Lecture 1: Giới thiệu về Xử lý ngôn ngữ tự nhiên

Ngày 22 tháng 9 năm 2024.

What is NLP?

NLP (Xử lý ngôn ngữ tự nhiên) là công nghệ sử dụng máy tính để xử lý và hiểu ngôn ngữ của con người, chủ yếu là văn bản.

NLP hỗ trợ giao tiếp giữa con người và máy (chẳng hạn như trả lời câu hỏi) cũng như giữa con người với nhau (ví dụ như dịch thuật máy, tạo đoạn mã).
Nó cũng giúp phân tích và hiểu ngôn ngữ thông qua các kỹ thuật như phân tích cú pháp, phân loại văn bản và nhận diện/liên kết thực thể.
Chúng ta thường sử dụng NLP hàng ngày mà đôi khi không nhận ra điều đó!

NLP can Answer Questions

Ví dụ, hỏi ChatGPT "Who is the current president of Carnegie Mellon University?" và ChatGPT đã trả lời rằng "I did a quick search for more information here is what I found: the current president of Carnegie Mellon University is Farnam Jahanian. He has been serving since July 1." Theo như tôi biết, thông tin này là chính xác.

NLP cannot Answer Questions

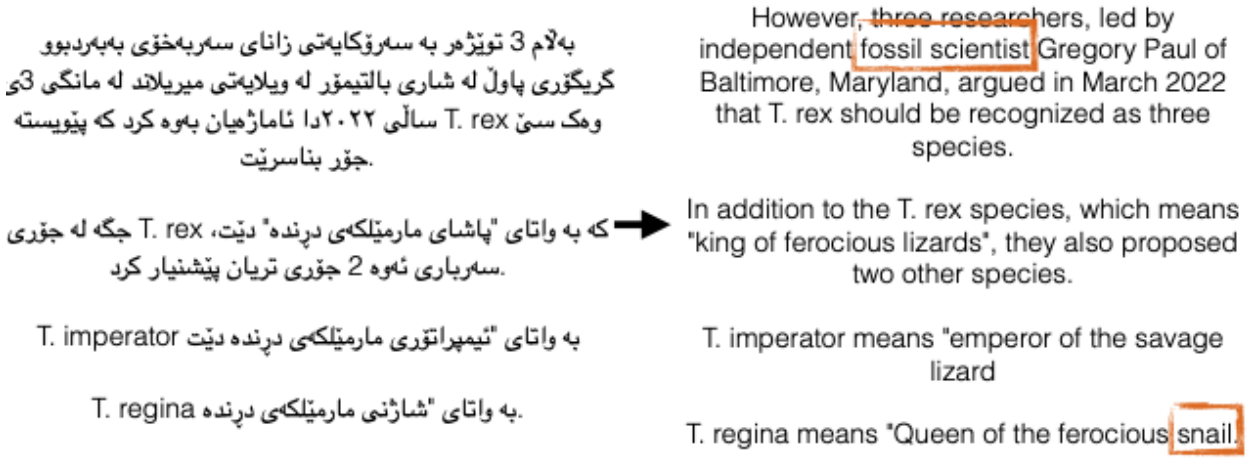
Ví dụ, hỏi về số lượng lớp trong kiến trúc GPT 3.5 turbo và nó đã trả lời rằng "GPT 3.5 turbo, một phiên bản tối ưu hóa của GPT 3.5 để có phản hồi nhanh hơn, không có cấu trúc lớp cụ thể như các mô hình GPT-3 truyền thống."
Tôi không biết điều này có đúng hay không, nhưng tôi khá chắc chắn là không đúng. Tôi nghĩ rằng GPT là một mô hình tương tự như các mô hình khác, vì vậy nó có thể đã tạo ra thông số kỹ thuật vì không có thông tin nào trên Internet hoặc không thể nói về nó.

NLP can Translate Text

NLP có khả năng dịch văn bản khá tốt. Ví dụ thử nghiệm với Google Translate trên một ví dụ tiếng Nhật. Nó cho kết quả khá ổn. Mặc dù nó không hoàn hảo, nhưng bạn vẫn có thể hiểu và nắm bắt được ý chính.

NLP cannot Translate Text

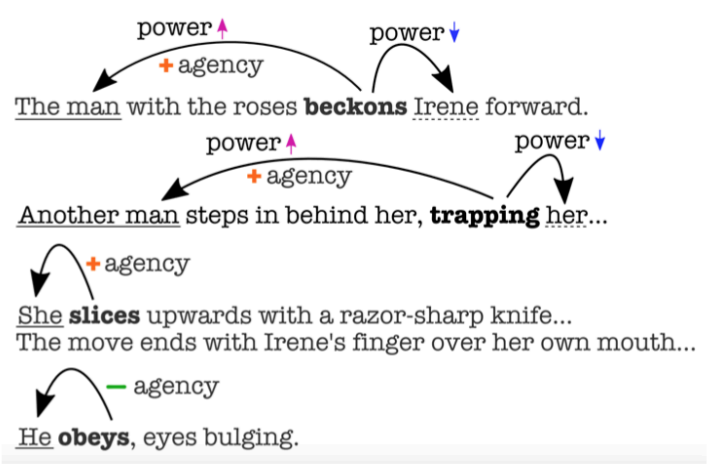
NLP không thể dịch văn bản một cách hoàn hảo, đặc biệt là với các ngôn ngữ tài nguyên thấp như tiếng Kurdish. Khi thử hiểu một đoạn văn về một phát hiện trong ngành cổ sinh vật học, thuật ngữ "fossil scientist" được sử dụng thay vì từ tiếng Anh rõ ràng hơn là "paleontologist". Đoạn văn này đề cập đến ba loài T-Rex khác nhau, trong đó "T-Rex" được gọi là "king of ferocious lizards", "emperator" là "emperor of Savaged lizards", và "T Regina" có nghĩa là "clean of ferocious snail", nhưng thực tế có thể là "lizard". Điều này cho thấy rằng khả năng dịch thuật vẫn còn nhiều hạn chế.



Front page news from Voice of America Kurdish, translated by Google Jan 5, 2024

Language Analysis Can Aid Scientific Inquiry

Phân tích ngôn ngữ có thể hỗ trợ nghiên cứu khoa học, ví dụ như trong khoa học xã hội tính toán, nhằm trả lời các câu hỏi về xã hội dựa trên dữ liệu quan sát. Một câu hỏi điển hình là: "Do movie scripts portray female or male characters with more power or agency?" (Các kịch bản phim có miêu tả nhân vật nữ hay nam với nhiều quyền lực hoặc quyền tự quyết hơn không?).



Frame	β	gender
$agency(\text{AG})=+$	-0.951	M**
$power(\text{AG}>\text{TH})$	-0.468	M**
$agency(\text{AG})=-$	0.277	F**
$power(\text{AG}<\text{TH})$	*not sig.*	

Sap et al. "Connotation Frames of Power and Agency in Modern Films" EMNLP 2017.

Nghiên cứu này đã sử dụng công nghệ NLP để phân tích văn bản, xác định các tác nhân và đối tượng trong kịch bản. Kết quả cho thấy các nhân vật nam thường được trao nhiều quyền lực và quyền tự quyết hơn so với các nhân vật nữ. Cụ thể, trong các ví dụ như "The man with the roses beckons Irene forward" (Người đàn ông với những bông hoa ra hiệu cho Irene tiến lên) và "Another man steps in behind her, trapping her" (Một người đàn ông khác bước vào phía sau cô, giam giữ cô), có thể thấy rõ sự chênh lệch về quyền lực và quyền tự quyết giữa các giới.

NLP đã giúp phân tích và trích xuất dữ liệu hữu ích, biến nó thành dạng dễ dàng để thực hiện các phân tích sâu hơn về sự phân bổ quyền lực và quyền tự quyết trong các kịch bản phim hiện đại.

Language Analysis Fails at Basic Tasks

Phân tích ngôn ngữ gặp khó khăn với những nhiệm vụ cơ bản. Trong bài viết đầu tiên trên New York Times ngày 29 tháng 8 năm 2021, được nhận diện bởi Stanford CoreNLP, tôi đã thử nghiệm một số công cụ phân tích ngôn ngữ. Những công cụ này, như Stanford CoreNLP và SpaCy, thường được nhiều người sử dụng. Tôi đã đưa vào một câu từ bài viết trên New York Times và phát hiện rằng chúng mắc ít nhất một lỗi trong câu đầu tiên. Cụ thể, công cụ nhận diện thực thể đã nhận diện "Baton Rouge" là một tổ chức và "hurricane EA" cũng được nhận diện là một tổ chức. Điều này cho thấy ngay cả những công cụ mà chúng ta kỳ vọng sẽ hoạt động tốt cũng có thể mắc những sai lầm khá nghiêm trọng. Trong lớp học này, tôi muốn đề cập đến những vấn đề này và những gì cần được cải thiện.

In this Class, we Ask:

Trong lớp học này, chúng ta sẽ tìm hiểu về những yếu tố cấu thành nên các hệ thống NLP (Xử lý ngôn ngữ tự nhiên) hiện đại, có khả năng hoạt động cực kỳ hiệu quả trong một số nhiệm vụ nhất định.
Chúng ta cũng sẽ bàn luận về những điểm còn hạn chế của các hệ thống NLP hiện tại, nguyên nhân của những thất bại đó.
Cuối cùng, chúng ta sẽ đề xuất cách cải tiến phù hợp nhằm đạt được các mục tiêu trong lĩnh vực NLP.

NLP System Building Overview

A General Framework for NLP Systems

Một khung tổng quát cho các hệ thống NLP (Xử lý ngôn ngữ tự nhiên) có thể được mô tả bằng cách xây dựng một hàm để ánh xạ một đầu vào X thành một đầu ra Y, trong đó X và/hoặc Y đều liên quan đến ngôn ngữ.

Đầu vào X	Đầu ra Y	Nhiệm vụ
Text	Continuing Text	Mô hình ngôn ngữ
Text	Text ở ngôn ngữ khác	Dịch thuật
Text	Label (Nhãn)	Phân loại text
Text	Cấu trúc ngôn ngữ	Phân tích ngôn ngữ
Ảnh	Text	Chú thích hình ảnh

Methods for Creating NLP Systems

Có nhiều cách để tạo ra hệ thống NLP, và tất cả đều đang được áp dụng vào năm 2024. Phương pháp đầu tiên là tạo hệ thống dựa trên quy tắc. Ví dụ, nếu bạn muốn xây dựng một bộ phân loại văn bản, bạn có thể viết một hàm Python đơn giản để phân loại văn bản thành "sports" (thể thao) hoặc "other" (khác). Hàm này sẽ kiểm tra xem các từ khóa như "baseball", "soccer", "football" và "tennis" có xuất hiện trong văn bản hay không. Nếu có, nó sẽ phân loại là "sports", nếu không thì là "other".

Hệ thống dựa trên quy tắc rất tiện lợi khi bạn không quá quan tâm đến độ chính xác của hệ thống hoặc khi bạn đang làm những tác vụ rất đơn giản. Chúng có thể hoạt động tốt ngay cả khi bạn chỉ thực hiện những điều rất cơ bản.

Phương pháp thứ hai, đã trở thành một trong những phương pháp chính trong NLP trong khoảng ba năm qua, là prompting. Trong phương pháp này, bạn hỏi một mô hình ngôn ngữ rằng nếu câu sau đây liên quan đến thể thao, hãy trả lời "sports", nếu không hãy trả lời "other". Bạn sẽ cung cấp câu hỏi này cho mô hình ngôn ngữ yêu thích của mình, thường là một mô hình như GPT, và nó sẽ đưa ra câu trả lời.

Cuối cùng là phương pháp fine-tuning, nơi bạn sử dụng dữ liệu đã được ghép cặp để thực hiện học máy. Bạn có thể có các câu như "I love to play baseball.", "The stock price is going up.", "He got a hat-trick yesterday.", và "He is wearing tennis shoes." và gán nhãn cho chúng. Sau đó, bạn sẽ huấn luyện một mô hình từ những dữ liệu này. Bạn cũng có thể bắt đầu với một mô hình dựa trên prompting và sau đó fine-tune một mô hình ngôn ngữ.

Data Requirements for System Building

Yêu cầu dữ liệu cho việc xây dựng hệ thống

Khi xây dựng hệ thống, có nhiều cấp độ yêu cầu dữ liệu khác nhau:

1. Quy tắc/prompting dựa trên trực giác:
Không cần dữ liệu, nhưng cũng không có đảm bảo về hiệu suất. Bạn có thể bắt đầu viết quy tắc mà không cần bất kỳ ví dụ nào. Tuy nhiên, điều này có thể dẫn đến việc bạn không biết hệ thống hoạt động tốt đến đâu.

2. Quy tắc/prompting dựa trên kiểm tra ngẫu nhiên (spot-checks):
Cần một lượng nhỏ dữ liệu với đầu vào X. Bạn có thể bắt đầu với một hệ thống dựa trên quy tắc hoặc prompting, sau đó chạy nó trên một số dữ liệu mà bạn quan tâm. Bạn sẽ xem xét chất lượng dữ liệu và điều chỉnh quy tắc hoặc prompting nếu cần.

3. Quy tắc/prompting với đánh giá nghiêm ngặt:
Cần một tập phát triển với đầu vào X và đầu ra Y (ví dụ: từ 200 đến 2000 ví dụ). Bạn cần một chỉ số đánh giá để đo lường độ chính xác của hệ thống. Đây là cấp độ khó khăn tiếp theo mà các kỹ sư NLP nghiêm túc thường phải thực hiện.

4. Tinh chỉnh (Fine-tuning):
Cần một tập huấn luyện bổ sung, thường lớn hơn 200 đến 2000 ví dụ. Quy tắc chung là mỗi khi bạn gấp đôi kích thước tập huấn luyện, độ chính xác sẽ tăng lên một cách ổn định. Nếu bạn bắt đầu với độ chính xác không có dữ liệu (zero-shot accuracy) và tạo ra một tập huấn luyện nhỏ, bạn sẽ thấy sự cải thiện lớn. Tuy nhiên, sau đó, sự cải thiện sẽ giảm dần và bạn sẽ phải dành nhiều thời gian cho việc chú thích dữ liệu.

Tóm lại, có một sự thay đổi đáng kể trong thực tiễn xây dựng hệ thống. Trước đây, việc tinh chỉnh là điều cần thiết, nhưng hiện nay, nó trở nên tùy chọn hơn.

Let's Try to Make a Rule-based NLP System!

Example Task: Review Sentiment Analysis

Khi nhận được một đánh giá trên một trang web đánh giá (X), hãy quyết định xem nhãn (Y) của nó là tích cực (1), tiêu cực (-1) hay trung lập (0).

Tôi muốn thử tạo một hệ thống dựa trên quy tắc cho phân tích cảm xúc. Đây là một ý tưởng không tốt và tôi không khuyến khích bạn làm điều này trong thực tế, nhưng tôi muốn làm điều này ở đây để cho bạn thấy tại sao nó lại là một ý tưởng tồi và những vấn đề khó khăn nào bạn sẽ gặp phải khi cố gắng tạo ra một hệ thống dựa trên quy tắc. Sau khi hoàn thành, chúng ta sẽ chuyển sang xây dựng một hệ thống dựa trên máy học.

Phân tích cảm xúc của đánh giá là một trong những nhiệm vụ giá trị nhất trong NLP hiện nay, vì nó cho phép mọi người biết khách hàng đang nghĩ gì về sản phẩm, cải thiện phát triển sản phẩm và theo dõi sự hài lòng của người dùng với dịch vụ truyền thông xã hội. Cách thức hoạt động là bạn có các câu đầu vào như "I hate this movie", "I love this movie", "I saw this movie" và chúng sẽ được ánh xạ thành tích cực, trung lập hoặc tiêu cực. Cụ thể, "I hate this movie" sẽ là tiêu cực, "I love this movie" là tích cực và "I saw this movie" là trung lập.

A Three-step Process for Making Predictions

Quá trình ba bước để đưa ra dự đoán bao gồm:

1. Trích xuất đặc trưng: Chúng ta sẽ trích xuất các đặc trưng nổi bật từ văn bản để đưa ra quyết định về đầu ra tiếp theo.
2. Tính toán điểm số: Tính toán một điểm số cho một hoặc nhiều khả năng.
3. Hàm quyết định: Lựa chọn một trong những khả năng đó.

Formally

1. Trích xuất đặc trưng: $h = f(x)$
2. Tính toán điểm số:
   Phân loại nhị phân $s = w \cdot h$

Phân loại đa lớp: s = Wh

3. Quyết định: $\widehat{y} = decide(s)$

Trong quá trình trích xuất đặc trưng, chúng ta có một hàm để trích xuất một vector đặc trưng cho việc tính toán điểm số. Điểm số được tính dựa trên phân loại nhị phân, nơi chúng ta có một vector trọng số và thực hiện phép nhân điểm với vector đặc trưng, hoặc trong trường hợp phân loại đa lớp, chúng ta có một ma trận trọng số và thực hiện phép nhân với vector, từ đó cho ra điểm số cho nhiều lớp khác nhau.

Sau đó, chúng ta áp dụng một quy tắc quyết định để xác định đầu ra. Một quy tắc quyết định điển hình là sử dụng ngưỡng. Đối với phân loại nhị phân, nếu điểm số vượt qua ngưỡng, chúng ta sẽ trả lời "có", còn nếu dưới ngưỡng, chúng ta sẽ trả lời "không". Một tùy chọn khác là có thể có một ngưỡng và trả lời "có", "không" hoặc "không đưa ra câu trả lời", tùy thuộc vào cách đánh giá một bộ phân loại tốt.

Đối với phân loại đa lớp, quy tắc quyết định tiêu chuẩn là argmax, tức là tìm chỉ số có điểm số cao nhất và xuất ra nó. Chúng ta sẽ thảo luận về các quy tắc quyết định khác như tự nhất quán và rủi ro tối thiểu sau này, đặc biệt trong bối cảnh tạo văn bản.

Now Let's Improve!

Làm thế nào để cải thiện 1 hệ thống Xử lý ngôn ngữ Tự nhiên NLP?

1. Xác định vấn đề:
Đầu tiên, chúng ta cần tìm hiểu những vấn đề gì đang xảy ra với hệ thống thông qua việc phân tích lỗi. Điều này giúp xác định các điểm yếu trong dự đoán của mô hình.

2. Chỉnh sửa hệ thống:
Sau khi xác định các vấn đề, bước tiếp theo là thực hiện các thay đổi cần thiết cho hệ thống. Điều này có thể bao gồm việc cải thiện quy trình trích xuất đặc trưng, điều chỉnh hàm tính toán điểm số hoặc các yếu tố khác trong mô hình.

3. Đo lường sự cải thiện về độ chính xác:
Sau khi thực hiện các chỉnh sửa, chúng ta cần đánh giá xem những thay đổi này có mang lại sự cải thiện về độ chính xác hay không. Dựa trên kết quả này, chúng ta sẽ quyết định chấp nhận hoặc từ chối các thay đổi đã thực hiện.

4. Lặp lại:
Quy trình này được lặp lại từ bước 1, nghĩa là tiếp tục xác định các vấn đề mới, thực hiện các thay đổi và kiểm tra kết quả cho đến khi đạt được kết quả như mong muốn.

5. Đánh giá cuối cùng:

Cuối cùng, khi đã hài lòng với độ chính xác trên tập dữ liệu phát triển (dev), chúng ta sẽ tiến hành đánh giá mô hình trên tập dữ liệu kiểm tra (test) để có cái nhìn khách quan về hiệu suất của hệ thống.

Quy trình này là một vòng lặp cải tiến liên tục, giúp tối ưu hóa mô hình trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP).

Some Difficult Cases

Low-frequency Words: Từ Ngữ Tần Suất Thấp

Ví dụ 1:
The action switches between past and present, but the material link is too **tenuous** to anchor the emotional connections that **purport** to span a 125-year divide.
Dịch nghĩa: Hành động chuyển đổi giữa quá khứ và hiện tại, nhưng liên kết vật chất quá yếu để gắn kết những cảm xúc mà nó cho rằng trải dài qua 125 năm.
Đây là một câu có ý nghĩa tiêu cực, với các từ như "tenuous" và "purport" mang sắc thái tiêu cực nhưng không xuất hiện thường xuyên. Việc tìm kiếm những từ này có thể tốn nhiều thời gian.

Ví dụ 2:
Here’s yet another studio horror franchise **mucking** up its storyline with **glitches** casual fans could correct in their sleep."
Đây cũng là một trường hợp tiêu cực.

Giải pháp cho vấn đề này có thể là tiếp tục làm việc cho đến khi chúng ta tìm ra tất cả các từ, mặc dù điều này có thể không thú vị. Một lựa chọn khác là tích hợp các nguồn tài nguyên bên ngoài như sentiment dictionaries mà mọi người đã tạo ra. Tuy nhiên, việc này đòi hỏi nhiều công sức kỹ thuật để triển khai.

Conjugation - Sự kết hợp từ

Trong lĩnh vực xử lý ngôn ngữ tự nhiên, việc phân tích hình thái và phân tích từ là rất quan trọng. Ví dụ, một đoạn phim operatic, sprawling picture được mô tả là "**entertainingly** acted, **magnificently** shot and gripping enough to sustain most of its 170-minute length" cho thấy sự tích cực trong cách diễn xuất và hình ảnh. Ngược lại, một nhận xét khác cho rằng "It’s basically an **overlong** episode of Tales from the Crypt" lại mang tính tiêu cực.

Để xử lý những ví dụ này, chúng ta có thể sử dụng các công cụ như stemmer hoặc các phương pháp chuẩn hóa khác để chuyển đổi các từ về dạng gốc của chúng. Điều này yêu cầu phân tích hình thái hoặc phân tích từ, nhằm giúp chúng ta nhận diện và xử lý các từ đã thấy trong dữ liệu của mình.

Negation - Phân tích Phủ định

Phủ định là một vấn đề phức tạp. Ví dụ, trong câu "This one is not nearly as dreadful as expected", từ "dreadful" là một từ tiêu cực, nhưng khi có phủ định "not nearly as dreadful", nghĩa của câu trở nên trung tính hoặc thậm chí tích cực. Câu này có thể được coi là trung tính, không phải tiêu cực.

Một ví dụ khác là "Serving Sara doesn't serve up a whole lot of laughs". Từ "laughs" rõ ràng mang nghĩa tích cực, nhưng với phủ định "doesn't serve up", nghĩa của câu trở nên tiêu cực.

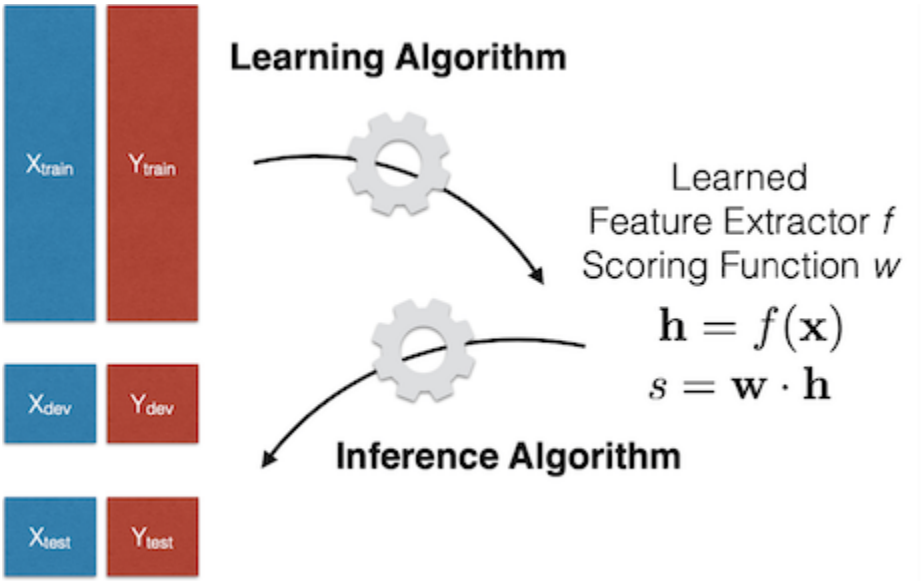
Giải pháp? Nếu một phủ định điều chỉnh một từ, hãy bỏ qua nó. Lưu ý: Có thể cần thực hiện phân tích cú pháp để làm rõ hơn.

Metaphor, Analogy - Phân Tích Ẩn Dụ và So Sánh

Phân tích ngữ nghĩa của một ẩn dụ hay phép so sánh, như "puts a human face on a land most westerners are unfamiliar with", điều này mang tính tích cực. Câu "Green might want to hang on to that ski mask as robbery may be the only way to pay for his next project" cho thấy rằng bộ phim quá tệ đến nỗi đạo diễn sẽ phải cướp tiền để có kinh phí cho dự án tiếp theo, điều này khá tiêu cực. Câu "has all the depth of a wading pool" là một trong những câu tôi thích nhất vì nó ngắn gọn, nhưng bạn cần biết độ sâu của một cái bể nông là như thế nào, và điều này cũng mang tính tiêu cực. Về giải pháp ở đây, tôi thực sự không biết cách xử lý điều này bằng một hệ thống dựa trên quy tắc, tôi không biết chúng ta có thể làm điều này như thế nào. Các mô hình dựa trên máy học dường như khá thích ứng.

Machine Learning-based NLP

Machine Learning



Bức ảnh minh họa là sự tổng hợp của các khái niệm cơ bản trong học máy (Machine Learning). Chúng ta sẽ đi qua từng phần của bức ảnh này để hiểu rõ hơn:

1. Dữ liệu: thường được chia làm 3 tập:
- Dữ liệu huấn luyện (Training Data):
  - X_train là tập các đặc trưng (features), ví dụ như chiều cao, cân nặng (nếu chúng ta dự đoán sức khỏe) hay số lượng phòng, diện tích (nếu dự đoán giá nhà).
  - Y_train là các nhãn (labels) tương ứng, đây là kết quả mà mô hình cần dự đoán. Ví dụ, bệnh trạng (khoẻ mạnh/bệnh) hay giá nhà.

- Dữ liệu phát triển (Development Data, hay Dev set):
  - X_dev và Y_dev được dùng để kiểm tra và tinh chỉnh mô hình trong quá trình phát triển.

- Dữ liệu kiểm tra (Test Data):
  - X_test và Y_test được sử dụng để đánh giá khả năng dự đoán của mô hình sau khi đã hoàn thành quá trình huấn luyện và tinh chỉnh.

2. Quá trình học máy:
- Thuật toán học (Learning Algorithm):
  - Mục tiêu của thuật toán này là học từ dữ liệu huấn luyện (X_train, Y_train) để tạo ra một mô hình có thể dự đoán chính xác.

- Extractor Feature (Trích xuất đặc trưng):
  - Hàm f học được từ X_train để biến đổi các đặc trưng đầu vào (X) thành một biểu diễn đặc trưng mới (h ~ embedding vector).
  - Biểu diễn này lưu trữ các thông tin quan trọng để dự đoán nhãn Y.

- Scoring Function (Hàm tính điểm):
  - Hàm w kết hợp với đặc trưng h để tính điểm (s), đại diện cho xác suất hoặc giá trị dự đoán rằng đầu vào thuộc về nhãn tương ứng.

3. Quá trình Dự báo (Inference Algorithm)
Sau khi mô hình đã học xong từ dữ liệu huấn luyện, nó được dùng để dự đoán với các dữ liệu mới. Quá trình dự báo xảy ra như sau:

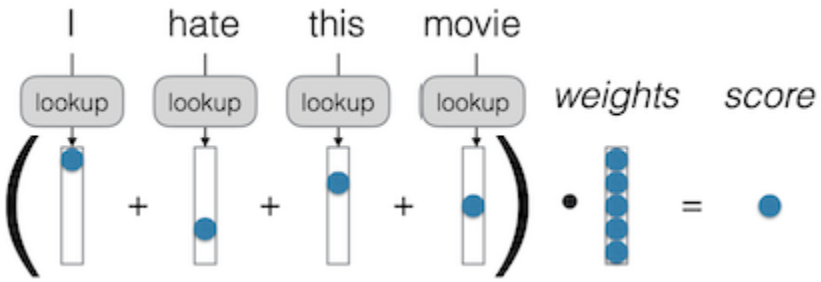
- Đầu vào (x): Lấy một mẫu dữ liệu mới cần dự đoán.
- Biểu diễn Đặc trưng h: Sử dụng hàm f đã học để chuyển đổi x thành h: $h = f(x)$
- Dự đoán s: Sử dụng hàm w đã học để tính điểm dự đoán: $s = w \cdot h$
Giá trị s này có thể được chuyển đổi sang dạng xác suất hoặc nhãn dự đoán cuối cùng tùy thuộc vào bài toán cụ thể (phân loại hay hồi quy).

Tóm tắt:

Quá trình học máy bắt đầu với việc sử dụng dữ liệu huấn luyện để học ra các hàm f và w. Sau đó, mô hình dùng các hàm này để chuyển đổi và tính toán dự đoán cho dữ liệu mới thông qua quá trình dự báo.

A First Attempt: Bag of Words (BOW)

Mô hình Bag of Words (BOW) sử dụng các đặc trưng (features) dựa trên danh tính của các từ trong văn bản, với trọng số (weights) được học từ dữ liệu.



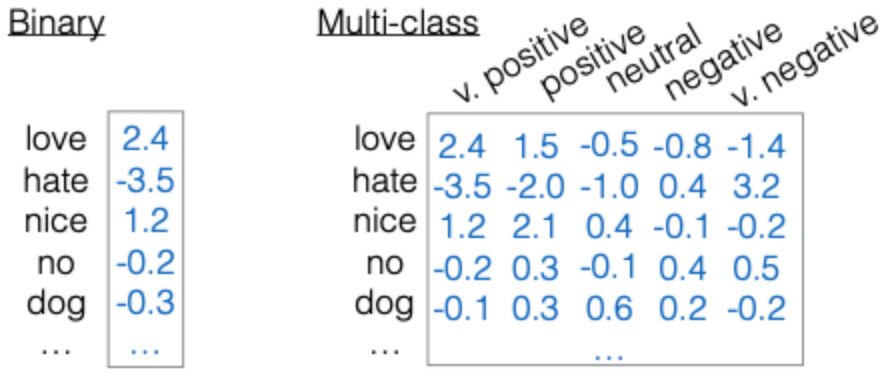
Mô hình Bag of Words hoạt động bằng cách tạo ra một vector one hot, có nghĩa là trong đó một phần tử có giá trị 1 và tất cả các phần tử khác có giá trị 0. Nếu từ khác nhau, vị trí trong vector sẽ khác nhau. Chúng ta cộng tất cả lại để có một vector, trong đó mỗi phần tử là tần suất của từ đó, sau đó nhân với trọng số để có được điểm số.

Mặc dù đây không phải là một bộ trích xuất đặc trưng đã học, nhưng trọng số thì được học. Các vấn đề như từ hiếm, biến thể, ngôn ngữ khác nhau, cú pháp và phép ẩn dụ có thể được cải thiện bằng cách tiếp cận này. Ví dụ, với việc "lookup", nếu có đủ dữ liệu huấn luyện, mô hình có thể cải thiện việc nhận diện các từ hiếm và các biến thể của từ.

Tuy nhiên, không có vấn đề nào được giải quyết hoàn toàn, nhưng nhiều vấn đề có thể được cải thiện. Nếu có đủ dữ liệu huấn luyện cho ngôn ngữ cụ thể, mô hình sẽ hoạt động tốt hơn.

What Do Our Vectors Represent?

Các vector của chúng ta đại diện cho điều gì?
- Phân loại nhị phân: Mỗi từ có một giá trị duy nhất, giá trị dương chỉ ra "yes" và giá trị âm chỉ ra "no".
- Phân loại đa lớp: Mỗi từ có 5 yếu tố tương ứng với [very good, good, neutral, bad, very bad].



Trong phần trình bày, tôi muốn xem xét các vector của chúng ta đại diện cho điều gì. Cụ thể, trong phân loại nhị phân, mỗi từ sẽ có một vector trọng số, và vector trọng số này sẽ dương nếu từ đó có xu hướng tích cực. Trong trường hợp phân loại đa lớp, chúng ta sẽ có một ma trận, trong đó mỗi cột hoặc hàng tương ứng với một từ, và mỗi hàng hoặc cột tương ứng với một nhãn. Giá trị trong ma trận sẽ cao hơn nếu hàng đó có xu hướng tương quan với từ đó.

Simple Training of BOW Models

Chúng ta có thể đào tạo mô hình Bag of Words (BOW) một cách đơn giản bằng cách sử dụng một thuật toán gọi là “structured perceptron”.
feature_weights = {}

```
for x, y in data:
      # Make a prediction
      features = extract_features(x)
      predicted_y = run_classifier(features)
      # Update the weights if the prediction is wrong
      if predicted_y != y:
            for feature in features:
                  feature_weights[feature] = (
                              feature_weights.get(feature, 0) +
                              y * features[feature]
                        )
```

Đầu tiên, chúng ta khởi tạo các trọng số đặc trưng (feature weights) và cho mỗi ví dụ trong tập dữ liệu, chúng ta sẽ trích xuất các đặc trưng. Cách trích xuất đặc trưng là chia tách các từ bằng hàm `split` trong Python và đếm số lần mỗi từ xuất hiện.

Sau đó, chúng ta chạy bộ phân loại (classifier) với các vector đặc trưng. Việc chạy bộ phân loại này tương tự như những gì chúng ta đã làm với hệ thống dựa trên quy tắc, chỉ khác là bây giờ chúng ta sử dụng các vector đặc trưng. Nếu giá trị dự đoán không đúng, chúng ta sẽ cập nhật trọng số cho từng đặc trưng trong không gian đặc trưng. Cụ thể, nếu giá trị `Y` là dương, chúng ta sẽ tăng trọng số theo kích thước vector, và nếu `Y` là âm, chúng ta sẽ giảm trọng số.

Đây là một thuật toán rất đơn giản cho việc đào tạo mô hình BOW. Ví dụ, một bộ phân loại BOW đã được đào tạo có thể được kiểm tra trên cùng một tập dữ liệu mà chúng ta đã sử dụng trước đó. Chúng ta sẽ đào tạo trên tập huấn luyện và đánh giá trên tập phát triển (dev set). Một điểm quan trọng là chúng ta cần xáo trộn thứ tự của các ID dữ liệu, điều này rất quan trọng khi sử dụng thuật toán gia tăng (incremental algorithm). Nếu tập dữ liệu được sắp xếp sao cho tất cả các nhãn dương nằm ở trên cùng và các nhãn âm ở dưới cùng, mô hình có thể chỉ thấy các nhãn âm ở cuối quá trình đào tạo và dẫn đến việc chỉ dự đoán nhãn âm.

Chúng ta sẽ chạy bộ phân loại qua năm lần lặp (epochs) và tính toán độ chính xác. Kết quả cho thấy mô hình đạt được 75% độ chính xác trên tập huấn luyện và 56% trên tập dev. So với bộ phân loại dựa trên quy tắc có độ chính xác 42%, mô hình dựa trên đào tạo đã cải thiện lên 56%, nhưng có dấu hiệu quá khớp (overfitting) với tập huấn luyện.

Điều này cho thấy rõ ràng lý do tại sao chúng ta nên sử dụng machine learning. Mặc dù mã cho mô hình machine learning này không sử dụng bất kỳ thư viện bên ngoài nào, nhưng chúng ta vẫn đạt được kết quả tốt hơn.

What’s Missing in BOW?

BOW vẫn còn nhiều hạn chế trong việc xử lý ngữ nghĩa và cấu trúc câu. Một số vấn đề chính bao gồm:

1. Xử lý từ liên hợp hoặc từ ghép:
Ví dụ, "I love this movie" có thể trở thành "I loved this movie". Việc nhận diện và xử lý các dạng từ khác nhau vẫn chưa hoàn hảo.

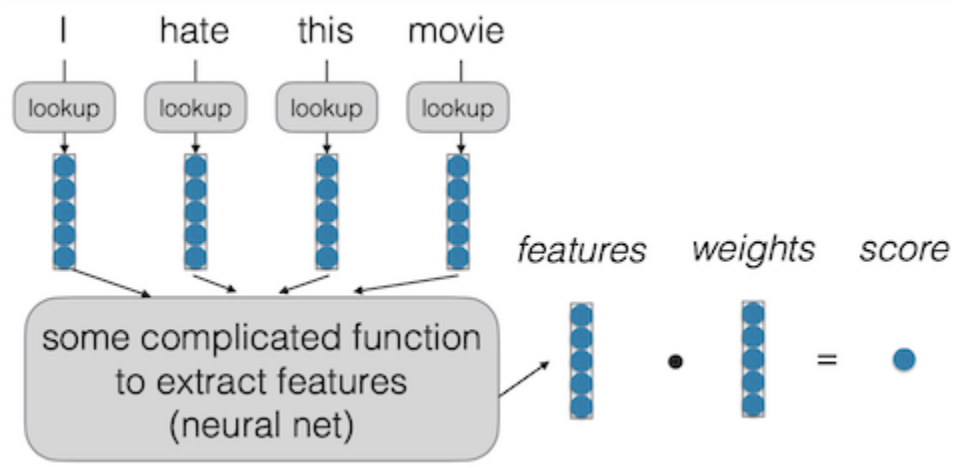
2. Xử lý sự tương đồng giữa các từ:
Chẳng hạn, "I love this movie" và "I adore this movie" thực chất có nghĩa tương tự nhau. Con người có thể nhận ra điều này, nhưng mô hình hiện tại không tận dụng được sự tương đồng này vì mỗi đơn vị từ được coi là một đơn vị độc lập.

3. Xử lý các đặc trưng kết hợp:
Ví dụ, "I love this movie" và "I don’t love this movie" hay "I hate this movie" và "I don’t hate this movie". Việc xác định ý nghĩa của các từ phủ định là khá phức tạp. Từ "love" mang nghĩa tích cực, trong khi "hate" mang nghĩa tiêu cực. Tuy nhiên, "don’t love" lại có nghĩa tích cực hơn "don’t hate", điều này cần phải được xem xét kỹ lưỡng.

4. Xử lý cấu trúc câu:
Một ví dụ phổ biến trong phân tích cảm xúc là từ "but". Từ này thường làm mất đi ý nghĩa của những gì đã nói trước đó, và chỉ cần chú ý đến những gì được nói sau từ "but". Nếu bạn muốn cải thiện độ chính xác của bộ phân loại dựa trên quy tắc, bạn có thể tìm kiếm từ "but" và xóa mọi thứ trước nó.

Những vấn đề này cho thấy rằng BOW cần được cải thiện để có thể hiểu và xử lý ngữ nghĩa một cách chính xác hơn.

A Better Attempt: Neural Network Models



Mạng nơ-ron là một công cụ mạnh mẽ có khả năng thực hiện nhiều nhiệm vụ như phân loại và mô hình ngôn ngữ. Thay vì sử dụng các vector rời rạc (one-hot vectors), mạng nơ-ron sử dụng các embedding từ dày đặc (dense word embeddings) để tra cứu và sau đó đưa vào một hàm phức tạp nhằm trích xuất các đặc trưng.

Quá trình này diễn ra như sau: các từ như "I", "hate", "this", và "movie" được tra cứu để lấy các embedding tương ứng. Những embedding này sau đó được đưa vào một hàm phức tạp để trích xuất các đặc trưng, và cuối cùng, các đặc trưng này được nhân với các trọng số (weights) để tính toán một điểm số (score).

Mạng nơ-ron lý thuyết có thể giải quyết bất kỳ nhiệm vụ nào nếu được thiết kế đủ sâu hoặc đủ rộng. Nếu ai đó nói rằng không thể giải quyết một vấn đề bằng mạng nơ-ron, thì có thể họ không đúng, vì về lý thuyết, mạng nơ-ron có thể giải quyết mọi vấn đề. Tuy nhiên, thực tế vẫn có những vấn đề về dữ liệu và các yếu tố khác có thể ảnh hưởng đến hiệu quả của chúng.

Resources

1. https://phontron.com/class/anlp2024/lectures/#introduction-overview-of-nlp-jan-16