

# Nhận diện thực thể tên riêng tiếng Việt (Vietnamese Named Entity Recognition)

Nhóm 4

Nguyễn Công Phát, Nguyễn Lê Phong, Phạm Trần Khánh Duy  
23521143, 23521168, 23520384

Trường Đại học Công nghệ Thông tin – ĐHQG TP.HCM

GVHD: TS. Nguyễn Thị Quý

# Mục lục

- 1 Tổng quan đề tài
- 2 Tổng quan các phương pháp
- 3 Thực nghiệm
- 4 Kết luận & Hướng phát triển

# Mục lục

- 1 Tổng quan đề tài
- 2 Tổng quan các phương pháp
- 3 Thực nghiệm
- 4 Kết luận & Hướng phát triển

## 1.1. Giới thiệu bài toán NER

### Định nghĩa

Nhận diện thực thể tên riêng (Named Entity Recognition - NER) là bài toán **gán nhãn cho từng từ/token trong câu** để xác định các thực thể như *tên người (PER)*, *tổ chức (ORG)*, *địa danh (LOC)*, v.v.

### Ví dụ (BIO)

Hà\_Nội/B-LOC là/O thủ\_đô/O của/O Việt\_Nam/B-LOC ./O

### Ứng dụng

- Trích xuất thông tin trong tin tức, báo cáo.
- Hỗ trợ hệ thống hỏi đáp, tìm kiếm thực thể.
- Tiền xử lý cho dịch máy, tóm tắt văn bản.

### Thách thức với tiếng Việt

- Tên riêng thường nhiều từ: *Thành\_phố\_Hồ\_Chí\_Minh*.
- Ngữ cảnh phụ thuộc mạnh, khó đoán chỉ từ một token.
- Chữ hoa không nhất quán trong corpora tiếng Việt.
- Nhiều tên tổ chức dùng địa danh ⇒ dễ nhầm ORG ↔ LOC.

## 1.2. Lý do chọn bài toán

### Bối cảnh

- Named Entity Recognition (NER) là bài toán nền tảng trong xử lý ngôn ngữ tự nhiên.
- Đóng vai trò quan trọng trong các hệ thống *trích xuất thông tin, hỏi–đáp và phân tích văn bản*.
- Với **tiếng Việt**, NER gặp nhiều thách thức do:
  - Ngôn ngữ đa âm tiết, cấu trúc tên riêng phức tạp.
  - Hạn chế về dữ liệu gán nhãn chất lượng cao.

Phương pháp	Mô tả	Hạn chế
<b>HMM</b>	Mô hình xác suất gán nhãn chuỗi dựa trên xác suất chuyển trạng thái (transition) và phát xạ (emission).	Giả định Markov, không học được ngữ cảnh dài.
<b>CRF</b>	Mô hình xác suất có điều kiện, tối ưu xác suất toàn chuỗi nhãn.	Phụ thuộc nhiều vào đặc trưng thủ công (features).
<b>BiLSTM-CRF</b>	Kết hợp LSTM hai chiều (học ngữ cảnh) và CRF (gán nhãn).	Cần dữ liệu lớn, chi phí huấn luyện cao.
<b>Transformers</b>	Các công trình NER tiếng Việt gần đây chủ yếu khai thác <i>fine-tuning</i> PhoBERT, XLM-R hoặc mT5 để tận dụng ngữ cảnh sâu.	Tốn tài nguyên; nhiều nghiên cứu dùng dữ liệu tự xây dựng/mở rộng và không công bố đầy đủ code & dữ liệu ⇒ khó tái lập.

Bảng: Tổng quan các hướng tiếp cận bài toán NER và hạn chế trong bối cảnh tiếng Việt.

### Mục tiêu đồ án

- So sánh định lượng ba mô hình: **HMM, CRF và BiLSTM-CRF**.
- Thực nghiệm trên cùng một tập dữ liệu chuẩn.
- Đánh giá mô hình bằng các độ đo phù hợp cho bài toán NER.

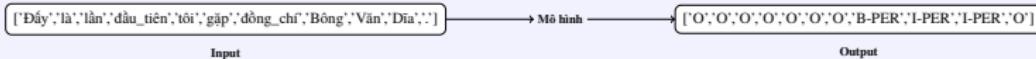
## 1.3. Phát biểu bài toán

### Input

- **Tập dữ liệu huấn luyện** gồm  $N$  câu đã gán nhãn theo chuẩn BIO:
  - Mỗi câu: dãy token  $(w_1, w_2, \dots, w_t)$  (token đã được tách sẵn trong dataset).
  - Mỗi token có nhãn tương ứng  $(l_1, l_2, \dots, l_t)$ .
- **Câu mới cần dự đoán:** chuỗi token  $(w_1, w_2, \dots, w_t)$ .

### Output

- Chuỗi nhãn dự đoán  $(l_1, l_2, \dots, l_t)$  ứng với từng token.
- Mỗi nhãn thuộc tập nhãn  $L$  theo chuẩn BIO:
  - **B-** và **I-**: bắt đầu và bên trong một thực thể (PER, ORG, LOC, MISC).
  - **O**: token không thuộc thực thể nào.



# Mục lục

- 1 Tổng quan đề tài
- 2 Tổng quan các phương pháp
- 3 Thực nghiệm
- 4 Kết luận & Hướng phát triển

## 2.1. Mô hình Hidden Markov Model (HMM)

### Tổng quan mô hình

- Hidden Markov Model (HMM)** là mô hình xác suất dùng để gán nhãn chuỗi.
- Giả định tồn tại chuỗi trạng thái ẩn  $Y = (y_1, \dots, y_T)$  sinh ra chuỗi quan sát:  $X = (x_1, x_2, \dots, x_T)$
- Hai xác suất cốt lõi:
  - Transition:**  $P(y_t | y_{t-1})$
  - Emission:**  $P(x_t | y_t)$
- Training:** ước lượng MLE;
- Decoding:** Viterbi (tối ưu chuỗi nhãn).

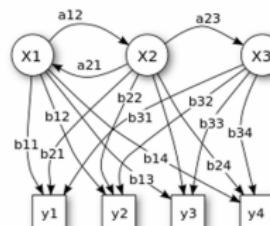
### Ưu điểm

- Dễ hiểu, nền tảng xác suất rõ ràng.
- Huấn luyện nhanh, phù hợp dữ liệu nhỏ.
- Mô hình hóa tốt bài toán gán nhãn chuỗi cơ bản.

### Nhược điểm

- Giả định Markov bậc 1  $\Rightarrow$  chỉ phụ thuộc trạng thái trước.
- Không tận dụng được ngữ cảnh dài.
- Hiệu quả thấp với dữ liệu mất cân bằng nhãn (NER tiếng Việt).

### Hidden Markov Model



Sơ đồ HMM: trạng thái ẩn (Y) sinh ra quan sát (X).

### Quy trình dự đoán (Decoding Process)

- Khởi tạo:** Tính xác suất cho các nhãn có thể ở vị trí đầu tiên.
- Đệ quy (Forward):** Tại mỗi bước  $t$ , tính xác suất tích lũy cực đại để đến trạng thái  $y_t$ :
$$\delta_t(y) = \max_{y'} [\delta_{t-1}(y') \cdot P(y|y')] \cdot P(x_t|y)$$
- Truy vết (Backtracking):** Từ trạng thái cuối cùng có điểm cao nhất, đi ngược về đầu câu để khôi phục chuỗi nhãn tối ưu.

## 2.2. Cài đặt HMM trong bài toán

### Cài đặt trong bài toán

#### Train + Decode:

- ➊ Đếm *transition*  $C(y_{t-1}, y_t)$  và *emission*  $C(y_t, x_t)$  (có trọng số).
- ➋ **Smoothing** (Xử lý zero probabilities):  $P = \frac{C + \epsilon}{\sum C + |\mathcal{Y}|\epsilon}$ ,  
 $\epsilon = 10^{-3}$ .
- ➌ Lấy log-prob, decode bằng Viterbi để ra chuỗi nhãn tốt nhất.

#### Trọng số lớp (mất cân bằng):

$$w(O) = 0.5, \quad w(y \neq O) = \min(5.0, \frac{\text{median\_freq}}{\text{count}(y)+10^{-5}})$$

### Các cải tiến được áp dụng

- ➊ **Balanced sampling**: oversample câu chứa nhãn hiếm đến ngưỡng  $\approx 0.5 \times$  lớp lớn nhất.
- ➋ **Class weighting**: Điều chỉnh quá trình ước lượng xuất hiện của nhãn bằng cách gán trọng số khác nhau cho từng loại nhãn, thay vì coi tất cả như nhau. (Giới hạn trọng số lớp: `max_weight = 5.0`).
- ➌ **Focal adjustment (tag  $\neq O$ )**: tăng xác xuất của các *emission* và *transition* hiếm bằng công thức:

$$p \leftarrow \min(1, p \cdot (1 + \alpha(1 - p)^\gamma)), \quad \alpha = 0.1, \quad \gamma = 2.0$$

Lưu ý: điều này khiến cho băng xuất không còn là true probability mà được xem như score.

### Pseudo-code (HMM + cải tiến)

```

# Input + hyper-params
Input: train_sents X, train_tags Y
Hyper: eps=1e-3, alpha=0.1, gamma=2.0,
       max_weight=5.0, w(0)=0.5

# 1) Imbalance handling
w(0)=0.5
w(y!=0)=min(max_weight,
            median_freq/(count(y)+1e-5))
X,Y = balanced_sampling(X,Y,
                        target=0.5*max_class)

# 2) Weighted counts
C_tr[y_{-t-1},y_t] += w(y_t)
C_em[y_t, x_t] += w(y_t)

# 3) Laplace smoothing
P_tr=(C_tr+eps)/(sum_next C_tr + |Y|*eps)
P_em=(C_em+eps)/(sum_word C_em + |V|*eps)

# 4) Focal adjustment (tag != O)
p = min(1, p*(1 + alpha*(1-p)^gamma))

# 5) Decode (Viterbi)
A=log(P_tr); B=log(P_em)
y_hat = Viterbi(A,B,x)

```

## 2.3. Mô hình Conditional Random Field (CRF)

### Tổng quan

- CRF mô hình hoá trực tiếp xác suất có điều kiện toàn chuỗi:  $P(Y | X)$ .
- Không già định độc lập Markov như HMM  $\Rightarrow$  khai thác **phụ thuộc nhãn liên kè**  $\Rightarrow$  biểu diễn trực tiếp mối quan hệ giữa các nhãn trong chuỗi..
- Giảm lỗi boundary (ví dụ I-ORG không thể theo sau B-PER)

### Đặc trưng (Features) sử dụng

- Token hiện tại và các token lân cận (context window).
- Hình thái: `is_upper`, `is_title`, `is_digit`, dấu gạch dưới.
- Tiền tố / hậu tố (prefix, suffix).
- Vị trí câu: BOS, EOS.

### Quy trình

- Trích xuất đặc trưng:** Tại mỗi vị trí  $t$ , xác định các đặc trưng  $f(y_t, y_{t-1}, X, t)$ .
- Tính điểm (Scoring):** Tính tổng trọng số các đặc trưng.
- Viterbi (Quy hoạch động):** Tại từ thứ  $t$ , tìm lộ trình có **điểm tích lũy cao nhất** để gán được **nhãn**  $j$ :

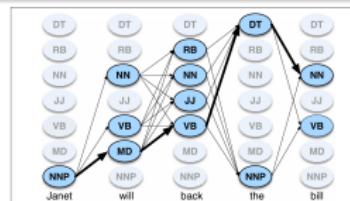
$$\nu_t(j) = \max_{i=1}^N \left[ \nu_{t-1}(i) + \sum_{k=1}^K w_k f_k(y_{t-1}, y_t, X, t) \right]$$

### Cài đặt CRF trong bài toán

- Thư viện: `sklearn-crfsuite`.
- Tối ưu: **L-BFGS** (`algorithm='lbfgs'`) – thuật toán tối ưu gradient hiệu quả cho CRF.
- Regularization:  $c1=0.1$  (L1) và  $c2=0.1$  (L2) nhằm giảm overfitting và kiểm soát độ phức tạp mô hình.
- `max_iterations=200`: giới hạn số vòng lặp huấn luyện để đảm bảo hội tụ.
- `all_possible_transitions=True`: cho phép xét cả các chuyển nhãn chưa xuất hiện trong tập huấn luyện, giúp mô hình xử lý tốt hơn các nhãn hiếm và giảm lỗi boundary.

### Lý do lựa chọn cấu hình:

Các tham số được chọn nhằm cân bằng giữa khả năng tổng quát hóa và hiệu năng mô hình trên tập dữ liệu, đồng thời đảm bảo CRF khai thác hiệu quả ràng buộc chuỗi nhãn trong bài toán NER tiếng Việt.



Sơ đồ mô hình CRF dùng thuật toán viterbi để decode nhãn.

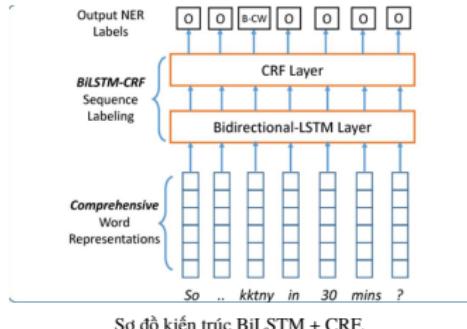
## 2.4. Mô hình BiLSTM + CRF

### Kiến trúc tổng thể

- ① **Embedding Layer:** Ánh xạ token → vector ngữ nghĩa.
- ② **BiLSTM Layer:** Học ngữ cảnh hai chiều (trái ↔ phải).
- ③ **CRF Layer:** Giải mã chuỗi nhãn tối ưu toàn cục.

### Luồng xử lý dự đoán (Prediction Pipeline)

- ① **Input:** Chuỗi index token.
- ② **BiLSTM Forward:**
  - Qua LSTM 2 chiều → vector ngữ cảnh  $h_t$ .
  - Tạo **Emission Scores** (điểm phát xạ).
- ③ **CRF Layer:**
  - Cung cấp **Transition Scores** (điểm chuyển nhãn).
- ④ **Viterbi Decoding:**
  - Tổng hợp Emission + Transition.
  - Tìm đường đi tối ưu.



Sơ đồ kiến trúc BiLSTM + CRF.

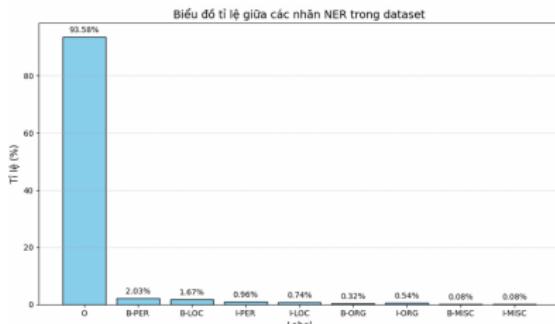
### Cài đặt trong bài toán

- **Embedding:** Train from scratch (random init), không dùng pre-trained.
- **CRF:** Decode bằng Viterbi.
- **Optimizer:** Adam, lr=0.001.
- **Config:** epochs=12, batch\_size=32, early\_stop=3.

## Mục lục

- 1 Tổng quan đề tài
  - 2 Tổng quan các phương pháp
  - 3 Thực nghiệm
  - 4 Kết luận & Hướng phát triển

# 3.1. Dataset



Biểu đồ tần suất (%) của các nhãn NER trong dataset.

## Thành phần nhãn trong VLSP2016:

- **O (Outside):** 93.58%
- **B-PER:** 2.03%
- **B-LOC:** 1.67%
- **B-ORG:** 0.32%
- **Các nhãn I-/MISC khác:** ~ 2.4%

## Nhận xét dữ liệu

Dữ liệu **mất cân bằng nghiêm trọng**: nhãn O chiếm **93.58%** tổng token.

## Thông tin dữ liệu

**Nguồn:** VLSP2016-NER-data (HuggingFace)

**Dạng dữ liệu:** file .parquet / token + nhãn IOB.

**Cấu trúc:** cột 1 = token, cột 2 = tag (B-, I-, O-).

**Quy mô:** ~13.5k câu train, ~3.4k câu test ( 300k token).

**Nhận xét:** gán nhãn thủ công, khá sạch nhưng có thể nhiễu nhẹ.

## Hệ thống nhãn BIO trong VLSP2016

Giá trị	Ý nghĩa	Loại thực thể
0	O – Outside	Không phải thực thể
1	B-PER (Begin of Person)	Bắt đầu tên người
2	I-PER (Inside Person)	Bên trong tên người
3	B-ORG (Begin of Org)	Bắt đầu tổ chức
4	I-ORG (Inside Org)	Bên trong tổ chức
5	B-LOC (Begin of Location)	Bắt đầu địa danh
6	I-LOC (Inside Location)	Bên trong địa danh
7	B-MISC (Begin of Misc)	Bắt đầu thực thể khác
8	I-MISC (Inside Misc)	Bên trong thực thể khác

## 3.2. Tiền xử lý dữ liệu

### Quy trình xây dựng bộ ngữ liệu

- Tải dữ liệu **VLSP2016 NER** từ Hugging Face.
- Chuyển về định dạng dòng: **token<space>tag**, dòng trống ngăn cách câu.
- Gán **sentence\_id** cho từng câu.
- Lưu ra **train.txt**, **test.txt**; loại bỏ dòng trống và token lỗi (không UTF-8).

### Quy tắc tiền xử lý

- Giữ nguyên token đã tách sẵn trong dataset.
- Gán nhãn BIO/IOB song song với **sentence\_id**.
- Dataset được chia sẵn theo tỉ lệ xấp xỉ 80% train và 20% test.
- Train set:** 13 486 câu – 294 501 token.
- Test set:** 3 372 câu – 74 899 token.
- Tổng số thực thể (tag ≠ O): **18 901**.

Nội dung văn bản	Ner_tags	Giải thích
[ "Cộng_tác_viên", "của", "Thanh", "ở", "Berlin", "tim", "đến", "khu_vực", "xưa", "Hùng", "cứ_trú", "." ]	[ 0, 0, 1, 0, 5, 0, 0, 0, 0, 1, 0, 0 ]	"Thanh" được gán 1 → B-PER ⇒ bắt đầu một thực thể tên người.  "Berlin" được gán 5 (sau từ "ở") → B-LOC ⇒ địa danh nổi tiếng (thủ đô của Đức).  "Hùng" được gán 1 → B-PER ⇒ bắt đầu một tên người khác (Hùng cư trú tại Berlin).
[ "Người", "Việt", "sinh_sống", "ở", "17", "", "19", "tính", "vìng", "này", "cô", "thời điểm", "lên", "đến", "10", "vạn", "người", "c", "những", "năm", "1946", "", "1953", "y", "." ]	[ 7, 8, 0 ]	"Người" → 7 = B-MISC "Việt" → 8 = I-MISC  → Người Việt là thực thể thuộc nhóm MISC vì không thuộc vào 1 trong 3 loại nhãn PER, ORG, LOC.

### Chú giải

- Mỗi dòng = 1 token + 1 nhãn (định dạng IOB/BIO).
- Dòng trống ngăn cách giữa các câu → gán **sentence\_id**.
- Nhãn B-/I- xác định ranh giới và loại thực thể.
- Ví dụ: "Thanh" → B-PER, "Berlin" → B-LOC, "Người Việt" → B/I-MISC.

### 3.3. Các phương pháp đánh giá

**Token-level Metrics (CoNLL)** – Áp dụng: HMM / CRF / BiLSTM-CRF

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

$$\text{Accuracy} = \frac{\#\text{token đúng}}{\#\text{token}}$$

**Cách tính trung bình** – Áp dụng: CRF / BiLSTM-CRF

- **Weighted:** có xét tần suất mỗi nhãn.
- **ALL:** tính trên toàn bộ nhãn (kể cả 0).
- **Non-O:** chỉ xét token thuộc thực thể.

**Token-level Evaluation (helpers)** – Áp dụng: CRF / BiLSTM-CRF

- **ALL labels:** tính F1 weighted trên toàn bộ nhãn (kể cả 0).
- **Non-O only:** loại bỏ 0, chỉ đánh giá token thuộc thực thể.
- Dùng `classification_report` để xem Precision / Recall / F1 theo nhãn.

**Span-level (Entity-level)** – Áp dụng: CRF / BiLSTM-CRF

- Chuyển chuỗi BIO → danh sách span (`type`, `start`, `end`).
- Đếm `TP` / `FP` / `FN` dựa trên span khớp hoàn toàn.
- Tính Precision / Recall / F1 ở mức **thực thể**.

**Error Analysis** – Áp dụng: CRF / BiLSTM-CRF

- **Confusion pairs:** thống kê cặp (`gold`, `pred`) sai nhiều nhất.
- Phát hiện lỗi phổ biến: ORG ↔ LOC, PER ↔ O.
- **Worst cases:** chọn câu có nhiều lỗi Non-O nhất để phân tích.

**Ý nghĩa của các phương pháp đánh giá**

- **Token-level metrics** phản ánh độ chính xác gần nhau từng token, nhưng dễ bị chi phối bởi nhãn 0.
- **Non-O evaluation** tập trung vào các token thuộc thực thể, giúp đánh giá thực chất khả năng nhận dạng NER.
- **Span-level metrics** do chất lượng nhận dạng **thực thể hoàn chỉnh**, phù hợp với yêu cầu ứng dụng thực tế.
- **Error analysis** hỗ trợ phân tích lỗi phổ biến và định hướng cải tiến mô hình.

## 3.4. Kết quả thực nghiệm – HMM (Test set)

Trước cải thiện (Baseline HMM)

	precision	recall	f1-score	support
B-PER	0.89	0.82	0.85	1502
I-PER	0.73	0.85	0.79	692
B-LOC	0.81	0.61	0.69	1314
I-LOC	0.69	0.48	0.57	594
B-ORG	0.82	0.43	0.57	266
I-ORG	0.76	0.55	0.64	477
B-MISC	0.00	0.00	0.00	54
I-MISC	0.00	0.00	0.00	56
micro avg	0.81	0.66	0.73	4955
macro avg	0.59	0.47	0.51	4955
weighted avg	0.79	0.66	0.71	4955

## Nhận xét:

- Macro F1 thấp (~0.51) ⇒ mô hình yếu ở nhãn hiếm.
- B-MISC, I-MISC có F1 = 0.00 (không nhận diện được).
- Dự đoán bị **chi phối bởi nhãn 0** (mất cân bằng dữ liệu).

Sau cải thiện (Optimized HMM – Test set)

	>>> Evaluating on Test Set...			
	precision	recall	f1-score	support
B-LOC	0.73	0.78	0.76	1314
B-MISC	0.35	0.94	0.52	54
B-ORG	0.53	0.76	0.62	266
B-PER	0.85	0.88	0.87	1502
I-LOC	0.72	0.71	0.71	594
I-MISC	0.35	0.91	0.51	56
I-ORG	0.60	0.67	0.63	477
I-PER	0.79	0.87	0.83	692
O	0.99	0.99	0.99	69944
accuracy				0.97
macro avg	0.66	0.83	0.72	74899
weighted avg	0.98	0.97	0.98	74899

## Cải thiện đạt được:

- Macro F1: 0.51 → 0.72.**
- Nhãn hiếm **B-MISC, I-MISC** đạt F1 ~0.5.
- Recall các nhãn thực thể tăng (ít bỏ sót thực thể hơn).
- Hiệu quả từ: **Balanced Sampling, Class Weighting, Focal Adjustment.**

## Nhận xét &amp; đánh giá

- Tăng Recall ở nhãn hiếm ⇒ đôi khi Precision giảm là đánh đổi hợp lý.
- Tối ưu giúp giảm thiên lệch nhãn 0; tuy nhiên HMM vẫn hạn chế do **Markov bậc 1**.

## Tóm tắt (Test set)

Macro F1: 0.72 | Weighted F1: 0.98 | Accuracy: 0.97

## 3.5. Phân tích lỗi – HMM

### Các lỗi NER phổ biến (Top errors)

```
>>> Top NER Errors:
O -> B-LOC: 280
B-LOC -> O: 168
O -> I-ORG: 164
O -> B-PER: 156
O -> B-ORG: 131
I-ORG -> O: 94
O -> I-LOC: 82
B-PER -> O: 79
O -> B-MISC: 69
O -> I-MISC: 60
```

#### Nhận xét:

- Lỗi phổ biến nhất là **O → B-\* / I-\*** và **B-\* / I-\* → O**.
- Cho thấy mô hình vẫn **thiên lệch về nhãn** O dù đã tối ưu.
- Các thực thể dài (ORG, LOC) dễ bị **dứt chuỗi**.

### Ví dụ câu bị gán nhãn sai

Sentence Bi:	MIDI	TRUE	PRED
lại	0	0	
một	0	0	
tên	0	0	
nửa	0	0	
x	0	0	
chính	0	0	
ba	0	0	
quản_lí	1-ORG	0	Wrong
rừng_phông_hộ	1-ORG	0	Wrong
đầu	1-ORG	0	Wrong
nguồn	1-ORG	0	Wrong
Sérêpôk	1-ORG	0	Wrong
đỗ	0	0	
*	0	0	
thà_cùa	0	0	
*	0	0	
cho	0	0	
sự	0	0	
bình_thành	0	0	
một	0	0	
khu	0	0	
đô_lịch	0	0	
trò_lợi	0	0	
như_thă	0	0	

### Phân tích nguyên nhân:

- Cụm ban quản\_lí rừng\_phông\_hộ đầu nguồn Sérêpôk là một thực thể ORG dài.
- HMM gán đúng B-ORG ở token đầu nhưng **mất toàn bộ chuỗi I-ORG phía sau**.
- Chỉ cần một token có **emission hoặc transition thấp** ⇒ toàn span bị gãy.
- Mô hình **không khai thác được ngữ cảnh dài** để duy trì chuỗi thực thể.

### Kết luận

- Đây là **lỗi mang tính cấu trúc** của HMM. Do giả định **Markov bậc 1** và không có biểu diễn ngữ cảnh dài hạn, HMM dễ làm **dứt chuỗi các thực thể dài** như ORG.
- Các kỹ thuật tối ưu giúp cải thiện **Recall** cho nhãn hiếm, nhưng **không thể khắc phục triệt để** hạn chế này, là động lực để sử dụng các mô hình ngữ cảnh sâu hơn như **CRF** và **BiLSTM+CRF**.

## 3.6. Kết quả thực nghiệm – CRF (Test set)

### Nhận xét tổng quan (Test set)

- Accuracy rất cao: **0.9904**.
- Weighted F1 (ALL): **0.9901** (bao gồm nhãn 0).
- Non-O Weighted F1: **0.9076** ⇒ nhận diện thực thể tốt.
- Span-level F1: **0.9191** (đánh giá ở mức thực thể hoàn chỉnh).

### Ý nghĩa:

- CRF học được **phụ thuộc nhãn toàn chuỗi**, giảm lỗi đứt span.
- Hiệu quả rõ rệt với thực thể dài (ORG, LOC).
- Vượt trội so với HMM trong bài toán NER tiếng Việt.

### So sánh theo đặc trưng (Test set)

Metric	Base	+Lower	+Pre/Suf	+Shape
Accuracy	0.9563	0.9850	0.9898	<b>0.9904</b>
Precision	0.9505	0.9843	0.9894	<b>0.9900</b>
Recall	0.9563	0.9850	0.9898	<b>0.9904</b>
F1-score	0.9484	0.9841	0.9895	<b>0.9901</b>

**Shape features:** is\_upper, is\_title,  
is\_digit

### Kết luận

CRF cho kết quả vượt trội nhờ mô hình hóa **phụ thuộc nhãn toàn chuỗi** và khai thác hiệu quả **đặc trưng hình thái + ngữ pháp**, đặc biệt phù hợp với NER tiếng Việt.

# 3.7. Phân tích lỗi – CRF

## Top confusions (Test set)

Top confusions:

- I-ORG → O : 26
- B-LOC → O : 21
- B-PER → O : 20
- B-ORG → O : 19
- O → I-ORG : 18
- B-LOC → I-LOC : 14
- O → B-LOC : 12
- I-LOC → B-LOC : 11
- O → B-ORG : 10
- B-PER → B-LOC : 10
- B-LOC → B-PER : 9
- I-PER → I-ORG : 8
- I-ORG → B-LOC : 8
- B-ORG → I-ORG : 7
- B-ORG → B-LOC : 7
- I-LOC → O : 7
- O → I-LOC : 7
- I-ORG → I-LOC : 7
- O → B-PER : 6
- I-PER → O : 6

## Nhận xét nhanh:

- Lỗi nổi bật là **I-ORG → O** và **B-ORG → O** ⇒ **dứt span ORG dài** (mất nhãn bên trong thực thể).
- Nhiều lỗi **B-LOC → O** / **I-LOC ↔ B-LOC** ⇒ **sai ranh giới thực thể** (boundary) trong chuỗi LOC.
- Nhầm **PER ↔ LOC** (vd: **B-PER → B-LOC**, **B-LOC → B-PER**) thường đến từ **viết hoa / token dạng tên riêng**.

## Worst case (ví dụ tiêu biểu)

Worst cases:  
Idx=899, non\_0\_errors=14

X_Nhà_n._Đô	gold=B-LOC	pred=O
X_bó_t	gold=I-LOC	pred=O
X_A_n._Nhân_Tây	gold=I-LOC	pred=B-LOC
X_bó_t	gold=I-LOC	pred=O
X_Trung_Hoa	gold=I-LOC	pred=B-LOC
X_bó_t	gold=I-LOC	pred=O
X_Phú_Hòa_Dòng	gold=I-LOC	pred=B-LOC
X_bó_t	gold=I-LOC	pred=O
X_câu	gold=I-LOC	pred=B-LOC
✓_Bến_Hàng	gold=I-LOC	pred=I-LOC
✓_nghé_1_ý	gold=I-LOC	pred=I-LOC
✓_Cù_Chí	gold=I-LOC	pred=I-LOC
X_cần_cứ	gold=I-LOC	pred=O
X_Rồng_Dù	gold=I-LOC	pred=B-LOC
X_sư_đoàn	gold=I-ORG	pred=O
X_dù	gold=I-ORG	pred=O
X_25	gold=I-ORG	pred=O
✓_ngh	gold=I-LOC	pred=B-LOC

## Giải thích nguyên nhân:

- Nhiều token **địa danh** nhiều từ bị gán **B-LOC** rồi các token sau lại rơi về **O** ⇒ **dứt chuỗi LOC** (boundary + transition yếu).
- Các token dạng **số / ký hiệu** (vd: 25, 101, 325) làm nhiều đặc trưng, dẽ kéo nhãn **I-ORG** về **O**.
- Một số cụm có **từ khóa gợi ORG/LOC** (vd: UBND, tinh, Uy...ban) nhưng ngữ cảnh cụ thể không đủ mạnh ⇒ dẽ nhầm **ORG ↔ LOC** hoặc **ORG → O**.

## Thông điệp chính

Các đặc trưng hình thái & ngữ pháp giúp CRF nhận diện tốt hơn, nhưng vẫn khó với **thực thể dài** và **boundary LOC/ORG**.

Để giảm lỗi span dài & boundary, cần **ngữ cảnh mạnh hơn** (BiLSTM/Transformer) hoặc bổ sung feature theo **cụm từ / gazetteer** cho LOC/ORG.

## 3.8. Kết quả thực nghiệm – BiLSTM-CRF (Test set)

### Kết quả trên Test set (BiLSTM-CRF)

[Token] Weighted F1 (ALL incl O): 0.9843

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

B-LOC	0.9092	0.7694	0.8335	1314
B-MISC	0.9259	0.9259	0.9259	54
B-ORG	0.8361	0.5752	0.6815	266
B-PER	0.9342	0.9268	0.9305	1502
I-LOC	0.8933	0.7189	0.7966	594
I-MISC	0.9091	0.8929	0.9009	56
I-ORG	0.8420	0.6143	0.7103	477
I-PER	0.9357	0.8829	0.9086	692
O	0.9896	0.9979	0.9937	69944

accuracy			0.9851	74899
macro avg	0.9083	0.8116	0.8535	74899
weighted avg	0.9843	0.9851	0.9843	74899

[Token] Weighted F1 (Non-O only): 0.8642

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

B-LOC	0.9405	0.7694	0.8464	1314
B-MISC	1.0000	0.9259	0.9615	54
B-ORG	0.9217	0.5752	0.7083	266
B-PER	0.9574	0.9268	0.9418	1502
I-LOC	0.9242	0.7189	0.8087	594
I-MISC	0.9259	0.8929	0.9091	56
I-ORG	0.9099	0.6143	0.7334	477
I-PER	0.9517	0.8829	0.9160	692

micro avg	0.9437	0.8046	0.8686	4955
macro avg	0.9414	0.7883	0.8532	4955
weighted avg	0.9417	0.8046	0.8642	4955

[Span] P=0.9253 R=0.8450 F1=0.8834 | TP=409 FP=33 FN=75

### Tóm tắt chỉ số (Test set)

#### Nhận xét nhanh

- Token-level (ALL) cao do nhãn O chi phối ⇒ **Weighted F1 = 0.9843**.
- Khi chỉ xét Non-O, F1 giảm còn **0.8642** ⇒ khó ở thực thể hiếm/dài.
- Lỗi phô biến: bỏ sót thực thể (dự đoán về O), đặc biệt LOC/ORG.

#### Token-level

- ALL incl O:** Weighted F1 = **0.9843**, Acc = **0.9851**, Macro F1 = **0.8535**
- Non-O only:** Weighted F1 = **0.8642**, Micro F1 = **0.8686**, Macro F1 = **0.8532**

#### Span-level (Entity-level)

P=0.9253 | R=0.8450 | F1=0.8834  
TP=409, FP=33, FN=75

#### Kết luận

BiLSTM-CRF cải thiện biểu diễn ngữ cảnh so với HMM, nhưng trong đồ án này CRF vẫn tốt hơn (Non-O F1 và Span-F1 cao hơn) ⇒ đặc trưng thủ công + CRF phù hợp dữ liệu hiện tại.



# 3.9. Phân tích lỗi – BiLSTM-CRF

## Top confusions (Test set)

Top confusions:

- B-LOC → O : 238
- I-ORG → O : 137
- B-ORG → O : 101
- I-LOC → O : 92
- B-PER → O : 91
- I-PER → O : 63
- O → B-LOC : 37
- O → B-PER : 36
- B-LOC → B-PER : 30
- O → I-ORG : 26
- I-LOC → B-LOC : 24
- I-ORG → B-LOC : 20
- I-LOC → I-PER : 20
- O → B-ORG : 17
- O → I-LOC : 16
- B-LOC → I-LOC : 16
- I-LOC → I-ORG : 15
- I-ORG → I-LOC : 14
- I-LOC → B-PER : 14
- I-PER → B-PER : 11

### Nhận xét:

- Lỗi lớn nhất: **B-LOC** → **O** (238), **I-ORG** → **O** (137).
- Mô hình thường bỏ sót thực thể (FN tăng) ⇒ Recall span giảm.
- Có hiện tượng nhầm ranh giới BIO: **I-LOC ↔ B-LOC, B-LOC → B-PER.**

## Ý nghĩa

Sai lệch chủ yếu đến từ **entity dài và tên riêng/viết tắt/ngoài ngữ** khiến emission không ổn định ⇒ dễ rơi về nhãn **O**.  
 ⇒ **Kết luận:** BiLSTM-CRF học ngữ cảnh tốt hơn HMM, nhưng vẫn cần **feature/embedding mạnh hơn hoặc pretrained encoder** (PhoBERT/XLM-R) để xử lý entity dài và tên riêng đa dạng.

TS. Nguyễn Thị Quý

CS221.Q12 - Xử lý Ngôn ngữ Tự nhiên

## Ví dụ worst case (Test set)

Worst cases:  
 idx=2662 nonO\_errors=14

✓ Jane	gold=B-PER	pred=B-PER
✓ Barton	gold=I-PER	pred=I-PER
X quán_lý	gold=O	pred=B-ORG
X Bệnh_viện	gold=B-LOC	pred=I-ORG
X Washington	gold=I-LOC	pred=I-ORG
X ;	gold=O	pred=I-ORG
X Sophie	gold=B-PER	pred=I-ORG
X Quinn	gold=I-PER	pred=I-ORG
✓ VN	gold=B-LOC	pred=B-LOC
✓ Trung_tâm	gold=B-ORG	pred=B-ORG
✓ Nghiên_cứu	gold=I-ORG	pred=I-ORG
✓ vē	gold=I-ORG	pred=I-ORG
✓ triết_học	gold=I-ORG	pred=I-ORG
X ,	gold=I-ORG	pred=O
X văn_hoá	gold=I-ORG	pred=O
X và	gold=I-ORG	pred=O
X xã_hội	gold=I-ORG	pred=O
X VN	gold=B-LOC	pred=B-LOC
X Trường	gold=B-ORG	pred=B-LOC
X ĐH	gold=I-ORG	pred=I-LOC
X Temple	gold=I-ORG	pred=O
✓ Nguyễn	gold=B-PER	pred=B-PER
✓ Thị	gold=I-PER	pred=I-PER
✓ Mai	gold=I-PER	pred=I-PER
✓ Quảng_Nam	gold=B-LOC	pred=B-LOC

### Giải thích nguyên nhân:

- Câu chứa nhiều thực thể dài/dan xen (ORG-LOC-PER) ⇒ khó giữ ranh giới span.
- Một vài token “nhiều” (dầu câu, viết tắt như ĐH, VN, tên ngoại như **Temple**) làm CRF layer khó nối chuỗi **I-\***.
- Khi một token bị gán **O**, cả thực thể dài có thể bị **đứt chuỗi** ⇒ giảm F1 span.

## 3.10. So sánh tổng quan các mô hình

### Nhận xét so sánh mô hình

- CRF tốt nhất (Test): Acc=0.9904, Span-F1=0.9191, Non-O F1=0.9076.
- BiLSTM+CRF đứng sau: Acc=0.9851, Span-F1=0.8834, Non-O F1=0.8642.
- HMM (tối ưu): Macro-F1 0.51→0.72, nhưng vẫn kém do hạn chế Markov.
- Token F1 (ALL) cao vì 0 nhiều ⇒ ưu tiên Non-O F1 & Span-F1.

### Bảng so sánh hiệu suất (Test set)

Chỉ số	HMM	CRF	BiLSTM
Accuracy	0.97	<b>0.9904</b>	0.9851
Token F1 (ALL incl O)	0.98	<b>0.9901</b>	0.9843
Token F1 (Non-O only)	–	<b>0.9076</b>	0.8642
Macro F1 (Token)	<b>0.72</b>	0.8875	0.8535
Span F1 (Entity-level)	–	<b>0.9191</b>	0.8834

Ghi chú: HMM có Macro/Weighted/Acc;  
CRF/BiLSTM-CRF có thêm Non-O & Span-F1.

### Kết luận ngắn

- CRF phù hợp nhất với dữ liệu hiện tại (feature thủ công + ràng buộc chuỗi).
- BiLSTM+CRF sẽ mạnh hơn nếu có data lớn hơn / embedding pretrained.

### Ý nghĩa chỉ số

- Token F1 (ALL): dễ “ảo” vì 0 nhiều.
- Non-O F1: chất lượng nhận diện thực thể (bỏ 0).
- Span F1: đúng thực thể hoàn chỉnh (khắt khe nhất).

# Mục lục

- 1 Tổng quan đề tài
- 2 Tổng quan các phương pháp
- 3 Thực nghiệm
- 4 Kết luận & Hướng phát triển

# Kết luận & Hướng phát triển

## Kết luận từ thực nghiệm

- Ba mô hình **HMM, CRF và BiLSTM+CRF** đều áp dụng được cho bài toán **Named Entity Recognition (NER)** tiếng Việt trên bộ dữ liệu **VLSPL 2016**.
- **CRF** cho kết quả tốt nhất trên Test set: **Accuracy = 0.9904**, **Non-O F1 = 0.9076**, **Span-F1 = 0.9191**, cho thấy khả năng nhận diện thực thể ổn định và chính xác ở mức entity hoàn chỉnh.
- Kết quả này khẳng định **đặc trưng hình thái & ngữ pháp** (lowercase, prefix/suffix, word shape, is\_upper, is\_title, is\_digit) vẫn **rất hiệu quả với tiếng Việt** khi kết hợp cùng CRF.
- **BiLSTM+CRF** đạt kết quả tốt nhưng thấp hơn CRF (**Non-O F1 = 0.8642**, **Span-F1 = 0.8834**), nguyên nhân chủ yếu do **dữ liệu huấn luyện chưa đủ lớn và chưa sử dụng embedding pretrained**.
- **HMM (tối ưu)** cải thiện đáng kể Macro-F1 (**0.51 → 0.72**), nhưng vẫn bị giới hạn bởi **giả định Markov bậc 1** và không mô hình hóa được ngữ cảnh dài.

## Hướng phát triển

### Ngắn hạn:

- Bổ sung **gazetteer / từ điển tên riêng** cho CRF để cải thiện nhận diện **thực thể hiếm và thực thể dài**.
- Kết hợp **đặc trưng ký tự** (character-level) thông qua các mô hình **BiLSTM-CNN-CRF**.

### Dài hạn:

- Fine-tune các mô hình ngôn ngữ pretrained cho tiếng Việt như **PhoBERT, XLM-R** kết hợp với CRF/Viterbi ở đầu ra.
- Khai thác các hướng **few-shot / zero-shot NER** với các mô hình mới như **GLiNER (2024)** để giảm phụ thuộc vào dữ liệu gán nhãn.
- Đóng gói hệ thống thành **pipeline suy diễn hoặc API** phục vụ các ứng dụng thực tế như chatbot, tìm kiếm, trích xuất thông tin.



# Tài liệu tham khảo

-  VLSPL 2016. *Vietnamese Named Entity Recognition Shared Task Dataset*. (HuggingFace).
-  Ngoc C. Le, Ngoc-Yen Nguyen, Anh-Duong Trinh. *On the Vietnamese Named Entity Recognition: A Deep Learning Method Approach*. In: IEEE RIVF, 2020.  
<https://arxiv.org/pdf/1912.01109>
-  Daniel Jurafsky, James H. Martin. *Speech and Language Processing (3rd ed. draft)*, 2023.  
[https://web.stanford.edu/~jurafsky/slp3/ed3book\\_aug25.pdf](https://web.stanford.edu/~jurafsky/slp3/ed3book_aug25.pdf)

# Cảm ơn!

Xin cảm ơn cô Nguyễn Thị Quý và các bạn!

Hỏi đáp — Demo