Nội dung

[Phần 1: Bài toán NLU 2](#_Toc130029636)

[Phần 2: Giải quyết bài toán 2](#_Toc130029637)

[2.1 Chuẩn bị dữ liệu 2](#_Toc130029638)

[2.2 Về model 3](#_Toc130029639)

[2.2.1 Model JointBERT+CRF 3](#_Toc130029640)

[2.2.2 Model JointIDSF 6](#_Toc130029641)

[2.3 Kết quả của mô hình huấn luyện 8](#_Toc130029642)

[2.3.1 Thuật toán tối ưu, tham số, siêu tham số 8](#_Toc130029643)

[2.3.2 Hiệu suất trên tập huấn luyện 9](#_Toc130029644)

[Phần 3: Kết quả của bài toán: đánh giá model với test data 10](#_Toc130029645)

Hình ảnh

[Hình 1 Biểu đồ tròn thể hiện tỷ lệ % của các nhãn intent trên tập train 3](#_Toc129610137)

[Hình 2 Biểu đồ tròn thể hiện tỷ lệ % của các nhãn intent trên tập dev 3](#_Toc129610138)

[Hình 3 Lớp mã hóa (encoding layer) 4](#_Toc129610139)

[Hình 4 Mô hình JointBERT+CRF 6](#_Toc129610140)

[Hình 5 Mô hình JointIDSF 8](#_Toc129610141)

Bảng

[Bảng 1 Bảng thể hiện hiệu suất NPU (%) trên tập kiểm tra của quá trình huấn luyện 9](#_Toc129615221)

[Bảng 2 Bảng thể hiện kết quả (%) chính xác của bài toán 10](#_Toc129615222)

# Phần 1: Bài toán NLU

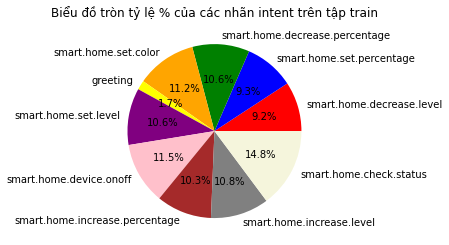
* Mô tả bài toán: Hệ thống nhà thông minh là 1 hệ thống nhà tiện lợi mà các thiết bị có thể được điều khiển từ xa thông qua câu lệnh của người dùng
* Mục tiêu bài toán: nhận diện ý định và các thực thể xuất hiện trong câu
* Đầu vào là 1 câu và đầu ra bao gồm ý định của câu và các thực thể trong câu
* Dạng dữ liệu cho bài toán phân loại slot được đánh nhãn theo định dạng BIO:
  + B- (beginning): đại diện cho từ đầu tiên của một thực thể
  + I- (inside): đại diện cho từ tiếp theo của một thực thể đã được bắt đầu bởi một nhãn B-
  + O- (outside): đại diện cho các từ không phải là một phần của thực thể
* Ví dụ: Bật hộ tôi cái đèn ở phòng ngủ
  + Ý định: Bật thiết bị
  + Thiết bị: Đèn
  + Vị trí của thiết bị: Phòng ngủ
* Kết quả: File submission\_results.csv không có header và sẽ chứa thông tin cho cả bài toán trích xuất ý định và bài toán trích xuất slot. File csv sẽ có dạng như sau:
  + ý định, các slot nối với nhau bởi dấu cách
  + ý định, các slot nối với nhau bởi dấu cách
  + …
* Độ đo đánh giá: Kết quả được đánh giá bằng sentence accuracy (số câu đúng cả intent lẫn tất cả các slot chia tổng số câu) trong file scores.txt

# Phần 2: Giải quyết bài toán

* Để giải quyết bài toán trên, em sử dụng 2 model:
  + JointBERT+CRF: BERT for Joint Intent Classification and Slot Filling. [monologg/JointBERT: Pytorch implementation of JointBERT: "BERT for Joint Intent Classification and Slot Filling" (github.com)](https://github.com/monologg/JointBERT)
  + JointIDSF: Joint intent detection and slot filling.   
    [VinAIResearch/JointIDSF: BERT-based joint intent detection and slot filling with intent-slot attention mechanism (INTERSPEECH 2021) (github.com)](https://github.com/VinAIResearch/JointIDSF)

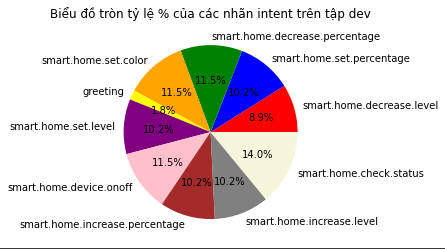
## 2.1 Chuẩn bị dữ liệu

* Dữ liệu huấn luyện gồm 2 thư mục là train và dev, trong mỗi thư mục có các file sau:
  + seq.in: chứa các câu cần phân loại intent và slot
  + label: chứa các nhãn intent cho từng câu trong seq.in
  + seq.out: chứa các nhãn slot cho từng từ của mỗi câu trong seq.in
  + intent\_label.txt: là tập từ điển cho các nhãn intent
  + slot\_label.txt: là tập từ điển cho các nhãn slot
* Trong thư mục train có chứa 1790 câu, 10 nhãn intent và 23 nhãn slot
* Biểu đồ phân chia các nhãn intent:



Hình Biểu đồ tròn thể hiện tỷ lệ % của các nhãn intent trên tập train

* Trong thư mục dev có chứa 392 câu, 10 nhãn intent và 24 nhãn slot



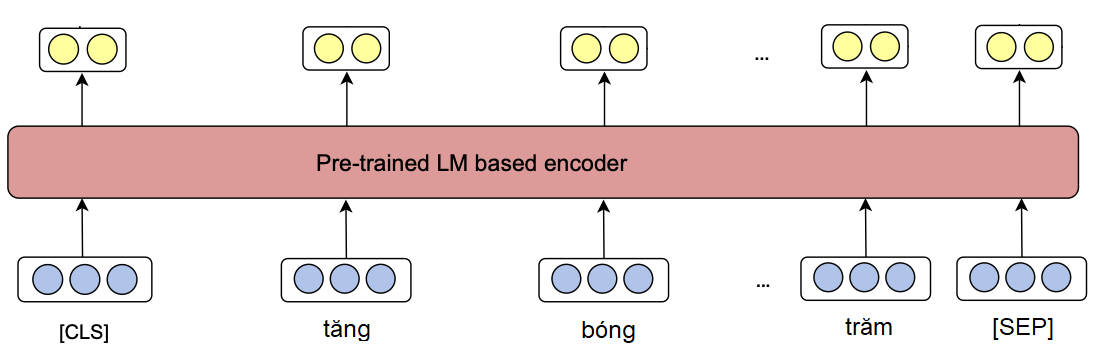
Hình Biểu đồ tròn thể hiện tỷ lệ % của các nhãn intent trên tập dev

## Về model

### Model JointBERT+CRF

* Mô hình JointBERT+CRF gồm 3 layers:
  + Encoding layer
  + Decoding layer of intent detection
  + Decoding layer of slot filling
* Encoding layer: Với mỗi câu đầu vào x bao gồm n tokens: x1, x2, … xn ta chèn token phân loại đặc biệt [CLS] ở đầu câu (x0) để xác định intent và token thông báo đặc biệt [SEP] ở cuối câu để thông báo kết thúc câu. Từ đó, ta có một câu đầu vào x với n+2 tokens x0, x1, x2, … xn, xn+1 cho lớp mã hóa (encoding layer). Lớp mã hóa sẽ sử dụng một mô hình ngôn ngữ dựa trên Transformer đã được tiền huấn luyện (pre-trained) để tạo ra các véc-tơ embedding đặc trưng có ngữ cảnh ci ∈ tương ứng của token xi:

(1)



Hình Lớp mã hóa (encoding layer)

Ở đây là kích thước véc-tơ embedding đầu ra của bộ mã hóa

Trong model này, ta sử dụng 2 pre-trained LM based encoder là PhoBERT encoder và XLM-R encoder:

* Bộ mã hóa PhoBERT là một biến thể đơn ngôn ngữ của RoBERTa cho tiếng Việt, sử dụng cấp độ từ (word level). PhoBERT được huấn luyện trên một tập dữ liệu lớn (20GB) được thu thập từ các nguồn đa dạng. Sau quá trình huấn luyện, bộ mã hóa này có khả năng hiểu được ngữ nghĩa của câu và từng từ trong câu.
* Bộ mã hóa XLM-R (Cross-lingual Language Model - RoBERTa) là một biến thể đa ngôn ngữ của RoBERTa được huấn luyện trên một tập dữ liệu lớn, đa ngôn ngữ (2.5TB chứa 137GB văn bản tiếng Việt), sử dụng cho cấp độ âm tiết (syllable level). Nó cải tiến đáng kể trong việc hiểu được các văn bản phức tạp đa ngôn ngữ giúp mô hình học được cách biểu diễn ngôn ngữ phổ quát và có khả năng xử lý văn bản ở các ngôn ngữ khác nhau dù cho không hiểu nghĩa của chúng.
* Intent detection layer: Lớp phát hiện ý định dựa trên véc-tơ embedding đặc trưng có ngữ cảnh c0 của token [CLS] . Layer sẽ đưa c0 vào một mạng nơ-ron truyền thẳng đơn giản (a single-layer feed-forward network FFNNID) với một bộ dự đoán softmax để dự đoán ý định (intent):

(2)

Ở đây, kích thước đầu ra của FFNNID là k (tổng số nhãn intent). Và cũng dựa theo véc-tơ xác suất p ∈ Rk ta tính toán được hàm mất mát cross-entropy LID cho việc phân loại ý định của quá trình huấn luyện.

* Slot filling layer: Lớp gán nhãn các thực thể (slot) có nhiệm vụ giống như một vấn đề gán nhãn chuỗi dựa trên BIO (a BIO-based sequence labeling problem). Layer sẽ đưa các véc-tơ embedding đặc trưng có ngữ cảnh ci tương ứng của token xi còn lại: c1, c2, …, cn vào một mạng nơ-ron truyền thẳng đơn giản FFNN khác (FFNNSF) để tạo ra các trạng thái ẩn (hidden states) h1, h2, …, hn tương ứng:

(3a)

Sau đó, đưa các trạng thái ẩn này vào một lớp softmax để gán nhãn các thực thể với kích thước đầu ra của FFNNSF là tổng số nhãn thực thể (slot):

(4)

Hàm mất mát cross-entropy LSF được tính toán cho việc gán nhãn thực thể của quá trình huấn luyện.

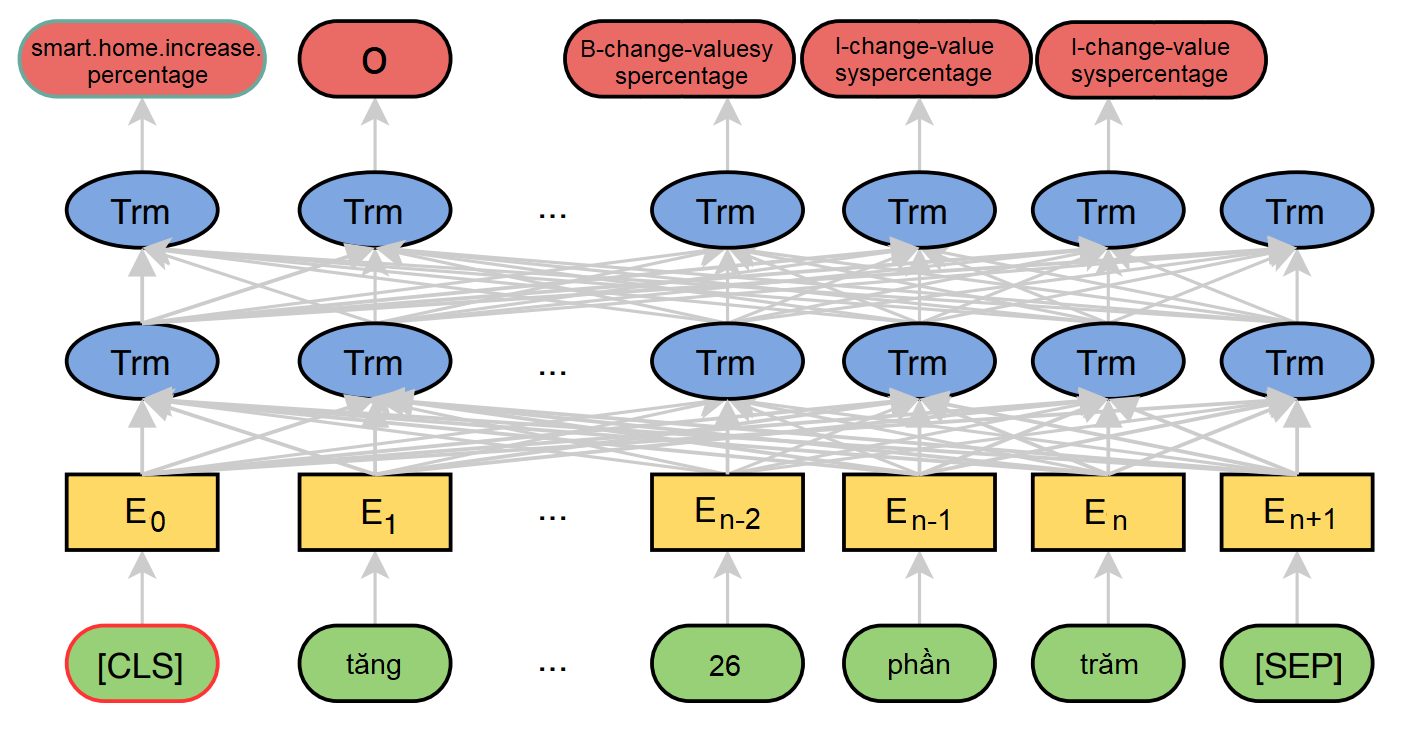
Cuối cùng slot filling layer và intent detection layer đưa các véc-tơ đầu ra trên vào một bộ dự đoán chuỗi tuyến tính có điều kiện CRF (a linear-chain Conditional Random Field (CRF) predictor) để dự đoán intent và slot:

Conditional Random Field (CRF): là thuận toán xác suất có điều kiện dùng để dự đoán các nhãn ý định (intent) và nhãn thực thể (slot) không chỉ dựa trên từng từ mà còn tính đến các từ khác trong cả câu. Mô hình tính toán xác suất của nhãn intent và nhãn slot dựa trên tích của xác suất nhãn intent dựa trên câu đầu vào x với tích của xác suất slot từng từ trong câu đầu vào x. Từ đó đưa ra kết quả dự đoán intent và slot của model.

* Joint training: Hàm mất mát đối tượng cuối cùng (The final training objective loss) L của mô hình JointBERT+CRF là tổng có trọng số của hàm mất mát cross-entropy LID cho việc phân loại ý định và hàm mất mát cross-entropy LSF cho việc gán nhãn thực thể:

(6)

* Ví dụ về mô hình JointBERT+CRF với câu đầu vào “tăng bóng 3 26 phần trăm”



Hình Mô hình JointBERT+CRF

### Model JointIDSF

* Mô hình JointIDSF là mô hình kế thừa mô hình JointBERT+CRF kể trên. Nghĩa là ta có thể nâng cấp mô hình JointBERT+CRF lên mô hình JointIDSF chỉ với việc thêm một layer attention (an intermediate intent-slot attention layer).
* Do đó, mô hình JointIDSF sẽ bao gồm 4 layers:
  + Encoding layer
  + Decoding layer of intent detection
  + Decoding layer of slot filling
  + Intermediate intent-slot attention layer
* Intent-slot attention layer:

Đầu tiên, em xin phép được giới thiệu về cấu trúc chính của layer. Layer sử dụng cơ chế attention: là trung bình có trọng số của những “thứ” mà chúng ta nghĩ nó cần thiết cho bài toán, điều đặc biệt là trọng số này do mô hình tự học được.

Cụ thể, layer sử dụng cơ chế attention để căn chỉnh mức độ quan trọng của ý định (intent) với slot của mỗi từ trong câu. Cơ chế làm việc: Intent-slot attention layer sử dụng đầu ra từ Encoding layer và Intent detection layer để tạo ra các véc-tơ cụ thể cho từng ý định, sau đó được sử dụng để làm một phần đầu vào cho Slot filling layer

Ban đầu, Attention layer sẽ tạo ra một nhãn nhúng embedding ý định mềm (a “soft” intent label embedding) w ∈ bằng cách nhân ma trận trọng số W ∈ với véc-tơ xác suất p = yi ∈ Rk:

(7)

Sau đó, sử dụng nhãn embedding ý định mềm (“soft” intent label embedding) w và các véc-tơ embedding đặc trưng có ngữ cảnh ci để tạo ra các véc-tơ cụ thể cho từng ý định si (i ∈ {1, 2, …, n}):

Sau đó ta nâng cấp Slot filling layer với cơ chế attention

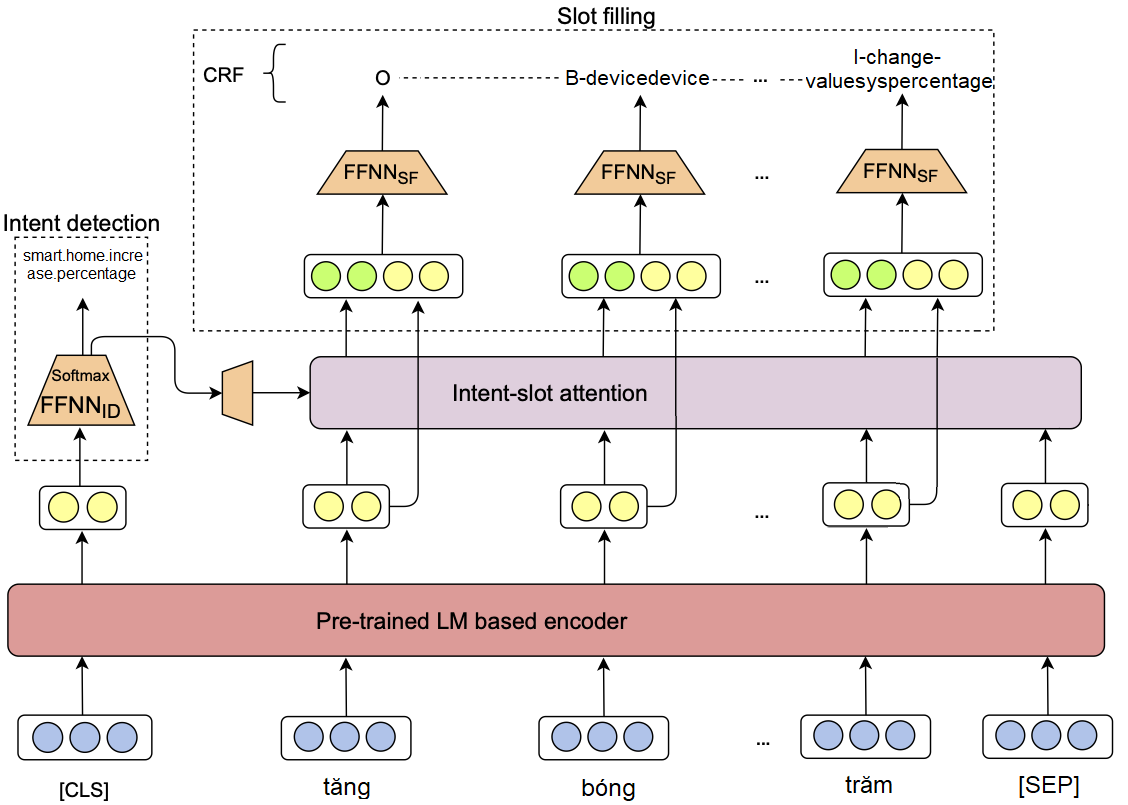
* Slot filling layer: Thay vì sử dụng các véc-tơ embedding đặc trưng có ngữ cảnh ci tương ứng của token xi để đưa vào FFNNSF (model JointBERT+CRF) thì model JointIDSF sẽ tạo ra một chuỗi các véc-tơ . Trong đó mỗi véc-tơ được tạo ra bằng cách nối véc-tơ cụ thể cho từng ý định si với véc-tơ embedding đặc trưng có ngữ cảnh ci tương ứng:

(10)

Sau đó, layer đưa từng véc-tơ vào mạng nơ-ron truyền thẳng đơn giản FFNNSF và tiếp tục thực hiện các bước tiếp theo giống với model JointBERT+CRF mà chỉ thay véc-tơ ci bằng véc-tơ vi :

(3b)

* Ví dụ về mô hình JointIDSF với câu đầu vào “tăng bóng 3 26 phần trăm”



Hình Mô hình JointIDSF

## Kết quả của mô hình huấn luyện

### Thuật toán tối ưu, tham số, siêu tham số

* Cả hai mô hình JointIDSF và JointBERT+CRF đều sử dụng thuật toán tối ưu hóa Adam (Adam optimizer) với batch size = 32. Adaptive Moment Estimation optimizer là một thuật toán cho phép tính tốc độ học (learing rate) thích ứng với mỗi trọng số. Adam là sự kết hợp của thuật toán tối ưu hóa Gradient descent with momentum với RMSprop.
  + Gradient descent with momentum: bỏ qua các local minimum để nghiệm tiến tới global minimum
  + RMSprop: tự động cải thiện learing rate theo thời gian

Adam không chỉ lưu trữ trung bình bình phương các gradient trước đó mà còn lưu cả giá trị trung bình mô-men, từ đó tự động cải thiện learning rate và bỏ qua các local minimum để nghiệm tiến tới global minimum.

* Ngoài ra các tham số, siêu tham số còn lại được tham khảo từ mô hình JointIDSF do em chưa có nhiều kinh nghiệm trong việc điều chỉnh tham số, siêu tham số
* Cả 2 model huấn luyện trong 50 epoch và tính điểm trung bình của độ chính xác khi phát hiện ý định (intent) và F1-score cho xác định các thực thể (slot) sau mỗi epoch huấn luyện trên tập validation (dev). Từ đó lấy mô hình điểm số cao nhất trên tập validation (dev) để sử dụng cho tập test.

### Hiệu suất trên tập huấn luyện

* Kết quả trên tập kiểm tra (test set):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Encoder** | **Intent Acc** | **Slot F1** | **Sent Acc** |
| JointBERT+CRF | PhoBERT | 88.61 | 92.35 | 90.48 |
| JointIDSF | PhoBERT | 81.01 | 92.55 | 86.78 |
| JointBERT+CRF | XLM-R | 79.75 | 91.58 | 85.66 |
| JointIDSF | XLM-R | 86.08 | 92.15 | 89.11 |

Bảng Bảng thể hiện hiệu suất NPU (%) trên tập kiểm tra của quá trình huấn luyện

Bảng 1 sử dụng các chỉ số đánh giá bao gồm:

* Intent Acc: độ chính xác của phân loại ý định (intent classification accuracy)
* Slot F1: F1-score cho xác định các thực thể (F1-score for slot filling)
* Sent Acc: Độ chính xác tổng thể (intent + slot) của câu

Với 2 bộ mã hóa là PhoBERT và XLM-R.

* Từ bảng 1 ta có thể thấy:
  + Với bộ mã hóa PhoBERT thì mô hình JointBERT+CRF trội hơn hoàn toàn so với mô hình JointIDSF về việc phân loại ý định (intent) [88.61 > 81.01] và độ chính xác tổng thể [90.48 >86.78] và chỉ nhỏ hơn 1 chút về xác định các thực thể (slot) [92.35 < 92.55]. Lý do: bộ mã hóa PhoBERT là bộ mã hóa cho cấp độ từ (word level) mà tập dữ liệu đang sử dụng cấp độ âm tiết (syllable level). Ví dụ: biểu diễn từ sân bay trong cấp độ word là “sân\_bay” còn cấp độ syllable là “sân bay”.
  + Với bộ mã hóa XLM-R thì mô hình JointIDSF trội hơn hoàn toàn so với mô hình JointBERT+CRF về cả việc phân loại ý định (intent) [86.08 > 79.75], xác định các thực thể (slot) [91.58 < 92.15] và độ chính xác tổng thể [89.11 >85.66]
  + Xét về 2 bộ mã hóa thì bộ mã hóa PhoBERT nhỉnh hơn so với bộ mã hóa XLM-R về độ chính xác tổng thể.

# Phần 3: Kết quả của bài toán: đánh giá model với test data

* Dữ liệu của mô hình đã được huấn luyện được lưu với tên: MODEL\_DIR\_“Tên model”\_“Tên encoding”encoder.   
  Vd: MODEL\_DIR\_JointBERT-CRF\_PhoBERTencoder
* Trong đó có 2 thư mục private\_test\_data và public\_test\_data lưu kết quả đánh giá model với file submission\_results.csv chứa kết quả của bài toán và file scores.txt chứa kết quả được đánh giá bằng sentence accuracy.
* Kết quả với test data:

|  |  |  |
| --- | --- | --- |
| **Model** | **Test data** | **Sentence accuracy** |
| JointBERT+CRF\_PhoBERT | Private | 64.17 |
| Public | 65.32 |
| JointBERT+CRF\_XLM-R | Private | 67.78 |
| Public | 66.47 |
| JointIDSF\_PhoBERT | Private | 64.72 |
| Public | 69.94 |
| JointIDSF\_XLM-R | Private | 67.5 |
| Public | 68.79 |

Bảng Bảng thể hiện kết quả (%) chính xác của bài toán