PART OF SPEECH

1. Ý nghĩa:

* Là bước tiếp theo của Tokenize
* Được sử dụng để gán nhãn các từ theo mức độ kiểu từ vựng (động từ, tính từ, danh từ,…)
* Tăng tính ngữ nghĩa của feature -> feature mang tính đặc trưng cao hơn
* Sử dụng các nhãn từ vựng của Penn Treebank
* Các phương pháp gán nhãn được giới thiệu là: Hidden Markov Model (HMM) and the Maximum Entropy Markov Model (MEMM).
* Làm tiền đề cho việc xác định các thực thể có nghĩa (NER)

Khó khăn:

* Mơ hồ, không rõ ràng: Một từ có thể thuộc nhiều loại từ và mang nghĩa khác nhau.

Ví dụ:

+ Book:

* Danh từ: quyển sách
* Động từ: đặt phòng/ lịch…

Độ chính xác phụ thuộc vào hai yếu tố:

* Tần suất xuất hiện loại từ của một từ nào đó.

Ví dụ: act có 2 loại từ là động từ và danh từ, nhưng động từ thường được sử dụng nhiều hơn danh từ.

* Ngữ cảnh của câu:

Ví dụ:

+ I have read a book

* Sau mạo từ thì book luôn được sử dụng như danh từ

+ I will book a meeting for us

* Sau modal verb thì book chắc chắn sẽ phải là động từ

1. Các hàm sử dụng, input và output:

* Hàm khởi tạo MaxentTagger: class này sử dụng để chạy, train và kiểm tra việc dán nhãn POS. Có 2 kiểu dán nhãn Tiếng Anh là:

+ bi-directional dependency network tagger (gán nhãn mạng lưới phụ thuộc hai chiều): sử dụng file english-bidirectional-distsim.tagger. Độ chính xác là 97.32%.

+ Gán nhãn sử dụng chuỗi thông tin một chiều từ trái sang nhưng gán chính xác ít từ vô danh hơn cũng như ít các đặc trưng từ vựng hơn, sử dụng file english-left3words-distsim.tagger. Độ chính xác là 96.92%, bù lại kiểu dãn nhãn này nhanh hơn hẳn, và được khuyến khích sử dụng.

+ Input: đường dẫn đến tập tin model

* Tokenize: Tạo thành một mảng các từ riêng biệt từ một đoạn văn bản

+ Input: tham số kiểu Reader (thư viện java.io)

* Có thể chuyển từ kiểu string ở c# sang kiểu StringReader bằng hàm khởi tạo StringReader

+ Output:

* TagSentence

1. Thuật toán sử dụng

* Sử dụng thuật toán MEMM cải tiến (đa chiều)