**­­TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----□□&□□-----

****

**ĐỒ ÁN MÔN HỌC**

**XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

**XÂY DỰNG MÔ HÌNH HỌC MÁY CHO BÀI TOÁN SỬA LỖI CHÍNH TẢ TRONG TIẾNG VIỆT**

|  |  |
| --- | --- |
| **Lớp: 64CS3 - Nhóm: 6** | |
| **Thành viên nhóm** | Nguyễn Xuân Cường - 1507764  Dương Gia Khánh - 1526864  Vương Trung Thành - 186864  Ngô Đức Thịnh - 189464  Nguyễn Trần Lê Tuấn - 1553564 |
| **Giảng viên hướng dẫn** | TS. Phạm Hồng Phong |

**HÀ NỘI, 06 - 2022**

# Lời nói đầu

Ngày nay, cùng với sự phát triển vượt bậc của con người về mọi mặt thì trí tuệ nhân tạo ngày càng được ứng dụng nhiều trong tất cả mọi lĩnh vực đời sống.Nó không chỉ làm cuộc sống trở lên tốt mà còn xử lý được những công việc khó khan tốn nhiều thơi gian.Học máy là một lĩnh vực thuộc trí tuệ nhân tạo. Gần đây, học máy đang dẫn đầu xuthế và tạo nên những thay đổi vượt bậc trong trí tuệ nhân tạo nói chung và công nghệthông tin nói riêng.

Trong đồ án “Xử lý ngôn ngữ tự nhiên”, nhóm 06 - 64CS3 thực hiệnmột project liên quan đến học máy giúp hiểu hơn một bài toán học máy cũngnhư tạo ra một giải pháp học máy có ứng dụng rộng rãi trong thực tế.Nhóm chúng em đã chọn chủ đề sửa lỗi chính tả trong tiếng Việt. Đây là 1 trong những chủ đề tương đối quan trọng trong xử lý ngôn ngữ tự nhiên. Do thời gian và khả năng có hạn nên không thể tránh khỏi sai sót. Vì vậy nhóm chúng em thầy có thể nhận xét và đưa ra ý kiến để chúng em hoàn thiện hơn bài báo cáo của nhóm mình. Cuối cùng, nhóm 06 xin chân thành cảm ơn thầy Phạm Hồng Phong đã hướng dẫnchúng em trong suốt thời gian làm báo cáo này.

Chúng em xin chân thành cảm ơn!

**MỤC LỤC**

[Lời nói đầu 1](#_Toc106391177)

[CHƯƠNG 1. GIỚI THIỆU 4](#_Toc106391178)

[1.1. ĐẶT VẤN ĐỀ 4](#_Toc106391179)

[1.2. GIỚI THIỆU BÀI TOÁN SỬA LỖI CHÍNH TẢ 5](#_Toc106391180)

[1.3. MÔ TẢ BÀI TOÁN 5](#_Toc106391181)

[1.4. ĐỊNH HƯỚNG ĐỀ TÀI 5](#_Toc106391182)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 7](#_Toc106391183)

[2.1. LÝ THUYẾT NGÔN NGỮ TIẾNG VIỆT 7](#_Toc106391184)

[2.1.1. Khái niệm về từ 7](#_Toc106391185)

[2.1.2. Hình thái từ tiếng Việt 7](#_Toc106391186)

[2.1.3. Khái niệm văn bản 9](#_Toc106391187)

[2.2. LỖI CHÍNH TẢ 10](#_Toc106391188)

[2.2.1. Các nguyên nhân gây ra lỗi chính tả 10](#_Toc106391189)

[2.2.2. Phân loại một số lỗi chính tả 11](#_Toc106391190)

[2.2.3. Phát hiện lỗi chính tả 11](#_Toc106391191)

[2.2.4. Sửa lỗi chính tả 11](#_Toc106391192)

[2.2.5. Một số phương pháp kiểm lỗi chính tả 12](#_Toc106391193)

[2.3. MÔ HÌNH NGÔN NGỮ N-GRAM 12](#_Toc106391194)

[2.3.1. Giới thiệu 12](#_Toc106391195)

[2.3.2. Công thức tính “xác suất thô” 13](#_Toc106391196)

[2.3.3. Tính xác suất dựa trên N-gram 14](#_Toc106391197)

[2.3.4. N-gram đơn giản 14](#_Toc106391198)

[2.3.5. Làm mịn (smoothing) 17](#_Toc106391199)

[2.3.6. Áp dụng cho tiếng Việt 18](#_Toc106391200)

[2.4. TÁCH TỪ TIẾNG VIỆT 19](#_Toc106391201)

[2.4.1. Xử lý nhập nhằng 19](#_Toc106391202)

[2.4.2. Tách từ tiếng Việt dùng mô hình Maximum Matching 19](#_Toc106391203)

[CHƯƠNG 3. MÔ HÌNH – CÀI ĐẶT 21](#_Toc106391204)

[3.1. MÔ HÌNH LSTM 21](#_Toc106391205)

[3.1.1. Lý do chọn LSTM 21](#_Toc106391206)

[3.1.2. Cơ sở lý thuyết 21](#_Toc106391207)

[3.1.3. Mạng long-short term memory 29](#_Toc106391208)

[3.2. CẢI TIẾN LSTM 34](#_Toc106391209)

[3.2.1. Mạng LSTM hai chiều (Bidirectional-LSTM) 34](#_Toc106391210)

[3.3. XÂY DỰNG MÔ HÌNH 35](#_Toc106391211)

[3.3.1. Thu thập dữ liệu 35](#_Toc106391212)

[3.3.2. Lựa chọn công cụ xử lý dữ liệu 36](#_Toc106391213)

[3.3.3. Tiền xử lý dữ liệu 37](#_Toc106391214)

[3.3.4. Xây dựng bộ từ điển 40](#_Toc106391215)

[3.3.5. Vector hóa từ và văn bản 40](#_Toc106391216)

[3.3.6. Huấn luyện mô hình 40](#_Toc106391217)

[CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM 42](#_Toc106391218)

[4.1. KIỂM THỬ 42](#_Toc106391219)

[4.2. NHẬN XÉT 43](#_Toc106391220)

[4.2.1. Phân loại lỗi chính tả 43](#_Toc106391221)

[4.2.2. Đánh giá độ chính xác 44](#_Toc106391222)

[CHƯƠNG 5. kết luận và hướng phát triển 45](#_Toc106391223)

[5.1. KẾT LUẬN 45](#_Toc106391224)

[5.2. HƯỚNG PHÁT TRIỂN 45](#_Toc106391225)

[TÀI LIỆU THAM KHẢO 46](#_Toc106391226)

# GIỚI THIỆU

## ĐẶT VẤN ĐỀ

Ngày nay với sự phát triển vượt bậc của ngành Công nghệ thông tin, con người đã thu được hàng loạt những thành tựu to lớn trong việc giải quyết các bài toán thực tiễn. Chính sự phát triển nhanh chóng của Công nghệ thông tin và truyền thông đã đem lại cho thế giới của chúng ta bộ mặt mới. Nền kinh tế tri thức, hợp tác toàn cầu, những công việc quản lý, vận hành những hoạt động từ vi mô tới vĩ mô của doanh nghiệp, tin học hóa quy trình hành chính, điều khiển tác nghiệp, giải trí, liên lạc, trợ giúp,... là những ứng dụng tiêu biểu của Công nghệ thông tin và truyền thông. Một trong các lĩnh vực khoa học ứng dụng thành tựu đó đang thu hút rất nhiều sự quan tâm của các nhà khoa học đó là Xử lý ngôn ngữ tự nhiên. Ứng dụng của xử lý ngôn ngữ tự nhiên trong rất nhiều lĩnh vực như: Dịch máy, điều khiển, nhận dạng, hệ hỗ trợ ra quyết định ..., đem lại lợi ích tối đa cho con người.

Ngôn ngữ là một phần quan trọng của đời sống, là phương tiện truyền tải thông tin trong đời sống. Trong thời đại bùng nổ thông tin hiện nay thì ngôn ngữ đóng vai trò hết sức quan trọng, đặc biệt là ngôn ngữ viết. Khi viết, đôi khi ta mắc phải những lỗi sai chính tả. Chữ quốc ngữ là thứ ngữ ghi âm nên một số âm tiết rất dễ nhầm lẫn, khó phân biệt rõ ràng. Ngôn ngữ nói ở những vùng khác nhau thì lại có những đặc điểm khác nhau. Những điểm khác nhau này rất dễ gây ra những lỗi chính tả khi viết nếu người viết không để ý sử dụng tiếng Việt. Những thao tác chuyển thông tin ở dạng văn bản khác nhau cũng có thể gây ra lỗi chính tả. Khi ghi lại lời nói của người khác mà người đó sử dụng giọng địa phương cũng có thể dẫn đến những lỗi chính tả. Quét các văn bản giấy thành văn bản điện tử, sử dụng chương trình nhận dạng chữ, cũng có thể dẫn đến lỗi chính tả do chương trình nhận dạng nhầm lẫn ... Văn bản dễ bị sai chính tả do nhiều yếu tố khách quan. Để kiểm lỗi chính tả những văn bản này đòi hỏi nhiều công sức và thời gian, đặc biệt là khi khối lượng văn bản bùng nổ như hiện nay. Do đó cần có một công cụ hỗ trợ kiểm tra lỗi chính tả, giúp nhanh chóng phát hiện lỗi chính tả và đề nghĩ khắc phục.

Trong thời đại tin học hóa, máy tính được tận dụng để giảm thiểu công sức của con người, đồng thời tăng tính hiệu quả. Tin học đã được áp dụng trong nhiều lĩnh vực khác nhau và chứng tỏ tính hiệu quả của nó. Những ứng dụng kiểm tra lỗi chính tả hiện có như VietRes, VietSpell, ... hiện vẫn còn khá đơn giản hoặc chưa hiệu quả, chưa đáp ứng được nhu cầu của thực tế. Trong bài toán này nhóm 06 muốn trình bày kết quả tìm hiểu về bài toán sửa lỗi chính tả tiếng việt trong văn bản.

## GIỚI THIỆU BÀI TOÁN SỬA LỖI CHÍNH TẢ

Bài toán sửa lỗi chính tả là một bài toán khá phức tạp, được không ít đơn vị nghiên cứu, phát triển và nó có tính ứng dụng cao, đặc biệt là trong các ứng dụng soạn thảo hay nhận dạng văn bản. Chương trình sửa lỗi chính tả cần có hai chức năng chính, cơ bản là chỉ ra lỗi sai và đưa ra gợi ý sửa lỗi. Tuy nhiên, các chức năng kiểm lỗi chính tả được tích hợp nhiều trong ứng dụng soạn thảo tiếng Việt hiện nay (Vietkey, Unikey, ...) không đưa ra gợi ý cho người dùng lựa chọn.

Để giải quyết bài toán này, hiện nay có một số cá nhân hoặc nhóm nhà nghiên cứu và phát triển phần mềm kiểm lỗi chính tả như:

* **Google Docs**: là ứng dụng dùng để hỗ trợ soạn thảo trực tuyến, được cung cấp bởi nhà phát hành Google. Google Docs đặc biệt ở chỗ cho phép người dùng kiểm tra chính tả một cách nhanh chóng ngay trên giao diện làm việc.
* **Vspell**: là website đầu tiên kiểm tra chính tả tiếng Việt từ năm 1990. Bạn không cần phải tải về hay cài đặt mà vẫn có thể soát lỗi chính tả tiếng Việt chính xác.
* **VcatSpell**: VCatSpell là phần mềm kiểm tra lỗi chính tả tiếng Việt đầu tiên và được ra mắt chính thức vào năm 1990. Cho đến hiện tại, phần mềm vẫn sở hữu nhiều điểm cộng được người dùng đánh giá cao.
* **Tummo Spell**: Tummo Spell là phiên bản mới nhất của phần mềm kiểm tra lỗi chính tả văn bản Word, Excel nhanh chóng và hiệu quả. Đồng thời cũng giúp bạn sửa lỗi nhanh hơn.
* **TinySpell**: TinySpell là một tiện ích nhỏ cho phép bạn kiểm tra và sửa lỗi chính tả dễ dàng và nhanh chóng trong bất kỳ ứng dụng Windows nào.

## MÔ TẢ BÀI TOÁN

Bài toán sửa lỗi chính tả tiếng Việt được mô tả như sau:

Input:

* Tập chuỗi các từ đầu vào *X* = {*x1, x2, …, xn*} với một vài từ sai chính tả *xi* bất kì.

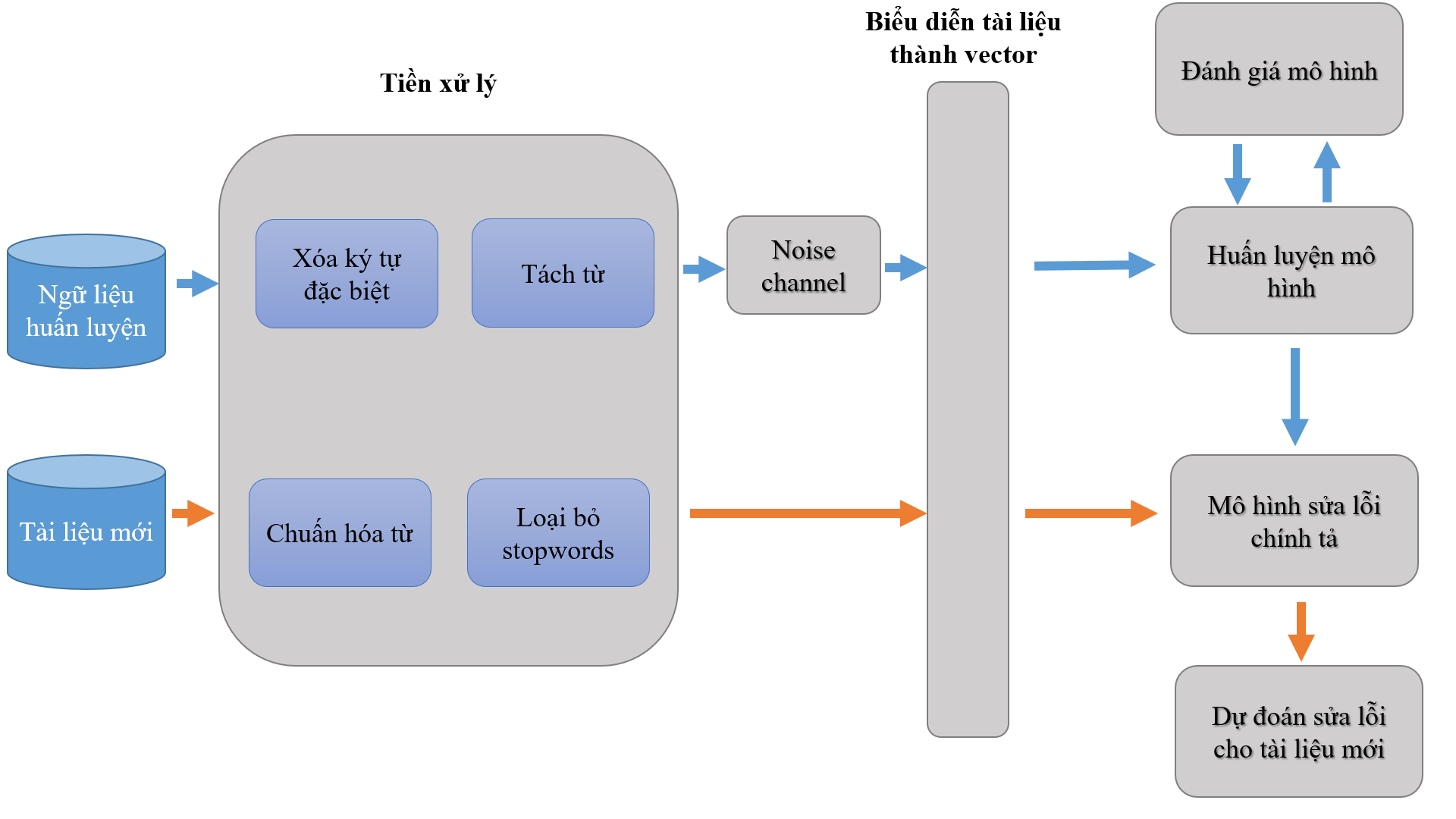
Output:

* Tập chuỗi các từ đầu ra *Y* = {*y1, y2, … yn*} với *yi* là từ đúng chính tả được sửa.

Trong bài toán này, với mỗi từ sai chính tả xi ta cần sửa thành một từ đúng yi, tức là ta cần phải xây dựng một hàm *f: X Y* thỏa mãn *f(xi) = yi.*

## ĐỊNH HƯỚNG ĐỀ TÀI

Trong đồ án sửa lỗi chính tả này, nhóm sẽ lần lượt thực hiện các bước sau: Đầu tiên nhóm sẽ thu thập bộ dữ liệu để huấn luyện. Tiếp theo, nhóm sẽ tiến hành quy trình tiền xử lý dữ liệu để làm sạch văn bản và đưa văn bản này thông qua một kênh nhiễu để tạo ra các lỗi chính tả một cách ngẫu nhiên.



Hình 1.1 Mô tả quy trình giải quyết bài toán xử lý văn bản của nhóm

Tập văn bản nhiễu lần lượt đưa chuyển hóa sang bộ vector. Sau đó ta thực hiện xong xong quy trình huấn luyện mô hình học máy và đánh giá quá trình huấn luyện với bộ vector này.

Mục tiêu của đề tài là với dữ liệu văn bản mới đầu vào, ta cũng thực hiện lần lượt bước tiền xử lý dữ liệu như với tập huấn luyện. Sau đó đưa tập văn bản mới vào mô hình để dự đoán sửa lỗi các từ sai chính tả.

# CƠ SỞ LÝ THUYẾT

## LÝ THUYẾT NGÔN NGỮ TIẾNG VIỆT

### Khái niệm về từ

Trong quá trình học tập và sử dụng ngôn ngữ trong đời sống hằng ngày, mỗi chúng ta đều quen thuộc với khái niệm về "tử". Nhưng để định nghĩa được chính xác từ là gì hoàn toàn không phải là một vấn đề đơn giản. Trong ngành ngôn ngữ học, đã có hàng trăm định nghĩa về từ được đưa ra, nhưng hầu như chưa có một định nghĩa nào có thể bao quát hết được mọi vấn đề liên quan đến khái niệm “từ”. Theo công trình của **PGS.TS. Đinh Điền**, có một số khái niệm tiêu biểu sau đây về từ:

* Theo L Bloomfield thì: "từ là một hình thái tự do nhỏ nhất”.
* B.Golovin quan niệm: “từ là đơn vị nhỏ nhất có nghĩa của ngôn ngữ, được vận dụng độc lập, tái hiện tự do trong lời nói đẻ xây dựng nên câu”.
* Còn Solncev thì lại quan niệm: "Từ là đơn vị ngôn ngữ có tính hai mặt : âm và nghĩa. Từ có khả năng độc lập về cú pháp khi sử dụng trong lời".

Trong tiếng Việt, cũng có nhiều định nghĩa về tử như:

* Theo nhà nghiên cứu Ngô ngữ học Trương Văn Trinh và Nguyễn Hiến Lê thì: “Từ là âm có nghĩa, dùng trong ngôn ngữ để diễn tả một ý đơn giản nhất, nghĩa là ý không thể phân tích ra được".
* PGS. Nguyễn Kim Thản thì định nghĩa: “Từ là đơn vị cơ bản của ngôn ngữ, có thể tách khỏi các đơn vị khác của lời nói để vận dụng một cách độc lập và là một khối hoàn chỉnh về ý nghĩa (từ vựng hay ngữ pháp) và cấu tạo”.
* Theo Hồ Lê, “Từ là đơn vị ngôn ngữ có chức năng định danh pgu liên kết hiện thực, hoặc chức năng mô phỏng tiếng động, có khả năng kết hợp tự do, có tính vững chắc về cấu tạo và tính nhất thể về ý nghĩa”.

### Hình thái từ tiếng Việt

Như trình bày trong phần trên, có rất nhiều định nghĩa về từ nhưng các nhà ngôn ngữ học vẫn chưa thống nhất quyết định chọn theo lối định nghĩa nào. Điều này cũng xảy ra trong tiếng Việt của chúng ta. Do vậy, với mục đích phục vụ thuận tiện cho việc xử lý tự động ngôn ngữ bằng máy tính, nhưng vẫn phù hợp với các định nghĩa về từ trong ngôn ngữ học đại cương cũng như tính đặc thủ của ngôn ngữ đơn lập như tiếng Việt.

#### Hình vị tiếng Việt

Đầu tiên, chúng tôi sử dụng quan niệm của công trình của **PGS.TS. Đinh Điền** như sau: tiếng là đơn vị cơ bản trong tiếng Việt vì nó có thể nhận diện tương đối dễ dàng bởi người bản ngữ cũng như nhận diện một cách tự động bởi máy tính. Xét về mặt kỹ thuật trên máy tính, ta cũng có thể thực hiện được các thao tác lưu trữ, xử lý, tìm kiếm và sắp xếp các tiếng một cách dễ dàng do số lượng cũng như chiều dài của các tiếng này là nhỏ (Trong tiếng Việt, có khoảng 9270 tiếng các loại, và chiều dài của mỗi tiếng cũng được giới hạn là 7 ký tự - ***nghiêng*** là tiếng dài nhất với 7 ký tự).

Ngoài ra, tiếng còn được xem là “từ chính tả". Tuy nhiên, nếu xét trên các tiêu chí của ngôn ngữ học, thì tiếng không thể được xem là một từ thực sự. Thậm chí, tiếng cũng chưa hoàn toàn dù tư cách để được xem là “hình vị thực sự” vì chưa thỏa tiêu chí về nội dung (phải có ý nghĩa hoàn chỉnh). Vì vậy, trong đồ án này, chúng tôi dựa theo quan điểm của **PGS.TS. Đinh Điền** trong công trình là xem tiếng chỉ là “hình vị tiếng Việt":

**Hình vị tiếng Việt** ở đây phải được hiểu là: bên cạnh khái niệm *hình vị* như trong ngôn ngữ học đại cương, còn phải xét đến yếu tố *hình tố*, là yếu tố thuần túy hình thức biểu hiện những kiểu quan hệ bên trong giữa các thành tố trong từ. Ta có thể gọi đây là những “tha hình vị” hay “á hình vị”. Như vậy, trong tiếng Việt sẽ có 3 loại hình vị như sau:

* **Hình vị gốc**: là những nguyên tố, đơn vị nhỏ nhất, có nghĩa, chúng có thể là hình vị thực (là những từ vựng) hay hình vị hư (ngữ pháp), chúng có thể đứng độc lập hay bị ràng buộc.
* **Tha hình vị**: vốn cũng là hình vị gốc, nhưng vì mối tương quan với các thành tổ khác trong từ mà chúng biến đổi đi về âm, nghĩa,... Tha hình vị bao gồm:
  + **Tha hình vị láy nghĩa:** trong các từ ghép bội nghĩa, như: giá cả, hỏi han, tuổi tác,...; nhà cửa, yêu thương, ngược xuôi,...
  + **Tha hình vị láy âm:** chúm chím, đo đỏ,...; lé đé, đủng đỉnh,....
  + **Tha hình vị định tính:** là các yếu tố phụ để miêu tả thuộc tính, như: xanh lè, tối om, cười khẩy,....
  + **Tha hình vị tựa phụ tố:** là đơn vị hoạt động giống như những phụ tố (affix) trong các ngôn ngữ biến hình, như: giáo viên, hiện đại hoá, tân tổng thống,…
* **Á hình vị:** là những chiết đoạn ngữ âm được phân xuất một cách tiêu cực, thuần túy dựa vào hình thức, không rõ nghĩa, song có giá trị khu biệt, làm chức năng cấu tạo từ. Ví dụ: dưa hấu, dưa gang, bí ử, đậu nành, cà niễng,....

#### Từ tiếng Việt

Trong đồ án này, chúng tôi sử dụng nghĩa từ theo “từ được cấu tạo bởi những hình vị”. Theo công trình này, thì “từ tiếng Việt được cấu tạo bởi những hình vị tiếng Việt”.

Từ tiếng Việt ở đây bao gồm: từ đơn, từ ghép, từ láy và từ ngẫu hợp.

Xuất phát từ nhu cầu xử lý tự động ngữ liệu tiếng Việt bằng máy tính, **PGS.TS. Đinh Điền** đã đề nghị cách thức hình thức hoá các quan niệm về hình vị tiếng Việt và từ tiếng Việt nói trên trong công trình như sau:

* Do “hình vị tiếng Việt” cũng chính là từ *chính tả* (từng chữ độc lập), nên việc hình thức hoá rất đơn giản, không cần đặt ra. Trong ngữ liệu tiếng Việt cũng như tiếng Anh, đơn vị cơ bản được lưu cũng chính là từ chính tả này. Tuy nhiên, nếu chỉ lưu trữ ở cấp độ hình vị như vậy, thì lượng thông tin trong kho ngữ liệu sẽ rất hạn chế và chúng ta sẽ không thể khai thác hiệu quả vốn có của nó được.
* Để lưu trữ thông tin về ranh giới từ tiếng Việt, chúng tôi sử dụng khái niệm *từ từ điển học*. *Từ từ điển học* ở đây được định nghĩa là “*những đơn vị mà căn cứ vào đặc điểm ý nghĩa của nó phải xếp riêng trong từ điển và có đánh dấu đây là đơn vị từ của ngôn ngữ*”. Việc chọn lựa những từ nào sẽ đưa vào từ điển là hoàn toàn do các nhà ngôn ngữ hay người xây dựng kho ngữ liệu quyết định, dựa theo quan điểm về từ đã nêu trên. Trong đồ án này chúng tôi sử dụng từ điển tiếng Việt của công trình của **GS Hoàng Phê.**

Do có nhiều thuật ngữ về từ” khác nhau (từ chính tả, từ từ điển học,... ), vì vậy, từ đây trở về sau, thuật ngữ “từ” được sử dụng trong luận văn được quy ước là để chỉ “*từ từ điển*”.

### Khái niệm văn bản

Trong ngôn ngữ (language), văn bản là 1 thuật ngữ rộng nói về 1 thứ gì đó mà chứa các từ ngữ diễn đạt 1 sự việc.

Trong ngôn ngữ học (linguistisc), văn bản là 1 hoạt động giao tiếp, thi hành 7 nguyên tắc cấu thành cơ bản và 3 nguyên tắc điều khiển của văn bản học. Cả tiếng nói, ngôn ngữ viết hay ngôn ngữ thông thường đều có thể xem như văn bản trong ngôn ngữ học.

Trong lý thuyết văn học, văn bản là 1 đối tượng (object) được nghiên cứu, dù nó là 1 cuốn tiểu thuyết, 1 bài thơ, 1 vở phim, 1 mẫu quảng cáo hay bất cứ thứ gì có thành phần thuộc về ký hiệu. Cách dùng rộng rãi thuật ngữ này được bát nguồn từ sự xuất hiện của ký hiệu những năm 1960 và được củng cố vững chắc bằng những nghiên cứu văn hoá sau đó trong những năm 1980.

Trong truyền thông các thiết bị di động, văn bản (hay tin nhắn văn bản) là 1 đoạn tin nhắn số hoá ngắn giữa những thiết bị.

Trong tin học, văn bản liên hệ đến dữ liệu ký tự (character data), hay đến 1 trong những thành phần của chương trình trong bộ nhớ.

Trong học thuật, văn bản thường được dùng như 1 hình thức viết tắt của sách giáo khoa.

## LỖI CHÍNH TẢ

Trước hết cần phải hiểu chính tả là gì. Chính tả được hiểu là “phép viết đúng” hoặc “lối viết hợp với chuẩn”. Nói cách khác, chính tả là việc tiêu chuẩn hóa chữ viết của một ngôn ngữ. Yêu cầu cơ bản của chính tả là phải thống nhất cách viết cụ thể trên phạm vi toàn quốc và trong tất cả các loại hình văn bản viết. Tiếp theo cần hiểu thế nào là lỗi chính tả: lỗi chính tả là lỗi viết sai chuẩn chính tả bao gồm các hiện tượng vi phạm các quy định chính tả về viết hoa, viết tắt, dùng số và biểu thị chữ số và hiện tượng vi phạm diện mạo ngữ âm của từ thể hiện trên chữ viết, tức chữ viết ghi sai từ, hay còn gọi là lỗi âm vị. Lỗi âm vị trong tiếng Việt thường thể hiện qua các dạng: lỗi âm vị âm đoạn tính và lỗi âm vị siêu âm đoạn tính. Lỗi âm vị âm đoạn tính bao gồm, lỗi sai về phụ âm đầu, âm đệm, âm chính, âm cuối. Lỗi âm vị siêu đoạn tính chính là hiện tượng viết sai thanh điệu.

### Các nguyên nhân gây ra lỗi chính tả

Có rất nhiều nguyên nhân dẫn tới thực trạng này nhưng có thể quy về một số nguyên nhân chính sau đây:

1) Thứ nhất là do không nắm vững chính tự. Ví dụ, lẽ ra phải viết là ngành thì lại viết là nghành. Điều này có nguyên nhân sâu xa từ một số bất hợp lí của chữ quốc ngữ. Sự bất hợp lí này thể hiện như sau: không đảm bảo sự tương ứng một đối một giữa âm và chữ. Chẳng hạn, âm |k| có 3 cách ghi là c, k, q; con chữ g ghi âm |z| và âm |γ|. Có những nhóm hai, ba con chữ để ghi một âm vị:ph, ngh. Điều này làm người nghe lúng túng vì tại sao cùng đọc là |k| nhưng lúc thì viết là c, lúc thì viết là k, lúc lại viết là q, cùng đọc là /ŋ/ mà lúc viết là ng lúc lại viết là ngh. Đã có nhiều ý kiến đề nghị khắc phục những bất hợp lí này nhưng cho đến nay vì nhiều nguyên nhân khác nhau nó vẫn tồn tại.

2) Thứ hai là do không hiểu nghĩa. Tuy chính tả tiếng Việt là chính tả ngữ âm nhưng trên thực tế, muốn viết đúng, nhiều trường hợp phải nắm được ngữ nghĩa. Ví dụ: lẽ ra phải viết là giành (với nghĩa là tranh) thì lại viết là dành (với nghĩa là giữ lại để sau này dùng hoặc để riêng cho ai, cho việc gì) và ngược lại; lẽ ra phải viết là tham quan (tham là tham gia, tham dự, tìm tòi, nghiên cứu, tìm hiểu, quan là nhìn trực tiếp một cách kĩ lưỡng, tỉ mỉ, sâu sắc) thì lại viết là thăm quan; lẽ ra phải viết là khúc chiết (có nghĩa là có từng đoạn, từng ý, rành mạch và gãy gọn) thì lại viết là khúc triết.

3) Thứ ba là do không cập nhật những quy định chính tả hiện hành. Chẳng hạn: trước đây do đề cao sự cân đối của chữ viết nên dấu thanh được đánh vào âm đứng giữa trong âm tiết. Ví dụ hoá được viết là hóa, thuý được viết là thúy. Nhưng hiện nay, với quy định dấu phải đánh vào âm chính thì cách viết như trên đã lạc hậu. Hoặc trước đây, tên cơ quan, tổ chức viết khác so với hiện nay. Ví dụ, trước đây viết là Trường đại học bách khoa Hà nội, còn hiện nay viết là Trường Đại học Bách khoa Hà Nội. Do không cập nhật điều đó nên nhiều người đã viết theo quy định cũ dẫn đến sai chính tả.

4) Thứ tư là do ảnh hưởng của cách phát âm địa phương. Ví dụ phương ngữ Bắc Bộ không có ba âm quặt lưỡi |ƫ| |ş| |ȥ| vì thế nhiều người gặp khó khăn khi phải viết các từ có chứa những phụ âm đầu ch – tr, r – d – gi, s – x. Người nói phương ngữ Bắc Trung Bộ lại nhầm giữa dấu hỏi (ˀ) và dấu ngã (~). Vì thế họ rất lúng túng khi gặp những từ có dấu hỏi và dấu ngã. Họ sẽ không hiểu: viết là mâu thuẫn đúng hay mâu thuẩn đúng. Cũng như vậy, phương ngữ Nam Bộ lại có vấn đề khi viết các âm đầu là v hay z, âm cuối là n hay ng, c hay t, viết dấu hỏi hay dấu ngã. Một số người sẽ rất lúng túng khi gặp những từ có chứa những phụ âm đầu, phụ âm cuối và thanh điệu này.

5) Thứ năm là do sự cẩu thả của người viết. Biểu hiện của loại lỗi do nguyên nhân này rất phong phú. Ví dụ, viết hoa không theo quy tắc nào (Nguyễn thị Kim Liên, Hải phòng). Hoặc đang viết bình thường lại viết chữ to hơn nên vô tình cũng mắc lỗi viết hoa bừa bãi (Đây là ngày thứ hai tôi ở Hà Nội.) Hoặc sau dấu chấm không viết hoa. Hoặc hường lại viết là hươǹg.

6) Thứ sáu là do ảnh hưởng của ngôn ngữ mạng. Ngôn ngữ mạng phù hợp với nhu cầu muốn giao tiếp nhanh, muốn thể hiện cá tính và sự cập nhật về công nghệ hiện đại của một số người, phần lớn là giới trẻ. Tuy nhiên, trong giao tiếp có nghi thức việc sử dụng ngôn ngữ này không phù hợp và khi viết sử dụng ngôn ngữ mạng sẽ bị coi là mắc lỗi chính tả. Ví dụ cần phải viết: ạ thì lại viết ah, ừ thì lại viết uh, được thì lại viết đk, trong thì lại viết (.)…

### Phân loại một số lỗi chính tả

Lỗi nhận thức: hay xảy ra khi sử dụng các từ đồng âm, gần âm. Ví dụ: dỗ tổ trong khi đáng ra phải là giỗ tổ.

Lỗi viết tắt: vô ý tạo ra hay nhiều từ không có nghĩa. Ví dụ: “kh” tắt cho từ không, "hc" tắt cho học.

Lỗi dùng từ lóng: là một từ ngữ không chính thức của một ngôn ngữ, thường được sử dụng trong đời sống thường ngày. Ví dụ “maj”, “thj”, ...

### Phát hiện lỗi chính tả

Với thời đại phát triển công nghệ như hiện nay, trên các trang mạng đã và đang có rất nhiều ứng dụng, phần mềm phát hiện lỗi chính tả.

### Sửa lỗi chính tả

Giáo dục ý thức viết đúng chính tả cho người dân. Cần phải làm cho mọi người hiểu rằng viết đúng chính tả không chỉ thể hiện trình độ văn hóa mà còn thể hiện ý thức tôn trọng cộng đồng, lòng yêu quý đối với tiếng Việt của người viết. Còn viết sai chính tả ảnh hưởng nghiêm trọng tới giao tiếp của từng người dân, của toàn xã hội và rất nhiều trường hợp ảnh hưởng tới quốc gia, dân tộc.

Tuyên truyền, phổ cập các chuẩn mực chính tả rộng rãi trong cộng đồng sử dụng tiếng Việt bằng các con đường khác nhau như qua các phương tiện thông tin đại chúng, qua nhà trường.

Duy trì các biện pháp giúp người sử dụng tiếng Việt viết đúng chính tả. Ví dụ:Văn bản của các cơ quan, tổ chứcvà văn bản trên các phương tiện thông tin đại chúng phải tuyệt đối tuân thủ các quy định về chính tả để người dân coi đó là các văn bản mẫu và làm theo; Duy trì mục dọn vườn trên đài truyền hình, báo chí giúp người dân nâng cao kĩ năng chính tả.

Có chính sách phát triển ngôn ngữ phù hợp trong bối cảnh tiếng Việt có nhiều sự biến đổi trước những biến động của thế giới. (Chẳng hạn chọn cách ứng xử phù hợp với ngôn ngữ mạng. Trong giao tiếp có nghi thức như khi làm các văn bản giấy tờ, trong học tập… không được sử dụng ngôn ngữ mạng. Giáo dục cho cá nhân ý thức rõ khi nào có thể sử dụng ngôn ngữ mạng, khi nào không được sử dụng ngôn ngữ mạng).

### Một số phương pháp kiểm lỗi chính tả

Chúng ta có thể tạm chỉ ra hai phương pháp chính đó là dựa vào luật và dựa vào thống kê. Các phương pháp dựa theo luật có ưu điểm là không tốn quá nhiều tài nguyên của thiết bị, tuy nhiên các chương trình sử dụng phương pháp này không có khả năng học, và hiện tại thì kết quả cũng chưa cao đối với nhiều ngôn ngữ.

Có khá nhiều phương pháp dựa vào thống kê khác nhau đã được đưa ra để kiểm lỗi chính tả tiếng Anh. Trong phạm vi giới hạn của đồ án, em xin chỉ liệt kê một vài phương pháp đánh giá là nổi bật.

## MÔ HÌNH NGÔN NGỮ N-GRAM

### Giới thiệu

Nhiệm vụ của mô hình ngôn ngữ là cho biết xác suất của một câu w1w2...wm  là bao nhiêu.

Theo công thức Bayes:

*P(AB) = P(B|A) \* P(A)*

thì:

*P(w1w2…wm) = P(w1) \* P(w2|w1) \* P(w3|w1w2) \*…\* P(wm|w1w2…wm-1)*

Theo công thức này, mô hình ngôn ngữ cần phải có một lượng bộ nhớ vô cùng lớn để có thể lưu hết xác suất của tất cả các chuỗi độ dài nhỏ hơn *m*. Rõ ràng, điều này là không thể khi *m* là độ dài của các văn bản ngôn ngữ tự nhiên (*m* có thể tiến tới vô cùng). Để có thể tính được xác suất của văn bản với lượng bộ nhớ chấp nhận được, ta sử dụng xấp xỉ Markov bậc *n*:

*P(wm|w1,w2,…, wm-1) = P(wm|wm-n,wn-m+1, …,wm-1)*

Nếu áp dụng xấp xỉ Markov, xác suất xuất hiện của một từ *(wm)* được coi như chỉ phụ thuộc vào *n* từ đứng liền trước nó *(wm-nwm-n+1…wm-1)* chứ không phải phụ thuộc vào toàn bộ dãy từ đứng trước *(w1w2…wm-1)*. Như vậy, công thức tính xác suất văn bản được tính lại theo công thức:

*P(w1w2…wm) = P(w1)\*P(w2|w1)\*P(w3|w1w2)\*…\* P(wm-1|wm-n-1wm-n …wm-2)\* P(wm|wm-nwm-n+1…wm-1)*

Với công thức này, ta có thể xây dựng mô hình ngôn ngữ dựa trên việc thống kê các cụm có ít hơn *n+1* từ. Mô hình ngôn ngữ này gọi là mô hình ngôn ngữ N-gram.

Một cụm N-gram là 1 dãy con gồm *n* phần tử liên tiếp nhau của 1 dãy các phần tử cho trước.

### Công thức tính “xác suất thô”

Gọi C(w...ww) là tần số xuất hiện của cụm w...ww trong tập văn bản huấn luyện.

Gọi *P(w|w...w)* là xác suất *w* đi sau cụm *w..ww.*

Ta có công thức tính xác suất như sau:

*P(w|w...w) =*

Dễ thấy,  *C(w..ww)* chính là tần số xuất hiện của cụm w...w trong văn bản huấn luyện. Do đó công thức trên viết lại thành:

*P(w|w...w) =*

Tỉ lệ ở vế phải còn gọi là tỉ lệ tần số. Cách tính xác suất dựa vào tỉ lệ tần số còn gọi là ước lượng xác suất cực đại. Cũng có thể gọi đây là công thức tính “*xác suất thô*” để phân biệt với các cách tính xác suất theo các thuật toán hiệu quả hơn.

### Tính xác suất dựa trên N-gram

Giả sử ta muốn dự đoán âm tiết tiếp theo *s5* dựa trên thông tin ngữ cảnh trước đó của các âm tiết *s1s2s3s4*. Chúng ta cần tính xác suất của sự kiện âm tiết tiếp theo là *s5* khi các âm tiết trước nó là *s1s2s3s4*, hay nói cách khác ta cần tìm xác suất:

*P(s1,...s5)*

Tổng quát với *sn*, ta cần tìm:

*P(s1,...,sn)*

Theo quy tắc dây chuyền (chain rule) ta có:

*P(s1,...,sn) = P(s1).P(s2|s1)...P(sn|s1,...,sn-1)*

Áp dụng công thức trên cho một câu có *n* âm tiết, ta sẽ tìm ra được tổ hợp âm tiết “tốt nhất” cho câu, và tổ hợp này được coi là đúng chính tả. Tuy nhiên, việc tính xác suất đó trong thực tế là điều gần như không thể, bởi chúng hiếm khi xảy ra và nếu có thể tính được thì cũng cần có một lượng dữ liệu rất lớn được dùng để tính toán.

Để giải quyết vấn đề trên, chúng ta có thể xấp xỉ những xác suất đó bằng một giá trị *n-gram* nhất định với *n* cho trước.

Với n = 1, ta có 1-gram (Unigram):

*P(sn|s1,...,sn-1) P(sn)*

Với n = 2, ta có 2-gram (Bigram)

*P(sn|s1,...,sn-1) P(sn|sn-1)*

### N-gram đơn giản

Ở phần này, chúng ta sẽ nêu lên cách sử dụng n-gram vào việc dự đoán một từ trong tiếng Anh, từ đó sẽ suy ra cách áp dụng phương pháp cho âm tiết trong tiếng Việt.

Giả sử chúng ta có thông tin lịch sử *h* là “*its water is so transparent that*” và chúng ta muốn tính xác suất từ *w* tiếp theo sẽ là “*the*”:

*P(the| its water is so transparent that)*

Để tính xác suất trên ta có thể dựa vào tần suất xuất hiện của *w* và *h*. Ví dụ, chúng ta có thể lấy một lượng dữ liệu đầu vào (corpus) rất lớn, đếm số lần xuất hiện của *h* và số lần xuất hiện *w* đi theo sau *h*. Việc này có thể trả lời được câu hỏi: “Trong tổng số lần xuất hiện *h* có bao nhiêu lần đứng tiếp sau nó là *w*”:

*P(the|its water is so transparent that) = C(its water is so transparent that the)/C(its water is so transparent that)*

Với một corpus đủ lớn, như là internet thì chúng ta có thể đếm được các tần suất và ước lượng được xác suất nêu ở công thức trên. Tuy nhiên, chúng ta có thể dễ dàng nhận thấy rằng ngay cả dữ liệu trên internet cũng không đủ lớn để cho ta những ước lượng tốt nhất trong hầu hết các trường hợp thực tế. Bởi lẽ ngôn ngữ con người hết sức sáng tạo, nó biến đổi không ngừng và do đó mà không phải lúc nào chúng ta cũng có thể đếm được toàn bộ các câu nói. Thậm chí chỉ một sửa đổi nhỏ của một câu thôi cũng có thể khiến số lần xuất hiện của nó trên web là bằng không.

Tương tự như vậy, nếu chúng ta muốn biết xác suất xuất hiện liên kết của toàn bộ chuỗi từ *W* như “*its water is so transparent that*”, chúng ta có thể đặt ra câu hỏi “trong toàn bộ tổ hợp khác nhau có thể có của chuỗi 5 từ đó, có bao nhiêu trong số chúng có thứ tự như trên?”. Chúng ta sẽ phải đếm số lần xuất hiện của “*its water is so transparent that*” và sau đó chia chúng cho tổng số lần xuất hiện của từng từ trong số 5 từ đó. Việc này đòi hỏi một lượng tính toán không hề nhỏ.

Chính bởi lẽ đó, chúng ta cần có những phương pháp tốt hơn để ước lượng giá trị xác suất của từ *w* với thông tin lịch sử *h* cho trước, và xác suất của toàn bộ chuỗi từ *W*. Để dễ dàng hơn trong việc diễn đạt, chúng ta sẽ có một vài quy ước như sau: để biểu thị xác suất của một biến độc lập ngẫu nhiên *Xt* lấy giá trị là “*the*”, hay *P(Xt = “the”)*, chúng ta sẽ rút gọn lại là *P(the)*. Chuỗi *N* từ có thể biểu diễn dạng *w1...wn* hoặc *wn1*. Với xác suất liên kết của từng từ riêng biệt trong chuỗi: *P(X = w, Y= w2...,)* được thay bằng *P(w1,w2,...,wn).*

Để tính xác suất *P(w1,w2,...,wn)* của các chuỗi từ, ta có thể phân tích xác suất này sử dụng quy tắc dây chuyền:

*P(X1...Xn) = P(X1)P(X2|X1)P(X3|X12)...P(Xn|X1k-1)*

Áp dụng cho từ, ta có:

*P(w1n) = P(w1)P(w2|w1)P(w3|w12)...P(wk|X1k-1)*

Quy tắc dây chuyền cho ta thấy mối liên hệ giữa việc tính xác suất liên kết của một chuỗi với việc tính xác suất điều kiện của một từ với các từ cho trước. Công thức trên cho ta thấy ta có thể ước lượng xác suất liên kết của một chuỗi bằng cách nhân các xác suất điều kiện với nhau. Tuy nhiên, việc sử dụng quy tắc dây chuyền vẫn chưa giải quyết được vấn đề mà chúng ta đã đề cập ở trên. Ta không thể ước lượng xác suất bằng cách đếm số lần xuất hiện của chuỗi từ, bởi ngôn ngữ rất phong phú, từng ngữ cảnh riêng biệt đều có thể chưa bao giờ xuất hiện trước đó.

Có thể hiểu việc sử dụng mô hình n-gram là thay vì tính xác suất của từ cho trước dựa trên toàn bộ thông tin lịch sử *h*, chúng ta xấp xỉ thông tin đó chỉ bằng một vài từ cuối gần nhất.

Ví dụ với mô hình *bigram*, xấp xỉ xác suất của một từ cho trước với toàn bộ từ trước đó *P(wn|w1n-1)* bằng xác suất điều kiện của từ đứng ngay trước nó

*P(wn|wn-1).* Hay nói cách khác, thay vì tính xác suất:

*P(the|Walden Pond’s water is so transparent that)*

Chúng ta tính xấp xỉ nó bằng xác suất:

*P(the|that)*

Việc này giả thiết rằng xác suất của một từ chỉ phụ thuộc vào từ trước đó, được gọi là một giả thuyết Markov (Markov assumption). Các mô hình Markov là một lớp các mô hình xác suất giả thiết rằng chúng ta có thể dự đoán được xác suất của một đơn vị (unit) nào đó trong tương lại mà không cần phải dựa quá nhiều vào thông tin trong quá khứ. Chúng ta có thể suy ra được *bigram* (lấy thông tin của một từ trước đó), *trigram* (lấy thông tin của hai từ trước đó) cho tới *n-gram* (lấy thông tin của *n-1* từ đứng trước).

Suy ra công thức tổng quát xấp xỉ *n-gram* cho xác suất điều kiện của từ tiếp theo trong một chuỗi là:

*P(Wn|W1n-1) P(Wn|)*

Từ giả thuyết *bigram* cho xác suất của một từ độc lập, chúng ta có thể tính xác suất của một chuỗi đầy đủ bằng cách áp dụng công thức trên với *n* = với công thức quy tắc dây chuyển:

Vậy làm thế nào để ước lượng được các giá trị xác suất bigram hay n-gram này. Có một cách đơn giản và dễ dàng nhận thấy nhất để ước lượng xác suất đó là ước lượng hợp lý cực đại (Maximum likelihood estimation hay MLE). Chúng ta dùng MLE cho các tham số của một mô hình n-gram bằng cách đếm trọng corpus và chuẩn hóa chúng sao cho các giá trị của nó nằm giữ 0 và 1.

Ví dụ, để tính một giá trị xác suất bigram của một từ y với từ cho tước x đứng trước nó, chúng ta sẽ tính toán giá trị đếm bigram C(xy) và chuẩn hóa nó với tổng số tất cả các bigram khác có từ đứng trước là x:

Chúng ta có thể đơn giản hóa công thức trên, khi tổng số lần xuất hiện của tất cả các bigram bắt đầu bởi từ wn-1 cho trước phải bằng với các giá trị unigram cho từ wn-1 đó:

Với trường hợp tổng quát sử dụng MLE chi n-gram ta có công thức:

### Làm mịn (smoothing)

Có một vấn đề gặp phải trong quá trình tính toán ước lượng hợp lý cực đại, đó là vấn đề dữ liệu thưa (sparse data) xảy ra do ước lượng hợp lý cực đại dựa trên một tập dữ liệu huấn luyện riêng biệt. Với một *n-gram* bất kỳ, đôi khi chúng ta có thể có được xấp xỉ tốt của xác suất. Nhưng do bất kỳ corpus nào cũng có giới hạn nhất định, một vài từ nào đó hoàn toàn chính xác nằm trong chuỗi lại không xuất hiện trong corpus. Lượng dữ liệu bị thiếu này cho thấy *n-gram* cho bất kỳ một corpus cho trước nào cũng có một số lượng lớn trường hợp xác suất *n-gram* bằng không cần có một xác suất nào đó khác không cho nó. Hơn nữa, phương pháp ước lượng hợp lý cực đại cũng sinh ra các xấp xỉ không tốt khi các giá trị đếm khác không nhưng lại quá nhỏ, dẫn đến giá trị ước lượng xấp xỉ bằng không.

Chúng ta cần có một phương pháp giúp có được những ước lượng tốt hơn so với những tần suất thấp hay tần suất bằng không. Các giá trị đếm bằng không thậm chí còn gây ra một vấn đề nghiêm trọng hơn. Khi xử lý hiện tượng nhập nhằng cho một câu mà câu đó lại chứa một giá trị *n-gram* không hề xuất hiện trong huấn luyện thì ước lượng hợp lý cực đại xác suất của *n-gram* này cũng như toàn bộ câu sẽ có một giá trị bằng không. Điều này có nghĩa rằng để đánh giá được mô hình ngôn ngữ chúng ta cần phải thay đổi phương thức ước lượng hợp lý cực đại sao cho tất cả các giá trị *n-gram* đều phải khác không kể cả khi chúng không xuất hiện trong tập huấn luyện.

Vì những lý do nêu trên, chúng ta cần sử dụng các phương pháp làm mịn để giải quyết, giảm thiểu các vấn đề gặp phải do lượng dữ liệu có hạn. ở đồ án này sẽ sử dụng phương pháp làm mịn Laplace (Laplace smoothing, hay còn gọi là add one smoothing – thêm một).

Một cách làm mịn đơn giản là cộng thêm 1 vào tất cả các giá trị đếm trước khi chúng ta chuẩn hóa chúng thành giá trị xác xuất. Giải thuật này được gọi là làm mịn *laplace* hay luật *laplace*

Chúng ta sẽ áp dụng làm mịn *laplace* bắt đầu với các xác suất *unigram*. Ước lượng xấp xỉ cực đại của xác suất *unigram* cho từ *wi* là giá trị đếm *ci* được chuẩn hóa bởi tổng số từ vựng *N*:

Làm mịn *laplace* chỉ đơn giản là cộng một vào mỗi gía trị đếm. Và khi trong bảng từ vựng có *V* từ (hay *v* là tổng số từ có trong tập dữ liệu), mỗi từ được cộng thêm một, chúng ta cũng cần cộng thêm vào mẫu số một giá trị là *V*:

Tương tự như vậy, với các giá trị xác suất *bigram* ta có công thức làm mịn:

Trong đó *V* là số loại từ (mỗi từ khác nhau được coi là một loại) có trong tâp dữ liệu, hay *V* chính là số lượng các *unigram*. Và như vậy ta có thể phát triển tính tương tự cho các giá trị *n-gram* khác như *3-gram*, *4-gram*,...

Có một giải pháp khá hiệu quả cho việc tính toán các giá trị xác suất *n-gram* trên máy tính đó là sử dung dạng logarit hóa của các giá trị xác suất. Vì sau khi làm mịn như ở trên, tất cả các xác suất *P* đều có giá trị nằm trong khoảng 0 < P < 1. Do đó, việc logrit hóa giúp cho tránh được vấn đề giới hạn dữ liệu do giá trị xác suất quá nhỏ, nằm ngoài khả năng biểu thị của kiểu dữ liệu (underflow), đặc biệt là khi ta nhân nhiều giá trị xác suất n-gram với nhau. Vì khi giá trị quá nhỏ, máy tính sẽ coi nó bằng không và đồng thời cũng làm tăng tốc độ tính toán thay vì nhân chia các giá trị với nhau thì với logarit, chúng trở thành các phép toán cộng - trừ, một công việc dễ dàng hơn cho bộ xử lý.

Ví dụ:

p1.p2.p3.p4 = exp(log p1 + log p2 + log p3 + log p4)

### Áp dụng cho tiếng Việt

Do đặc điểm của tiếng Anh các từ ngăn cách nhau bởi khoảng trắng hoặc dấu câu. Tuy nhiên trong tiếng Việt, một từ có thể được kết hợp với nhiều âm tiết khác nhau bởi khoảng trắng. Tuy nhiên trong tiếng Việt, trật tự của các tiếng trong một từ là một trong những yếu tố quyết định ý nghĩa của từ đó, do vậy ta hoàn toàn có thể áp dụng mô hình n-gram cho tiếng Việt đơn vị là một âm tiết. Và khi đó, các giá trị biểu thức cho một từ wi ở trên có thể được hiểu như một âm tiết trong tiếng Việt.

## TÁCH TỪ TIẾNG VIỆT

### Xử lý nhập nhằng

Nhập nhằng trong tách từ được phân thành hai loại:

* Nhập nhằng chồng (Overlapping Ambiguity)
* Nhập nhằng hợp (Combination Ambiguity)

Ta gọi *D* là tập hợp các từ tiếng việt (từ điển tiếng Việt). Các trường hợp nhập nhằng trên được mô tả hình thức như sau:

* Chuỗi được gọi là nhập nhằng chồng nếu {, }  *D*
* Chuỗi được gọi là nhập nhằng hợp nếu {,}  *D*

Trong thực tế, loại nhập nhằng chồng xảy ra thường xuyên hơn loại nhập nhằng hơn, bởi vì hầu hết các tiếng của tiếng Việt đều có thể đóng vai trò là một từ đơn độc lập. Do đó, hầu hết các từ ghép đều có thể bị nhập nhằng hơn. Tuy nhiên, hầu như mọi trường hợp này đều được giải quyết tốt bằng giải thuật **Maxium Matching.** Vì thế mọi hệ thống nhận diện nhập nhằng hiện tại đều chỉ chú ý đến việc giải quyết loại nhập nhằng đầu tiên là nhập nhằng chồng.

### Tách từ tiếng Việt dùng mô hình Maximum Matching

**Maximum Matching (MM)** được xem như là phương pháp tách từ dựa trên từ điển đơn giản nhất. **MM** cố gắng so khớp với từ dài nhất có thể có trong từ điển. Đó là một thuật toán ăn tham. (Greedy Algorithms) nhưng bằng thực nghiệm đã chứng minh được rằng thuật toán này đạt được độ chính xác >90% nếu từ điển đủ lớn. Tuy nhiên, nó không thể giải quyết vấn đề nhập nhằng và không thể nhận diện được các từ chưa biết bởi vì chỉ những từ tồn tại trong từ điển mới được phân đoạn đúng

Giải quyết **MM** gồm hai giải thuật con: **FMM** (Forward Maximum Matching: so khớp cực đại theo chiều tiến) và **BMM** (Backward Maximum Matching: so khớp cực đại theo chiều lùi). Nếu chúng ta nhìn vào kết quả của **FMM** và **BMM** thì sự khác biệt này cho chúng ta biết nơi nào nhập nhằng xảy ra. Ngoài ra, **MM** là phương pháp tách từ hoàn toàn phụ thuộc vào từ điển, từ điển phải đủ lớn, đủ chính xác và độ tin cậy phải cao thì mới cho kết quả tách từ chấp nhận được. Đây cũng là nhược điểm rất lớn của phương pháp này.

Ví dụ: Người nông dân ra sức cải tiến bộ công cụ lao động của mình.

**Đầu ra FMM:**

Người#nông dân#ra sức#**cải tiến**#**bộ**#công cụ#lao động#của#mình#.

**Đầu ra BMM:**

Người#nông dân#ra sức#**cải#tiến bộ**#công cụ#lao động#của#mình#.

# MÔ HÌNH – CÀI ĐẶT

## MÔ HÌNH LSTM

### Lý do chọn LSTM

Neural Network có 2 mô hình lớn là Convolutional Neural Network (CNN) cho bài toán có input là ảnh và Recurrent neural network (RNN) cho bài toán dữ liệu dạng chuỗi (sequence). Về cơ bản nếu thấy sequence data hay time-series data mà muốn áp dụng Neural Network sẽ nghĩ ngay đến RNN. Tuy nhiên mạng RNN gặp phải một số hạn chế đó là phải thực hiện tuần tự, đạo hàm bị triệt tiêu (Vanishing gradient), bùng nổ đạo hàm (Exploding gradient). LSTM là một mạng cải tiến của RNN nhằm giải quyết phần nào các vấn đề mà mạng RNN gặp phải. Rất nhiều các bài toán học máy sử dụng LSTM đem lại kết quả rất đáng chú ý so với việc sử dụng các phương pháp khác. Dữ liệu văn bản mà ta thu thập được là một dạng dữ liệu kiểu chuỗi tuần tự vì vậy khá phù hợp khi sử dụng LSTM huấn luyện dữ liệu mà nhóm đã thu thập được.

### Cơ sở lý thuyết

#### Mạng hồi quy RNN

Recurrent Neural Networks (RNN) - một cách dịch đại khái sang tiếng Việt là mạng thần kinh hồi quy. Sở dĩ, nó được gọi là Recurrent (hồi quy) vì nó thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng ghi nhớ các thông tin được tính toán trước đó (như là việc nhìn vào các bối cảnh phía trước rồi đưa ra các hành động tiếp theo). RNN là mô hình mạng thần kinh được thiết kế để nhận dữ liệu tuần tự hoặc kiểu dữ liệu time series (chuỗi thời gian).

Đầu tiên, trước khi tìm hiểu về RNN, chúng ta hãy cùng xem cách mạng Neural thông thường lan truyền như hình dưới đây:

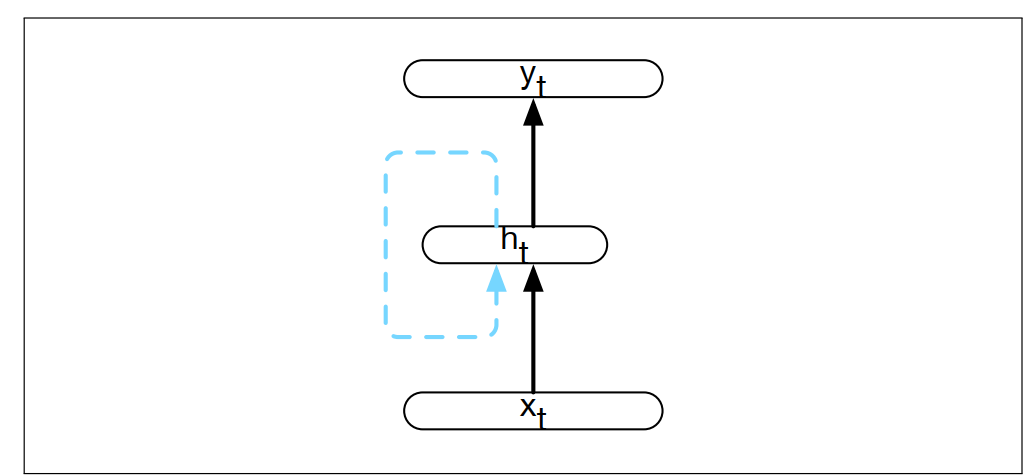


Hình 3.1 Kiến trúc mạng Neuron thông thường

Với mạng Neural Network thông thường, các nút màu đỏ là giá trị đầu vào còn màu xanh lá cây là giá trị đầu ra. Ví dụ trong bài toán về phân loại hình ảnh dùng Convolution Neural Network đầu vào sẽ là các pixel và đầu ra là các ảnh cần phân loại. Theo cách này, ta có thể thiết thế một mạng Neural Network có độ dài đầu vào và đầu ra cố định.

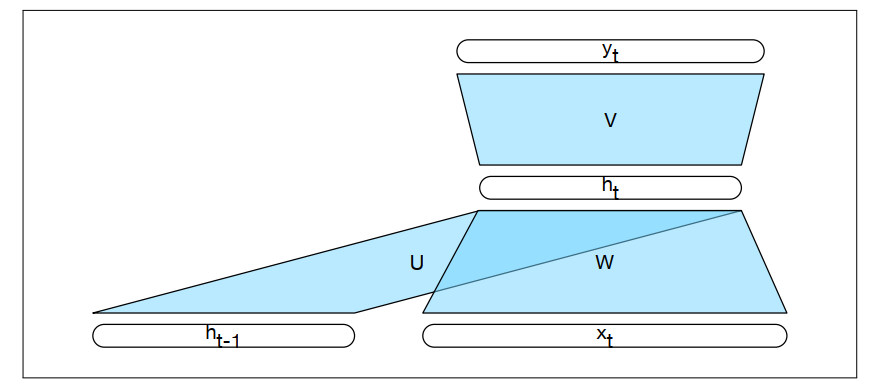
Tuy nhiên, các dữ liệu trong thực tế thường không có chiều dài cố định. Ví dụ: khi ta viết một Status trên Facebook, nó không cố định chiều dài số lượng từ mà ta viết. Chính vì vậy mà mạng RNN được thiết kế để xử lý các dữ liệu có đầu vào và đầu ra thay đổi.

Về cơ bản, một mô hình đơn giản của RNN có dạng như sau:



Hình 3.2 Cấu trúc đơn giản hóa mạng RNN

Điểm khác biệt lớn nhất của một mạng Neuron thông thường với RNN chính là phần mũi tên màu xanh lặp lại ở trạng thái ẩn ht. Đường mũi tên xanh này ngụ ý rằng tại thời điểm *t* bất kì, đầu vào của mạng sẽ là tổ hợp của trạng thái ẩn ht trước đó và vector đầu vào xt tại thời điểm hiện tại. Ta có thể coi trạng thái ẩn **ht** là "bộ nhớ" của mạng. **ht**sẽ lưu trữ các thông tin đã được tính toán ở phía trước. Đầu ra ở bước **yt**chỉ được tính dựa trên bộ nhớ ở bước thứ *t*.

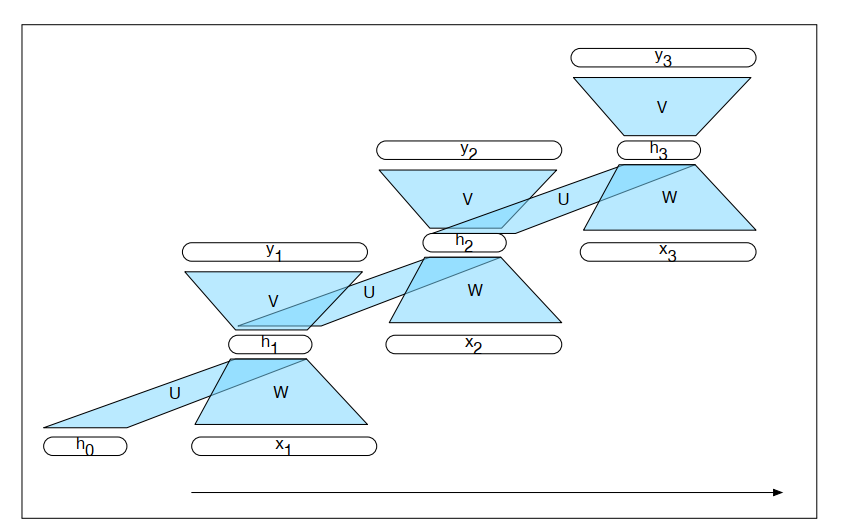


Hình 3.3 Mô tả RNN dưới dạng tham số

Bản chất hình 3.3 và 3.2 là như nhau. Sự khác biệt duy nhất là ở hình 3.3, ta đã có thêm ma trận trọng số **U**, ma trận **U** có khả năng kết nối thông tin từ trạng thái trước đó ht-1 với trạng thái hiện tại ht. Và đây chính là điểm mấu chốt của RNN, với **U**, RNN có thể lưu trữ được thông tin ngữ cảnh trước để làm đầu vào cho phép tính hiện tại.

#### Lan truyền thuận trong mạng hồi quy RNN

Lan truyền thuận trong RNN đơn giản là phép ánh xạ từ mỗi chuỗi đầu vào đến một chuỗi đầu ra tương ứng (giống hệt mạng Neuron thông thường).



Hình 3.4 Minh họa các lớp trong mạng RNN

Để dễ hình dung, chúng ta sẽ khai triển (unfold) mạng RNN như trên. Ví dụ, ta có chuỗi đầu vào là “Tôi đi học” thì mạng RNN sẽ triển khai thành 3 layer-Neural Network. Trong đó, mỗi Layer tương ứng với 1 từ. Lúc này, việc tính toán bên trong RNN sẽ xảy ra như sau:

* **xt** là đầu vào của bước *t*. Ví dụ: **x2**là một one-hot vector tương ứng với từ thứ hai là "đi".
* **ht** là trạng thái ẩn ở bước thứ *t*. Nó là “memory (bộ nhớ)” của mạng. **ht** là một tổ hợp tuyến tính của trọng số **U** với bộ nhớ **ht-1** ở bước trước đó và đầu vào **xt** hiện tại với trọng số **W**. Hàm g thường là một hàm phi tuyến như tanh hoặc ReLU. Đối với bước đầu tiên, giá trị **h0**sẽ là một vector 0.

***ht*** *= g (****Uht-1******+ Wxt****)*

* **yt** là output (đầu ra) ở bước *t*. Ví dụ: nếu ta muốn dự đoán từ tiếp theo trong câu thì **yt**là một vector xác suất các từ trong danh sách từ vựng.

***yt*** *= softmax (****Vht****)*.

Nếu ta gọi số chiều của lớp đầu vào, lớp ẩn và lớp đầu ra lần lượt là din, dh và dout. Bảng dưới đây mô tả số chiều của các tham số trong mạng RNN

|  |  |  |
| --- | --- | --- |
|  |  | Số chiều |
| Input | **x** | din |
| Hidden state | **h** | dh |
| Output | **y** | dout |
| Tham số học | **W** | dh din |
| **U** | dh dh |
| **V** | dout dh |

Mã giả dưới đây mô tả quy trình lan truyền thuận của mạng RNN.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 3.5 Mã giả thuật toán lan truyền thuận trong mạng RNN

Hàm FORWARDRNN nhận tham số câu đầu vào là vector **x**. Ban đầu lớp ẩn **h** được khởi tạo là 1 vector 0. Vòng lặp chạy với mỗi từ **x**i ta tính được giá trị lớp ẩn **h**i và **y**i mới, trong khi bộ tham số là (**U, V, W**) là không đổi trong tất cả các bước. Điều này nói lên rằng ta thực hiện cùng 1 nhiệm vụ ở tất cả các bước. Và điều này làm giảm đáng kể số lượng tham số cần học trong mô hình.

#### Huấn luyện mạng hồi quy RNN cho bài toán sửa lỗi chính tả

Về cơ bản, các bài toán về RNN có thể phân loại như sau:

Ảnh có chứa văn bản, dụng cụ vệ sinh cá nhân, mẫu họa, mỹ phẩm

Mô tả được tạo tự động

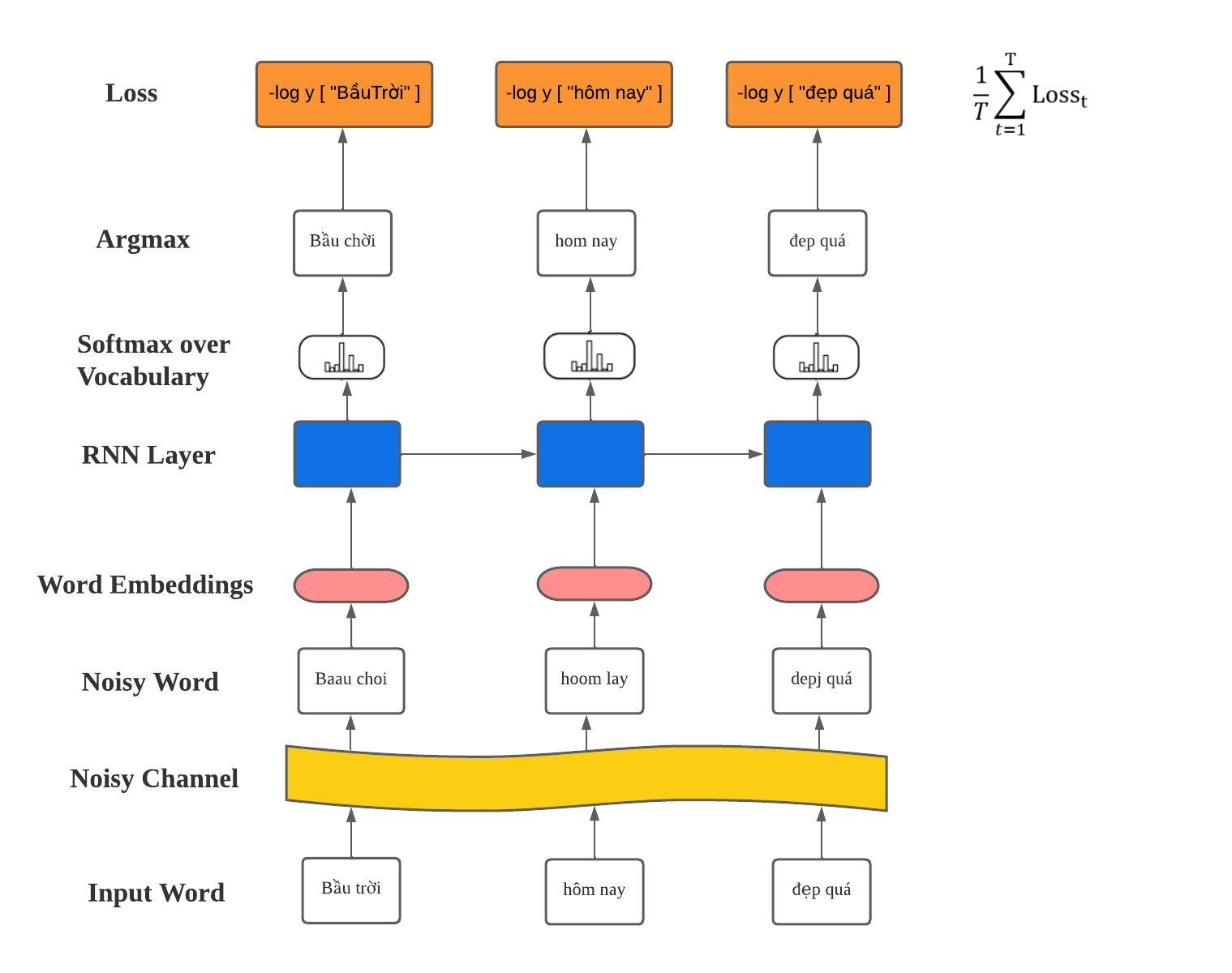
Hình 3.6 Minh họa mô hình được sử dụng trong mạng RNN

Trong bài toán sửa lỗi chính tả nhóm thực hiện lần này, mô hình được sử dụng sẽ là many-to-many. Tức là ta có một chuỗi input và đầu ra sẽ là một chuỗi output tương ứng được dự đoán là kết quả sau khi sửa lỗi chính tả. Với các mô hình còn lại, ta có thể đọc thêm tại đây [1]

Sau đây là mô tả quy trình huấn luyện mạng RNN cho bài toán sửa lỗi chính tả.

* Input: Chuỗi đầu vào **X** = [**x**1, **x**2, …, **x**n] với **x**t là một bi-gram (chương 2.1). Các bigram được biểu bởi một one-hot vector có số chiều *|V|* 1, với *|V|* là số từ trong tập từ điển.
* Output: Chuỗi câu dự đoán vector **Y\*** = [, , … ] với là **yt**là một vector xác suất các từ trong danh sách từ vựng. **yt** có số chiều *|V|* 1.

Lưu ý rằng, các bi-gram **x**t từ văn bản gốc sẽ được đi qua một kênh làm nhiễu. Mục đích của việc này là ta muốn huấn luyện mô hình có khả năng dự đoán (predict) được những từ **x**t gốc thực sự (target) sau khi các **x**t gốc đầu vào (input) đã bị làm nhiễu (hay nói nôm na là những từ bị cố tình làm cho sai chính tả). Hình dưới đây sẽ mô tả chi tiết quá trình huấn luyện này.



Hình 3.7 Quy trình huấn luyện mạng RNN cho bài toán sửa lỗi chính tả

*Ban đầu*, các bi-gram được đưa qua kênh làm nhiễu (Noisy channel) và được ánh xạ thành các noisy word. Ta tiến hành mã hóa từng bi-gram thành các vector (Word Embeddings). *Tiếp theo*, tại thời điểm *t*, các vector bi-gram **x**t này lần lượt được đưa vào lớp RNN (ô màu xanh trên hình). Các trạng thái ẩn **h**t trong lớp RNN được tính theo công thức sau:

***ht*** *= g (****Uht-1******+ Wxt****)*

Với:

* **U: ma trận trọng số kết nối trạng thái ẩn ht-1 sang ht**
* **W: ma trận trọng số kết nối từ đầu vào xt đến lớp ẩn ht**

Đầu ra của các lớp ẩn **h**t sau đó được đưa qua hàm softmax để sinh ra một vector xác suất các từ trong danh sách từ vựng:

***yt*** *= softmax (****Vht****)*.

Với:

* **h**t: lớp ẩn thứ t
* **V**: ma trận trọng số kết nối lớp ẩn **h**t với đầu ra hàm softmax **y**t

Giả sử vector **y**t có xác suất là [0.8, 0.2, 0.2] ứng với các từ [Bầu chời, Bầu zời, Bâu trời]. Ta sẽ lấy giá trị max trong vector **y**t, tức là 0.8 tương ứng với từ “Bầu chời” để làm đầu ra đự doán cho mô hình tại thời điểm *t*.

*Và cuối cùng*, ta cần một hàm mục tiêu để đánh giá độ tốt của mô hình sau mỗi lần dự đoán, cross-entropy sẽ là hàm mục tiêu được chúng tôi dùng ở đây. Nếu ta có một vector dự đoán xác suất đầu ra và vector **y**t chứa các giá trị nhãn thực sự (các nhãn này chính là mục tiêu để mô hình học), thì tại thời điểm *t* ta có hàm đánh giá Losst như sau:

*Losst =*

Hàm Losst này được hiểu là là tổng của các tích giữa các phần tử wt của vector **y**t và các phần tử wt của vector dự đoán . Ví dụ, ta có vector dự đoán = [0.8, 0.2, 0.2] và vector chứa các giá trị nhãn thực sự **y**t = [1, 0, 0], thì ta có Losstại thời điểm *t* như sau:

*Losst = 1 \* log(0.8) + 0 0.2) + 0 log(0.2)*

Chúng tôi coi bài toán sửa lỗi chính tả thuộc bài toán dự đoán từ trong mô hình ngôn ngữ (**2.3**), tức là ta chỉ quan tâm tới từ hiện tại được dự đoán để sửa lỗi chính tả sau khi đã quan sát được các từ trước đó. Trong mô hình ngôn ngữ thì vector đầu ra ***y*** chứa các xác suẩt của từ được dự đoán. Chính vì vậy, ta có thể lược giản hàm Losst như sau:

*Losst =*

Với:

* Wt: từ đúng thật sự tại thời điểm *t*
* : vector chứa xác suất các từ được dự đoán là sửa lỗi chính tả

Và cuối cùng, hàm đánh giá mất mát của cả mô hình sẽ là trung bình của từng hàm Loss tại thời điểm *t*:

Mục tiêu của ta là cực tiểu hóa hàm L bằng cách sử dụng cơ chế lan truyền ngược, trái ngược với cơ chế lan truyện thuận như ở **3.1.2.2**. Do ngày nay, hầu như các công cụ hỗ trợ tính toán đã cung cấp khả năng xử lý cơ chế này, nên với giới hạn của đồ án, chúng tôi xin trích dẫn tham khảo chứng minh chi tiết cơ chế này tại đây [2].

Trong thực tế, mạng RNN không thể lưu trữ được quá nhiều thông tin [3], và mạng LSTM ở **3.1.3** sẽ khắc phục được điều này.

#### Mạng RNN hai chiều (Bidirectional RNN)

Trong một mạng hồi quy đơn giản, trạng thái ẩn ở một thời điểm nhất định *t* đại diện cho mọi thứ mà mạng biết về trình tự cho đến thời điểm đó trong trình tự. Trạng thái ẩn tại thời điểm *t* là kết quả của một hàm của các đầu vào từ lúc khởi động cho đến khoảng thời gian *t*. Chúng ta có thể coi đây là bối cảnh của mạng ở phía bên trái thời điểm hiện tại.

*htf = RNNforward(x1t)*

*htf* tương ứng với trạng thái ẩn thông thường tại thời điểm *t*, và đại diện cho mọi thứ mà mạng đã thu thập được từ trình tự đến thời điểm đó. Trong nhiều ứng dụng, chúng ta có quyền truy cập vào toàn bộ chuỗi đầu vào cùng một lúc. Một câu hỏi được đặt ra là liệu chúng ta có thể tận dụng ngữ cảnh ở bên phải của đầu vào hiện tại? Một cách để khôi phục thông tin là huấn luyện RNN trên một trình tự đầu vào ngược lại, áp dụng đúng loại mạng mà chúng ta đang nghiên cứu. Với cách tiếp cận này, trạng thái ẩn tại thời điểm *t* hiện tại đại diện cho thông tin về trình tự ở bên phải của đầu vào hiện tại.

*htb = RNNbackward(xtn)*

Trạng thái ẩn *htb* đại diện cho tất cả thông tin chúng ta đã xác minh về chuỗi từ *t* đến điểm cuối của chuỗi

Kết hợp giữa hai loại mạng: mạng tiến và mạng lùi dẫn đến mạng RNN hai chiều. Một Bi-RNN bao gồm hai RNN độc lập, trong đó một mạng đầu vào của nó sẽ được xử lý từ đầu đến cuối, và mạng còn lại sẽ được xử lý từ cuối đến đầu. Sau đó chúng ta kết hợp các đầu ra của hai mạng thành một biểu diễn duy nhất nắm bắt cả bối cảnh bên trái và bên phải của đầu vào tại mỗi thời điểm.

*ht* = *htf htb*

Hình dưới minh hoạ một mạng hai chiều trong đó các đầu ra của mạng tiến và mạng lùi được kết nối. Cách đơn giản khác để kết nối giữa ngữ cảnh tiến và lùi bao gồm phép cộng hoặc phép nhân thông thường. Đầu ra ở mỗi bước trong khoảng thời gian do đó nắm bắt thông tin ở bên trái và bên phải của đầu vào hiện tại. Trong các ứng dụng gán nhãn theo trình tự, các đầu ra được nối này có thể đóng vai trò là cơ sở cho một quyết định gán nhãn địa phương.

RNN hai chiều cũng đã được chứng minh là khá hiệu quả trong bài toán phân loại chuỗi. Chúng ta sẽ chỉ cần kết hợp trạng thái ẩn từ chiều tiến và chiều lùi, sử dụng nó làm đầu vào để theo dõi quá trình xử lý. Một lần nữa, kết nối là một cách tiếp cận phổ biến để kết hợp hai kết quả đầu ra nhưng tính tổng, nhân hoặc tính trung bình theo phần tử cũng được sử dụng.

Diagram

Description automatically generated

*Hình 3.8 Một mạng RNN hai chiều. Chia mô hình huấn luyện thành chiều tiến và chiều lùi với đầu ra của mỗi mô hình tại mỗi thời điểm kết nối để thể hiện trạng thái của các vấn đề tại thời điểm đó. Hộp bao quanh mạng tiến và lùi nhấn mạnh bản chất modul của kiến trúc này*

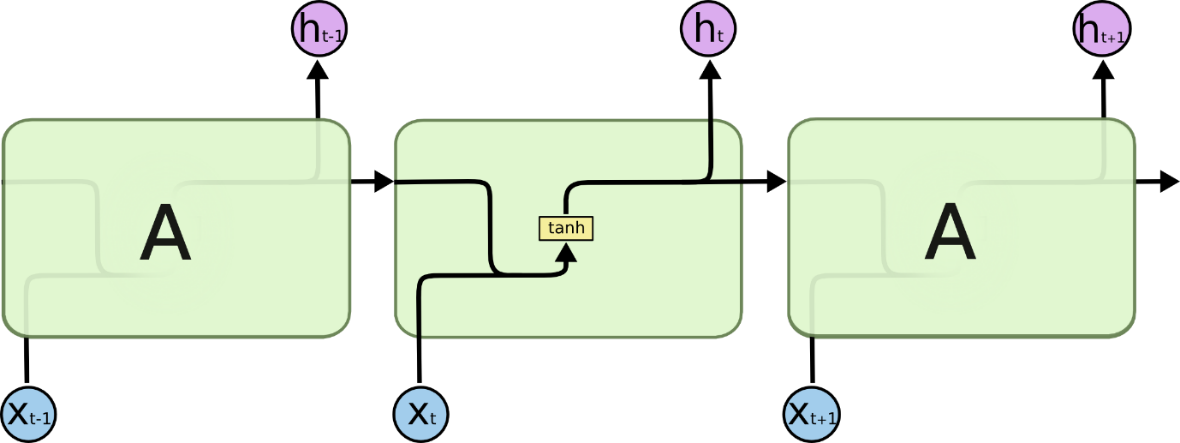
### Mạng long-short term memory

Trong phần lí thuyết này được chúng tôi dịch chủ yếu từ [6].

Mạng bộ nhớ dài-ngắn (Long Short-Term Memory networks) thường được gọi là LSTM - là một biến thể của RNN, nó có khả năng học được các phụ thuộc xa. LSTM được giới thiệu bởi **Hochreiter & Schmidhuber** và sau đó đã được cải tiến và phổ biến bởi rất nhiều người trong ngành. Chúng hoạt động cực kì hiệu quả trên nhiều bài toán khác nhau nên dần đã trở nên phổ biến như hiện nay.

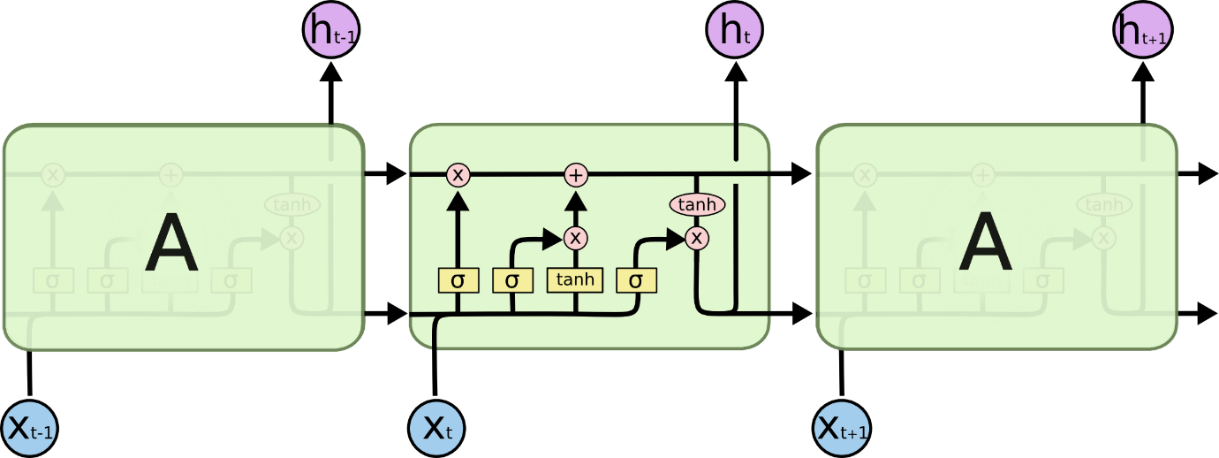
LSTM được thiết kế để tránh vấn đề phụ thuộc xa (long-term dependency). Việc nhớ thông tin trong suốt thời gian dài là đặc tính mặc định của chúng, chứ ta không cần phải huấn luyện nó để có thể nhớ được. Tức là ngay nội tại của nó đã có thể ghi nhớ được mà không cần bất kì can thiệp nào.

Mọi mạng hồi quy đều có dạng là một chuỗi các mô-đun lặp đi lặp lại của mạng nơ-ron. Với mạng RNN chuẩn, các mô-dun này có cấu trúc rất đơn giản, thường là một tầng tanh.

[](https://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-SimpleRNN.png)

Hình 3.6 Mô-đun trong RNN tiêu chuẩn chứa một layer

LSTM cũng có kiến trúc dạng chuỗi như vậy, nhưng các mô-đun trong nó có cấu trúc khác với mạng RNN chuẩn. Thay vì chỉ có một tầng mạng nơ-ron, chúng có tới 4 tầng tương tác với nhau một cách rất đặc biệt.

[](https://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-chain.png)

Hình 3.7 Mô-đun trong LSTM chứa 4 layer

Một số ký hiệu trong LSTM:

[Diagram

Description automatically generated](https://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM2-notation.png)

Ở sơ đồ trên, mỗi một đường mang một véc-tơ từ đầu ra của một nút tới đầu vào của một nút khác. Các hình trong màu hồng biểu diễn các phép toán như phép cộng véc-tơ chẳng hạn, còn các ô màu vàng được sử dụng để học trong các từng mạng nơ-ron. Các đường hợp nhau kí hiệu việc kết hợp, còn các đường rẽ nhánh ám chỉ nội dung của nó được sao chép và chuyển tới các nơi khác nhau.

a, Ý tưởng cốt lõi của LSTM

Chìa khóa của LSTM là trạng thái tế bào (cell state) - chính đường chạy thông ngang phía trên của sơ đồ hình vẽ. Trạng thái tế bào là một dạng giống như băng truyền. Nó chạy xuyên suốt tất cả các mắt xích (các nút mạng) và chỉ tương tác tuyến tính đôi chút. Vì vậy mà các thông tin có thể dễ dàng truyền đi thông suốt mà không sợ bị thay đổi.

Ảnh có chứa văn bản, đồng hồ

Mô tả được tạo tự động

Hình 3.8 Cell state (C)

LSTM có khả năng bỏ đi hoặc thêm vào các thông tin cần thiết cho trạng thái tế báo, chúng được điều chỉnh cẩn thận bởi các nhóm được gọi là cổng (gate). Các cổng là nơi sàng lọc thông tin đi qua nó, chúng được kết hợp bởi một tầng mạng sigmoid và một phép nhân.



Hình 3.9 Gates

Tầng sigmoid sẽ cho đầu ra là một số trong khoản [0, 1], mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là 0 thì có nghĩa là không cho thông tin nào qua cả, còn khi là 1 thì có nghĩa là cho tất cả các thông tin đi qua nó. Một LSTM gồm có 3 cổng như vậy để duy trì và điều hành trạng thái của tế bào.

b, Cấu trúc LSTM

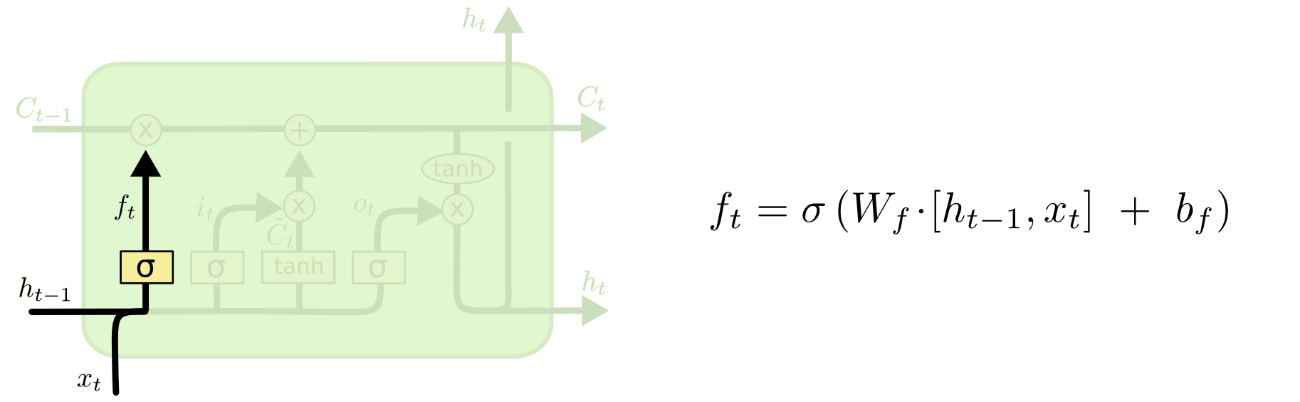
Bước đầu tiên của LSTM là quyết định xem thông tin nào cần bỏ đi từ trạng thái tế bào. Quyết định này được đưa ra bởi tầng sigmoid - gọi là "tầng cổng quên" (forget gate layer). Nó sẽ lấy đầu vào là và rồi đưa ra kết quả là một số trong khoảng [0, 1] cho mỗi số trong trạng thái tế bào . Đầu ra là 1 thể hiện rằng nó giữ toàn bộ thông tin lại, còn 0 chỉ rằng toàn bộ thông tin sẽ bị bỏ đi.

LSTM gồm 4 thành phần chính: **cell state**, **input gate**, **output gate** và **forget gate**. Các cell có nhiệm vụ nhớ các giá trị trong khoảng thời gian phụ thuộc và giá trị của cổng **forget gate**. Ba cổng có nhiệm vụ điều chỉnh luồng thông tin vào ra khỏi cell.

Mỗi cell trong LSTM network có khả năng xử lý dữ liệu một cách tuần tự. LSTM rất phù hợp cho bài toán phân loại với dữ liệu theo chuỗi thời gian. LSTMs được phát triển để giải quyết vấn đề đạo hàm bị triệt tiêu (Vanishing gradient)

Với ví dụ mô hình ngôn ngữ dự đoán từ tiếp theo dựa trên tất cả các từ trước đó, với những bài toán như vậy, thì trạng thái tế bào có thể sẽ mang thông tin về giới tính của một nhân vật nào đó giúp ta sử dụng được đại từ nhân xưng chuẩn xác. Tuy nhiên, khi đề cập tới một người khác thì ta sẽ không muốn nhớ tới giới tính của nhân vật nữa, vì nó không còn tác dụng gì với chủ thể mới này. Đầu ra nằm trong khoảng [0, 1] sẽ cho biết sẽ loại đi những gì.

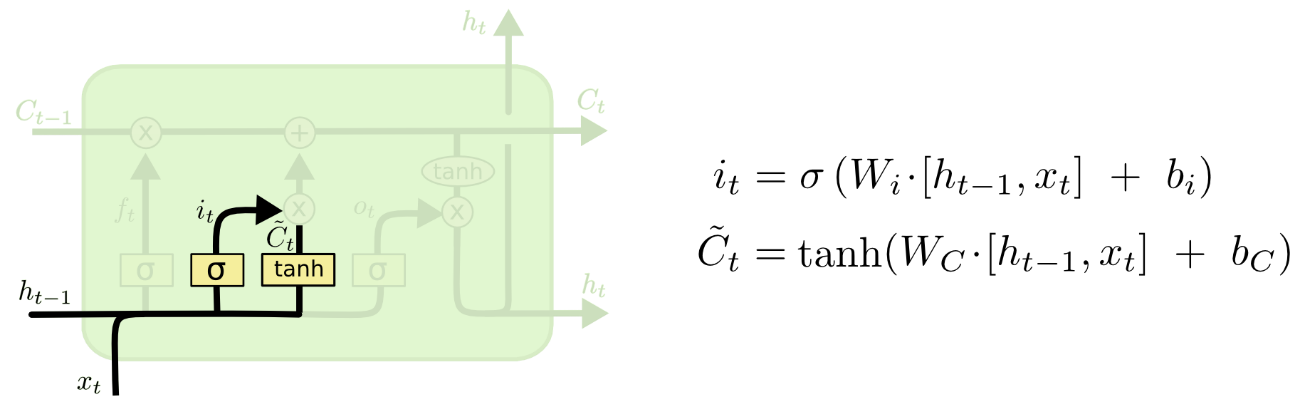
Công thức:



Hình 3.10 Forget gate

Bước tiếp theo là quyết định xem thông tin mới nào ta sẽ lưu vào trạng thái tế bào. Việc này gồm 2 phần. Đầu tiên là sử dụng một tầng sigmoid được gọi là “tầng cổng vào" (input gate layer) để quyết định giá trị nào ta sẽ cập nhập. Tiếp theo là một tầng tanh tạo ra một véc-tơ cho giá trị mới nhằm thêm vào cho trạng thái. Trong bước tiếp theo, ta sẽ kết hợp 2 giá trị đó lại để tạo ra một cập nhập cho trạng thái. Chẳng hạn với ví dụ mô hình ngôn ngữ của ta, ta sẽ muốn thêm giới tính của nhân vật mới này vào trạng thái tế bào và thay thế giới tính của nhân vật trước đó.

Công thức:



Hình 3.11 Input gate

Giờ là lúc cập nhập trạng thái tế bào cũ thành trạng thái mới . Ở các bước trước đó đã quyết định những việc cần làm, nên giờ ta chỉ cần thực hiện là xong. Ta sẽ nhân trạng thái cũ với để bỏ đi những thông tin ta quyết định quên lúc trước. Sau đó cộng thêm \* . Trạng thái mới thu được này phụ thuộc vào việc ta quyết định cập nhập mỗi giá trị trạng thái ra sao.

Với bài toàn mô hình ngôn ngữ, chính là việc ta bỏ đi thông tin về giới tính của nhân vật cũ, và thêm thông tin về giới tính của nhân vật mới như ta đã quyết định ở các bước trước đó.

Công thức:

Ảnh có chứa văn bản, đồng hồ

Mô tả được tạo tự động

Hình 3.12 Update Cell state

Cuối cùng, ta cần quyết định xem ta muốn đầu ra là gì. Giá trị đầu ra sẽ dựa vào trạng thái tế bào, nhưng sẽ được tiếp tục sàng lọc. Đầu tiên, ta chạy một tầng sigmoid để quyết định phần nào của trạng thái tế bào ta muốn xuất ra. Sau đó, ta đưa nó trạng thái tế bảo qua một hàm tanh để co giá trị nó về khoảng [-1, 1], và nhân nó với đầu ra của cổng sigmoid để được giá trị đầu ra ta mong muốn.

Với ví dụ về mô hình ngôn ngữ, chỉ cần xem chủ thế mà ta có thể đưa ra thông tin về một trạng từ đi sau đó. Ví dụ, nếu đầu ra của chủ thể là số ít hoặc số nhiều thì ta có thể biết được dạng của trạng từ đi theo sau nó phải như thế nào.

Công thức:

Ảnh có chứa văn bản, đồng hồ

Mô tả được tạo tự động

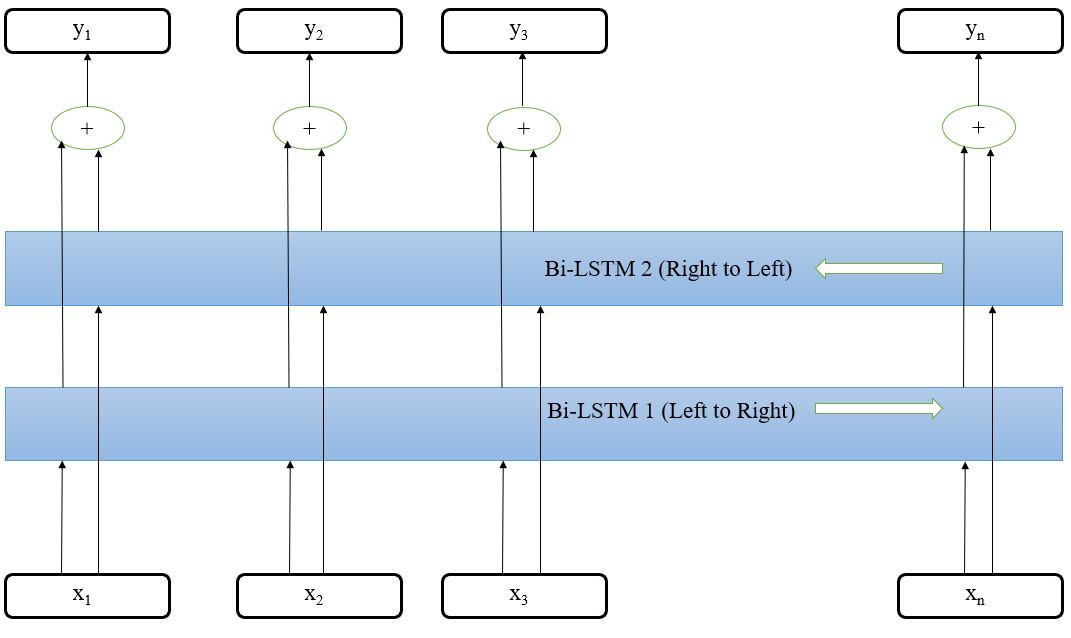
Hình 3.13 Output gate

## CẢI TIẾN LSTM

### 34Mạng LSTM hai chiều (Bidirectional-LSTM)

Như ở 3.1.2.4, ta có thể sử dụng mạng nơ ron hồi quy theo hai chiều ngược nhau để xử lý Một đơn vị RNN sẽ làm như thường lệ, tức là ta sẽ dùng nó để học các tín hiệu đầu vào từ thời điểm ban đầu tới thời điểm kết thúc (đi xuôi). Còn đơn vị RNN còn lại, ta sẽ đọc theo thứ tự thời điểm từ kết thúc trở lại ban đầu (đi ngược). Sau khi có cả hai kết quả, chúng sẽ được gom lại thành một để có thể dự đoán. Với ý tưởng như vậy, tại một thời điểm bất kỳ, mạng sẽ có được các thông tin trước và sau thời điểm t hiện tại.

Do bản chất LSTM là cải tiến của RNN, cho nên ta có thể áp dụng nó và biến nó thành mạng nơ ron dài ngắn song song (BiLSTM). Mỗi LSTM sẽ vẫn có khả năng quên thông tin cũ (cổng quên), lọc thông tin mới (cổng đầu vào), hoặc giấu bớt kết quả (cổng đầu ra) như bình thường. Chính vì vậy, các thông tin từ quá khứ tới tương lai của mạng BiLSTM đều có thể tự học để tự điều chỉnh. Dẫn tới việc với các bài toán mà ta cần biết nhiều hơn về ngữ cảnh hiện tại của nó, thì mạng BiLSTM cho kết quả tốt hơn [4]. Hình dưới đây mô tả mạng BiLSTM:



Hình 3.14 Mô tả mạng LSTM 2 chiều

## XÂY DỰNG MÔ HÌNH

### Thu thập dữ liệu

[VNTC/Data/10Topics/Ver1.1 at master · duyvuleo/VNTC · GitHub](https://github.com/duyvuleo/VNTC/tree/master/Data/10Topics/Ver1.1)

Dữ liệu mà nhóm chúng em sử dụng trong đề tài nghiên cứu lần này được tổng hợp từ nhiều trang báo điện tử khác nhau như **vnexpress.net**, **tuoitre.vn**, **thanhnien.vn**, **nld.com.vn**. Các bài báo từ rất nhiều lĩnh vực trong cuộc sống như chính trị xã hội, đời sống, khoa học, kinh doanh, pháp luật, sức khoẻ, thế giới, thể thao, văn hoá, … được chia thành các file nhỏ giúp việc huấn luyện mô hình trở nên dễ dàng và thuận tiện hơn. Bộ dữ liệu sẽ được chia thành hai tập: tập train và tập test. Ta có thể đọc thêm về phương pháp lấy dữ liệu ở [5].

**Train:**

|  |  |  |
| --- | --- | --- |
| **Topic** | **Topic ID** | **Files** |
| Chính trị xã hội | XH | 5219 |
| Đời sống | DS | 3159 |
| Khoa học | KH | 1820 |
| Kinh doanh | KD | 2552 |
| Pháp luật | PL | 3868 |
| Sức khoẻ | SK | 3384 |
| Thế giới | TG | 2898 |
| Thể thao | TT | 5298 |
| Văn hoá | VH | 3080 |
| Công nghệ thông tin | IT | 2481 |
| Total |  | 33759 |

**Test:**

|  |  |  |
| --- | --- | --- |
| **Topic** | **Topic ID** | **Files** |
| Chính trị xã hội | XH | 7567 |
| Đời sống | DS | 2036 |
| Khoa học | KH | 2096 |
| Kinh doanh | KD | 5276 |
| Pháp luật | PL | 3788 |
| Sức khoẻ | SK | 5417 |
| Thế giới | TG | 6716 |
| Thể thao | TT | 6667 |
| Văn hoá | VH | 6250 |
| Công nghệ thông tin | IT | 4560 |
| Total |  | 50373 |

### Lựa chọn công cụ xử lý dữ liệu

Trong đồ án này để thử nghiệm được mô hình chúng em đã kết hợp sử dụng các thư viện mã nguồn mở và các công cụ tự xây dựng để xử lý dữ liệu, huấn luyện mô hình và dự báo.

* **NLTK**: Công cụ xử lý ngôn ngữ tự nhiên mã nguồn mở dành riêng cho NLP và được tích hợp vào Python. Nó đang ngày càng hoàn thiện và tích hợp các công cụ mới bởi hàng ngàn lập trình viên và cộng tác viên trên khắp thế giới. NLTK bao gồm những thư viện hàm, các công cụ phân tích, các corpus, wordnet, … giúp đơn giản hoá, tiết kiệm thời gian và công sức cho các lập trình viên.
* **Underthesea**: Do sự khác nhau về tiếng Anh và tiếng Việt, dựa trên sự phát triển của gói công cụ **NLTK**, nhóm phát triển người Việt đã tạo nên công cụ thuần Việt nhất, hỗ trợ xử lý các bài toán trong ngôn ngữ tiếng Việt một cách tối ưu nhất. Đây là một công cụ mã nguồn mở tích hợp cho Python, giải các bài toán tách từ, gán nhãn từ loại tiếng Việt một cách thuận tiện nhất.
* **Tensorflow**: Một khung làm việc mã nguồn mở do Google phát hành được sử dụng để xây dựng các mô hình học máy, tạo môi trường nghiên cứu, thực hiện các thử nghiệm một cách nhanh chóng và dễ dàng, đặc biệt là có khả năng chuyển đổi các bản thiết kế prototype tới các ứng dụng trong sản xuất.
* **Python**: Ngôn ngữ lập trình để xây dựng mô hình đối thoại tiếng Việt.
* **Google Collab**: Cho phép viết và thực thi Python trong trình duyệt với các lợi ích: Không yêu cầu cấu hình, sử dụng miễn phí GPU, chia sẻ dễ dàng

### Tiền xử lý dữ liệu

#### **Xây dựng tập ngữ liệu chứa các câu**

Ta xây dựng bộ ngữ liệu là 1 list với từng phần tử trong list tương ứng từng câu. Mỗi từ trong các câu đều phải là chữ cái Latin. Nếu 1 câu bất kì chứa 1 từ không thuộc chữ cái Latin thì ta không chọn câu đấy vào tập ngữ liệu.

Đầu tiên, ta import thư viện để xử lý với biểu thức chính quy. Nhắc lại, một biểu thức chính quy là một chuỗi các ký tự được định nghĩa theo một quy tắc định trước nhằm xác định một tập hơn các chuỗi phù hợp với nó.

|  |
| --- |
| import re |

Tiếp theo, ta khai báo tập chứa các ký tự chữ cái Latin, đây chính là các quy tắc của biểu thức chính quy

|  |
| --- |
| alphabet = '^[ \_abcdefghijklmnopqrstuvwxyz0123456789áàảãạâấầẩẫậăắằẳẵặóòỏõọôốồổỗộơớờởỡợéèẻẽẹêếềểễệúùủũụưứừửữựíìỉĩịýỳỷỹỵđ!\"\',\-\.:;?\_\(\)]+$' |

Ta xây dựng hàm có đầu vào là tập ngữ liệu chứa các văn bản, đầu ra là tập chứa các câu

|  |
| --- |
| def latin\_extract(data):      # khởi tạo tập chứa các câu      latin\_extract\_data=[]      # duyệt qua từng văn bản      for i in data:        if i == 1:          break        # thay thế ký tự xuống dòng là dấu châm câu        i=i.replace("\n",".")        # tách văn bản theo từng dấu chấm câu        sentences=i.split(".")        for j in sentences:    # kiểm tra xem độ dài của câu có lớn hơn 2 và có chứa các quy tắc trong biểu thức chính quy            if len(j.split()) > 2 and re.match(alphabet, j.lower()):               # nếu thỏa mãn thì thêm các câu vào tập ngữ liệu                latin\_extract\_data.append(j)  # đầu ra là tập chứa các câu      return latin\_extract\_data |

Gọi hàm trả về kết quả:

|  |
| --- |
| training\_data = latin\_extract(data)  print(len(training\_data)) #kiểm tra độ dài tập chứa các câu  training\_data[10] # câu thứ 10 trong tập |
| 593088  “Vậy chủ trương đổi mới phương thức và nội dung lãnh đạo của Đảng được đặt ra như thế nào? Nhiệm kỳ khóa X sẽ thực hiện đến đâu?” |

Ta dễ nhận thấy rằng câu trên có chứa ký tự đặc biệt là “?”, và qua thực nghiệm chúng tôi còn ghi nhân một số ký tự như “@#$%^” hoặc các icon và các thẻ html. Các ký tự này là các nhiễu làm giảm độ chính xác của quá trình huấn luyện mô hình. Chính vì vậy ta sẽ cần loại bỏ các ký tự đặc biệt này.

#### **Làm sạch văn bản**

Đầu tiên ta cần import thư viện xử lý các vòng lặp trong python

|  |
| --- |
| import itertools |

Ta khai báo hàm trích xuất các cụm từ, có đầu vào là từng câu trong tập ngữ liệu. Đầu ra sẽ là các chuỗi chứa bất kỳ tự nào trong nhóm từ a tới Z, từ 0 tới 9 và dấu gạch dưới “\_”.

|  |
| --- |
| def extract\_phrases(text):      return re.findall(r'\w[\w ]+', text) |

Sau khi loại bỏ các ký tự dặc biệt, ta sẽ bị thừa các khoảng trắng của các ký tự này. Chính vì vậy ta cũng cần loại bỏ các khoảng trắng thừa.

|  |
| --- |
| def \_extract\_phrases(data):      phrases = itertools.chain.from\_iterable(extract\_phrases(text) for text in data)      phrases = [p.strip() for p in phrases if len(p.split()) > 1]      return phrases |

Gọi hàm trả về kết quả và hiển thị kết quả

|  |
| --- |
| phrases = \_extract\_phrases(training\_data)  print(len(phrases)) # độ dài tập ngữ liệu sau khi làm sạch  print(phrases[10]) # trích xuất thử một câu trong tập |
| 1416112  “xảy ra ở nơi này nơi khác” |

#### **Tách từ**

Trong tiếng Việt, dấu cách (space) không được sử dụng như 1 kí hiệu phân tách từ, nó chỉ có ý nghĩa phân tách các âm tiết với nhau. Vì thế, để xử lý tiếng Việt, công đoạn tách từ (word segmentation) là 1 trong những bài toán cơ bản và quan trọng bậc nhất. Ví dụ : từ “đất nước” được tạo ra từ 2 âm tiết “đất” và “nước”, cả 2 âm tiết này đều có nghĩa riêng khi đứng độc lập, nhưng khi ghép lại sẽ mang một nghĩa khác. Vì đặc điểm này, bài toán tách từ trở thành 1 bài toán tiền đề cho các ứng dụng xử lý ngôn ngữ tự nhiên khác như phân loại văn bản, tóm tắt văn bản, máy dịch tự động, Thuật toán tách từ dã được chúng tôi làm rõ ở **2.4**

Ta sẽ chia bộ ngữ liệu ra thành các bi-grams (2.3). Đầu tiên cần import các thư viện cần thiết xử lý:

|  |
| --- |
| from nltk import ngrams  import string  import numpy as np  from tqdm import tqdm |

Khai báo các hàm biến đổi bi-grams:

|  |
| --- |
| def gen\_ngrams(words, n=2):      return ngrams(words.split(), n)  def generate\_bi\_grams(phrases):      list\_ngrams = []      for p in tqdm(phrases):          # nếu các phrases không nằm trong bảng chữ cái  # thì bỏ qua        if not re.match(alphabet, p.lower()):          continue        # tách câu thành các bi-grams        for ngr in gen\_ngrams(p, NGRAM):          if len(" ".join(ngr)) < MAXLEN:            list\_ngrams.append(" ".join(ngr))        return list\_ngrams |

#### **Chuẩn hóa từ**

Mục đích là đưa văn bản từ các dạng không đồng nhất về cùng một dạng. Dưới góc độ tối ưu bộ nhớ lưu trữ và tính chính xác cũng rất quan trọng. Thường thì ở bước này sẽ chuẩn hóa font chữ, các ký tự viết hoa chuyển thành không viết hoa…

#### **Loại bỏ stopword**

StopWords là những từ xuất hiện nhiều trong ngôn ngữ tự nhiên, tuy nhiên lại không mang nhiều ý nghĩa. Ở tiếng việt StopWords là những từ như: để, này, kia... Tiếng anh là những từ như: is, that, this.. Có 2 cách chính để loại bỏ stopword : Dùn**g từ điển stopword** (xây dựng từ điển stopword và loại bỏ các từ đó)  và dùng **tần suất xuất hiện** (**tf-idf**, đơn giản là các từ càng xuất hiện nhiều thì càng có khả năng là stopword). Ta có thể đọc thêm về **tf-idf** ở đây [4]

### Xây dựng bộ từ điển

Chúng ta cần biến tất cả các từ của trong văn bản của chúng ta thành dạng biểu diễn số. Cách đơn giản nhất mà chúng ta có thể làm đó chính là xây dựng một bộ từ điển rồi sau đó thay thế từ đó bằng thứ tự xuất hiện trong từ điển.

### Vector hóa từ và văn bản

Bước này mục đích là vector hoá từ trong từng câu. Thuật toán dùng để vector hóa từ hay dùng có thể kể đến word2vect : biểu thị mỗi từ thành 1 vector. Ngoài ra ta cũng có thể dùng doc2vect : biểu thị văn bản thành 1 vector. Gensim là thư viện thường được dùng khi vector hóa từ và văn bản.

### Huấn luyện mô hình

Với việc huấn luyện mô hình, chúng tôi sử dụng công cụ TensorFlow (mục **3.3.2**). Chúng tôi sử dụng mô hình mạng BiLSTM ở mục **3.2** để có mạng có khả năng ghi nhớ thông tin dài hạn, và cũng có khả năng xử lý thông tin hai chiều một cách song song.

Dưới đây là các tham số cấu hình của mô hình mạng BiLSTM:

|  |  |
| --- | --- |
| Lớp | Tham số |
| Input Embedding | 256 neurons |
| BiLSTM | 256 neurons |
| Đầu ra activation (Softmax) | 256 neurons |

Số vòng lặp huấn luyện (epochs): 10.

Kích cỡ lô (batch size): 512, tức là ta sẽ gom nhóm 512 mẫu cùng 1 lúc để làm đầu vào cho mạng BiLSTM. Mà ta biết 1 epoch tương đương với việc huấn luyện cả bộ dữ liệu huấn luyện. Ở đây ta khai ta báo epochs là 10, tức là ta sẽ huấn luyện lặp lại 10 lần liên tục. Mỗi lần huấn luyện, ta cần đưa vào mạng số lô (batch size) là:

Số lô trên một epoch =

Phương pháp tối ưu hóa sử dụng: Gradient hướng giảm

Hàm mục tiêu: Độ hỗn loạn (cross-entropy) (**3.1.2.3**)

# KẾT QUẢ THỰC NGHIỆM

## KIỂM THỬ

Để kiểm tra độ hiệu quả của mô hình, nhóm 6 sẽ cho vào những câu sai chính tả để mô hình có thể dự đoán kết quả như sau:

|  |  |
| --- | --- |
| Câu 1 | Những người trẻ tuổi mới bước chân vao xả hội, cho dù là trong cuộc sống hay trong côg việc đều có thể cảm thấy mình nói chuyện không khéo và luôn sợ mình sẽ nói sai, do đó không dám chủ ddộng nói chyuện voi người khác, để su e ngại vaf nỗi no lắng gây áp lực cho chính mình khiến barn thân lơ laf nhiều việc tốt và mất đi cơ hội thăng tiến. Thực tế, không ai sinh ra đã khéo ăn nói, những người khéo nói cnũg phải trải qua rèn luyện mới thafnh. Chỉ cần nỗ lực học tập, bạn sẽ trowr thành người tuyệt vời nhất. |
| Những người trẻ tuổi mới bước chân vào xã hội cho dù là trong cuộc sống hay trong công việc đều có thể cảm thấy mình nói chuyện không khéo và luôn sợ mình sẽ nói sai do đó không dám chủ động nói chuyện với người khác để sự e ngại và nỗi lo lắng gây áp lực cho chính mình khiến bản thân lơ là nhiều việc tốt và mất đi cơ hội thăng tiến Thực tế không ai sinh ra đã khéo ăn nói những người khéo nói cũng phải trải qua rèn luyện mới thành Chỉ cần nỗ lực học tập bạn sẽ trở thành người tuyệt vời nhất |

|  |  |
| --- | --- |
| Câu 2 | Trong lớp hojc, Hillary tích cực tam gia các buổi thảo luận với giáo viên và với bạn hcọ. Bà luôn suy nghĩ và tìm ra các đề tài hay để tất cả ọi người cùng tranh luận. Ngoafi ra, bà còn tập nhữzg bạn học co cùng sở thích, thành lập một câu lạc bộ chuyên tổ chức các buổi thảo luận ve mọi đeef tài, từ việc quốc gia đại sự đến cuộc sống thường ngày, từ cuủ đề khoa học kĩ htuật đếd van hóa nghệ thuậ... Chisnh từ những buổi tranh luận đó, tài ăn nosi của Hillary ddax được nâng cao rõ rệt. |
| Trong lớp học Hillary tích cực tham gia các buổi thảo luận với giáo viên và với bạn học Bà luôn suy nghĩ và tìm ra các đề tài hay để tất cả mọi người cùng tranh luận Ngoài ra bà còn tập những bạn học có cùng sở thích thành lập một câu lạc bộ chuyên tổ chức các buổi thảo luận về mọi đề tài từ việc quốc gia đại sự đến cuộc sống thường ngày từ chủ đề khoa học kĩ thuật đến văn hóa nghệ thuật Chính từ những buổi tranh luận đó tài ăn nói của Hillary đã được nâng cao rõ rệt |

|  |  |
| --- | --- |
| Câu 3 | Các phát thank viên, ngưowfi dẫn chương trìnk đeu là cac chuyeen gja sử dụng ngôn ngữ. Thế nkưng đa phần họ ềđu cho rằng mìnk ko có khiếu ăn nosi từ nkỏ, vậy tạo sao hoj vẫn thafnh công nkờ vào tài ăn nói của mìnk? Nguyên nkân rất đơn giản, đó là vì họ tự nkận thấy mìnk nói năng ko tốt, nên luôn cố gắng để nâng cao kĩ năng giao tiế |
| Các phát thanh viên người dẫn chương trình đều là các chuyên gia sử dụng ngôn ngữ Thế nhưng đa phần họ đều cho rằng mình không có khiếu ăn nói từ nhỏ vậy tạo sao họ vẫn thành công nhờ vào tài ăn nói của mình Nguyên nhân rất đơn giản đó là vì họ tự nhận thấy mình nói năng không tốt nên luôn cố gắng để nâng cao kĩ năng giao tiếp |

|  |  |
| --- | --- |
| Câu 4 | Mỗi con người, từ xin việc đến thăng tiến, từ tifnh iu đến hôn nhân, từ tiếp thị cho ddến đám phán, từ xã giao đến làm việc... ko thể ko cần đến kĩ năng giao tiếp. Neesu khéo ăn nói, bạn sẽ dễ dàng vượt qua những rắc rối nhỏ và có thể tự bảo vệ mình trong nhữn rắc rối lớn. Còn nếu ko khéo léo ăn nói, rắc riố nhỏ sẽ gây trở ngại, và rắc rối lớn sẽ gây thất bại. Cũng dễ hieeri vì sao có những người lại xếp khả năng ăn nói vào danh sch bn năng sinh tồn mà con người hiện đại cần phải có. |
| Mỗi con người từ xin việc đến thăng tiến từ tình yêu đến hôn nhân từ tiếp thị cho đến đám phán từ xã giao đến làm việc không thể không cần đến kĩ năng giao tiếp Nếu khéo ăn nói bạn sẽ dễ dàng vượt qua những rắc rối nhỏ và có thể tự bảo vệ mình trong những rắc rối lớn Còn nếu không khéo léo ăn nói rắc rối nhỏ sẽ gây trở ngại và rắc rối lớn sẽ gây thất bại Cũng dễ hiểu vì sao có những người lại xếp khả năng ăn nói vào danh sách bản năng sinh tồn mà con người hiện đại cần phải có |

## NHẬN XÉT

### Phân loại lỗi chính tả

Trong Tiếng Việt có thể chia lỗi chính tả thành năm loại như sau:

|  |  |
| --- | --- |
| Sai chính tả không có trong từ điển | Đầu vào: Trới hôm nay đẹp quá |
| Sửa: Trời hôm nay đẹp quá |
| Sai chính tả không có trong từ điển | Đầu vào: Sướng còn đọng trên lá |
| Sửa: Sương còn đọng trên lá |
| Sai chính tả phát âm | Đầu vào: Thuyền của tôi đang leo trong bến |
| Sửa: Thuyền của tôi đang neo trong bến |
| Sai chính tả do viết tắt | Đầu vào: Ko cho |
| Đầu ra: Không cho |
| Sai chính tả do sử dụng “tiếng lóng” | Đầu vào: Tôi iu em |
| Đầu ra: Tôi yêu em |

Ở đồ án lần này, nhóm chúng em tiếp cận chủ yếu vào lỗi S*ai chính tả không có từ trong từ điển* và *Sai chính tả phát âm.* Có thể thấy nhưng câu sai chính tả được nhập trong thực nghiệm đã được sửa lỗi khá chính xác. Lỗi phát âm được sửa **no – lo**, lỗi viết thiếu dấu cũng được mô hình sửa như **neesu – nếu, tifnh – tình,…** Ngoài ra mô hình còn có thể sửa một số lỗi tiếng lóng, viết tắt, thiếu dấu như: **ko – không, iu – yêu, ve – về**.

### Đánh giá độ chính xác

Độ chính xác của mô hình là một trong những yếu tố quan trọng để đánh gia mô hình. Độ chính xác được tính như sau:

Mô hình tính độ chính xác bằng cách thêm nhiễu vào tập test sau đó so sánh đầu ra mô hình với tập test ban đầu. Kết quả sau khi test mô hình có độnh chính xác 99%. Có thể thấy mô hình đã bị overfit nên cần tìm xác khắc phục

# kết luận và hướng phát triển

## KẾT LUẬN

Đồ án này đã đưa ra các lý thuyết và vấn đề trong quá trình thiết lập, huấn luyện và xây dựng một mô hình sửa lỗi chính tả cho tiếng Việt. Kết quả ban đầu đạt được là tiền đề để tạo ra các trợ lý ảo, xây dựng các ứng dụng thông minh có thể hiểu được ngôn ngữ tiếng Việt. Có khả năng áp dụng vào các bài toán thực tế, ví dụ như các gợi ý sửa lỗi chính tả trong tình soạn thảo, tự động sửa lỗi chính tả trong tìm kiếm, gợi ý từ tiếp theo trong soạn tin nhắn, …

Từ kết quả thực nghiệm của luận văn này, nhóm 6 có một số nhận xét:

* Với các chuỗi câu dài thì mạng huấn luyện mất nhiều thời gian hơn.
* Xây dựng hệ thống sửa lỗi chính tả dựa trên mô hình phân loại đầu vào theo hướng mạng nơ ron Bộ nhớ dài ngắn song song BiLSTM: Bộ dữ liệu đầu vào là các văn bản được thêm nhiễu và văn bản gốc. Quá trình huấn luyện được tiến hành dựa trên kỹ thuật mạng nơ ron sâu thông qua hàm softmax để thể hiện xác suất của lớp và Entropy chéo được định nghĩa để đánh giá mục tiêu của đầu ra để dự đoán các câu hỏi được đưa vào của người sử dụng. Phương pháp đánh giá dựa trên độ đo chính xác được sử dụng trong mô hình này nhằm đánh giá kết quả để đưa ra mô hình dự đoán tối ưu.
* Với độ đo chính xác (Accuracy) đã giải quyết mặt hạn chế do dữ liệu thu thập đầu vào không được phong phú. Mô hình BiLSTM tốn nhiều thời gian huấn luyện hơn nhưng cho kết quả tốt hơn mô hình LSTM.

## HƯỚNG PHÁT TRIỂN

Tiếp tục kế thừa những nghiên cứu trước đây và phát triển mô hình sửa lỗi chính tả mới có khả năng sửa lỗi ở các loại khác ngoài lỗi phát âm, lỗi viết tắt, lỗi tiếng lóng, lỗi “quên bật Vietkey” còn có thể sửa lỗi theo ngữ cảnh.

Để áp dụng vào thực tiễn, cần lấy dữ liệu thực tế thay vì tự tạo. Do tự tạo nhiễu sẽ không thể bao quát được lỗi của tiếng VIệt.

Áp dụng các phương pháp học sâu khác để cải thiện độ chính xác của chương trình cao hơn như: Attention, Beam search

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | [[1609.08144] Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation (arxiv.org)](https://arxiv.org/abs/1609.08144) |
| [2] | P. J. Werbos, "Backpropagation through time: what it does and how to do it," in Proceedings of the IEEE, vol. 78, no. 10, pp. 1550-1560, Oct. 1990, doi: 10.1109/5.58337 |
| [3] | [CS 230 - Recurrent Neural Networks Cheatsheet (stanford.edu)](https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks) |
| [4]  [5] | https://nlp.stanford.edu/IR-book/pdf/06vect.pdf  V. C. D. Hoang, D. Dinh, N. le Nguyen and H. Q. Ngo, "A Comparative Study on Vietnamese Text Classification Methods," 2007 IEEE International Conference on Research, Innovation and Vision for the Future, 2007, pp. 267-273, doi: 10.1109/RIVF.2007.369167. |
| [6] | [Understanding LSTM Networks -- colah's blog](https://colah.github.io/posts/2015-08-Understanding-LSTMs/) |
| [7] | Khóa luận tốt nghiệp cử nhân CNTT, Tìm kiếm văn bản tiếng Việt theo chủ đề, Hoàng Công Duy Vũ – Nguyễn Lê Nguyên |
| [8] | [VNTC/Data/10Topics/Ver1.1 at master · duyvuleo/VNTC · GitHub](https://github.com/duyvuleo/VNTC/tree/master/Data/10Topics/Ver1.1) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |