì nó sẽ chỉ gợi ý những khóa học có nội dung tương tự với các khóa user đã đăng ký, thiếu đi tính đa dạng.

#### **Matrix factorization:**

- Ưu điểm:
  - Đầu tiên, nó có thể xử lý dữ liệu thưa thớt và không đầy đủ, điều này thường xảy ra đối với xếp hạng mục cũng như việc đăng ký khóa học của người dùng. MF có thể điền vào các giá trị còn thiếu và dự đoán xếp hạng cho các mục hoặc người dùng chưa nhìn thấy.
  - Thứ hai, nó có thể làm giảm chiều và độ phức tạp của dữ liệu, điều này có thể cải thiện hiệu quả và khả năng mở rộng của hệ thống gợi ý. MF có thể nén một ma trận lớn thành các ma trận nhỏ hơn để nắm bắt thông tin cần thiết và giảm nhiễu.
  - Thứ ba, nó có thể khám phá các đặc trưng và mẫu tiềm ẩn không rõ ràng hoặc rõ ràng trong dữ liệu. MF có thể tiết lộ những điểm tương đồng và sở thích tiềm ẩn giữa người dùng và mặt hàng, điều này có thể nâng cao chất lượng và tính đa dạng của các đề xuất
- Nhược điểm:
  - Đầu tiên, nó có thể bị overfitting và underfitting, điều này có thể ảnh hưởng đến tính chính xác và tính khái quát của các đề xuất. Overfitting



xảy ra khi các vectơ đặc trưng fit quá tốt với dữ liệu và thu được nhiều hoặc các ngoại lệ, trong khi underfitting xảy ra khi các vectơ đặc trưng fit quá tệ với dữ liệu và bỏ lỡ thông tin quan trọng. Để tránh những vấn đề này, việc phân tích FM cần phải cân bằng sự đánh đổi giữa việc overfitting và việc regularize các vectơ đặc trưng.

- Thứ hai, nó có thể nhạy cảm với việc lựa chọn các tham số, chẳng hạn như số lượng đặc trưng, tốc độ học và regularization term. Các tham số này có thể ảnh hưởng đến hiệu suất và độ hội tụ của thuật toán nhân tử hóa ma trận.
- Thứ ba, nó có thể bị giới hạn bởi các giả định về tính tuyến tính và tính độc lập, những giả định này có thể không đúng đối với một số dữ liệu hoặc kịch bản. MF giả định rằng xếp hạng là sự kết hợp tuyến tính của các đặc điểm và các đặc điểm này độc lập với nhau. Tuy nhiên, trong một số trường hợp, xếp hạng có thể phụ thuộc vào các đặc điểm phi tuyến tính hoặc tương tác hoặc vào các yếu tố bên ngoài như bối cảnh, thời gian hoặc ảnh hưởng xã hội.
- Thứ 4, không có khả năng mô hình hóa các thông tin bổ trợ về người dùng và sản phẩm.

### **Factorization machine:**

- Ưu điểm:
  - FM là một phương pháp mở rộng của MF ở đó thông tin về sự tương tác giữa nhiều thành phần thông tin khác nhau được mô hình hóa dưới dạng một biểu thức bậc hai hoặc cao hơn. Thông thường, chỉ các tương tác bậc hai được sử dụng để giảm độ phức tạp tính toán.
  - Có khả năng mô hình hóa trên cả đặc trưng thưa thớt và dày đặc.
  - Có khả năng mô hình hóa các thông tin bổ trợ về người dùng và sản phẩm.
  - FM cũng giải quyết được vấn đề “khởi đầu lạnh” khi một người dùng hoặc sản phẩm chưa hề có tương tác nhưng đã có thông tin riêng về người dùng/sản phẩm đó.



- **Nhược điểm:**
  - Nó có thể bị overfitting nếu số lượng hệ số quá lớn hoặc dữ liệu quá .
  - Nó có thể không nắm bắt được các tương tác đặc trưng phi tuyến hoặc phức tạp, thứ không thể xấp xỉ bởi inner products
  - Nó có thể nhạy cảm với việc lựa chọn các siêu tham số, chẳng hạn như regularization term, learning rate.

### **Neural Factorization Machine:**

- **Ưu điểm:**
  - Neural network trong NFM có thể nắm bắt các mối quan hệ phi tuyến tính phức tạp giữa các đặc trưng, dẫn đến độ chính xác dự đoán tốt hơn so với FM, đặc biệt là đối với dữ liệu phức tạp.
  - Tương tự như FM, NFM vượt trội trong việc xử lý dữ liệu thưa thớt, điều này thường gặp trong các hệ thống đề xuất nơi người dùng chỉ có thể tương tác với một phần rất nhỏ các mục.
  - NFM có thể được mở rộng quy mô để xử lý các tập dữ liệu lớn một cách hiệu quả nhờ các kỹ thuật tham số hóa hiệu quả của nó.
  - NFM cho phép kết hợp nhiều loại đặc trưng, bao gồm dữ liệu phân loại và dữ liệu số mà không cần feature engineering thủ công cần thiết trong các mô hình truyền thống.
- **Nhược điểm:**
  - Neural network có độ phức tạp cao hơn khi so với FM. Điều này có thể khiến việc đào tạo và diễn giải kết quả của NFM trở nên khó khăn hơn.
  - Đào tạo NFM đòi hỏi nhiều tài nguyên tính toán hơn so với các mô hình đơn giản hơn như FM do kiến trúc mạng thần kinh phức tạp.
  - Giống như FM, NFM phụ thuộc rất nhiều vào chất lượng và số lượng dữ liệu để có hiệu suất tối ưu. Dữ liệu không đầy đủ có thể dẫn đến việc overfitting hoặc khái quát hóa kém.



- Có thể không sử dụng hiệu quả các thông tin hỗ trợ của người dùng và sản phẩm, dẫn đến cho kết quả thấp hơn so với MF

### **Knowledge Graph Attention Network:**

- Ưu điểm:
  - Mô hình hóa các kết nối bậc cao trong đồ thị tri thức theo kiểu từ đầu đến cuối
  - Sử dụng embedding của các node lân cận để điều chỉnh embedding của 1 node và sử dụng kỹ thuật attention để phân biệt mức độ quan trọng của các lân cận. Điều này giúp tạo các embedding hiệu quả hơn, từ đó giúp hệ thống khuyến nghị có performance tốt hơn.
- Nhược điểm: KGAT có độ phức tạp cao nhất trong các phương pháp được sử dụng. Điều này có thể khiến việc đào tạo và diễn giải kết quả của KGAT trở nên khó khăn hơn.

## **6.2. Hướng phát triển tiềm năng**

### **Dữ liệu:**

- Thu thập thêm các thông tin của người dùng, khóa học để có thể tạo thêm nhiều đặc trưng cho mô hình.

### **Mô hình:**

- Sử dụng thêm thông tin giới tính người dùng cho KGAT
- Thực nghiệm trên các phương pháp khác như BERT4Rec, GRU4Rec, TrueLearn,...

### **Ứng dụng web:**

- Bổ sung các tính năng cần thiết để tạo nên một ứng dụng học tập trực tuyến hoàn chỉnh: gợi ý tên khi nhập tên tìm kiếm người dùng, hỗ trợ phân quyền,...
- Xây dựng web responsive với các thiết bị di động khác như điện thoại, iPad,...



- Sử dụng các cơ sở dữ liệu, công cụ tìm kiếm hiệu quả hơn giúp tăng tốc độ truy vấn.
- Đưa ứng dụng lên cloud computing, tự động hóa toàn bộ quá trình từ ingest data, store data, đến huấn luyện mô hình máy học, triển khai ứng dụng web.
- Tích hợp tính năng gợi ý vào một ứng dụng học tập trực tuyến hiện có, ví dụ như: XueTangX, Coursera,...