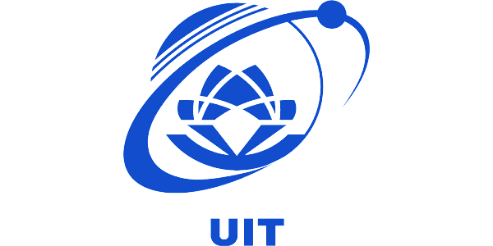
ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐỒ ÁN

**CS221 – XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

Giảng viên hướng dẫn: TS. Nguyễn Trọng Chỉnh

*Đề tài:*

**Xây dựng hệ thống tự động điền (autocomplete) với mô hình N-grams và Add-K Smoothing**

Sinh viên thực hiện: Nguyễn Thành Đạt – 21520705

TP. HỒ CHÍ MINH - 2023

**NHẬN XÉT CỦA GIẢNG VIÊN**

TP HCM, ngày … tháng … năm …

Người nhận xét

(Ký tên)

MỤC LỤC

1. Giới thiệu 1

2. Ngữ liệu 3

2.1 Giới thiệu tổng quan 3

2.2 Phân tích một số ngữ liệu 4

2.3 Tiền xử lí dữ liệu 5

3. Phương pháp luận 8

3.1 Mô hình ngôn ngữ N-gram 8

3.2 Smoothing 10

3.3 Unknow word 11

3.4 Phương pháp đánh giá 11

4. Cài đặt thử nghiệm 13

5. Nhận xét một số trường hợp 14

6. Tài liệu tham khảo 15

# Giới thiệu

Autocomplete đề cập đến việc tự động dự đoán và hoàn thành một từ hoặc cụm từ mà người dùng cần nhập mà không cần gõ hoàn toàn. Bằng cách cung cấp những lựa chọn có sẵn để tăng tốc quá trình nhập liệu, hỗ trợ người gặp vấn đề khi nhập liệu. Tính năng tự động sửa lỗi, hoàn thành hoặc thay thế văn bản đã được gõ trước đó. Nó có thể được sử dụng để sửa từ chứa lỗi ngữ pháp hoặc xác định từ đúng trong từ điển. Mặc dù quá trình chèn từ thường bắt đầu sau khi người dùng kết thúc việc gõ, nó vẫn là một loại dự đoán văn bản.

Hệ thống autocomplete có thể rất hữu ích trong một số trường hợp, phụ thuộc vào tình huống sử dụng và cách triển khai. Autocomplete hoạt động tốt trong các ngữ cảnh như soạn thảo mã nguồn, tài liệu kỹ thuật, văn bản pháp lý,… Nhiều thiết bị di động tích hợp cơ chế này cho phép nhập liệu nhanh hơn.

Language Model (LM - mô hình ngôn ngữ) là một mô hình đại diện cho những kiến thức đã biết về một ngôn ngữ, những kiến thức ấy có thể là những từ, chuỗi các từ có thể có, hay mức độ thường xuyên mà chúng xuất hiện. Language Modeling (mô hình hóa ngôn ngữ) là việc sử dụng các kỹ thuật xác suất và thống kê khác nhau để xác định xác suất của một chuỗi từ nhất định xuất hiện trong một câu. Các mô hình ngôn ngữ phân tích nội dung của dữ liệu văn bản để cung cấp cơ sở cho các dự đoán từ. Chúng được sử dụng trong nhiều ứng dụng của Xử lý ngôn ngữ tự nhiên (NLP). Một số ứng dụng của LM có thể kể đến như dịch máy, hệ thống hỏi đáp, nhận diện lời nói,.. Ngày nay với sự phát triển mạnh mẽ của các công nghệ sản xuất phần cứng máy tính, các mô hình ngôn ngữ ứng dụng Deep Learning có độ chính xác cao hơn nhiều so với các phương pháp truyền thống. Những mô hình ngôn ngữ này lại cần rất nhiều tài nguyên tính toán, thời gian để huấn luyện được một mô hình là rất lớn. Trong khi đó các mô hình ngôn ngữ truyền thống, mặc dù có độ chính xác thấp hơn, không cần quá nhiều tài nguyên và thời gian huấn luyện. Mô hình ngôn ngữ được chia thành ba nhóm đó là Statistical Language Model (Count-based), Neural Network Language Model (Continuous-space) và Knowledge-based Language Model. Ngoài ra còn một số LM khác chẳng hạn như KenLM, ... Trong đó Statistical LM được sử dụng phổ biến nhất nên chúng tôi chọn nó và xây dựng theo mô hình N-grams sẽ đề cập dưới đây.

# Ngữ liệu

Twitter là một nền tảng truyền thông xã hội nổi tiếng cho phép người dùng chia sẻ thông điệp ngắn gọi là "tweet". Trong đề tài này, chúng tôi sử dụng bộ dữ liệu bao gồm hàng ngàn tweets, bài đăng trên blog và tin tức bằng ngôn ngữ tiếng Anh. Bộ dữ liệu được thu thập và tổng hợp thành EN\_US-Twitter gồm những ý kiến, suy nghĩ hoặc sự kiện nhỏ hàng ngày với độ dài ngắn, có liên quan đến tương tác xã hội.

## **2.1 Giới thiệu tổng quan**

Đây là bộ dữ liệu thô vì còn chứa các từ không thể xác định về nghĩa. Bô dữ liệu bao gồm 47962 câu (616296 từ) (khoảng 3.3Mb) được chia thành 2 phần: train (80%), test (20%). Việc phân chia này nhằm đáp ứng cho việc huấn luyện, điều chỉnh các thông số sao cho phù hợp và kiểm thử trên bộ test. Bộ dữ liệu gồm 14944 các từ riêng biệt được đưa vào từ điển.

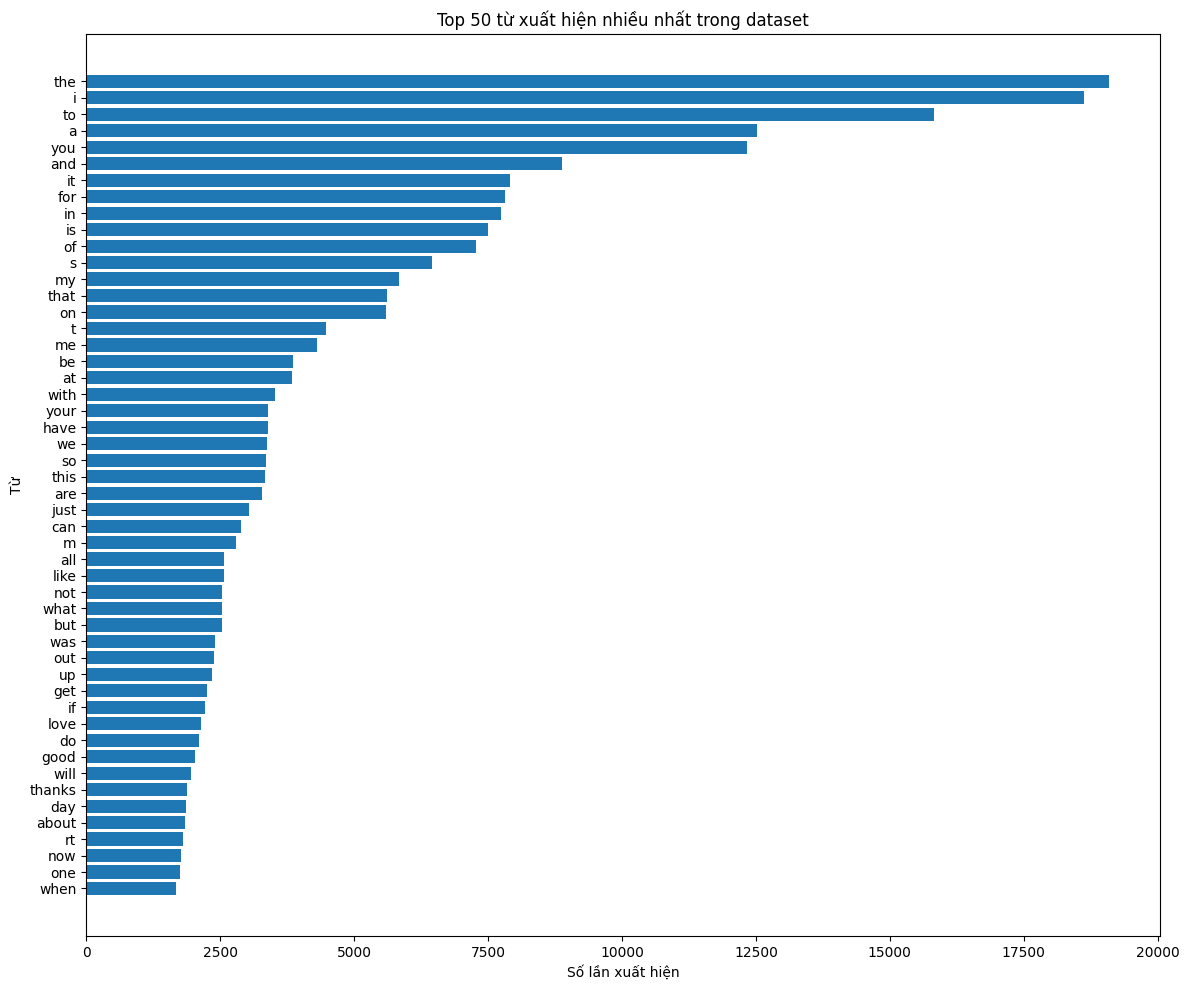
|  |  |  |
| --- | --- | --- |
| **Train** | **Test** | **Vocabulary** |
| 38368 | 9593 | 14944 |

A close up of words

Description automatically generated

Hình : Wordcloud của dataset EN\_US Twitter

Nhìn chung, vì đây là những đoạn tweet thể hiện tâm trạng, suy nghĩ, trạng thái của người dùng twitter nên những từ biểu thị cảm xúc thường chiếm tỉ trọng nhiều trong dataset. Dễ thấy, trong tiếng anh các mạo từ như a, an, the xuất hiện hầu hết trong các câu, chiếm ~ 50% trên tổng số lượng câu trong EN\_US\_Twitter. Số lần xuất hiện này có thể nằm trong một câu hoặc nhiều câu khác nhau.



Hình : Những từ / từ viết tắt xuất hiện nhiều trong EN\_US\_Twitter

## **2.2 Phân tích một số ngữ liệu**

|  |  |
| --- | --- |
| **Nội dung** | **Nhận xét** |
| I want to meet ceaser millaaaan!!!!! | Câu chứa nhiều dấu ! liên tiếp |
| #BadCauldronSongIdeas Barracuda by Heart? That'd be a helluva stretch. | Chứa những #hashtag thường thấy ở những bài tweet nhưng sẽ không sử dụng việc dự đoán |
| Watching The Lion King <3 <3 | Chứa những icon, sẽ được sử dụng trong việc dự đoán |
| No but I hope to soon - how is it? :D Looks beautiful! |
| Way to go, #DrStevenZeitels at, for saving 's voice. 's singers are so lucky to have you! | Quan hệ sở hữu ‘s hoặc một số từ viết tắt như *btw* cũng sẽ được sử dụng để dự đoán |
| oh and btw: happy birthday, celina. i love and miss you. nov 2003 & july 2004. |
| save me some passsssghetttttttti!!!!!! | Một số từ được kéo dài sẽ được loại bỏ vì tần suất xuất hiện không nhiều |
| I decided I'm going!! I think my dad may comee with me if not its jus me n my baby girl!!!! She's so excited!!;) |
| #MyPastRelationships sucked. That's why they're.........wait for it.........PAST RELATIONSHIPS! |
| "Sefl-praise is for losers.Be a winner.Stand for Something, Always have class and be humble"-John Madden NFL Coach | Một số trích dẫn có -Jo xuất hiện nhiều trong bộ ngữ liệu, bên cạnh đó còn sai chính tả. |
| -AKA- Young Blake. I just finished listening to 'Audio Thoughts Vol.' U SPIT FIRE MY BOY. KEEP IT MOVIN & GOD BLESS! :-) | Một số ký tự đại diện cho từ như & sẽ không được dự đoán |
| yikes.."since 2007, RIM has introduced 37 models" -- A Boggle of Blackberrys nyti.ms/tKK5hZ | Ngữ liệu chứa một số hyperlink gây ảnh hưởng đến tách từ |
| I will <3 | Một số câu chưa hoàn thành đầy đủ ý nghĩa |

## **2.3 Tiền xử lí dữ liệu**

*2.3.1 Phân tách bộ ngữ liệu thành các câu.*

Bộ ngữ liệu (corpus) gồm nhiều câu đa dạng chủ đề vì thế cần được tách thành nhiều câu để tổ chức dữ liệu dễ dàng hơn, có thể triển khai tính toán dễ dàng hơn.

Ví dụ về tách câu một phần ngữ liệu nhỏ trong dataset:

['How are you? Btw thanks for the RT. You gonna be in DC anytime soon? Love to see you. Been way, way too long.',

"When you meet someone special... you'll know. Your heart will beat more rapidly and you'll smile for no reason.",

"they've decided its more fun if I don't.",

'So Tired D; Played Lazer Tag & Ran A LOT D; Ughh Going To Sleep Like In 5 Minutes ;)',

...]

*2.3.2 Tokenization*

Mỗi từ trong câu sẽ được tách thành các tokens gồm các đơn vị ngôn ngữ nhỏ như từ, ký tự. Mục đích làm cho việc xử lý ngôn ngữ thuận tiện hơn, bước xử lý đầu tiên cho việc xây dựng từ điển.

Ví dụ về tokenization:

[

['how', 'are', 'you', '?', 'btw', 'thanks', 'for', 'the', 'rt', '.', 'you', 'gon', 'na', 'be', 'in', 'dc', 'anytime', 'soon', '?', 'love', 'to', 'see', 'you', '.', 'been', 'way', ',', 'way', 'too', 'long', '.'],

...

]

*2.3.3 Xử lý trường hợp unknown word (từ không có trong từ điển) (OOV)*

Để giải quyết vấn đề này, chúng tôi thay thế các từ không biết bằng một token đặc biệt <unk> đảm bảo giữ nguyên kích thước của từ vựng và cung cấp thông tin rằng có từ không biết trong câu, những từ này có tần suất xuất hiện thấp, và chúng tôi đặt ngưỡng threshold = 2 cho mỗi từ được liệt kê vào từ điển.

Ví dụ:

|  |  |
| --- | --- |
| Câu gốc | Tiền xử lý (OOV) |
| Where to begin! I've been lucky, privileged really, to be schooled by so many wonderful teachers. | ['where',  'to',  'begin',  '!',  'i',  "'ve",  'been',  'lucky',  ',',  '**<unk>**',  'really',  ',',  'to',  'be',  '**<unk>**',  'by',  'so',  'many',  'wonderful',  'teachers',  '.'] |

Các từ như ‘privileged’, ‘schooled’ chỉ xuất hiện ở câu này nên đã được thay thế bằng token <unk> vì không có trong bộ từ điển.

*2.3.4.* Thêm tokens <s> ở đầu và <e> ở cuối của mỗi câu

Việc thêm các tokens đặc biệt này giúp mô hình có thể dễ dàng nhận biết được điểm bắt đầu và kết thúc của mỗi câu trong dữ liệu.

# Phương pháp luận

Ảnh có chứa văn bản, biểu đồ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

## **3.1 Mô hình ngôn ngữ N-gram**

N-gram là một Statistical Language Models (Mô hình thống kê). Những mô hình thuộc nhóm này đều dựa trên việc đếm tần suất xuất hiện của từng từ hay chuỗi, do đó nó cần một lượng dữ liệu lớn. Statistical LM là phân bố xác suất P(W), W ∈ V+ trên tập hợp tất cả các chuỗi V+ có trong tập dữ liệu thuộc bộ từ vựng V của một ngôn ngữ. Trong đó, xác suất của một chuỗi có độ dài K được tính như sau:

là xác suất của từ khi đã biết chuỗi và được tính bằng cách đếm chuỗi như sau:

Nếu K càng lớn thì tần suất xuất hiện của chuỗi K từ càng nhỏ và sẽ có trường hợp tần suất xuất hiện của chuỗi K−1 từ dẫn trước là lớn, khiến cho xác suất

P(wk | w1w2...,wk-1) gần như bằng 0. Để giảm thiểu vấn đề này giả định Markov chỉ ra rằng xác suất của một từ chỉ phụ thuộc vào một số hữu hạn các từ trước đó, chứ không phải toàn bộ lịch sử. Điều này giúp giảm bớt độ phức tạp tính toán và giữ cho mô hình khả thi trong việc xử lý ngôn ngữ tự nhiên. Thay vì tính theo K – 1 thì ta chỉ chọn N − 1 từ, và từ đó ta có **N-gram Language Models**.

Theo đó, công thức tổng quát của mô hình N-gram là:

Nếu chỉ chọn một từ dẫn trước (N = 2) ta sẽ có xác suất xuất hiện một từ như sau:

Chúng ta hãy xem xét một ví dụ bằng cách sử dụng một bộ ngữ liệu nhỏ gồm ba câu. Trước tiên cần thêm mỗi câu bằng ký hiệu đặc biệt <s> ở đầu của câu, để cung cấp cho chúng ta ngữ cảnh bigram của từ đầu tiên. Chúng tôi cũng sẽ cần một ký hiệu kết thúc đặc biệt <e>.

<s> I am Sam <e>

<s> Sam I am <e>

<s> I do not like green eggs and ham <e>

Dưới đây là các tính toán cho một số xác suất từ bộ văn bản này:

* Bigram (N=2)

P(I|<s>) = = 0.33 P(Sam|<s>) = = 0.33

P(am|I) = = 0.67 P(<e>|Sam) = =0.5

P(Sam|am) = =0.5 P(do|I) = = 0.33

* Trigram (N=3)

P(I am|<s>) = = 0.22 P(Sam I|<s>) = = 0.167

P(am Sam|I) = = 0.33 P(I do|<s>) = = 0.11

* 4-gram (N=4)

P(I am Sam|<s>) = = 0.11 P(Sam I am|<s>) = = 0.11

P(am Sam <s/>|I) = = 0.167 P(I do not|<s>) = = 0.11

Chúng ta cần biểu diễn và tính toán xác suất của mô hình ngôn ngữ dưới dạng **log** để đơn giản hóa quá trình tính toán. Vì theo định nghĩa, xác suất luôn nhỏ hơn hoặc bằng 1, nên càng nhân nhiều xác suất lại với nhau, tích của chúng càng trở nên nhỏ hơn. Nhân tất cả n-gram lại với nhau có thể dẫn đến hiện tượng mất mát số liệu do quá nhỏ. Bằng cách sử dụng log xác suất thay vì xác suất thô, chúng ta thu được các số liệu quá nhỏ như vậy.

Thêm vào đó, phép cộng trong log tương đương với việc nhân trong tuyến tính, nên chúng ta kết hợp log xác suất bằng cách cộng chúng lại. Kết quả của việc thực hiện tất cả các phép tính và lưu trữ trong không gian log là chúng ta chỉ cần chuyển đổi trở lại thành xác suất nếu chúng ta cần báo cáo chúng cuối cùng. Sau đó, chúng ta chỉ cần lấy mũ của log xác suất để có giá trị xác suất.

## **3.2 Smoothing**

Smoothing là phương pháp được sử dụng để giải quyết các trường hợp các từ không xuất hiện trong tập dữ liệu training nhưng có xuất hiện trong tập dữ liệu test. Smoothing còn được gọi là discounting. Để giữ cho Language Model không gán xác suất bằng 0 cho những từ (cụm từ) này, chúng ta phải giảm bớt xác suất của những từ (cụm từ) hay xuất hiện và tăng xác suất cho những từ (cụm từ) không có trong tập dữ liệu training. Tuy nhiên phải đảm bảo tổng xác suất của tất cả các từ vẫn bằng 1. Có nhiều phương pháp smoothing như: Laplace (add-one) smoothing, add-k smoothing, stupid backoff, và Kneser-Ney smoothing,...Trong thực nghiệm này chúng tôi chỉ thực nghiệm add-k smoothing.

## **3.3 Unknow word**

Đối với trường hợp những từ không xuất hiện trong tập train của chúng ta nhưng xuất hiện trong tập test thì sẽ có cách giải quyết sau đây. Các từ chưa bao giờ gặp trước đó, được gọi là từ không biết, hoặc từ nằm ngoài từ vựng (OOV - out of vocabulary). Tỷ lệ từ không biết (OOV) xuất hiện trong tập kiểm tra được gọi là tỷ lệ OOV. Một cách để tạo ra hệ thống từ vựng mở (open vocabulary) là mô phỏng những từ không biết có thể xuất hiện trong tập kiểm tra bằng cách thêm một từ giả định được gọi là <unk>, thay thế các từ trong dữ liệu huấn luyện bằng <unk> dựa trên tần suất của chúng. Ví dụ, chúng ta có thể thay thế bằng <unk> tất cả các từ xuất hiện ít hơn n lần trong tập huấn luyện sau đó tiếp tục huấn luyện mô hình ngôn ngữ như trước, xử lý <unk> như là một từ thông thường.

## **3.4 Phương pháp đánh giá**

Để đánh giá hiệu suất của một mô hình ngôn ngữ là tích hợp nó vào một ứng dụng và đo lường mức độ cải thiện của ứng dụng. Việc chạy các hệ thống xử lý ngôn ngữ tự nhiên lớn từ đầu đến cuối thường rất tốn kém. Thay vào đó, việc có một chỉ số có thể được sử dụng để đánh giá nhanh chóng sự cải thiện tiềm năng trong một mô hình ngôn ngữ. Chúng ta sử dụng phương pháp đánh giá là Perplexity (độ hỗn loạn thông tin). Perplexity của một mô hình ngôn ngữ là xác suất nghịch đảo của bộ dữ liệu thử nghiệm và được chuẩn hóa bởi số từ của toàn bộ dữ liệu thử nghiệm. Công thức tính Perplexity của dữ liệu thử nghiệm :

Sử dụng quy luật chain rule để mở rộng xác suất của W:

Trong mô hình ngôn ngữ, độ hỗn loạn thông tin của một văn bản với từ “this” thể hiện số từ có thể đi sau từ “this”. Độ hỗn loạn thông tin của một mô hình ngôn ngữ nói chung, có thể hiểu đơn giản là số lựa chọn từ trung bình mà mô hình ngôn ngữ phải đưa ra quyết định. Như vậy, độ hỗn loạn thông tin càng thấp, thì độ chính xác của mô hình ngôn ngữ càng cao.

Dựa vào công thức ta thấy rằng xác suất điều kiện của cụm từ càng cao thì độ hỗn loạn thông tin càng thấp. Vì vậy giảm thiểu độ hỗn loạn thông tin đồng nghĩa với việc tăng cực đại xác suất của tập thực nghiệm tương ứng với mô hình ngôn ngữ.

Vì là nghịch đảo xác suất, nên giữa 2 mô hình thì perplexity của mô hình nào thấp hơn thì mô hình đó hiệu quả hơn.

Để đánh giá hiệu quả của mô hình, chúng tôi huấn luyện bigram, trigram, 4-gram và 5-gram dưới dạng truy hồi và làm mịn bằng add-k trên tập train đã đề cập ở trên và kiểm thử trên tập test. Bảng dưới cho kết quả tương ứng trung bình perlexity của tập validation cho từng mô hình:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n-gram | Bigram | Trigram | 4-gram | 5-gram |
| Perplexity (train) | 179 | 549 | 321 | 180 |
| Perplexity (test) | 1236 | 1733 | 822 | 446 |

Dựa vào bảng ta có thể thấy rằng, càng nhiều thông tin mà n-gram cung cấp về các cụm từ thì độ hỗn loạn thông tin càng nhỏ.

# Cài đặt thử nghiệm

Việc thực nghiệm sẽ giúp chúng ta hiểu rõ hơn cách hoạt động của N-gram Language Model. Như đã đề cập ở trên, ngữ liệu sử dụng trong việc thực nghiệm là EN\_US\_Twitter. Mô hình được cài đặt bằng ngôn ngữ lập trình Python. Nền tảng thí nghiệm chúng tôi sử dụng là Google Colab, nền tảng này cho phép chạy các chương trình viết bởi python trên nền web và cung cấp các phần cứng mạnh mẽ giúp cho việc chạy các chương trình với tốc độ nhanh hơn. Demo của thí nghiệm được cài đặt trên nền Web. Đối với bước phân tích dữ liệu và tiền xử lí dữ liệu, chúng tôi sử dụng Pandas và nltk. Pandas một thư viện của Python chuyên dùng xử lí dữ liệu. Thư viện có sẵn các hàm phân tích, sàng lọc giúp cho mô hình cài đặt hoạt động tốt hơn với bộ dữ liệu thực nghiệm. Thư viện ntlk có các hàm chuyên làm việc với dữ liệu dạng text và có các bộ dữ liệu riêng biệt về ngôn ngữ. Tiếp theo đó là việc cài đặt mô hình, chúng tôi sử dụng thư viện Numpy của Python để cài đặt mô hình giúp tăng hiệu suất về mặt thời gian trong việc huấn luyện 2 mô hình.

**Source code:**

Google Colab: <https://drive.google.com/file/d/1q4XaxIcNrPs0qFhks7XFDvVI5OPNqmRu/view?usp=sharing>

N-gram API: <https://github.com/khaifade/ngram>

**Live demo**: <https://autocomplete-one.vercel.app/> (server sẽ vào trạng thái sleeping sau mỗi 15 phút không có request, vì thế khi có request mới server cần thời gian khởi động lại)

# Nhận xét một số trường hợp

|  |  |
| --- | --- |
| **Trường hợp** | **Nhận xét** |
|  | Kết quả trả về lần lượt của bigram, trigram, 4-gram và 5-gram dễ thấy, với input là ‘how to’ thì mỗi ngram cho kết quả khác nhau và xác suất của bigram vẫn là cao nhất. |
|  | Với keyword ‘there’s a …’ bigram lại cho ra kết quả **<unk>** đây chính là từ mà khi mô hình đã loại bỏ do tần suất xuất hiện ít nếu chỉ dựa vào 1 từ trước đó. Các mô hình trigram, 4-gram và 5-gram cho kết quả có ý nghĩa hơn. |
|  | Một trường hợp khác thể hiện các mô hình với ngram lớn hơn sẽ cho kết quả mang ý nghĩa hơn. |
|  | Trong một số trường hợp khi ngam lớn tính xác suất quá nhỏ với những từ ít xuất hiện, thì mô hình ngram nhỏ hơn lại thể hiện tốt hơn. |
|  | Với những câu hoàn chỉnh, các mô hình ngram đều đề xuất là kết thúc câu, đặc biệt với những kết thúc bằng biểu tượng - đây là văn phong thường được sử dụng trong ngôn ngữ hàng ngày |

# Tài liệu tham khảo

Standford University, *N-gram Language Models*, <https://web.stanford.edu/~jurafsky/slp3/3.pdf>

Cao Văn Kiệt,  *Xây dựng mô hình ngôn ngữ cho Tiếng Việt,* Khoá luận tốt nghiệp, K51, Khoa học máy tính, Trường Đại học Công nghệ.

Lectures online: Natural Language Processing with Probabilistic Models – Deep Learning AI