Lecture 18: Knowledges and Language Models

Giới thiệu

Hôm nay, tôi muốn thảo luận về việc học từ “knowledge bases” (các cơ sở tri thức). Đây là một sự biến đổi so với nhiều nội dung mà chúng ta đã thực hiện cho đến nay. Tôi sẽ nói về một nguồn thông tin khác và một số thuật toán tương đối khác biệt so với những gì chúng ta đã thảo luận trước đây. Điều này có thể sẽ thú vị và mới mẻ, vì vậy hãy cùng bắt đầu.

Knowledge Bases

Cơ sở tri thức là các cơ sở dữ liệu có cấu trúc chứa đựng tri thức. Chúng có thể bao gồm nhiều thành phần, nhưng phổ biến nhất là cơ sở tri thức quan hệ, bao gồm các thực thể (entities) là các nút trong đồ thị và các quan hệ (relations) là các cạnh giữa các nút.

Một số câu hỏi quan trọng mà chúng ta đặt ra về cơ sở tri thức là:

- Làm thế nào để học cách tạo ra và mở rộng cơ sở tri thức bằng các phương pháp dựa trên mạng nơ-ron?
- Làm thế nào để học từ thông tin trong cơ sở tri thức nhằm cải thiện các mô hình mạng nơ-ron hoặc sử dụng chúng một cách hiệu quả?
- Làm thế nào để sử dụng tri thức có cấu trúc để trả lời các câu hỏi?

Types of Knowledge Bases

WordNet (Miller 1995)

WordNet là một cơ sở dữ liệu lớn chứa các từ vựng được tổ chức theo cấu trúc đồ thị, trong đó mỗi nút (node) được gọi là "synset" - tập hợp các từ đồng nghĩa. Mỗi synset có thể là danh từ, động từ hoặc tính từ và được kết nối với nhau thông qua các mối quan hệ ngữ nghĩa.

Các mối quan hệ chính trong WordNet:

- Với danh từ:
  - ➢ Quan hệ "is-a": ví dụ "hatchback is a car"
  - ➢ Quan hệ "part-of": ví dụ "wheel is part of car"
  - ➢ Phân biệt giữa type và instance: "Joe Biden" là một instance của type "president"
- Với động từ: được sắp xếp theo mức độ cụ thể
  - ➢ "communicate" là khái niệm rộng hơn "talk"
  - ➢ "whisper" là khái niệm hẹp hơn "talk"
- Với tính từ: chủ yếu là quan hệ trái nghĩa (antonym)
  - ➢ Ví dụ: "wet" vs "dry"

Mặc dù từng rất phổ biến, hiện nay WordNet ít được sử dụng hơn vì có nhiều giải pháp thay thế hiệu quả hơn như:

- Sử dụng word embeddings để tìm các từ có ngữ nghĩa gần nhau

- Sử dụng các mô hình ngôn ngữ lớn như Mistral để trực tiếp phân tích và trích xuất thông tin từ văn bản

Điểm đặc biệt của WordNet là mặc dù có tên là "Word"-net nhưng mỗi node không phải là một từ đơn lẻ mà là một tập hợp các từ đồng nghĩa. Ví dụ "artifact" và "thing" có thể cùng thuộc một synset vì chúng có nghĩa tương tự nhau.

Cyc (Lenant 1995)

Một trong những cơ sở dữ liệu đáng chú ý là Psych, được phát triển song song với WordNet. Đây là một cơ sở dữ liệu được biên soạn thủ công, nhằm mã hóa tất cả kiến thức thông thường. Dự án này kéo dài khoảng 30 đến 40 năm và có thể vẫn còn tồn tại đến ngày nay.

Psych xây dựng một hệ thống phân cấp lớn về các loại kiến thức khác nhau, bao gồm thông tin về các sự kiện và thứ tự xảy ra của chúng. Tuy nhiên, vấn đề lớn nhất của dự án này là tính tham vọng của nó; việc mã hóa tất cả kiến thức này bằng tay là không khả thi. Mặc dù dự án đã đạt được một số tiến bộ, nhưng những gì đạt được vẫn chưa đủ để ứng dụng trong các hệ thống thực tiễn. Do đó, phương pháp này hiện không còn được sử dụng phổ biến.

DBPedia (Auer et al. 2007)

DBpedia là một trong những cơ sở dữ liệu tri thức phổ biến nhất hiện nay, được phát triển như một sự kế thừa của dự án Psych. Khác với Psych, nơi mà các chuyên gia phải tham gia vào việc biên soạn các quy tắc cho máy móc, DBpedia dựa vào sự đóng góp của một lượng lớn người dùng để tạo ra dữ liệu có cấu trúc về các thực thể trên thế giới.

Mục tiêu của DBpedia là phục vụ cho con người, vì vậy thông tin được biên soạn sẽ xuất hiện trên các trang Wikipedia. Ví dụ, trang của "Carnegie Mellon University" không chỉ cung cấp tên gọi cũ mà còn có khẩu hiệu, loại hình thực thể, người sáng lập và thời gian thành lập.

Sự tham gia của cộng đồng là động lực chính giúp DBpedia thu thập một lượng dữ liệu đáng kể, bao quát nhiều thực thể nổi bật trên toàn cầu, đặc biệt là những thực thể có sự tham gia cao trên Wikipedia.

WikiData (Bollacker et al. 2008)

Hiện nay, nhiều người sử dụng một công cụ gọi là WikiData. Tên gọi này có phần không chính xác vì WikiData không hoàn toàn liên kết chặt chẽ với Wikipedia. Nó không chỉ trích xuất dữ liệu từ Wikipedia mà còn từ nhiều nguồn khác nhau. WikiData là một cơ sở dữ liệu được biên soạn về các thực thể, có quy mô rất lớn và hỗ trợ đa ngôn ngữ.

Một ví dụ điển hình là nhóm nghiên cứu Mamba, viết tắt của Modeling and Analysis for Medicine Research Group, tập trung vào sinh học, toán học và thuộc Trung tâm Nghiên cứu Khoa học Quốc gia tại Pháp. Thông tin về nhóm nghiên cứu này được tổ chức thành các nút

trong đồ thị, với các cạnh được gán nhãn như "instance of" và "research group". Điều này cho phép người dùng dễ dàng truy cập và tìm hiểu thêm về các thực thể.

Một điểm thú vị khác là WikiData có một ngôn ngữ truy vấn gọi là SPARQL, tương tự như SQL nhưng dành cho các cơ sở dữ liệu tri thức. Ví dụ, bạn có thể viết một truy vấn SPARQL để tìm tất cả các chủ tịch của Carnegie Mellon University. SPARQL cho phép bạn thực hiện các truy vấn phức tạp và thu thập thông tin một cách hiệu quả.

Tuy nhiên, có những câu hỏi mà các mô hình ngôn ngữ như ChatGPT gặp khó khăn trong việc trả lời, chẳng hạn như "kể tên tất cả các vị tổng thống sinh ra ở phía đông sông Mississippi". Việc sử dụng SPARQL sẽ giúp bạn dễ dàng hơn trong việc truy xuất thông tin chính xác từ các cơ sở dữ liệu tri thức như WikiData.

Learning Representations for Knowledge Bases

Knowledge Base Incompleteness

Trong phần này, chúng ta sẽ tìm hiểu về việc học các biểu diễn cho cơ sở tri thức. Mặc dù cơ sở tri thức rất hữu ích, nhưng một vấn đề lớn là chúng thường không đầy đủ. Ngay cả với quy mô rất lớn, việc đảm bảo tính hoàn chỉnh là điều không thể. Chẳng hạn, trong Freebase, một nền tảng tiền thân của WikiData, có đến 71% số người không có ngày sinh. Tuy nhiên, thực tế là hầu như mọi người đều có ngày sinh.

Đối với những thực thể nổi tiếng, chúng ta có thể thu thập rất nhiều thông tin chi tiết, như về Joe Biden hay Barack Obama. Nhưng đối với những thực thể ít nổi bật hơn, mặc dù vẫn muốn đưa chúng vào cơ sở tri thức, nhưng việc thu thập toàn bộ thông tin có thể gặp khó khăn, hoặc thậm chí không nên làm vì lý do bảo mật.

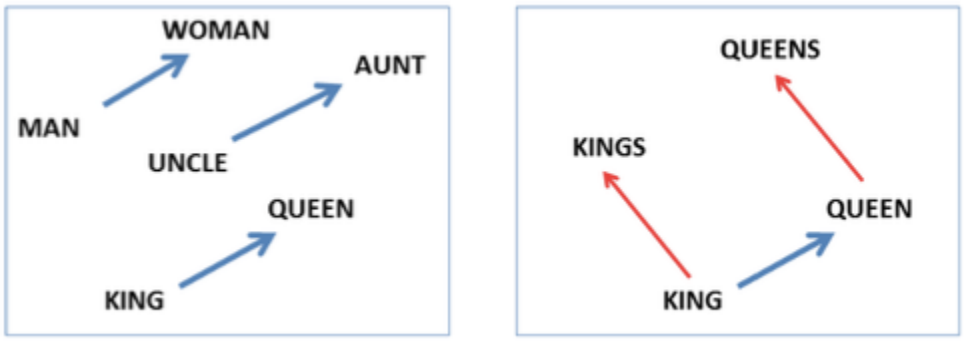
Vì vậy, một ý tưởng là sử dụng phương pháp trích xuất quan hệ (relation extraction) để lấy thông tin từ internet, từ đó tạo ra các cơ sở tri thức riêng. Điều này cũng có thể hữu ích khi bạn muốn xây dựng cơ sở tri thức cho một lĩnh vực chuyên biệt hoặc các mục đích khác.

Consistency in Embeddings

Có nhiều phương pháp khác nhau để thực hiện trích xuất quan hệ. Một trong những cách phổ biến mà nhiều người đã thử là tận dụng tính nhất quán trong không gian embedding. Ví dụ nổi tiếng nhất về điều này xuất phát từ nghiên cứu về Word2Vec, được công bố vào năm 2013. Trong bài báo về Word2Vec, một trong những điểm nổi bật là việc chứng minh rằng các vector trong không gian embedding không chỉ có ý nghĩa mà còn có thể biểu thị các quan hệ giữa các từ.

Cụ thể, các vector này có thể thể hiện mối quan hệ giữa các từ như "man" và "woman" theo cùng một hướng với "Uncle" và "Aunt", hay "King" và "Queen". Điều này cho phép thực hiện các phép toán vector như lấy vector của "King", trừ đi vector biểu thị tính số nhiều, sau đó cộng

với vector chuyển từ từ nam tính sang nữ tính. Kết quả là, bạn có thể xác định được từ số nhiều chỉ bằng cách sử dụng hai vector này, ví dụ như tìm ra từ số nhiều của "queen" chỉ bằng cách biết hai vector đó.

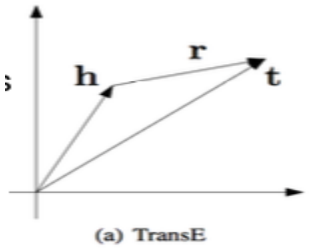


Learning Knowledge Graph Embeddings (Bordes et al. 2013)

Việc học các embeddings từ văn bản hoặc sử dụng cơ sở tri thức lớn như WikiData để cải thiện embeddings của mô hình neural là một hướng nghiên cứu quan trọng. Một trong những thách thức lớn là làm thế nào để có được embeddings tốt cho một Knowledge Graph. Điều này đặc biệt quan trọng khi bạn muốn thực hiện các tác vụ như Knowledge Graph Search.

Knowledge Graph có thể chứa thông tin về nhiều thực thể hiếm gặp mà không được đề cập nhiều trên internet. Do đó, bạn có thể kết hợp cấu trúc của Knowledge Graph với văn bản để học các embeddings tốt hơn. Một ví dụ điển hình là một bài báo nghiên cứu đã đề xuất cách biểu diễn các triples của Knowledge Graph dưới dạng các phép biến đổi cộng và tối thiểu hóa khoảng cách của các triples hiện có bằng một hàm mất mát dựa trên biên (margin-based loss).

$$\sum_{(h,\ell,t)\in S} \sum_{(h',\ell,t')\in S'_{(h,\ell,t)}} \left[\gamma + d(\boldsymbol{h}+\boldsymbol{\ell},\boldsymbol{t}) - d(\boldsymbol{h'}+\boldsymbol{\ell},\boldsymbol{t'})\right]_+$$



(a) TransE

Cụ thể, trong phương pháp này, họ sử dụng các vector để biểu diễn các thành phần của triple: "head" (H), "tail" (T), và "relation" (L hoặc R). Mục tiêu là chuyển từ H đến T theo vector quan hệ R, và sử dụng hàm mất mát hinge. Trong hàm này, bạn có một tham số hinge và cố gắng tăng trọng số cho các ví dụ triple đúng và giảm trọng số cho các ví dụ triple sai, ví dụ như những triple được lấy mẫu ngẫu nhiên để trở thành không chính xác.

Relation Extraction with Neural Tensor Networks (Socher et al. 2013)

Một điều thú vị về Knowledge Graph Embeddings là nhiều nhà nghiên cứu AI nổi tiếng đã bắt đầu sự nghiệp từ lĩnh vực này. Điển hình như Richard Socher - hiện là CEO của công ty vi.com, một công ty chuyên về công cụ tìm kiếm.

Trong nỗ lực đầu tiên để dự đoán các mối quan hệ (relations) trong knowledge graph, các nhà nghiên cứu đã tạo ra một mô hình MLP (Multi-Layer Perceptron). Mô hình này được thiết kế với công thức cơ bản: $u_R^T f\left(W_{R,1} e_1 + W_{R,2} e_2\right)$

Trong đó:

- u_R: vector học được cho relation R
- f: hàm phi tuyến (activation function)
- W_R,1 và W_R,2: ma trận trọng số cho relation R tương ứng với hai entities
- e1 và e2: vector embedding của hai entities

Kết quả được đưa qua hàm sigmoid:

- Nếu kết quả là 1: relation có khả năng tồn tại
- Nếu kết quả là 0: relation không có khả năng tồn tại

Tiếp theo, họ đề xuất "Neural Tensor Network" với một bilinear feature extractor, được biểu diễn bằng công thức:

$$g(e_1, R, e_2) = u_R^T f\left(e_1^T W_R^{[1:k]} e_2 + V_R \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + b_R\right)$$

Trong đó:

- g(e1, R, e2): hàm scoring cho một bộ ba (triple) gồm hai entities e1, e2 và relation R
- e1^T: chuyển vị của vector embedding của entity thứ nhất
- W_R^[1:k]: tensor trọng số cho relation R có k lớp
- V_R: ma trận trọng số bổ sung cho relation R
- [e1; e2]: vector nối (concatenation) của hai entity embeddings
- b_R: vector bias cho relation R

Cấu trúc này có những điểm đáng chú ý:

- Có embedding ở cả hai phía
- Một ma trận transformation
- Tính dot product giữa các embedding sau khi transformation
- Cấu trúc này khá giống với cơ chế attention, đặc biệt là bilinear attention
- Kết hợp với MLP và một bias term

Mặc dù đây là một mô hình mạnh mẽ, nhưng nó bị overparameterized (quá nhiều tham số). Về sau, các nhà nghiên cứu chuyển sang sử dụng các mô hình đơn giản hơn, chủ yếu dựa trên các phép chiếu tuyến tính giữa hai phía.

Các phương pháp này được áp dụng trong hai trường hợp:

1. Khi đã có Knowledge Graph embeddings và muốn học các relations
2. Khi chưa có thông tin về knowledge graph và muốn học chính các knowledge graph embeddings

Hiện nay, những phương pháp này chủ yếu được sử dụng cho việc học Knowledge Graph embeddings, một bước quan trọng trong việc xây dựng các mô hình dựa trên Knowledge Graph.

Learning from Text Directly

Distance Supervision for Relation Extraction (Mintz et al. 2009)

Một phương pháp phổ biến để trích xuất quan hệ là học từ văn bản trực tiếp. Một câu hỏi quan trọng là làm thế nào để có được dữ liệu huấn luyện cho việc trích xuất quan hệ. Một bài báo có ảnh hưởng lớn trong lĩnh vực này là "Distant Supervision for Relation Extraction", được coi là một trong những bài báo đầu tiên hoặc ít nhất là có ảnh hưởng lớn về tăng cường dữ liệu hoặc dữ liệu tổng hợp cho xử lý ngôn ngữ tự nhiên.

Ý tưởng cơ bản là sử dụng một cơ sở tri thức đã có sẵn, như WikiData, với các bộ ba thực thể-quan hệ-thực thể. Từ đó, có thể trích xuất tất cả các văn bản khớp với loại quan hệ cụ thể này và sử dụng chúng để huấn luyện một mô hình trích xuất quan hệ có giám sát. Ví dụ, nếu cơ sở tri thức có thông tin rằng Steven Spielberg là đạo diễn của bộ phim "Saving Private Ryan", hệ thống sẽ tìm tất cả các câu có chứa cả "Steven Spielberg" và "Saving Private Ryan" và gán nhãn chúng là ví dụ tích cực cho quan hệ này.

Phương pháp này thường hoạt động khá tốt, nhưng cũng có thể tạo ra các ví dụ tiêu cực. Chẳng hạn, câu "Steven Spielberg's film Saving Private Ryan" không thực sự chỉ ra rằng ông là đạo diễn; ông có thể là nhà biên kịch hoặc diễn viên. Do đó, mặc dù đây là một cách tạo dữ liệu gần như miễn phí, nhưng cũng có thể tạo ra các ví dụ nhiễu, gây ra vấn đề cho mô hình.

Relation Classification with Neural Nets (Zeng et al. 2014)

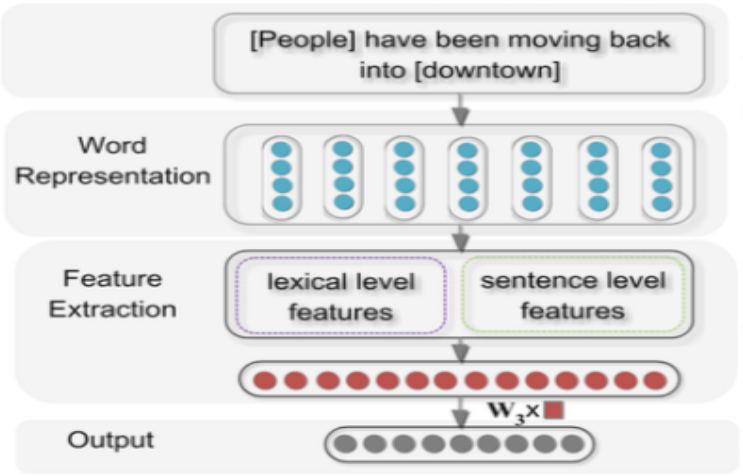
Phân loại quan hệ (relation classification) trong mạng neural đã có nhiều phương pháp nghiên cứu khác nhau. Phần lớn các phương pháp đều hoạt động bằng cách trích xuất đặc trưng và sau đó phân loại, mặc dù gần đây đã có một số phương pháp dựa trên mô hình ngôn ngữ lớn.

Một điểm đáng chú ý về trích xuất quan hệ nói riêng và trích xuất thông tin nói chung là chúng thường được áp dụng trên tập dữ liệu rất lớn như toàn bộ internet. Mặc dù có thể sử dụng Mistral hay GPT-4 để có câu trả lời, chi phí để chạy các mô hình này trên toàn bộ internet là rất cao. Do đó, việc có các phương pháp nhẹ và tiết kiệm vẫn mang lại nhiều lợi ích.

Về mặt kỹ thuật, các phương pháp hiện đại thường trích xuất đặc trưng từ:

- Phần đầu của thực thể thứ nhất

- Phần cuối của thực thể thứ nhất
- Phần đầu của thực thể thứ hai
- Phần cuối của thực thể thứ hai



Hình: Kiến trúc mạng nơ-ron sử dụng để phân loại quan hệ

Các embedding này sau đó được đưa vào một MLP (Multi-layer Perceptron) để dự đoán xem quan hệ có tồn tại hay không. Quá trình này khá đơn giản nếu bạn có sẵn mô hình embedding - chỉ cần đưa nó qua RoBERTa hoặc Mistral để lấy embedding cho từng token, rồi thực hiện dự đoán dựa trên bốn embedding đó.

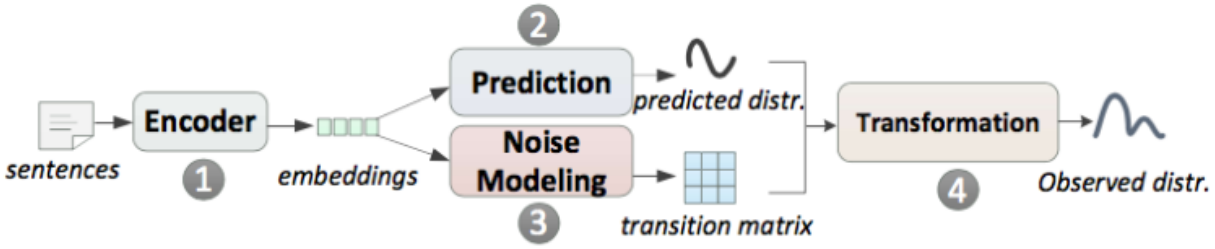
Modeling Distant Supervision Noise in Neural Models (Luo et al. 2017)

Một khía cạnh thú vị trong trích xuất quan hệ là cách xử lý nhiễu, đặc biệt khi làm việc với dữ liệu được gán nhãn bán tự động (semi-noisy data). Nhiều nghiên cứu đã tập trung vào việc xử lý nhiễu trong distant supervision.

Phương pháp được đề xuất trong bài báo này sử dụng hai tập dữ liệu:

1. Một tập dữ liệu nhỏ (khoảng 5,000 mẫu) có chất lượng cao, được gán nhãn thủ công và có độ tin cậy cao
2. Một tập dữ liệu lớn hơn nhiều (khoảng 5 triệu mẫu) được gán nhãn tự động, có thể chứa nhiễu

Quy trình xử lý bao gồm:



- Sử dụng encoder để tạo embedding và dự đoán trên tập dữ liệu chất lượng cao
- Thêm một lớp mô hình hóa nhiễu (noise modeling layer) sử dụng ma trận chuyển đổi (transition matrix)
- Ma trận này chuyển đổi xác suất dự đoán để tạo ra phân phối cuối cùng có tính đến nhiễu

Để cải thiện hiệu quả, họ áp dụng "Trace normalization" lên ma trận chuyển đổi, giúp đưa ma trận này gần hơn với hàm định danh (identity function). Điều này dựa trên giả định rằng phần lớn các dự đoán là chính xác, dù không phải tất cả đều đúng.

Phương pháp này không chỉ hữu ích cho trích xuất quan hệ mà còn có thể áp dụng cho các bài toán khác có dữ liệu nhiễu. Đây là một hướng nghiên cứu đáng quan tâm cho việc xử lý dữ liệu được gán nhãn có nhiễu trong các ứng dụng thực tế.

Using Knowledge Bases to Inform Neural Models

Retrofitting of Embeddings to Existing Lexicons (Faruqui et al. 2015)

Trong phần này, chúng ta sẽ tìm hiểu cách sử dụng cơ sở tri thức để cải thiện các mô hình ngữ nghĩa, đặc biệt là cách cải thiện embeddings bằng cách sử dụng các từ điển hiện có.

Một phương pháp phổ biến là cải thiện embeddings không ngữ cảnh (non-contextual embeddings) bằng cách sử dụng các từ điển hiện có. Ví dụ, trong một nghiên cứu, các nhà nghiên cứu đã điều chỉnh embeddings để phù hợp hơn với từ điển bằng cách thực hiện các phép biến đổi sau khi huấn luyện (post hoc transformations).

Họ bắt đầu với embeddings đã được huấn luyện trước và đặt ra mục tiêu kép: làm cho embeddings sau khi biến đổi gần với các từ láng giềng và gần với embeddings gốc. Điều này được thực hiện thông qua một thuật ngữ điều chuẩn (regularization term) nhằm giữ cho embeddings không di chuyển quá xa so với vị trí ban đầu, đồng thời làm cho các embeddings của các từ đồng nghĩa gần nhau hơn. Họ sử dụng WordNet để điều chỉnh các từ đồng nghĩa và cũng có các ví dụ về việc đẩy các từ trái nghĩa ra xa nhau.

$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j)\in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

Mặc dù phương pháp trên áp dụng cho embeddings không ngữ cảnh, chúng ta có thể thực hiện điều tương tự cho các mô hình hiện đại như Knowledge Graph embeddings. Ví dụ, nếu chúng ta có một mô hình nhận diện các thực thể và các ví dụ khác nhau của những thực thể đó trong các ngữ cảnh khác nhau, chúng ta có thể khuyến khích embeddings của các thực thể này gần nhau hơn. Điều này có thể cải thiện các tác vụ như trả lời câu hỏi hoặc tra cứu thông tin.

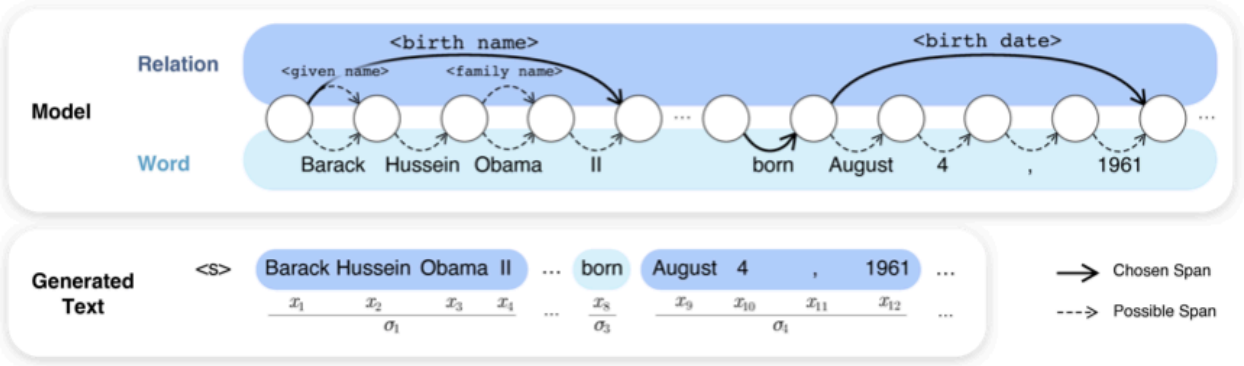
Một câu hỏi thú vị được đặt ra là: "Điều gì xảy ra nếu bạn thực hiện subword modeling và không có embedding cho toàn bộ chuỗi ký tự đó?" Có nhiều cách để xử lý vấn đề này. Một phương

pháp là sử dụng mô hình để lấy embedding từ đầu và cuối của một đoạn văn bản, cùng với embedding trung bình của tất cả các từ trong đoạn đó. Sau đó, các vector này được đưa qua một mạng nơ-ron để tạo ra một embedding mới. Phương pháp này có thể được sử dụng cho nhiều tác vụ khác nhau như nhận diện thực thể có tên, trích xuất quan hệ, và nhiều tác vụ phân tích khác.

Injecting Knowledge into Language Models (Hayashi et al. 2020)

Một câu hỏi quan trọng là làm thế nào để tích hợp kiến thức vào các mô hình ngôn ngữ. Có nhiều cách khác nhau để thực hiện điều này. Một phương pháp đơn giản là tra cứu kiến thức liên quan trong đồ thị tri thức (Knowledge Graph) và cung cấp nó cho mô hình thông qua prompting. Tuy nhiên, vấn đề với phương pháp này là mô hình có thể không tận dụng được kiến thức ít phổ biến, do các biểu diễn (embeddings) của các thực thể có thể chưa được học tốt.

Một phương pháp khác mà chúng tôi đã đề xuất là thay vì dự đoán trực tiếp các từ, mô hình có thể dự đoán các thẻ như "birth name", "given name", hay "family name". Sau đó, mô hình sẽ điền thông tin từ cơ sở tri thức vào các thẻ này. Ví dụ, khi viết một bài viết Wikipedia về Barack Obama, mô hình có thể dự đoán "birth name, born in birth date", một cấu trúc rất phổ biến trên Wikipedia. Điều này giúp giảm thiểu hiện tượng "hallucination" và đảm bảo thông tin chính xác.



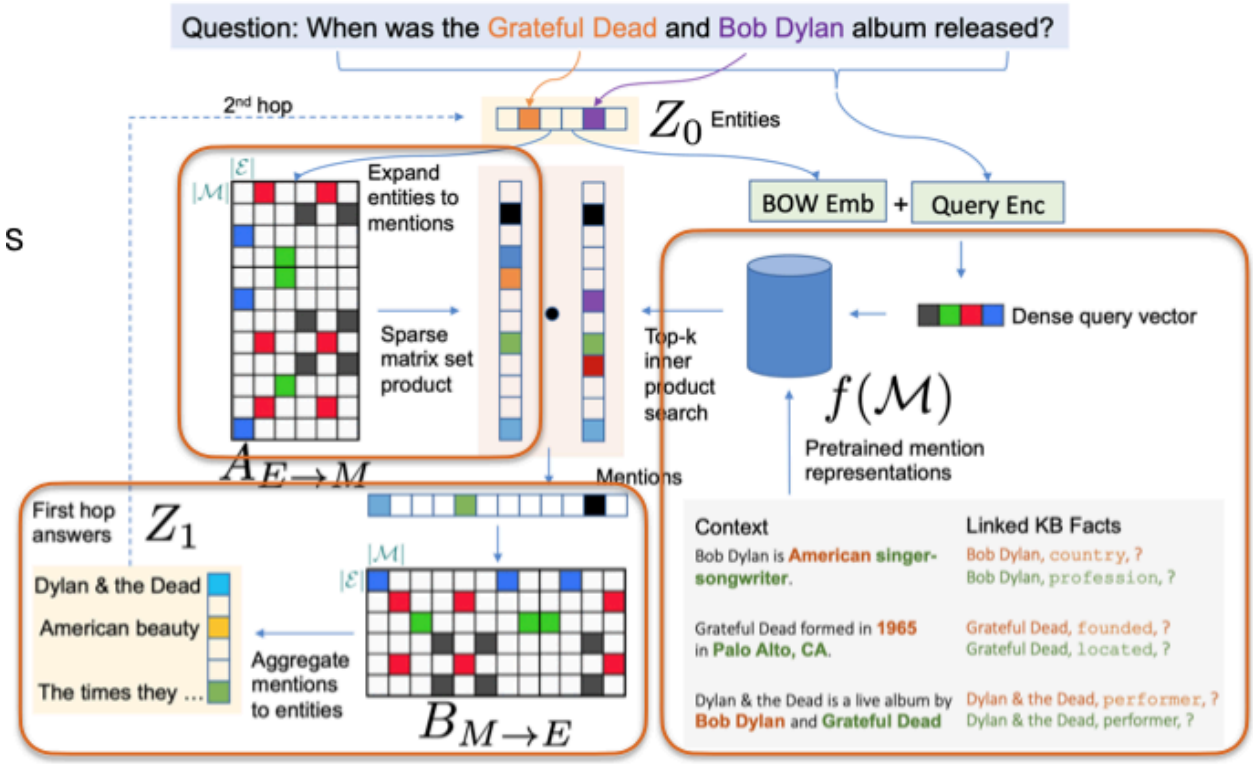
Có nhiều cách để tích hợp kiến thức vào mô hình ngôn ngữ. Một cách là sử dụng prompting, cách khác là tạo ra các mẫu (templatic generation) với các chỗ trống để điền thông tin từ cơ sở tri thức. Chi tiết về cách xây dựng mục tiêu huấn luyện cho phương pháp này được trình bày trong bài báo, bao gồm cả việc xác định khi nào cần thay thế thông tin và cách lựa chọn giữa các thẻ như "birth name" hay "given name".

Reasoning over Text Corpus as a Knowledge Base (Dhingra et al. 2020)

Một câu hỏi thú vị mà chúng ta thường gặp, và cũng là chủ đề của một bài báo bốn năm trước, đó là: Làm sao chúng ta có thể lý luận dựa trên một lượng lớn văn bản như cách chúng ta lý luận dựa trên cơ sở tri thức?

Trong nghiên cứu này, các tác giả đã trả lời các câu hỏi sử dụng tập văn bản như là các cơ sở tri thức có thể truy vết. Họ thực hiện việc khớp độ tương thích dựa trên các tham chiếu (mentions). Họ đã tạo ra các vector tham chiếu, đại diện cho tất cả các tham chiếu trong cơ sở tri thức của các thực thể cụ thể và lấy các tham chiếu liên quan từ các mô hình tiền huấn luyện. Họ cũng chạy embeddings và tạo ra embeddings cho từng tham chiếu trong toàn bộ tập văn bản.

Quá Trình Huấn Luyện



- Tạo Vector Truy Vấn: Họ tạo ra một vector truy vấn dày đặc, được huấn luyện để nhận diện các tham chiếu thực thể trả lời câu hỏi. Ví dụ, câu hỏi "When was The Grateful Dead and Bob Dylan album released?" sẽ có vector của "Bob Dylan" và "The Grateful Dead". Mô hình sẽ được huấn luyện để khi lấy embedding của thực thể này và đối chiếu với embedding trong tập mã hóa lớn, nó sẽ trả về thông tin liên quan nhất.

- Huấn Luyện với Giám sát yếu (Weak Supervision): Chúng tôi sử dụng Giám sát yếu bằng cách lấy một cơ sở tri thức lớn hiện có và xác định tất cả các câu mà câu trả lời được bao gồm. Ví dụ, "Steven Spielberg is the director of Saving Private Ryan". Chúng tôi tạo ra các câu hỏi như "Who was the director of Saving Private Ryan?" và upweight tất cả các embedding của "Saving Private Ryan" trong toàn bộ Wikipedia nơi "Steven Spielberg" cùng xuất hiện trong câu đó.

Điều này giúp mô hình học được cách đối chiếu các câu hỏi nhân tạo với các câu có thể trả lời câu hỏi đó. Nhờ vào lượng dữ liệu lớn để huấn luyện từ Giám sát yếu, chúng ta có thể phát triển một mô hình mạnh mẽ.

Đây là một ví dụ về cách chúng ta có thể thực hiện phương pháp RAG được thông tin bởi các cơ sở tri thức. Phương pháp này có thể được áp dụng rộng rãi trong nhiều hệ thống hiện đại ngày nay.

Schema-Free Extraction

Một khái niệm quan trọng là "schema-free extraction" (trích xuất không dựa trên lược đồ). Để hiểu rõ hơn, hãy xem xét WikiData, nơi có một lược đồ xác định các quan hệ trong cơ sở dữ liệu, chẳng hạn như "instance of", "image", "signature", "sex or gender", và "country of citizenship". Những quan hệ này được quyết định trước bởi những người tạo ra WikiData. Tuy nhiên, điều này không đảm bảo rằng tất cả các quan hệ cần thiết đều có sẵn, tương tự như vấn đề thiếu các thực thể.

Ví dụ, khi chuẩn bị cho bài giảng về mô hình ngôn ngữ lớn, giáo sư đã tạo ra một số dữ liệu có cấu trúc về các mô hình này, bao gồm đa dạng "positional embedding". Đây là một khái niệm không có trong WikiData. Khi đi sâu vào các khái niệm chuyên ngành hoặc lĩnh vực đặc thù, thường sẽ thiếu các thực thể hoặc quan hệ cần thiết.

Vấn đề này là điều mà "schema-free extraction" cố gắng giải quyết. Nó tìm cách xác định lược đồ cùng với thông tin mà bạn muốn trích xuất, giúp mở rộng khả năng xử lý dữ liệu mà không bị giới hạn bởi lược đồ có sẵn.

Open Information Extraction (Banko et al 2007)

Một ví dụ nổi bật là "open information extraction" (trích xuất thông tin mở). Phương pháp này không yêu cầu một schema (lược đồ) cố định; thay vào đó, lược đồ duy nhất chính là văn bản trong các câu mà chúng ta đang phân tích. Ví dụ, với câu "United has a Hub in Chicago which is the headquarters of United Continental Holdings", quan hệ được trích xuất là "has a Hub in".

Tuy nhiên, vấn đề của phương pháp này là không thể trừu tượng hóa các quan hệ. Nếu có một câu khác như "United Continental Holdings has its headquarters in Chicago", thì hệ thống sẽ coi đây là một quan hệ hoàn toàn khác và không thể nhóm hai câu này lại với nhau.

Rule-based Open IE

Trong lĩnh vực trích xuất thông tin mở (Open Information Extraction), nhiều phương pháp hiện nay vẫn sử dụng hệ thống dựa trên luật, một trong số ít lĩnh vực mà cách tiếp cận này vẫn gần như đạt đến mức tiên tiến. Lý do là việc trích xuất các chuỗi liên quan giữa hai thực thể không quá phức tạp, do đó cả độ chính xác (Precision) và độ bao phủ (Recall) đều khá cao.

Ngoài ra, việc sử dụng hệ thống dựa trên luật còn do chi phí thấp hơn khi chạy trên toàn bộ web so với mô hình học sâu. Một số ví dụ về hệ thống này bao gồm TextRunner và ReVerb. Ý tưởng cơ bản là sử dụng bộ phân tích cú pháp để thực hiện phân tích cú pháp của câu theo các quy tắc, chẳng hạn như quan hệ phải chứa một động từ, chủ ngữ và tân ngữ phải là cụm danh từ.

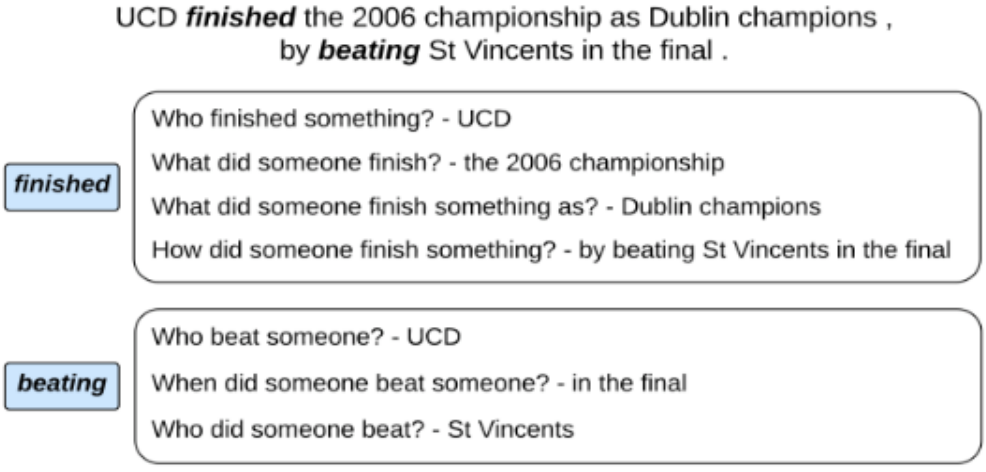
Trong một số nghiên cứu, người ta đã huấn luyện một mô hình nhanh hơn để trích xuất dữ liệu lớn, sử dụng hệ thống dựa trên luật như một dạng giám sát yếu (weak supervision) và sau đó huấn luyện một mô hình dựa trên chuỗi để thực hiện nhanh hơn.

Một phương pháp khác là tổng hợp nhiều bằng chứng để tìm ra các trích xuất chung và do đó có thể đáng tin cậy. Ví dụ, bất kỳ thông tin nào trên internet đều có thể sai lệch. Nếu ai đó viết trên blog rằng "United has a Hub in Pittsburgh", điều này có thể sai. Họ sử dụng các phương pháp dựa trên tần suất để lọc thông tin sai lệch, ví dụ như nếu "United" và "Pittsburgh" đều phổ biến nhưng câu "United has a Hub in Pittsburgh" rất hiếm, thì khả năng cao thông tin này không chính xác.

Neural Models for Open IE

Có nhiều mô hình mạng nơ-ron được áp dụng cho Open Information Extraction (Open IE), mặc dù chúng có thể ít được sử dụng hơn so với các phương pháp dựa trên heuristic. Một vấn đề lớn khi không dựa vào heuristic là cần có dữ liệu huấn luyện, điều này không phải lúc nào cũng dễ dàng. Một nghiên cứu thông minh đã chỉ ra rằng có thể tạo ra các bộ dữ liệu lớn bằng cách đặt những câu hỏi đơn giản cho con người.

Cụ thể, nghiên cứu này tập trung vào việc tạo ra các bộ dữ liệu trích xuất quan hệ. Ví dụ, câu hỏi như "Who finished something?" có thể có câu trả lời là "UCD", và "What did someone finish?" có thể có câu trả lời là "the 2006 Championship". Thay vì yêu cầu người tham gia chọn các phần tử như thực thể đầu, quan hệ, và thực thể đuôi trên giao diện phức tạp, họ chỉ cần trả lời các câu hỏi đơn giản, từ đó suy ra được quan hệ cần trích xuất.



Phương pháp này tạo ra cái gọi là "semantic roles", có thể được sử dụng để chú thích dữ liệu trích xuất quan hệ. Sau đó, họ huấn luyện một mô hình mạng nơ-ron có giám sát để thực hiện tác vụ này.

Learning Relations from Relations

Modeling Word Embeddings vs. Modeling Relations

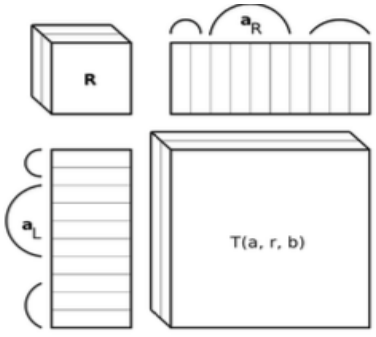
Một chủ đề thú vị mà tôi muốn đề cập là việc học thông tin về các thực thể từ các embedding thực thể (entity embeddings). Tuy nhiên, chúng ta cũng có thể học thông tin về các quan hệ từ thông tin quan hệ của các quan hệ khác. Điều này có thể giúp giải quyết vấn đề mà Open IE không thể trừu tượng hóa và tổng quát hóa.

Embedding từ hay embedding thực thể cung cấp thông tin về từ trong ngữ cảnh, điều này có thể chỉ ra kiến thức cho các cơ sở tri thức. Tuy nhiên, các quan hệ khác hoặc sự kết hợp của chúng cũng có thể chỉ ra điều tương tự. Nếu ai đó quen thuộc với đồ thị hoặc xử lý đồ thị, có một ý tưởng gọi là "link prediction", nơi bạn được cung cấp một số lượng nhỏ các liên kết trong một đồ thị và bạn muốn dự đoán các liên kết khác có khả năng tồn tại.

Tensor Decomposition (Sutskever et al. 2009)

Nhiều nhà nghiên cứu AI nổi tiếng đã bắt đầu sự nghiệp của họ từ lĩnh vực trích xuất quan hệ, và Sutskever là một trong số đó. Bài báo năm 2009 của ông đề xuất sử dụng phân rã tensor (tensor decomposition) để thực hiện quy nạp quan hệ.

Phương pháp này hoạt động bằng cách mô hình hóa các mối quan hệ thông qua việc phân rã một tensor chứa các bộ ba entity-relation-entity. Cụ thể, tensor này bao gồm: thực thể bên trái, thực thể bên phải, và ở giữa là một tensor lớn thể hiện sự tồn tại của các mối quan hệ. Trong đó, các thực thể được biểu diễn dưới dạng embedding vectors.



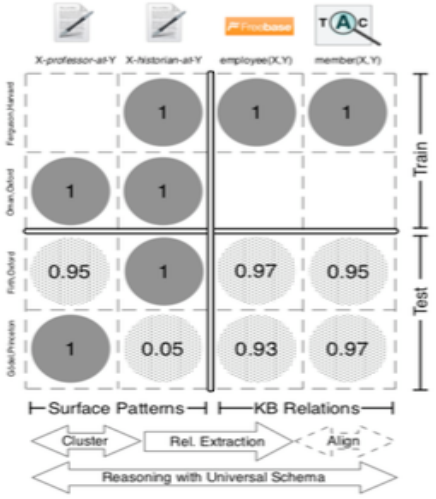
Khi chúng ta biết chắc chắn một mối quan hệ tồn tại, ta gán giá trị 1, ngược lại gán giá trị 0 cho những mối quan hệ không tồn tại. Sau đó, chúng ta thực hiện phép xấp xỉ hạng thấp (low rank approximation) của tensor này. Quá trình này tạo ra một tensor tái tạo, trong đó một số giá trị ban đầu là 0 có thể được chuyển thành gần với 1. Những trường hợp này được xem là các mối

quan hệ tiềm năng - những quan hệ có thể tồn tại trong thực tế nhưng chưa được ghi nhận trong cơ sở tri thức ban đầu do tính không đầy đủ của dữ liệu.

Matrix Factorization to Reconcile Schema-based and Open IE Extractions (Riedel et al. 2013)

Một bước tiến xa hơn trong lĩnh vực này là việc xử lý khi chúng ta có một hoặc nhiều cơ sở tri thức, ví dụ như Wikidata, WordNet và các nguồn khác, kết hợp với các trích xuất từ Open IE. Ý tưởng này được gọi là Universal Schema - một phương pháp nhúng (embed) các quan hệ từ nhiều schema khác nhau vào cùng một không gian vector. Dựa trên điều này, hệ thống có thể dự đoán những quan hệ nào có khả năng tồn tại hoặc không.

Ví dụ, chúng ta có thể có dữ liệu từ Freebase hoặc Wikidata, kết hợp với một tập dữ liệu trích xuất quan hệ khác như TAC. Trong tập huấn luyện, ta có các mẫu được gán nhãn tích cực hoặc tiêu cực. Trong khi đó, ở tập kiểm thử (held-out dataset), ta chỉ có thông tin từ Open IE. Đối với các thực thể đã tồn tại trong cơ sở tri thức, chúng ta biết được các quan hệ liên quan, nhưng với những thực thể chưa có trong cơ sở dữ liệu, ta không có thông tin này. Tuy nhiên, chỉ từ sự tồn tại hoặc không tồn tại của các quan hệ trong Open IE, chúng ta có thể dự đoán được các quan hệ khác.



Đây là một cách hiệu quả để kết hợp ưu điểm của hai phương pháp: Open IE có thể chạy trên các tập dữ liệu rất lớn nhưng không có schema cụ thể, trong khi Wikidata có schema tốt nhưng được tạo thủ công. Một điểm thú vị là phương pháp này không chỉ gợi ý các quan hệ mới có thể tồn tại trong Wikidata, mà còn có thể truy ngược về văn bản gốc làm căn cứ cho gợi ý đó. Điều này cho phép con người kiểm tra lại để đảm bảo tính chính xác và đáng tin cậy của thông tin.

Modeling Relation Path (Lao and Cohen 2010)

Một khía cạnh thú vị mà tôi muốn thảo luận là mô hình hóa các đường dẫn quan hệ. Đây là một ý tưởng rất hay, cho phép suy luận qua nhiều bước nhảy của các quan hệ dựa trên sự tồn tại của các quan hệ cụ thể. Các đường dẫn nhiều bước có thể cung cấp thông tin hữu ích để chỉ ra

liệu các quan hệ cá nhân có tồn tại hay không. Ví dụ, với một từ cụ thể trong tiêu đề bài báo, hệ thống có thể đề xuất một hội nghị phù hợp để gửi bài báo.

ID	Weight	Feature
1	26.9	$word \xrightarrow{HasTitle^{-1}} paper \xrightarrow{In} journal$
2	4.5	$word \xrightarrow{HasTitle^{-1}} paper \xrightarrow{FirstAuthor} author \xrightarrow{FirstAuthor^{-1}} paper \xrightarrow{In} journal$
3	2.8	$word \xrightarrow{HasTitle^{-1}} paper \xrightarrow{AnyAuthor} author \xrightarrow{AnyAuthor^{-1}} paper \xrightarrow{In} journal$
4	1.1	$gene \xrightarrow{GeneticallyRelated} gene \xrightarrow{HasGene^{-1}} paper \xrightarrow{In} journal$
5	0.9	$gene \xrightarrow{HasGene^{-1}} paper \xrightarrow{In} journal$
6	0.6	$e^* \xrightarrow{AnyPaper} paper \xrightarrow{Cite} paper \xrightarrow{In} journal$

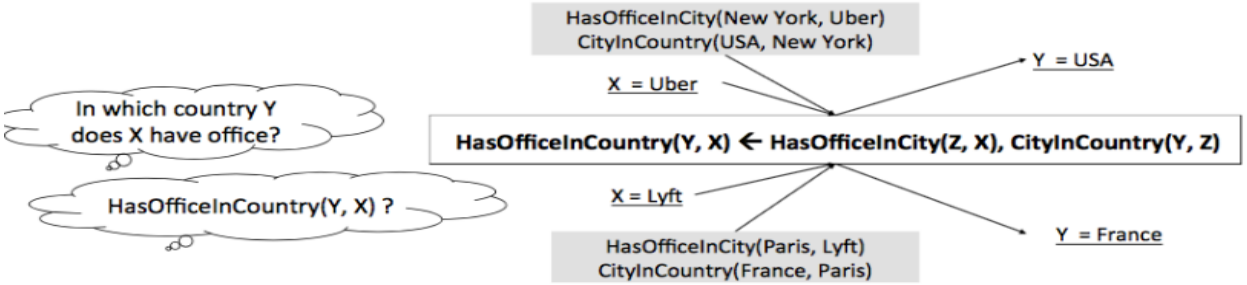
Vấn đề mà họ đang cố gắng giải quyết là: khi có một từ trong tiêu đề bài báo, tìm các bài báo khác có từ đó trong tiêu đề, và những bài báo này nằm trong một tạp chí. Điều này giúp xác định mức độ liên quan của bài báo với tạp chí đó. Ngoài ra, có thể tìm các bài báo với từ đó trong tiêu đề, xác định tác giả đầu tiên của bài báo, tìm bài báo khác mà tác giả đó cũng là tác giả đầu tiên, và sau đó tìm tạp chí của bài báo đó. Họ đã trình bày cách mở rộng các đường dẫn này và đưa chúng vào mô hình dự đoán để dự đoán các quan hệ bổ sung.

Khác với phương pháp chỉ ra rằng các quan hệ đơn lẻ có thể chỉ ra sự tồn tại của một quan hệ cụ thể, bài báo này cho thấy không chỉ các quan hệ đơn lẻ mà cả các đường dẫn quan hệ cũng có thể chỉ ra sự tồn tại của một quan hệ. Điều này làm cho phương pháp trở nên biểu đạt hơn.

Differentiable Logic Rules (Yang et al. 2017)

Một bài báo tiếp theo đã giới thiệu khái niệm "differentiable logic rules", cho phép huấn luyện mô hình theo phương pháp end-to-end. Phương pháp này cho phép xem xét toàn bộ các đường dẫn (paths) trong một framework có thể vi phân.

Ví dụ, nếu ta có các quan hệ "City in Country" và "has office in City", điều này ngụ ý về quan hệ "has office in Country". Mặc dù nhiều người đã học về logic và suy luận từ các luật logic, vấn đề là suy luận từ luật logic thường rất mong manh vì luôn tồn tại các trường hợp ngoại lệ. Do đó, thay vì nói một điều gì đó luôn đúng theo phép suy diễn, trong thực tế, khi có hai mẩu thông tin, một điều gì đó có thể trở nên "rất có khả năng" xảy ra hơn.



Lấy ví dụ, việc một người học tại CMU khiến cho khả năng họ học ngành Computer Science cao hơn nhiều, và khả năng họ học ngành Medicine thấp hơn nhiều. Tuy nhiên, điều này không có nghĩa là hoàn toàn không thể có sinh viên CMU học ngành Medicine - có thể có một số ít người đang theo học chương trình liên kết. Điều này cho thấy rất hiếm khi các luật logic là tuyệt đối và bất biến.

Về mặt kỹ thuật, họ xử lý mỗi đường dẫn như một chuỗi phép nhân ma trận, trong đó có một trọng số quy tắc (rule weight) α. Cuối cùng, điều này cho phép đưa ra dự đoán về việc một luật logic có đúng hay không. $\sum_l \alpha_l \prod_{k \in \beta_l} M_{R_k}$

Phần lớn các nghiên cứu trong lĩnh vực này tập trung vào không gian kiến thức có cấu trúc (structured knowledge) và đồ thị tri thức (knowledge graphs). Hiện tại, chưa có nhiều nghiên cứu áp dụng trực tiếp differentiable logic rules vào các mô hình ngôn ngữ, chủ yếu vì bài toán trở nên phức tạp và khó xử lý hơn nhiều trong bối cảnh này.

Probing Knowledge in LMs

Probing Knowledge in LMs

Chủ đề cuối cùng mà tôi muốn trình bày là việc thăm dò kiến thức trong các mô hình ngôn ngữ. Chúng ta có những cơ sở dữ liệu kiến thức chứa đựng một lượng lớn thông tin, cho phép chúng ta đánh giá mức độ hiểu biết của các mô hình ngôn ngữ về những điều này. Các mô hình truyền thống như QA (hỏi đáp), đọc hiểu, hay RAG thường tham khảo các nguồn tài nguyên bên ngoài để trả lời câu hỏi, chẳng hạn như các bài viết trên Wikipedia. Tuy nhiên, câu hỏi đặt ra là liệu chúng ta có thể trả lời các câu hỏi mà không cần sử dụng RAG hay không, và kiến thức nào đã được mã hóa trong các mô hình ngôn ngữ này.

LMs and KBs? (Petroni et al. 2019)

Một trong những bài báo đầu tiên giải quyết vấn đề sử dụng ngôn ngữ tự nhiên để truy vấn mô hình ngôn ngữ là bài báo có tên "Wama". Bài báo này đã giới thiệu một tài nguyên gọi là "Llama", viết tắt là L M A. Thay vì sử dụng các truy vấn cấu trúc như SQL hay SPARQL để truy vấn cơ sở tri thức (KBS), nhóm nghiên cứu đã thử nghiệm sử dụng các câu lệnh ngôn ngữ tự nhiên để truy vấn mô hình ngôn ngữ.

Cách tiếp cận của họ là sử dụng các câu như "Dante was born in [mask]" và sau đó điền vào chỗ trống bằng một “mask language model”, ví dụ như điền "Florence". Mặc dù hiện nay phương pháp này không còn phổ biến, nhưng vào thời điểm đó, nhóm nghiên cứu đã sử dụng cơ sở tri thức làm chuẩn để kiểm tra xem kiến thức trong cơ sở tri thức có thể được phục hồi từ mô hình ngôn ngữ hay không.

Họ đã đề xuất một tiêu chuẩn đánh giá gọi là "Llama Benchmark", sử dụng các câu lệnh thủ công cho 41 mối quan hệ. Ví dụ, với mẫu câu "X was founded in Y", họ điền vào chủ ngữ và sử

dụng các mô hình ngôn ngữ như BERT để dự đoán tân ngữ, ví dụ như "Bloomberg LP was founded in [mask]". Kết quả cho thấy các mô hình như ELMo, Transformer-XL và BERT base đạt độ chính xác lên đến 31%. Tuy nhiên, với các mô hình ngôn ngữ hiện đại, độ chính xác này chắc chắn sẽ cao hơn nhiều.

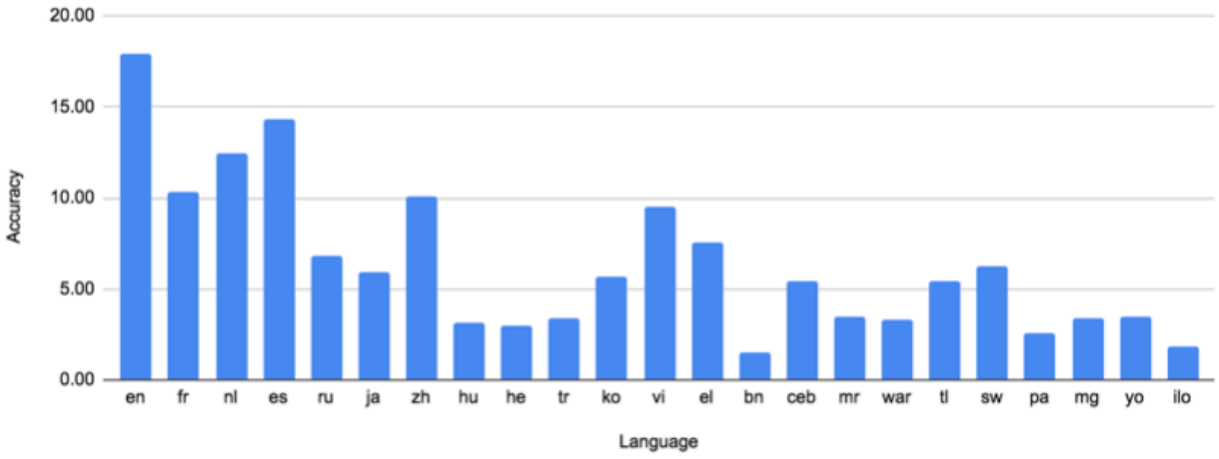
X-FACTR: Multilingual Factual Knowledge Probing (Jiang et al. 2020)

Trong một nghiên cứu tiếp theo, các tác giả đã thử nghiệm khả năng đa ngôn ngữ của các mô hình ngôn ngữ. Điểm thú vị của nghiên cứu này đó, là sự phân đôi giữa kiến thức được mã hóa trong các mô hình ngôn ngữ và khả năng truy xuất kiến thức đó.

Họ tạo các câu truy vấn từ cơ sở tri thức đa ngôn ngữ, cho phép tạo cùng một câu hỏi về cùng một thực thể bằng nhiều ngôn ngữ khác nhau. Nhóm nghiên cứu đã nhờ nhiều người tạo ra các prompt bằng nhiều ngôn ngữ khác nhau. Kết quả cho thấy mô hình hoạt động tốt hơn nhiều với các truy vấn bằng tiếng Anh so với các ngôn ngữ khác, đặc biệt là với các ngôn ngữ ít nguồn lực hoặc khác biệt nhiều so với tiếng Anh.

Trong thí nghiệm, các tác giả có hai cách đánh giá: một là chỉ tính câu trả lời đúng khi nó được đưa ra bằng ngôn ngữ truy vấn, hai là tính câu trả lời đúng nếu nó được đưa ra bằng bất kỳ ngôn ngữ nào, ngay cả khi mô hình không biết tên thực thể trong ngôn ngữ truy vấn. Với tiếng Anh, các mô hình được thử nghiệm có thể trả lời đúng 77% câu truy vấn.

Điều này cho thấy một nghịch lý thú vị: nếu mô hình có thể trả lời bằng tiếng Anh, điều đó chứng tỏ nó "biết" câu trả lời. Tuy nhiên, khi được hỏi chính xác cùng một câu hỏi bằng ngôn ngữ khác, nó lại không thể truy xuất được câu trả lời đó.



Max performance of M-BERT, XLM, XLM-R

Một nghiên cứu gần đây khác cũng chỉ ra rằng việc sử dụng personas (như "I am an old man", "I am an old woman", "I am a young man", "I am a young woman", "I am a child") có thể thay đổi đáng kể khả năng trả lời câu hỏi của mô hình. Điều này cho thấy các mô hình có khả năng trả

lời câu hỏi, nhưng cách chúng trả lời phụ thuộc vào cách tiếp cận - có thể là qua ngôn ngữ khác nhau hoặc qua personas khác nhau.

Một ví dụ thú vị trong lĩnh vực sinh mã là việc sử dụng prompt "I have checked this carefully and all the unit tests pass" có thể cải thiện độ chính xác của việc sinh mã lên khoảng 5 điểm. Điều này cho thấy việc đặt mô hình vào đúng "mood" có thể giúp nó trả lời câu hỏi chính xác hơn.

Close-book T5: Directly Fine-tune with QA Pairs (Roberts et al. 2020)

Một phương pháp khác để cải thiện hiệu suất của các mô hình ngôn ngữ là fine-tune chúng để đặc biệt giỏi trong việc trả lời các câu hỏi dựa trên cơ sở tri thức (knowledge-based questions). Một nghiên cứu đã chứng minh rằng việc fine-tune các mô hình trên tập dữ liệu câu hỏi được tạo tổng hợp về cơ sở tri thức có thể cải thiện đáng kể khả năng trả lời các câu hỏi liên quan đến kiến thức của mô hình. Đây là một phương pháp khá đơn giản nhưng hiệu quả.

Multi-hop Factual Reasoning in LMs (Jiang et al. 2022)

Nghiên cứu cuối cùng tôi muốn đề cập đến là suy luận đa bước (multihop reasoning). Sau khi đã thảo luận về chuỗi suy luận trong cơ sở tri thức, đây là một ví dụ về suy luận đa bước trong tham số của mô hình, nhằm kiểm tra khả năng trả lời các câu hỏi đa bước của mô hình.

Các tác giả bắt đầu bằng việc sử dụng một cơ sở tri thức có các mối quan hệ như "country", "US president", "birthday". Từ đó, họ có thể tạo ra các câu hỏi đa bước bằng cách theo dõi các liên kết quan hệ và biết được câu trả lời thông qua việc đi theo các liên kết này. Ví dụ, từ hai câu hỏi đơn:

1. "Return the artist who recorded Party 8 Over"
2. "Where in Georgia does Usher live?"

Chúng ta có thể tạo một câu hỏi tổng hợp: "In which part of Georgia does the artist that recorded the Party 8 Over live?"

Trong nghiên cứu, các tác giả đã đo lường khả năng của mô hình trong việc trả lời câu hỏi đầu tiên, câu hỏi thứ hai và câu hỏi tổng hợp. Theo lý thuyết, nếu mô hình là một bộ xử lý kiến thức hoàn hảo, ta kỳ vọng rằng nếu nó biết câu trả lời cho cả hai câu hỏi đơn lẻ, nó sẽ trả lời đúng câu hỏi tổng hợp. Ngược lại, nếu nó trả lời sai một trong hai câu hỏi đơn lẻ, nó sẽ trả lời sai câu hỏi tổng hợp.

Tuy nhiên, kết quả thực nghiệm cho thấy điều này hoàn toàn không đúng. Thay vào đó, các tác giả đã phát hiện ra rằng nếu mô hình có thể trả lời đúng câu hỏi thứ hai, nó có khả năng cao hơn nhiều trong việc trả lời đúng câu hỏi tổng hợp, trong khi khả năng trả lời câu hỏi đầu tiên gần như không liên quan đến việc trả lời câu hỏi tổng hợp.

Điều này có thể được giải thích qua ví dụ: Nếu câu hỏi thứ hai là một câu hỏi khó như "Who are all the presidents of the United States?", thì câu hỏi tổng hợp "Who are all the presidents of the country where Washington DC is located in?" cũng sẽ khó trả lời. Ngược lại, với câu hỏi như

"What is the capital of the country where the most people live?", ngay cả khi không chắc chắn về "quốc gia đông dân nhất", mô hình vẫn có thể đoán ngẫu nhiên một thủ đô và đôi khi đoán đúng.

Nghiên cứu này cho thấy cơ sở tri thức là công cụ hữu ích để đặt những câu hỏi thú vị về những gì mô hình ngôn ngữ biết hoặc không biết một cách có cấu trúc, đặc biệt hữu ích cho việc đánh giá khả năng suy luận và logic của các mô hình này.

Resources

1. https://phontron.com/class/anlp2024/lectures/#knowledge-based-qa-march-21
2. Required Reading: Relation Extraction Jurafsky and Martin Chapter 17.2
3. Reference: Relation Extraction Survey (Nickel et al. 2016)
4. Reference: WordNet (Miller 1995)
5. Reference: Cyc (Lenant 1995)
6. Reference: DBPedia (Auer et al. 2007)
7. Reference: YAGO (Suchanek et al. 2007)
8. Reference: Babelnet (Navigli and Ponzetto 2010)
9. Reference: Freebase (Bollacker et al. 2008)
10. Reference: Wikidata (Vrandečić and Krötzsch 2014)
11. Reference: Relation Extraction by Translating Embeddings (Bordes et al. 2013)
12. Reference: Relation Extraction with Neural Tensor Networks (Socher et al. 2013)
13. Reference: Relation Extraction by Translating on Hyperplanes (Wang et al. 2014)
14. Reference: Relation Extraction by Representing Entities and Relations (Lin et al. 2015)
15. Reference: Relation Extraction w/ Decomposed Matrices (Xie et al. 2017)
16. Reference: Distant Supervision for Relation Extraction (Mintz et al. 2009)
17. Reference: Relation Classification w/ Recursive NNs (Socher et al. 2012)
18. Reference: Relation Classification w/ CNNs (Zeng et al. 2014)
19. Reference: Open IE from the Web (Banko et al. 2007)
20. Reference: ReVerb Open IE (Fader et al. 2011)
21. Reference: Supervised Open IE (Stanovsky et al. 2018)
22. Reference: Universal Schema (Riedel et al. 2013)
23. Reference: Joint Entity and Relation Embedding (Toutanova et al. 2015)
24. Reference: Distant Supervision for Neural Models (Luo et al. 2017)
25. Reference: Relation Extraction w/ Tensor Decomposition (Sutskever et al. 2009)
26. Reference: Relation Extraction via. KG Paths (Lao and Cohen 2010)
27. Reference: Relation Extraction by Traversing Knowledge Graphs (Guu et al. 2015)
28. Reference: Relation Extraction via Differentiable Logic Rules (Yang et al. 2017)
29. Reference: Improving Embeddings w/ Semantic Knowledge (Yu et al. 2014)
30. Reference: Improving Embeddings w/ Semantic Knowledge (Yu et al. 2014)
31. Reference: Retrofitting Word Vectors to Semantic Lexicons (Faruqui et al. 2015)
32. Reference: Multi-sense Embedding with Semantic Lexicons (Jauhar et al. 2015)
33. Reference: Antonymy and Synonym Constraints for Word Embedding (Mrksic et al. 2016)

34. Reference: Language Models as Knowledge Bases? (Petroni et al. 2019)
35. Reference: How Can We Know What Language Models Know? (Jiang et al. 2019)
36. Reference: AUTOPROMPT: Eliciting Knowledge from Language Models with Automatically Generated Prompts (Shin et al. 2020)
37. Reference: GPT Understands, Too (Liu et al. 2021)
38. Reference: How Much Knowledge Can You Pack Into the Parameters of a Language Model? (Roberts et al. 2020)
39. Reference: X-FACTR: Multilingual Factual Knowledge Retrieval from Pretrained Language Models (Jiang et al. 2020)
40. Reference: REALM: Retrieval-Augmented Language Model Pre-Training (Guu et al. 2020)
41. Reference: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks (Lewis et al. 2020)
42. Reference: Multi-hop Reasoning in LMs (Jiang et al. 2022)