



**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

XỬ LÝ NGÔN NGỮ TỰ NHIÊN – CS221.Q12

Learned Incremental Representations for Parsing

GVHD :
TS. Nguyễn Thị Quý

Nhóm sinh viên :
Trần Ngọc Cẩm Nguyên - 23521061
Ngô Phạm Phương Uyên - 23521762
Dương Tấn Lộc - 23520854

Ngày 25 tháng 12 năm 2025

1 Giới thiệu

Trong lĩnh vực xử lý ngôn ngữ tự nhiên, **constituency parsing** (phân tích thành phần cú pháp) đóng vai trò quan trọng trong việc xác định cấu trúc phân cấp của câu. Tuy nhiên, các mô hình đạt hiệu suất cao nhất hiện nay thường yêu cầu toàn bộ câu đầu vào và sử dụng cơ chế xử lý hai chiều (bidirectional) phức tạp trước khi đưa ra kết quả. Cách tiếp cận này khác biệt hoàn toàn với khả năng của con người—vốn có thể hiểu và xử lý ngôn ngữ theo thời gian thực ngay khi từng từ được phát âm.

Phương pháp từ [1] sử dụng tập trung vào tính tăng dần trong phân tích cú pháp với hai lý do chính:

- Mô phỏng nhận thức con người: Thay vì chờ đợi câu kết thúc, hệ thống thực hiện gán một nhãn rời rạc duy nhất cho mỗi từ ngay khi nó xuất hiện (dựa trên tiền tố của câu).
- Loại bỏ phân tích giả định (Speculation-free): Thay vì dùng beam search để giữ nhiều kết quả dự phòng và loại bỏ sau này, phương pháp này chỉ đưa ra quyết định cú pháp khi thông tin đã rõ ràng qua các từ đã đọc.

Báo cáo này phân tích lại cách phương pháp từ paper [1] xây dựng cây cú pháp và đánh giá trên bộ dữ liệu WSJ.

2 Hướng dẫn cài đặt

Clone repo: locluclak/Demo-Incremental-Representation-Parsing

Các yêu cầu hệ thống:

- Python: Phiên bản 3.7 trở lên.
- PyTorch: Phiên bản 1.6.0 hoặc các phiên bản mới hơn có tương thích.
- Thư viện Benepar: Cần cài đặt đầy đủ các phụ thuộc: NLTK, torch-struct, transformers
- pytokenizations: Phiên bản 0.7.2 hoặc tương thích.
- clusopt: chạy bằng Rust nên cài Rust trước nếu chưa có
- torch_struct
- streamlit
- Khuyên dùng hệ điều hành Linux để tương thích với thư viện, tránh lỗi.

Tải model tại link, đặt trong thư mục của repo.

Cuối cùng, chạy lệnh để mở giao diện:

```
streamlit run app.py
```

Dưới đây là lệnh thực hiện đánh giá trên Recall, Precision, F1-score và CompleteMatch:

```
python ../src/main.py test \
--model-path ../32_model \
--test-path ../data/23.auto.clean \
--subbatch-max-tokens 750 \
--evalb-dir ../EVALB/
```

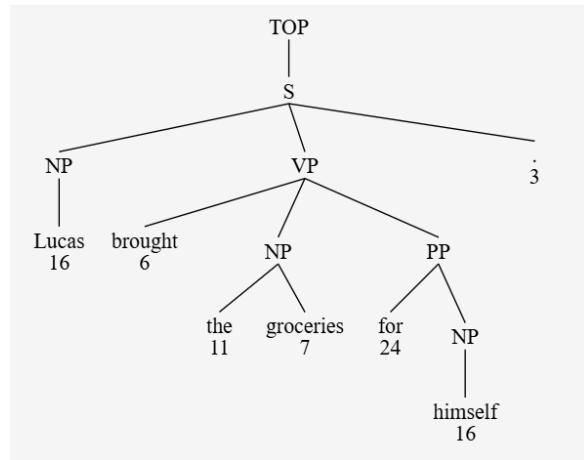
3 Mô hình sử dụng

Model: Được tác giả train và công bố với kích bottleneck là 32 tags.

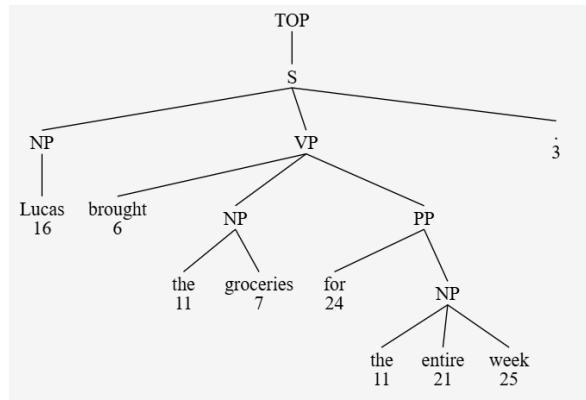
Với cấu trúc chia làm 2 phần:

1. **Encoder** (Bộ mã hóa tăng tiến): Sử dụng mô hình ngôn ngữ GPT-2 để xử lý văn bản theo chiều từ trái sang phải. Do đặc tính tự hồi quy (autoregressive), Encoder chỉ tiếp nhận thông tin từ các từ đã xuất hiện trong quá khứ và từ hiện tại mà không thể nhìn trước tương lai. Trạng thái ẩn (hidden state) mỗi từ đi qua một lớp Information Bottleneck để rời rạc hóa thành một discrete tag duy nhất thuộc một bảng mã có kích thước cố định (32). Quá trình này ép mô hình phải nén toàn bộ thông tin cú pháp cần thiết của từ đó vào một giá trị số rời rạc.
2. **Read-out Network** (Mạng giải mã cấu trúc): Thành phần này chịu trách nhiệm tái tạo cây phân tích cú pháp từ chuỗi tags rời rạc đã được tạo ra. Nó sử dụng mạng Transformer + MLP để tính toán *score* cho các tag. Cuối cùng, Span-Based CKY được áp dụng để tìm ra cây cú pháp có tổng điểm cao nhất.

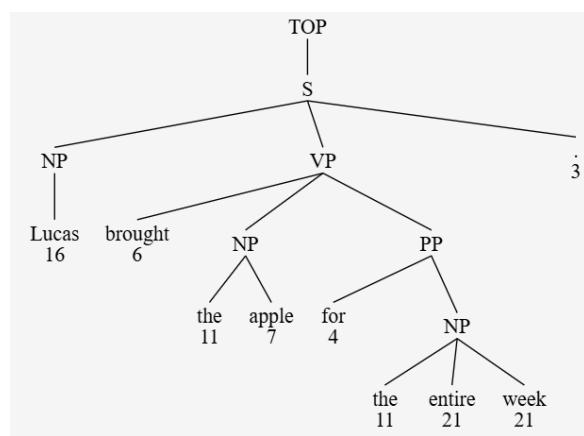
4 Cách mô hình xây dựng cây cú pháp



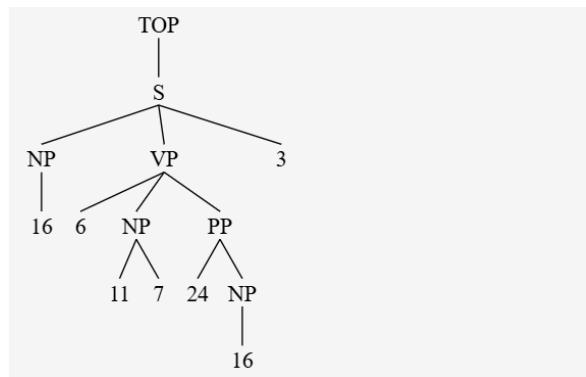
(a) Lucas brought the groceries for himself .



(b) Lucas brought the groceries for the entire week .



(c) Lucas brought the apple for the entire week .



(d) Cây cú pháp tạo từ tag [16, 6, 11, 7, 24, 16, 3]

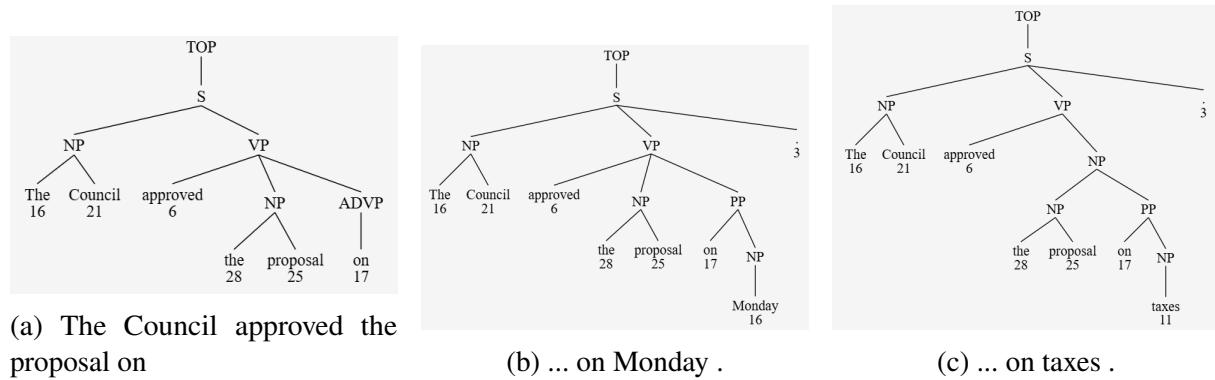
Hình 1: Quá trình xây dựng cây cú pháp dựa trên các tag được nhúng.

Quan sát các hình 1a, 1b và 1c, ta rút ra các nhận định:

- Trong Hình 1a và 1b, nhãn (tag symbol) của từ “*for*” không thay đổi (giá trị bằng 24) ngay cả khi về sau của câu thay đổi.
- Tuy nhiên, khi từ ngữ phía trước thay đổi (từ “*groceries*” thành “*apple*” trong Hình 1c), nhãn của từ “*for*” chuyển sang giá trị 4.

⇒ Điều này chứng minh nhãn của mỗi từ chỉ phụ thuộc vào thông tin từ các từ phía trước (tiền tố) mà không bị ảnh hưởng bởi các từ phía sau, khẳng định tính tăng dần nghiêm ngặt của mô hình.

4.1 Ví dụ 2



Hình 2: Sự thay đổi cấu trúc cây dựa trên thông tin tăng dần.

Ví dụ 2 phân tích trường hợp nhập nhằng (ambiguity) khi câu kết thúc bằng giới từ “*on*”. Tùy thuộc vào từ ngữ xuất hiện sau đó, cấu trúc cây sẽ được điều chỉnh linh hoạt:

- Nếu là “*Monday*” (Hình 2b), cụm giới từ đóng vai trò trạng ngữ chỉ thời gian.
- Nếu là “*taxes*” (Hình 2c), cụm giới từ lại bổ nghĩa trực tiếp cho danh từ “*proposal*”.

⇒ Qua đó minh họa khả năng của mô hình: dù chỉ duy trì một chuỗi nhãn duy nhất cho mỗi từ theo thời gian thực, hệ thống vẫn đảm bảo tính linh hoạt để tái cấu trúc cây chính xác theo ngữ cảnh xuất hiện sau đó.

5 Đánh giá

5.1 Bộ dữ liệu

Mô hình được đánh giá trên tập **Section 23** thuộc bộ dữ liệu **Penn Treebank (WSJ)**. Dữ liệu đầu vào bao gồm **2.416** mẫu đã được xử lý dưới dạng cây chuẩn (gold tree).

5.2 Kết quả và Phân tích

Kết quả thực nghiệm trên tập kiểm thử (WSJ Section 23) được tổng hợp trong Bảng 1.

Nhận xét:

Bảng 1: Hiệu suất mô hình trên tập kiểm thử WSJ Section 23.

Bottleneck	Recall	Precision	F1-Score	Complete Match
32 tags	93,09%	94,37%	93,72%	45,36%
256 tags	94,65%	95,30%	94,97%	51,32%

- Khả năng nén mạnh mẽ: Mặc dù chỉ sử dụng bảng mã tối giản 32 nhãn, mô hình vẫn đạt F1 ấn tượng (93,72%). Điều này chứng minh các cấu trúc cú pháp phức tạp có thể được nén gọn vào không gian nhãn rời rạc mà không gây mất mát thông tin đáng kể.
- Hiệu suất tối ưu: Khi mở rộng lên 256 nhãn, các chỉ số đều tăng trưởng ổn định, đặc biệt là tỷ lệ Complete Match hơn 50% khớp hoàn toàn gold tree.
- Độ tin cậy: Chỉ số Precision ở cả hai cấu hình đều cao hơn Recall, cho thấy mô hình rất chính xác trong việc xác định các phân đoạn cú pháp và hạn chế sai lệch ngữ pháp.

6 Kết luận

Phương pháp biểu diễn tăng dần chứng minh hiệu quả trong việc nén cấu trúc cú pháp phức tạp vào hệ thống nhãn rời rạc tối giản mà vẫn duy trì độ chính xác cao. Việc loại bỏ các phân tích giả định không chỉ tối ưu hóa hiệu suất tính toán mà còn mô phỏng sát thực khả năng xử lý ngôn ngữ theo thời gian thực của con người. Đây là nền tảng tiềm năng cho các ứng dụng phân tích cú pháp trực tiếp và gọn nhẹ trong tương lai.

Tài liệu

- [1] Nikita Kitaev, Thomas Lu, and Dan Klein. Learned incremental representations for parsing. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3086–3095, Dublin, Ireland, May 2022. Association for Computational Linguistics.