TRƯỜNG ĐẠI HỌC THỦY LỢI

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

Description automatically generated with low confidence

**BÀI TẬP LỚN**

HỌC PHẦN: XỬ LÝ NGÔN NGỮ TỰ NHIÊN

**ĐỀ TÀI:** Nhận diện thực thể bằng các phương pháp

BERT và BiLSTM

Giảng viên hướng dẫn: **GV. Nguyễn Thị Kim Ngân**

Sinh viên/nhóm sinh viên thực hiện:

1. Phạm Ngọc Ánh - 2151264645
2. Đỗ Thị Nguyệt Ánh - 2151260833
3. Nguyễn Ngọc Bách - 2151260830
4. Nguyễn Tuấn Dũng – 2151260829

**Hà Nội, năm 2024**

**LỜI NÓI ĐẦU**

Trong thời đại số hóa ngày nay, thông tin và dữ liệu văn bản không ngừng bùng nổ, trở thành một nguồn tài nguyên quý giá nhưng đầy thách thức trong việc khai thác và xử lý. Nhận diện thực thể (Named Entity Recognition - NER) nổi lên như một viên ngọc quý trong kho tàng Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP), mang trong mình sứ mệnh quan trọng: nhận biết và phân loại các thực thể như tên người, tổ chức, và địa điểm từ văn bản thô. Việc ứng dụng NER không chỉ giới hạn trong việc cải thiện tìm kiếm thông tin, mà còn mở ra những cánh cửa mới cho trích xuất dữ liệu, phân tích cảm xúc, và dịch thuật tự động.

Trong bối cảnh đó, báo cáo này tập trung vào hai phương pháp học máy hiện đại trong lĩnh vực NER: BERT (Bidirectional Encoder Representations from Transformers) và BiLSTM (Bidirectional Long Short-Term Memory). BERT, một sáng tạo tiên phong của Google, vượt trội với khả năng nắm bắt ngữ cảnh hai chiều trong văn bản nhờ vào cấu trúc transformer, đã tạo nên những thành tựu đáng kể trong nhiều nhiệm vụ NLP. Với khả năng hiểu biết ngữ cảnh toàn diện, BERT đã trở thành ngọn hải đăng dẫn đường trong xử lý ngôn ngữ tự nhiên.

Ngược lại, BiLSTM, một biến thể của mạng nơ-ron hồi quy (RNN), dù mang trong mình vẻ ngoài đơn giản hơn, vẫn có sức mạnh đáng nể trong việc xử lý ngữ cảnh từ cả hai hướng của chuỗi văn bản. BiLSTM sử dụng cấu trúc hai chiều để tích hợp thông tin, tạo nên một cách tiếp cận tinh tế và hiệu quả trong nhiều ứng dụng NER. Mặc dù có thể không đạt đến độ tinh vi như BERT, BiLSTM vẫn là một công cụ mạnh mẽ trong kho vũ khí của NLP.

Mục tiêu của báo cáo này không chỉ dừng lại ở việc mô tả cơ chế hoạt động và ưu nhược điểm của BERT và BiLSTM. Nhóm chúng em còn tiến hành thực nghiệm để so sánh hiệu suất của cả hai phương pháp trên một tập dữ liệu chung, từ đó đánh giá các chỉ số như độ chính xác (accuracy), độ chính xác (precision), độ nhạy (recall), và điểm F1 (F1-score). Những kết quả thu được sẽ được phân tích kỹ lưỡng để hiểu rõ hơn về yếu tố ảnh hưởng đến hiệu suất và khả năng ứng dụng của từng phương pháp.

Thông qua phân tích và so sánh, báo cáo này hy vọng sẽ mang đến cái nhìn sâu sắc và cụ thể về các kỹ thuật NER tiên tiến, giúp các nhà nghiên cứu và chuyên gia định hướng trong việc lựa chọn phương pháp phù hợp với các tình huống thực tế. Từ đó, có thể khai thác hiệu quả hơn tiềm năng vô tận của dữ liệu văn bản trong thế giới số hóa đang không ngừng phát triển.

**MỤC LỤC**

[**PHẦN 1. LÝ THUYẾT** 4](#_Toc170384970)

[**1. Nhận dạng thực thể (Named-Entity Recognition)**  4](#_Toc170384971)

[**1.1. Nhận dạng thực thể là gì?** 4](#_Toc170384972)

[**1.2. Các phương pháp nhận dạng thực thể** 5](#_Toc170384973)

[**2. Phương pháp áp dụng trong đề tài** 6](#_Toc170384974)

[**2.1. Mô hình BERT (Bidirectional Encoder Representations from Transformers)** 6](#_Toc170384975)

[**PHẦN 2. ỨNG DỤNG** 18](#_Toc170384976)

[**1. Mô tả tập dữ liệu** 18](#_Toc170384977)

[**2. Thiết kế mô hình** 18](#_Toc170384978)

[**3. Kết luận** 23](#_Toc170384979)

[**TÀI LIỆU THAM KHẢO** 25](#_Toc170384980)

# **PHẦN 1. LÝ THUYẾT**

## **1. Nhận dạng thực thể (Named-Entity Recognition)** [1]

### **1.1. Nhận dạng thực thể là gì?**

- Nhận dạng thực thể được đặt tên (NER) là một kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) tập trung vào việc xác định và phân loại các thực thể. Mục đích của NER là tự động trích xuất thông tin có cấu trúc từ văn bản phi cấu trúc, cho phép máy hiểu và phân loại các thực thể một cách có ý nghĩa cho các ứng dụng khác nhau như tóm tắt văn bản, xây dựng biểu đồ tri thức, trả lời câu hỏi và xây dựng biểu đồ tri thức.

- NER là thành phần trích xuất thông tin nhằm mục đích xác định và phân loại các thực thể được đặt tên trong văn bản phi cấu trúc. NER liên quan đến việc xác định thông tin chính trong văn bản và phân loại thành một tập hợp các danh mục được xác định trước. Thực thể là thứ được nói đến hoặc đề cập đến một cách nhất quán trong văn bản, chẳng hạn như tên người, tổ chức, địa điểm, cách diễn đạt thời gian, số lượng, tỷ lệ phần trăm và nhiều danh mục được xác định trước khác.

- Các ứng dụng vây của hệ thống NER trên nhiều lĩnh vực khác nhau, bao gồm trả lời câu hỏi, truy xuất thông tin và dịch máy. NER đóng một vai trò quan trọng trong việc nâng cao độ chính xác của các tác vụ NLP khác như gắn thẻ và phân tích cú pháp một phần giọng nói. Về cốt lõi, NLP chỉ là một quá trình gồm hai bước, dưới đây là hai bước có liên quan:

* Phát hiện các thực thể từ văn bản
* Phân loại chúng thành các loại khác nhau

### **1.2. Các phương pháp nhận dạng thực thể**

* Phương pháp dựa trên từ điển

NER sử dụng từ điển với danh sách các từ hoặc thuật ngữ. Quá trình này bao gồm việc kiểm tra xem có bất kỳ từ nào trong số này có trong một văn bản nhất định hay không. Tuy nhiên, cách tiếp cận này không được sử dụng phổ biến vì nó đòi hỏi phải cập nhật liên tục và bảo trì từ điển cẩn thận để luôn chính xác và hiệu quả.

* Phương pháp dựa trên quy tắc

Phương pháp NER dựa trên quy tắc sử dụng một tập hợp các quy tắc được xác định trước để hướng dẫn việc trích xuất thông tin. Các quy tắc này dựa trên mô hình và bối cảnh. Các quy tắc dựa trên mẫu tập trung vào cấu trúc và hình thức của từ, xem xét các mẫu hình thái của chúng. Mặt khác, các quy tắc dựa trên ngữ cảnh xem xét các từ xung quanh hoặc ngữ cảnh mà một từ xuất hiện trong tài liệu văn bản. Sự kết hợp giữa các quy tắc dựa trên mẫu và dựa trên ngữ cảnh này giúp nâng cao độ chính xác của việc trích xuất thông tin trong Nhận dạng thực thể được đặt tên (NER).

* Phương pháp dựa trên học máy
  + Phân loại nhiều lớp bằng thuật toán học máy

Một cách là huấn luyện mô hình phân loại nhiều lớp bằng các thuật toán học máy khác nhau, nhưng nó đòi hỏi nhiều nhãn. Ngoài việc gắn nhãn, mô hình còn đòi hỏi sự hiểu biết sâu sắc về ngữ cảnh để giải quyết sự mơ hồ của câu. Điều này làm cho nó trở thành một nhiệm vụ đầy thách thức đối với một thuật toán học máy đơn giản.

* + Conditional Random Field (CRF)

Conditional Random Field được triển khai bởi cả NLP Speech Tagger và NLTK. Đây là một mô hình xác suất có thể được sử dụng để mô hình hóa dữ liệu tuần tự như các từ

* Phương pháp dựa trên Deep Learning

Hệ thống NER học sâu chính xác hơn nhiều so với phương pháp trước đó vì nó có khả năng tập hợp các từ. Điều này là do nó đã sử dụng một phương pháp gọi là nhúng từ, có khả năng hiểu được mối quan hệ ngữ nghĩa và cú pháp giữa các từ khác nhau.

Nó cũng có thể tự động học phân tích các chủ đề cụ thể cũng như các từ cấp cao.

Điều này làm cho NER học sâu có thể áp dụng được để thực hiện nhiều nhiệm vụ. Học sâu có thể tự thực hiện hầu hết các công việc lặp đi lặp lại, do đó, các nhà nghiên cứu chẳng hạn có thể sử dụng thời gian của họ hiệu quả hơn.

## **2. Phương pháp áp dụng trong đề tài**

### **2.1. Mô hình BERT (**[**Bidirectional Encoder Representations from Transformers**](https://arxiv.org/abs/1810.04805)**)**

**2.1.1. Lý thuyết về BERT**

- BERT là viết tắt của Bidirectional Encoder Representations from Transformers được hiểu là một mô hình học sẵn hay còn gọi là pre-train model, học ra các vector đại diện theo ngữ cảnh 2 chiều của từ, được sử dụng để transfer sang các bài toán khác trong lĩnh vực xử lý ngôn ngữ tự nhiên. BERT đã thành công trong việc cải thiện những công việc gần đây trong việc tìm ra đại diện của từ trong không gian số (không gian mà máy tính có thể hiểu được) thông qua ngữ cảnh của nó. [2]

- Đào tạo trước và tinh chỉnh

Mô hình BERT trải qua quá trình gồm hai bước:

* Đào tạo trước về số lượng lớn văn bản chưa được gắn nhãn để tìm hiểu cách nhúng theo ngữ cảnh.
* Tinh chỉnh dữ liệu được gắn nhãn cho các nhiệm vụ NLP cụ thể .

- Đào tạo trước về dữ liệu lớn

* BERT được đào tạo trước về số lượng lớn dữ liệu văn bản chưa được gắn nhãn. Mô hình này tìm hiểu các phần nhúng theo ngữ cảnh, là cách thể hiện các từ có tính đến ngữ cảnh xung quanh chúng trong một câu.
* BERT tham gia vào nhiều nhiệm vụ đào tạo trước không có giám sát. Ví dụ: nó có thể học cách dự đoán các từ còn thiếu trong một câu (Mô hình ngôn ngữ đeo mặt nạ hoặc nhiệm vụ MLM), hiểu mối quan hệ giữa hai câu hoặc dự đoán câu tiếp theo trong một cặp.

- Tinh chỉnh trên dữ liệu được gắn nhãn

* Sau giai đoạn đào tạo trước, mô hình BERT, được trang bị các phần nhúng theo ngữ cảnh, sau đó được tinh chỉnh cho các tác vụ xử lý ngôn ngữ tự nhiên (NLP) cụ thể. Bước này điều chỉnh mô hình cho phù hợp với các ứng dụng được nhắm mục tiêu hơn bằng cách điều chỉnh cách hiểu ngôn ngữ chung của nó cho phù hợp với các sắc thái của nhiệm vụ cụ thể.
* BERT được tinh chỉnh bằng cách sử dụng dữ liệu được dán nhãn cụ thể cho các nhiệm vụ quan tâm phía sau. Các nhiệm vụ này có thể bao gồm phân tích tình cảm, trả lời câu hỏi, nhận dạng thực thể được đặt tên hoặc bất kỳ ứng dụng NLP nào khác. Các tham số của mô hình được điều chỉnh để tối ưu hóa hiệu suất của nó cho các yêu cầu cụ thể của nhiệm vụ hiện tại.

**2.1.2. Cách hoạt động của mô hình** [3]

- BERT được thiết kế để tạo ra một mô hình ngôn ngữ nên chỉ sử dụng cơ chế mã hóa. Chuỗi mã thông báo được đưa đến bộ mã hóa Transformer. Các mã thông báo này trước tiên được nhúng vào các vectơ và sau đó được xử lý trong mạng lưới thần kinh. Đầu ra là một chuỗi các vectơ, mỗi vectơ tương ứng với một mã thông báo đầu vào, cung cấp các biểu diễn theo ngữ cảnh.

- Khi đào tạo mô hình ngôn ngữ, việc xác định mục tiêu dự đoán là một thách thức. Nhiều mô hình dự đoán từ tiếp theo trong một chuỗi, đây là cách tiếp cận có định hướng và có thể hạn chế việc học theo ngữ cảnh. BERT giải quyết thách thức này bằng hai chiến lược đào tạo sáng tạo:

* *Mô hình ngôn ngữ đeo mặt nạ (MLM)*
* *Dự đoán câu tiếp theo (NSP)*

*1. Mô hình ngôn ngữ đeo mặt nạ (MLM)*

- Trong quy trình đào tạo trước của BERT, một phần từ trong mỗi chuỗi đầu vào được ẩn đi và mô hình được đào tạo để dự đoán giá trị ban đầu của các từ được ẩn này dựa trên ngữ cảnh được cung cấp bởi các từ xung quanh.

* Masking words: Trước khi BERT học từ các câu, nó sẽ ẩn một số từ (khoảng 15%) và thay thế chúng bằng một ký hiệu đặc biệt, như [MASK].
* Đoán các từ ẩn: Công việc của BERT là tìm ra những từ ẩn này là gì bằng cách nhìn vào các từ xung quanh chúng. Nó giống như một trò chơi đoán xem một số từ bị thiếu ở đâu và BERT cố gắng điền vào chỗ trống.
* BERT học như thế nào:

BERT thêm một lớp đặc biệt lên trên hệ thống học tập của mình để đưa ra những phỏng đoán này. Sau đó, nó sẽ kiểm tra mức độ dự đoán của nó gần với các từ ẩn thực tế đến mức nào.

Nó thực hiện điều này bằng cách chuyển đổi những dự đoán của mình thành xác suất và nói: “Tôi nghĩ từ này là X và tôi rất chắc chắn về nó”.

* Đặc biệt chú ý đến những từ ẩn (Hidden words)

Trọng tâm chính của BERT trong quá trình đào tạo là hiểu đúng những từ ẩn này. Nó ít quan tâm đến việc dự đoán những từ không bị ẩn.

Điều này là do thách thức thực sự là tìm ra những phần còn thiếu và chiến lược này giúp BERT trở nên thực sự giỏi trong việc hiểu ý nghĩa và ngữ cảnh của các từ.

- Về mặt kỹ thuật,

* BERT thêm một lớp phân loại lên trên đầu ra của bộ mã hóa. Lớp này rất quan trọng để dự đoán các từ bị che giấu.
* Các vectơ đầu ra từ lớp phân loại được nhân với ma trận nhúng, chuyển đổi chúng thành thứ nguyên từ vựng. Bước này giúp căn chỉnh các cách biểu diễn được dự đoán phù hợp với không gian từ vựng.
* Xác suất của mỗi từ trong từ vựng được tính bằng hàm kích hoạt SoftMax . Bước này tạo ra sự phân bố xác suất trên toàn bộ từ vựng cho từng vị trí bị che.
* Hàm mất mát được sử dụng trong quá trình huấn luyện chỉ xem xét dự đoán của các giá trị bị che. Mô hình bị phạt vì sai lệch giữa dự đoán của nó và giá trị thực tế của các từ bị che giấu.
* Mô hình hội tụ chậm hơn mô hình định hướng. Điều này là do trong quá trình đào tạo, BERT chỉ quan tâm đến việc dự đoán các giá trị bị che mà bỏ qua việc dự đoán các từ không bị che. Nhận thức về bối cảnh ngày càng tăng đạt được thông qua chiến lược này sẽ bù đắp cho sự hội tụ chậm hơn.

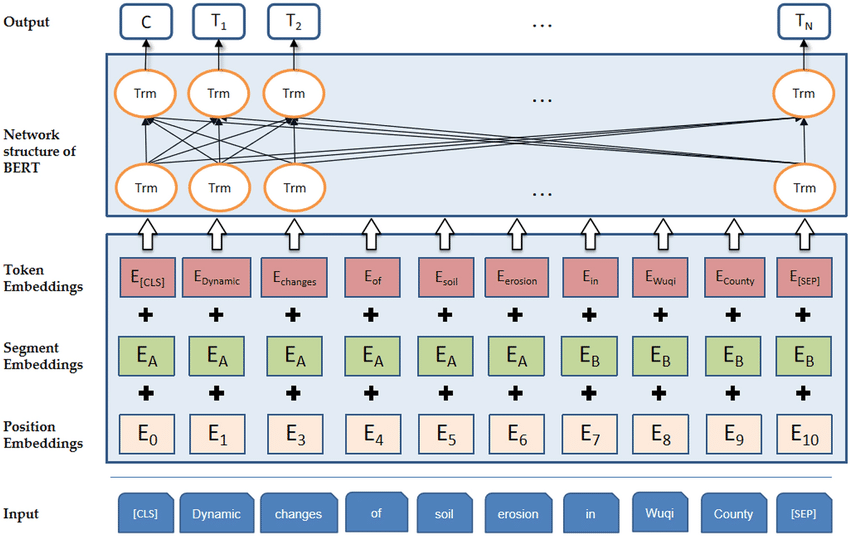
*2. Dự đoán câu tiếp theo (NSP)*

BERT dự đoán liệu câu thứ hai có được kết nối với câu đầu tiên hay không. Điều này được thực hiện bằng cách chuyển đổi đầu ra của ký tự [CLS] thành vectơ 2×1 bằng cách sử dụng lớp phân loại, sau đó tính toán xác suất xem câu thứ hai có tuân theo câu đầu tiên hay không bằng cách sử dụng SoftMax.

* Trong quá trình đào tạo, BERT học cách hiểu mối quan hệ giữa các cặp câu, dự đoán xem câu thứ hai có theo sau câu đầu tiên trong tài liệu gốc hay không.
* 50% cặp đầu vào có câu thứ hai là câu tiếp theo trong tài liệu gốc và 50% còn lại có câu được chọn ngẫu nhiên.
* Để giúp mô hình phân biệt được các cặp câu nối và ngắt kết nối. Đầu vào được xử lý trước khi vào mô hình:
  + Ký tự [CLS] được chèn vào đầu câu đầu tiên và Ký tự [SEP] được thêm vào cuối mỗi câu.
  + Việc nhúng câu chỉ ra Câu A hoặc Câu B sẽ được thêm vào mỗi mã thông báo.
  + Việc nhúng vị trí cho biết vị trí của từng mã thông báo trong chuỗi.
* BERT dự đoán liệu câu thứ hai có được kết nối với câu đầu tiên hay không. Điều này được thực hiện bằng cách chuyển đổi đầu ra của Ký tự [CLS] thành vectơ 2×1 bằng cách sử dụng lớp phân loại, sau đó tính toán xác suất xem câu thứ hai có tuân theo câu đầu tiên hay không bằng cách sử dụng SoftMax.

Trong quá trình đào tạo mô hình BERT, Masked LM và Next sentence prediction được đào tạo cùng nhau. Mô hình này nhằm mục đích giảm thiểu hàm mất mát kết hợp của Masked LM và Next sentence prediction, dẫn đến một mô hình ngôn ngữ mạnh mẽ với khả năng nâng cao trong việc hiểu ngữ cảnh trong câu và mối quan hệ giữa các câu.

**2.1.3. Cấu trúc mô hình BERT**



Giải thích:

* Input (Đầu vào):
  + Tokens: BERT sử dụng tokenization để chia văn bản đầu vào thành các token. Trong hình, ví dụ về văn bản là "Dynamic changes of soil erosion in Wuqi County".
  + [CLS] và [SEP] Tokens: Token [CLS] (Classification) được thêm vào đầu chuỗi văn bản và token [SEP] (Separation) được thêm vào cuối chuỗi văn bản. [CLS] thường dùng để tổng hợp thông tin của chuỗi cho các tác vụ phân loại, và [SEP] dùng để tách biệt các phần của văn bản hoặc phân cách các câu trong các tác vụ NER, QA.
* Embeddings (Các lớp nhúng):
  + Token Embeddings (Các lớp nhúng token): Mỗi token đầu vào được chuyển thành vector nhúng (embedding vector) với độ dài cố định. Ví dụ, token "Dynamic" được chuyển thành vector "changes" thành vector , ...
* Segment Embeddings (Các lớp nhúng đoạn): Được sử dụng để phân biệt giữa các đoạn khác nhau trong văn bản. Các token trong cùng một đoạn có các vector nhúng đoạn giống nhau. Trong hình, các đoạn A và B được đại diện bởi các vector ,.
* Position Embeddings (Các lớp nhúng vị trí): Mỗi vị trí trong chuỗi được ánh xạ đến một vector nhúng vị trí, giúp mô hình nhận biết được thứ tự của các token. Ví dụ, các vị trí 0, 1, 2, ... 10 được đại diện bởi các vector

,,.

Các embeddings này được cộng lại với nhau để tạo thành đầu vào cho tầng tiếp theo.

* Network Structure (Cấu trúc mạng):
* Transformers (Các tầng Transformer): BERT sử dụng nhiều tầng transformer để xử lý đầu vào. Mỗi tầng transformer bao gồm các khối attention đa đầu (multi-head attention) và các mạng nơ-ron hồi quy đầy đủ (feed-forward neural networks).
* Self-Attention Mechanism (Cơ chế tự chú ý): Tại mỗi tầng, cơ chế attention cho phép mô hình học cách phân phối sự chú ý (attention) đến các từ khác trong câu, giúp nắm bắt được ngữ cảnh từ hai hướng (trái và phải). Điều này được thể hiện qua các kết nối giữa các tầng Transformer trong hình.
* Output (Đầu ra):
* [CLS] Output: Đầu ra tại vị trí token [CLS] chứa thông tin tổng hợp từ toàn bộ chuỗi và thường được sử dụng cho các tác vụ như phân loại văn bản.
* Token Outputs: Đầu ra từ các token khác chứa các vector đặc trưng tương ứng, được sử dụng trong các tác vụ như NER (Nhận diện thực thể), QA (Câu hỏi & Trả lời), hoặc dự đoán từ tiếp theo. Trong hình, các đầu ra từ token , ,..., đại diện cho các vector đặc trưng của các token tương ứng trong câu.

Tổng quan cách hoạt động:

1. **Tiền xử lý (Preprocessing):** Văn bản đầu vào được chia thành các token, và các token này được thêm các embeddings về token, đoạn, và vị trí.
2. **Xử lý qua các tầng Transformer:** Các embeddings đầu vào đi qua nhiều tầng transformer. Mỗi tầng xử lý thông tin bằng cách áp dụng cơ chế tự chú ý để nắm bắt ngữ cảnh từ cả hai chiều của văn bản.
3. **Tính toán đầu ra:** Đầu ra từ tầng cuối cùng của transformer là các vector đặc trưng cho mỗi token. Các vector này được sử dụng cho các tác vụ khác nhau, như phân loại toàn văn bản (sử dụng đầu ra từ [CLS]), hay nhận diện thực thể (sử dụng đầu ra từ các token tương ứng).

Nhờ cấu trúc này, BERT có khả năng hiểu ngữ cảnh của từ trong câu một cách toàn diện hơn so với các mô hình trước đó, cho phép nó thực hiện tốt hơn trong các tác vụ NLP như NER, QA, và dịch máy.

**2.2. Mô hình BiLSTM (Bidirectional LSTM)**

**2.2.1. Lý thuyết mô hình BiLSTM** [4]

- LSTM hai chiều hoặc BiLSTM là thuật ngữ được sử dụng cho mô hình trình tự chứa hai lớp LSTM , một lớp để xử lý đầu vào theo hướng thuận và lớp kia để xử lý theo hướng ngược. Nó thường được sử dụng trong các nhiệm vụ liên quan đến NLP. Trực giác đằng sau cách tiếp cận này là bằng cách xử lý dữ liệu theo cả hai hướng, mô hình có thể hiểu rõ hơn mối quan hệ giữa các chuỗi (ví dụ: biết các từ sau và các từ trước trong câu).

- Để hiểu rõ hơn điều này chúng ta hãy xem một ví dụ. Câu đầu tiên là “Người phục vụ có thể mang cho tôi món ăn này không” và câu thứ hai là “Anh ấy đã làm hỏng máy chủ”. Trong cả hai câu này, từ máy chủ có ý nghĩa khác nhau và mối quan hệ này phụ thuộc vào các từ sau và trước trong câu lệnh. LSTM hai chiều giúp máy hiểu mối quan hệ này tốt hơn so với LSTM một chiều. Khả năng này của BiLSTM làm cho nó trở thành kiến ​​trúc phù hợp cho các nhiệm vụ như phân tích tình cảm , phân loại văn bản và dịch máy .

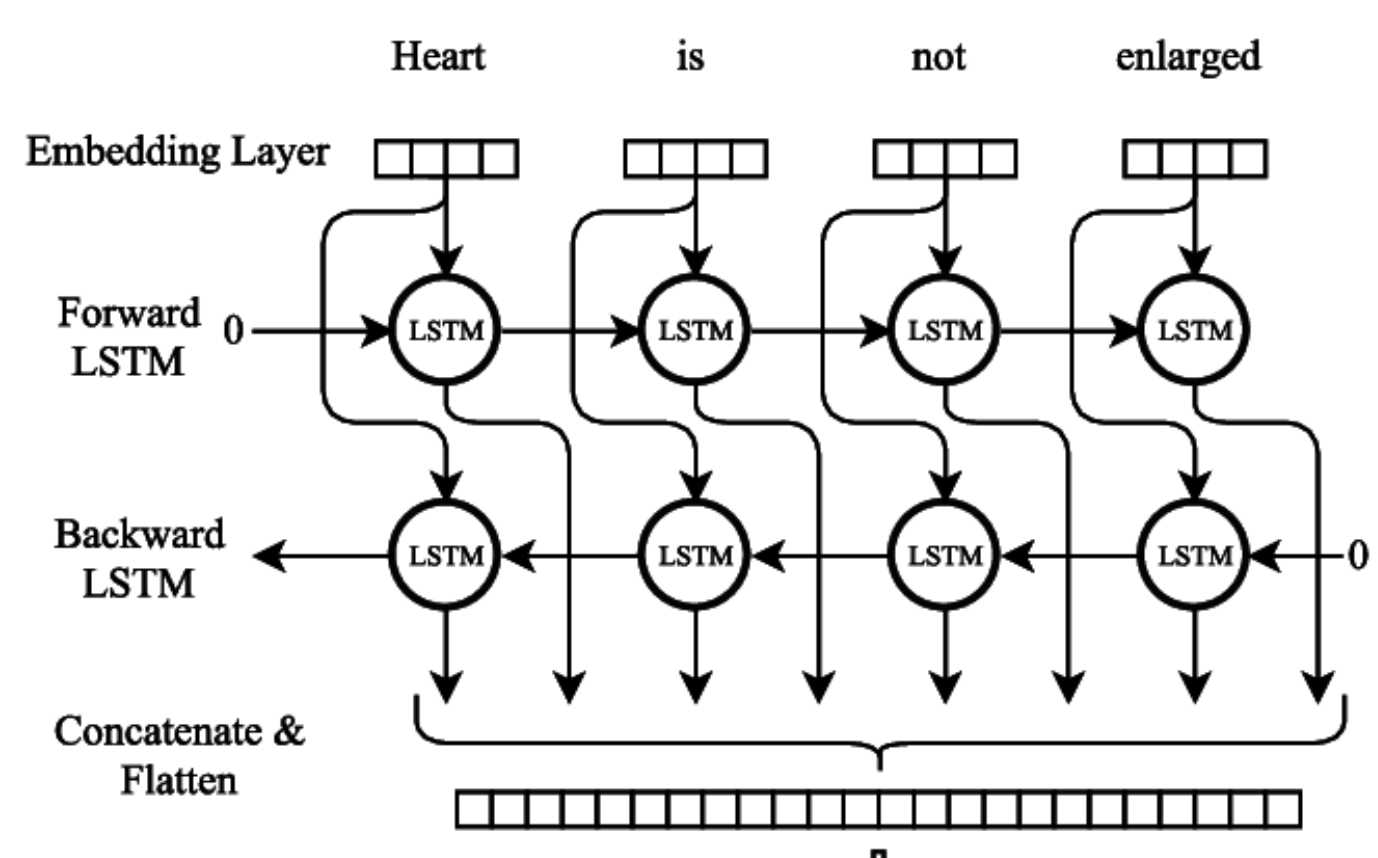
- Kiến trúc của LSTM hai chiều bao gồm hai LSTM một chiều xử lý chuỗi theo cả hai hướng tiến và lùi. Kiến trúc này có thể được hiểu là có hai mạng LSTM riêng biệt, một mạng lấy chuỗi mã thông báo như cũ trong khi mạng kia nhận theo thứ tự ngược lại. Cả hai mạng LSTM này đều trả về một vectơ xác suất làm đầu ra và đầu ra cuối cùng là sự kết hợp của cả hai xác suất này. Nó có thể được biểu diễn dưới dạng:

: Vectơ xác suất cuối cùng của mạng.

: Vectơ xác suất từ ​​mạng LSTM chuyển tiếp.

: Vectơ xác suất từ ​​mạng LSTM ngược.

**2.2.2. Cách hoạt động mô hình**



- Cấu trúc mô hình:

* Lớp nhúng (Embedding Layer):
* Chức năng: Chuyển đổi các từ trong văn bản đầu vào thành các vector nhúng có độ dài cố định, chứa thông tin ngữ nghĩa của từ.
* Ý nghĩa: Cung cấp các biểu diễn nhúng cho từ ngữ để làm đầu vào cho các tầng LSTM.
* LSTM xuôi (Forward LSTM):
* Chức năng: Xử lý chuỗi từ trái sang phải, tạo ra các trạng thái ẩn (hidden states) theo hướng xuôi của văn bản.
* Ý nghĩa: Nắm bắt ngữ cảnh từ quá khứ đến hiện tại trong chuỗi đầu vào, giúp hiểu được ngữ nghĩa khi đọc từ đầu đến cuối câu.
* LSTM ngược (Backward LSTM):
* Chức năng: Xử lý chuỗi từ phải sang trái, tạo ra các trạng thái ẩn theo hướng ngược của văn bản.
* Ý nghĩa: Nắm bắt ngữ cảnh từ hiện tại đến quá khứ, giúp hiểu được ngữ nghĩa khi đọc từ cuối lên đầu câu.
* Kết hợp và làm phẳng (Concatenate & Flatten):
* Chức năng: Kết hợp các trạng thái ẩn từ cả LSTM xuôi và LSTM ngược tại mỗi thời điểm, sau đó làm phẳng (flatten) thành một vector duy nhất.
* Ý nghĩa: Tạo ra một biểu diễn tổng hợp chứa thông tin ngữ cảnh từ cả hai hướng của chuỗi đầu vào, giúp mô hình hiểu ngữ cảnh một cách đầy đủ hơn.

- Cách Hoạt Động của BiLSTM

* Tiền xử lý (Preprocessing):

Văn bản đầu vào được chuyển thành các vector nhúng thông qua lớp nhúng. Mỗi từ trong chuỗi "Heart is not enlarged" được ánh xạ đến một vector nhúng tương ứng.

* Xử lý qua LSTM xuôi (Forward LSTM):

Forward LSTM nhận các vector nhúng và xử lý chúng từ trái sang phải. Mỗi từ được truyền qua một khối LSTM, tạo ra các trạng thái ẩn (hidden states) tương ứng, đại diện cho ngữ cảnh trong chuỗi từ đầu đến vị trí hiện tại.

* Xử lý qua LSTM ngược (Backward LSTM):

Đồng thời, Backward LSTM xử lý chuỗi từ phải sang trái, cũng tạo ra các trạng thái ẩn cho mỗi từ. Những trạng thái ẩn này đại diện cho ngữ cảnh từ cuối chuỗi đến vị trí hiện tại.

* Kết hợp và làm phẳng (Concatenate & Flatten):

Các trạng thái ẩn từ Forward LSTM và Backward LSTM tại mỗi thời điểm được kết hợp để tạo ra một vector kết hợp. Quá trình này tạo ra một biểu diễn tổng hợp cho mỗi từ trong chuỗi, nắm bắt thông tin ngữ cảnh từ cả hai hướng. Vector kết hợp sau đó có thể được làm phẳng và sử dụng cho các tác vụ tiếp theo như phân loại, dự đoán, hoặc nhận diện thực thể.

# **PHẦN 2. ỨNG DỤNG**

## **1. Mô tả tập dữ liệu**

- Nguồn lấy dữ liệu: [5]

- Tập dữ liệu gồm 47.959 câu được gán nhãn.

## **2. Thiết kế mô hình**

- Thiết kế của mô hình:

A screen shot of a computer program

Description automatically generated

1. Khởi tạo input: Nhận chuỗi văn bản đầu vào với độ dài cố định.

* ‘max\_len’ là độ dài tối đa của chuỗi đầu vào.
* Lớp Input định hình đầu vào như một chuỗi có chiều dài cố định là ‘max\_len’.

2. Lớp Embedding: Chuyển các từ thành vector nhúng.

* ‘input\_dim=num\_words’ xác định số lượng từ khác nhau (kích thước từ vựng).
* ‘output\_dim=max\_len’ xác định độ dài của mỗi vector nhúng.
* ‘input\_length=max\_len’ xác định chiều dài của chuỗi đầu vào.
* Lớp Embedding giúp tạo ra các biểu diễn vector cho các từ trong chuỗi.

3. Lớp SpatialDropout1D: Giảm overfitting bằng cách bỏ ngẫu nhiên một phần của các vector đầu vào.

* SpatialDropout1D(0.1) bỏ ngẫu nhiên 10% của các đơn vị đầu ra của chuỗi.

4. Lớp Bidirectional LSTM: Xử lý thông tin từ cả hai chiều của chuỗi, nắm bắt ngữ cảnh toàn diện.

* ‘units=100’ là số đơn vị (nút) trong lớp LSTM.
* ‘return\_sequences=True’ trả về toàn bộ chuỗi đầu ra cho mỗi thời điểm thay vì chỉ trạng thái cuối.
* ‘recurrent\_dropout=0.1’ áp dụng dropout cho các trạng thái ẩn bên trong LSTM để tránh overfitting.

5. Lớp TimeDistributed Dense: Áp dụng một lớp Dense cho mỗi từ trong chuỗi để dự đoán thẻ.

* num\_tags là số lượng thẻ (tags) cần dự đoán cho mỗi từ.
* activation='softmax' chuyển đổi đầu ra thành xác suất cho mỗi thẻ. (Softmax là một hàm kích hoạt thường được sử dụng trong các lớp đầu ra của các mô hình phân loại đa lớp. Softmax chuyển đổi các đầu vào thành xác suất, đảm bảo rằng tổng của các xác suất này là 1)

6. Tạo mô hình: Định nghĩa mô hình với đầu vào là ‘input\_word’ và đầu ra là ‘out’.

7. Tóm tắt mô hình: In ra cấu trúc của mô hình bao gồm tên các lớp, hình dạng của đầu ra, và số lượng tham số cần huấn luyện.

A black background with white text

Description automatically generated

8. Compile mô hình:

* optimizer='adam' sử dụng thuật toán tối ưu Adam. ()
* loss='categorical\_crossentropy' sử dụng hàm mất mát cross-entropy để huấn luyện mô hình phân loại đa lớp.
* metrics=['accuracy'] tính toán độ chính xác của mô hình trong quá trình huấn luyện.

- **Adam (Adaptive Moment Estimation)** là một thuật toán tối ưu hóa gradient descent, thường được sử dụng để huấn luyện các mô hình học sâu. Adam kết hợp những lợi ích của hai kỹ thuật tối ưu hóa phổ biến là **AdaGrad** và **RMSProp**, giúp tự động điều chỉnh tốc độ học cho từng tham số trong quá trình huấn luyện.

Ưu điểm:

* Điều chỉnh tốc độ học động: Tự động điều chỉnh tốc độ học cho mỗi tham số.
* Khả năng hội tụ nhanh: Kết hợp ưu điểm của SGD với động lực (momentum) và RMSProp, dẫn đến hội tụ nhanh hơn.
* Ổn định: Khả năng quản lý biến thiên của gradient tốt hơn nhờ trung bình động của bình phương gradient.

A screen shot of a computer

Description automatically generated

9. Khởi tạo callbacks: Định nghĩa các chức năng bổ trợ (callbacks) để giám sát và điều chỉnh quá trình huấn luyện.

* EarlyStopping dừng quá trình huấn luyện nếu độ chính xác trên tập validation không cải thiện sau patience epoch.
* PlotLossesKeras() (giả định rằng đây là một callback tuỳ chỉnh để hiển thị quá trình mất mát trong quá trình huấn luyện).

10. Huấn luyện mô hình:

* ‘x\_train’ là tập dữ liệu đầu vào để huấn luyện.
* ‘y\_train’ là nhãn tương ứng với tập ‘x\_train’.
* ‘validation\_split=0.2’ giữ lại 20% dữ liệu để làm tập validation.
* ‘batch\_size=32’ chia dữ liệu thành các lô với kích thước 32 mẫu mỗi lô.
* ‘epochs=3’ số lượng lần duyệt qua toàn bộ tập dữ liệu.
* ‘verbose=1’ hiển thị tiến trình huấn luyện.
* ‘callbacks=callbacks’ sử dụng các callback để giám sát và điều chỉnh quá trình huấn luyện.

- Biểu đồ so sánh kết quả 4 độ đo của bài 1 và bài 2:

A graph of blue and red bars

Description automatically generated

## **3. Kết luận**

Dựa trên kết quả so sánh các độ đo giữa hai phương pháp nhận diện thực thể BERT và BiLSTM, có thể thấy rằng mỗi phương pháp có những ưu và nhược điểm riêng. Phương pháp BERT vượt trội về độ chính xác (Accuracy) với giá trị 0.9855 so với 0.7287 của BiLSTM, cho thấy BERT có khả năng phân loại đúng phần lớn các thực thể trong tập dữ liệu. BERT cũng có độ chính xác về mặt dương tính (Precision) cao hơn, đạt 0.7207 so với 0.6037 của BiLSTM, nghĩa là nó ít có khả năng nhận diện sai các thực thể dương tính hơn.

Tuy nhiên, BiLSTM lại tỏ ra ưu việt hơn về độ nhạy (Recall) với giá trị 0.7287, cao hơn so với 0.5114 của BERT. Điều này cho thấy BiLSTM nhận diện được nhiều thực thể dương tính hơn trong tổng số thực thể dương tính có trong tập dữ liệu. Điểm F1 (F1 Score), trung bình hài hòa của Precision và Recall, của BiLSTM cũng cao hơn (0.7247 so với 0.5368 của BERT), cho thấy BiLSTM có sự cân bằng tốt hơn giữa độ chính xác và độ nhạy.

Tóm lại, tùy vào yêu cầu cụ thể của bài toán mà ta có thể lựa chọn phương pháp phù hợp. Nếu mục tiêu là đạt độ chính xác cao và giảm thiểu các lỗi dương tính, BERT là lựa chọn tốt. Ngược lại, nếu cần nhận diện được nhiều thực thể nhất có thể và có sự cân bằng giữa độ chính xác và độ nhạy, BiLSTM là phương pháp thích hợp hơn.

Cuối cùng, nhóm em đã hoàn thành mục tiêu xây dựng 2 mô hình BERT và BiLSTM cho tập dữ liệu văn bản để có thể nhận diện thực thể đã được gán nhãn. Tuy nhiên, sau khi kiểm tra chất lượng của các mô hình bằng 4 độ đo đã cho thấy kết quả không tốt, do chất lượng dữ liệu và việc xây dựng mô hình chưa được tốt có thể là nguyên nhân chính dẫn đến thang điểm không mong muốn này. Do thời gian, tài nguyên cũng như vốn kiến thức còn hạn chế nên báo cáo chưa cho ra kết quả tốt nhất. Do đó, nhóm em mong nhận được lời đóng góp của cô Nguyễn Thị Kim Ngân để có thể phát triển hơn trong tương lai.

Một lần nữa nhóm chúng em xin chân thành cảm ơn!

# **TÀI LIỆU THAM KHẢO**

|  |  |
| --- | --- |
| [1] | Geeksforgeeks, "Geeksforgeeks," Geeksforgeeks production, 03 01 2024. [Online]. Available: https://www.geeksforgeeks.org/named-entity-recognition/. [Accessed 27 06 2024]. |
| [2] | P. H. Quang, "VIBLO," VIBLO production, 27 11 2018. [Online]. Available: https://viblo.asia/p/bert-buoc-dot-pha-moi-trong-cong-nghe-xu-ly-ngon-ngu-tu-nhien-cua-google-RnB5pGV7lPG. [Accessed 27 06 2024]. |
| [3] | Geeksforgeeks, "Geeksforgeeks," Geeksforgeeks production, 10 01 2024. [Online]. Available: https://www.geeksforgeeks.org/explanation-of-bert-model-nlp/. [Accessed 27 06 2024]. |
| [4] | Geeksforgeeks, "Geeksforgeeks," Geeksforgeeks production, 08 06 2023. [Online]. Available: https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/. [Accessed 27 06 2024]. |
| [5] | N. JASWANI, "Kaggle," NAMAN JASWANI production, 2020. [Online]. Available: https://www.kaggle.com/datasets/namanj27/ner-dataset/data. [Accessed 27 06 2024]. |