**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**

**KHOA CÔNG NGHỆ THÔNG TIN**





**TIỂU LUẬN  
CHUYÊN ĐỀ TỐT NGHIỆP KHOA HỌC DỮ LIỆU 2**

**DỰ ĐOÁN TỪ TIẾP THEO TRONG CHUỖI VĂN BẢN BẰNG TENSORFLOW**

**GIẢNG VIÊN HƯỚNG DẪN :   THS. SỬ NHẬT HẠ**

**SINH VIÊN THỰC HIỆN :   VÕ THÁI HIỂN**

**MSSV :   2000005690**

**LỚP :   20DTH1D**

**CHUYÊN NGÀNH :   KHOA HỌC DỮ LIỆU**

**MÔN HỌC :   CHUYÊN ĐỀ TN KHDL 2**

**KHÓA :   2020**

**Tp.HCM, Tháng  11  Năm  2023**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**

**KHOA CÔNG NGHỆ THÔNG TIN**





**TIỂU LUẬN  
CHUYÊN ĐỀ TỐT NGHIỆP KHOA HỌC DỮ LIỆU 2**

**DỰ ĐOÁN TỪ TIẾP THEO TRONG CHUỖI VĂN BẢN BẰNG TENSORFLOW**

**GIẢNG VIÊN HƯỚNG DẪN :   THS. SỬ NHẬT HẠ**

**SINH VIÊN THỰC HIỆN :   VÕ THÁI HIỂN**

**MSSV :   2000005690**

**LỚP :   20DTH1D**

**CHUYÊN NGÀNH :   KHOA HỌC DỮ LIỆU**

**MÔN HỌC :   CHUYÊN ĐỀ TN KHDL 2**

**KHÓA :   2020**

**Tp.HCM, Tháng  11  Năm  2023**

LỜI CẢM ƠN

Em xin gửi lời cảm ơn từ tận đáy lòng tới Thầy Sử Nhật Hạ và trường Đại học Nguyễn Tất Thành (NTTU) về sự hỗ trợ và cơ hội quý báu trong quá trình thực hiện chuyên đề tốt nghiệp khoa học dữ liệu 2. Đây là một hành trình quan trọng và đáng nhớ, và em không thể không bày tỏ lòng biết ơn của mình.

Thầy Sử Nhật Hạ, với kiến thức sâu rộng và tâm huyết với giảng dạy, đã là người hướng dẫn tận tâm và thông thái. Sự hướng dẫn của thầy không chỉ giúp em phát triển kỹ năng nghiên cứu mà còn giúp em hiểu sâu hơn về lĩnh vực em nghiên cứu. Em cảm ơn thầy về sự khuyến khích, hỗ trợ, và sự lãnh đạo trong suốt thời gian qua. Thầy là nguồn động viên quý báu cho sự thành công của em trong dự án này.

Tôi cũng muốn bày tỏ lòng biết ơn tới trường đại học NTT vì cơ hội và môi trường học tập tốt nghiệp chất lượng mà trường đã mang lại. Trường đại học NTT đã cung cấp cho em sự hỗ trợ về cơ sở vật chất, tài liệu tham khảo, và một cộng đồng học thuật đáng kính. Em rất tự hào vì đã được đào tạo tại Trường đại học NTT và mang danh hiệu cử nhân của trường.

Cuối cùng, em muốn gửi lời cảm ơn sâu sắc tới gia đình, bạn bè, và người thân của mình. Họ luôn ở bên em, hỗ trợ tinh thần và động viên khi tôi gặp khó khăn. Những người này là nguồn động viên quan trọng giúp em vượt qua mọi thử thách trong quá trình nghiên cứu và viết chuyên đề tốt nghiệp này.

Bài khoá luận này không thể hoàn thành mà không có sự đóng góp của tất cả những người trên. Em hi vọng rằng công trình này có thể mang lại giá trị cho cộng đồng và đóng góp vào lĩnh vực của Khoa học dữ liệu.

Cảm ơn tất cả mọi người một lần nữa vì sự hỗ trợ và đóng góp.

***Sinh viên thực hiện***

**Võ Thái Hiển**

LỜI MỞ ĐẦU

Chào các thầy cô hướng dẫn, các bạn học viên, và quý khách mời, em rất hân hạnh và tự hào được trình bày trước mặt quý vị chuyên đề tốt nghiệp khoa học dữ liệu lần thứ hai. Chuyên đề này đánh dấu một bước ngoặt quan trọng trong hành trình học tập và nghiên cứu của chúng tôi tại khoa khoa học dữ liệu.

Trong thời đại số hóa và thông tin hiện nay, dữ liệu đang trở thành nguồn tài nguyên quý báu và có tiềm năng biến đổi toàn bộ xã hội và kinh tế. Lĩnh vực khoa học dữ liệu đang trỗi dậy, và em đã dành nhiều tháng, thậm chí nhiều năm để tìm hiểu, nghiên cứu, và áp dụng các phương pháp và kỹ thuật trong lĩnh vực này. Em tin rằng chuyên đề này sẽ là cơ hội để em chia sẻ kiến thức và kết quả nghiên cứu của mình, đồng thời hi vọng rằng nó sẽ đem lại giá trị và kiến thức mới mẻ cho cộng đồng.

Chuyên đề tốt nghiệp khoa học dữ liệu lần này bao gồm những dự án thú vị và đa dạng, từ việc áp dụng machine learning và deep learning để giải quyết các vấn đề thực tế, đến việc phân tích và trực quan hóa dữ liệu để hiểu sâu hơn về xu hướng và mối quan hệ. Em cũng đã nghiên cứu các khía cạnh đạo đức và bảo mật trong việc sử dụng dữ liệu.

Chuyên đề này không thể thiếu sự hỗ trợ và động viên từ các thầy cô hướng dẫn, những người đã dành nhiều thời gian và kiến thức để hỗ trợ em trong suốt quá trình nghiên cứu và viết báo cáo. Em xin chân thành cảm ơn sự hướng dẫn và động viên của quý thầy cô.

Cuối cùng, em hy vọng rằng chuyên đề tốt nghiệp khoa học dữ liệu này sẽ đem lại cái nhìn sâu sắc hơn về cách chúng ta có thể sử dụng dữ liệu để giải quyết các vấn đề phức tạp và đóng góp vào sự phát triển của lĩnh vực khoa học dữ liệu. Em rất mong nhận được sự quan tâm và phản hồi tích cực từ quý vị.

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH  **TRUNG TÂM KHẢO THÍ** | **KỲ THI KẾT THÚC HỌC PHẦN**  **HỌC KỲ …..… NĂM HỌC …….. - ….…** |

**PHIẾU CHẤM THI TIỂU LUẬN/BÁO CÁO**

BM-ChT-11

Môn thi: **Chuyên đề tốt nghiệp KHDL 2** Lớp học phần: **20DTH1D**

Nhóm sinh viên thực hiện : **1**

1………………….**Võ Thái Hiển**………..……..Tham gia đóng góp: **100%**

2…………………………………...…………….Tham gia đóng góp:

Ngày thi: Phòng thi:

Đề tài tiểu luận/báo cáo của sinh viên : **Dự đoán từ tiếp theo trong chuỗi văn bản Tensorflow**

Phần đánh giá của giảng viên (căn cứ trên thang rubrics của môn học):

|  |  |  |  |
| --- | --- | --- | --- |
| **Tiêu chí (theo CĐR HP)** | **Đánh giá của GV** | **Điểm tối đa** | **Điểm đạt được** |
| Cấu trúc của báo cáo | ………………………………………………………… |  |  |
| Nội dung | …………………………………………………………  …………………………………………………………  ………………………………………………………… |  |  |
| * Các nội dung thành phần | ………………………………………………………… |  |  |
| * Lập luận | ………………………………………………………… |  |  |
| * Kết luận | …………………………………………………………  ………………………………………………………… |  |  |
| Trình bày | ………………………………………………………..  ………………………………………………………… |  |  |
| **TỔNG ĐIỂM** |  |  |  |

|  |  |
| --- | --- |
|  | **Giảng viên chấm thi**  *(ký, ghi rõ họ tên)* |

**ThS. Sử Nhật Hạ**

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc150522634)

[LỜI MỞ ĐẦU ii](#_Toc150522635)

[MỤC LỤC iv](#_Toc150522636)

[DANH MỤC HÌNH v](#_Toc150522637)

[CHƯƠNG 1./ TỔNG QUAN ĐỀ TÀI 6](#_Toc150522638)

[1.1./ Lý do chọn đề tài 6](#_Toc150522639)

[1.2./ Mục tiêu nghiên cứu 6](#_Toc150522640)

[1.3./ Phương pháp nghiên cứu 6](#_Toc150522641)

[1.4./ Phạm vi nghiên cứu 7](#_Toc150522642)

[CHƯƠNG 2./ GIẢI QUYẾT BÀI TOÁN 8](#_Toc150522643)

[2.1/ Cơ sở lý thuyết 8](#_Toc150522644)

[2.1.1./ Deep Learning 8](#_Toc150522645)

[2.1.2./ Xử lý ngôn ngữ tự nhiên (NLP) 9](#_Toc150522646)

[2.1.3./ TensorFlow 11](#_Toc150522647)

[2.2./ Ứng dụng vào đề tài 12](#_Toc150522648)

[2.2.1./ TensorFlow trong dự đoán từ tiếp theo trong chuỗi văn bản 12](#_Toc150522649)

[2.2.2./ Deep learning trong dự đoán từ tiếp theo trong văn bản 13](#_Toc150522650)

[2.2.3./ Xử lý ngôn ngữ tự nhiên (NLP) trong dự đoán từ tiếp theo trong chuỗi văn bản 14](#_Toc150522651)

[2.3./ Thiết kế mô hình 15](#_Toc150522652)

[CHƯƠNG 3./ DỰ ĐOÁN TỪ TIẾP THEO TRONG CHUỖI VĂN BẢN BẰNG TENSORFLOW 17](#_Toc150522653)

[3.1./ Bộ dữ liệu 17](#_Toc150522654)

[3.2./ Dự đoán từ tiếp theo trong chuỗi văn bản bằng Tensorflow 18](#_Toc150522655)

[3.3./ Kết luận 25](#_Toc150522656)

[3.4./ Ưu điểm và nhược điểm của mô hình 25](#_Toc150522657)

[3.5./ Phương hướng khắc phục 26](#_Toc150522658)

[3.6./ Kết luận mô hình dự đoán 27](#_Toc150522659)

[TÀI LIỆU THAM KHẢO 29](#_Toc150522660)

DANH MỤC HÌNH

[Hình 3. 1./ Ví dụ Import các thư viện 18](#_Toc150522617)

[Hình 3. 2./ Ví dụ đọc tệp CSV 18](#_Toc150522618)

[Hình 3. 3./ Ví dụ code chuyển đổi thành số 19](#_Toc150522619)

[Hình 3. 4./ Ví dụ code phương pháp n-gram 19](#_Toc150522620)

[Hình 3. 5./ Ví dụ code thêm padding 20](#_Toc150522621)

[Hình 3. 6./ Ví dụ code in ra thử 21](#_Toc150522622)

[Hình 3. 7./ Ví dụ code chuẩn bị đầu vào và nhãn 21](#_Toc150522623)

[Hình 3. 8./ Ví dụ code xây dựng, biên dịch và huấn luyện mô hình dự đoán 22](#_Toc150522624)

[Hình 3. 9./ Kết quả của huấn luyện 23](#_Toc150522625)

[Hình 3. 10./ Ví dụ code test mô hình 23](#_Toc150522626)

[Hình 3. 11./ Kết quả của test 24](#_Toc150522627)

CHƯƠNG 1./ TỔNG QUAN ĐỀ TÀI

1.1./ Lý do chọn đề tài

Tính thú vị và đóng góp tri thức: Chủ đề này hấp dẫn vì nó đòi hỏi sử dụng công nghệ học máy và mạng nơ-ron sâu để giải quyết các vấn đề phức tạp trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). Khi nghiên cứu về dự đoán từ tiếp theo, bạn có cơ hội hiểu sâu hơn về cách máy tính hiểu và tạo ra ngôn ngữ, điều này có thể đóng góp vào tri thức lĩnh vực.

Ứng dụng thực tế: Các kỹ thuật dự đoán từ tiếp theo trong chuỗi văn bản có nhiều ứng dụng thực tế, như dịch máy, tóm tắt văn bản, và gợi ý từ. Nghiên cứu về chủ đề này có thể tạo ra giải pháp thực tế và hữu ích cho nhiều lĩnh vực, từ công nghệ thông tin đến ngành truyền thông và giáo dục.

Thách thức và cơ hội: Xây dựng mô hình dự đoán từ tiếp theo trong văn bản là một thách thức kỹ thuật, đặc biệt trong việc hiểu cấu trúc ngôn ngữ và ngữ cảnh. Nghiên cứu trong lĩnh vực này có thể giúp bạn phát triển kỹ năng và kiến thức quan trọng trong học máy và NLP.

Phù hợp với lĩnh vực quan tâm: Lựa chọn một đề tài liên quan đến lĩnh vực quan tâm của bạn có thể giúp bạn nghiên cứu một cách tận tâm và đam mê. Nếu bạn đã có sự quan tâm sâu sắc vào lĩnh vực NLP hoặc học máy, thì việc chọn đề tài này có thể phù hợp.

1.2./ Mục tiêu nghiên cứu

Mục tiêu chính của đề tài này là xây dựng và đánh giá một hệ thống dự đoán từ tiếp theo trong chuỗi văn bản. Điều này đòi hỏi sử dụng các kỹ thuật học máy để hiểu cấu trúc ngôn ngữ và ngữ cảnh trong văn bản. Chúng ta sẽ sử dụng TensorFlow, một thư viện phổ biến cho học máy và mạng nơ-ron, để thực hiện nghiên cứu này.

1.3./ Phương pháp nghiên cứu

**Thu thập dữ liệu:** Thu thập dữ liệu văn bản cần thiết cho nghiên cứu của bạn. Dữ liệu này có thể bao gồm các nguồn từ sách, bài báo, tin tức, hoặc bất kỳ nguồn nào phù hợp với mục tiêu của bạn. Đảm bảo dữ liệu đã được tiền xử lý để loại bỏ dấu câu, ký tự đặc biệt, và chuẩn hóa văn bản.

**Tiền xử lý dữ liệu:** Tiền xử lý dữ liệu là bước quan trọng để chuẩn bị dữ liệu cho việc huấn luyện mô hình. Bạn cần thực hiện tokenization để chia văn bản thành các từ hoặc subword, và sau đó ánh xạ chúng thành biểu diễn số học sử dụng mã hóa từ (word embedding) như Word2Vec, GloVe, hoặc Số hóa từ BERT.

**Xây dựng kiến trúc mô hình:** Lựa chọn kiến trúc mô hình là một phần quan trọng. Có thể sử dụng mô hình RNN (Recurrent Neural Networks), LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit), hoặc Transformer, tùy thuộc vào mục tiêu nghiên cứu. TensorFlow cung cấp các API cho việc xây dựng mô hình mạng nơ-ron sâu.

**Huấn luyện mô hình**: Sử dụng dữ liệu tiền xử lý để huấn luyện mô hình. Bạn cần thiết lập các thông số, như kích thước batch, số vòng lặp, và tốc độ học (learning rate). Cân nhắc việc sử dụng kỹ thuật tối ưu hóa như Adam, RMSProp, hoặc SGD để cải thiện quá trình huấn luyện.

**Đánh giá mô hình**: Đánh giá hiệu suất của mô hình bằng cách sử dụng các phép đo thích hợp cho bài toán, chẳng hạn như perplexity, BLEU score (đối với bài toán dịch máy), hoặc các phép đo khác. Đánh giá trên tập kiểm tra để đảm bảo mô hình tổng quát hóa tốt.

**Tối ưu hóa mô hình:** Nếu mô hình không đạt hiệu suất mong muốn, bạn có thể thử tối ưu hóa bằng cách điều chỉnh siêu tham số, thay đổi kiến trúc mô hình, hoặc sử dụng các biến thể của mô hình để cải thiện kết quả.

**Viết báo cáo và trình bày kết quả:** Cuối cùng, viết báo cáo tóm tắt quá trình nghiên cứu, kết quả, và phân tích. Trình bày kết quả trước các đồng nghiệp hoặc giáo viên hướng dẫn.

1.4./ Phạm vi nghiên cứu

Xây dựng một mô hình mạng nơ-ron sâu (Deep Neural Network) hoặc biến thể của Transformer để dự đoán từ tiếp theo trong chuỗi văn bản.

Xử lý và biểu diễn dữ liệu văn bản một cách hiệu quả để sử dụng cho huấn luyện mô hình.

Thu thập và tiền xử lý dữ liệu huấn luyện, sau đó tiến hành huấn luyện mô hình trên tập dữ liệu này.

Đánh giá hiệu suất của mô hình bằng các phép đo như perplexity, BLEU score, hay các phép đo phù hợp cho bài toán cụ thể.

CHƯƠNG 2./ GIẢI QUYẾT BÀI TOÁN

2.1/ Cơ sở lý thuyết

2.1.1./ Deep Learning

**Mạng nơ-ron sâu (Deep Neural Network - DNN):** là một mô hình học máy với nhiều lớp nơ-ron liên kết với nhau. Mạng này được sử dụng để mô hình hóa và học biểu diễn phức tạp từ dữ liệu đầu vào. DNN có khả năng tự học và rút trích các đặc trưng quan trọng từ dữ liệu, thường được sử dụng trong nhiều nhiệm vụ như phân loại và dự đoán. Các lớp trong DNN thực hiện phép tính phi tuyến tính và sử dụng hàm kích hoạt để đưa ra quyết định phi tuyến tính.

DNN là mô hình học máy cơ bản trong Deep Learning. Nó bao gồm nhiều lớp (layers) của nơ-ron (units) liên kết với nhau và sử dụng hàm kích hoạt (activation function) để học biểu diễn dữ liệu. Trong đề tài, một kiến thức về cách xây dựng và huấn luyện DNN có thể rất hữu ích.

**Mạng nơ-ron hồi quy (RNN)**: là một loại mô hình học máy được sử dụng cho dữ liệu chuỗi, như văn bản hoặc thời gian. Nó có khả năng duy trì trạng thái và sử dụng thông tin từ quá khứ để dự đoán tương lai. Mặc dù RNN hữu ích cho nhiều ứng dụng chuỗi, nó có thể gặp vấn đề với gradient biến mất. LSTM và GRU là các biến thể của RNN giúp giải quyết vấn đề này và được sử dụng rộng rãi trong xử lý ngôn ngữ tự nhiên và các lĩnh vực khác.

RNN là mô hình đặc biệt quan trọng khi làm việc với dữ liệu chuỗi, như văn bản. Nó có khả năng lưu trạng thái trước đó và sử dụng nó để ảnh hưởng đến dự đoán tiếp theo. LSTM (Long Short-Term Memory) và GRU (Gated Recurrent Unit) là biến thể của RNN phổ biến có khả năng nắm bắt thông tin ngữ cảnh trong dữ liệu chuỗi.

**Mô hình Transformer** là một mô hình mạng nơ-ron sâu đặc biệt trong lĩnh vực xử lý ngôn ngữ tự nhiên và dữ liệu chuỗi. Nó sử dụng cơ chế chú ý để hiệu quả hóa việc mô hình hóa thông tin quan trọng trong dữ liệu, loại bỏ sự phụ thuộc vào chuỗi, và đã được áp dụng rộng rãi trong nhiều ứng dụng NLP và các lĩnh vực khác.

Mô hình Transformer đã trở thành một kiến thức quan trọng trong NLP. Nó sử dụng cơ chế attention để đặc biện văn bản và dự đoán từ tiếp theo. Trong nghiên cứu của bạn, việc hiểu cách Transformer hoạt động và cách thực hiện chúng trong TensorFlow có thể rất quan trọng.

**Hàm mất mát (Loss Function)** là một phần quan trọng trong quá trình huấn luyện mô hình học máy. Nó đo lường sự sai khác giữa dự đoán của mô hình và giá trị thực tế (ground truth). Mục tiêu của việc tối ưu hóa mô hình là tìm ra các tham số mô hình sao cho giá trị của hàm mất mát là nhỏ nhất.

Hàm mất mát có nhiều loại khác nhau, tùy thuộc vào loại bài toán mà bạn đang giải quyết. Dưới đây là một số hàm mất mát phổ biến trong các loại bài toán khác nhau:

* Hàm mất mát Mean Squared Error (MSE): Thường được sử dụng trong bài toán hồi quy (regression). Nó đo lường sai số bình phương giữa dự đoán và giá trị thực tế. Công thức: MSE = (1/n) \* Σ(y - ŷ)^2, trong đó y là giá trị thực tế, ŷ là dự đoán, và n là số lượng mẫu.
* Hàm mất mát Cross-Entropy (CE): Thường được sử dụng trong bài toán phân loại (classification). Nó đo lường mức độ không chắc chắn trong dự đoán và giá trị thực tế. Công thức: CE = -Σ(y \* log(ŷ)), trong đó y là phân phối xác suất thực tế và ŷ là phân phối xác suất dự đoán.
* Hàm mất mát Hinge Loss (SVM Loss): Thường được sử dụng trong bài toán phân loại SVM (Support Vector Machine). Nó đo lường khoảng cách từ các điểm dữ liệu đến siêu phẳng phân chia. Công thức: Hinge Loss = Σ(max(0, 1 - y \* ŷ)), trong đó y là nhãn thực tế và ŷ là dự đoán.
* Hàm mất mát Log Loss (Logistic Loss): Thường được sử dụng trong bài toán phân loại nhị phân. Nó đo lường mức độ sai lệch giữa xác suất dự đoán và xác suất thực tế. Công thức: Log Loss = -Σ(y \* log(ŷ) + (1 - y) \* log(1 - ŷ)), trong đó y là nhãn thực tế và ŷ là xác suất dự đoán.
* Hàm mất mát Triplet Loss: Thường được sử dụng trong bài toán xác định khoảng cách giữa các điểm dữ liệu. Nó đo lường khoảng cách giữa điểm gốc và điểm gần nhất so với điểm gốc và điểm xa nhất. Công thức: Triplet Loss = max(0, margin + d(G, P) - d(G, N)), trong đó G là điểm gốc, P là điểm gần nhất, N là điểm xa nhất, và d là hàm khoảng cách.

Lựa chọn hàm mất mát phù hợp cho bài toán dự đoán từ tiếp theo trong chuỗi văn bản là quan trọng. Điều này có thể liên quan đến việc đánh giá sai số giữa dự đoán và thực tế, và cách tính toán hàm mất mát sẽ ảnh hưởng đến quá trình huấn luyện.

2.1.2./ Xử lý ngôn ngữ tự nhiên (NLP)

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực quan trọng trong trí tuệ nhân tạo (AI) liên quan đến việc làm cho máy tính hiểu và tạo ra văn bản ngôn ngữ tự nhiên (NN). Dưới đây là một số khía cạnh chi tiết về lý thuyết NLP:

**Tokenization** là quá trình chia một đoạn văn bản thành các phần nhỏ hơn gọi là "token." Một token có thể là một từ, một dấu câu, hoặc một đơn vị ngữ cảnh khác. Quá trình tokenization giúp máy tính hiểu cấu trúc của văn bản và làm cho việc xử lý ngôn ngữ tự nhiên trở nên dễ dàng hơn.

Ví dụ, khi bạn có một câu như "Ngày mai là thứ Năm.", quá trình tokenization sẽ chia nó thành các token như sau:

"Ngày"

"mai"

"là"

"thứ"

"Năm"

"."

Tokenization là quá trình chia văn bản thành các đơn vị nhỏ hơn, gọi là "token." Token có thể là từ, dấu câu, hoặc đơn vị ngữ cảnh khác. Tokenization quyết định cách dữ liệu văn bản được chia thành và ảnh hưởng đến quá trình xử lý ngôn ngữ tự nhiên.

**Biểu diễn từ (Word Embeddings)** là một kỹ thuật trong xử lý ngôn ngữ tự nhiên (NLP) để biểu diễn từ vựng trong dữ liệu văn bản dưới dạng các vector số học. Các vector này có thể được sử dụng để biểu diễn và hiểu các mối quan hệ ngữ nghĩa giữa từ và từ, làm cho việc làm việc với văn bản trở nên hiệu quả hơn.

Một số phương pháp biểu diễn từ phổ biến trong Word Embeddings bao gồm:

* Word2Vec: Sử dụng mô hình CBOW (Continuous Bag of Words) hoặc Skip-gram để học biểu diễn từ dựa trên ngữ cảnh. Word2Vec tạo ra vector số học sao cho các từ cùng ngữ cảnh thường có các vector gần giống nhau.
* GloVe (Global Vectors for Word Representation): Sử dụng thống kê toàn bộ dữ liệu văn bản để học biểu diễn từ. GloVe tạo ra các vector sao cho các từ thường xuất hiện cùng nhau có các vector gần giống nhau.
* FastText: Mở rộng Word2Vec bằng cách biểu diễn từng từ dưới dạng các subword hoặc n-gram. Điều này cho phép FastText xử lý từ mới và từ lạ.

**Mạng nơ-ron sâu (Deep Learning)** trong xử lý ngôn ngữ tự nhiên (NLP) là việc sử dụng mạng nơ-ron có nhiều lớp (lớp ẩn) để mô hình hóa và xử lý dữ liệu văn bản. Mạng nơ-ron sâu (DNN) trong NLP được sử dụng để tạo ra biểu diễn phức tạp của ngôn ngữ và hiểu các mối quan hệ ngữ nghĩa trong văn bản.

Một số loại mạng nơ-ron sâu thường được sử dụng trong NLP bao gồm:

* RNN (Recurrent Neural Networks): RNN là một loại mạng nơ-ron sâu có khả năng xử lý dữ liệu chuỗi, chẳng hạn như văn bản. RNN có các trạng thái ẩn để duy trì thông tin về quá khứ và sử dụng thông tin này để dự đoán tương lai. Tuy nhiên, RNN có thể gặp vấn đề với gradient biến mất và khó khăn trong việc kết nối thông tin xa trong chuỗi.
* LSTM (Long Short-Term Memory): LSTM là một biến thể của RNN được thiết kế để giải quyết vấn đề gradient biến mất. LSTM sử dụng cơ chế cổng để kiểm soát việc cập nhật và ghi nhớ thông tin trong trạng thái ẩn, giúp nó học các phụ thuộc xa trong chuỗi một cách hiệu quả.
* GRU (Gated Recurrent Unit): GRU cũng là một biến thể của RNN với cơ chế cổng, nhưng đơn giản hơn so với LSTM. Nó giữ lại một phần của thông tin trong trạng thái ẩn và có hiệu suất tương đương với LSTM trong nhiều tác vụ.
* Mạng nơ-ron sâu với cơ chế chú ý (Deep Learning with Attention): Mạng nơ-ron sâu có thể được kết hợp với cơ chế chú ý, chẳng hạn như Transformer, để cải thiện khả năng hiểu ngữ cảnh và quan hệ ngữ nghĩa giữa các từ trong văn bản.

Mạng nơ-ron sâu như RNN, LSTM, GRU, và Transformer được sử dụng trong NLP để mô hình hóa dữ liệu chuỗi, xử lý ngôn ngữ tự nhiên và thực hiện các tác vụ như phân loại văn bản, dự đoán từ tiếp theo, tóm tắt văn bản, và dịch máy.

2.1.3./ TensorFlow

TensorFlow là một thư viện mã nguồn mở phát triển bởi Google dùng để xây dựng và huấn luyện mô hình học máy. Nó là một trong những thư viện phổ biến nhất trong lĩnh vực học máy và deep learning. Dưới đây là một số chi tiết về lý thuyết và cách TensorFlow hoạt động:

**Khái niệm TensorFlow**: TensorFlow được xây dựng trên ý tưởng về biểu đồ tính toán dưới dạng các đồ thị, trong đó các node đại diện cho các phép tính và các cạnh đại diện cho dữ liệu (tensor) chuyển từ node này sang node khác. Điều này giúp TensorFlow thực hiện tính toán song song và tận dụng sức mạnh của các thiết bị tính toán, chẳng hạn như CPU hoặc GPU.

**Tensor:** Tensor là một cấu trúc dữ liệu chính trong TensorFlow. Nó tương tự như một mảng nhiều chiều (ví dụ: vector, ma trận) có thể chứa dữ liệu số học. Tensors là đơn vị dữ liệu được truyền qua các node trong biểu đồ tính toán TensorFlow.

Biểu đồ tính toán (Computational Graph): Mô hình tính toán trong TensorFlow được biểu diễn dưới dạng biểu đồ tính toán. Biểu đồ này định nghĩa cách các tensor được tạo ra và phép tính được thực hiện trong mô hình.

**Session**: Session là môi trường thực thi biểu đồ tính toán TensorFlow. Bạn cần tạo một session để chạy biểu đồ và thực hiện tính toán.

**Variable**: TensorFlow cung cấp biến (Variable) để lưu trữ các tham số của mô hình mà bạn muốn tối ưu hóa trong quá trình huấn luyện. Biến này có thể được cập nhật theo gradient để điều chỉnh mô hình.

**Optimizer**: Optimizer là một thành phần quan trọng của TensorFlow được sử dụng để tối ưu hóa mô hình bằng cách điều chỉnh các biến để giảm hàm mất mát (loss function).

**Keras Integration**: TensorFlow có tích hợp sâu với Keras, một thư viện học máy cao cấp, để giúp xây dựng mô hình dễ dàng và nhanh chóng.

**GPU Acceleration**: TensorFlow hỗ trợ tích hợp với GPU để tăng tốc quá trình huấn luyện mô hình. Điều này làm cho việc huấn luyện mô hình deep learning trở nên hiệu quả hơn.

TensorFlow cung cấp một nền tảng mạnh mẽ cho việc phát triển và triển khai các mô hình học máy và deep learning. Nó đã được sử dụng rộng rãi trong nhiều ứng dụng, chẳng hạn như xử lý ngôn ngữ tự nhiên, thị giác máy tính, dự đoán chuỗi thời gian, và nhiều ứng dụng AI khác.

2.2./ Ứng dụng vào đề tài

2.2.1./ TensorFlow trong dự đoán từ tiếp theo trong chuỗi văn bản

TensorFlow là một thư viện mạnh mẽ để xây dựng mô hình dự đoán từ tiếp theo trong chuỗi văn bản. Việc sử dụng TensorFlow cho dự đoán từ tiếp theo trong chuỗi văn bản liên quan đến việc xây dựng mô hình mạng nơ-ron sâu (deep learning) như RNN (Recurrent Neural Network) hoặc Transformer để xử lý chuỗi văn bản.

**Chuẩn bị dữ liệu:** Đầu tiên, bạn cần chuẩn bị dữ liệu văn bản cho mô hình. Dữ liệu này có thể được chia thành các chuỗi (sequences) và biểu diễn dưới dạng các vector số học, ví dụ: biểu diễn từ (word embeddings).

**Xây dựng mô hình:** Sử dụng TensorFlow, bạn có thể xây dựng một mô hình mạng nơ-ron sâu, chẳng hạn như RNN hoặc LSTM, để mô hình hóa dữ liệu chuỗi và dự đoán từ tiếp theo. Mô hình này sẽ có một lớp đầu vào để chứa chuỗi và một lớp đầu ra để dự đoán từ tiếp theo.

**Huấn luyện mô hình:** Bạn sẽ sử dụng dữ liệu huấn luyện để huấn luyện mô hình. Trong quá trình này, mô hình sẽ cố gắng học cách dự đoán từ tiếp theo trong chuỗi văn bản dựa trên thông tin từ chuỗi trước đó.

**Dự đoán từ tiếp theo:** Sau khi mô hình đã được huấn luyện, bạn có thể sử dụng nó để dự đoán từ tiếp theo trong chuỗi văn bản. Bạn cung cấp một chuỗi đầu vào và mô hình sẽ tạo ra dự đoán cho từ tiếp theo.

TensorFlow cung cấp các tính năng mạnh mẽ để tối ưu hóa mô hình, quản lý dữ liệu và triển khai trên các nền tảng khác nhau. Nó đã được sử dụng rộng rãi trong các ứng dụng dự đoán từ tiếp theo trong chuỗi văn bản, chẳng hạn như tạo văn bản tự động, gợi ý từ, hoặc dự đoán ngôn ngữ tự nhiên.

2.2.2./ Deep learning trong dự đoán từ tiếp theo trong văn bản

Deep learning có một vai trò quan trọng trong bài toán dự đoán từ tiếp theo trong chuỗi văn bản. Các mô hình sâu có khả năng học và hiểu các mối quan hệ phức tạp trong dữ liệu, giúp cải thiện khả năng dự đoán và tạo ra kết quả chính xác hơn. Dưới đây là một số phương pháp deep learning thường được sử dụng trong bài toán này:

**Recurrent Neural Networks (RNN):** RNN là một loại mạng nơ-ron sâu được thiết kế đặc biệt để xử lý dữ liệu chuỗi. Trong bài toán dự đoán từ tiếp theo trong chuỗi văn bản, RNN có thể duyệt qua từng từ trong chuỗi và giữ lại thông tin ngữ cảnh từ các từ trước đó để dự đoán từ tiếp theo.

**Long Short-Term Memory (LSTM) Networks**: Là một biến thể của RNN, LSTM giải quyết vấn đề gradient biến mất và giữ cho mô hình có khả năng học các phụ thuộc xa trong dữ liệu chuỗi.

**Gated Recurrent Unit (GRU)**: GRU cũng là một biến thể của RNN giống như LSTM, nhưng có cấu trúc đơn giản hơn. Nó giữ lại thông tin quan trọng từ quá khứ để ảnh hưởng đến dự đoán từ tiếp theo.

**Transformer-Based Models**: Các mô hình như GPT (Generative Pre-trained Transformer) và BERT (Bidirectional Encoder Representations from Transformers) sử dụng kiến trúc Transformer, không sử dụng RNN. Transformer cho phép mô hình chú ý đến các phần quan trọng của chuỗi văn bản một cách hiệu quả.

**Sequence-to-Sequence Models**: Các mô hình này, như Seq2Seq, được sử dụng để ánh xạ một chuỗi đầu vào sang một chuỗi đầu ra. Trong trường hợp dự đoán từ tiếp theo trong chuỗi văn bản, chúng có thể được sử dụng để dự đoán từ kế tiếp dựa trên ngữ cảnh trước đó.

**Transfer Learning**: Sử dụng pre-trained mô hình như GPT hoặc BERT để học biểu diễn từ có chất lượng cao từ dữ liệu lớn trước đó, sau đó fine-tune mô hình cho công việc cụ thể của bạn.

**Attention Mechanisms:** Cơ chế chú ý, như được sử dụng trong Transformer, cho phép mô hình tập trung vào các phần quan trọng của chuỗi văn bản khi thực hiện dự đoán từ tiếp theo.

Sử dụng deep learning trong dự đoán từ tiếp theo trong chuỗi văn bản có thể giúp mô hình hiểu ngữ cảnh, mối quan hệ ngữ nghĩa, và tạo ra dự đoán chính xác dựa trên thông tin từ các từ trước đó trong chuỗi.

2.2.3./ Xử lý ngôn ngữ tự nhiên (NLP) trong dự đoán từ tiếp theo trong chuỗi văn bản

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực quan trọng trong bài toán dự đoán từ tiếp theo trong chuỗi văn bản. Dưới đây là một số kỹ thuật và phương pháp NLP thường được sử dụng để cải thiện hiệu suất dự đoán từ tiếp theo trong mô hình:

**Tokenization và Embedding**: Tokenization là quá trình chia văn bản thành các đơn vị nhỏ hơn, gọi là token (từ, dấu câu, hoặc ngữ cảnh khác). Embedding là quá trình ánh xạ các từ này sang các vector số học. Việc này giúp mô hình hiểu được ngữ cảnh và mối quan hệ ngữ nghĩa giữa các từ.

**Word Embeddings**: Sử dụng các mô hình như Word2Vec, GloVe, hoặc FastText để tạo ra biểu diễn số học cho từng từ. Các biểu diễn này giữ lại mối quan hệ ngữ nghĩa giữa từng từ trong không gian nhiều chiều.

**Bidirectional Context:** Mô hình có thể học từ cả hai hướng (bidirectional) để có cái nhìn toàn diện về ngữ cảnh. Mô hình bidirectional RNN hoặc Transformer giúp giữ lại thông tin từ cả phía trước và phía sau của từ đang xem xét.

**Attention Mechanisms:** Sử dụng cơ chế chú ý để tập trung vào các phần quan trọng của chuỗi văn bản khi thực hiện dự đoán từ tiếp theo. Điều này giúp mô hình tập trung vào các từ quan trọng trong ngữ cảnh.

**Mô hình mạng nơ-ron gia đình Transformer**: Các mô hình như GPT (Generative Pre-trained Transformer) cung cấp biểu diễn ngôn ngữ tự nhiên mạnh mẽ và có khả năng thực hiện dự đoán từ tiếp theo dựa trên ngữ cảnh.

**Transfer Learning:** Fine-tune mô hình đã được huấn luyện trước đó trên một tác vụ NLP lớn và sau đó sử dụng nó để dự đoán từ tiếp theo trong chuỗi văn bản.

**Generative Models**: Sử dụng mô hình sinh văn bản để tạo ra các từ tiếp theo dựa trên ngữ cảnh hiện tại. Các mô hình như LSTM và Transformer có thể được sử dụng để tạo ra văn bản mới từ mô hình đã được huấn luyện.

**Metric Learning:** Sử dụng kỹ thuật này để tối ưu hoá không gian biểu diễn, giúp mô hình hiểu mối quan hệ ngữ nghĩa giữa các từ và từ đó cải thiện khả năng dự đoán từ tiếp theo.

Sự kết hợp của những phương pháp này có thể tạo ra mô hình mạnh mẽ trong bài toán dự đoán từ tiếp theo trong chuỗi văn bản, đặc biệt là khi đối mặt với ngôn ngữ tự nhiên và đa dạng.

2.3./ Thiết kế mô hình

Để thiết kế một mô hình dự đoán từ tiếp theo trong chuỗi văn bản, chúng ta cần cân nhắc các yếu tố như loại mô hình, kiến trúc mạng nơ-ron, và các tham số cụ thể. Dưới đây là một thiết kế bằng lý thuyết với sự hướng dẫn chi tiết:

**Lựa chọn Loại Mô Hình:**

Sử dụng một mô hình mạng nơ-ron sâu, vì chúng thường xuất sắc trong việc học mối quan hệ phức tạp trong dữ liệu chuỗi văn bản.

Cân nhắc sử dụng mô hình RNN, LSTM, hoặc Transformer. Mỗi loại có những ưu điểm và hạn chế riêng.

**Xử Lý Dữ Liệu:**

Chuẩn bị dữ liệu văn bản và thực hiện Tokenization để chia văn bản thành các đơn vị nhỏ hơn như từ hoặc ký tự.

Tạo biểu diễn số học cho từng từ, chẳng hạn như sử dụng Embedding Layer hoặc pre-trained Word Embeddings.

**Kiến Trúc Mạng Nơ-Ron:**

Sử dụng Embedding Layer để chuyển đổi từng từ thành biểu diễn số học.

Sử dụng một hoặc nhiều layer RNN hoặc LSTM để học ngữ cảnh chuỗi. Đối với Transformer, sử dụng các layer Attention.

Thêm một hoặc nhiều layer Dense ở cuối để dự đoán từ tiếp theo.

**Học và Tối Ưu Hóa:**

Sử dụng hàm mất mát phù hợp, ví dụ như categorical crossentropy, vì đây là bài toán phân loại đa lớp.

Chọn optimizer, chẳng hạn như Adam, và cân nhắc sử dụng learning rate schedule để điều chỉnh tốc độ học.

**Fine-tuning và Đánh Giá:**

Huấn luyện mô hình trên dữ liệu huấn luyện và đánh giá trên dữ liệu kiểm thử.

Fine-tune các tham số mô hình để cải thiện hiệu suất, chẳng hạn như số lượng đơn vị trong mỗi layer, số lượng layer, hoặc learning rate.

**Overfitting và Regularization:**

Đối mặt với vấn đề overfitting bằng cách sử dụng kỹ thuật như dropout trong các layer hoặc sử dụng các biện pháp regularization.

**Transfer Learning (Tuỳ Chọn):**

Nếu có sẵn dữ liệu lớn, cân nhắc sử dụng mô hình đã được huấn luyện trước (pre-trained model) và fine-tune nó cho bài toán cụ thể của bạn.

**Đánh Giá Hiệu Suất:**

Đánh giá hiệu suất của mô hình trên các metric như accuracy, precision, recall, và F1-score trên tập kiểm thử.

**Tối Ưu Hóa và Điều Chỉnh:**

Nếu cần, thực hiện các bước điều chỉnh để cải thiện hiệu suất của mô hình.

CHƯƠNG 3./   
DỰ ĐOÁN TỪ TIẾP THEO TRONG CHUỖI VĂN BẢN BẰNG TENSORFLOW

3.1./ Bộ dữ liệu

Bộ dữ liệu "medium\_data.csv." Bộ dữ liệu này bao gồm thông tin về các bài viết trên nền tảng Medium, với mỗi dòng biểu diễn một mẫu dữ liệu độc lập. Dưới đây là một cái nhìn tổng quan về cấu trúc và nội dung của bộ dữ liệu:

**1. Các Cột Chính:**

**id:** ID duy nhất của mỗi mẫu dữ liệu.

**url:** Đường link đến bài viết trên Medium.

**title:** Tiêu đề của bài viết.

**subtitle:** Phần mô tả bổ sung hoặc tiêu đề phụ của bài viết.

**image:** Tên file ảnh đính kèm với bài viết.

**claps:** Số lượng claps (lượt nhấn nút "clap" trên Medium) cho bài viết.

**responses:** Số lượng phản hồi (comments) của độc giả.

**reading\_time:** Thời gian ước lượng để đọc bài viết (trong phút).

**publication:** Tên của phương tiện xuất bản trên Medium.

**date:** Ngày xuất bản bài viết.

**2. Mục Tiêu của Đề Tài:**

Mục tiêu chính của đề tài là dự đoán từ tiếp theo trong chuỗi văn bản, có thể được liên kết với nội dung trong cột "title" hoặc "subtitle."

**3. Dữ Liệu Chuỗi Văn Bản:**

- Cột "title" chứa thông tin về tiêu đề của bài viết.

- Cột "subtitle" cung cấp mô tả bổ sung hoặc tiêu đề phụ.

**4. Dữ Liệu Số Lượng và Thời Gian:**

- Cột "claps" có thể sử dụng làm yếu tố số lượng để đo lường sự quan tâm từ cộng đồng.

- Cột "responses" cung cấp thông tin về mức độ tương tác qua các phản hồi từ độc giả.

- Cột "reading\_time" có thể đóng vai trò như yếu tố thời gian để đọc bài viết.

**5. Ảnh Đính Kèm:**

Cột "image" chứa tên file ảnh đính kèm với mỗi bài viết, có thể là một hình minh họa hoặc hình ảnh liên quan.

**6. Phương Tiện Xuất Bản:**

Cột "publication" chứa tên phương tiện xuất bản trên Medium, cung cấp nguồn gốc của bài viết.

Dữ liệu này mang lại cơ hội để xây dựng mô hình dự đoán từ tiếp theo trong chuỗi văn bản, kết hợp thông tin văn bản và các yếu tố số lượng để hiểu rõ hơn về sự tương tác và quan tâm của độc giả đối với các bài viết trên Medium.

3.2./ Dự đoán từ tiếp theo trong chuỗi văn bản bằng Tensorflow

Triển khai một mô hình dự đoán từ tiếp theo trong chuỗi văn bản bằng cách sử dụng mạng nơ-ron sâu (Deep Neural Network - DNN) với kiến trúc LSTM (Long Short-Term Memory) và Embedding Layer

Ảnh có chứa văn bản, phần mềm, Phần mềm đa phương tiện, ảnh chụp màn hình

Mô tả được tạo tự động

Hình 3. 1./ Ví dụ Import các thư viện

**Pandas:** Thư viện được sử dụng để làm việc với dữ liệu bảng (DataFrame) từ tệp CSV.

**Numpy:** Thư viện để thực hiện các phép toán số học trên mảng và ma trận.

**TensorFlow:** Thư viện máy học sâu để xây dựng và huấn luyện mô hình nơ-ron.

**Keras:** Giao diện trên TensorFlow, cung cấp mô hình và lớp cho việc xây dựng mạng nơ-ron.

Tiếp theo chúng ta tiến hành sử dụng Pandas để đọc dữ liệu từ tệp CSV.

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Mô tả được tạo tự động

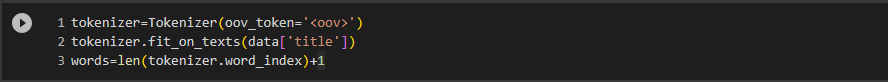
Hình 3. 2./ Ví dụ đọc tệp CSV

Sau đó chuẩn bị dữ liệu văn bản cho quá trình xây dựng mô hình dự đoán từ tiếp theo trong chuỗi văn bản. Cụ thể:

**Tokenization:** Chuyển đổi văn bản thành chuỗi số, trong đó mỗi từ được ánh xạ thành một số nguyên.

**Xây dựng từ điển từ vựng:** Dùng để ánh xạ từng từ trong văn bản thành một số nguyên.

**Chuẩn bị cho Embedding Layer trong mô hình:** Số lượng từ trong từ điển được sử dụng để cấu hình đầu vào của lớp Embedding, một lớp quan trọng trong mô hình nơ-ron, giúp biểu diễn từng từ dưới dạng các vector số.



Hình 3. 3./ Ví dụ code chuyển đổi thành số

**Tokenizer:** Lớp chuyển đổi văn bản thành chuỗi số.

**oov\_token='<oov>':** Token đặc biệt để đại diện cho các từ không xuất hiện trong từ điển.

**fit\_on\_texts(data['title']):** Huấn luyện Tokenizer trên cột "title" của DataFrame để xây dựng từ điển từ vựng.

**len(tokenizer.word\_index) + 1:** Số lượng từ trong từ điển, bao gồm cả token đặc biệt **"<oov>".**

Tiếp theo chúng ta tạo ra các chuỗi đầu vào cho mô hình dự đoán từ tiếp theo trong chuỗi văn bản bằng cách sử dụng phương pháp "n-gram"

Ảnh có chứa văn bản, Phần mềm đa phương tiện, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động

Hình 3. 4./ Ví dụ code phương pháp n-gram

**input\_sequences:** List chứa các chuỗi đầu vào cho mô hình.

**for line in data['title']:** Duyệt qua mỗi dòng trong cột "title" của DataFrame data.

**token\_list = tokenizer.texts\_to\_sequences([line])[0]:** Chuyển đổi mỗi dòng văn bản thành chuỗi số bằng cách sử dụng Tokenizer. Dòng mã này trả về một danh sách các số nguyên.

**for i in range(1, len(token\_list)):** Duyệt qua từng vị trí trong danh sách số nguyên, bắt đầu từ vị trí thứ nhất (vị trí 0 là độ dài 1).

**n\_gram\_sequence = token\_list[:i+1]:** Tạo một chuỗi n-gram (có nghĩa là một chuỗi con từ đầu đến vị trí hiện tại i). Điều này tạo ra các chuỗi từ 2 đến len(token\_list) từ mỗi dòng văn bản.

**input\_sequences.append(n\_gram\_sequence):** Thêm chuỗi n-gram này vào danh sách input\_sequences. Các chuỗi này sẽ được sử dụng làm đầu vào cho mô hình để dự đoán từ tiếp theo trong chuỗi văn bản.

Đoạn mã này tạo ra một tập hợp các chuỗi đầu vào được sử dụng để huấn luyện mô hình, trong đó mỗi chuỗi bao gồm một phần của văn bản và mục tiêu dự đoán là từ tiếp theo trong chuỗi.

Sau khi đã có chuỗi đầu vào để có độ dài thống nhất ta thực hiện việc thêm padding



Hình 3. 5./ Ví dụ code thêm padding

**max([len(x) for x in input\_sequences]):** Tìm độ dài lớn nhất của các chuỗi trong **input\_sequences**. Điều này sẽ được sử dụng làm độ dài tối đa cho chuỗi đầu vào.

**pad\_sequences(input\_sequences, maxlen=max\_sequence\_len, padding='pre'):** Sử dụng hàm pad\_sequences từ thư viện Keras để thêm padding cho các chuỗi đầu vào. Các tham số cụ thể:

**input\_sequences**: Các chuỗi đầu vào cần được thêm padding.

**maxlen=max\_sequence\_len:** Độ dài tối đa của chuỗi sau khi thêm padding, đã được xác định bởi độ dài lớn nhất trước đó.

**padding='pre':** Thêm padding ở phía trước của mỗi chuỗi.

**np.array(...):** Chuyển đổi kết quả của pad\_sequences thành một mảng NumPy để sử dụng trong quá trình huấn luyện mô hình.

Đoạn mã này đảm bảo rằng tất cả các chuỗi đầu vào có cùng độ dài bằng cách thêm padding ở phía trước của mỗi chuỗi để có độ dài bằng với max\_sequence\_len. Điều này làm cho dữ liệu đầu vào có độ dài đồng nhất, phù hợp với yêu cầu đầu vào của mô hình nơ-ron.

Để hiểu rõ hơn về cấu trúc của **input\_sequences**, bạn có thể in **input\_sequences** hoặc kiểm tra các chiều của nó bằng **input\_sequences.shape**.

Ảnh có chứa ảnh chụp màn hình, văn bản

Mô tả được tạo tự động

Hình 3. 6./ Ví dụ code in ra thử

Kết quả của **input\_sequences[2]** là một mảng số nguyên có độ dài 39, được hiển thị dưới dạng mảng NumPy. Mỗi số nguyên trong mảng này biểu diễn một từ trong văn bản sau khi đã được chuẩn bị và chuyển đổi thành số bằng Tokenizer.

**Phần đầu của mảng là các giá trị 0**: Đây là padding được thêm vào mỗi chuỗi để đảm bảo chúng có cùng độ dài. Số lượng giá trị 0 này phụ thuộc vào độ dài tối đa của chuỗi trong tập dữ liệu.

**Sau đó là các số nguyên khác nhau**: Các số nguyên này biểu diễn các từ trong văn bản. Số 5 là một trong những từ xuất hiện đầu tiên trong từ điển từ vựng, và các số nguyên tiếp theo là mã hóa cho các từ tiếp theo trong chuỗi.

Tiếp theo chuẩn bị dữ liệu đầu vào và nhãn cho quá trình huấn luyện mô hình.



Hình 3. 7./ Ví dụ code chuẩn bị đầu vào và nhãn

**xs**: Là các chuỗi đầu vào, được tạo bằng cách lấy tất cả các cột của **input\_sequences** trừ cột cuối cùng.

**labels**: Là các nhãn tương ứng với mỗi chuỗi đầu vào, được tạo bằng cách lấy cột cuối cùng của **input\_sequences.**

**to\_categorical(labels, num\_classes=words):** Chuyển đổi các nhãn thành dạng one-hot encoding, là một biểu diễn nhãn dạng vectơ có độ dài bằng số lượng từ trong từ điển (words). Cụ thể, mỗi nhãn là một vectơ có độ dài **words** với giá trị 1 tại vị trí tương ứng với từ có nhãn và giá trị 0 ở các vị trí khác.

Tổng cộng, **xs** là tập hợp các chuỗi đầu vào, **labels** là tập hợp các nhãn tương ứng, và **ys** là biểu diễn one-hot encoding của nhãn. Các tập dữ liệu này sẽ được sử dụng để huấn luyện mô hình dự đoán từ tiếp theo trong chuỗi văn bản.

Sau khi đã chuẩn bị đầy đủ, bây giờ chúng ta có thể bắt đầu xây dựng, biên dịch và huấn luyện mô hình dự đoán từ tiếp theo trong chuỗi văn bản

Ảnh có chứa văn bản, phần mềm, Phần mềm đa phương tiện, Phông chữ

Mô tả được tạo tự động

Hình 3. 8./ Ví dụ code xây dựng, biên dịch và huấn luyện mô hình dự đoán

**Embedding(words, 100, input\_length=max\_sequence\_len-1):** Lớp Embedding trong mô hình, trong đó:

**words**: Số lượng từ trong từ điển (đã được xác định trước đó).

**100**: Kích thước của vectơ embedding cho mỗi từ.

**input\_length=max\_sequence\_len-1**: Độ dài của mỗi chuỗi đầu vào, đã được xác định trước đó.

**Bidirectional(LSTM(150)):** Lớp LSTM (Long Short-Term Memory) được sử dụng để học các mối quan hệ dài hạn trong dữ liệu chuỗi. Bidirectional LSTM cho phép mô hình học từ cả hai hướng của chuỗi. Số 150 là số đơn vị LSTM.

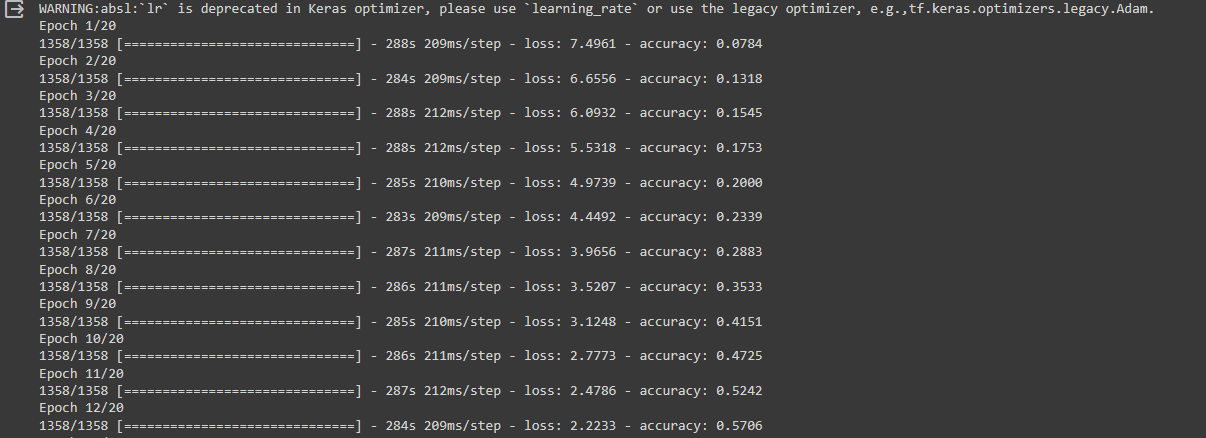
**Dense(words, activation='softmax'):** Lớp kết nối đầy đủ (fully connected) với activation function là softmax, được sử dụng để dự đoán xác suất của từ tiếp theo trong chuỗi.

**Adam(lr=0.01):** Sử dụng tối ưu hóa Adam với learning rate là 0.01 để cập nhật trọng số của mô hình.

**model.compile(...):** Biên dịch mô hình với hàm mất mát là categorical crossentropy (do ta sử dụng softmax ở lớp cuối) và tối ưu hóa bằng Adam. Đồng thời, mô hình sẽ theo dõi độ chính xác trong quá trình huấn luyện.

**model.fit(...):** Huấn luyện mô hình trên dữ liệu đầu vào **(xs)** và nhãn **(ys)** trong 20 epochs. **verbose=1** để hiển thị thông tin huấn luyện.

Đoạn mã này xây dựng một mô hình mạng nơ-ron dự đoán từ tiếp theo trong chuỗi văn bản sử dụng Embedding, Bidirectional LSTM và Dense layers. Mô hình được huấn luyện trên dữ liệu đã được chuẩn bị và biên dịch sử dụng hàm mất mát là categorical crossentropy và tối ưu hóa bằng Adam.

Output: 

Hình 3. 9./ Kết quả của huấn luyện

Các giá trị loss giảm và độ chính xác tăng là những dấu hiệu tích cực. Tuy nhiên, việc đánh giá mô hình tốn khá nhiều thời gian

Như vậy là chúng ta đã hoàn thành xong quá trình chuẩn bị và huấn luyện cho mô hình bây giờ chúng ta sẽ thử xem nó có hoạt động được hay không.

Chúng ta sẽ bắt đầu thử xem với 3 ví dụ:

Ảnh có chứa văn bản, đồ điện tử, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động

Hình 3. 10./ Ví dụ code test mô hình

Cho một câu văn đầu vào

sentence = "We have subject as Machine"

Chuyển đổi câu văn thành chuỗi số bằng Tokenizer và thêm padding để có độ dài phù hợp với đầu vào của mô hình.

tokenized\_sentence = tokenizer.texts\_to\_sequences([sentence])

input\_sequence = pad\_sequences(tokenized\_sentence, maxlen=max\_sequence\_len-1, padding='pre')

Dự đoán xác suất từng từ cho từng từ trong chuỗi đầu vào, sau đó chọn từ có xác suất cao nhất làm từ tiếp theo.

predicted\_probs = model.predict(input\_sequence)

predicted\_word\_index = np.argmax(predicted\_probs)

predicted\_word = tokenizer.index\_word[predicted\_word\_index]

In ra từ được dự đoán là tiếp theo trong câu.

print("Next word:", predicted\_word)

Ví dụ 2 và 3 tương tự như vậy nhưng khác câu văn

Output:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Hình 3. 11./ Kết quả của test

**Example 1:**

Câu văn: "We have subject as Machine"

Dự đoán từ tiếp theo: "learning"

Kết quả này có vẻ hợp lý với bối cảnh, khi mô hình dự đoán từ tiếp theo là "learning" sau khi xuất hiện từ "Machine". Điều này có thể là một dự đoán chính xác.

**Example 2:**

Câu văn: "I am living city new"

Dự đoán từ tiếp theo: "in"

Mô hình dự đoán "in" là từ tiếp theo, có vẻ hợp lý khi câu văn đang mô tả việc sống ở một thành phố mới.

**Example 3:**

Câu văn: "I am working as a"

Dự đoán từ tiếp theo: "better"

Dự đoán là "better" có vẻ không phù hợp với ngữ cảnh, có thể là do dữ liệu huấn luyện không đủ hoặc mô hình cần được tinh chỉnh thêm.

Như vậy là em đã hoàn thành xong mô hình dự đoán từ tiếp theo trong chuỗi văn bản bằng Tensorflow

3.3./ Kết luận

Mô hình được xây dựng sử dụng mạng nơ-ron hồi quy kép (Bidirectional LSTM) cùng với lớp nhúng (Embedding). Quá trình huấn luyện được thực hiện trên tập dữ liệu đã được chuẩn bị với các chuỗi đầu vào được mã hóa và các nhãn được biểu diễn dưới dạng one-hot encoding. Mục tiêu của mô hình là dự đoán từ tiếp theo trong chuỗi văn bản.

Kết quả của quá trình huấn luyện cho thấy sự giảm thiểu hàm mất mát và tăng độ chính xác qua từng epoch, cho thấy mô hình đang học được mối quan hệ trong dữ liệu. Tuy nhiên, để đánh giá chính xác, cần kiểm tra trên tập dữ liệu kiểm thử và đánh giá các yếu tố như overfitting.

Các ví dụ dự đoán từ tiếp theo trong câu văn cho thấy mô hình có khả năng tạo ra các dự đoán hợp lý dựa trên ngữ cảnh. Điều này có thể là một bước quan trọng trong việc tạo ra mô hình tự động sinh văn bản hoặc hỗ trợ các ứng dụng xử lý ngôn ngữ tự nhiên.

Tuy nhiên, việc cải thiện mô hình có thể đòi hỏi kiểm soát và điều chỉnh hyperparameters, sử dụng tập dữ liệu lớn và đa dạng hơn, cũng như kiểm tra trên các tác vụ dự đoán dài hạn.

3.4./ Ưu điểm và nhược điểm của mô hình

**Ưu điểm:**

**Khả năng Dự Đoán Từ Tiếp Theo:** Mô hình có khả năng học được các mối quan hệ trong chuỗi văn bản và dự đoán từ tiếp theo dựa trên ngữ cảnh của chuỗi. Điều này làm cho mô hình hữu ích trong các tác vụ như tạo văn bản tự động hoặc dự đoán từ vựng tiếp theo.

**Sử Dụng Mạng Nơ-ron Hồi Quy Kép (Bidirectional LSTM):** Bidirectional LSTM cho phép mô hình học được thông tin từ cả hai hướng của chuỗi, giúp nắm bắt các mối quan hệ phức tạp và dài hạn trong dữ liệu chuỗi.

**Lớp Nhúng (Embedding):** Sử dụng lớp nhúng giúp biểu diễn từng từ trong chuỗi dưới dạng vectơ số, tăng cường khả năng của mô hình trong việc hiểu ngữ cảnh và tương tác giữa các từ.

**Sự Linh Hoạt và Tích Hợp với TensorFlow:** Mô hình được xây dựng và huấn luyện bằng TensorFlow, một thư viện mạnh mẽ trong lĩnh vực machine learning và deep learning. Điều này mang lại sự linh hoạt và tích hợp tốt với các công cụ và thư viện khác.

**Kiểm Soát và Điều Chỉnh:** Mô hình cung cấp khả năng kiểm soát và điều chỉnh các hyperparameters như learning rate, số lượng epochs, kích thước batch, giúp tối ưu hóa hiệu suất của mô hình.

**Nhược điểm:**

**Overfitting**: Có nguy cơ mô hình quá mức tinh chỉnh (overfitting) nếu không kiểm soát kích thước mô hình, sử dụng các kỹ thuật như dropout hoặc regularization.

**Dữ Liệu Huấn Luyện Hạn Chế:** Hiệu suất của mô hình phụ thuộc lớn vào chất lượng và đa dạng của dữ liệu huấn luyện. Nếu dữ liệu không đủ hoặc quá ít, mô hình có thể không học được mối quan hệ phức tạp trong dữ liệu.

**Khả Năng Dự Đoán Bị Hạn Chế**: Mô hình có thể gặp khó khăn khi đối mặt với từ vựng mới hoặc cấu trúc câu phức tạp mà nó chưa từng gặp trong quá trình huấn luyện.

**Hyperparameter Sensitivity:** Cần sự điều chỉnh kỹ lưỡng của các tham số như learning rate, số lượng epochs, và kích thước batch để đạt được hiệu suất tốt nhất. Sự nhạy cảm của mô hình với các hyperparameters có thể làm tăng độ phức tạp trong việc điều chỉnh.

**Khó Khăn Trong Việc Đánh Giá:** Đánh giá hiệu suất của mô hình dự đoán từ tiếp theo trong chuỗi văn bản có thể là một thách thức, đặc biệt là khi không có một tiêu chí rõ ràng để đánh giá độ chính xác.

**Khả Năng Dự Đoán Có Thể Bị Hạn Chế Trong Các Tình Huống Khác Nhau:** Mô hình có thể không linh hoạt đối với các loại văn bản khác nhau, ví dụ như văn bản chuyên ngành, ngôn ngữ nguyên tắc, hoặc các ngữ cảnh đặc biệt.

3.5./ Phương hướng khắc phục

Để khắc phục các nhược điểm đã nêu trên và cải thiện hiệu suất của mô hình Dự đoán từ tiếp theo trong chuỗi văn bản bằng TensorFlow

**Kiểm Soát Overfitting**:Sử dụng các kỹ thuật như dropout hoặc regularization để kiểm soát overfitting. Điều này giúp mô hình tổng quát hóa tốt hơn trên dữ liệu mới.

**Tăng Cường Dữ Liệu Huấn Luyện**:Tăng kích thước của tập dữ liệu huấn luyện hoặc sử dụng các kỹ thuật tăng cường dữ liệu để tạo thêm sự đa dạng trong dữ liệu.

**Sử Dụng Mô Hình Mạnh Mẽ Hơn**:Nếu có khả năng, xem xét việc sử dụng mô hình mạnh mẽ hơn hoặc kiến trúc mạng nơ-ron phức tạp hơn để có khả năng học được mối quan hệ phức tạp hơn.

**Fine-tuning Hyperparameters**: Tiếp tục điều chỉnh các hyperparameters như learning rate, số lượng epochs, và kích thước batch để tối ưu hóa hiệu suất. Sử dụng các phương pháp tinh chỉnh tự động nếu có thể.

**Xem Xét Sử Dụng Mô Hình Mạng Nơ-ron Hồi Quy Gốc (RNN) Hoặc Transformer:** Nếu mô hình hiện tại không đạt được hiệu suất mong muốn, xem xét việc sử dụng các kiến trúc mạng nơ-ron hồi quy gốc hoặc Transformer, đặc biệt là trong các nhiệm vụ xử lý ngôn ngữ tự nhiên.

**Dữ Liệu Đa Dạng và Đồng Nhất:** Đảm bảo rằng dữ liệu huấn luyện đa dạng và đồng nhất, bao gồm cả nhiều loại ngôn ngữ và ngữ cảnh khác nhau.

**Kiểm Tra và Đánh Giá Kỹ Lưỡng:** Sử dụng các phương pháp kiểm tra và đánh giá phù hợp để đảm bảo mô hình hoạt động tốt trên các tình huống khác nhau và không gặp vấn đề overfitting.

**Sử Dụng Embeddings Tiên Tiến**: Thử nghiệm với các biểu diễn từ tiên tiến hơn như Word2Vec, GloVe hoặc embeddings BERT để cải thiện khả năng hiểu ngữ cảnh của mô hình.

**Sử Dụng Tập Dữ Liệu Lớn Hơn**: Nếu có khả năng, sử dụng tập dữ liệu lớn hơn để cung cấp đủ dữ liệu đa dạng cho mô hình học.

**Tích Hợp Mechanisms Attention**: Nếu mô hình là RNN hoặc Transformer, tích hợp các cơ chế attention để tăng khả năng hiểu ngữ cảnh và tập trung vào các phần quan trọng của chuỗi.

3.6./ Kết luận mô hình dự đoán

Trong kết luận, mô hình Dự đoán từ tiếp theo trong chuỗi văn bản bằng TensorFlow là một công cụ mạnh mẽ có khả năng học và dự đoán từ tiếp theo dựa trên ngữ cảnh của chuỗi. Mô hình sử dụng mạng nơ-ron hồi quy kép (Bidirectional LSTM) và lớp nhúng để biểu diễn và học mối quan hệ phức tạp trong dữ liệu văn bản.

Các ưu điểm của mô hình bao gồm khả năng dự đoán từ tiếp theo chính xác, sự linh hoạt trong tích hợp với TensorFlow, và khả năng điều chỉnh các hyperparameters để tối ưu hóa hiệu suất. Tuy nhiên, cũng có nhược điểm như nguy cơ overfitting, phụ thuộc vào chất lượng dữ liệu huấn luyện, và khả năng hạn chế khi đối mặt với từ vựng mới.

Để khắc phục những nhược điểm này, việc kiểm soát overfitting, tăng cường dữ liệu huấn luyện, sử dụng mô hình mạnh mẽ hơn, và điều chỉnh các hyperparameters là quan trọng. Ngoài ra, sự đa dạng và đồng nhất của dữ liệu, tích hợp các embeddings tiên tiến, và sử dụng cơ chế attention có thể cải thiện hiệu suất của mô hình.

Cuối cùng, quá trình phát triển và tối ưu hóa mô hình là một quá trình liên tục, và việc kỳ vọng sự linh hoạt và sẵn sàng thí nghiệm là quan trọng để đạt được kết quả tốt nhất trong ứng dụng cụ thể của bạn.

TÀI LIỆU THAM KHẢO

[1] Brownlee, J. (2017). "Develop a Deep Learning Model for Text Sequences in Python." Machine Learning Mastery.

Link: <https://machinelearningmastery.com/develop-word-based-neural-language-models-python-keras/>

[2] Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). "On the properties of neural machine translation: Encoder-decoder approaches." arXiv preprint arXiv:1409.1259.

Link: <https://arxiv.org/abs/1409.1259>

[3] Colah, C. (2015). "Understanding LSTM Networks." colah's blog.

Link: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). "Attention is all you need." In Advances in neural information processing systems (pp. 5998-6008).

Link: <https://arxiv.org/abs/1706.03762>

[5] Géron, A. (2019). "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow." O'Reilly Media.

[6] Ruder, S. (2016). "An overview of gradient descent optimization algorithms." arXiv preprint arXiv:1609.04747.

Link: <https://arxiv.org/abs/1609.04747>

[7] TensorFlow Documentation.

Link: <https://www.tensorflow.org/guide>