TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHÓA LUẬN TỐT NGHIỆP**

**KIỂM TRA LỖI CHÍNH TẢ TIẾNG VIỆT VÀ ỨNG DỤNG TRÊN MS WORD**

*Người hướng dẫn*: : **PGS.TS. LÊ ANH CƯỜNG**

*Người thực hiện*: **LÊ NGỌC THÂN – 51303400**

**MAI ANH KIỆT – 51303318**

Lớp **: 13050303**

Khoá  **: 17**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2017**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHÓA LUẬN TỐT NGHIỆP**

**KIỂM TRA LỖI CHÍNH TẢ TIẾNG VIỆT VÀ ỨNG DỤNG TRÊN MS WORD**

*Người hướng dẫn*: : **PGS.TS. LÊ ANH CƯỜNG**

*Người thực hiện*: **LÊ NGỌC THÂN – 51303400**

**MAI ANH KIỆT – 51303318**

Lớp **: 13050303**

Khoá  **: 17**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2017**

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin chân thành gửi lời cảm ơn đến PGS. TS. Lê Anh Cường, giảng viên khoa Công Nghệ Thông Tin – trường Đại học Tôn Đức Thắng, người thầy đã tận tình giúp đỡ chúng tôi thực hiện khóa luận này trong học kỳ vừa qua. Thầy đã hướng dẫn chúng em bằng sự nhiệt thành của mình cũng như sẵn sàng trả lời bất kỳ thắc mắc nào của chúng tôi một cách đầy đủ nhất trong quá trình thực hiện khóa luận tốt nghiệp này.

Chúng em xin chân thành cảm ơn các thầy cô giáo trong trường Đại học Tôn Đức Thắng nói chung, các thầy cô trong khoa Công Nghệ Thông Tin nói riêng, đã dạy dỗ cho chúng em những kiến thức về các môn đại cương cũng như các môn chuyên ngành, và luôn quan tâm, giúp đỡ chúng em trong suốt quá trình học tập.

Cuối cùng, chúng em xin gửi những lời cảm ơn chân thành nhất đến gia đình, người thân và bạn bè chúng em, những người đã tạo điều kiện, quan tâm, giúp đỡ, động viên chúng em trong suốt thời gian qua.

**KHÓA LUẬN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm khóa luận của riêng chúng tôi và được sự hướng dẫn của PGS. TS. Lê Anh Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Lê Ngọc Thân*

*Mai Anh Kiệt*

TÓM TẮT

Một văn bản được soạn thảo đúng chính tả là một việc hết sức quan trọng nhất là trong các văn bản hành chính, pháp quy. Sai chính tả ở những văn bản hành chính như vậy sẽ gây hậu quả nghiêm trọng. Cùng với sự phát triển của công nghệ thông tin trong nhiều năm trở lại đây, hầu hết văn bản được soạn thảo trên các phần mềm máy tính. Nhưng hiện nay trong các trình soạn thảo văn bản phổ biến lại không tích hợp khả năng kiểm tra lỗi chính tả tiếng Việt một cách mạnh mẽ, và những phần mềm khác bên thứ ba cũng không thực sự có tiếng tăm. Đối với những văn bản, tài liệu dài người soạn thảo sẽ phải mất nhiều công sức để kiểm tra lỗi chính tả, thậm chí bỏ qua nhiều lỗi do nhầm lẫn. Do đó, một công cụ hỗ trợ kiểm tra lỗi chính tả chính xác và đưa ra những gợi ý tốt một cách tự động là điều cần thiết.

Khóa luận này nhằm mục đích nghiên cứu, đưa ra phương pháp giải quyết bài toán kiểm tra lỗi chính tả tiếng Việt và xây dựng hệ thống kiểm tra, gợi ý sửa lỗi chính tả tiếng Việt ứng dụng vào thực tế, thông qua mô hình ngôn ngữ N-gram. Hệ thống được tích hợp vào trình soạn thảo văn bản MS Word. Bằng cách sử dụng mô hình ngôn ngữ N-gram, từ điển tiếng Việt và kho ngữ liệu (corpus) kết hợp với các thuật toán, để hệ thống kiểm tra từ sai lỗi chính tả và đưa ra từ gợi ý. Qua thực nghiệm cho thấy, hệ thống đã đạt được những kết quả khả quan trong cả việc kiểm tra từ sai chính tả và gợi ý sửa lỗi. Điều này hứa hẹn về khả năng đưa hệ thống vào thực tiễn và được sử dụng rộng rãi là khá cao.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc488956326)

[TÓM TẮT iii](#_Toc488956327)

[MỤC LỤC 1](#_Toc488956328)

[DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT 4](#_Toc488956329)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 5](#_Toc488956330)

[CHƯƠNG 1 – GIỚI THIỆU 6](#_Toc488956331)

[1.1 Đặt vấn đề 6](#_Toc488956332)

[1.2 Phát biểu bài toán 6](#_Toc488956333)

[1.3 Mục tiêu và phạm vi của khóa luận 7](#_Toc488956334)

[1.4 Một số nghiên cứu liên quan 8](#_Toc488956335)

[1.5 Cấu trúc khóa luận 9](#_Toc488956336)

[CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT 10](#_Toc488956337)

[2.1 Đặc điểm của ngữ pháp tiếng Việt 10](#_Toc488956338)

[2.1.1 Đặc điểm của tiếng việt 10](#_Toc488956339)

[2.1.2 Các đơn vị của tiếng việt 10](#_Toc488956340)

[2.2 Chuẩn chính tả 12](#_Toc488956341)

[2.3 Các nguyên nhân gây ra lỗi chính tả trong tiếng Việt 13](#_Toc488956342)

[2.3.1 Lỗi do âm đầu 14](#_Toc488956343)

[2.3.2 Lỗi do âm chính 14](#_Toc488956344)

[2.3.3 Lỗi do âm cuối 14](#_Toc488956345)

[2.3.4 Lỗi do thanh điệu 14](#_Toc488956346)

[2.4 Các trường hợp lỗi chính tả trong văn bản tiếng Việt 14](#_Toc488956347)

[2.4.1 Lỗi do đánh máy sai 15](#_Toc488956348)

[2.4.2 Lỗi do phát âm 15](#_Toc488956349)

[2.5 Giới thiệu kho ngữ liệu 17](#_Toc488956350)

[2.5.1 Kho ngữ liệu là gì? 17](#_Toc488956351)

[2.5.2 Các bài toán cần xử lý trong quá trình tạo kho ngữ liệu 17](#_Toc488956352)

[2.5.3 Các bước tạo kho ngữ liệu cho chương trình 18](#_Toc488956353)

[2.6 Một số phương pháp kiểm lỗi chính tả 19](#_Toc488956354)

[2.6.1 Kiểm tra lỗi chính tả đối với tiếng nước ngoài 19](#_Toc488956355)

[2.6.2 Kiểm lỗi chính tả đối với tiếng Việt 21](#_Toc488956356)

[CHƯƠNG 3 – MÔ HÌNH NGÔN NGỮ N-GRAM 24](#_Toc488956357)

[3.1 Mô hình ngôn ngữ 24](#_Toc488956358)

[3.2 Mô hình ngôn ngữ N-gram 24](#_Toc488956359)

[3.2.1 Tính xác suất bằng N-gram 25](#_Toc488956360)

[3.2.2 Khó khăn khi xây dựng mô hình ngôn ngữ N-gram 26](#_Toc488956361)

[3.2.3 Các phương pháp làm mịn 27](#_Toc488956362)

[3.2.4 Phương pháp làm giảm kích thước dữ liệu 33](#_Toc488956363)

[CHƯƠNG 4 – GIỚI THIỆU PHƯƠNG PHÁP 37](#_Toc488956364)

[4.1 Mô hình tổng quát 37](#_Toc488956365)

[4.1.1 Phần xử lý văn bản 38](#_Toc488956366)

[4.1.2 Phần kiểm tra lỗi 39](#_Toc488956367)

[4.1.3 Phần sửa lỗi 39](#_Toc488956368)

[4.2 Xây dựng Add-in vào Word 44](#_Toc488956369)

[4.2.1 Tạo Word VSTO Add-in 44](#_Toc488956370)

[4.2.2 Tạo Ribbon (Visual Designer) 45](#_Toc488956371)

[4.2.3 Tạo User Control 47](#_Toc488956372)

[4.2.4 Làm việc với namespace Microsoft.Office.Interop.Word 48](#_Toc488956373)

[CHƯƠNG 5 – CÀI ĐẶT, THỰC NGHIỆM VÀ ĐÁNH GIÁ 52](#_Toc488956374)

[5.1 Cài đặt 52](#_Toc488956375)

[5.1.1 Xây dựng tập huấn luyện (Corpus) 52](#_Toc488956376)

[5.1.2 Xây dựng tập dữ liệu 52](#_Toc488956377)

[5.1.3 Viết chương trình 53](#_Toc488956378)

[5.2 Thực nghiệm 53](#_Toc488956379)

[5.2.1 Môi trường thực nghiệm 54](#_Toc488956380)

[5.2.2 Kết quả thực nghiệm 54](#_Toc488956381)

[5.3 Đánh giá 56](#_Toc488956382)

[CHƯƠNG 6 – TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN 57](#_Toc488956383)

[TÀI LIỆU THAM KHẢO 58](#_Toc488956384)

DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT

**CÁC KÝ HIỆU**

|  |  |
| --- | --- |
|  | Từ thứ i trong câu |
|  | Từ gợi ý thứ j của từ Wi |
|  | Xác suất của B dựa trên A |
| P(W1 W2 …Wn) | Xác suất từ từ W1 đến Wn |
| C(W1 W2 …Wn) | Tần số suất hiện của W1 đến Wn |
|  | Độ phát hiện chính xác |
|  | Độ hồi tưởng |
|  | Độ điều chỉnh chính xác |
|  | Độ đo F |

**CÁC CHỮ VIẾT TẮT**

|  |  |
| --- | --- |
| MS | Microsoft |
| HTML | HyperText Markup Language |
| XML | Extensible Markup Language |
|  |  |
|  |  |

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1. Mô hình tổng quát phương pháp kiểm lỗi 7](#_Toc488956389)

[Hình 2. Ví dụ cho mô hình Nagata 20](#_Toc488956390)

[Hình 3. Ví dụ mô hình N-gram kết hợp Viterbi 23](https://d.docs.live.net/3ce21a64bfca4749/Documents/Thesis/Thesis_VietnameseCheckingSpelling/report.docx#_Toc488956391)

[Hình 4. Mô hình tổng quát hệ thống kiểm lỗi chính tả 37](#_Toc488956392)

[Hình 5. Mô hình tổng quát tạo tập gợi ý 40](#_Toc488956393)

[Hình 6. Tạo Word Add-in 44](#_Toc488956394)

[Hình 7. Tạo Ribbon (Visual Designer) 45](#_Toc488956395)

[Hình 8. Thiết kế Ribbon 46](https://d.docs.live.net/3ce21a64bfca4749/Documents/Thesis/Thesis_VietnameseCheckingSpelling/report.docx#_Toc488956396)

[Hình 9. Tạo User Control 47](https://d.docs.live.net/3ce21a64bfca4749/Documents/Thesis/Thesis_VietnameseCheckingSpelling/report.docx#_Toc488956397)

[Hình 10. Thiết kế User Control 48](https://d.docs.live.net/3ce21a64bfca4749/Documents/Thesis/Thesis_VietnameseCheckingSpelling/report.docx#_Toc488956398)

[Hình 11. Kết quả thực nghiệm 55](#_Toc488956399)

**DANH MỤC BẢNG**

[Bảng 1. Các thành phần âm tiết tiếng Việt 12](#_Toc488956385)

[Bảng 2. Liệt kê các lỗi phát âm thường gặp 16](#_Toc488956386)

[Bảng 3. Độ lớn tập huấn luyện 52](#_Toc488956387)

[Bảng 4. Môi trường thực nghiệm 54](#_Toc488956388)

CHƯƠNG 1 – GIỚI THIỆU

1.1 Đặt vấn đề

Tiếng Việt là ngôn ngữ vốn phong phú và đa dạng. Trong việc sử dụng một ngôn ngữ, đảm bảo việc dùng chính xác từ ngữ là một điều quan trọng, nhất là đối với các văn bản pháp luật, hành chính nhà nước. Sai chính tả là điều rất phổ biến từ xưa đến nay. Hậu quả của việc sai lỗi chính tả cũng sẽ ảnh hưởng đến nhiều thứ phụ thuộc vào độ ảnh hưởng của văn bản, nếu văn bản là các văn bản hành chính, pháp luật thì độ chính xác cần phải được đảm bảo. Nguyên nhân của việc sai lỗi chính tả rất nhiều, phần nhiều là do phương ngữ hoặc do đánh nhầm trong quá trình soạn thảo văn bản.

Tuy nhiên, việc kiểm tra lỗi chính tả hầu hết do người soạn thảo tự kiểm tra. Sau khi hoàn thành một văn bản, người soạn thảo phải rà soát lại văn bản một cách kỹ lưỡng để tìm ra những sai sót trong quá trình đánh máy. Điều này rất khó khăn và tốn thời gian, đặc biệt là đối với những văn bản, tài liệu dài. Hiện này cũng có rất nhiều chương trình kiểm tra lỗi chính tả tiếng Việt, ngay cả trên trình soạn thảo MS Word cũng có thích hợp chương trình kiểm lỗi chính tả, nhưng độ hiệu quả và độ phổ biến của những ứng dụng này chưa cao.

Vì vậy, để có thể khắc phục được việc sai chính tả trong khi soạn thảo văn bản, cần một chương trình có thể kiểm tra lỗi tiếng Việt hiệu quả, tiện dụng cho người soạn thảo văn bản. Trong khóa luận này tập trung vào việc nghiên cứu, đưa ra các phương pháp và xây dựng một hệ thống kiểm tra lỗi chính tả và gợi ý sửa lỗi chính tả tiếng Việt, tích hợp vào trình soạn thảo văn bản MS Word.

1.2 Phát biểu bài toán

Bài toán kiểm tra lỗi chính tả là bài toán kiểm tra, phát hiện những từ sai lỗi chính tả trong văn bản và đưa ra gợi ý từ đúng. Lỗi chính tả được chia thành hai loại, đó là:

* Lỗi sai hoàn toàn âm tiết và không có trong từ điển tiếng Việt.

Ví dụ: iển ( từ đúng là: điển )

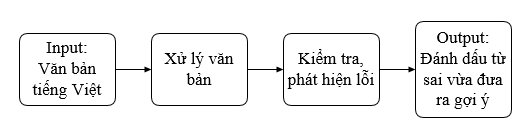
* Lỗi sai ngữ cảnh, âm tiết này có trong từ điển nhưng trong ngữ cảnh của câu hoặc đoạn văn đó thì không phù hợp.

Ví dụ: kiểm cha ( từ đúng là: kiểm tra )

Trong hệ thống kiểm tra lỗi chính tả sẽ có những chức năng chính sau:

* Xử lý văn bản: văn bản đầu vào sẽ được phân chia, lọc nhiều.
* Kiểm tra, phát hiện lỗi: tìm kiếm những âm tiết bị sai lỗi chính tả trong văn bản.
* Đánh dấu và gợi ý: đánh dấu nhưng từ lỗi và gợi ý âm tiết đúng để sửa lỗi.

Tổng quan chương trình được thể hiện ở hình 1.



Hình 1. Mô hình tổng quát phương pháp kiểm lỗi

1.3 Mục tiêu và phạm vi của khóa luận

Mục tiêu của khóa luận này là nghiên cứu, đưa ra phương pháp và xây dựng ứng dụng kiểm tra lỗi và gợi ý sửa lỗi chính tả, tích hợp vào ứng dụng soạn văn bản Microsoft Word, có thể sử dụng thực tế. Phương pháp được sử dụng trong khóa luận này là phương pháp kiểm lỗi chính tả cảm ngữ cảnh. Phương pháp này sẽ kiểm tra một từ và so sánh từ đó với các từ lân cận (ngữ cảnh của từ đó), bằng một số phương pháp kiểm tra, ta sẽ đánh giá từ đang xét có phù hợp ngữ cảnh hay không, nếu không thì từ đó là từ sai. Thông thường, các phương pháp kiểm tra lỗi truyền thống xem các từ độc lập với nhau và dùng từ điển để kiểm tra, nhưng khi đó sẽ không phát hiện được những từ sai ngữ cảnh. Bằng phương pháp kiểm tra lỗi chính tả cảm ngữ cảnh này sẽ giúp phát hiện được những lỗi sai về cách dùng, nghĩa là xét một từ mà từ này không sai chính tả khi đứng một mình, nhưng lại không đúng trong ngữ cảnh của câu văn đó thì việc sử dụng từ này là sai. Bằng phương pháp này, sẽ giúp chương trình đạt hiệu quả cao hơn trong việc kiểm tra lỗi.

Để đạt được mục tiêu trên, ngoài hiệu quả, thì còn cần một một mô hình ngôn ngữ không quá phức tạp và tốn nhiều không gian nhớ. Do đó mô hình ngôn ngữ N-gram được chọn làm hướng tiếp cận của khóa luận này.

Do ngôn ngữ là một lĩnh vực rộng và để phù hợp chương trình. Nên khóa luận này chỉ giới hạn ở các văn bản hành chính, các bài báo.

1.4 Một số nghiên cứu liên quan

Bài toán kiểm lỗi chính tả là bài toán nhận được sự quan tâm của rất nhiều nhà khoa học khắp thế giới từ lâu. Các ngôn ngữ phổ biến như tiếng Anh, tiếng Pháp, tiếng Nhật,…đều có rất nhiều phương pháp kiểm lỗi. Ở Việt Nam, bài toán này cũng đã có rất nhiều nghiên cứu và hệ thống kiểm lỗi chính tả, nhưng chưa có nhiều hệ thống đáp ứng nhu cầu người dùng và được sử dụng rộng rãi.

Đối với các ngôn ngữ nước ngoài như tiếng Anh, Pháp, Hoa hay Nhật có một số phương pháp như là winnow [9], từ ngữ cảnh[10], lai Bayes[11] cho tiếng Anh, mô hình Nagata cho tiếng Nhật[15], mô hình CInsunspell cho tiếng Hoa[13] …

Đối với tiếng Việt có một số cách như là dựa vào từ điển, sử dụng mô hình lưới từ, sử dụng mô hình N-gram kết hợp với phương pháp ước lượng Bayes để tính xác suất N-gram. Ngoài ra còn có nhưng nghiên cứu có liên quan như phương pháp phân tích cú pháp câu[6], phương pháp mạng lưới từ [7] hoặc sử dụng mô hình N-gram kết hợp thuật toán Viterbi[5].

1.5 Cấu trúc khóa luận

Khóa luận này được chia làm 6 chương:

Chương 1: Giới thiệu: Giới thiệu tổng quan về bài toán kiểm tra lỗi chính tả tiếng Việt, mục tiêu và phạm vi của khóa luận. Ngoài ra, phần này còn trình bày tình hình, một số phương pháp kiểm lỗi chính tả ở trong nước cũng như ngoài nước.

Chương 2: Cơ sở lý thuyết: Giới thiệu tổng quan về các đơn vị cũng như đặc điểm của tiếng Việt, chuẩn tiếng Việt và kho ngữ liệu (Corpus).

Chương 3: Mô hình ngôn ngữ N-gram: Trình bày chi tiết về mô hình ngôn ngữ N-gram.

Chương 4: Giới thiệu phương pháp: Nghiên cứu và đưa ra phương pháp để xây dựng hệ thống.

Chương 5: Cài đặt, thực nghiệm và đánh giá: Trình bày phương pháp cài đặt, thực nghiệm và đánh giá kết quả thực nghiệm độ hiệu quả của hệ thống

Chương 6: Tổng kết: Tổng hợp kết quả đã đạt được của khóa luận, đề xuất phương hướng phát triển trong tương lai.

CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT

2.1 Đặc điểm của ngữ pháp tiếng Việt

2.1.1 Đặc điểm của tiếng việt

Tiếng việt thuộc hệ ngôn ngữ Nam á và thuộc ngôn ngữ đơn lập, tức là mỗi một âm tiết được phát âm thanh tách rời nhau và được thể hiện bằng một chữ viết. Đặc điểm này thể hiện ở các mặt ngữ âm, từ vụng, ngữ pháp[6].

Đặc điểm ngữ âm: Trong tiếng Việt có một đơn vị được gọi là tiếng. Về mặt ngữ âm, mỗi tiếng được coi là một âm tiết. Hệ thống âm vị của tiếng Việt phong phú và có tính cân đối.

Đặc điểm từ vựng: Mỗi tiếng là một yếu tố có nghĩa. Tiếng là đơn vị cơ sở của hệ thống các đơn vị có nghĩa của tiếng Việt. Từ tiếng kết hợp hai phương thức ghép và láy, ta có thể tạo ra các đơn vị từ vựng khác để định danh sự vật, hiện tượng.

2.1.2 Các đơn vị của tiếng việt

2.1.2.1 Chữ cái

Chữ cái dùng để đại diện cho âm trong ngôn ngữ nói. Theo quy định của bộ Giáo dục và đào tạo, bảng chữ cái có 29 chữ cái bao gồm : A, Ă, Â, B, C, D, Đ, E, Ê, G, H, I, K, L, M, N, O, P, Q, R, S, T, U, V, X, Y.

Ngoài ra còn có 11 phụ âm ghép là: CH, GI, GH, KH, NH, NG, NGH, PH, QU, TH, TR.

2.1.2.2 Âm tiết

Âm tiết là một đơn vị cấu tạo nên một sự phối hợp trong tiếng nói**.** Âm gồm có các thành phần là: nguyên âm, phụ âm, bán âm, âm đệm [5].

Nguyên bao gồm có 12 nguyên âm đơn là A, Ă, Â, E, Ê, I, Y, O, Ô, Ơ, U, Ư và 3 loại nguyên âm đôi đó là IÊ, YÊ, IA, YA / ƯƠ, ƯA / UÔ, UA.

Phụ âm gồm có 23 phụ âm: B, C / K / Q, CH, D, Đ, G / GH, GI, H, KH, L, M, N, NH, NG / NGH, P, PH, R, S, T, TH, TR, V, X.

Bán Âm có hai loại và là bán âm cuối. Loại đầu tiên gồm hai chữ cái là I và Y. Loại thứ hai gồm hai chữ cái là O và U.

Âm đệm gồm có hai chữ cái O và U.

2.1.2.3 Dấu thanh

Tiếng Việt có 6 thanh [5] : ngang, sắc, huyền, hỏi, ngã, nặng và được biểu diễn bằng 5 dấu sắc, huyền, hỏi, ngã, nặng. Thanh ngang không có dấu ghi.

Sau đây là một số quy tắc đánh dấu thanh:

* Với những âm tiết chỉ có một chữ nguyên âm, thì dấu thanh được đặt vào chữ nguyên âm đó. Ví dụ: á à, ì ạch, ọ ẹ, ủ rũ, ọp ẹp, ục ịch, hà, lán, giá, giục, quả, quỹ, quỵt... (Chú ý: gi và qu được coi là phụ âm).
* Với những âm tiết, mà trong âm tiết đó chỉ cần có một chữ nguyên âm mang dấu phụ (Ă, Â, Ê, Ô, Ơ, Ư) và không kể kết thúc bằng chữ gì, thì dấu thanh bao giờ cũng đặt ở chữ đó (riêng ƯƠ, dấu đặt ở Ơ). Ví dụ: thuyền, trường…
* Với những âm tiết có hai chữ nguyên âm và kết thúc bằng một chữ phụ âm hoặc tổ hợp chữ phụ âm, thì dấu thanh được đặt vào chữ nguyên âm cuối cùng.
* Với những trường hợp còn lại thì dấu thanh được đặt vào con chữ nguyên âm áp chót.

2.1.2.4 Tiếng

Tiếng trong tiếng Việt còn được gọi là âm tiết [5]. Tiếng là đơn vị nhỏ nhất có thể có nghĩa. Một tiếng khi được người nói phát ra bao giờ cũng phải đi với một thanh điệu. Trong chữ viết, mỗi tiếng được ghi thành một chữ.

Cấu tạo của một tiếng bao gồm: phụ âm đầu, nguyên âm, phụ âm cuối, dấu thanh.

|  |  |
| --- | --- |
| Phụ âm đầu | b c d đ g h k l m n q r s t v x ch gh gi kh ng nh ph qu th tr ngh |
| Nguyên âm | a â ă e ê i o ô ơ u ư y ai ao au ay âu ây eo êu ia iu iê oa oi oe oă oo ôi ơi ua uy ui uâ uô uê uơ ưa ưi ươ ưu yê iêu oai oao oay oeo uôi uây uyê ươi ươu uya uyu uêu yêu |
| Phụ âm cuối | c p t m n ch ng nh |
| Dấu thanh | ngang, huyền, hỏi, ngã, sắc, nặng |

Bảng 1. Các thành phần âm tiết tiếng Việt

2.1.2.5 Hình vị

Hình vị là đơn vị nhỏ nhất có ý nghĩa, tương tự với tiếng. Hình vị cũng có cấu tạo là một âm tiết. Trên chữ viết hình vị được viết thành một chữ.

Trong tiếng Việt hình vị có thể đóng vai trò là một từ hoặc cũng có thể làm thành tố cấu tạo nên một từ, nhưng nó chỉ được phân xuất ra nhờ phân tích bản thân các từ.

Trong giáo trình Xử lý ngôn ngữ tự nhiên của TS. Đinh Điền[2], bên cạnh đặc điểm như hình vị của ngôn ngữ học đại cương, còn phải có “hình tố”.

2.1.2.6 Câu

Câu là đơn vị ở bậc cao nhất. Được cấu tạo bởi nhiều từ theo những quy tắc nhất định [5].

2.2 Chuẩn chính tả

Theo [1], chuẩn chính tả bao gồm các chuẩn viết các âm (phụ âm, nguyên âm) và các thanh, chuẩn viết tên riêng (viết hoa), chuẩn viết phiên âm từ và các thuật ngữ vay mượn (tiếng nước ngoài).

Hiện nay, chuẩn viết các âm và thanh tiếng Việt đã được xác định theo hệ thống ngữ âm của tiếng Việt. Chuẩn viết hoa chưa được thống nhất nhưng theo xu hướng viết hoa chữ cái đầu của âm tiết thuộc tên riêng (Ví dụ: Việt Nam, Hồ Chí Minh…). Đối với chuẩn phiên âm từ vay mượn (tiếng nước ngoài) thì phức tạp hơn. Hiện nay đang tồn tại 2 cách viết phiên âm, đó là phiên âm âm tiết và có gạch nối giữa các tiếng (Ví dụ: Lê-nin, Các-Mác…) và viết liền âm tiết, tôn trọng âm tiết và chữ theo hệ Latinh (Ví dụ: Paris, London…).

Chính tả là cách viết chữ được xem là chuẩn, tức viết đúng âm đầu, đúng vần, đúng thanh, đúng quy định về viết hoa, viết tắt, viết thuật ngữ. Ngoài ra, việc bỏ dấu thanh phải tuân theo 4 quy cách sau:

Dấu chỉ ghi trên hoặc dưới nguyên âm, không ghi trên hoặc dưới phụ âm. (Ví dụ đúng: láng, bệnh, mình)

Dấu chỉ ghi trên hoặc dưới nguyên âm chính, không ghi trên hoặc dưới bán âm cuối. (Ví dụ đúng: sáu, máy – Ví dụ sai: saú, maí)

Đối với nguyên âm đôi :

Khi nguyên âm đôi đứng ở cuối từ, ta ghi dấu trên hoặc dưới nguyên âm thứ nhất. Ví dụ: mía, táo

Khi nguyên âm đôi đứng ở giữa từ, ta ghi dấu trên hoặc dưới nguyên âm thứ hai: liền, muốn, hướng.

Chuẩn chính tả còn quy định tiếng phải được viết đúng thanh điệu. Hiện nay cư dân ở thành phố không phân biệt được hai thanh hỏi ngã. Tuy chỉ có hai dấu nhưng số lượng lỗi này không ít mà rất phổ biến, kể cả ở những người có trình độ văn hóa cao. Ngoài ra, chuẩn chính tả còn yêu cầu âm tiết viết đúng âm đầu, âm chính và âm cuối. Hiện nay, do việc phát âm của từng vùng miền khiến cho các âm này bị nhầm lẫn với nhau như ch-tr, l-n, an/ang…

2.3 Các nguyên nhân gây ra lỗi chính tả trong tiếng Việt

Nguyên nhân dẫn đến lỗi rất đa dạng. Lỗi thường xảy ra do phát âm sai dẫn đến viết theo phát âm nên cũng sai, hoặc trường hợp khác là sai do gõ nhầm khi đánh máy. Khóa luận này chỉ xét về lỗi do phát âm, bởi vì lỗi do đánh máy nguyên nhân thường là do người soạn thảo văn bản gõ nhầm nên không có loại lỗi cụ thể.

2.3.1 Lỗi do âm đầu

Lỗi này thường xảy ra đối với khu vực miền Bắc và miền Nam. Khu vực miền Bắc thường sai ở những âm như L\N, còn miền Nam thường sau các âm như S\X, Ch\Tr, D\Gi,…Ngoài ra, trong tiếng Việt có một số âm đọc giống nhau nhưng lại cố nhiều cách ghi. Ví dụ như âm được đọc là/k/ có các cách ghi C, K.

2.3.2 Lỗi do âm chính

Có 2 nguyên nhân gây nên sự nhầm lẫn âm chính. Nguyên nhân đầu tiên là do quy định ghi âm đặc biệt của chữ Quốc ngữ. Nguyên nhân thứ hai là cách phát âm lẫn lộn, đặc biệt là vùng Nam Bộ, đối với các âm chính trong hầu hết các vần. (Ví dụ: đạp 🡪 độp).

2.3.3 Lỗi do âm cuối

Lỗi này thường xảy ra ở khu vực miền Nam. Khu vực này Nam rất dễ bị nhầm lẫn giữa các âm cuối n/ng/nh và t/c/ch. Mặt hai bán âm cuối là /j/-/u/ được ghi bằng 4 chữ cái i-y, u-o rất dễ bị nhầm lẫn.

2.3.4 Lỗi do thanh điệu

Lỗi này thường xảy ra đối với khu vực miền Trung và miền Nam. Ở hai khu vực này không thể phân biệt được giữa dấu hỏi và dấu ngã.

2.4 Các trường hợp lỗi chính tả trong văn bản tiếng Việt

Có hai loại lỗi chính tả chính tả. Loại lỗi thứ nhất là lỗi do sai âm tiết và hoàn toàn không có trong từ điển. Loại lỗi thứ hai là âm tiết đúng và có trong từ điển, nhưng đặt trong ngữ cảnh của câu văn thì nó lại sai.

Loại lỗi thứ nhất rất dễ để tìm ra được lỗi và sửa lỗi. Đơn giản chỉ là cần sử dụng từ điển để tìm từ sai. Nhưng trong các văn bản tiếng Việt có một điều làm cho việc tìm lỗi này tương đối phức tạp thêm là do trong các văn bản đó có sử dụng từ mượn từ tiếng nước ngoài. Do vậy cần một chuẩn để xác định tiếng nước ngoài.

Loại lỗi thứ hai rất khó để tìm ra được lỗi này. Bởi vì máy tính thì khó mà hiểu được nghĩa của một văn bản. Do đó cần phải xét các tập nhầm lẫn âm tiết để tìm lỗi. Tập nhầm lẫn âm tiết là tập hợp các âm tiết có thể bị nhầm lẫn với âm tiết đang xét. Ví dụ: Xét từ “tìm kiếm”, tập nhầm lẫn có thể có các trường hợp là: tiềm kiếm, tìm kiếm, tìm kiến…

Có 2 trường hợp lỗi chính khi soạn thảo văn bản đó là do đánh máy và phát âm.

2.4.1 Lỗi do đánh máy sai

Đây là lỗi chính tả phổ biến, hầu hết lỗi ở mức độ âm tiết. Lỗi sai chính tả này gây ra các lỗi sai chính tả đơn và sai chính tả phức. Có bốn loại lỗi chính tả đơn:

* Chèn: “tín” 🡪 “tính” (chèn dư thêm chứ “h”).
* Xóa: “khoảng” 🡪 “khoản” (xóa chữ “g”).
* Thay thế: “tre” 🡪 “che” (thay thế chữ “tr” thành chữ “ch”)
* Hoán vị: “chính” 🡪 “chíhn” (hoán vị chữ “nh” thành “hn”).

Lỗi chính tả phức là kết hợp các lỗi trên.

2.4.2 Lỗi do phát âm

Lỗi do phát âm xảy ra nguyên nhân là do có thói quen “đọc thế nào thì viết như thế đó”. Nên lỗi này dễ xảy ra ở cả văn bản viết tay và đánh máy, cụ thể 2 loại lỗi [3] như sau:

* Khi chữ viết phân biệt âm tiết mà phát âm theo một phương ngữ nào đó lại không phân biệt. Cho nên với những phương ngữ khác nhau có những vấn đề chính tả khác nhau. Trong khi với người nói phương ngữ miền Bắc có các vấn đề chính tả “viết ch- hay tr-”, “viết -iêu hay -ươu”, v.v. thì với người nói phương ngữ miền Nam lại có các vấn đề “viết -n hay -ng”, “viết dấu hỏi hay dấu ngã”, v.v.
* Khi chữ viết phân biệt âm tiết mà phát âm tiếng Việt ngày nay không còn phân biệt, đó là trường hợp “viết d- hay gi-”, một vấn đề chính tả chung cho mọi miền trong cả nước.

Ví dụ: Âm tiết “nóng” 🡪 Tập nhầm lẫn: “lóng”

Âm tiết “tre” 🡪 Tập nhầm lẫn: “che”

Âm tiết “sữa” 🡪 Tập nhầm lẫn: “sửa”, “xữa”

Âm tiết “siêu” 🡪 Tập nhầm lẫn: “siu”, “sươu”

Sau đây chúng tôi sẽ liệt kê một số lỗi phát âm thường gặp [5]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CH- | TR- | | | |
| D- | GI- | | | |
| D- | GI- | NH- | | |
| D- | GI- | R- | | |
| D- | GI- | V- | | |
| HW- | NG- | QU- | | W- |
| L- | N- | | | |
| S- | X- | | | |
| -C | -T | | | |
| -N | -NG | | | |
| -AI | -AY | | | |
| -EM | -ÊM | | | |
| -ÊCH | -ÊT | | | |
| -IÊM | -IM | | | |
| -IÊU | -IU | | | |
| -IÊU | -ƯƠU | | | |
| -OAI | -OI | | | |
| -OM | -ÔM | | -ƠM | |
| Hỏi | ngã | | | |
| Ngã | Nặng | | | |

Bảng 2. Liệt kê các lỗi phát âm thường gặp

2.5 Giới thiệu kho ngữ liệu

Kho ngữ liệu (Corpus) là một tài nguyên quan trọng trong các bài toán xử lý ngôn ngữ tự nhiên bằng thống kê. Tùy vào mỗi bài toán mà có những loại kho ngữ liệu khác nhau. Chất lượng của kho ngữ liệu là điều quan trọng cần quan tâm, bởi vì nó ảnh hưởng rất lớn tới độ chính xác của hệ thống.

2.5.1 Kho ngữ liệu là gì?

Kho ngữ liệu là tập hợp văn bản, ngôn ngữ được số hóa. Nó có thể ở dạng văn bản hoặc âm thanh. Trong khóa luận này loại kho ngữ liệu sẽ được sử dụng là kho ngữ liệu văn bản.

2.5.2 Các bài toán cần xử lý trong quá trình tạo kho ngữ liệu

2.5.2.1 Xử lý văn bản

Do dữ liệu thu thập được phụ vào nguồn của kho ngữ liệu (corpus) nên dữ liệu gặp được có thể ở các dạng khác nhau. Do đó cần có công cụ phù hợp để trích tách văn bản thô từ dữ liệu ban đầu. Nguồn Corpus để thu thập thường sẽ là từ Internet, mà hầu hết các file chứa văn bản là HTML hoặc XML, khi đó sẽ cần chương trình đọc loại file này và lấy văn bản bên trong đó.

Sau khi được lấy được văn bản, tiếp đó sẽ chuyển những văn bản này thành một bộ mã thống nhất.

2.5.2.2 Cắt câu

Các câu sẽ được cắt dựa vào các dấu hiệu như “.”, “?”, “!”. Nhưng đối với một số ký tự có nhiều chức năng khác nhau, đặc biệt có thể thấy là dấu chấm, do dữ liệu thu thập được bao gồm rất nhiều thứ như những từ viết tắt (như Mr. A), địa chỉ Internet, … Vì vậy cần phải xác định chính xác ký hiệu nào là kết thúc câu để đảm bảo chính xác dữ liệu thu thập được.

2.5.2.3 Phân tích từ tố

Đây là quá trình tách các câu trong văn bản thành các đơn vị được gọi là từ tố ( mỗi từ tố có thể là từ, số,…).

Ví dụ:

Cho câu đầu vào: “Nước biển có màu xanh”.

Kết quả sau khi tách từ tố: “Nước”, “biển”, “có”, “màu”, “xanh”.

2.5.2.4 Phân đoạn từ

Trong tiếng Việt, ranh giới giữa các từ không rõ ràng. Trong mỗi câu, có rất nhiều khả năng tách từ khác nhau do sự đa dạng trong việc kết hợp với nhau giữa các từ. Vì thế bài toán càng trở nên khó khăn.

Ví dụ: Cho câu: “Khả năng lãnh đạo tốt”.

Câu này có các khả năng tách từ như sau:

Khả / năng / lãnh / đạo / tôt.

Khả năng / lãnh / đạo / tôt.

Khả / năng / lãnh đạo / tôt.

2.5.3 Các bước tạo kho ngữ liệu cho chương trình

Tập ngữ liệu (corpus) là phần quan trọng để chương trình có thể đạt hiệu quả cao trong việc kiểm lỗi nên việc xây dựng một Corpus tốt là điều hết sức cần thiết. Các bước tạo kho ngữ liệu bao gồm:

* Tải file text từ Internet: internet là nguồn tài nguyên phong phú, thu thập tập ngữ liệu từ các trang uy tín và đa dạng giúp tập ngữ liệu đạt chất lượng cao hơn.
* Lấy text: bước tiếp theo sẽ bóc tách phần text ra khỏi các thẻ html.
* Chuẩn hóa dữ liệu: ở bước này, tập ngữ liệu sẽ được loại bỏ những đoạn text kém chất lượng, xử lý nhiễu,…

2.6 Một số phương pháp kiểm lỗi chính tả

2.6.1 Kiểm tra lỗi chính tả đối với tiếng nước ngoài

2.6.1.1 Phương pháp từ ngữ cảnh

Ngữ cảnh của một từ, hay một đoạn văn là những từ, những đoạn xung quanh nó. Để quyết định chọn một từ với những từ nhập nhằng với nó thì những từ xung quanh nó là một đầu mối quan trọng. Ví dụ, ta xét hai lựa chọn giữa desert và dessert, nếu có các từ lân cận là sand, arid, sun,…ta sẽ chọn desert, còn nếu ta gặp những từ như delicious, chocolate,…ta sẽ chọn dessert. Khi còn trong giai đoạn học, hệ thống sẽ lưu những từ lân cận của mỗi từ sai khi những từ này bị hệ thống phát hiện. Áp dụng thực tế, khi phát hiện từ sai, hệ thống sẽ kiểm tra những từ lân cận từ đó và chọn từ thích hợp để sửa lỗi.

2.6.1.2 Mô hình CInsunSpell cho tiếng Hoa

Mô hình CInsunSpell [13] được xây dựng bởi Li Juanhua và Wang XiaoLong và được áp dụng trong sửa lỗi chính tả tiếng Hoa. Mô hình là sự kết hợp giữa N-gram, phương pháp ước lượng xác suất Bayes và phân phối trọng số tự động. Trong mô hình này, N-gram được sử dụng trong bước kiểm lỗi. CInsunSpell sử dụng trigram, xét xem độ liên kết của từ trung tâm và 4 tiếng lân cận nó. Tiếng trung tâm được coi như sai chính tả nếu như độ liên kết của nó với 4 tiếng lân cận nhỏ hơn một ngưỡng nào đó.

Tuy nhiên mô hình này cũng có nhược điểm và đó cũng là nhược điểm chung của mô hình N-gram, đó là dữ liệu quá thưa thớt, không đầy đủ, dẫn đến các xác suất N-gram sẽ có giá trị rất thấp (gần bằng 0). Để giải quyết vấn đề trên, mô hình CInsunSpell đã sử dụng thêm bigram kèm theo phương pháp làm mịn để nâng cao hiệu suất phát hiện lỗi.

2.6.1.3 Phương pháp Nagata cho tiếng Nhật

Phương pháp Nagata được đề xuất bởi Misaaki Nagata [15]. Ban đầu mô hình được áp dụng cho các hệ thống nhận dạng chữ viết. Sau đó, mô hình này được áp dụng để sửa lỗi chính tả. Phương pháp gồm 2 bước:

* Bước 1: Câu đầu vào được chia thành nhiều chuỗi tiếng, mỗi chuỗi tiếng tạo thành một từ có trong từ điển hoặc gần giống với một từ nào đó trong từ điển.
* Bước 2: Bằng cách sử dụng phương pháp thống kê, hệ thống chọn ra những câu tốt nhất được chọn để sửa lỗi.

Ví dụ: Con ngừ Việt Nam rất tôt bụng.

dất

Việt Nam

Con

người

ngùi

ngườ

Com

ngừ

Con người

rất

tốt

tố

tót

tốt bụng

bụng

Hình 2. Ví dụ cho mô hình Nagata

Hệ thống sử dụng hai phương pháp để chia chuỗi đầu vào thành các chuỗi con một cách hợp lý đó là mô hình ngôn ngữ thống kê và thuật toán Forward-DB Backward A\* để tách từ. Trong đó, ngôn ngữ thống kê dùng để tính xác suất kết hợp của chuỗi con và chuỗi nhãn từ loại và thuật toán Forward-DB Backward A\* [14] được dùng để đánh giá các xác suất kết hợp và tìm ra cách kết hợp có xác suất lớn nhất làm lời giải bài toán tách từ.

2.6.2 Kiểm lỗi chính tả đối với tiếng Việt

Kiểm lỗi chính tả là một bài toán khó đối với ngôn ngữ đơn lập như tiếng Việt. Phần này sẽ giới thiệu một số phương pháp kiểm lỗi chính tả phổ biến trong tiếng Việt.

2.6.2.1 Kiểm lỗi chính tả dựa trên từ điển

Đây là phương pháp kiểm lỗi chính tả đơn giản nhất. hai ý tưởng chính của phương pháp này như sau [6]:

* Có khả năng một âm tiết là lỗi nếu tồn tại âm tiết trong tập nhầm lẫn của nó kết hợp với các âm tiết xung quanh tạo thành từ ghép.
* Nếu câu không chia được thành dãy từ thì có khả năng âm tiết ở vị trí bị ngắt là lỗi.

Ví dụ 1:

“Công ty đạt *tổn* doanh số rất cao trong năm nay”

Dễ dàng thấy được từ “tổn” không thuộc về một từ nào cả, do đó có khả năng đó là âm tiết lỗi.

Ví dụ 2:

“Máy *vu* tính ngày cang phổ biến trên thế giới”

Xét tập nhầm lẫn của âm tiết “vu”, ta thấy có âm tiết “vi” kết hợp với các âm tiết “máy” và “tính” tạo thành từ “máy vi tính”, vậy có khả năng “vu” là sai.

2.6.2.2 Kiểm lỗi chính tả bằng mô hình lưới từ

Mô hình này hoạt động dựa vào việc tách từ mờ, sau đó đánh giá các cách tách từ khác nhau để tìm ra cách tách từ tốt nhất để làm câu sửa lỗi [7]. Trong lưới từ là một cấu trúc dữ liệu đặc biệt, giống như là một đồ thị có hướng và không chu trình xuất phát từ âm tiết đầu tiên đến âm tiết cuối cùng của câu. Nhờ có thể áp dụng những thuật toán tìm đường đi trong cấu trúc này làm cho việc tách từ mờ nhanh chóng hơn.

Về mặt lý thuyết thì lưới từ là một mô hình rất chặt chẽ và nếu được huấn luyện tốt thì nó có thể mang lại hiệu quả rất cao. Mặt khác, mô hình này cũng có nhược điểm đó là sự bùng nổ tổ hợp. Tiếng Việt khá đa dạng và phong phú nên việc tách từ cũng không hề đơn giản. Điều này làm cho việc tách từ tốn thời gian dẫn đến hệ thống của chương trình cũng trở nên chậm chạp hơn.

Trong mô hình này, tác giả còn sử dụng N-gram để ước lượng những giá trị cần thiết. Nhưng trở ngại do dữ liệu huấn luyện chưa đủ tốt nên việc ước lượng chưa cho được độ chính xác.

2.6.2.3 Kiểm lỗi chính tả bằng mô hình ngôn ngữ N-gram

Các tác giả trong [16] đã áp dụng mô hình ngôn ngữ n-gram để kiểm tra lỗi chính tả tiếng Việt. Mô hình được tạo ra dựa trên cách tiếp cận thống kê. Vấn đề chung của mô hình này là dữ liệu, cần phải có dữ liệu đủ lớn để mô hình đạt hiệu quả. Tác giả sử dụng kho ngữ liệu (corpus) để huấn luyện cho chương trình. Kho ngữ liệu này được thu thập từ các tài nguyên có chủ đề khác nhau để giảm số lượng từ chưa biết. Nó còn giúp gia tăng khả năng kết hợp giữa các âm tiết. Thay vì sử dụng thông tin ngôn ngữ học như phân đoạn từ, thẻ POS,... một kho ngữ liệu lớn sẽ cung cấp đầy đủ thông tin để có thể tìm lỗi và sửa lỗi.

Kết quả của chương trình đạt hiệu quả tốt đối với các văn bản chính thức như các văn bản hành chính, báo,… nhưng không đạt hiệu quả tốt với dữ liệu của mạng xã hội. Điều này làm giới hạn khả năng của hệ thống khi áp dụng thực tế.

2.6.2.4 Kiểm lỗi chính tả bằng mô hình ngôn ngữ và thuật toán viterbi

Hệ thống được tác giả Nguyễn Hữu Tiến Quang [5] xây dựng dựa mô hình n-gram kết hợp với thuật toán viterbi để tìm lỗi chính tả. Do sử dụng mô hình n-gram nên hệ thống này cũng phụ thuộc vào kho ngữ liệu (corpus).

Khi văn bản đầu vào được xử lý, hệ thống sẽ chuẩn hóa văn bản đầu vào thành dạng văn bản text đơn thuần, không chứa ảnh, link, mục lục,… tiếp đó văn bản này sẽ tiếp tục được tách thành từng câu và từng tiếng, các tiếng này sẽ được lọc nhiễu. Nghĩa là các tiếng này không được là tiếng Anh, địa chỉ email, số, viết tắt,… Tiếp theo, hệ thống sẽ xây dựng tập nhầm lẫn âm tiết, trong giai đoạn này, hệ thống sẽ chọn ra những âm tiết dễ bị nhầm lẫn nhất và có tần suất kết hợp với các tiếng xung quanh cao hơn so với âm tiết gốc và gán trọng số cho từng âm tiết đó.

Bước tiếp theo là tìm lỗi và sửa lỗi, hệ thống sử dụng mô hình n-gram và phân đoạn từ để đo độ liên kết giữa một tiếng với các tiếng lân cận. Tiếng nào có độ liên kết thiếu chặt chẽ sẽ được xem như bị lỗi chính tả. Sau đó, hệ thống tiến hành sinh âm tiết khác để sửa lỗi cho câu. Sau khi có tập nhầm lẫn âm tiết, tác giả áp dụng thuật toán Viterbi để tìm ra dãy tiếng có xác suất lớn nhất làm câu sửa lỗi. Các xác suất được ước lượng và so sánh bằng n-gram kết hợp phân đoạn từ.

Hình 3. Ví dụ mô hình N-gram kết hợp Viterbi

2.6.2.5 Kiểm lỗi chính tả bằng MS Word

Vào đầu năm 2005, hãng Microsoft đã phát hành MS Word phiên bản tiếng Việt. Trong phiên bản này, Microsoft đã tích hợp tính năng kiểm tra lỗi chính tả cho tiếng Việt. Tuy nhiên, tính năng này chỉ có thể kiểm tra lỗi ở mức tiếng, một mức độ kiểm tra rất thấp.

CHƯƠNG 3 – MÔ HÌNH NGÔN NGỮ N-GRAM

3.1 Mô hình ngôn ngữ

Mô hình ngôn ngữ là một trong những phương pháp xử lý ngôn ngữ tự nhiên. Mô hình ngôn ngữ là một phân bố xác xuất trên các tập văn bản. Nói một cách đơn giản, mô hình ngôn ngữ có thể cho biết xác suất một câu (hoặc cụm từ) thuộc một ngôn ngữ là bao nhiêu.

Ví dụ: Khi ta áp dụng mô vào ngôn ngữ vào Tiếng Việt.

P[“hôm nay thật là vui”] = 0,001.

P[“vui là thật nay hôm”] = 0.

Mô hình ngôn ngữ được áp dụng trong rất nhiều lĩnh vực khác nhau của xử lý ngôn ngữ tự nhiên như: kiểm lỗi chính tả, phân đoạn từ…Vì vậy, nghiên cứu mô hình ngôn ngữ là tiền đề để nghiên cứu các lĩnh vực khác của xử lý ngôn ngữ tự nhiên.

Có rất nhiều hướng tiếp cận mô hình ngôn ngữ, nhưng chủ yếu là được xây dựng theo mô hình N-gram.

3.2 Mô hình ngôn ngữ N-gram

Mô hình N-gram là mô hình ngôn ngữ rất phổ biến và được sử dụng rất nhiều trong xử lý ngôn ngữ tự nhiên. Áp dụng tư tưởng “Dựa vào những dữ liệu cho trước, đã biết để đoán cái tiếp theo, chưa biết”. N-grams ước lượng xác suất xuất hiện của một từ khi đã biết N-1 từ ngay phía trước nó [12].

Một N-gram là một bộ các yếu tố (từ, tiếng, nhãn từ loại,…) liên tiếp trong ngữ liệu. Số yếu tố trong một N-gram được gọi là bậc của N-gram. Thông thường, người ta tính bậc của N-gram từ bậc 1 đến bậc 4. N-gram bậc 1 được gọi là unigram, bậc 2 được gọi là bigram, bậc 3 được gọi là trigram, bậc 4 được gọi là quadgram.

Một mô hình ngôn ngữ có nhiệm vụ là phải cho ta biết được xác suất của một câu là bao nhiêu.

Ví dụ: ta có một câu W1W2…Wn.

Để tính xác suất câu trên, ta sử dụng công thức Bayes:

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

Với công thức này ta sẽ được:

P(W1W2…Wn) = P(W1) \* P(W2|W1) \* P(W3| W1W2) \* … \* P(Wn|W1W2…Wn-1)

Nếu ta áp dụng như vậy, ta sẽ cần phải có một dung lượng bộ nhớ vô cùng lớn để có thể lưu trữ hết các xác suất của tất cả các chuỗi có độ dài bé hơn n. Điều là bất khả thi khi trong các văn bản ngôn ngữ tự nhiên, độ dài n có thể tiến tới vô cùng. Do đó, ta cần một cách khác để có thể tính xác suất của văn bản với lượng bộ nhớ chấp nhận được, ta mô hình Markov bậc m:

P(Wn| W1W2…Wn-1) = P(W­n|Wn-m Wn-m+1…Wn-1)

Khi dùng mô hình này, xác suất của một từ Wn thay vì phụ thuộc vào toàn bộ những từ nằm trước nó thì chỉ phụ thuộc vào một khoảng m từ đứng trước nó. Công thức ban đầu sẽ được viết lại như sau:

P(W1W2…Wn) = P(W1) \* P(W2|W1) \* P(W3| W1W2) \* … \* P(Wn-1|Wn-m-1Wn-m…Wn-2) \* P(Wn|Wn-mWn-m+1…Wn-1)

Mô hình N-gram là mô hình được xây dựng dựa trên công thức này. N-grams ước lượng xác suất xuất hiện của một từ khi đã biết N-1 ngay phía trước nó.

3.2.1 Tính xác suất bằng N-gram

Ta có công thức tổng quát để tính xác suất bằng N-gram như sau [5]:

P(Wn| Wn-m+1 …Wn-1) =

Trong đó:

C(Wn-m+1 …Wn-1 Wn) là tần số xuất hiện của một câu Wn-m+1 …Wn-1 Wn.

C(Wn-m+1 …Wn-1) là tần số xuất hiện của một câu Wn-m+1 …Wn-1 Wn.

P(Wn| Wn-m+1 …Wn-1) là xác suất của một từ Wn với cụm từ nằm trước nó là Wn-m+1 …Wn-1

Một cách tổng quát hơn, ta có P(w|h) với w là từ ta cần tính xác xuất, h là dãy những từ nằm trước w.

Ta có:

Unigram: h rỗng.

Bigram: h là 1 từ nằm ngay trước w

Trigram: h là 2 từ nằm ngay trước w

N-gram: h là n-1 từ nằm ngay trước w

Ví dụ: ta có kho ngữ liệu mẫu nhỏ sau:

<s> Tôi đi học hằng ngày </s>

<s> Hằng ngày tôi đều phải đi học </s>

<s> Tôi học giỏi tiếng anh </s>

Ta sẽ tính bằng Bigram một số từ:

P(đi|Tôi) = P(học|đi) = P(học|Tôi) =

3.2.2 Khó khăn khi xây dựng mô hình ngôn ngữ N-gram

3.2.2.1 Phân bố không đều

Khi sử dụng mô hình ngôn ngữ N-gram, sự phân bố không đều trong tập huấn luyện có thể dẫn đến các ước lượng không được chính xác. Khi các n-gram phân bố thưa, nhiều cụm n-gram xuất hiện rất ít hoặc không xuất hiện, do đó việc ước lượng các câu có chứa các cụm n-gram này sẽ có kết quả không tốt. Với V là kích thước của bộ từ vựng, ta sẽ có V­n cụm n-gram có thể sinh ra từ bộ từ vựng. Tuy nhiên, trên thực tế thì số lượng cụm n-gram có nghĩa và thường gặp chiếm rất ít [8].

Ví dụ: tiếng Việt có khoảng 5000 âm tiết khác nhau, tổng số cụm 3-gram có thể có là 5000­3 = 125 tỷ. Tuy nhiên, số cụm 3-gram thống kê được chỉ xấp xỉ 1 triệu 500 ngàn. Như vậy sẽ có rất nhiều cụm 3-gram không xuất hiện hoặc chỉ xuất hiện rất ít.

Khi tính toán xác suất của một câu, có rất nhiều trường hợp sẽ gặp cụm n-gram chưa bao giờ xuất hiện trong tập dữ liệu huấn luyện. Điều này làm cho xác suất của cả câu bằng 0, trong khi câu này có thể là một câu hoàn toàn đúng về mọi mặt, đúng về cả ngữ pháp lẫn ngữ nghĩa. Để khắc phục vấn đề này cần sử dụng một số phương pháp làm mịn.

3.2.2.2 Kích thước bộ nhớ của mô hình ngôn ngữ

Khi kích thước của tập huấn luyện quá lớn, các cụm n-gram và kích thước của mô hình ngôn ngữ cũng rất lớn. Điều này không những gây khó khăn trong việc lưu trữ mà còn làm cho tốc độ xử lý của mô hình giảm xuống do hạn chế của bộ nhớ máy tính. Để xây dựng được một mô hình ngôn ngữ hiệu quả thì cần phải giảm kích thước của mô hình ngôn ngữ mà vẫn đảm bảo được độ chính xác [8].

3.2.3 Các phương pháp làm mịn

Để tránh tình trạng các cụm n-gram phân bố không đồng đều cần dùng phương pháp “làm mịn” các kết quả thống kê nhằm đánh giá chính xác hơn xác suất của các cụm n-gram [12]. Các phương pháp “làm mịn” đánh giá lại xác suất của các cụm n-gram bằng cách:

* Gán một giá trị khác 0 cho các cụm n-gram có xác suất bằng 0 (các cụm không xuất hiện trong kho ngữ liệu.
* Thay đổi lại giá trị xác suất cho các cụm n-gram có xác suất khác 0 khác ( các cụm có xuất hiện khi thống kê ) thành một giá trị phù hợp.

3.2.3.1 Phương pháp Add-one (Laplace)

Theo phương pháp này, ta sẽ cộng cả tử và mẫu số với một giá trị thích hợp để đảm báo kết quả sẽ luôn khác 0. Ta sẽ cộng tử số thêm 1 và mẫu số với một giá trị V, với V là tổng số từ có trong từ điển [5].

Công thức sẽ được tính lại như sau:

Qua công thức có thể thấy rằng, có rất nhiều cụm n-gram không nhìn thấy ( bậc thấp ) so với những n-gram nhìn thấy (bậc cao). Trong khi đó, những cụm n-gram cần thiết bị giảm đi, còn những cụm n-gram tối nghĩa lại có xác suất tăng lên. Để hạn chế điều này, người ta đã đưa vào thêm hệ số α thay vì cộng thêm 1 để cân đối lại xác suất (phương pháp làm mịn Add-α). Lúc đó công thức sẽ có dạng:

3.2.3.2 Phương pháp cộng hệ số

Trên thực tế, phương pháp cộng 1 cho kết quả không hợp lý bởi dưới một góc độ nào đó, giá trị 1 đôi khi là quá lớn và sai số của xác suất trở nên quá nhiều. Vì vậy, thay bởi cộng thêm 1 cho tử số, phương pháp cộng hệ số sẽ tăng một giá trị δ (0 < δ < 1) [8].

Theo đó, xác suất được tính lại như sau:

3.2.3.3 Phương pháp GoodTuring

Phương pháp GoodTuring sử dụng để tránh một cụm n-gram có nghĩa nhưng lại có tần suất xuất hiện thấp. Phương pháp làm mịn này dựa trên ước lượng cùng tần suất, có nghĩa là nhóm các cụm n-gram có số lần xuất hiện như nhau [8].

Theo phương pháp này, với cụm n-gram chưa từng xuất hiện trước đó được tính như sau:

Còn với mỗi cụm n-gram xuất hiện c lần, ra có thể xem như nó xuất hiện c\* lần, với c\* được tính như sau:

Trong đó:

c: số lần xuất hiện của cụm n-gram.

N: tổng số cụm n-gram.

Nc+1: tổng số cụm n-gram xuất hiện của c+1 lần.

Ví dụ: Trong một túi đựng bi, các bi có số lần xuất hiện như sau: bi đỏ 10 lần, bi vàng 3 lần, bi cam 2 lần, bi đen 1 lần, bi trắng 1 lần, bi xám 1 lần. Tổng số là 18 lần. Dựa vào công thức, xác suất để xuất hiện bi tím, một bi chưa xuất hiện lần nào, là . Xác suất để xuất hiện bi trắng là

3.2.3.4 Phương pháp truy hồi (Back-Off)

Trong phương pháp Add-one (hoặc Add-α) đã nêu trên, nếu cụm W1,W2…Wk-1 không xuất hiện trong tập huấn luyện thì cụm W­1W2…Wk sau khi được làm mịn xác suất vẫn là 0. Kết hợp với phương pháp truy hồi sẽ giúp giải quyết được vấn đề này [8].

Phương pháp truy hồi là phương pháp mà xác suất được tính từ điều kiện của cụm n-gram trước đó. Cụ thể, phương pháp này ước lượng xác suất của những cụm n-gram không xuất hiện dựa vào xác suất các cụm n-gram ngắn hơn mà xác suất khác 0 trong kho ngữ liệu [8].

Công thức có dạng như sau:

Trong đó, là xác suất theo mô hình đã tiên đoán và hàm chiết khấu .

Trong phương pháp truy hồi này, sự chính xác của mô hình phụ thuộc nhiều vào tham số α. Có một số cách để có thể chọn được α tùy theo mô hình n-gram và tập huấn luyện. Thêm vào đó, để mô hình có độ chính xác cao hơn thì cần kết hợp thêm một số phương pháp khác như Good-Turing để làm giảm xác suất của các cụm n-gram đã xuất hiện.

Xác suất tính cho một unigram:

Phương pháp GoodTuring điều chỉnh c về kì vọng , với ≤ c. Khi đó:

Dẫn đến chiết khấu được tính lại là:

3.2.3.5 Phương pháp nội suy

Phương pháp này có chung nguyên lý với phương pháp truy hồi:” sử dụng các cụm n-gram ngắn hơn để tính xác suất của cụm n-gram dài hơn” [8].

Tuy nhiên, phương pháp này khác với phương pháp truy hồi ở điểm: phương pháp này không phụ thuộc vào sự xuất hiện của các cụm n-gram.

Công thức tính xác suất theo phương pháp nội suy như sau:

Áp dụng cho Bi-gram và Tri-gram ta có:

với

Ở công thức trên, do tổng của tất cả các tham số 𝜆 bằng 1 nên để đơn giản ta có thể chọn tất cả 𝜆 bằng nhau và bằng .

Tuy nhiên, cũng có thể chọn các tham số 𝜆 như là một hàm của n-gram:

3.2.3.6 Phương pháp Kneser – Ney

Thuật toán Kneser – Ney được xây dựng theo hai mô hình: truy hồi và nội suy. Tuy nhiên, trong thuật toán này không cần phải áp dụng các thuật toán chiết khấu trước khi áp dụng công thức truy hồi [8].

* Mô hình truy hồi:

Trong đó:

* Với N(vw) là số lượng từ v khác nhau xuất hiện trước w trong tập huấn luyện.
* ()=

Như vậy:

* Mô hình nội suy:

Trong đó:

* Với N là số lượng từ V khác nhau xuất hiện liền sau cụm trong tập huấn luyện.
* với N(VW) là số lượng từ V khác nhau xuất hiện liền trước từ W trong tập huấn luyện.

Như vậy:

Trong cả 2 mô hình nội suy và truy hồi, D được chọn :

3.2.4 Phương pháp làm giảm kích thước dữ liệu

Giảm kích thước dữ liệu là phương pháp để khắc phục vấn đề kích thước của tập huấn luyện quá lớn. Có ba phương pháp chính để làm giảm kích thước dữ liệu [8]:

* Loại bỏ (pruning): phương pháp này làm giảm số lượng các cụm n-gram trong mô hình bằng cách loại bỏ các cụm n-gram không liên quan.
* Đồng hóa (Quantization): phương pháp đồng hóa này thay đổi cấu trúc thông tin của mỗi cụm n-gram trong mô hình ngôn ngữ.
* Nén (Compression): nén cấu trúc dữ liệu sử dụng trong việc lưu trữ các cụm n-gram trong mô hình ngôn ngữ.

3.2.4.1 Phương pháp loại bỏ (Pruning)

Số lượng các cụm n-gram xuất hiện với tần số rất thấp trong tập huấn luyện thường lớn so với tổng số các cụm n-gram. Các cụm n-gram thường là lỗi ngữ pháp hoặc là một số dạng như: từ viết tắt, tên riêng,… Những cụm n-gram này thường rất ít khi được sử dụng trong thực tế, do đó việc tồn tại những cụm này có thể làm ảnh hưởng đến độ chính xác của mô hình ngôn ngữ. Vì lý do này, phương pháp này tập trung vào việc loại bỏ những cụm n-gram không cần thiết đó. Có 2 phương pháp để thực hiện việc loại bỏ này [8]:

* Cắt bỏ (Cut-off): phương pháp này tập trung vào việc loại bỏ các cụm n-gram có tần số thấp trong tập huấn luyện.
* Khác biệt trọng số (Weighted differnce): phương pháp này đánh giá và loại bỏ các cụm n-gram không hiệu quả dựa vào xác suất của các cụm n-gram trước và sau khi làm mịn theo phương pháp truy hồi.

*a. Cắt bỏ (Cut-off)*

Phương pháp này [8] khá thông dụng để làm giảm kích thước cho mô hình ngôn ngữ. Thực tế, trong tập huấn luyện, có rất nhiều những cụm bi-gram và tri-gram chỉ xuất hiện vài lần trong đoạn văn bản chưa trên một triệu từ. Khi loại bỏ các cụm n-gram này ra khỏi mô hình ngôn ngữ, thông tin về chúng (tần số và xác suất) vẫn có thể nhận lại được thông qua việc sử dụng mô hình truy hồi hay nội suy.

Cách thức hoạt động của phương pháp này như sau: Nếu cụm n-gram nào xuất hiện ít hơn k lần trong tập văn bản huấn luyện thì cụm n-gram đó sẽ bị loại bỏ. Khi tính toán, nếu gặp lại các cụm n-gram này thì tần số và xác suất của các cụm này sẽ được tính toán thông qua các phương pháp làm mịn được trình bày ở trên. Tham số k có thể sử dụng khác nhau phụ thuộc vào các cụm n-gram có độ dài khác nhau. Ví dụ: với uni-gram thì cho k = 20, với bi-gram cho k = 16, tri-gram cho k = 9.

Như vậy, có thể thấy được rằng việc chọn tham số k là vấn đề của phương pháp cắt bỏ này. Nếu chọn k quá lớn, sẽ làm bỏ sót thông tin về một số cụm n-gram, hiệu suất của ứng dụng cũng sẽ bị giảm đi. Ngược lại, nếu k quá nhỏ, thì kích thước của mô hình ngôn ngữ cũng giảm không đáng kể. Có 2 cách để chọn tham số k: chọn k theo phương pháp chạy thử nhiều lần hoặc chọn theo tỉ lệ phần trăm số lượng các cụm n-gram.

Với cách chọn đầu tiên là chọn k theo phương pháp chạy thử nhiều lần, nghĩa là ta dùng phương pháp cut-off cho mô hình ngôn ngữ với nhiều giá trị k khác nhau rồi đánh giá độ hỗn loạn thông tin của tập văn bản đầu vào. Sau khi có kết quả, tham số k sẽ được chọn để mô hình ngôn ngữ đạt hiệu quả cao nhất (độ hỗn loạn thông tin của tập văn bản huấn luyện và kích thước của mô hình ngôn ngữ đều ở mức thấp). Kỹ thuật này sẽ giúp chọn được tham số k phù hợp, nhưng đánh đổi lại thì cũng mất rất nhiều thời gian do phải chạy thử với nhiều giá trị k khác nhau.

Cách thứ hai, k sẽ được chọn theo tỷ lệ phần trăm của số lượng các cụm n-gram, phải đảm bảo rằng số cụm n-gram xuất hiện không quá k lần chiếm h% tổng số các cụm n-gram.

Ví dụ: nếu h = 45, thì k sẽ được chọn sao cho số lượng cụm n-gram xuất hiện không quá k lần chiếm 45% tổng số các cụm n-gram đã thống kê. Phương pháp này có tốc độ nhanh hơn phương pháp đầu tiên, tuy nhiên độ chính xác sẽ không cao bằng.

*b. Sự khác biệt trọng số (Weighted difference)*

Ý tưởng của phương pháp cắt bỏ [8] là chỉ quan tâm việc loại bỏ các cụm n-gram có tần số thấp, trong khi phương pháp weighted difference thì quan tâm đến nhiều thông tin trong mô hình ngôn ngữ hơn như mối quan hệ giữa các cụm n-gram, xác suất từng cụm n-gram,… Như đã trình bày ở các mục trên, nếu một cụm n-gram không xuất hiện trong tập huấn luyện thì xác suất của cụm đó sẽ được ước lượng thông qua xác suất của các cụm n-gram ngắn hơn ( phương pháp làm mịn truy hồi). Do đó, Nếu xác suất của một cụm n-gram xấp xỉ với xác suất có được theo công thức truy hồi, thì không cần lưu trữ cụm n-gram đó nữa. Đó là ý tưởng của phương pháp weighted difference. Sự khác biệt trọng số được định nghĩa bằng:

K là tham số sử dụng trong phương pháp làm mịn GoodTuring. Dựa vào wdFactor để biết được nên loại bỏ hay giữ lại một cụm n-gram. Nếu wdFactor nhỏ hơn một ngưỡng nhất định thì cụm n-gram đó sẽ bị loại bỏ. Ngưỡng này được xác định bằng phương pháp thử sai hoặc đặt nó bằng một giá trị hằng số.

Phương pháp này mất nhiều thời gian hơn phương pháp loại bỏ (cut-off) do phải tính toán hệ số wdFactor cho tất cả các cụm n-gram trong mô hình ngôn ngữ. Sự khác biệt giữa hai phương pháp này là phương pháp weighted difference chỉ hoạt động trong mô hình ngôn ngữ kiểu truy hồi, còn phương pháp cut-off thì chỉ hoạt động trong những mô hình ngôn ngữ lưu trữ dữ liệu dưới dạng tần số.

3.2.4.2 Đồng hóa (Quantization)

Phương pháp sử dụng thuật toán đồng hóa làm giảm số lượng bit dùng để lưu các biến trong mô hình ngôn ngữ [8]. Thuật toán này có hai bước:

Bước đầu tiên, liệt kê và lưu trữ tần số của các cụm n-gram vào một bảng. Tiếp theo đó, thay thế tần số của các cụm n-gram trong mô hình ngôn ngữ bằng chỉ số của tần số trong bảng. Nhờ vậy, thay vì sử dụng b = log2 (tấn số lớn nhất) bit để lưu trữ tần cố của một cụm n-gram, thì chỉ cần sử dụng b’ = log2 (kích thước của bảng) bit cho mỗi cụm n-gram. Vì kích thước của bảng nhỏ hơn nhiều so với giá trị tần số lớn nhất của các các cụm n-gram nên b’ < b, tức là kích thước của mô hình ngôn ngữ đã giảm đi so với các lưu trữ ban đầu.

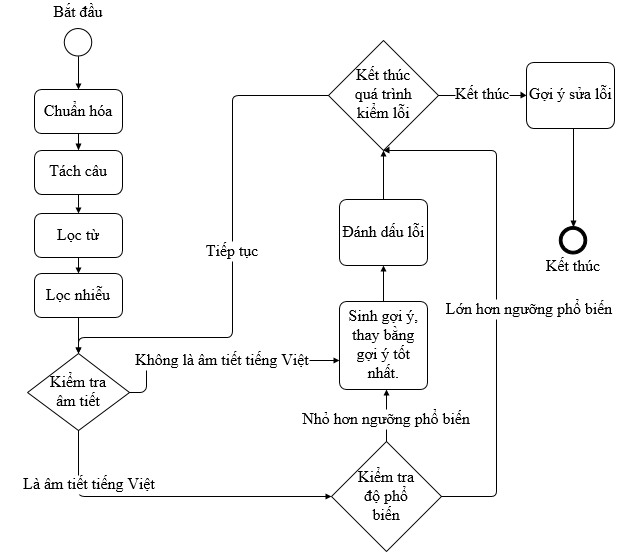
Để tăng thêm hiệu quả, ở bước hai, thuật toán này đồng hóa một số giá trị trong bảng tần số. Các giá trị trong bảng có giá trị gần với nhau sẽ được thay thế bằng một con số chung. Sau bước này, sẽ thu được một bảng tần số với giá trị ít hơn, tức là kích thước của mô hình ngôn ngữ lại giảm một lần nữa.

3.2.4.3 Nén (Compression)

Mô hình ngôn ngữ nào cũng có một cấu trấu dữ liệu [8]. Nếu cấu trúc dữ liệu đó được nén lại bằng các thuật toán nén, thì kích thước của mô hình ngôn ngữ cũng sẽ giảm. Nhưng khi một mô hình ngôn ngữ bị nén, thì tốc độ và độ chính xác của mô hình ngôn ngữ đều sẽ giảm. Do không đạt hiệu quả nên phương pháp này không còn phổ biến.

CHƯƠNG 4 – GIỚI THIỆU PHƯƠNG PHÁP

4.1 Mô hình tổng quát



Hình 4. Mô hình tổng quát hệ thống kiểm lỗi chính tả

* Phần xử lý văn bản:
  + Chuẩn hóa lại văn bản đầu vào.
  + Tách văn bản thành từng câu dựa vào nhưng ký tự kết thúc.
  + Tách những câu ở bước trên thành từng từ riêng biệt với nhau.
  + Lọc những cụm từ tiếng Anh, tên riêng, các từ nước ngoài, viết tắt, số, các ký hiệu đặc biết, email, …
* Phần kiểm tra lỗi: tìm những từ có khả năng sai và đánh dấu để người dùng nhận biết trên Microsoft Word
* Phần sửa lỗi: đưa ra những từ gợi ý cho người dùng chọn để sửa lỗi.

4.1.1 Phần xử lý văn bản

4.1.1.1 Chuẩn hóa văn bản

Văn bản đầu vào sẽ bao gồm nhiều đối tượng khác nhau nhưng không phải là text như mục lục, link,... Nên đầu tiên cần phải loại bỏ những đối tượng này.

4.1.1.2 Tách câu

Sau khi chuẩn hóa, tiếp đến là bước tách văn bản thành từng câu. Bước này dùng làm tiền đề cho bước tách từ và dùng để xử lý riêng khi kiểm tra lỗi chính tả cho từng từ.

Tách thành từng câu dựa vào các ký tự kết thúc, việc này giúp giảm không gian xử lý của chương trình và không ảnh hưởng đến hiệu quả cũng như nâng hiệu năng hơn.

4.1.1.3 Tách từ

Những câu được tách ở bước trên sẽ tiếp tục tách thành các từ riêng biệt.

4.1.1.4 Lọc nhiễu

Bước cuối cùng trong phần xử lý văn bản là lọc nhiễu. Trong các văn bản hiện nay thường xử dụng khá nhiều từ tiếng nước ngoài, đặc biệt là tiếng Anh, hoặc những từ mượn. Hệ thống sẽ phải lọc và xử lý. Ngoài những vấn đề về ngôn ngữ thì việc lọc còn phải xét những vấn đề khác như lọc ra những cụm số, tên riêng, địa chỉ email, địa chỉ website, các từ viết tắt,…Bước này là bước quan trọng để giúp tính toán N-gram được chính xác hơn.

4.1.2 Phần kiểm tra lỗi

Tiếp theo sẽ tiến hành tìm lỗi và đánh dấu những lỗi đó lên Word. Ý tưởng để kiểm tra lỗi là khi duyệt một từ, dùng từ điển tiếng Việt để xem từ đó có thuộc từ điển không, nếu không thì đánh dấu từ đó vào trường hợp sai hoàn toàn và đánh dấu trên Word. Trường hợp từ đó đúng thì tiếp đó sẽ sử dụng mô hình N-gram để đo độ liên kết của từ đó với các từ lân cận, nếu N-gram thấp thì từ đó sẽ được xem là lỗi chính tả.

4.1.3 Phần sửa lỗi

4.1.3.1 Tạo tập từ gợi ý

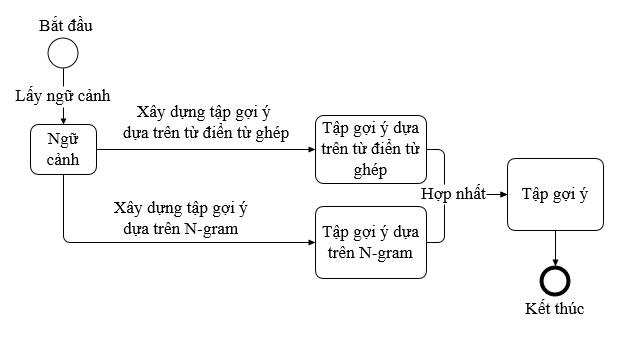
Việc xây dựng tập từ gợi ý cho mỗi âm tiết dựa trên sự liên kết giữa âm tiết đó với từ điển Tiếng Việt và từ điển N-gram. Với mỗi loại lỗi, hệ thống cho ra những từ gợi ý tương ứng.

Quá trình xử lý tập hợp từ gợi ý xây dựng có ba bước, bao gồm:

• Bước 1: Lấy ngữ cảnh hiện tại xung quanh âm tiết: Gồm âm tiết, hai âm tiết liền trước và hai âm tiết liền sau (nếu có).

• Bước 2: Xây dựng tập hợp từ gợi ý cho từng phần, bao gồm: dựa trên từ điển từ ghép Tiếng Việt và từ điển N-gram.

• Bước 3: Hợp nhất hai tập hợp từ gợi ý để tạo ra một tập hợp gồm những từ gợi ý là duy nhất.



Hình 5. Mô hình tổng quát tạo tập gợi ý

Trong đó:

* Xây dựng tập gợi ý dựa trên từ điển từ ghép tiếng Việt: lấy toàn bộ những từ tạo với ngữ cảnh của âm tiết hiện tại thành từ ghép tiếng Việt có hai hoặc ba âm tiết.

Cụ thể, với ngữ cảnh là , ta cần tìm một tập hợp tạo với ngữ cảnh thành một từ ghép tiếng Việt có hai hoặc ba âm tiết. Gồm 5 trường hợp sau:

* Xây dựng tập gợi ý dựa trên N-gram: tương tự như trường hợp xây dựng tập gợi ý dựa trên từ điển từ ghép tiếng Việt, trường hợp xây dựng tập gợi ý dựa trên N-gram cũng tìm một tập hợp các từ có trong từ điển tạo với ngữ cảnh một cụm từ có trong N-gram. Nhưng để đảm bảo tốc độ chương trình, hệ thống chúng tôi chỉ sử dụng bi-gram nên ngữ cảnh lúc này trở thành: . Do đó để tạo tập hợp chỉ có hai trường hợp là:

Vì số lượng những phần tử trong tập hợp các từ thỏa mãn những điều kiện trên là rất nhiều, nên trong quá trình hợp nhất hai tập hợp (được xây dựng dựa trên từ điển từ ghép và N-gram) ta cần so sánh độ tương tự giữa với để chọn ra những phần tử gần giống với nhất.

4.1.3.2 Tính điểm và lựa chọn từ gợi ý

Sau khi có tập gợi ý, những từ này sẽ được tính điểm để xác định đó có phải là từ gợi ý hợp lý hay không. Điểm này sẽ được so sánh với một chỉ số được gọi là “ngưỡng sửa lỗi”. Ngưỡng này được xác định dựa vào việc thống kê nhiều trường hợp khác nhau dựa vào Corpus. Nếu điểm của từ nào lớn hơn hoặc bằng chỉ số này sẽ được xác định là hợp lý để gợi ý.

Điểm của những từ được chọn để gợi ý được dựa trên ba tiêu chí tính bằng công thức sau:

Trong đó:

* + là những tham số, có tổng được ước lượng dựa vào việc huấn luyện N-gram.
  + S (Similarity) có giá trị từ 0 đến 1 là điểm của từ gợi ý dựa vào độ tương tự đối với âm tiết hiện tại, điểm càng cao từ gợi ý càng giống với âm tiết hiện tại.
  + L (Model Language) có giá trị từ 0 đến 1 là điểm của từ gợi ý được tính dựa vào N-gram.
  + D (Dictionary) có giá trị từ 0 đến 1 là điểm của từ gợi ý được tính dựa vào từ điển từ ghép.

S thể hiện độ tương tự giữa âm tiết hiện tại và ứng cử viên. Trên thực tế, nhầm lẫn do lỗi đánh máy và do lỗi vùng miền thường xuyên xảy ra. Trong đó, lỗi vùng miền còn do nhầm lẫn trong mỗi thành phần của âm tiết. Vậy nên khi tính độ tương tự, phải cân nhắc tất cả những yếu tố kể trên. Qua học hỏi và nghiên cứu, chúng tôi tách phần dấu thanh của âm tiết và candidate để xét trường hợp nhầm lẫn về dấu thanh, sau đó sử dụng công thức bên dưới để tính độ tương tự giữa hai từ:

.

Với:

* S (Syllable): Âm tiết
* C (Candidate): Từ gợi ý
* WMS (WithoutMarkSyllable): Âm tiết sau khi loại bỏ dấu thanh
* WMC (WithoutMarkCandidate): Từ gợi ý sau khi loại bỏ dấu thanh
* MS (MarkSyllable): Dấu thanh của âm tiết
* MC (MarkCandidate): Dấu thanh của từ gợi ý
* DWM (DistanceWithoutMark): Phương thức tính khoảng cách giữa WMS và WMC được tính như sau:
* DM (DistanceMark): Phương thức tính khoảng cách giữa WS và WC

L (Model Language) thể hiện khả năng xuất hiện của từ gợi ý so với ngữ cảnh trong câu, hay hiểu rộng hơn là độ phổ biến của ngữ cảnh của từ gợi ý trong tập huấn luyện của N-gram. Dựa trên mô hình ngôn ngữ N-gram, hệ thống sẽ tính L dựa trên tần số của bigram, nếu từ gợi ý là hợp với ngữ cảnh của câu thì L sẽ rất cao, L được tính theo công thức bên dưới:

Với:

* V là kích thước của từ điển N-gram
* x, y: là tham số với tổng x + y = 1

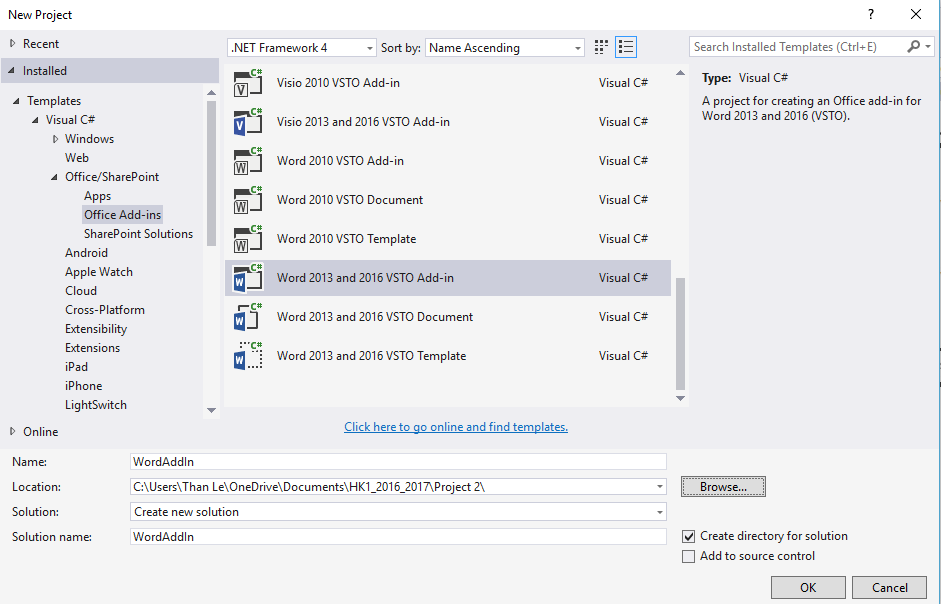
Như đã trình bày ở trên, do việc Corpus không đủ lớn sẽ khiến cho một số xác suất quá nhỏ hoặc thậm chí bằng 0. Nên cần phải kết hợp sử dụng phương pháp làm mịn, chương trình sử dụng phương pháp Add-one để làm mịn. Phương pháp này áp dụng nhanh, đơn giản mà đạt hiệu quả khả quan.

Trong thực tế, một số âm tiết dùng đúng ngữ cảnh nhưng có tần số quá thấp trong từ điển N-gram dẫn đến việc điểm L thấp, dẫn đến việc âm tiết đó có thể bị coi là sai chính tả. Trong trường hợp này nếu âm tiết đó cùng với ngữ cảnh xung quanh âm tiết là từ ghép thì tổng điểm Score sẽ được cải thiện dựa vào điểm D. D là điểm mà đánh giá từ gợi ý và ngữ cảnh xung quanh có thuộc từ điển từ ghép hay không. D được tính như sau:

4.2 Xây dựng Add-in vào Word

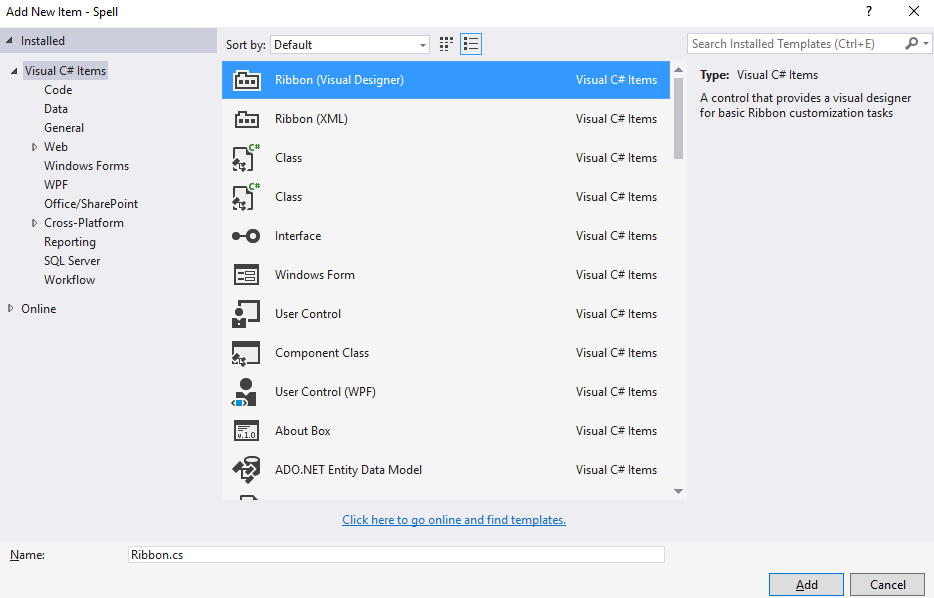
Add-in trong kỹ thuật máy tính là bộ phần mềm do hãng thứ ba tạo ra, nhằm thêm những tính năng cụ thể cho một phần mềm lớn hơn, cụ thể ở đây là MS Word.

4.2.1 Tạo Word VSTO Add-in

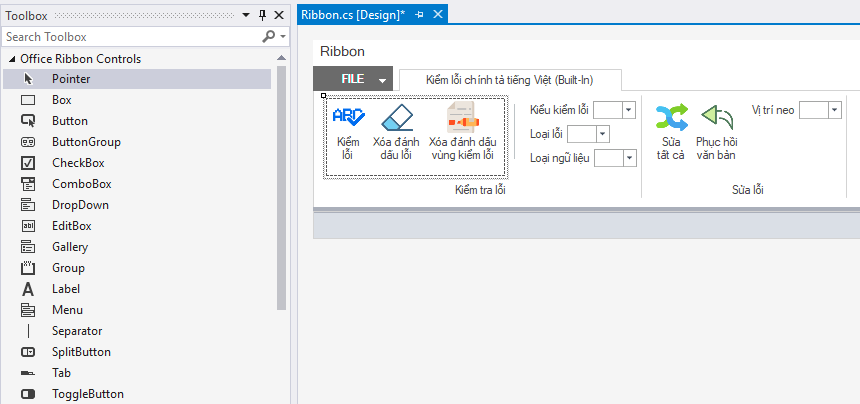
Trong phạm vi đồ án, chúng tôi dùng Microsoft Visual Studio 2015, và kể từ .Net Framework 4 trở lên, chúng ta đã có thể tạo được một Word Add-in bằng đường dẫn: New project → Installed → Templates → Visual C# → Office/ SharePoint → Office Add-ins và chọn Word 2013 and Word 2016 VSTO Add-in. Bằng cách này, chúng ta đã có thể tạo một Add-in trên Word 2013 và Word 2016.

Hình 6. Tạo Word Add-in

4.2.2 Tạo Ribbon (Visual Designer)

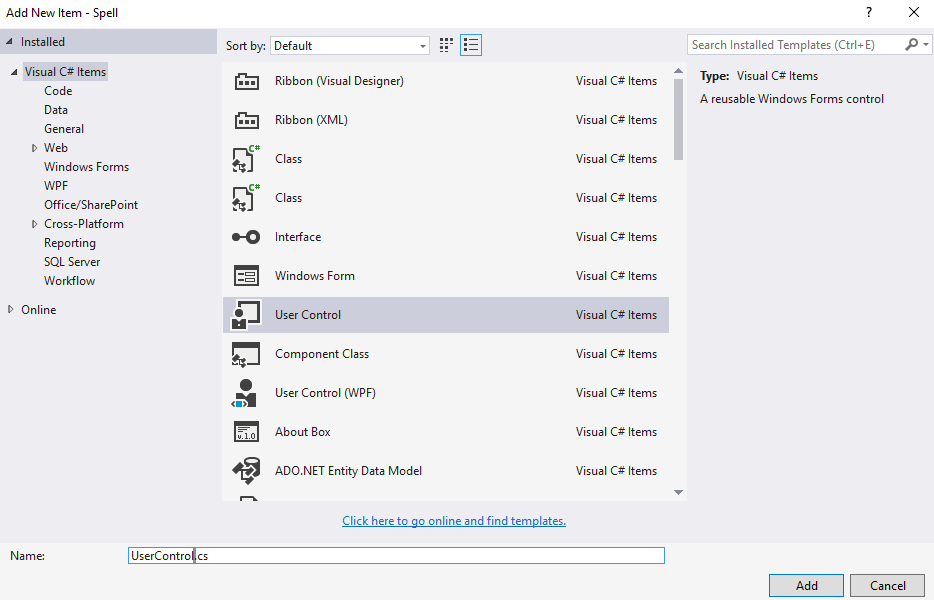
Sau khi tạo mới một Project, ta click chuột phải vào project chọn Add New Item. Tại đây, chọn Ribbon (Visual Desinger) theo đường dẫn Installed → Visual C# Items.

Hình 7. Tạo Ribbon (Visual Designer)

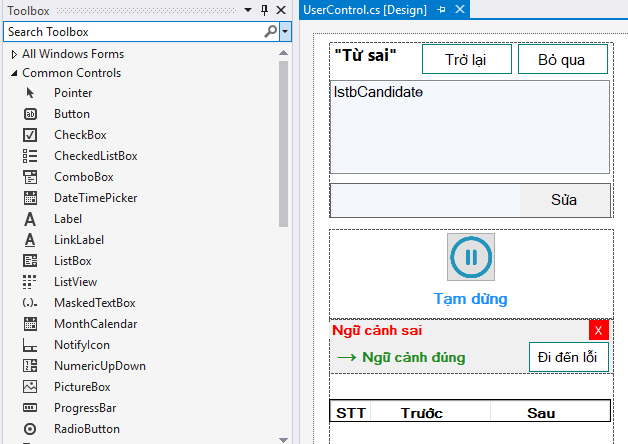
VSTO được tích hợp trên Visual Studio nên có thể sử dụng hầu như tất cả những tính năng sẵn có của bộ IDE mạnh mẽ này. Riêng với Ribbon trong VSTO, Visual Studio hỗ trợ những controls riêng trong Toolbox, được gọi là Office Ribbon Controls, tại đây, lập trình viên có thể kéo thả, và tùy chỉnh tương tự như làm việc trên Windows Form C#.

Hình 8. Thiết kế Ribbon

4.2.3 Tạo User Control

Sau khi có được Ribbon, ta lại chuột phải vào project và chọn Add New Item. Cũng trong đường dẫn Installed → Visual C# Items, ta chọn User Control không có WPF.

Hình 9. Tạo User Control

Đến đây, việc kéo thả để tạo giao diện trong User Control thực chất chính là làm việc với Windows Form C#, và bằng những kiến thức về Windows Form C#, ta tạo một giao diện cho User Control tương tự như bên dưới.

Hình 10. Thiết kế User Control

4.2.4 Làm việc với namespace Microsoft.Office.Interop.Word

Để có thể tạo một add-in mà những đối tượng trong đó có thể tương tác được với văn bản trong MS Word, ta sử dụng namespace *Microsoft.Office.Interop.Word*. Để cho tiện trong coding, chúng tôi gán *Word = Microsoft.Office.Interop.Word* dùng từ khóa using

using Word = Microsoft.Office.Interop.Word;

4.2.4.1 Microsoft.Office.Interop.Word.Range (Word.Range)

Để có thể “get” và “set” được một từ, cụm từ, hay đoạn văn nào đó trong Word Document, ta cần biết từ, cụm từ hay đoạn văn đó nằm trong phạm vi nào, bắt đầu và kết thúc ở ký tự thứ bao nhiêu. Để làm được điều đó, ta cần sử dụng một đối tượng tên là Range trong namespace *Microsoft.Office.Interop.Word.*

Định nghĩa một Word.Range ta cần truyền vị trí bắt đầu và kết thúc của vùng ta muốn đến phương thức của lớp *Microsoft.Office.Interop.Word.Document* như sau:

*object start = 0;*

*object end = 7;*

*Word.Range rng = this.Range(start, end);*

Trong đối tượng Range, có rất nhiều properties cũng như methods. Trong phạm vi đồ án này, chúng tôi sử dụng chủ yếu 4 properties là Word.Range.Font, Word.Range.HighlightColorIndex, Word.Range.Text và Word.Range.Underline và phương thức Word.Range.Select().

* Word.Range.Font: với property này, ta có thể quyết định “vùng” được chọn, format theo font gì, màu gì, có đậm, nghiêng, gạch chân hay không thông qua: Word.Range.Font.Color, Word.Range.Font.Bold, …
* Word.Range.HighlightColorIndex: HighlightColorIndex cho phép ta chọn màu highlight cho “vùng” được chọn.
* Word.Range.Text: property này cho phép ta “get” và “set” văn bản trong “vùng” được chọn, kiểu trả về là string.
* Word.Range.Underline: cho phép ta “get” và “set” kiểu gạch chân được áp dụng trong “vùng” được chọn.
* Word.Range.Select(): phương thức này dùng để bôi đen tất cả nội dung của range. Đồng thời, con trỏ văn bản cũng được đưa đến vị trí Range.Start khi sử dụng phương thức này. Đặc biệt, khi Range.Start bằng với Range.End, đó chính là vị trí con trỏ văn bản nhấp nháy trên giao diện MS Word.

4.2.4.2 Microsoft.Office.Interop.Word.Words (Word.Words)

Với Word.Words, ta thu về được một Words Collection. Mỗi phần tử trong Words Collection là một đối tượng Range, được đại diện cho một từ, không phải là đối tượng Word. Words(index) index là số chỉ mục, trả về một đối tượng đại diện cho một từ. Số chỉ mục thể hiện vị trí của từ trong Words Collection.

Thuộc tính Count trả về số lượng phần tử có trong Words Collection. Phương thức Add không có sẵn trong Words Collection. Thay vào đó, sử dụng phương pháp InsertAfter hoặc phương pháp InsertBefore để thêm văn bản vào một đối tượng Range.

4.2.4.3 Microsoft.Office.Interop.Word.Sentences (Word.Sentences)

Tương tự như Word.Words, với Word.Sentences, ta thu về được một Sentences Collection. Mỗi phần tử trong Sentences Collection là một đối tượng Range, được đại diện cho một câu, không phải là đối tượng Sentence. Sentences(index) index là số chỉ mục, trả về một đối tượng đại diện cho một câu. Số chỉ mục thể hiện vị trí của câu trong Sentences Collection.

Thuộc tính Count trả về số lượng phần tử có trong Words Collection. Phương thức Add không có sẵn trong Sentences Collection. Thay vào đó, sử dụng phương pháp InsertAfter hoặc phương pháp InsertBefore để thêm văn bản vào một đối tượng Range.

4.2.4.4 Lấy văn bản từ MS Word

Đoạn code sau giúp thu về Words Collection hay Sentences Collection hay Range:

Word.Words words = Globals.ThisAddIn.Application.Selection.Words;

Word.Words words = Globals.ThisAddIn.Application.ActiveDocument.Words;

Word.Sentences sentences = Globals.ThisAddIn.Application.Selection.Sentences;

Word.Sentences sentences = Globals.ThisAddIn.Application.ActiveDocument.Sentences;

Word.Range rng = Globals.ThisAddIn.Application.Selection.Range;

Word.Range rng = Globals.ThisAddIn.Application.ActiveDocument.Range(start, end);

Trong đó, ActiveDocument trả về một tập hợp toàn bộ văn bản có trong document, còn Selection trả về tập hợp văn bản được bôi đen.

CHƯƠNG 5 – CÀI ĐẶT, THỰC NGHIỆM VÀ ĐÁNH GIÁ

5.1 Cài đặt

Dựa vào mô hình lý thuyết nêu trên, chương trình sẽ được xây dựng bằng ngôn ngữ C#. Ngôn ngữ này có thể đảm bảo được tốc độ và độ bảo mật của chương trình, ngoài ra còn giúp tích hợp chương trình vào Microsoft Word 2013 và 2016. Chương trình đã được thực nghiệm thành công trên môi trường Window 10.

Các bước cài đặt hệ thống:

* Tạo Corpus.
* Đếm tần suất N-gram và lưu trữ.
* Viết chương trình
* Tích hợp vào Word.

5.1.1 Xây dựng tập huấn luyện (Corpus)

Như đã trình bày các bước tạo Corpus ở chương 2. Do trong quá trình chuẩn hóa Corpus và xử lý nhiễu còn nhiều khó khăn, chúng tôi đã loại bỏ những uni-gram và bi-gram có tần số thấp với khả năng chúng là các trường hợp lỗi, và sẽ ít có khả năng xuất hiện trong các văn bản, nên số lương uni-gram và bi-gram đã giảm đáng kể. Đây là kết quả Corpus thô đã được thu thập

|  |  |
| --- | --- |
| Độ lớn của Corpus | 1.38 GB |
| Số lượng uni-gram khác nhau | 4815 |
| Số lượng bi-gram khác nhau | 387405 |

Bảng 3. Độ lớn tập huấn luyện

5.1.2 Xây dựng tập dữ liệu

Tương tự như quá trình xây dựng tập huấn luyện, dữ liệu được thu thập từ Internet bao gồm tin tức, giấy tờ, truyện ngắn, tiểu thuyết trên các trang, ... như dantri.com.vn, vnexpress.net. Những tài liệu này được chúng tôi kiểm tra cẩn thận để đảm bảo rằng không có lỗi chính tả trong các tài liệu này. Sau đó, chúng tôi tạo ra lỗi chính tả trên tập dữ liệu này bao gồm: sai chính tả và sai ngữ cảnh. Những lỗi này được chúng tôi phân phối trên tất cả các loại lỗi phổ biến. Ví dụ:

Câu lỗi: “Theo chũ trương được phê duyệt, bên cạh xây dựng khu siêu thị, nhà ở thơng mại và kinh danh nhà ở, công ty còn đầu tư khu chợ mớ để di dời trên 400 tiểu thương đang mua bán ở khu chợ củ đã xuống cấp.”

Câu đúng: “Theo chủ[chũ] trương được phê duyệt, bên cạnh[cạh] xây dựng khu siêu thị, nhà ở thương[thơng] mại và kinh doanh[danh] nhà ở, công ty còn đầu tư khu chợ mới[mớ] để di dời trên 400 tiểu thương đang mua bán ở khu chợ cũ[củ] đã xuống cấp.”

5.1.3 Viết chương trình

Bước cuối cùng của phần cài đặt là viết chương trình, trình tự và phương pháp thực hiện theo mô hình đã được nêu ở chương 4. Chương trình yêu cầu đảm bảo tốc độ xử lý, độ chính xác cao, đồng thời giao diện cần được xây dựng thân thiện, dễ sử dụng và có thể nâng cấp phát triển thêm.

5.2 Thực nghiệm

Để đánh giá hệ thống kiểm tra chính tả, chúng tôi sử dụng 4 tiêu chí sau:

Độ phát hiện chính xác (Detection precision): cho thấy hệ thống phát hiện hữu ích như thế nào, được xem là một thước đo chính xác.

Độ hồi tưởng (Detection recall): cho thấy mức độ hoàn thành việc phát hiện của hệ thống, được xem là một thước đo hoàn chỉnh.

Độ điều chỉnh chính xác (Correction precision): cho thấy một thước đo về chất lượng của sự sửa lỗi.

Độ đo F-score: là một thước đo của việc thử nghiệm tính chính xác trong các lỗi phát hiện.

5.2.1 Môi trường thực nghiệm

|  |  |
| --- | --- |
| Bộ xử lý | Intel® 2 Core™ i7-2620M CPU @ 2.70GHz (4 CPUs), ~ 2.7GHz |
| Bộ nhớ | 4.0 GB RAM |
| Hệ điều hành | Windows 10 Educatoin 64 bit |
| Công cụ lập trình | C#, Visual Studio 2015 |

Bảng 4. Môi trường thực nghiệm

5.2.2 Kết quả thực nghiệm

Kích thước dữ liệu thực nghiệm: 140kb

Tổng số lỗi được sinh: 1726

Tổng số lỗi phát hiện: 1778

Tổng số lỗi phát hiện sai: 203

Tổng số lỗi phát hiện đúng: 1535

Tổng số lỗi sửa đúng: 1449

Đô phát hiện chính xác:

Độ hồi tưởng:

Độ hiệu chỉnh chính xác:

Độ đo F:

Hình 11. Kết quả thực nghiệm

Ví dụ kiểm lỗi của chương trình kiểm lỗi chính tả của khóa luận trong dữ liệu thực nghiệm:

Ngoài ra, *côgn*[công] ty này còn *inh*[kinh] *danh*[doanh] nhà ở dưới hình thức *hy*[huy] động vốn theo hình thức bán nhà ở hình thành *trông*[trong] tương *nai*[lai] khi chưa đáp ứng điều kiện kinh *danh*[doanh] bất động sản.

*Trông*[trong] khoảng thời gian đầy *dan*[gian] khó để hòa nhập vào một môi trường sống mới, nơi có bắt nạt và hiếp đáp, ma cũ ma mới...; ta dần dần được chứng *kiếm*[kiến] những câu chuyện về những đứa trẻ bị bạo hành, mang *trông*[trong] mình một vết sẹo không dễ gì lành ở lứa tuổi niên thiếu.

Có đứa thì cha mẹ bỏ *dơi*[rơi] vì nghiện ngập ma túy, có đứa thì cha mẹ bị tống *dam*[giam] vào tù, có đứa mẹ bị trục *suất*[xuất] và có đứa thì cha giết mẹ vì ngoại tình rồi sau đó tự sát...

**5**.3 Đánh giá

Quá trình thực nghiệm loại 1 cho thấy chương trình kiểm lỗi chính tả của khóa luận đạt được kết quả kiểm lỗi khá cao (khoảng 87.61%), trong đó khả năng sửa đúng lỗi là 91.79%. Tuy rằng đây chỉ là những thực nghiệm đầu tiên của chương trình, nhưng nó hứa hẹn chương trình sẽ đạt được thành công khi áp dụng vào thực tế.

CHƯƠNG 6 – TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

**Tổng kết**

Trong nội dung khóa luận này, chúng tôi đã nghiên cứu về lĩnh vực kiểm lỗi chính tả tiếng Việt, về một số mô hình kiểm lỗi chính tả đã có, đồng thời giới thiệu phương pháp sử dụng mô hình N-gram, phương pháp tích hợp hệ thống vào Microsoft Word, từ đó đề xuất những giải pháp mới để áp dụng vào kiểm lỗi chính tả trong Microsoft Word. Vì mục tiêu chính của khóa luận là tính thực tế, hệ thống phải xử lý trong khoảng thời gian ngắn mà vẫn phải cho kết quả tốt. Hệ thống kiểm lỗi chính tả của chúng tôi áp dụng mô hình N-gram không quá phức tạp, cho kết quả tốt trong một khoảng thời gian chấp nhận được.

Trong bước đầu cài đặt và thực nghiệm, kết quả đạt được của hệ thống kiểm lỗi chính tả của chúng tôi là rất khả quan. Hệ thống soát được lỗi với hiệu suất khá tốt với thời gian thực hiện là chấp nhận được đối với tập dữ liệu thực nghiệm khách quan và thuộc nhiều thể loại.

Tuy nhiên, hệ thống vẫn còn một số nhược điểm, đó là tỷ lệ lỗi vẫn còn khá cao (khoảng 12%) và việc kiểm lỗi mất khá nhiều thời gian. Với những ngữ cảnh xuất hiện ít trong tập ngữ liệu, hiệu suất kiểm lỗi của những trường hợp này là khá thấp. Thêm vào đó, với sự đa dạng của trong cách tạo văn bản, việc kiểm lỗi trong hệ thống vẫn chưa được xử lý thực sự tốt.

**Hướng phát triển**

Trong tương lai, chúng tôi dự định sẽ cải tiến phương pháp, thu thập thêm tập ngữ liệu để có thể phù hợp với nhiều thể loại văn bản khác nhau, tăng hiệu năng cũng như hiệu quả kiểm tra và sửa lỗi. Về mặt giao diện, chúng tôi sẽ cải thiện bố cục, thêm nhiều tính năng để giúp người dùng có những trải nghiệm tốt hơn khi kiểm tra và sửa lỗi chính tả tiếng Việt.

TÀI LIỆU THAM KHẢO

**Tiếng Việt**

|  |  |  |  |
| --- | --- | --- | --- |
| [1] | | Đặng Thị Lanh, Bùi Minh Toán, Lê Hữu Tỉnh, *Tiếng Việt,* Giáo trình chính thức đào tạo giáo viên tiểu học hệ CĐSP và SP 12+2. tập 1. NXB Giáo Dục, 1997. | |
| [2] | | Đinh Điền, Giáo trình xử lý ngôn ngữ tự nhiên, ĐH KHTN TPHCM, 2004. | |
| [3] | | Hoàng Phê. 1999. Chính tả tiếng Việt. Nhà xuất bản Đà Nẵng. | |
| [4] | | Nguyễn Đức Hải và Nguyễn Phạm Hạnh Nhi, *Phân tích cú pháp câu tiếng Việt và ứng dụng vào việc bắt lỗi chính tả,* luận văn cử nhân tin học, ĐH KHTN TPHCM, 1999 | |
| [5] | | Nguyễn Hưu Tiến Quang, *Kiểm lỗi chính tả Tiếng Việt sử dụng mô hình ngôn ngữ và phân đoạn từ*, Luận văn cử nhân, ĐH Công Nghệ Hà Nội, 2012. | |
| [6] | | Nguyễn Phương Thái, *Kiểm lỗi Chính tả Cảm Ngữ cảnh Tiếng Việt*, Luận văn thạc sĩ, Khoa Công nghệ, 2003. | |
| [7] | | Nguyễn Thái Ngọc Duy, Đinh Điền, *Một cách tiếp cận trong kiểm lỗi chính tả tiếng Việt,* Luận văn cử nhân tin học, ĐH KHTN TPHCM, 2004. | |
| [8] | | Vũ Chí Hiếu, *Nghiên cứu mô hình ngôn ngữ N-gram cho Tiếng Việt và ứng dụng sửa lỗi dấu thanh trong Tiếng Việt*, Luận văn thạc sĩ, ĐH Thái Nguyên.  **Tiếng Anh** | |
| [9] | | Andrew R.Golding and Dan Roth, *A winnow-based approach to context-sensitive correction,*Machine Learning, Special issue on Machine Learning and Nature Language Processing, 1999. |
| [10] | | Andrew R.Golding and Yves Schabes, *Combining trigram-based ang feature-based methods for context-sensitive spelling correction,*Proceedings of the 34th Annual Meeting of Association for Computational Linguistics, 1996. |
| [11] | | Andrew R.Golding, *A Bayesian hybrid methods for context-sensitive correction,* Proceedings of the Third Workshop on Very Large Copora, 1995. |
| [12] | | Jurafsky-Martin, Speech and Language Processing 2ed, 2006. |
| [13] | | Li Jianhua and Wang XiaoLong, Combine trigram and Automatic Weight Distribution in Chinese Spelling ErrorCorrection, 2000. |
| [14] | | Masaaki Nagata, A Stochastic Japanese Morphological Analyzer Using a Forward-BD Backward-A\*N-best search Algorithm, In Proceedings of COLING-94, 1994. |
| [15] | | Masaaki Nagata, Context-Based Spelling Correction for Japanese OCR, 1998. |
| [16] | | Nguyen Thi Xuan Huong, Tran-Thai Dang, The-Tung Nguyen and Anh-Cuong Le, *“Using Large N-gram for Vietnamese Spell Checking”* ,Proceedings of the Sixth International Conference KSE 2014, Part III, 617-627 |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |