TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN GIỮA KÌ**

**XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

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

*Người thực hiện*: **HOÀNG ĐÌNH QUÝ VŨ – 521H0517**

**LÊ HUỲNH HUYỀN TRANG - 520C0156**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

A red and blue logo

Description automatically generated

**TIỂU LUẬN GIỮA KÌ**

**XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

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

*Người thực hiện*: **HOÀNG ĐÌNH QUÝ VŨ – 521H0517**

**LÊ HUỲNH HUYỀN TRANG - 520C0156**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

LỜI CẢM ƠN

Đây là phần tác giả **tự viết** ngắn gọn, thể hiện sự biết ơn của mình đối với những người đã giúp mình hoàn thành Luận văn/Luận án. Tuyệt đối không sao chép theo mẫu những “lời cảm ơn” đã có.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án 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)*

*Hoàng Đình Quý Vũ*

*Lê Huỳnh Huyền Trang*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

MỤC LỤC

[MỤC LỤC 1](#_Toc168848587)

[CHƯƠNG 1 – Tổng Quan và Ứng Dụng Word2Vec trong Xử Lý Ngôn Ngữ Tự Nhiên 6](#_Toc168848588)

[1.1 Word Embedding 6](#_Toc168848589)

[1.1.1 Tại sao chúng ta cần Word Embedding ? 7](#_Toc168848590)

[1.1.1.1 One-hot-encoding 7](#_Toc168848591)

[1.1.1.2 TF-IDF (Term Frequency – Inverse Document Frequency) 8](#_Toc168848592)

[1.1.2 Mục tiêu 9](#_Toc168848593)

[1.1.3 Ý nghĩa 10](#_Toc168848594)

[1.2 Các hàm giải thích thêm các hàm sử dụng. 11](#_Toc168848595)

[1.2.1 Softmax 11](#_Toc168848596)

[1.2.2 Gradient Descent and Stochastic 12](#_Toc168848597)

[1.2.2 Sigmoid 13](#_Toc168848598)

[1.3 Skip-gram 14](#_Toc168848599)

[1.3.1 Cấu trúc mạng nơ-ron của Skip-Gram 16](#_Toc168848600)

[1.3.2 Training: forward propagation 18](#_Toc168848601)

[1.3.2.1 Tầng đầu vào - Input layer (x) 18](#_Toc168848602)

[1.3.2.2 **Ma trận trọng số đầu vào và đầu ra ()** 18](#_Toc168848603)

[1.3.2.3 Tầng ẩn – Hidden (projection) layer (h) 19](#_Toc168848604)

[1.3.3 Training: backward propagation 20](#_Toc168848605)

[1.3.4 Lỗi dự đoán – Prediction error () 21](#_Toc168848606)

[1.3.5 Quá trình chạy của skip-gram 22](#_Toc168848607)

[1.3.6 Negative sampling 25](#_Toc168848608)

[1.3.6.1 Cách negative sampling hoạt động 26](#_Toc168848609)

[1.3.6.2 Ví dụ 28](#_Toc168848610)

[1.4 Continuous Bag Of Words – CBOW 28](#_Toc168848611)

[1.5 Các phương pháp Word2Vec khác: Fasttext, Glove 29](#_Toc168848612)

[1.5.1 Fasttext 29](#_Toc168848613)

[1.5.2 Glove 30](#_Toc168848614)

[1.6 Thư viện Gensim 31](#_Toc168848615)

[1.7 Biểu diễn Vector của 1 câu - Sentence2Vec 32](#_Toc168848616)

[1.7.1 Bag-of-words(BoW): 32](#_Toc168848617)

[1.7.2 TF-IDF 33](#_Toc168848618)

[1.7.3 Word2Vec 34](#_Toc168848619)

[1.7.4 Pre-trained Language Models: 35](#_Toc168848620)

[CHƯƠNG 2 – PHÂN LOẠI VĂN BẢN ĐA NHÃN BẰNG NEURAL NETWORK VÀ CÁC PHƯƠNG PHÁP TRUYỀN THỐNG 36](#_Toc168848621)

[2.1 Giới thiệu 36](#_Toc168848622)

[2.1.1 Các thách thức 36](#_Toc168848623)

[2.1.2 Ứng dụng 37](#_Toc168848624)

[2.2 Feed Forward Neural Network 38](#_Toc168848625)

[2.2.1 Cấu trúc của FFNN: 39](#_Toc168848626)

[2.2.2 Quá trình hoạt động của FFNN: 39](#_Toc168848627)

[2.2.4 Lan truyền ngược (Backpropagation): 41](#_Toc168848628)

[2.2.5 Thuật toán tối ưu hóa 41](#_Toc168848629)

[2.3 RNN (Recurrent Neural Networks) 42](#_Toc168848630)

[2.3.1. Cấu trúc của RNN 43](#_Toc168848631)

[2.3.2. Lan truyền tiến (Forward Propagation) trong RNN 43](#_Toc168848632)

[2.3.3. Hàm mất mát (Loss Function) 44](#_Toc168848633)

[2.3.4. Lan truyền ngược qua thời gian (Backpropagation Through Time - BPTT) 44](#_Toc168848634)

[2.3.5. Các vấn đề và giải pháp trong RNN 45](#_Toc168848635)

[2.4 Các phương pháp truyền thống khác 45](#_Toc168848636)

[2.4.1 Chuyển đổi vấn đề 45](#_Toc168848637)

[2.4.1.1 Binary Relevance 45](#_Toc168848638)

[2.4.1.1 Classifier Chains 46](#_Toc168848639)

[2.4.1.1 Label Powerset 47](#_Toc168848640)

[TÀI LIỆU THAM KHẢO 48](#_Toc168848641)

**DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT**

**CÁC KÝ HIỆU**

*f Tần số của dòng điện và điện áp (Hz)*

*p Mật độ điện tích khối (C/m3)*

**CÁC CHỮ VIẾT TẮT**

CSTD Công suất tác dụng

MF Máy phát điện

BER Tỷ lệ bít lỗi

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 2.1: Kiến trúc FTP 1](#_Toc387689394)

**DANH MỤC BẢNG**

[Bảng 3.1 Ví dụ cho chèn bảng 1](#_Toc387689363)

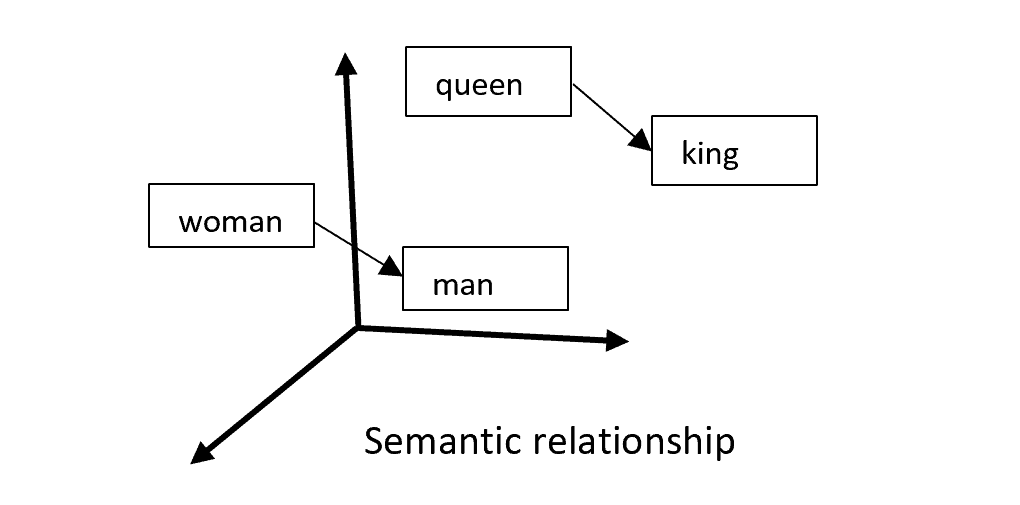
CHƯƠNG 1 – Tổng Quan và Ứng Dụng Word2Vec trong Xử Lý Ngôn Ngữ Tự Nhiên

1.1 Word Embedding

Word Embedding là một không gian vector dùng để biểu diễn dữ liệu có khả năng miêu tả được mối liên hệ, sự tương đồng về mặt ngữ nghĩa, văn cảnh(context) của dữ liệu. Không gian này bao gồm nhiều chiều và các từ trong không gian đó mà có cùng văn cảnh hoặc ngữ nghĩa sẽ có vị trí gần nhau.

Về cơ bản, thuật toán lấy một lượng lớn văn bản làm đầu vào và tạo ra một vectơ, được gọi là vectơ ngữ cảnh, làm đầu ra.

Ví dụ: “vua” và “nữ hoàng” có thể có ý nghĩa tương tự nhau, ở gần nhau trong tài liệu và có các từ liên quan như “đàn ông” hoặc “phụ nữ”.



Ví dụ như ta có hai câu : "Hôm nay ăn táo " và "Hôm nay ăn xoài ". Khi ta thực hiện Word Embedding, "táo" và "xoài" sẽ có vị trí gần nhau trong không gian chúng ta biễu diễn do chúng có vị trị giống nhau trong một câu .

1.1.1 Tại sao chúng ta cần Word Embedding ?

1.1.1.1 One-hot-encoding

Dưới đây là một cách biểu diễn chúng ta thường sử dụng trong các bài toán multi-label, multi-task là one-hot encoding. Nếu chúng ta sử dụng one-hot encoding, dữ liệu sẽ được biểu diễn dưới dạng sau đây:

|  |  |  |
| --- | --- | --- |
| Document | Index | One-hot-encoding |
| a | 1 | [1, 0, 0, ...., 0] (9999 số 0) |
| b | 2 | [0, 1, 0, ...., 0] |
| c | 3 | [0, 0, 1, ...., 0] |
| … | … | … |
| vũ | 999 | [0, 0, 0, ..., 1, 0] |
| mẹ | 1000 | [0, 0, 0, ..., 0, 1] |

Nhìn vào bảng trên, chúng ta có thể nhận thấy 3 vấn đề khi biểu diễn dữ liệu dưới dạng one-hot encoding:

1. Chi phí tính toán lớn: Khi số lượng từ trong dữ liệu tăng lên, độ dài của vector one-hot cũng tăng theo. Điều này có thể gây khó khăn trong việc tính toán và lưu trữ, đặc biệt khi dữ liệu có hàng triệu từ.
2. Mang ít giá trị thông tin: Các vector hầu như toàn số 0. Đối với dữ liệu văn bản, giá trị thông tin không nằm trong các pixel như dữ liệu hình ảnh, mà chủ yếu nằm trong vị trí tương đối giữa các từ và quan hệ ngữ nghĩa giữa chúng. Tuy nhiên, vector one-hot không thể biểu diễn điều này vì nó chỉ đánh index theo thứ tự từ điển đầu vào, không phản ánh được vị trí các từ trong một ngữ cảnh cụ thể.
3. Độ khái quát yếu: Ví dụ, ta có bốn từ "bu", "u", "mẹ", và "má" đều chỉ người mẹ. Tuy nhiên, các từ "bu" và "u" khá hiếm gặp trong tiếng Việt. Khi biểu diễn bằng one-hot encoding, mỗi từ sẽ được biểu diễn bằng một vector khác nhau, với chỉ một phần tử là 1 và phần còn lại là 0. Khi đưa vào mô hình huấn luyện, các từ "bu" và "u" sẽ bị coi là hoàn toàn khác biệt với "mẹ" và "má" do cách biểu diễn khác nhau, mặc dù cả bốn từ này đều chỉ người mẹ.

1.1.1.2 TF-IDF (Term Frequency – Inverse Document Frequency)

TF-IDF (Term Frequency – Inverse Document Frequency) là 1 kĩ thuật sử dụng trong khai phá dữ liệu văn bản. Trọng số này được sử dụng để đánh giá tầm quan trọng của một từ trong một văn bản. Giá trị cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu.

TF: Term Frequency(Tần suất xuất hiện của từ) là số lần từ xuất hiện trong văn bản. Vì các văn bản có thể có độ dài ngắn khác nhau nên một số từ có thể xuất hiện nhiều lần trong một văn bản dài hơn là một văn bản ngắn. Như vậy, term frequency thường được chia cho độ dài văn bản( tổng số từ trong một văn bản).

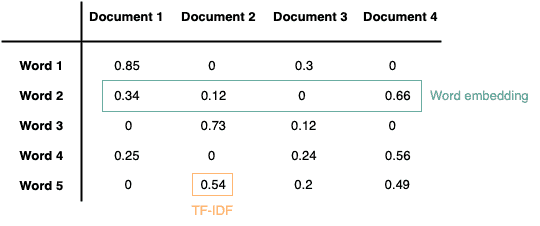
* Trong đó:
  + tf(t, d): tần suất xuất hiện của từ t trong văn bản d
  + f(t, d): Số lần xuất hiện của từ t trong văn bản d
  + max({f(w, d) : w ∈ d}): Số lần xuất hiện của từ có số lần xuất hiện nhiều nhất trong văn bản d.

IDF: Inverse Document Frequency - Tần suất nghịch của văn bản giúp đánh giá mức độ quan trọng của một từ. Khi tính toán TF, tất cả các từ được coi là có độ quan trọng như nhau. Tuy nhiên, một số từ như "is", "of" và "that" thường xuất hiện rất nhiều lần nhưng không quan trọng. Do đó, chúng ta cần giảm độ quan trọng của những từ này.

* Trong đó:
* idf(t, D): giá trị idf của từ t trong tập văn bản
* |D|: Tổng số văn bản trong tập D
* |{d ∈ D : t ∈ d}|: thể hiện số văn bản trong tập D có chứa từ t.

Công thức hoàn chỉnh là :

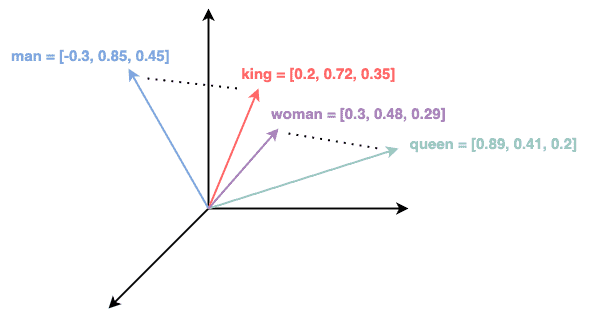
* **Những từ có giá trị TF-IDF cao là những từ xuất hiện nhiều trong văn bản này, và xuất hiện ít trong các văn bản khác. Việc này giúp lọc ra những từ phổ biến và giữ lại những từ có giá trị cao (từ khoá của văn bản đó).**



**Nhược điểm:**

* **Giống như one-hot encoding, TF-IDF không biểu diễn được ngữ nghĩa của từ. Nó chỉ dựa trên tần suất từ mà không phản ánh được các mối quan hệ ngữ nghĩa giữa các từ.**
* **TF-IDF không nắm bắt được ngữ cảnh của từ trong câu. Nó không thể phân biệt được các từ có nhiều nghĩa khác nhau dựa trên ngữ cảnh**
* **Giá trị TF-IDF của một từ phụ thuộc vào tập tài liệu cụ thể mà nó được tính toán, do đó có thể không nhất quán giữa các tập dữ liệu khác nhau.**

1.1.2 Mục tiêu



* + Mục tiêu chính là xây dựng ma trận nhúng (word embedding) giúp nắm bắt tốt mối quan hệ giữa các từ trong một vector không gian.
  + Biểu diễn từ vựng trong không gian liên tục: Word2Vec chuyển đổi các từ thành các vector số trong không gian nhiều chiều. Bằng cách này, các từ tương tự về ngữ nghĩa sẽ có các vector gần nhau trong không gian vector, giúp cho việc tính toán và so sánh giữa các từ trở nên hiệu quả hơn.
  + Ghi nhận ngữ nghĩa của từ: Các vector nhúng từ tạo ra bởi Word2Vec cần phản ánh ngữ nghĩa và ngữ cảnh của từ trong văn bản. Điều này đảm bảo rằng các từ có ý nghĩa tương tự sẽ có các vector gần nhau trong không gian vector, giúp cho các mô hình có thể hiểu được mối quan hệ ngữ nghĩa giữa các từ.
  + Cải thiện hiệu quả của các mô hình học máy: Các word embeddings tạo ra bởi Word2Vec có thể được sử dụng làm đầu vào cho các mô hình học máy khác. Việc này giúp cải thiện hiệu suất của các mô hình này, bởi vì thông tin ngữ nghĩa được biểu diễn dưới dạng các vector số, giúp cho mô hình hiểu được ngữ cảnh và mối quan hệ giữa các từ trong dữ liệu đầu vào.

1.1.3 Ý nghĩa

Word embeddings giúp giảm chiều dữ liệu so với phương pháp one-hot encoding bằng cách biểu diễn mỗi từ thành một vector có số chiều nhỏ hơn đáng kể. Đồng thời, chúng cũng cải thiện khả năng tổng quát hóa của mô hình bằng cách sử dụng ngữ cảnh để đặt các từ có ý nghĩa gần nhau trong không gian vector.

Như vậy, từ có thể được dự đoán dựa trên các từ xung quanh trong câu, như trong ví dụ "Hà Nội là ... của Việt Nam", mô hình có thể dự đoán từ còn thiếu là "thủ đô" với xác suất cao.

Điều này giúp mô hình hiểu và ánh xạ ngữ nghĩa của từ một cách chính xác hơn, tạo ra các embedding có thể tổng quát hóa tốt hơn cho các từ không xuất hiện trong tập huấn luyện.

1.2 Các hàm giải thích thêm các hàm sử dụng.

1.2.1 Softmax

Softmax chuyển đổi một vector các giá trị thành một vector các xác suất, trong đó tổng tất cả các xác suất là 1. Đây là lý do tại sao Softmax thường được sử dụng ở đầu ra của các mô hình phân loại đa lớp.

Cho 1 vector đầu vào:

Công thức Softmax:

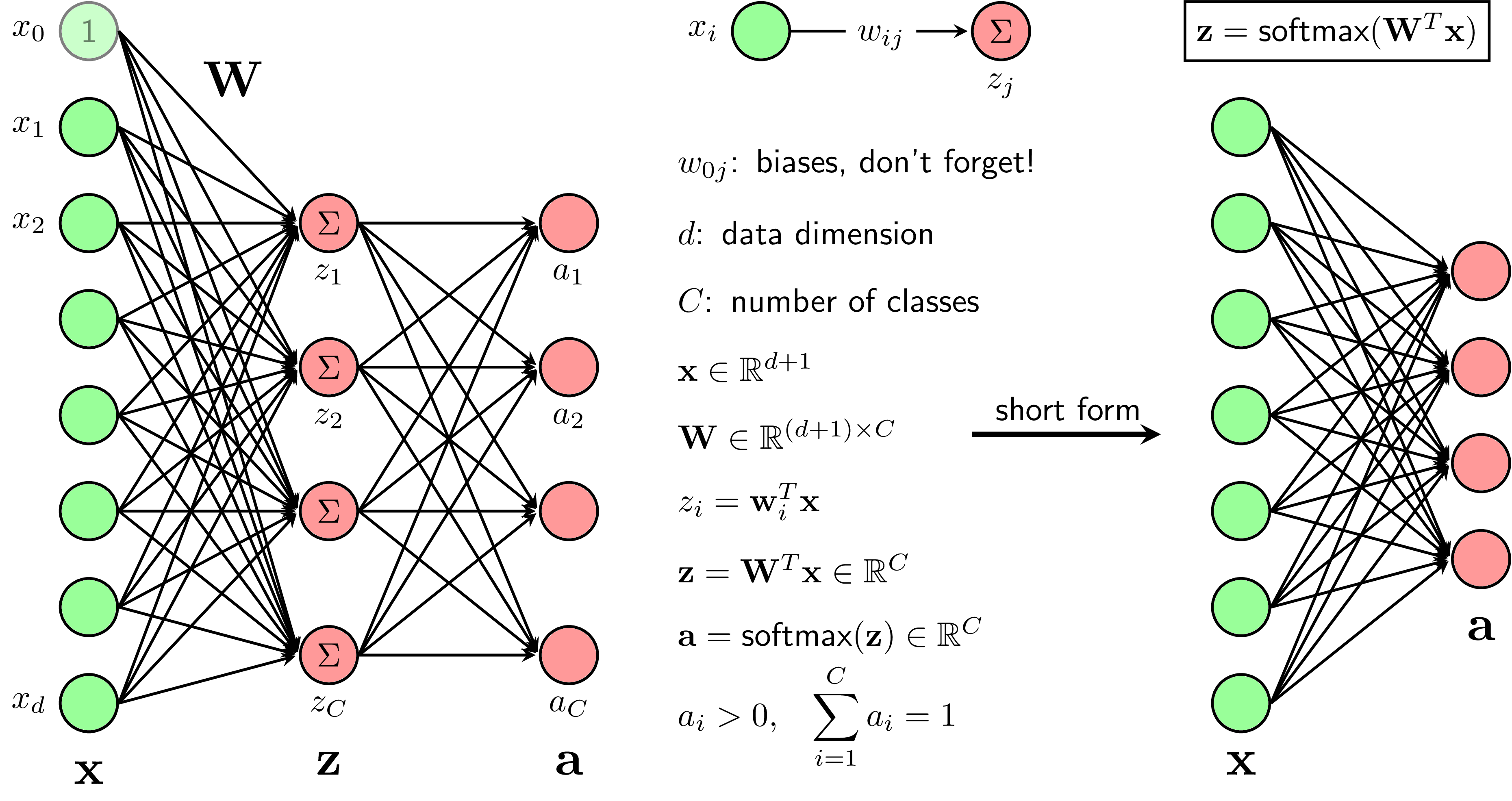
Trong đó :

* thể hiện xác suất để input đó rơi vào class i, với lớn hơn 0 và có tổng bằng 1
* , càng lớn thì xác suất dữ liệu rơi vào class i càng cao. có thể nhận giá trị âm hoặc dương.
* là một hàm mượt, có nghĩa là nó liên tục và khả vi vô hạn. Nghĩ là là 1 giá trị dương thì sẽ trở thành 1 hàm đồng biến(đơn điệu tăng) với tăng thì tăng.
* A graph of a function

  Description automatically generated

Ta có thể viết thành

* được hiểu là xác suất để một điểm dữ liệu x rơi vào class thứ i nếu biết tham số mô hình (ma trận trọng số) là W

Hình vẽ dưới đây thể hiện mạng Softmax Regression dưới dạng neural network: 

1.2.2 Gradient Descent and Stochastic

Gradient Descent cổ điển: Cập nhật tham số sau khi tính toán gradient dựa trên toàn bộ tập dữ liệu.

* Dự đoán 1 điểm khởi tạo:
* Cập nhật đến khi đạt được kết quả chấp nhận được:
* là đạo hàm của hàm mất mát tại

Stochastic Gradient Descent: Cập nhật tham số dựa trên một mẫu ngẫu nhiên (mini-batch) từ tập dữ liệu.

* Mỗi lần duyệt một lượt qua tất cả các điểm trên toàn bộ dữ liệu được gọi là một **epoch**. Với **GD thông thường** thì mỗi **epoch** ứng với **1 lần** cập nhật θ, với **SGD** thì mỗi epoch ứng với **N lần cập nhật** θ với N là số điểm dữ liệu (sau mỗi epoch, chúng ta cần shuffle (xáo trộn) thứ tự của các dữ liệu để đảm bảo tính ngẫu nhiên).
* Nhìn vào một mặt, việc cập nhật từng điểm một như thế này có thể làm giảm đi tốc độ thực hiện 1 epoch. Nhưng nhìn vào một mặt khác, SGD chỉ yêu cầu một lượng epoch rất nhỏ (thường là 10 cho lần đầu tiên, sau đó khi có dữ liệu mới thì chỉ cần chạy dưới một epoch là đã có nghiệm tốt).
* là hàm mất mát với chỉ 1 cặp điểm dữ liệu (input, label) là

1.2.2 Sigmoid

Hàm sigmoid được sử dụng cho hồi quy logistic hai lớp.Nó được sử dụng để phân loại xem một mẫu nhất định là Đúng hay Sai dựa trên xác suất được tính toán. Mẫu được phân loại là True nếu giá trị lớn hơn 0,5 và ngược lại.

A graph of a function

Description automatically generated

1.3 Skip-gram

Mô hình Skip-Gram cố gắng tối ưu hóa ma trận trọng số từ (nhúng) bằng cách dự đoán chính xác các từ ngữ cảnh(context), dựa trên một từ trung tâm(center). Nói cách khác, mô hình muốn tối đa hóa xác suất dự đoán chính xác tất cả các từ ngữ cảnh cùng một lúc, dựa trên một từ trung tâm.

Tối đa hóa xác suất dự đoán các từ ngữ cảnh dẫn đến việc tối ưu hóa ma trận trọng số (θ) thể hiện tốt nhất các từ trong không gian vectơ. θ là sự kết hợp của ma trận trọng số đầu vào và đầu ra - [, ]

Và nó được biểu dưới công thức:

Trong thực tế phổ biến là lấy log tự nhiên của hàm mục tiêu để đơn giản hóa việc lấy đạo hàm. Và công thức sẽ trở thành:

Ứng với công thức của 1 center và 1 context, Công thức softmax:

A mathematical equation with numbers and symbols

Description automatically generated

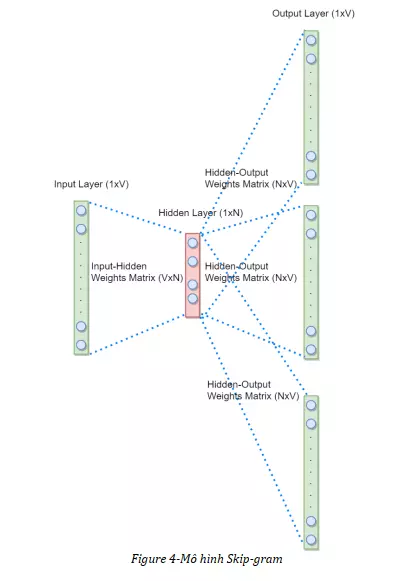
Tổng quát hóa lên với công thức sau:

A black text with black letters

Description automatically generated with medium confidence

Lưu ý:

* Sử dụng natural log để cho công thức trở nên dễ dàng hơn mà cũng sẽ không ảnh hưởng đến trọng số.
* Với
* Nó sẽ được viết lại: =



1.3.1 Cấu trúc mạng nơ-ron của Skip-Gram

Ví dụ được trích dẫn từ Game of Thrones, "The man who passes the sentence should swing the sword", của Ned Stark. Có 10 từ (T=10), và 8 từ duy nhất (V=8).

A screenshot of a computer

Description automatically generated

Một số giải thích trước khi đi vào quá trình chạy của thuật toán. Nhìn hình có thể thấy rằng:

* Chúng ta cho chiều của tầng ẩn là 3 (N = 3)
* có kích thước là 8x3 và có kích thước là 3x8.

Xác định [đầu vào | đầu ra] của mô hình, mỗi dòng dưới đây là một ô cửa sổ ngữ cảnh( C= 2 – window size = 3).

A close-up of a text

Description automatically generated

Với như hình trên, **'passes'** là từ trung tâm hiện tại, tạo thành **'who'** và **'the'** là các từ ngữ cảnh. Tương tự với **'sentence'** là từ trung tâm hiện tại, tạo thành **'the'** và **'should'** là các từ ngữ cảnh.

1.3.2 Training: forward propagation

Các ma trận nhúng từ (, ) trong Skip-Gram được tối ưu hóa thông qua việc truyền tiến và lùi. Đối với mỗi lần lặp của quá trình truyền tiến + lùi (forward + backward propagations), mô hình sẽ học cách giảm lỗi dự đoán bằng cách tối ưu hóa ma trận trọng số (θ). Đầu tiên với quá trình tiến, Truyền lan về phía trước bao gồm việc thu được phân bố xác suất của các từ (ypred) cho một từ ở giữa

1.3.2.1 Tầng đầu vào - Input layer (x)

Biến các từ các từ duy nhất, sắp xếp theo thứ tự bảng chữ cái và one-hot-encoding.

* Voca = [man, passes, sentence, should, swing, sword, the, who]
* man = [1,0,0…0] – 7 số 0
* passes = [0,1,0…0] – 7 số 0
* …
* who = [0,0,0…1] – 7 số 0

A screenshot of a computer

Description automatically generated

1.3.2.2 **Ma trận trọng số đầu vào và đầu ra ()**

Mục tiêu chính của mô hình không chỉ là dự báo các từ ngữ cảnh mà còn tạo ra các ma trận nhúng từ **()** giúp nắm bắt hiệu quả các kết nối giữa các từ trong không gian vectơ. Skip-Gram thực hiện điều này thông qua mạng lưới thần kinh, mạng này tinh chỉnh các ma trận trọng số (nhúng) bằng cách điều chỉnh chúng để giảm thiểu lỗi dự đoán ().

1.3.2.3 Tầng ẩn – Hidden (projection) layer (h)

Skip-Gram sử dụng mạng lưới thần kinh với một lớp ẩn(hidden) gọi là lớp chiếu(projection) trong ngữ cảnh xử lý ngôn ngữ tự nhiên. Lớp chiếu thực chất là một vectơ N-chiều được chiếu từ vectơ đầu vào (x) được mã hóa one-hot.

A screenshot of a math equation

Description automatically generated with medium confidence

Công thức của hidden layer:

Với việc h là phép của ma trận nhúng đầu vào với vector đầu vào.

1.3.2.3 **Lớp đầu ra softmax ()**

Trong skip-gram, họ sử dụng ký hiệu , để biểu thị xác suất có điều kiện của việc quan sát một từ ngữ cảnh với một từ trung tâm . Nó có được bằng cách sử dụng hàm softmax:

Đây là công thức chỉ ứng với 1 center với context. Nhưng thực tế thì tính xác suất này được tính V lần để thu được phân bố xác suất có điều kiện của việc quan sát từng **từ vựng duy nhất** trong kho ngữ liệu, cho trước một từ trung tâm.

A diagram of a mathematical equation

Description automatically generated

* Phép lũy thừa đảm bảo rằng các giá trị được chuyển đổi là dương và hệ số chuẩn hóa trong mẫu số đảm bảo rằng các giá trị có phạm vi [0,1).

1.3.3 Training: backward propagation

Lan truyền ngược liên quan đến việc tính toán các lỗi dự đoán và cập nhật ma trận trọng số (θ) để tối ưu hóa cách biểu diễn vector của các từ.

Đây là, hàm chi phí sử dụng SGD

Và vì được nối buổi 2 ma trận (, ). Nên ta tách ra để thuận lợi cho việc đạo hàm.

* A diagram of a mathematical equation

  Description automatically generated

Với là lỗi dự đoán (prediction error).

1.3.4 Lỗi dự đoán – Prediction error ()

A screenshot of a computer

Description automatically generated  
Sau khi ra được nó sẽ trừ với ma trận . Trong ví dụ này đưa vào 1 center và trả ra được 2 giá trị mong muốn (Vì C = 2). Tiếp tục ta đem tổng lại để tính toán độ dốc trọng số cho ma trận trọng số cập nhật.

A screenshot of a math equation

Description automatically generated

Khi ma trận trọng số được tối ưu hóa, lỗi dự đoán cho tất cả các từ trong vectơ lỗi dự đoán sẽ hội tụ về 0.

A diagram of numbers and a number of numbers

Description automatically generated with medium confidence

1.3.5 Quá trình chạy của skip-gram

**Forward propagation: tính toán từng ẩn với size = 1**

A screenshot of a computer

Description automatically generated

**Forward propagation: tính toán ra tầng đầu ra softmax**

A screenshot of a computer

Description automatically generated

**Backward propagation: tổng dựa đoán lỗi**

A screenshot of a computer

Description automatically generated

**Backward propagation: tính toán**∇Winput

A screenshot of a computer

Description automatically generated

**Backward propagation: tính toán**∇Woutput

A screenshot of a computer

Description automatically generated

**Backward propagation: cập nhật trọng số ma trận**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Chú ý: đây mới là quá trình cập forward và backward của 1 center và 2 context word. Vì sử dụng SGD, V= 8 và N = 3, nên nó sẽ cập nhật với từng trọng số đầu vào và đầu ra khi duyệt hết các từ duy nhất (V = 8). Vậy 24 lần cập nhật ứng với 1 lần epoch. Và nếu với 10 epoch thì số lần cập nhật của 2 trọng số này là 24 x 10 = 240.

1.3.6 Negative sampling

Có một vấn đề với Skip-Gram vanilla — softmax rất tốn kém về mặt tính toán, vì nó yêu cầu quét qua toàn bộ ma trận nhúng đầu ra (Woutput ) để tính phân bố xác suất của tất cả V từ, trong đó V có thể là hàng triệu hoặc hơn. Do tính toán kém hiệu quả này, softmax không được sử dụng trong hầu hết các triển khai Skip-Gram. Thay vào đó, chúng tôi sử dụng một phương pháp thay thế gọi là lấy mẫu âm với hàm sigmoid

A math equations and formulas

Description automatically generated with medium confidence

Với trước đó thì độ phức tạp O(V) thì sau khi sử dụng giải thuật negative sampling thì độ phức tạp là O(K+1), với K thuộc [5,20]

* Bài báo (Mikolov và cộng sự, 2013) nói rằng K=2 ~ 5 phù hợp với các tập dữ liệu lớn và K=5 ~ 20 đối với các tập dữ liệu nhỏ.

A screenshot of a computer

Description automatically generated

1.3.6.1 Cách negative sampling hoạt động

Mô hình sử dụng hàm sigmoid để học cách phân biệt các từ ngữ cảnh thực tế (dương) với các từ được rút ngẫu nhiên (âm) từ phân bố nhiễu

A screenshot of a computer screen

Description automatically generated

Ý tưởng là nếu mô hình có thể phân biệt giữa các cặp có khả năng (tích cực) và các cặp không chắc chắn (âm), thì các vectơ từ tốt sẽ được học. Hay nói cách khác là sẽ làm **tối đa** hóa xác suất của cặp tích cực và làm cực tiểu đối với xác suất của cặp tiêu cực.

Công thức xác suất được tính toán K+1 lần để thu được phân bố xác suất của một từ ngữ cảnh thực và mẫu âm K (tích cực + tiêu cực):

A math equation with numbers and equations

Description automatically generated

Trong đó:

* : Các từ thực sự xuất hiện trong cửa sổ ngữ cảnh của từ ở giữa.
* : Các từ được rút ra ngẫu nhiên từ phân bố tiếng ồn . Mục đích : Nó cố gắng chống lại sự mất cân bằng giữa các từ thông dụng và từ hiếm bằng cách giảm xác suất vẽ được các từ phổ biến và tăng xác suất vẽ được các từ hiếm.

1.3.6.2 Ví dụ

A screenshot of a computer

Description automatically generated

Cho đoạn văn: từ trung tâm (w) đang “drilling”, kích thước cửa sổ là 3 và số lượng mẫu âm tính (K là 5). Với K = 5, chọn ra 5 từ phủ định ngẫu nhiên: "minimized", "primary", "concerns", "led", and "page".

A diagram of numbers and symbols

Description automatically generated with medium confidence

Vào thời điểm phân phối xác suất đầu ra được mã hóa gần như một lần nóng như trong iter=4 của hình trên, ma trận trọng số θ được tối ưu hóa và các vectơ từ tốt được học.

1.4 Continuous Bag Of Words – CBOW

Về phần chạy từng bước và cách chạy của CBOW tương tự với Skip-gram. Tuy nhiên về đầu vào, đầu ra và cấu trúc có sự khác biệt. CBOW hoạt động dựa trên cách thức là nó sẽ dự đoán **xác suất của một từ** được đưa ra theo **ngữ cảnh** (một ngữ cảnh có thể gồm một hoặc nhiều từ).

A diagram of a network

Description automatically generated

1.5 Các phương pháp Word2Vec khác: Fasttext, Glove

1.5.1 Fasttext

FastText là một phương pháp học biểu diễn từ, được phát triển bởi nhóm nghiên cứu AI của Facebook (Facebook AI Research - FAIR) vào năm 2016, và được viết bằng ngôn ngữ C++. Phương pháp này có nhiều điểm tương đồng với Word2Vec, nhưng cũng có những cải tiến đáng kể giúp xử lý tốt hơn các vấn đề liên quan đến từ hiếm và các ngôn ngữ phức tạp.

Ví dụ, từ "đồng" với n-gram = 3 sẽ được tách thành các n-gram như sau: ["<đồng>", "<đồ", "đồn", "ồng", "ng>"]. Các dấu "<" và ">" được sử dụng để phân biệt n-gram với từ hoàn chỉnh, ví dụ như "<đồn>" khác với "đồn".

Sau khi huấn luyện, vector nhúng từ của "đồng" sẽ là tổng của các vector nhúng của các n-gram ["<đồng>", "<đồ", "đồn", "ồng", "ng>"], nhân với vector của từ "đồng".

Nhờ cơ chế tách từ thành các n-gram, FastText có thể tạo ra vector nhúng từ cho các từ chưa từng xuất hiện trong quá trình huấn luyện bằng cách cộng các vector n-gram có sẵn lại. Điều này giúp FastText có khả năng tổng quát hóa tốt hơn và xử lý hiệu quả các từ hiếm và từ mới trong ngôn ngữ.

1.5.2 Glove

GloVe (Global Vectors) là một thuật toán nhúng từ được phát triển bởi Đại học Stanford vào năm 2014. Ý tưởng chính của GloVe dựa trên quan sát rằng các từ có nghĩa tương tự nhau hoặc có mối quan hệ gần gũi có nhiều khả năng xuất hiện cùng nhau hơn.

A table with numbers and symbols

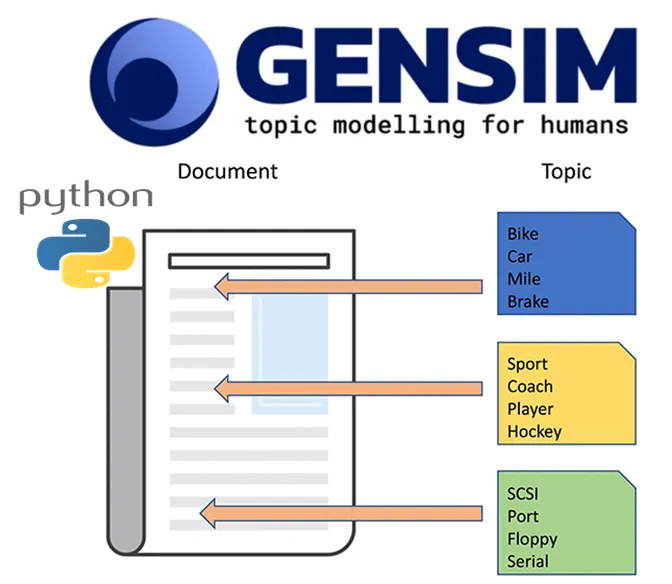
Description automatically generated

Từ bảng trên, ta có thể thấy xác suất xuất hiện gần nhau của cặp từ [“băng”, “rắn”] cao hơn 2.8 lần khi so với xác suất của cặp từ [“băng”, “khí”]. Hay xác suất của cặp từ [“băng”, “áo”] rất thấp.

Điều này phù hợp với các quan điểm như “băng là thể rắn”, “băng trái ngược với khí”, “băng không liên quan tới áo quần”.

Do đó, mục tiêu của GloVe là học ra các vector từ ngữ để tích vô hướng của chúng sẽ bằng xác suất xuất hiện đồng thời của chúng trong tập dữ liệu, với kỳ vọng sẽ phù hợp với các quan điểm thông thường.

1.6 Thư viện Gensim



Gensim là một thư viện mã nguồn mở viết bằng Python, được thiết kế đặc biệt cho việc xử lý và khai thác dữ liệu văn bản. Nó cung cấp các công cụ mạnh mẽ cho việc mô hình hóa chủ đề, phân tích văn bản và các tác vụ xử lý ngôn ngữ tự nhiên khác. Gensim nổi bật với khả năng xử lý dữ liệu lớn và hiệu suất cao.

Gensim dựa trên ý tưởng về 'khoảng cách' hoặc 'sự tương đồng' giữa các tài liệu, được xây dựng xung quanh một kho văn bản hoặc một tập hợp các tài liệu. Nó sử dụng các biểu diễn vectơ để biểu diễn các tài liệu nhằm tạo điều kiện thuận lợi cho việc đo khoảng cách giữa các tài liệu. Gensim cung cấp hai cách tiếp cận chính để so sánh tài liệu: mô hình chủ đề (LDA, LSI) và nhúng từ (word2vec, Glove).

**Các đặc điểm chính của Gensim:**

* **Mô hình hóa chủ đề (Topic Modeling)**: Gensim hỗ trợ các mô hình phổ biến như Latent Dirichlet Allocation (LDA), Latent Semantic Indexing (LSI) và Hierarchical Dirichlet Process (HDP).
* **Biểu diễn văn bản (Text Representation):**Thư viện cung cấp các phương pháp để biểu diễn văn bản dưới dạng vector, chẳng hạn như Word2Vec, Doc2Vec và FastText. Các mô hình này cho phép ánh xạ từ văn bản sang không gian vector, từ đó giúp máy tính hiểu và phân tích ngữ nghĩa của từ ngữ.
* **Xử lý dữ liệu lớn:** Gensim được thiết kế để xử lý dữ liệu lớn với hiệu suất cao, cho phép việc xử lý và phân tích dữ liệu văn bản lớn mà không cần phải tải toàn bộ dữ liệu vào bộ nhớ.
* **Dễ dàng tích hợp và mở rộng:** Thư viện có thể dễ dàng tích hợp với các công cụ và thư viện khác như NumPy, SciPy, và scikit-learn, cho phép người dùng mở rộng chức năng và tùy chỉnh theo nhu cầu.

1.7 Biểu diễn Vector của 1 câu - Sentence2Vec

Biểu diễn vector của câu (Sentence2Vec) là một kỹ thuật trong xử lý ngôn ngữ tự nhiên (NLP) để ánh xạ một câu hoặc một đoạn văn thành một vector trong không gian nhiều chiều. Vector này có thể được sử dụng cho nhiều ứng dụng như phân loại văn bản, tìm kiếm thông tin, và phân tích ngữ nghĩa. Dưới đây là các phương pháp phổ biến để biểu diễn vector của câu và ứng dụng chúng trong phân loại câu hoặc đo độ tương tự giữa các câu.

1.7.1 Bag-of-words(BoW):

Bag-of-words (BoW) là một cách biểu diễn văn bản đơn giản và phổ biến trong xử lý ngôn ngữ tự nhiên (NLP). BoW coi một văn bản như một "túi" chứa các từ, bỏ qua trật tự xuất hiện của các từ trong văn bản.

* **Cách thức hoạt động:**

Để tạo một biểu diễn BoW cho một văn bản, ta thực hiện các bước sau:

1. **Tokenization:** Chia văn bản thành các từ riêng lẻ (token).
2. **Vocabulary creation:** Tạo một danh sách các từ duy nhất (vocabulary) xuất hiện trong văn bản hoặc trong tập dữ liệu.
3. **Feature vector creation:** Tạo một vector có kích thước bằng số lượng từ trong vocabulary. Mỗi phần tử trong vector tương ứng với một từ trong vocabulary, giá trị của phần tử là số lần xuất hiện của từ đó trong văn bản.

**Ví dụ:**

Văn bản: "Tôi yêu thích học máy."

Vocabulary: ["Tôi", "yêu", "thích", "học", "máy"]

Feature vector: [1, 1, 1, 1, 1]

* **Ưu điểm:**
* Dễ dàng thực hiện.
* Không cần huấn luyện mô hình.
* Có thể kết hợp với các kỹ thuật khác như TF-IDF để tăng hiệu quả.
* **Nhược điểm:**
* Bỏ qua trật tự từ trong câu, không thể nắm bắt ngữ nghĩa của câu.
* Không thể xử lý các từ đồng nghĩa hoặc đa nghĩa.
* Kích thước vector có thể lớn, gây tốn kém về mặt tính toán.

1.7.2 TF-IDF

TF-IDF là viết tắt của ***Term Frequency-Inverse Document Frequency***, là một kỹ thuật được sử dụng để đánh giá mức độ quan trọng của một từ trong một văn bản.

**Cách thức hoạt động:**

* TF-IDF được tính bằng cách nhân hai yếu tố:
* **Term Frequency (TF):** Tần suất xuất hiện của một từ trong văn bản.
* **Inverse Document Frequency (IDF):** Độ hiếm của một từ trong một tập dữ liệu.

**Công thức tính TF-IDF:**

TF-IDF(t, d) = TF(t, d) \* IDF(t)

* **Ưu điểm:**
* TF-IDF có thể đánh giá mức độ quan trọng của một từ trong một văn bản, giúp phân biệt các từ quan trọng với các từ không quan trọng.
* TF-IDF có thể xử lý các từ đồng nghĩa hoặc đa nghĩa.
* TF-IDF có thể được sử dụng để so sánh các văn bản khác nhau.
* **Nhược điểm:**
* TF-IDF có thể bị ảnh hưởng bởi độ dài của văn bản.
* TF-IDF có thể bị ảnh hưởng bởi các từ dừng (stop words).

1.7.3 Word2Vec

Word2Vec là một mô hình ngôn ngữ được phát triển bởi Google vào năm 2013. Nó có khả năng học và biểu diễn các từ dưới dạng vector, giúp cho việc xử lý ngôn ngữ tự nhiên (NLP) trở nên hiệu quả hơn.

**Nguyên lý hoạt động:**

* Word2Vec sử dụng hai kiến trúc mạng nơ-ron chính:
* **CBOW (Continuous Bag-of-Words):** Dự đoán từ trung tâm dựa trên các từ xung quanh.
* **Skip-gram:** Dự đoán các từ xung quanh dựa trên từ trung tâm.

Cả hai kiến trúc này đều sử dụng thuật toán ***backpropagation*** để học các vector biểu diễn từ. Sau khi được huấn luyện, mỗi từ sẽ được biểu diễn bởi một vector có kích thước cố định, chứa thông tin ngữ nghĩa của từ đó.

**Lợi ích của Word2Vec:**

* Hiệu quả: Word2Vec có thể học được các vector biểu diễn từ từ một lượng dữ liệu lớn một cách hiệu quả.
* Đa dạng: Word2Vec có thể được sử dụng cho nhiều nhiệm vụ NLP khác nhau, chẳng hạn như phân loại văn bản, tóm tắt văn bản, dịch máy, v.v.
* Độ chính xác cao: Word2Vec có khả năng học được các vector biểu diễn từ có độ chính xác cao, giúp cho các ứng dụng NLP đạt hiệu quả tốt hơn.

**Ví dụ:**

Giả sử chúng ta có hai câu sau:

* Câu 1: "Tôi yêu thích học máy."
* Câu 2: "Tôi đang học tiếng Anh."

Sử dụng Word2Vec, bạn có thể học được các vector biểu diễn từ cho các từ trong hai câu này. Sau đó, bạn có thể sử dụng các vector này để tính toán độ tương đồng giữa hai câu.

1.7.4 Pre-trained Language Models:

**ELMo (Embeddings from Language Models:**  Sử dụng các mô hình mạng nơ-ron hồi tiếp (RNN) để tạo ra các vector biểu diễn từ dựa trên ngữ cảnh. Các vector này được kết hợp lại để tạo thành vector biểu diễn câu.

**BERT (Bidirectional Encoder Representations from Transformers):** Sử dụng kiến trúc Transformer để tạo ra các vector biểu diễn từ dựa trên ngữ cảnh. Có thể sử dụng vector của token [CLS] ở đầu câu để biểu diễn toàn bộ câu.

**GPT (Generative Pre-trained Transformer):** Tương tự như BERT, nhưng tập trung vào việc sinh ngôn ngữ. Các vector biểu diễn từ được tạo ra có thể kết hợp để tạo thành vector của câu

CHƯƠNG 2 – PHÂN LOẠI VĂN BẢN ĐA NHÃN BẰNG NEURAL NETWORK VÀ CÁC PHƯƠNG PHÁP TRUYỀN THỐNG

* 1. Giới thiệu

Trong phân loại nhiều lớp, mỗi phiên bản (ví dụ: tài liệu, hình ảnh, điểm dữ liệu) thuộc về chính xác một lớp trong một tập hợp các lớp loại trừ lẫn nhau. Điều này có nghĩa là đối với một thể hiện nhất định, mô hình phải chọn một và chỉ một lớp từ nhiều lớp có thể.

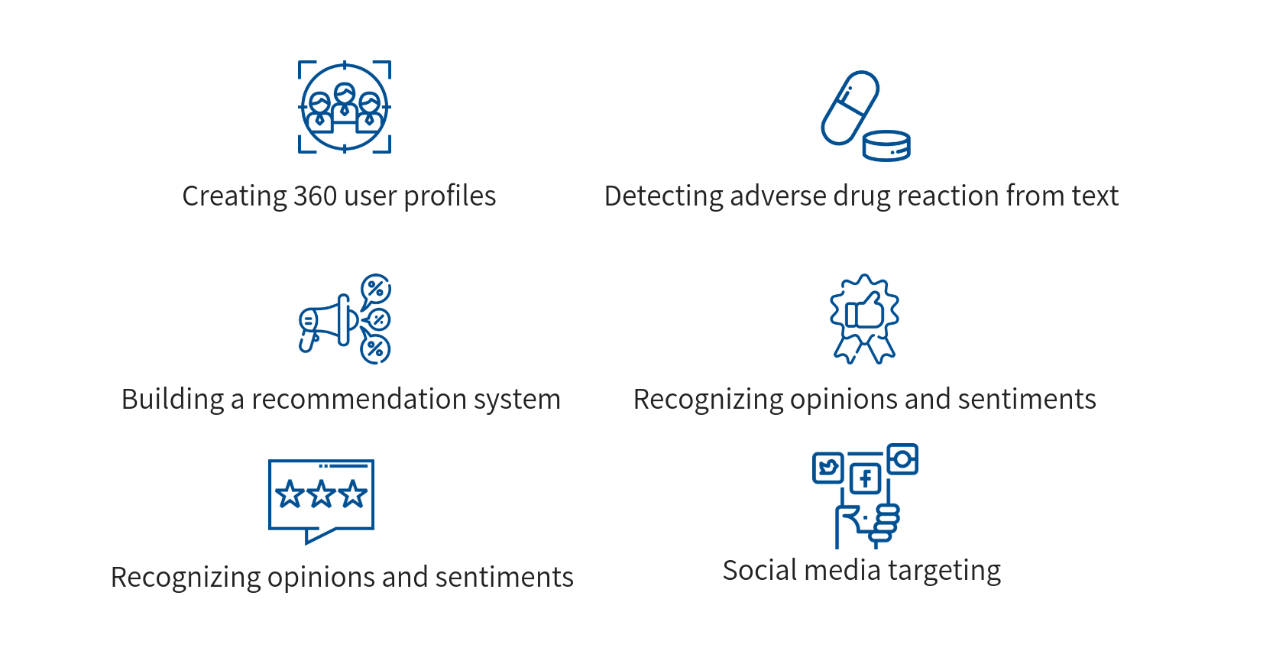
Ví dụ:Phân loại các loại trái cây: táo, chuối, cam,… Mỗi loại trái cây thuộc một và chỉ một loại.

Bài toán phân loại văn bản nhiều nhãn: Đây là bài toán trong lĩnh vực phân loại văn bản trong đó một tài liệu đầu vào có thể thuộc nhiều nhãn. Ví dụ: một tờ báo trực tuyến có thể được phân loại là thông tin chính trị, xã hội hoặc cả hai.

A screenshot of a computer screen

Description automatically generated

* + 1. Các thách thức
* Sự mất cân bằng nhãn: Một số nhãn có thể xuất hiện nhiều hơn so với các nhãn khác, dẫn đến vấn đề mất cân bằng dữ liệu. Các nhãn hiếm có thể khó được dự đoán chính xác hơn.
* Kích thước không gian nhãn: Khi số lượng nhãn tăng lên, không gian nhãn trở nên rất lớn, làm tăng độ phức tạp của bài toán. Hay là kích thức văn bản bản cũng khác nhau.
* Đánh Giá Hiệu Suất Khó Khăn: Đánh giá mô hình đa nhãn phức tạp hơn so với đơn nhãn. Các chỉ số đánh giá như độ chính xác, độ thu hồi và F1-score cần phải được tính toán cho từng nhãn và kết hợp lại để đánh giá tổng thể.
  + - Chỉ số đa chiều: Cần phải xem xét các chỉ số đa chiều như macro/micro-averaging để đánh giá hiệu suất tổng thể của mô hình.
    - Còn nhiều cái khác như : Hamming-Loss, Jaccard similarity coefficient, AUC
    1. Ứng dụng



Kỹ thuật học máy đa nhãn mở ra nhiều ứng dụng tiềm năng trong nhiều lĩnh vực, cho phép ta đạt được những lợi ích sau:

* Tiếp thị, bán hàng: Tạo hồ sơ khách hàng chi tiết, ví dụ: khách hàng vừa là người dùng Apple vừa là người hâm mộ thể thao.
* Hệ thống khuyến nghị: Đề xuất sản phẩm, dịch vụ phù hợp cho từng khách hàng trong các chương trình khách hàng thân thiết hoặc cửa hàng thương mại điện tử.
* Nhắm mục tiêu truyền thông xã hội: Quảng cáo, tiếp thị nội dung, bán hàng trên mạng xã hội cho đối tượng cụ thể.
* Ngành dược phẩm: Phát hiện phản ứng có hại của thuốc từ văn bản, phân loại tác dụng phụ từ dữ liệu trên Twitter, mô tả phản ứng với thuốc.
* Nghiên cứu thị trường: Phân tích tình cảm, đánh giá phản hồi của khách hàng về sản phẩm.
  1. Feed Forward Neural Network

Feed Forward Neural Network (FFNN) là một loại mạng neural cơ bản và phổ biến trong học sâu. Đây là loại mạng neural đơn giản nhất, trong đó dữ liệu di chuyển theo một hướng từ đầu vào qua các lớp ẩn đến đầu ra mà không có chu kỳ (loop).

A diagram of a machine

Description automatically generated

2.2.1 Cấu trúc của FFNN:

FFNN bao gồm ba loại lớp chính:

* **Lớp đầu vào (Input Layer):** Nhận dữ liệu đầu vào. Số lượng nút trong lớp này bằng với số lượng tính năng trong dữ liệu đầu vào.
* **Lớp ẩn (Hidden Layers):** Các lớp giữa đầu vào và đầu ra. Chúng thực hiện các phép biến đổi và tính toán phức tạp. Một FFNN có thể có một hoặc nhiều lớp ẩn.
* **Lớp đầu ra (Output Layer):** Cung cấp đầu ra của mạng. Số lượng nút trong lớp này tùy thuộc vào bài toán cụ thể (ví dụ: một nút cho bài toán hồi quy, nhiều nút cho bài toán phân loại đa lớp).

2.2.2 Quá trình hoạt động của FFNN:

**Lan truyền tiến (Forward Propagation):** Dữ liệu đầu vào di chuyển qua các lớp của mạng, mỗi lớp áp dụng một phép biến đổi tuyến tính và sau đó là một hàm kích hoạt (activation function). Kết quả của mỗi lớp trở thành đầu vào cho lớp tiếp theo.

A math equations on a white background

Description automatically generated

* Trong đó:
* 𝑥 là vector đầu vào.
* 𝑊 là ma trận trọng số.
* 𝑏 là vector hệ số chặn (bias).
* 𝜎 là hàm kích hoạt.

**Hàm kích hoạt (Activation Function):** Áp dụng một hàm phi tuyến để giới thiệu tính phi tuyến vào mô hình, giúp mạng học được các mối quan hệ phi tuyến trong dữ liệu.

Các hàm kích hoạt phổ biến:

* Sigmoid: ​
* Tanh: tanh(z) =
* ReLU: ReLU(z)=max(0,z)

**2.2.3 Hàm mất mát (Loss Function):**

Hàm mất mát đo lường sự khác biệt giữa giá trị dự đoán của mạng và giá trị thực tế. Mục tiêu của việc huấn luyện là tối thiểu hóa hàm mất mát.

Một số hàm mất mát phổ biến:

* Mean Squared Error (MSE) cho bài toán hồi quy.
* Cross-Entropy Loss cho bài toán phân loại.

2.2.4 Lan truyền ngược (Backpropagation):

Lan truyền ngược là quá trình điều chỉnh trọng số của mạng dựa trên lỗi của dự đoán. Nó bao gồm hai bước chính:

* **Lan truyền ngược gradient:** Tính gradient của hàm mất mát đối với từng trọng số bằng cách sử dụng quy tắc chuỗi. Gradient này cho biết mức độ ảnh hưởng của từng trọng số đến lỗi.
* **Cập nhật trọng số:** Điều chỉnh trọng số bằng cách sử dụng gradient đã tính toán. Quá trình này thường sử dụng một thuật toán tối ưu hóa như Gradient Descent.
* Quy tắc cập nhật trọng số:

A black and white math symbols

Description automatically generated with medium confidence

* Trong đó:
* η là tốc độ học (learning rate).

* ​ là gradient của hàm mất mát đối với trọng số.

2.2.5 Thuật toán tối ưu hóa

Các thuật toán tối ưu hóa được sử dụng để cập nhật trọng số nhằm giảm thiểu hàm mất mát. Các thuật toán phổ biến bao gồm:

* **Gradient Descent**: Cập nhật trọng số bằng cách di chuyển ngược theo gradient của hàm mất mát.
* **Stochastic Gradient Descent (SGD):** Phiên bản ngẫu nhiên của Gradient Descent, cập nhật trọng số dựa trên một mẫu ngẫu nhiên từ tập dữ liệu.
* **Adam**: Một thuật toán tối ưu hóa phổ biến kết hợp các ưu điểm của AdaGrad và RMSProp.

**2.2.6 Quá trình huấn luyện:**

Quá trình huấn luyện FFNN bao gồm các bước:

* **Khởi tạo trọng số**: Thường khởi tạo ngẫu nhiên các trọng số.
* **Lan truyền tiến**: Tính toán đầu ra của mạng cho dữ liệu đầu vào.
* **Tính toán lỗi**: Sử dụng hàm mất mát để đo lường lỗi của dự đoán.
* **Lan truyền ngược**: Tính toán gradient của lỗi đối với các trọng số và cập nhật trọng số.
* **Lặp lại**: Lặp lại quá trình trên cho đến khi mạng hội tụ hoặc đạt được số epoch quy định.

**2.2.7 Đánh giá và kiểm thử:**

Sau khi huấn luyện, mạng cần được đánh giá trên tập dữ liệu kiểm thử để kiểm tra hiệu suất và khả năng tổng quát hóa. Các chỉ số đánh giá phổ biến bao gồm độ chính xác, F1-score, MSE, và AUC-ROC, tùy thuộc vào bài toán cụ thể.

* 1. RNN (Recurrent Neural Networks)

Recurrent Neural Networks (RNN) là một loại mạng neural có kiến trúc cho phép xử lý dữ liệu tuần tự và duy trì thông tin qua các bước thời gian. Khả năng này làm cho RNN trở nên hữu ích trong các ứng dụng liên quan đến dữ liệu tuần tự như xử lý ngôn ngữ tự nhiên, phân tích chuỗi thời gian, và nhận dạng giọng nói.

A diagram of a machine

Description automatically generated

2.3.1. Cấu trúc của RNN

RNN có cấu trúc đặc biệt với các kết nối tuần hoàn (recurrent connections), cho phép thông tin được truyền qua các bước thời gian. Mỗi nút trong mạng không chỉ nhận đầu vào từ các bước thời gian trước mà còn từ trạng thái ẩn (hidden state) của chính nó ở bước thời gian trước đó.

* **Lớp đầu vào (Input Layer)**: Nhận dữ liệu tuần tự.
* **Lớp ẩn (Hidden Layer)**: Lưu giữ trạng thái ẩn, cập nhật qua mỗi bước thời gian.
* **Lớp đầu ra (Output Layer)**: Cung cấp đầu ra của mạng.

2.3.2. Lan truyền tiến (Forward Propagation) trong RNN

Ở mỗi bước thời gian t, RNN nhận đầu vào xt​ và trạng thái ẩn trước đó ht−1. Trạng thái ẩn hiện tại ht​ được tính toán như sau:

A black text on a white background

Description automatically generated

Trong đó:

* Whh là ma trận trọng số cho trạng thái ẩn.
* Wxh là ma trận trọng số cho đầu vào.
* bh là hệ số chặn (bias).
* σ là hàm kích hoạt (thường dùng hàm tanh hoặc ReLU).

Đầu ra yt​ được tính như sau:

A black letter on a white background

Description automatically generated

Trong đó:

* Why là ma trận trọng số cho đầu ra.
* by​ là hệ số chặn cho đầu ra.
* ϕ là hàm kích hoạt đầu ra (thường dùng hàm softmax cho phân loại).

2.3.3. Hàm mất mát (Loss Function)

Hàm mất mát trong RNN thường được tính toán dựa trên toàn bộ chuỗi đầu ra so với chuỗi mục tiêu. Đối với các bài toán phân loại, hàm mất mát phổ biến là Cross-Entropy Loss.

2.3.4. Lan truyền ngược qua thời gian (Backpropagation Through Time - BPTT)

Lan truyền ngược qua thời gian là phương pháp mở rộng của lan truyền ngược (Backpropagation) để áp dụng cho RNN. BPTT tính toán gradient của hàm mất mát đối với các trọng số của mạng qua tất cả các bước thời gian:

1. **Lan truyền tiến**: Tính toán trạng thái ẩn và đầu ra cho mỗi bước thời gian.
2. **Tính toán lỗi**: Đo lường lỗi của đầu ra so với giá trị thực.
3. **Lan truyền ngược**: Tính toán gradient của lỗi đối với từng trạng thái ẩn và trọng số, sau đó cập nhật trọng số bằng cách sử dụng một thuật toán tối ưu hóa (ví dụ: Gradient Descent).

2.3.5. Các vấn đề và giải pháp trong RNN

**Vanishing Gradient**: Gradient có thể trở nên rất nhỏ qua các bước thời gian dài, làm cho mạng khó học các phụ thuộc dài hạn.

**Exploding Gradient**: Gradient có thể trở nên rất lớn, làm cho việc cập nhật trọng số không ổn định.

**Giải pháp**:

* **LSTM (Long Short-Term Memory)**: Một loại RNN với các cổng đặc biệt để kiểm soát luồng thông tin và duy trì thông tin dài hạn.
* **GRU (Gated Recurrent Unit)**: Một biến thể đơn giản hơn của LSTM với hiệu suất tương tự.
* **Gradient Clipping**: Kỹ thuật để giới hạn giá trị của gradient nhằm tránh exploding gradient.

2.4 Các phương pháp truyền thống khác

2.4.1 Chuyển đổi vấn đề

Với phương pháp này chúng ta cố gắng chuyển đổi multi-label problem về single-label problem. Phương pháp này có thể thực hiện qua một số cách sau:

* Binary Relevance
* Classifier Chains
* Label Powerset

2.4.1.1 Binary Relevance

Trong binary relevance, chúng ta sẽ chia bài toán thành 3 bài toán phân loại theo từng lớp đơn (do có 3 lớp ở đây). Từ đó, chúng ta có thể dễ dàng xử lý từng bài toán riêng lẻ và sau đó kết hợp kết quả cuối cùng lại với nhau. Và sử dụng Thư viện Scikit-multilearn

A screenshot of a computer

Description automatically generated

2.4.1.1 Classifier Chains

Chuỗi phân loại (CC): Sắp xếp các nhãn theo một thứ tự cụ thể và mỗi nhãn sẽ được dự đoán dựa trên các nhãn trước đó trong chuỗi.

* Dữ liệu gốc:

A grid of numbers and letters

Description automatically generated

Trong phân loại chuỗi, chúng tôi cũng có 3 bảng toán nhãn đơn tương ứng với 3 lớp. Với:

* Phân loại 1: Đặc điểm: X —- mục tiêu: Y1
* Phân loại 2: Đặc điểm: X, Y1 —- mục tiêu: Y2
* Phân loại 2: Đặc điểm: X, Y1, Y2 —- mục tiêu: Y3

A screenshot of a computer

Description automatically generated

2.4.1.1 Label Powerset

Label Powerset (LP): Xử lý mọi tổ hợp nhãn duy nhất dưới dạng một lớp duy nhất, chuyển vấn đề thành vấn đề nhiều lớp.

A screenshot of a computer

Description automatically generated

* + Như thể hiện trong hình trên, các điểm dữ liệu x2 và x3 được coi là có cùng nhãn,
  + x4 và x6 được coi là có cùng nhãn. x1 có nhãn riêng, x5 có nhãn riêng.
  + Tại thời điểm này, chúng tôi có tổng cộng 4 lớp mới cho mô hình mới. Mô hình này phục vụ cho các vấn đề phân loại nhiều lớp. Người ta nhận thấy rằng bài toán phân loại đa nhãn hiện đã được chuyển thành bài toán phân loại nhiều lớp.

TÀI LIỆU THAM KHẢO

<https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>

<https://machinelearningcoban.com/2017/02/17/softmax/#-bat-dau-voi-du-lieu-nho>

<https://github.com/sonlam1102/job-prediction-multilabel-vietnamese>

https://aegis4048.github.io/demystifying\_neural\_network\_in\_skip\_gram\_language\_modeling

https://aegis4048.github.io/optimize\_computational\_efficiency\_of\_skip-gram\_with\_negative\_sampling

<https://machinelearningcoban.com/2017/01/27/logisticregression/#mo-hinh-logistic-regression>

<https://machinelearningcoban.com/2017/02/17/softmax/>

Link data:  
https://drive.google.com/drive/folders/1DD9LJ8JZhjoI8ltPgz-l0-BkiTcRQIJm?usp=sharing

**PHỤ LỤC**

Phần này bao gồm những nội dung cần thiết nhằm minh họa hoặc hỗ trợ cho nội dung luận văn như số liệu, biểu mẫu, tranh ảnh. . . . nếu sử dụng những câu trả lời cho một *bảng câu hỏi thì bảng câu hỏi mẫu này phải được đưa vào phần Phụ lục ở dạng nguyên bản* đã dùng để điều tra, thăm dò ý kiến; **không được tóm tắt hoặc sửa đổi**. Các tính toán mẫu trình bày tóm tắt trong các biểu mẫu cũng cần nêu trong Phụ lục của luận văn. Phụ lục không được dày hơn phần chính của luận văn