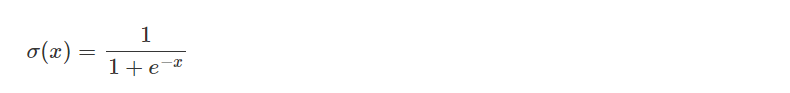
**1, So sánh Glove và Word2vec**

|  |  |  |
| --- | --- | --- |
|  | Glove | Word2vec |
| Giống nhau | * Đều cho phép convert từ một từ sang dạng vector (thường gọi là embedding) * Mối quan hệ giữa các từ được tính toán dựa vào khoảng cách giữa các từ. Khoảng cách giữa các vector từ tương tự với sự tương đồng giữa các từ với nhau * Các từ tương tự nhau về mặt ngữ nghĩa cũng sẽ có khoảng cách cosine gần nhau. * Có thể sử dụng các toán từ số học với các embedding words | |
| Khác nhau | * Glove dựa vào ý tưởng : tính số lần đồng xuất hiện của các từ trên toàn cục. * Cách huấn luyện mô hình Glove: dựa trên kĩ thuật matrix factorization (ma trận thừa số hóa) trên ma trận các từ ngữ cảnh. Ban đầu nó thiết lập một ma trận (từ x ngữ cảnh) thông tin đồng xuất hiện, cụ thể với mỗi từ (dòng), chúng ta phải đếm mức độ thường xuyên xuất hiện của từ này trong các ngữ cảnh (cột) trong một kho lớn các văn bản. Số lượng các contexts là rất lớn nên do đó, kết hợp các ngữ cảnh là việc cần thiết. * Sau đó, thừa số hóa ma trận này để giảm số chiều ma trận, mỗi dòng lúc này sẽ là một vector đại diện cho các từ tương ứng. Nhìn chung, nó được tạo ra bằng việc tối thiểu hóa “reconstruction loss”. Mất mát này cố gắng tìm ma trận với số chiều ít hơn đại diện cho hầu hết các phương sai trong dữ liệu của ma trận với số chiều lớn hơn. | * Word2vec dựa trên 1 ý tưởng ngược lại: tính số lần đồng xuất hiện của các từ trong ngữ cảnh văn bản (các từ hàng xóm) * Cách huấn luyện mô hình: Word2vec là 1 mạng nơ ron tiến để tìm ra các word embeddings: có hai mô hình thường được sử dụng nhất là : skip-gram và CBOW * Mô hình Skip-gram nhận đầu vào là mỗi từ trong kho văn bản, các input được đưa đến tầng ẩn (embedding layer) và từ đó chúng dự đoán các từ ngữ cảnh (context words). * Mô hình CBOW: nhận đầu vào là các từ ngữ cảnh (context words), chúng được gửi đến tầng ẩn và từ đó dự đoán ra từ gốc. * Sau khi huấn luyện ở cả hai mô hình trên, embedding của một từ cụ thể có được bằng việc đưa một từ như đầu vào và lấy giá trị của tầng ẩn chính là giá trị của vector embedding. |

**2, Các hàm kích hoạt**

**2.1: Sigmoid**

* Công thức:



* Hàm Sigmoid nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (0;1)
* Ưu điểm:
* Có đạo hàm đẹp
* Nhược điểm:
* **Hàm Sigmoid bão hào và triệt tiêu gradient:**  Một nhược điểm dễ nhận thấy là khi đầu vào có trị tuyệt đối lớn (rất âm hoặc rất dương), gradient của hàm số này sẽ rất gần với 0. Điều này đồng nghĩa với việc các hệ số tương ứng với unit đang xét sẽ gần như không được cập nhật (còn được gọi là vanishing gradient).
* **Hàm Sigmoid không có trung tâm là 0 gây khó khăn cho việc hội tụ.**

**2.2: Tanh**

- Công thức:



* Hàm **tanh** nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (-1; 1). Cũng như Sigmoid, hàm Tanh bị bão hoà ở 2 đầu (gradient thay đổi rất ít ở 2 đầu). Tuy nhiên hàm Tanh lại đối xứng qua 0 nên khắc phục được một nhược điểm của Sigmoid

**2.3: ReLU**

- Công thức:



* ReLU lọc các giá trị <0.
* Ưu điểm:
* Tốc độ hội tụ nhanh hơn hẳn. ReLU có tốc độ hội tụ nhanh gấp 6 lần Tanh. Điều này có thể do ReLU không bị bão hoà ở 2 đầu như Sigmoid và Tanh.
* Tính toán nhanh hơn. Tanh và Sigmoid sử dụng hàm exp và công thức phức tạp hơn ReLU rất nhiều do vậy sẽ tốn nhiều chi phí hơn để tính toán.
* Nhược điểm:
* Với các node có giá trị nhỏ hơn 0, qua ReLU activation sẽ thành 0, hiện tượng đấy gọi là “Dying ReLU“. Nếu các node bị chuyển thành 0 thì sẽ không có ý nghĩa với bước linear activation ở lớp tiếp theo và các hệ số tương ứng từ node đấy cũng không được cập nhật với gradient descent.
* Khi learning rate lớn, các trọng số (weights) có thể thay đổi theo cách làm tất cả neuron dừng việc cập nhật.

**2.4: Leaky ReLU**

- Công thức:

Text

Description automatically generated with medium confidence 

* Leaky ReLU là biến thể của ReLU. Thay vì trả về 0 với các giá trị nhỏ hơn 0 thì Leakey ReLu tạo ra 1 đường xiên với độ dốc nhỏ.
* Ưu điểm so với ReLU: hiệu quả hơn trong việc loại bỏ “dying ReLU” đã nêu ở trên.

**3, Áp dụng CNN trong việc phân tích quan điểm bộ dữ liệu IMDB**

* Sử dụng mã nguồn đã cho của TA có điều chỉnh.
* Sỗ mẫu sử dụng được lấy ngẫu nhiên như trên, số mẫu cụ thể là 12500 mẫu.

Text

Description automatically generated with medium confidence

* Áp dung phương pháp CNN cho bộ dữ liệu trên. Với kiến trúc cụ thể:

Text

Description automatically generated

* Sử dụng kiến trúc tuần tự với các layer lần lượt là:
* Layer 1: Sử dụng thư viện Embedding của keras chuyển các câu thành dạng vector (có vai trò tương tự như Word2vec hay Glove …)
* Layer 2: Convolution 1D layer
* Layer 3: Max-over time pooling 1D
* Layer 4: Flatten: chuyển câu sang dạng vector
* Layer 5: Dense: đóng vai trò như tầng fully connected layer trong kiến trúc cơ bản của CNN
* Layer 6: CNN
* Kết quả cụ thể: Không thực hiện được việc load toàn bộ tập dữ liệu negative của tập test.

**4, Link bài làm**

* Link google colab: <https://colab.research.google.com/drive/1pTQkoVINFyWjWT4QTYhuEzy7SdcDfY2B#scrollTo=xPqgV9lCUYdK>
* Link driver chứa các file code:

https://drive.google.com/drive/folders/1X9zr88vDDM5Y9WkjZRtQcltzFtIpi9Mw?usp=sharing