TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN &

TRUYỀN THÔNG VIỆT HÀN

Khoa Khoa Học Máy Tính



ĐỒ ÁN CHUYÊN NGÀNH 3

**HỆ THỐNG SỬA LỖI CHÍNH TẢ TIẾNG VIỆT SỬ DỤNG TRÍ TUỆ NHÂN TẠO**

|  |  |  |
| --- | --- | --- |
| Sinh viên thực hiện: | Lê Văn Phước | 20IT487 |
|  | Nguyễn Khắc Đông | 20IT146 |
|  |  |  |
|  |  |  |
| Giảng viên hướng dẫn: | TS.LÊ THỊ THU NGA | |

Đà Nẵng, 01 tháng 05 năm 2024

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN &

THÔNG VIỆT HÀN

Khoa Khoa Học Máy Tính



ĐỒ ÁN CHUYÊN NGÀNH 3

**HỆ THỐNG SỬA LỖI CHÍNH TẢ TIẾNG VIỆT SỬ DỤNG TRÍ TUỆ NHÂN TẠO**

|  |  |  |
| --- | --- | --- |
| Sinh viên thực hiện: | Lê Văn Phước | 20IT487 |
|  | Nguyễn Khắc Đông | 20IT146 |
|  |  |  |
|  |  |  |
| Giảng viên hướng dẫn: | TS.LÊ THỊ THU NGA | |

Đà Nẵng, 01 tháng 5 năm 2024

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

…..

|  |  |
| --- | --- |
|  | Chữ ký của giảng viên  *(Ký và ghi rõ họ tên)* |

**LỜI CẢM ƠN**

Đồ án chuyên ngành này được hoàn thành tại trường Đại học Công nghệ thông tin và Truyền thông Việt-Hàn. Trong quá trình học tập, triển khai và hoàn thành đồ án với đề tài “HỆ THỐNG SỬA LỖI CHÍNH TẢ TIẾNG VIỆT SỬ DỤNG TRÍ TUỆ NHÂN TẠO”, chúng em đã nhận được rất nhiều sự giúp đỡ, vì vậy chúng em muốn viết lời cảm ơn này đến:

Chúng em xin gửi lời cảm ơn chân thành tới thầy cô trường Đại học Công nghệ thông tin và Truyền thông Việt Hàn đã giảng dạy và nâng đỡ cho chúng em thời gian qua. Đặc biệt là TS.Lê Thị Thu Nga là người trực tiếp chỉ bảo và hướng dẫn chúng em hoàn thành đồ án này. Bên cạnh đó để hoàn thành đồ án này chúng em đã nhận được những lời khuyên, sự giúp đỡ tận tình từ anh chị em bạn bè, chúng em xin cảm ơn.

Mặc dù đã vận dụng tất cả kiến thức đã được học tập và kinh nghiệm thực tế từ bản thân để hoàn thành đồ án này, song có thể còn có những mặt hạn chế, thiếu sót. Chúng em rất mong nhận được ý kiến đóng góp và sự chỉ dẫn của các thầy cô giáo. Chúng em xin chân thành cảm ơn.

MỤC LỤC

[DANH MỤC HÌNH ẢNH 7](#_Toc168584601)

[CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ ĐỀ TÀI 8](#_Toc168584602)

[1.1 ĐẶT VẤN ĐỀ 8](#_Toc168584603)

[1.2 GIỚI THIỆU BÀI TOÁN SỬA LỖI CHÍNH TẢ 8](#_Toc168584604)

[1.3 MÔ TẢ BÀI TOÁN 9](#_Toc168584605)

[1.4 ĐỊNH HƯỚNG ĐỀ TÀI 9](#_Toc168584606)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 10](#_Toc168584607)

[2.1 LÝ THUYẾT NGÔN NGỮ TIẾNG VIỆT 10](#_Toc168584608)

[2.1.1 Khái niệm về từ 10](#_Toc168584609)

[2.1.2 Hình thái từ tiếng Việt 10](#_Toc168584610)

[2.1.3 Khái niệm văn bản 12](#_Toc168584611)

[2.2 LỖI CHÍNH TẢ 12](#_Toc168584612)

[2.2.1 Các nguyên nhân gây ra lỗi chính tả 13](#_Toc168584613)

[2.2.2 Phân loại một số lỗi chính tả 14](#_Toc168584614)

[2.2.3 Phát hiện lỗi chính tả 14](#_Toc168584615)

[2.2.4 Sửa lỗi chính tả 14](#_Toc168584616)

[2.2.5 Một số phương pháp kiểm lỗi chính tả 14](#_Toc168584617)

[CHƯƠNG 3: PHƯƠNG PHÁP THỰC HIỆN 15](#_Toc168584618)

[3.1 N-gram 15](#_Toc168584619)

[3.2 Recurrent Neural Network 16](#_Toc168584620)

[3.3 Bi-dierctional RNN 19](#_Toc168584621)

[3.4 Long Short-term Memory 19](#_Toc168584622)

[3.4.1Giới thiệu mô hình 19](#_Toc168584623)

[3.4.2Cách thức hoạt động 21](#_Toc168584624)

[3.5 Bidirectional-LSTM 24](#_Toc168584625)

[CHƯƠNG 4: THỬ NGHIỆM VÀ ĐÁNH GIÁ 25](#_Toc168584626)

[4.1 Dữ liệu 25](#_Toc168584627)

[4.1.1 Thu thập dữ liệu 25](#_Toc168584628)

[4.2.2 Tiền xử lí văn bản 26](#_Toc168584629)

[4.2.3 Vector hóa từ và văn bản 33](#_Toc168584630)

[4.2 Xây dựng mô hình 34](#_Toc168584631)

[4.3 Dùng mô hình và các phương pháp xử lý ngôn ngữ tự nhiên để sửa lỗi chính tả cho văn bản 35](#_Toc168584632)

[4.4 Đánh giá hai phương pháp 4-gram và 5-gram 38](#_Toc168584633)

[4.4.1 Bộ dữ liệu dùng để đánh giá và thông số của nó 38](#_Toc168584634)

[4.4.2 Thông số của kết quả đầu ra với phương pháp 4-gram với bộ dữ liệu ở 4.4.1 38](#_Toc168584635)

[4.4.3 Thông số của kết quả đầu ra với phương pháp 5-gram với bộ dữ liệu ở 4.4.1 38](#_Toc168584636)

[4.4.4 Nhận xét 39](#_Toc168584637)

[4.5 Sản phẩn 39](#_Toc168584638)

[KẾT LUẬN 40](#_Toc168584639)

[TÀI LIỆU THAM KHẢO 41](#_Toc168584640)

# DANH MỤC HÌNH ẢNH

[Hình 3.1: Hình ảnh minh hoạ mô hình N-gram với nhiều mức N khác nhau. 14](#_Toc168585451)

[Hình 3.2: Hình ảnh minh hoạ cách thức hoạt động của mạng RNN. 15](#_Toc168585452)

[Hình 3.3: Hình ảnh minh hoạ lớp kích hoạt trong RNN. 16](#_Toc168585453)

[Hình 3.4: Kiến trúc của mạng RNN (Bên trái là kiến trúc tổng quan; Bên phải là cấu tạo của một lớp Feedforward). 16](#_Toc168585454)

[Hình 3.5: Mô hình RNN được sử dụng theo các loại bài toán. 17](#_Toc168585455)

[Hình 3.6: Giải pháp cho vấn đề Vanishing Gradient (Bên trái là hàm kích hoạt ReLU; Bên phải là skip connec- tion. 17](#_Toc168585456)

[Hình 3.7: Hình ảnh mô hình Bi-directional RNN. 18](#_Toc168585457)

[Hình 3.8: Hình ảnh mô hình Long Short-term Memory 19](#_Toc168585458)

[Hình 3.9: Hình ảnh Cell state 19](#_Toc168585459)

[Hình 3.10: Hình ảnh Cổng Forget 20](#_Toc168585460)

[Hình 3.11: Hình ảnh cổng Tạo ra các giá trị ứng viên vào Cell state 21](#_Toc168585461)

[Hình 3.12: Hình ảnh cập nhật Cell state 21](#_Toc168585462)

[Hình 3.13: Hình ảnh đầu ra của Cell state 22](#_Toc168585463)

[Hình 3.14: Hình ảnh mô tả mạng BiLSTM 23](#_Toc168585464)

[Hình 4.2: Hình ảnh dòng lệnh thực hiện việc tách câu, loại bỏ kí tự. 25](#_Toc168585465)

[Hình 4.3: Hình ảnh dòng lệnh thực hiện việc tách câu thành các cụm từ dài nhất không chứa dấu câu. 26](#_Toc168585466)

[Hình 4.4.1: Hình ảnh dòng lệnh thực hiện việc tách 5-gram. 27](#_Toc168585467)

[Hình 4.4.2: Hình ảnh dòng lệnh thực hiện việc tách 4-gram. 27](#_Toc168585468)

[Hình 4.5: Hình ảnh dòng lệnh biến phục vụ việc sinh lỗi chính tả trong hàm add\_noise. 28](#_Toc168585469)

[Hình 4.6: Hình ảnh dòng lệnh hàm teen\_code. 28](#_Toc168585470)

[Hình 4.7.1: Hình ảnh dòng lệnh hàm add\_noise. 30](#_Toc168585471)

[Hình 4.7.2: Hình ảnh dòng lệnh hàm add\_noise(tt). 31](#_Toc168585472)

[Hình 4.8: Hình ảnh dòng lệnh test hàm add\_noise. 31](#_Toc168585473)

[Hình 4.9: Bộ từ vựng dùng để One-Hot Encoding. 32](#_Toc168585474)

[Hình 4.10: Hình ảnh mô hình Bi-directional LSTM. 33](#_Toc168585475)

[Hình 4.11.1: Hình ảnh thông số mô hình B-LSTM với phương pháp 4-gram. 34](#_Toc168585476)

[Hình 4.11.2: Hình ảnh thông số mô hình B-LSTM với phương pháp 5-gram. 34](#_Toc168585477)

[Hình 4.12 Thông số bộ dữ liệu dùng để đánh giá hai phương pháp 37](#_Toc168585478)

[Hình 4.13 Thông số của kết quả đầu ra với phương pháp 4-gram với bộ dữ liệu ở 4.4.1 37](#_Toc168585479)

[Hình 4.14 Thông số của kết quả đầu ra với phương pháp 4-gram với bộ dữ liệu ở 4.4.1 37](#_Toc168585480)

[Hình 4.15: Hình ảnh website sửa lỗi chính tả. 38](#_Toc168585481)

# CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ ĐỀ TÀI

## 1.1 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 muốn trình bày kết quả tìm hiểu về bài toán sửa lỗi chính tả tiếng việt trong văn bản.

## 1.2 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.

## 1.3 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.*

## 1.4 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.

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, chọn ra những văn bản tương đồng với tập huấn luyện. Sau đó đưa tập văn bản đó vào mô hình để dự đoán sửa lỗi các từ sai chính tả.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## 2.1 LÝ THUYẾT NGÔN NGỮ TIẾNG VIỆT

### 2.1.1 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 "từ". 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 từ 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”.

### 2.1.2 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.

2.1.2.1 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,....

2.1.2.2 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*”.

### 2.1.3 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.

## 2.2 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.

### 2.2.1 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 (.)…

### 2.2.2 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”, ...

### 2.2.3 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.

### 2.2.4 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).

### 2.2.5 Một số phương pháp kiểm lỗi chính tả

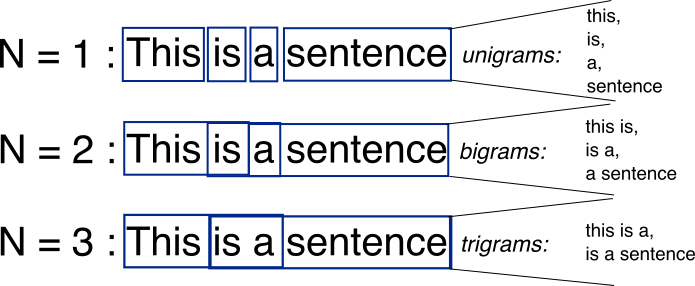
Chúng ta có thể tạm chỉ ra hai phương pháp chính đó là dựa vào luật và dựa vào thống kê. Các phương pháp dựa theo luật có ưu điểm là không tốn quá nhiều tài nguyên của thiết bị, tuy nhiên các chương trình sử dụng phương pháp này không có khả năng học, và hiện tại thì kết quả cũng chưa cao đối với nhiều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

# CHƯƠNG 3: PHƯƠNG PHÁP THỰC HIỆN

## 3.1 N-gram

Mô hình N-gram từ là một mô hình xác suất truyền thống trong ngôn ngữ học. Khung cửa sổ từ được tạo ra có độ dài là n kí tự dùng để tính xác suất xuất hiện của các từ với nhau. Mô hình dự đoán một từ tiếp theo trong chuỗi dựa trên cửa sổ từ trước đó (Hình [3.1).](#_bookmark6)



Hình 3.1: Hình ảnh minh hoạ mô hình N-gram với nhiều mức N khác nhau.

Công thức chung của N-gram có dạng như sau:

P(wn|wn−1,wn−2,...,w1) = P(wn−1,wn−2,...,w1,wn) P(wn−1,wn−2,...,w1) (3.1)

Trong đó :

* *P*(*wn−*1*, wn−*2*, . . . , w*1*, wn*) là xác suất kết hợp của chuỗi từ *wn−*1*, wn−*2*, . . . , w*1*, wn* xuất hiện theo thứ tự cho trước.
* *P*(*wn|wn−*1*, wn−*2*, . . . , w*1) là xác suất có điều kiện để từ *wn* xuất hiện với chuỗi từ cho trước.

Mô hình N-gram là một mô hình đơn giản nhưng hiệu quả cho việc dự đoán từ bằng việc biến đổi các chuỗi từ thành các đại diện xác suất. Tuy nhiên, mô hình này có nhược điểm là chỉ có thể xem xét những chuỗi từ có độ dài nhất định, không thể nắm bắt tốt được ngữ nghĩa của từ hay mối liên hệ giữa các từ trong chuỗi từ dài.

Khi thực hiện làm mịn (smoothing) N-gram để phát hiện từ không có trong tập huấn luyện thì ta có thể làm giảm xác suất của những từ có thể xuất hiện và tăng xác suất của những từ không có ý nghĩa, không xuất hiện trong tập huấn luyện.

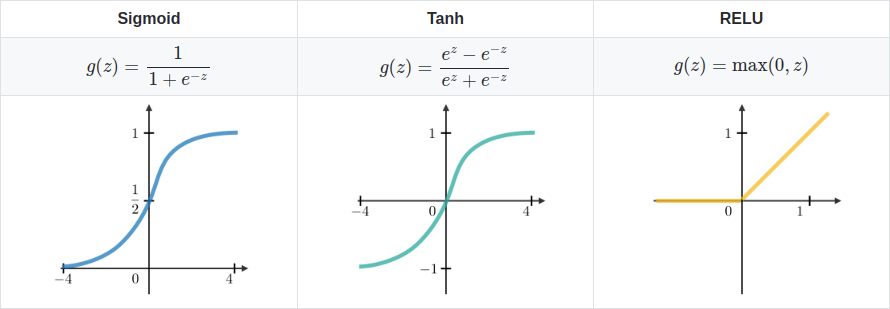
## 3.2 Recurrent Neural Network

Hình 3.2: Hình ảnh minh hoạ cách thức hoạt động của mạng RNN.

Recurrent Neural Network (RNN) được thiết kế để xử lí thông tin dạng chuỗi, với thứ tự thành phần của thông tin đầu vào được nhấn mạnh. Trong việc xử lí ngôn ngữ tự nhiên, RNN có thể nắm bắt ngữ nghĩa, ngữ cảnh và mối quan hệ giữa cá từ một cách hiệu quả nhờ việc có những liên kết được hình thành và lưu giữ thông tin từ đầu vào ban đầu đến các thành phần thông tin sau này (Hình [3.2).](#_bookmark9)

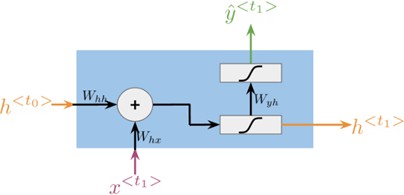
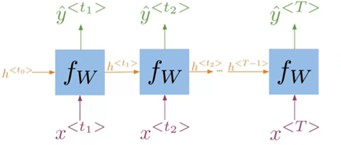
Trong RNN, có một tham số học sẽ lưu lại thông tin về các tính toán về chuỗi thông tin được gọi là trạng thái ẩn (hidden state).

Hidden state và những trọng số đầu ra ở các bước của RNN được quyết định bởi lớp kích hoạt (Hình [3.3)](#_bookmark10) để đưa dữ liệu vào chuỗi sau.



Hình 3.3: Hình ảnh minh hoạ lớp kích hoạt trong RNN.

Kiến trúc mạng RNN cho phép đầu ra từ những chuỗi thông tin trước trở thành đầu vào của những chuỗi thông tin sau, và có cấu trúc như hình [3.4:](#_bookmark11)

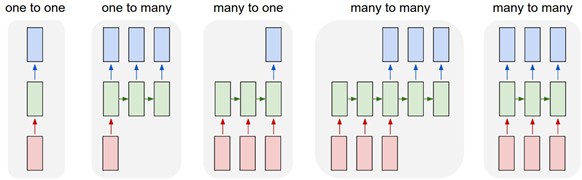


Hình 3.4: Kiến trúc của mạng RNN (Bên trái là kiến trúc tổng quan; Bên phải là cấu tạo của một lớp Feedforward).

Trong đó:

* *h<t>* = *g*(*Wh*[*h<t−*1*>, x<t>*] + *bh* : Hidden state của mô hình RNN tại mỗi bước
* *y<t>* = *g*(*Whhh<t−*1*>* +*Whxx<t>* + *bh*) : Đầu ra của mô hình RNN với *x<t>* là đầu vào

Bài toán sử dụng RNN sẽ được phân loại dựa trên cách *y<t>* được đưa ra khỏi mô hình từ đầu vào là *x<t>*, bao gồm các loại bài toán như hình [3.5.](#_bookmark12)



Hình 3.5: Mô hình RNN được sử dụng theo các loại bài toán.

Ưu điểm của RNN:

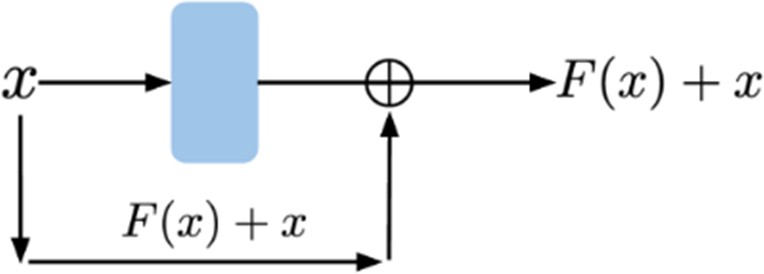
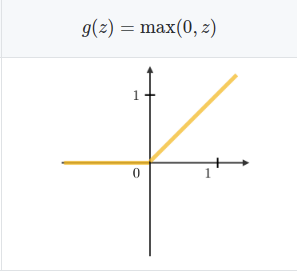
* Giúp nắm bắt được các phụ thuộc từ về mặt ngữ pháp và ngữ nghĩa trong một khoảng ngắn.
* Tốn ít tài nguyên bộ nhớ hơn mô hình N-gram.

Nhược điểm của RNN:

* Gặp khó khăn với các chuỗi thông tin có độ dài lớn.
* Đạo hàm bị triệt tiêu (vanishing gradient), exploding gradient(đạo hàm bùng nổ) do các trọng số trở nên nhỏ đi khi đi qua nhiều lớp, đến khi không còn giá trị sử dụng được hoặc tăng quá mức kiểm soát khi có quá nhiều dữ liệu.

Giải pháp cho vấn đề Vanishing Gradient (Hình [3.6):](#_bookmark13)

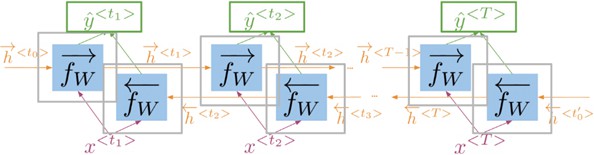
* RNN với hàm kích hoạt ReLu.
* Gradient clipping.
* Skip connection.



Hình 3.6: Giải pháp cho vấn đề Vanishing Gradient (Bên trái là hàm kích hoạt ReLU; Bên phải là skip connec- tion.

## 3.3 Bi-dierctional RNN

Bi-dierctional RNN là mạng RNN nhưng thay vì chỉ di chuyển một chiều dữ liệu từ trái sang phải, thì lúc này Bi-dierctional RNN có hai luồng dữ liệu đi theo hai hướng ngược chiều nhau (Hình [3.7).](#_bookmark15)



Hình 3.7: Hình ảnh mô hình Bi-directional RNN.

Bi-dierctional RNN rất quan trọng, vì nó cho ta biết ngữ cảnh tiếp theo sau từ cần phải dự đoán trong mô hình xử lí ngôn ngữ tự nhiên.

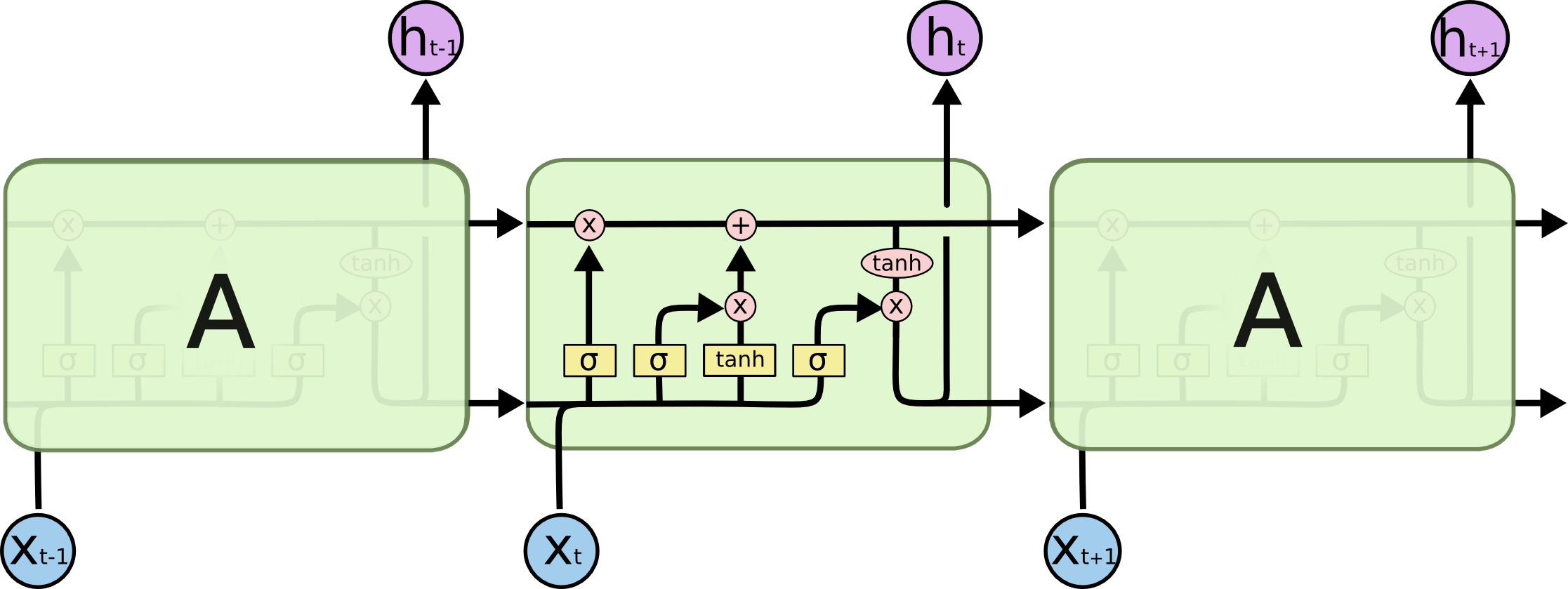
Để có thể dự đoán được *y<t>*, ta cần hidden state của cả hai luồng và kết hợp chúng lại để tạo ra một hidden state mới và tiền hành như RNN thông thường.

## 3.4 Long Short-term Memory

### 3.4.1Giới thiệu mô hình

Mạng Long Short-term Memory (LSTM) là một biến thể của RNN có khả năng học hỏi những phụ thuộc theo thứ tự trong vấn đề dự đoán chuỗi dữ liệu

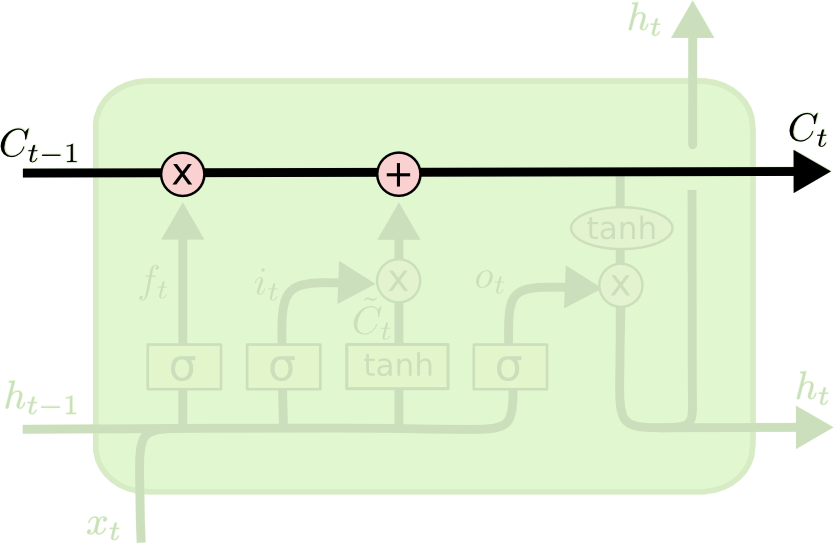
LSTM giải quyết hai vấn đề của RNN là vanishing gradient và exploding gradi- ent bằng cách chọn lựa ra những thông tin cần thiết ở mỗi bước của mô hình và quên những thông tin còn lại. Lúc mô hình có thể lưu lại thông tin ở xa thông tin ở đầu chuỗi (Hình [3.8).](#_bookmark18)



Hình 3.8: Hình ảnh mô hình Long Short-term Memory

LSTM gồm một cell state và một hidden state với ba cổng:

* + - * Cổng Input i: cho biết bao nhiêu thông tin được input vào tại một thời điểm nhất định.
      * Cổng Forget f: cho biết bao nhiều thông tin cần quên tại một thời điểm.
      * Cổng Output o: cho biết bao nhiều thông tin truyền qua tại một thời điểm.
      * Cell state: Giúp mô hình giữ lại thông tin quan trọng và quên đi thông tin không quan trọng (Hình [3.9).](#_bookmark19)



Hình 3.9: Hình ảnh Cell state

LSTM có khả năng thêm hoặc bớt đi thông tin ở Cell state thông qua các cổng Input, Output và Forget. Cổng (Gate) là một cách để lựa chọn thông tin đi qua mạng LSTM. Chúng được cấu thành từ lớp output sigmoid và một phép toán nhân. Lớp output sigmoid cho ra một số thuộc về giá trị 0 hoặc 1, quyết mỗi thành phần được duyệt qua mạng neural hay không.

Những tham số trong LSTM được tính như sau:

* + - * Cổng Input:

*it* = *σ* (*Wiixt* + *bii* + *Whiht−*1 + *bhi*) (3.2)

* + - * Cổng Forget:

*ft* = *σ* (*Wi f xt* + *bi f* + *Wh f ht−*1 + *bh f* ) (3.3)

* + - * Cổng Cell:

*gt* = tanh(*Wigxt* + *big* + *Whght−*1 + *bhg*) (3.4)

* + - * Cổng Output:

*ot* = *σ* (*Wioxt* + *bio* + *Whoht−*1 + *bho*) (3.5)

* + - * Cell State:

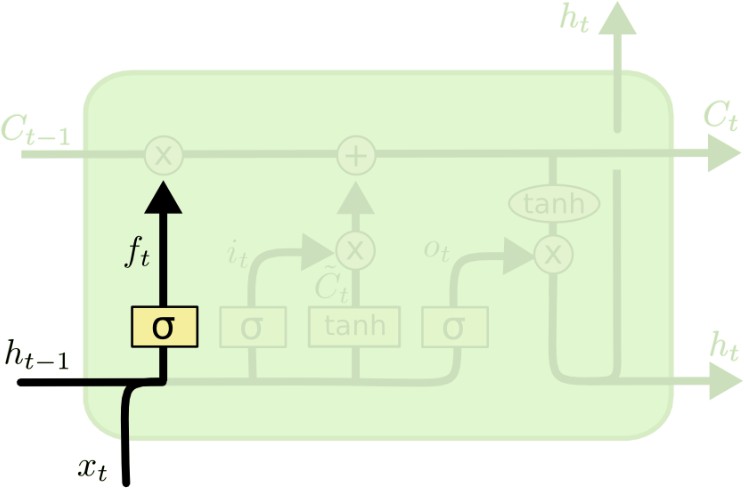
*ct* = *ft ⊙ct−*1 + *it ⊙gt* (3.6)

* + - * Hidden State:

*ht* = *ot ⊙* tanh(*ct*) (3.7)

### 3.4.2Cách thức hoạt động

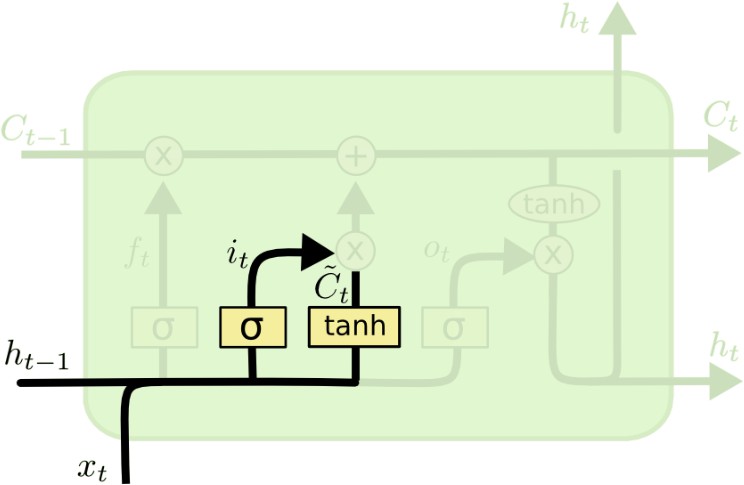
Trong mô hình LSTM, bước đầu tiên phải làm là xác định thông tin nào trong Cell state cần được quên đi (đi qua cổng Forget [3.10)](#_bookmark25) bằng công thức [3.2.](#_bookmark20)



Hình 3.10: Hình ảnh Cổng Forget

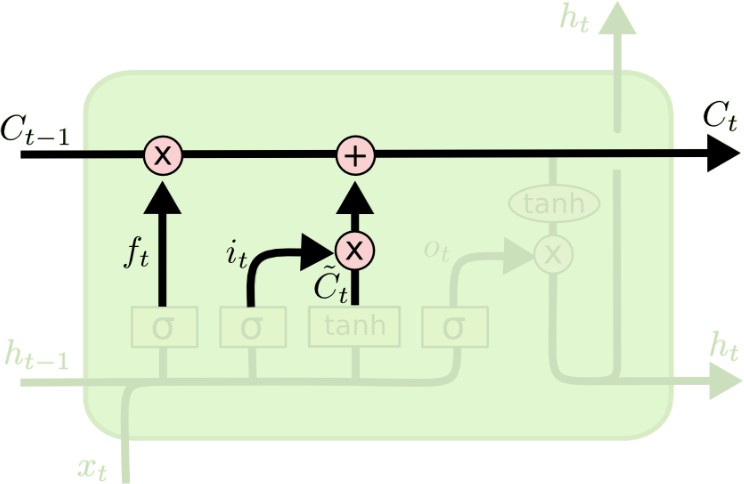
Bước tiếp theo là quyết định thông tin mới được đưa vào Cell state và gồm hai phần như sau (Hình [3.11):](#_bookmark26)

* + - * Một lớp sigmoid gọi là cổng Input sẽ quyết định giá trị nào sẽ được cập nhập theo công thức [3.1.](#_bookmark7)
      * Lớp tanh sẽ tạo ra một vector các giá trị ứng viên đề đưa vào Cell state là *gt* theo công thức [3.3.](#_bookmark21)



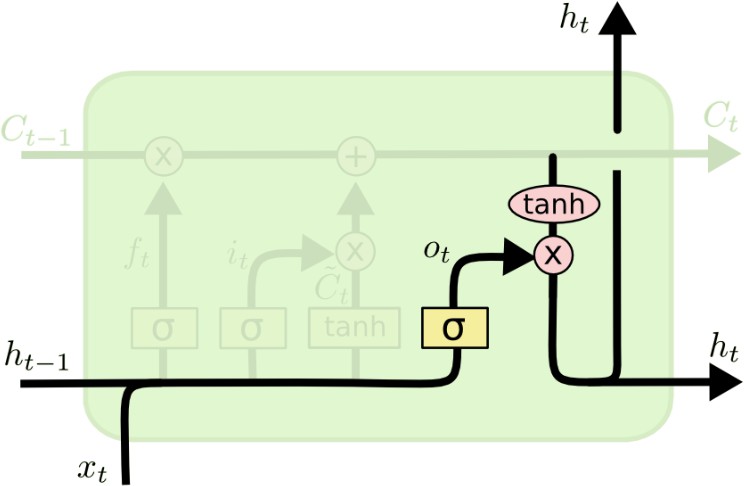
Hình 3.11: Hình ảnh cổng Tạo ra các giá trị ứng viên vào Cell state

Cập nhập Cell state (Hình [3.12)](#_bookmark27) cũ là *Ct−*1 và Cell state mới *Ct* bằng cách nhân giá trị cũ với *ft* và cộng với giá trị mới nhân cho cổng Input theo công thức [3.5.](#_bookmark23)



Hình 3.12: Hình ảnh cập nhật Cell state

Bước cuối cùng là đầu ra của mô hình thông qua Cell state (Hình [3.13),](#_bookmark28) lúc này sẽ đi qua lớp sigmoid để quyết định phần nào trong Cell state được đưa ra theo công thức [3.5.](#_bookmark23) Sau đó Cell state sẽ đi qua lớp tanh để chuyển giá trị về khoảng (-1, 1) và nhân nó với đầu ra của cổng sigmoid theo công thức [3.4.](#_bookmark22)

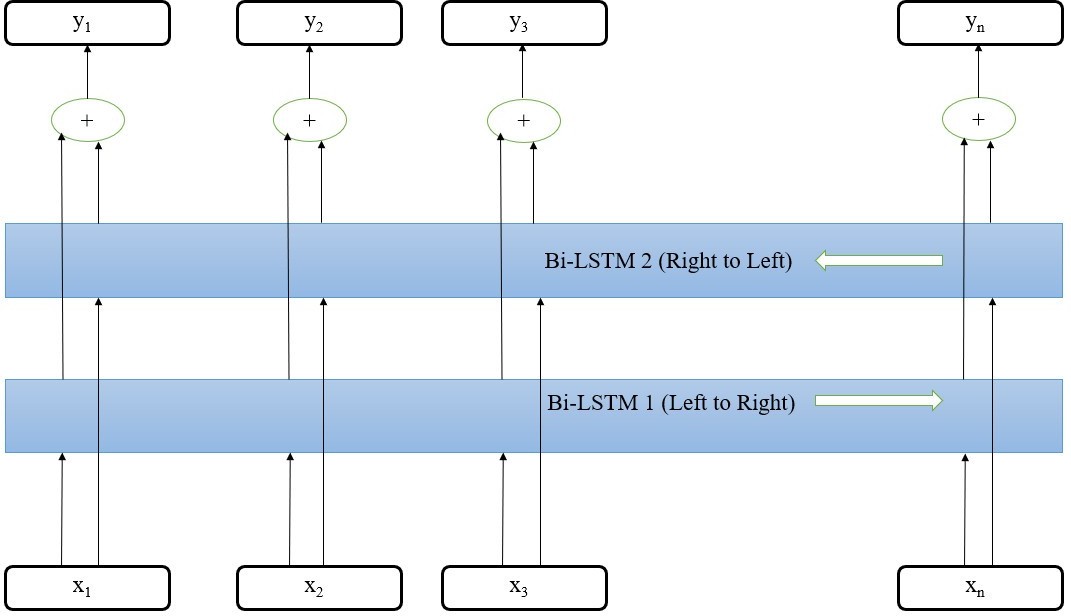


Hình 3.13: Hình ảnh đầu ra của Cell state

## 3.5 Bidirectional-LSTM

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). Bên cạnh đó, 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. Hình [3.14](#_bookmark30) dưới đây mô tả mạng BiLSTM:



Hình 3.14: Hình ảnh mô tả mạng BiLSTM

# CHƯƠNG 4: THỬ NGHIỆM VÀ ĐÁNH GIÁ

Quá trình này nhóm đã xây dựng hai phương pháp là 4-gram, 5-gram để sửa lỗi chính tả cho văn bản

## 4.1 Dữ liệu

### 4.1.1 Thu thập dữ liệu

Dữ liệu huấn luyện được lấy từ tập dữ liệu VNTC 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 tập văn bản nhỏ. Tập dữ liệu được chia thành hai phần là tập huấn luyện và tập kiểm thử.

Link dữ liệu: [https://github.com/duyvuleo/VNTC/tree/master/Data/](https://github.com/duyvuleo/VNTC/tree/master/Data/10Topics/Ver1.1) [10Topics/Ver1.1](https://github.com/duyvuleo/VNTC/tree/master/Data/10Topics/Ver1.1)

|  |  |  |
| --- | --- | --- |
| **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 |
| **Tổng cộng** |  | **33759** |

|  |  |  |
| --- | --- | --- |
| **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 |
| **Tổng cộng** |  | **50373** |

Bảng 4.1: Tệp huấn luyện Bảng 4.2: Tệp kiểm thử

Bảng 4.3: Tổng số tệp văn bản có trong tệp dữ liệu

Do nếu sử dụng tất cả văn bản trong tập huấn luyện sẽ làm cho việc huấn luyện mô hình trở nên kéo dài, nhóm đã quyết định cắt giảm các files trong tập huấn luyện và kiểm thử nhằm giảm số lượng dữ liệu phải xử lí

|  |  |  |
| --- | --- | --- |
| **Topic** | **Topic ID** | **Files** |
| Chính trị xã hội  Đời sống | XH  DS | 1499  1499 |
| Khoa học | KH | 1499 |
| Kinh doanh | KD | 1499 |
| Pháp luật | PL | 1499 |
| Sức khoẻ | SK | 1499 |
| Thể thao  Văn hóa  Công nghệ thông Tin | TT  VH  IT | 1499  1499  1499 |
| **Tổng cộng** |  | **14990** |

|  |  |  |
| --- | --- | --- |
| **Topic** | **Topic ID** | **Files** |
| Chính trị xã hội  Đời sống | XH  DS | 1499  1499 |
| Khoa học | KH | 1499 |
| Kinh doanh | KD | 1499 |
| Pháp luật | PL | 1499 |
| Sức khoẻ | SK | 1499 |
| Thể thao  Văn hóa  Công nghệ thông Tin | TT  VH  IT | 1499  1499  1499 |
| **Tổng cộng** |  | **14990** |

Bảng 4.4: Tệp huấn luyện Bảng 4.5: Tệp kiểm thử

Bảng 4.6: Tổng số tệp văn bản sau khi cắt giảm

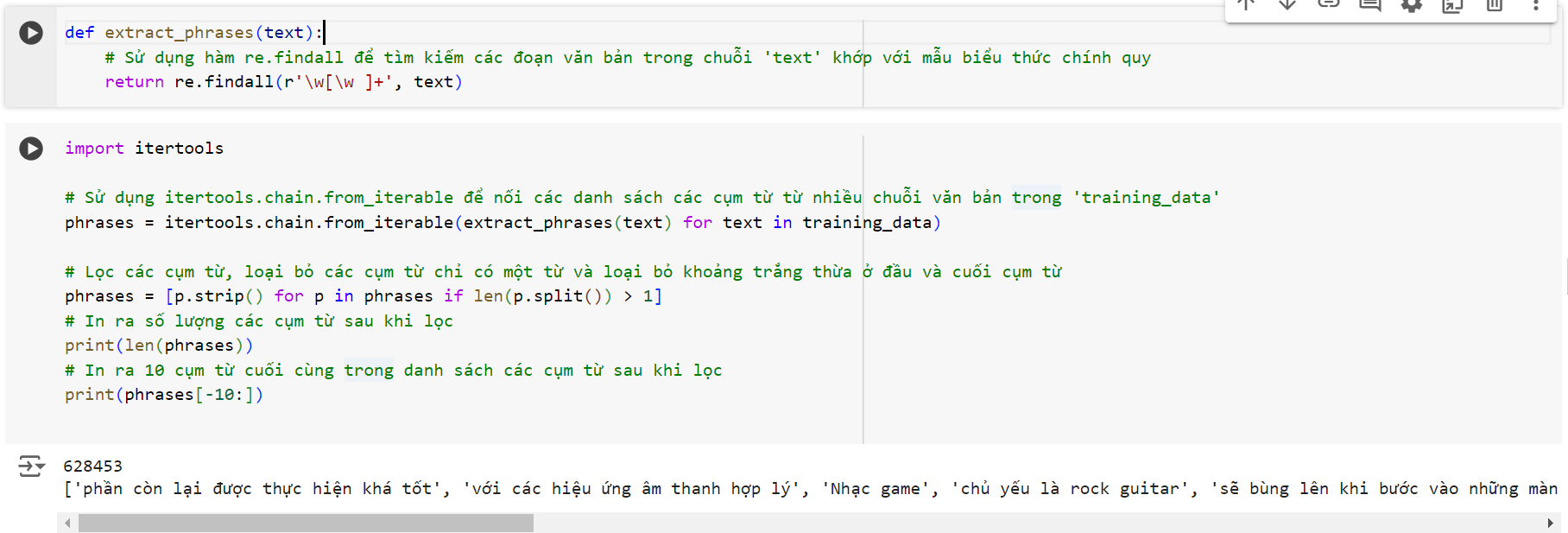
### 4.2.2 Tiền xử lí văn bản

Dữ liệu huấn luyện sau khi cắt giảm sẽ được chia thành từng câu một, loại bỏ đi các kí tự đặc biệt và kí tự phi Latin bằng cách quy định chỉ các kí tự trong danh sách mới được phép có trong văn bản, nếu không thì sẽ bị xóa (Hình [4.2).](#_bookmark37)



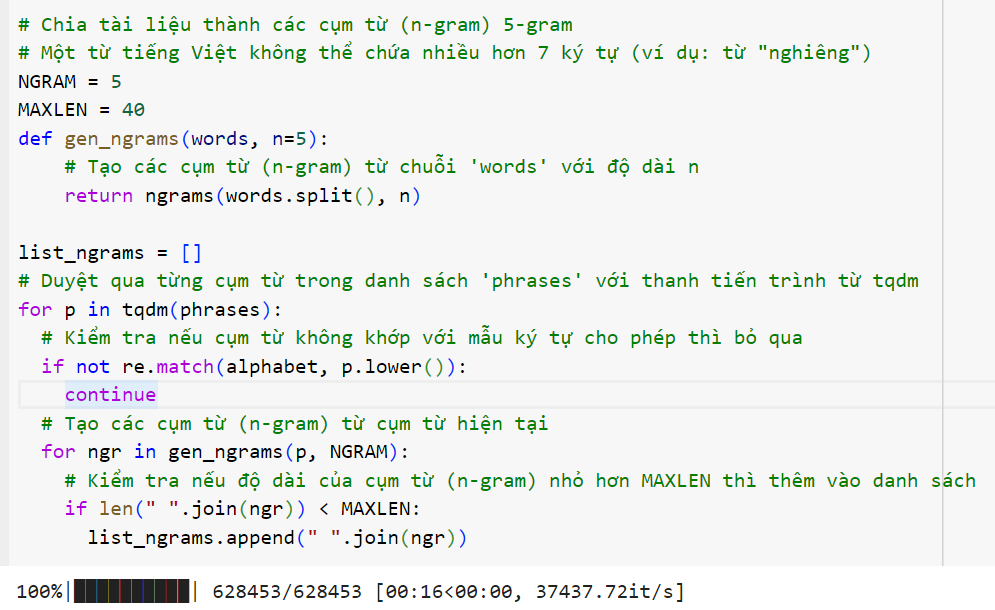
Hình 4.2: Hình ảnh dòng lệnh thực hiện việc tách câu, loại bỏ kí tự.

Sau đó, tách các câu thành các các cụm từ dài nhất có thể sao cho cụm từ đó thỏa mản biểu thức chính quy '\w[\w ]+' (Hình 4.3)

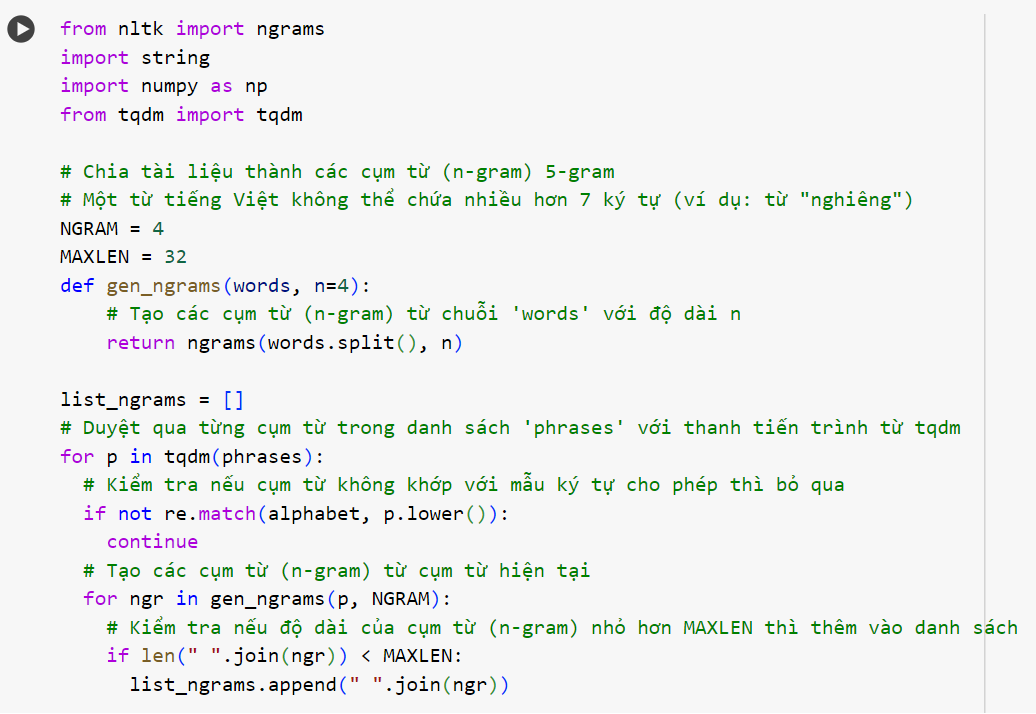


Hình 4.3: Hình ảnh dòng lệnh thực hiện việc tách câu thành các cụm từ dài nhất không chứa dấu câu.

Sau đó, tách các cụm từ trong câu với n-gram là 4 với phương pháp 4-gram và 5 với phương pháp 5-gram (Hình 4.4)



Hình 4.4.1: Hình ảnh dòng lệnh thực hiện việc tách 5-gram.



Hình 4.4.2: Hình ảnh dòng lệnh thực hiện việc tách 4-gram.

Trường hợp sai lỗi chính tả bao gồm có:

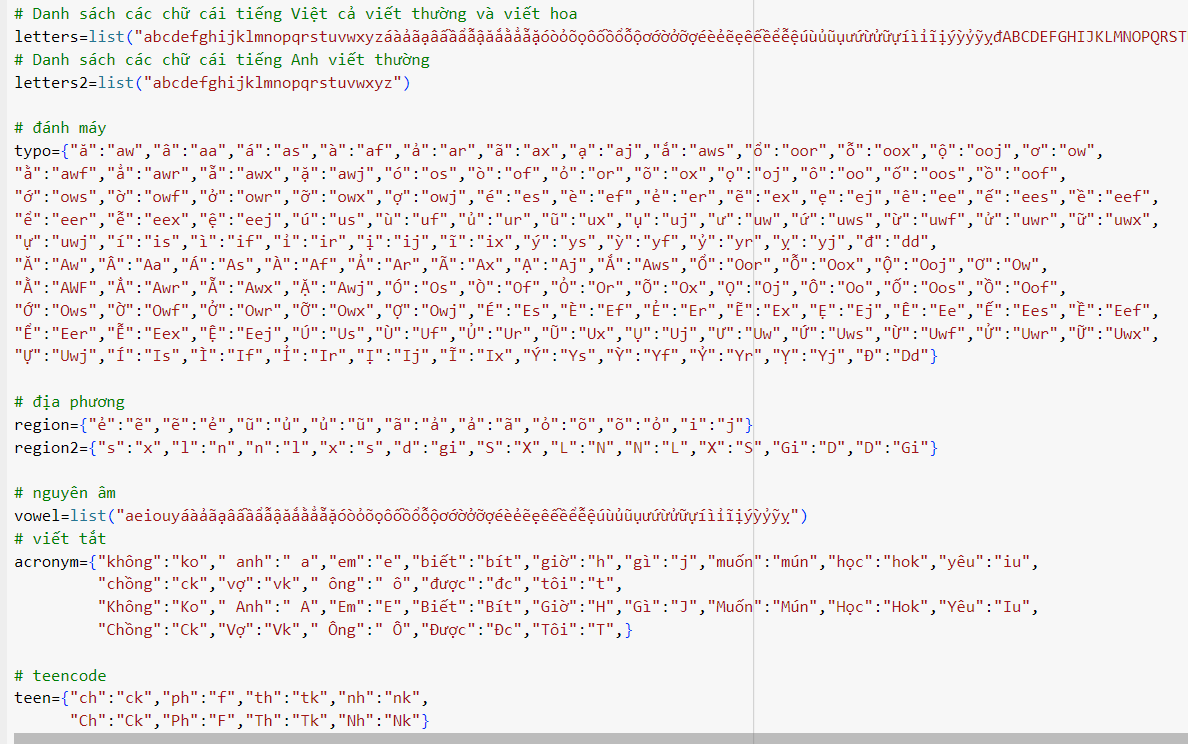
+ Lỗi sai chính tả thông thường khi gõ bàn phím.

+ Từ viết tắt.

+ Từ địa phương.

+ Teencode: Từ ngữ mới được sử dụng bởi thanh thiếu niên hiện nay.

Các biến dữ liệu phục vụ cho việc sinh lỗi chính tả trong danh sách các cụm từ trong n-gram. (Hình 4.5)

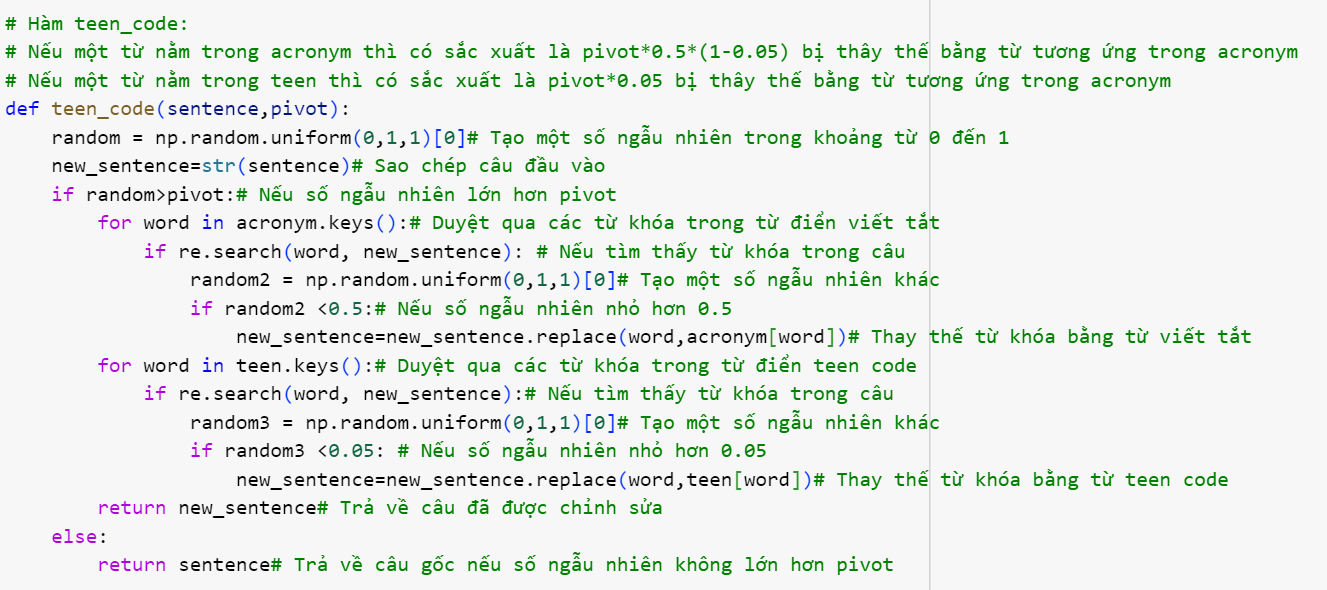


Hình 4.5: Hình ảnh dòng lệnh biến phục vụ việc sinh lỗi chính tả trong hàm add\_noise.

Hàm teen\_code(Hình 4.6). Với mổi từ trong sentence, ta có:

+ Nếu một từ nằm trong acronym thì có sắc xuất là pivot\*0.5\*(1-0.05) bị thay thế bằng từ tương ứng trong acronym

+ Nếu một từ nằm trong teen thì có sắc xuất là pivot\*0.05 bị thây thế bằng từ tương ứng trong acronym



Hình 4.6: Hình ảnh dòng lệnh hàm teen\_code.

Hàm add\_noise(Hình 4.7).

- Bước 1: Thực thi hàm teen\_code với pivot = 0.5

- Bước 2: Với mổi ký tự trong sentence. Phụ thuộc vào các điều kiện với một sắc xuất cụ thể ta sẽ:

+ Thay thế bằng phiên bản lỗi đánh máy

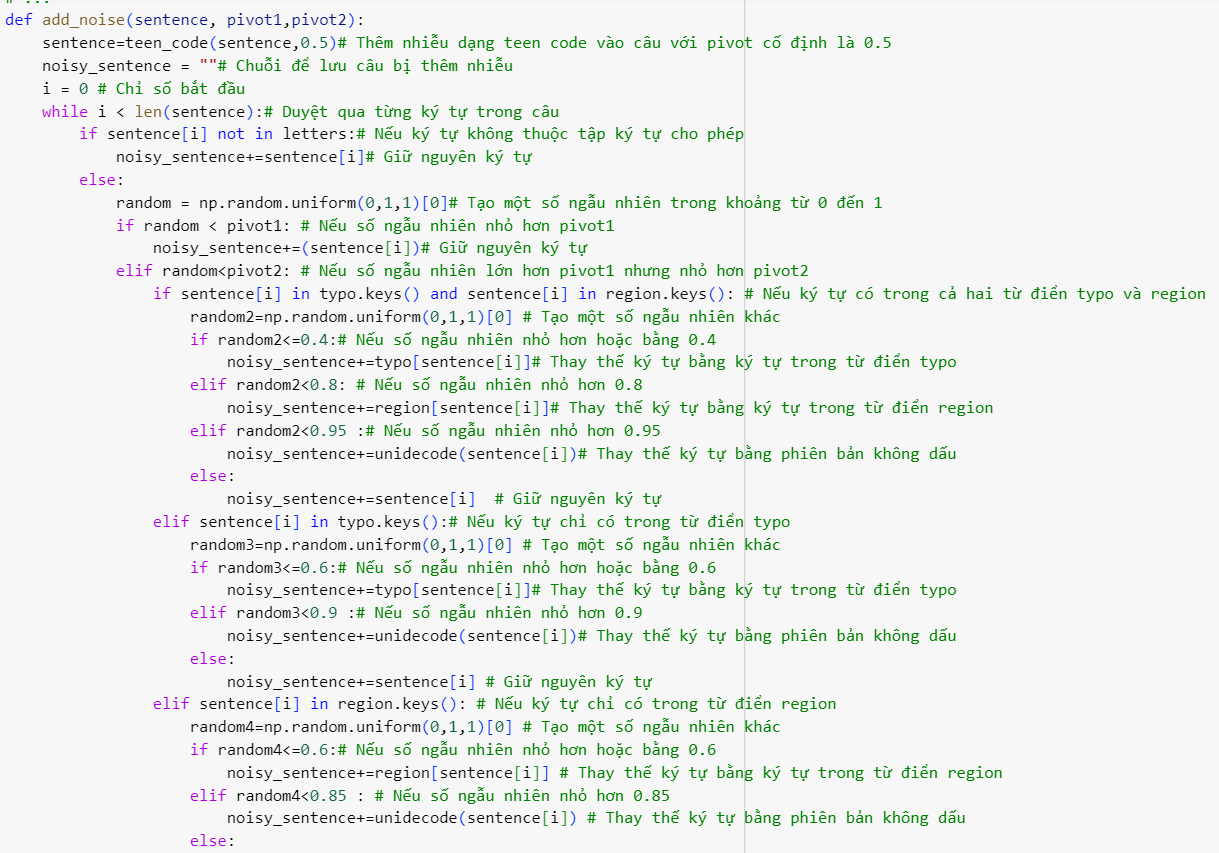
+ Thay thế bằng phiên bản địa phương

+ Thay thế bằng phiên bản không dấu

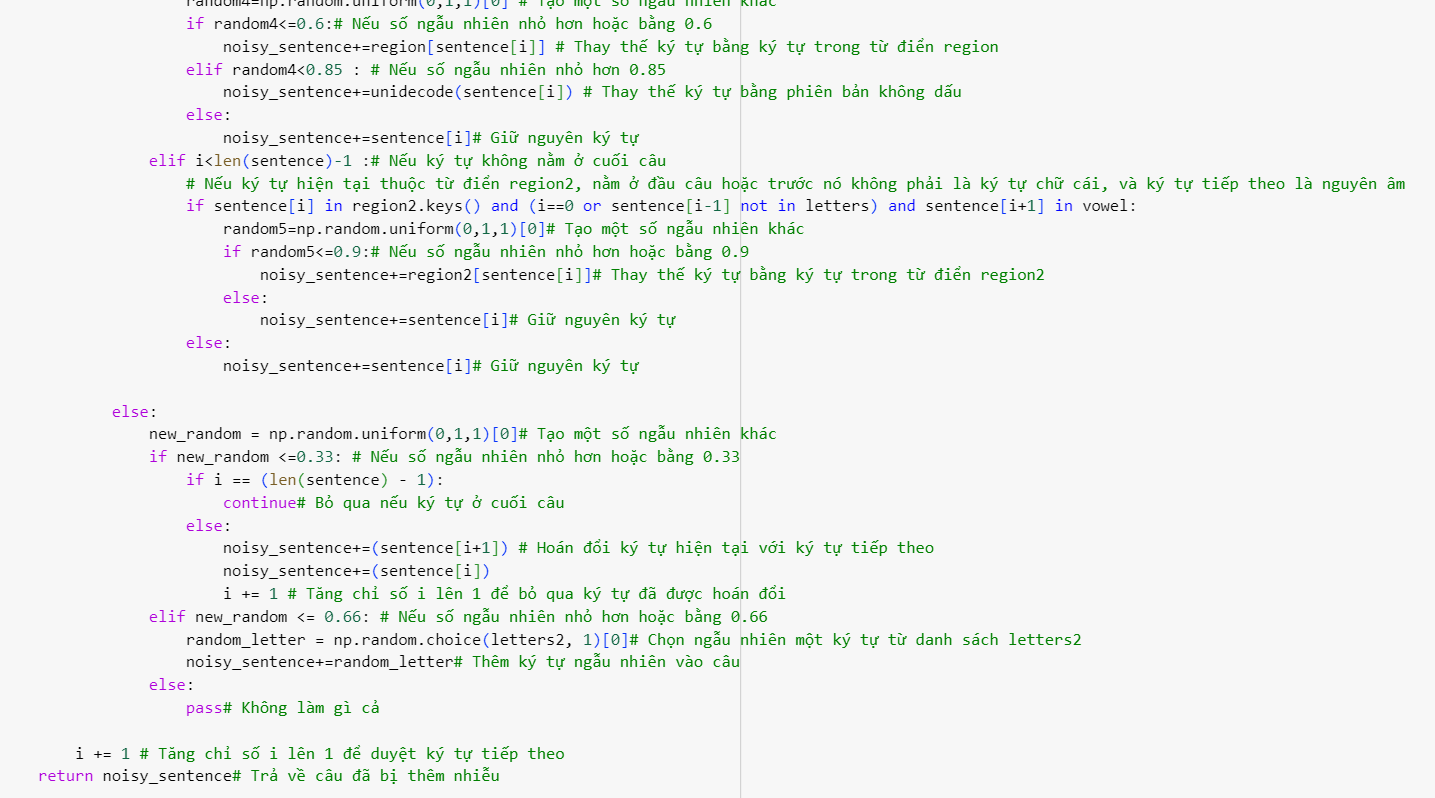
+ Giữ nguyên ký tự

+ Thay thế chử cái đầu bằng ký tự tiếng anh bất kỳ

+ Hoán đổi với kỹ tự phía trước và bỏ qua ký tự phía trước



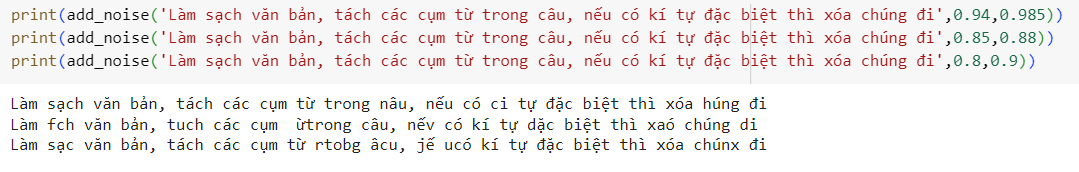
Hình 4.7.1: Hình ảnh dòng lệnh hàm add\_noise.



Hình 4.7.2: Hình ảnh dòng lệnh hàm add\_noise(tt).

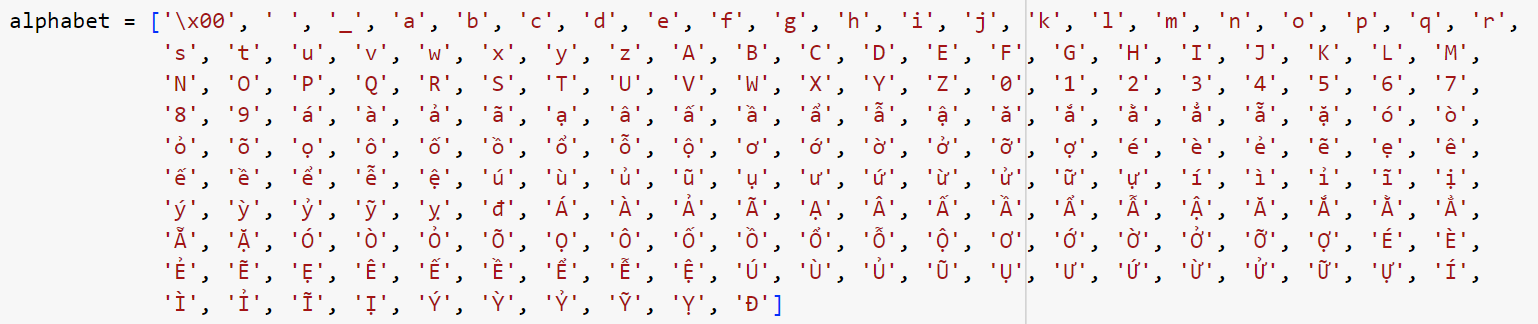
Ví dụ cho việc sử dụng lớp nhiễu như sau, với câu đầu tiên là câu gốc, và sâu thứ hai là câu sau khi đã đưa vào lớp nhiễu. (Hình 4.8)

* Original: "Làm sạch văn bản, tách các cụm từ trong câu, nếu có kí tự đặc biệt thì xóa chúng đi"
* Sau khi Add Noise: "Làm sạch văn bản, tách các cụm từ trong câu, nếu có kí tự đặc biệt thì xóa chúng đi"



Hình 4.8: Hình ảnh dòng lệnh test hàm add\_noise.

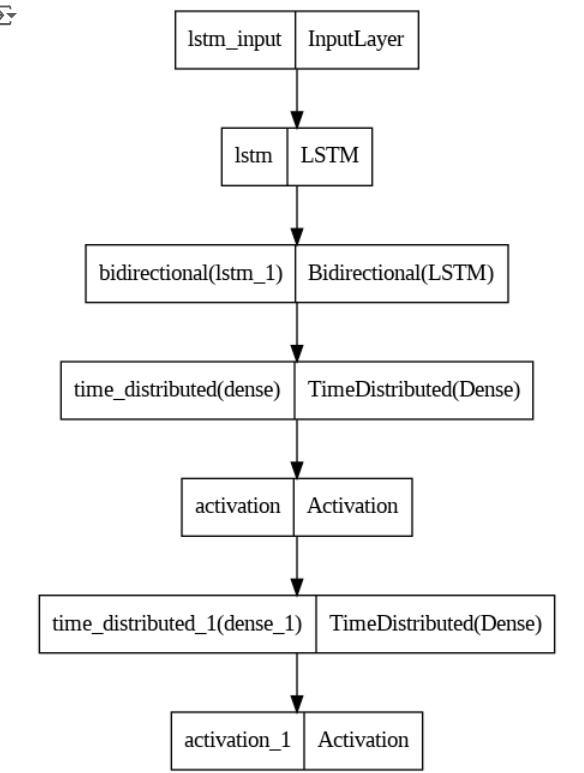
### 4.2.3 Vector hóa từ và văn bản

Bước này mục đích là vector hoá từ trong từng câu. Nhóm đã sử dụng phương pháp One-Hot Encoding để vector hóa từ và văn bản với bộ từ vựng alphabet (Hình 4.9).

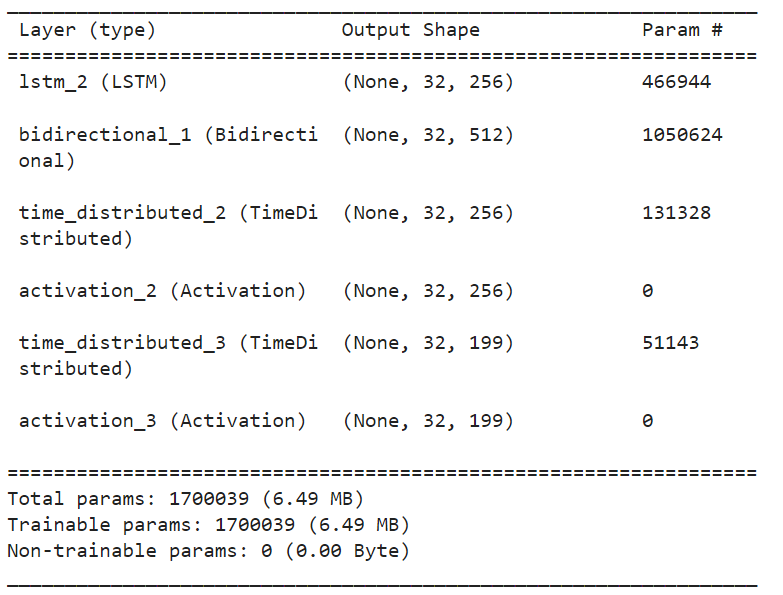
Hình 4.9: Bộ từ vựng dùng để One-Hot Encoding.

## 4.2 Xây dựng mô hình

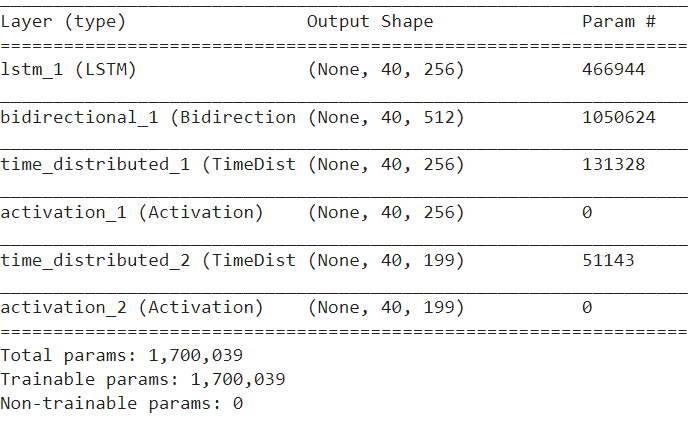
Mô hình Bi-directional LSTM (hình [4.6)](#_bookmark42) có khả năng nắm bắt được ngữ cảnh của đoạn văn bản đang xét, và lúc huấn luyện có thể lưu lại những thông tin cần thiết cho việc sửa lỗi văn bản. Đầu vào của mô hình là các chuỗi từ Bi-gram có lỗi sai chính tả và đầu ra là những Bi-gram từ đã được sửa lỗi. Đây là những lí do cho việc lựa chọn mô hình Bi-directional LSTM dạng many-to-many kết hợp với mô hình Bi-gram.



Hình 4.10: Hình ảnh mô hình Bi-directional LSTM.



Hình 4.11.1: Hình ảnh thông số mô hình B-LSTM với phương pháp 4-gram.



Hình 4.11.2: Hình ảnh thông số mô hình B-LSTM với phương pháp 5-gram.

## 4.3 Dùng mô hình và các phương pháp xử lý ngôn ngữ tự nhiên để sửa lỗi chính tả cho văn bản

**4.3.1 Hàm tách văn bản thành các cụm từ dài nhất có thể, cụm các ký tự đặt biệt**

|  |
| --- |
| def extract\_phrases(text):      pattern = r'\w[\w ]\*\w|\s\W+|\W+|\n'      return re.findall(pattern, text) |

Cụm từ ở đây là danh sách các từ dài nhất liền kề nhau bằng dấu cách

**4.3.2 Hàm sử dụng mô hình ngram và các phương pháp xử lý ngôn ngữ tự nhiên để sửa lỗi chính tả cho một cụm từ**

|  |
| --- |
| def correct(sentence):  # Lặp qua từng ký tự trong câu.  for i in sentence:  # Kiểm tra xem ký tự hiện tại có trong danh sách các ký tự được chấp nhận hay không.  # Nếu không, thay thế ký tự đó bằng khoảng trắng.  if i not in accepted\_char:  sentence = sentence.replace(i, " ")  # Kiểm tra số từ trong câu  num\_words = len(sentence.split())  # Nếu số từ không đủ để tạo n-gram, trả về câu không đổi  if num\_words < NGRAM:  ngrams = list(nltk\_ngrams(sentence, n=num\_words))  # Đoán các từ trong các n-gram.  guessed\_ngrams = list(guess(ngram) for ngram in ngrams)  # Tạo một danh sách các bảng ứng viên từ.  candidates = [Counter() for \_ in range(len(guessed\_ngrams) + num\_words - 1)]  # Lặp qua các n-gram đã đoán.  for nid, ngram in (enumerate(guessed\_ngrams)):  # Tách các từ từ n-gram.  for wid, word in (enumerate(re.split(' +', ngram))):  # Cập nhật bảng ứng viên từ với từ hiện tại.  candidates[nid + wid].update([word])  # Xây dựng câu đúng bằng cách chọn từ phổ biến nhất từ mỗi bảng ứng viên từ.  output = ' '.join(c.most\_common(1)[0][0] for c in candidates)  else:  # Tạo các n-gram từ câu đã được xử lý.  ngrams = list(nltk\_ngrams(sentence, n=NGRAM))  # Đoán các từ trong các n-gram.  guessed\_ngrams = list(guess(ngram) for ngram in ngrams)  # Tạo một danh sách các bảng ứng viên từ.  candidates = [Counter() for \_ in range(len(guessed\_ngrams) + NGRAM - 1)]  # Lặp qua các n-gram đã đoán.  for nid, ngram in (enumerate(guessed\_ngrams)):  # Tách các từ từ n-gram.  for wid, word in (enumerate(re.split(' +', ngram))):  # Cập nhật bảng ứng viên từ với từ hiện tại.  candidates[nid + wid].update([word])  # Xây dựng câu đúng bằng cách chọn từ phổ biến nhất từ mỗi bảng ứng viên từ.  output = ' '.join(c.most\_common(1)[0][0] for c in candidates)  return output |

**4.3.3 Hàm sử dụng mô hình ngram và các phương pháp xử lý ngôn ngữ tự nhiên để sửa lỗi chính tả cho một văn bản**

|  |
| --- |
| def check\_text(regex, text):      pattern = re.compile(regex)      if pattern.fullmatch(text):          return True      else:          return False  def correct\_2(text):    ep = extract\_phrases(text)    rs\_ = ""    for value in ep:      regex = r'\w[\w ]+'      if check\_text(regex, value):        rs\_ = rs\_ + correct(value)      else:        rs\_ = rs\_ + value    return rs\_ |

## 4.4 Đánh giá hai phương pháp 4-gram và 5-gram

### 4.4.1 Bộ dữ liệu dùng để đánh giá hai phương pháp và thông số của nó

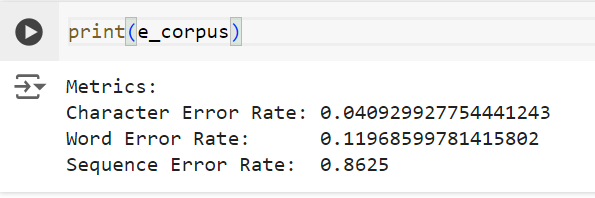
Vì phương pháp 4-gram, 5-gram chỉ khác nhau ở việc sửa lỗi chính tả ở cụm từ nên chúng ta hoàn toàn có thể sử dụng các câu thay cho các văn bản để làm bộ dữ liệu đánh giá cho hai phương pháp.

Bộ dữ liệu dùng để đánh giá hai phương pháp gồm 2000 câu được sinh ra tự động bằng hàm “add\_noise” (Hình 4.7). Bộ dữ liệu có các thông số như sau:

+ Character Error Rate: 0.040929927754441243

+ Word Error Rate: 0.11968599781415802

+ Sequence Error Rate: 0.8625



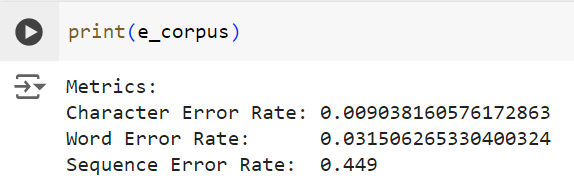
Hình 4.12 Thông số bộ dữ liệu dùng để đánh giá hai phương pháp

### 4.4.2 Thông số của kết quả đầu ra với phương pháp 4-gram với bộ dữ liệu ở 4.4.1

Character Error Rate: 0.009038160576172863

Word Error Rate: 0.031506265330400324

Sequence Error Rate: 0.449



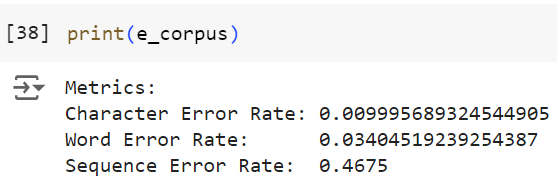
Hình 4.13 Thông số của kết quả đầu ra với phương pháp 4-gram với bộ dữ liệu ở 4.4.1

### 4.4.3 Thông số của kết quả đầu ra với phương pháp 5-gram với bộ dữ liệu ở 4.4.1

Character Error Rate: 0.009995689324544905

Word Error Rate: 0.03404519239254387

Sequence Error Rate: 0.4675



Hình 4.14 Thông số của kết quả đầu ra với phương pháp 4-gram với bộ dữ liệu ở 4.4.1

### 4.4.4 Nhận xét

Phương pháp 4-gram cho kết quả tốt hơn một chút so với phương pháp 5-gram. Điều này được thể hiện qua các thông số lỗi như CER, WER, và SER. Cụ thể:

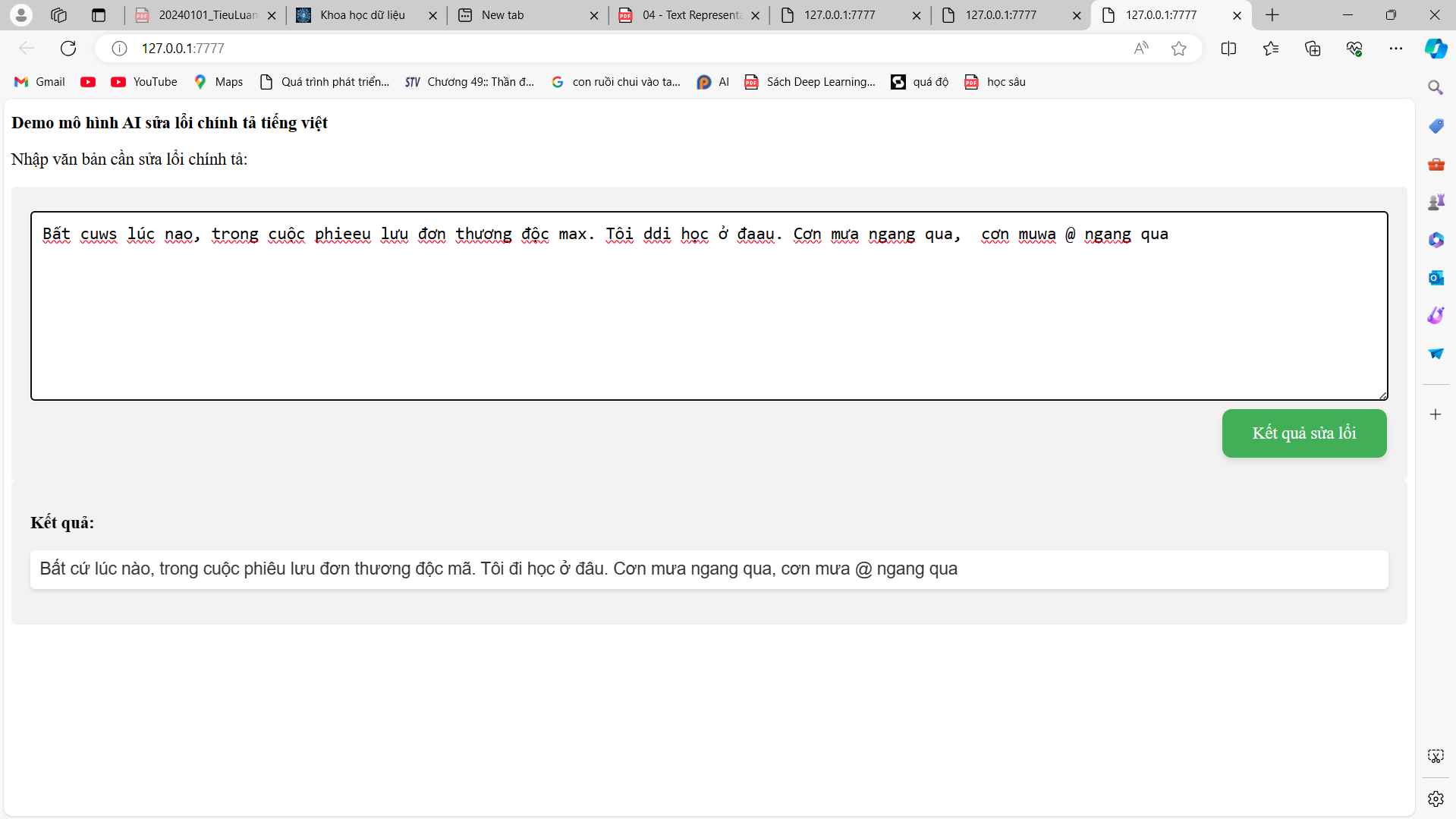
+ CER kết quả đầu ra của phương pháp 4-gram thấp hơn 5-gram (0.009038160576172863 so với 0.009995689324544905).

+ WER kết quả đầu ra của phương pháp 4-gram thấp hơn 5-gram (0.031506265330400324 so với 0.03404519239254387).

+ SER kết quả đầu ra của phương pháp 4-gram thấp hơn 5-gram (0.449 so với 0.4675).

Từ các thông số trên, có thể thấy rằng phương pháp 4-gram có độ chính xác cao hơn và hiệu quả hơn trong việc giảm thiểu lỗi khi so sánh với phương pháp 5-gram.

## 4.5 Sản phẩm



Hình 4.15: Hình ảnh website sửa lỗi chính tả.

# KẾT LUẬN

**1. Kết quả đạt được**

Sau một thời gian nghiên cứu và phát triển, nhóm đã đạt được các kết quả chính như sau:

- 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.

- Nhóm đã huấn luyện mô hình và xây dựng được website sửa lỗi chính tả

**2. Hướng phát triển**

Mặc dù đã đạt được những kết quả đáng khích lệ, nhưng nhóm vẫn nhận thấy một số hạn chế và tồn tại như sau:

- Độ chính xác của mô hình còn thấp so với các mô hình hiện hành.

Dựa trên những hạn chế và tồn tại nêu trên, nhóm dự kiến sẽ tiếp tục nghiên cứu và phát triển theo các hướng sau:

- 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. Áp dụng các phương pháp học sâu khác để cải thiện độ chính xác như: Attention, Beam search.

# TÀI LIỆU THAM KHẢO

[1] Do DT, Nguyen HT, Bui TN, Vo HD. PRICAI'21. VSEC: Transformer-based Model for Vietnamese Spelling Correction

[2] Tran, Hieu, et al. IEA/AIE'21. Hierarchical Transformer Encoders for Vietnamese Spelling Correction

[3] Dang TDA, Nguyen TTT. PACLIC'20. TDP – A Hybrid Diacritic Restoration with Transformer Decoder

[4] Nguyen HT, Dang TB, Nguyen LM. PACLING'19. Deep learning approach for vietnamese consonant misspell correction

[5] Pham Dinh Khanh. Ly thuyet ve mang lstm. 2019.

[6] VNTC/Data/10Topics/Ver1.1 at master · duyvuleo/VNTC · GitHub

[7] Khóa luận tốt nghiệp cử nhân CNTT, Tìm kiếm văn bản tiếng Việt theo chủ đề, Hoàng Công Duy Vũ – Nguyễn Lê Nguyên