LỜI CẢM ƠN

Em xin chân thành cảm ơn các thầy, các cô khoa Công nghệ Thông tin Trường Đại học Dân lập Hải Phòng đã tận tình dạy dỗ, truyền đạt cho chúng em những kiến thức quý báu trong suốt bốn năm học vừa qua.

Em xin tỏ lòng biết ơn sâu sắc đến Ths Nguyễn Thị Xuân Hương, người đã tận tình giúp đỡ và truyền đạt nhiều kinh nghiệm để đề tài có thể được thực hiện và hoàn thành.

Cuối cùng em xin gửi lời cảm ơn chân thành tới tất cả người thân và bạn bè đã giúp đỡ, động viên em rất nhiều trong quá trình học tập cũng như thực hiện đề tài.

Em xin trân thành cảm ơn!

Hải Phòng, tháng 07 năm 2011.

Sinh viên

Lưu Văn Sơn

MỤC LỤC

DANH MỤC CHỮ VIẾT TẮT

- LM: Mô hình ngôn ngữ
- MKN: Phương pháp làm mịn Kneser-Ney cải tiến
- MLE: Ước lượng cực đại hóa khả năng
- MSE: Lỗi trung bình bình phương
- MT: Dịch máy
- SMT: Dịch máy bằng phương pháp thống kê
-

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

MỞ ĐẦU

Mô hình ngôn ngữ là một thành phần quan trọng trong các ứng dụng như nhận dạng tiếng nói, phân đoạn từ, dịch thống kê, … Và chúng thường được mô hình hóa sử dụng các n-gram. Trên thế giới đã có rất nhiều nước công bố nghiên cứu về mô hình ngôn ngữ áp dụng cho ngôn ngữ của họ nhưng ở Việt Nam, việc nghiên cứu và xây dựng một mô hình ngôn ngữ chuẩn cho tiếng Việt vẫn còn mới mẻ và gặp nhiều khó khăn. Chính điều này đã gợi ý và thúc đẩy chúng tôi lựa chọn và tập trung nghiên cứu vấn đề này để có thể tạo điều kiện cho việc xử lý ngôn ngữ tiếng Việt vốn vô cùng phong phú của chúng ta.

Đồ án gồm có 3 chương:

Chương 1: Tổng quan về mô hình ngôn ngữ: trình bày khái quát lý thuyết về mô hình ngôn ngữ, các khó khăn còn tồn tại và phương pháp khắc phục, trong đó trọng tâm nghiên cứu các phương pháp làm mịn,

Chương 2: Ứng dụng của mô hình ngôn ngữ trong dịch máy thống kê, là các giới thiệu về hệ dịch máy thống kê và ứng dụng của mô hình ngôn ngữ trong đó,

Chương 3 là kết quả thực nghiệm khi cài đặt và sử dụng bộ công cụ mã nguồn mở SRILIM để xây dựng mô hình ngôn ngữ cho tiếng Việt và MOSES để dịch máy thống kê,

Cuối cùng là phần kết luận.

CHƯƠNG 1: TỔNG QUAN VỀ MÔ HÌNH NGÔN NGỮ

Mô hình ngôn ngữ (Language Model - LM) là các phân phối xác suất trên một ngữ liệu đơn ngữ, được sử dụng trong nhiều bài toán khác nhau của xử lý ngôn ngữ tự nhiên, ví dụ như: dịch máy bằng phương pháp thống kê, nhận dạng giọng nói, nhận dạng chữ viết tay, sửa lỗi chính tả, …. Thực chất, LM là một hàm chức năng có đầu vào là một chuỗi các từ và đầu ra là điểm đánh giá xác suất một người bản ngữ có thể nói chuỗi đó. Chính vì vậy, một mô hình ngôn ngữ tốt sẽ đánh giá các câu đúng ngữ pháp, trôi chảy cao hơn một chuỗi các từ có thứ tự ngẫu nhiên, như trong ví dụ sau:

P("hôm nay trời nắng") > P("trời nắng nay hôm")

1.1 N-gram

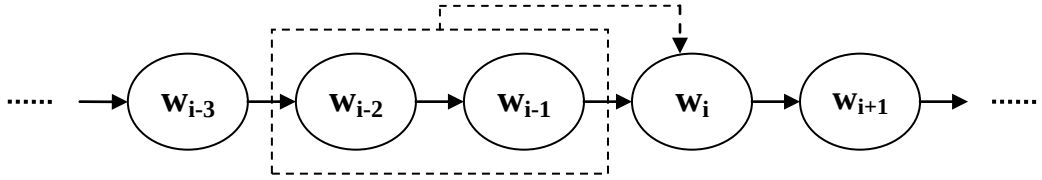
Cách thông dụng nhất được dùng để mô hình hóa ngôn ngữ vào trong LM là thông qua các n-gram. Với mô hình n-gram, chúng ta coi một văn bản, đoạn văn bản là chuỗi các từ liền kề nhau, w_1, w_2, …, w_{n-1}, w_n, và sau đó phân tích xác suất của chuỗi với công thức xác suất kết hợp:

$$P(w_1 w_2 \ldots w_m) = P(w_1) * P(w_2|w_1) * P(w_3|w_1 w_2) * \ldots * P(w_m|w_1 w_2 \ldots w_{m-1})$$

và do vậy mỗi từ sẽ liên quan có điều kiện tới toàn bộ các từ trước nó (ta sẽ gọi đây là lịch sử của sự kiện hoặc từ đó).

Tuy nhiên, việc sử dụng toàn bộ các từ trước đó để đoán nhận từ tiếp theo là không thể thực hiện được vì hai nguyên nhân sau. Đầu tiên là phương pháp này không khả thi về mặt tính toán do tốn quá nhiều thời gian, tài nguyên hệ thống cho mỗi lần dự đoán. Hai là, trong rất nhiều trường hợp, chỉ sau khi duyệt vài từ trong lịch sử, ta đã nhận thấy rằng đó là một câu chưa từng gặp trước đây. Bởi vậy kể cả khi đã biết toàn bộ lịch sử của một từ, xác suất của nó vẫn có thể là không biết. Thay vào đó, các mô hình ngôn ngữ thường ước lượng tương đối xác suất dựa trên giả định Markov (hay mô hình Markov ẩn), rằng từ tiếp theo chỉ chịu ảnh hưởng từ một vài từ trước đó. Một mô hình Markov bậc n giả định rằng chỉ n từ trước đó có liên hệ ngữ cảnh với từ đang cần xác định. Việc quyết định bao nhiêu từ trước đó mà LM quan tâm được gọi là bậc n (order) của LM, và thường được gọi là 1-gram (unigram), 2-gram (bigram), 3-gram (trigram), 4-gram (fourgram) tương ứng với các mô hình Markov bậc một, hai, ba, bốn.

Ví dụ, nếu chúng ta muốn ước lượng xác suất 2-gram của một từ wi với mô hình Markov bậc 2 thì chúng ta sẽ dựa trên hai từ trước đó:$P(w_1,w_2,\ldots,w_i)=P(w_i/w_{i-2},w_{i-1})$



Hình 1.1: Mô hình Markov bậc 2

Một cách tổng quát, xác suất xuất hiện của một từ (w_m) được coi như chỉ phụ thuộc vào n từ đứng liền trước nó ($w_{m-n}w_{m-n+1}\ldots w_{m-1}$) chứ không phải phụ thuộc vào toàn bộ dãy từ đứng trước ($w_1w_2\ldots w_{m-1}$). Như vậy, công thức tính xác suất văn bản được tính lại theo công thức:

$P(w_1w_2\ldots w_m) = P(w_1) * P(w_2|w_1) * P(w_3|w_1w_2) *\ldots* P(w_{m-1}|w_{m-n-1}w_{m-n}\ldots w_{m-2})* P(w_m|w_{m-n}w_{m-n+1}\ldots w_{m-1})$

1.2 Xây dựng mô hình ngôn ngữ

Để xây dựng (huấn luyện) một mô hình ngôn ngữ ta cần một ngữ liệu đơn ngữ (corpus) có kích thước tương đối và một bộ ước lượng thống kê có nhiệm vụ mô hình hóa lượng xác suất của ngữ liệu. Các bộ ước lượng được mà LM sử dụng, theo những cách khác nhau, đều cần đến tần suất của các n-gram, do đó chúng ta cần phải đếm số lần xuất hiện của các n-gram từ 1-gram cho đến số bậc mô hình chúng ta đang huấn luyện.

1.2.1 Ước lượng cực đại hóa khả năng (MLE)

Chúng ta có thể sử dụng kết quả đếm các n-gram để xây dựng một mô hình ước lượng cực đại hóa khả năng (Maximium Likelihood Estimation - MLE) với tần suất tương đối của các n-gram trong ngữ liệu. Với MLE, xác suất một unigram nhất định nào đó sẽ xuất hiện tiếp theo đơn giản là tần suất nó xuất hiện trong ngữ liệu.

$$P_{MLE}(w_i) = \frac{c(w_i)}{\sum_{i'} c(w_{i'})}$$

trong đó $c(w_{i'}) = |w_{i'}|$ chính là số lần xuất hiện của từ $w_{i'}$ trong ngữ liệu. Phương pháp này được gọi như vậy bởi vì nó cực đại hóa giá trị đầu ra để mô hình hóa ngữ liệu huấn luyện. Ví dụ, trong ngữ liệu Brown, một ngữ liệu với một triệu

từ, từ khóa "Chinese" xuất hiện 400 lần. Vậy thì xác suất mà một mô hình ngôn ngữ dùng MLE sẽ gán cho unigram "Chinese" là $P_{MLE}(Chinese) = \frac{400}{1000000} = .0004$.

Xác suất điều kiện của một n-gram tổng quát với bậc > 1 là:

$$P_{MLE}(w_i \mid w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})}$$

tức là tần suất một từ nào đó thường xuyên xuất hiện sau lịch sử có bậc n-1. Để minh họa, ta tiếp tục ví dụ trên, xác suất bigram "Chinese food" xuất hiện là số lần từ "food" xuất hiện sau từ "Chinese" chia cho c(Chinese) = 400. Trong ngữ liệu Brown, cụm từ "Chinese food" xuất hiện 120 lần, nên: P_{MLE}(food|Chinese) = 0.3

1.2.2 Các phương pháp làm mịn

Tuy MLE là một phương pháp dễ hiểu, dễ sử dụng để ước lượng xác suất cho mô hình, nhưng trong thực tế ta gặp phải vấn đề dữ liệu thưa (data sparseness problem). Tức là tập ngữ liệu dùng để xây dựng LM dù lớn đến mấy, cũng chỉ là tập hữu hạn các câu trong vô số câu có thể của một ngôn ngữ tự nhiên. Do đó một LM chỉ sử dụng MLE sẽ gán xác suất bằng 0 cho nhiều n-gram tốt. Để giảm thiểu vấn đề này, người ta thường không sử dụng MLE mà thay vào đó là các phương pháp ước lượng xác suất thống kê phức tạp hơn. Các phương pháp này được gọi là làm mịn (smoothing) hay trừ hao (discounting), khi mà một phần xác suất từ các sự kiện trong mô hình sẽ được dành cho những sự kiện chưa từng xuất hiện. Việc lấy từ cái gì và trừ hao như thế nào là một đề tài vẫn đang được nghiên cứu nhiều. Ví dụ, cách cổ điển nhất của làm mịn là phương pháp Add-one smoothing, trong phương pháp này, ta thêm một lượng $l \leq 1$ vào kết quả đếm số lần xuất hiện của mọi từ vựng trong ngữ liệu.

Hai khái niệm quan trọng được sử dụng trong quá trình làm mịn các mô hình ngôn ngữ là backoff và interpolation. Khi LM gặp một n-gram chưa biết, việc tính xác suất sẽ sử dụng thông tin từ (n-1)-gram, nếu sự kiện (n-1)-gram cũng chưa từng xuất hiện trong quá trình huấn luyện thì LM lại sử dụng thông tin xác suất từ (n-2)-gram, … Và cứ tiếp tục như vậy cho đến khi tính được xác suất của n-gram. Quá trình này được gọi là backoff và được định nghĩa như sau:

$$P_{BO}(w_i \mid w_{i-n+1}^{i-1}) = \begin{cases} \delta(w_{i-n+1}^{i-1}) \Pr_{LM}(w_i \mid w_{i-n+1}^{i-1}) & \text{if } c(w_{i-n+1}^{i-1}) > 0 \\ \alpha \Pr_{BO}(w_{i-n+2}^{i}) & \text{otherwise.} \end{cases}$$

Trong đó δ là hệ số trừ hao dựa trên tần suất xuất hiện của w_{i-n+1}^{i-1} trong lịch sử và α là tham số backoff. Khi số lượng từ vựng đủ lớn, chúng ta có thể sẽ cần gán xác suất bằng 0 cho một số từ ngoài từ điển (out of vocabulary - OOV) khi ở mức unigram. Chẳng hạn khi ta có một cuốn từ điển chuyên ngành và không muốn chia sẻ lượng xác suất của các từ vựng đó (các danh từ chung, các số thực đặc biệt, …) cho các OOV. Một cách khác là chúng ta làm mịn LM và dành một lượng xác suất nhỏ gán cho các từ ngoài từ điển khi ở mức unigram.

Phương pháp Interpolation kết hợp thông tin thống kê n-gram qua tất cả các bậc của LM. Nếu bậc của LM là n thì công thức đệ quy interpolation như sau:

$$P(w_i|w_{i-n+1}...w_{i-1}) = \lambda P(w_i|w_{i-n+1}...w_{i-1}) + (1-\lambda)P(w_i|w_{i-n+2}...w_{i-1})$$

Trong đó λ là trọng số quyết định bậc nào của LM có ảnh hưởng lớn nhất đến giá trị đầu ra. Tổng trọng số λ được sử dụng cho tất cả các bậc n-gram bằng một. Có nhiều cách để xác định giá trị cho các trọng số λ này, đối với phương pháp interpolation đơn giản thì các giá trị λ này giảm theo số bậc n-gram. Tuy nhiên thường thì chúng sẽ được tính toán tùy theo điều kiện ngữ cảnh cụ thể, tức là theo tần suất của các bậc n-gram trong lịch sử. Các trọng số này không được tính toán từ dữ liệu huấn luyện, mà sử dụng tập dữ liệu held-out riêng biệt – tập này chỉ được dùng để huấn luyện các tham số, mà trong trường hợp này là các giá trị λ. Cần phải nhận thấy rằng sự khác biệt cơ bản giữa hai phương pháp này là interpolation sử dụng thông tin từ các bậc thấp hơn ngay cả khi dữ liệu xác suất của n-gram cần tính đã khác 0; trong khi backoff thì lại chỉ tìm kiếm đến dữ liệu khác 0 gần nhất.

Những tiểu mục tiếp theo trong phần này sẽ trình bày về một số phương pháp làm mịn phổ biến nhất hiện nay.

- Chiết khấu (Discounting): giảm (lượng nhỏ) xác suất của các cụm Ngram có xác suất lớn hơn 0 để bù cho các cụm Ngram không xuất hiện trong tập huấn luyện.
- Truy hồi (Back-off) : tính toán xác suất các cụm Ngram không xuất hiện trong tập huấn luyện dựa vào các cụm Ngram ngắn hơn có xác suất lớn hơn 0
- Nội suy (Interpolation): tính toán xác suất của tất cả các cụm Ngram dựa vào xác suất của các cụm Ngram ngắn hơn.

1.2.2.1 Các thuật toán chiết khấu (Discounting)

Nguyên lý của các thuật toán chiết khấu là giảm xác suất của các cụm Ngram có xác suất lớn hơn 0 để bù cho các cụm Ngram chưa từng xuất hiện trong tập huấn luyện. Các thuật toán này sẽ trực tiếp làm thay đổi tần số xuất hiện của tất cả các cụm Ngram. Ở đây đề cập đến 3 thuật toán chiết khấu phổ biến:

- Thuật toán Add-one

Phương pháp làm mịn add-one cộng thêm 1 vào tần số xuất hiện của tất cả các cụm N-gram rồi nhân với phân số chuẩn hóa (để bảo toàn tổng xác suất).

$$P(w_i|w_{i-n+1}...w_{i-1}) = \frac{C(w_{i-n+1}...w_{i-1}w_i) + 1}{C(w_{i-n+1}...w_{i-1}) + V}$$

Trong đó V là kích thước bộ từ vựng

Chúng ta có thể thấy thuật toán này sẽ làm thay đổi đáng kể xác suất của các cụm Ngram đã xuất hiện trong tập huấn luyện nếu kích thước bộ từ điển V là rất lớn. Trong thực nghiệm, một vài cụm Ngram có xác suất giảm đi gần 10 lần, do kích thước bộ từ điển là lớn trong khi tần số xuất hiện của cụm Ngram đó không cao. Để thuật toán thêm hiệu quả, người ta sử dụng công thức sau:

$$P(w_1w_2...w_n) = \frac{C(w_1w_2...w_n) + \lambda}{C(w_1w_2...w_{n-1}) + M\lambda}$$

Trong đó λ được chọn trong khoảng [0, 1], với một số giá trị thông dụng sau:

$\lambda = 0$: không làm mịn (MLE)

$\lambda = 1$: phương pháp add-one

$\lambda = \frac{1}{2}$: được gọi là phương pháp Jeffreys – Perks

Và M là cụm N-gram có thể có bằng V^N

- Thuật toán Witten-Bell

Thuật toán Witten-Bell hoạt động dựa trên nguyên tắc:

Khi gặp những cụm N-gram có tần số 0, ta coi đây là lần đầu tiên cụm từ này xuất hiện. Như vậy, xác suất của cụm N-gram có tần số bằng 0 có thể tính dựa vào xác suất gặp một cụm N-gram lần đầu tiên.

Với unigram, gọi T là số cụm unigram khác nhau đã xuất hiện, còn M là tổng số các cụm unigram đã thống kê, khi đó tổng số sự kiện sẽ là (T+M), và xác suất để

gặp cụm unigram lần đầu tiên (hay tổng xác suất của các cụm unigram chưa xuất hiện lần nào) được tính bằng: $\frac{T}{T+M}$

Gọi V là kích thước bộ từ vựng, còn Z là số cụm unigram chưa xuất hiện lần nào: Z = V - T

Xác suất xuất hiện của một cụm unigram chưa xuất hiện lần nào (có tần số bằng 0) được tính bằng:

$$P^* = \frac{T}{Z(T+M)}$$

Và xác suất xuất hiện của các cụm unigram có tần số khác 0 được tính lại theo công thức:

$P(w) = \frac{c(w)}{T+M}$ với c(w) là số lần xuất hiện của cụm w

Cũng giống thuật toán add-one, khi xét các cụm N-gram với N>1, thay M bằng $C(w_{i-n+1}...w_{i-1})$ thì xác suất của cụm $w_{i-n+1}...w_{i-1}w_i$ với $C(w_{i-n+1}...w_{i-1}w_i) = 0$ được tính theo công thức sau:

$$P(w_i|w_{i-n+1}...w_{i-1}) = \frac{T(w_{i-n+1}...w_{i-1})}{Z(w_{i-n+1}...w_{i-1})(C(w_{i-n+1}...w_{i-1}) + T(w_{i-n+1}...w_{i-1}))}$$

Với $C(w_{i-n+1}...w_{i-1}w_i) > 0$, thì xác suất cụm $w_{i-n+1}...w_{i-1}w_i$ tính bằng công thức:

$$P(w_i|w_{i-n+1}...w_{i-1}) = \frac{C(w_{i-n+1}...w_{i-1}w_i)}{C(w_{i-n+1}...w_{i-1}) + T(w_{i-n+1}...w_{i-1})}$$

- Thuật toán Good-Turing

Thuật toán Good-Turing dựa trên việc tính toán N_c, với N_c là số cụm N-gram xuất hiện c lần. Như vậy:

N_0 là số cụm n-gram có tần số 0 (số cụm N-gram không xuất hiện lần nào)

N_1 là số cụm n-gram có tần số 1 (số cụm N-gram xuất hiện 1 lần)

…

N_c có thể hiểu đơn giản là: $N_c = \sum_{w:count(w)=c} 1$

Khi đó, thuật toán Good-Turing sẽ thay thế tần số c bằng một tần số mới c* theo công thức:

$$c^* = (c+1) * \frac{N_{c+1}}{N_c}$$

Xác suất của một cụm N-gram với tần số là c được tính lại theo công thức:

$$P(w) = \frac{c^*}{N} \text{ với } N = \sum_{c=0}^{c=\infty} N_c c = \sum_{c=0}^{c=\infty} N_c c^* = \sum_{c=0}^{c=\infty} N_{c+1}(c+1)$$

Với công thức của Good-Turing được áp dụng như trên thì các tần số lớn sẽ ít có thay đổi, trong khi đó các tần số nhỏ hơn sẽ có thay đổi lớn. Tỉ lệ chênh lệch giữa tần số mới và tần số cũ này hợp lý hơn các thuật toán trước, vì các tần số nhỏ thường là tần số nhiễu, do vậy sự thay đổi của chúng không ảnh hưởng nhiều tới kết quả ước lượng, đồng thời bảo đảm được sự ổn định của các tần số lớn.

Trên thực tế, người ta không tính toán và thay thế mọi tần số c bởi một tần số mới c*. Người ta chọn một ngưỡng k nhất định, và chỉ thay thế tần số c bởi tần số mới c* khi c nhỏ hơn hoặc bằng k, còn nếu c lớn hơn k thì giữ nguyên tần số.

1.2.2.2 Phương pháp truy hồi (Back-off)

Trong các phương pháp chiết khấu như Add-One hay Witten-Bell, nếu cụm $w_{i-n+1}...w_{i-1}w_i$ không xuất hiện trong tập huấn luyện, và cụm $w_{i-n+1}...w_{i-1}$ cũng không xuất hiện, thì xác suất của cụm $w_{i-n+1}...w_{i-1}w_i$ sau khi làm mịn vẫn bằng 0. Phương pháp truy hồi tránh rắc rối trên bằng cách ước lượng xác suất các cụm Ngram chưa xuất hiện lần nào dựa vào xác suất của các cụm Ngram ngắn hơn có xác suất khác 0.

Cụ thể, xác suất của cụm $w_{i-n+1}...w_{i-1}w_i$ được tính lại theo công thức sau:

$$P_B(w_i|w_{i-n+1}...w_{i-1}) = \begin{cases} P(w_i|w_{i-n+1}...w_{i-1}) & \text{nếu } C(w_{i-n+1}...w_{i-1}w_i) > 0 \\ \alpha * P_B(w_i|w_{i-n+2}...w_{i-1}) & \text{nếu } C(w_{i-n+1}...w_{i-1}w_i) = 0 \end{cases}$$

Áp dụng cho bigram, ta có:

$$P_B(w_i|w_{i-1}) = \begin{cases} P(w_i|w_{i-1}) \text{ nếu } C(w_{i-1}w_i) > 0 \\ \alpha * P(w_i) \text{ nếu } C(w_{i-1}w_i) = 0 \end{cases}$$

Công thức trên có thể viết lại thành:

$$P_B(w_i|w_{i-1}) = P(w_i|w_{i-1}) + \mu(w_{i-1}w_i) * \alpha * P(w_i) \text{ với } u(x) = \begin{cases} 1 \text{ nếu } C(x) = 0 \\ 0 \text{ nếu } C(x) > 0 \end{cases}$$

Tương tự, khi áp dụng cho trigram ta có:

$$P_B(w_i|w_{i-2}w_{i-1}) = \begin{cases} P(w_i|w_{i-2}w_{i-1}) & \text{nếu } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_1 * P(w_i|w_{i-1}) & \text{nếu } C(w_{i-2}w_{i-1}w_i) = 0 \text{ và } C(w_{i-1}w_i) > 0 \\ \alpha_2 * P(w_i) & \text{nếu } C(w_{i-2}w_{i-1}w_i) = 0 \text{ và } C(w_{i-1}w_i) = 0 \end{cases}$$

Công thức trên cũng có thể viết lại thành:

$P_B(w_i|w_{i-2}w_{i-1}) = P(w_i|w_{i-2}w_{i-1}) + \mu(w_{i-2}w_{i-1}w_i) * \alpha_1 * P(w_i|w_{i-1}) + \mu(w_{i-1}w_i) * \alpha_2 * P(w_i)$

Với nguyên lý thực hiện của thuật toán truy hồi, người ta có thể loại bỏ khỏi mô hình ngôn ngữ những cụm n-gram có xác suất có thể tính theo công thức truy hồi (với sai số cho phép) dựa vào các cụm khác, nhờ vậy giảm được kích thước của bộ mô hình ngôn ngữ.

Sự chính xác của mô hình truy hồi phụ thuộc vào các tham số α_1 và α_2. Có vài kỹ thuật giúp lựa chọn được những tham số này, tùy theo tập huấn luyện và mô hình ngôn ngữ.

Một cách đơn giản, có thể chọn α_1 và α_2 là các hằng số. Tuy nhiên rất khó có thể chọn được hai hằng số để tổng xác suất của tất cả các cụm Ngram không thay đổi. Việc chọn hằng số không chính xác, sẽ làm ảnh hưởng lớn đến độ chính xác của cả mô hình ngôn ngữ. Do đó, ta có thể chọn tham số α như một hàm của Ngram:

$\alpha_1 = \alpha_1(w_{i-1}w_i)$ và $\alpha_2 = \alpha_2(w_{i-1}w_i)$

Tuy nhiên, trong phương pháp truy hồi, tổng xác suất của tất cả các cụm Ngram sẽ luôn lớn hơn 1, do xác suất của các cụm Ngram đã xuất hiện thì không thay đổi, trong khi xác suất của các cụm Ngram chưa xuất hiện thì được tăng lên. Do đó, để thuật toán chính xác hơn, thì ta cần kết hợp nó với một thuật toán chiết khấu như Witten-Bell hay Good-Turing để làm giảm xác suất của các cụm Ngram đã xuất hiện. Do đó, trong thực tế, chúng ta có công thức sau:

$$P(w_i|w_{i-2}w_{i-1}) = \begin{cases} P'(w_i|w_{i-2}w_{i-1}) & \text{nếu } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_1 * P'(w_i|w_{i-1}) & \text{nếu } C(w_{i-2}w_{i-1}w_i) = 0 \text{ và } C(w_{i-1}w_i) > 0 \\ \alpha_2 * P'(w_i) & \text{nếu } C(w_{i-2}w_{i-1}w_i) = 0 \text{ và } C(w_{i-1}w_i) = 0 \end{cases}$$

Trong đó P' chính là xác suất của cụm Ngram khi áp dụng thuật toán làm mịn chiết khấu.

1.2.2.3 Phương pháp nội suy (Interpolation)

Thuật toán nội suy có nguyên lý tương tự thuật toán truy hồi: "tính xác suất của các cụm n-gram dựa trên các cụm ngắn hơn". Điểm khác biệt chính so với thuật

toán truy hồi là thuật toán nội suy đưa ra công thức tính lại xác suất đối với mọi cụm n-gram, chứ không chỉ riêng các cụm n-gram không xuất hiện..

Công thức tính xác suất theo phương pháp nội suy như sau:

$$P_I(w_i|w_{i-n+1}...w_{i-1}) = \lambda P(w_i|w_{i-n+1}...w_{i-1}) + (1-\lambda)P_I(w_i|w_{i-n+2}...w_{i-1})$$

Áp dụng cho bigram và trigram ta có:

$$P_I(w_i|w_{i-1}) = \lambda P(w_i|w_{i-1}) + (1-\lambda)P(w_i)$$

$$P_I(w_i|w_{i-n+1}...w_{i-1}) = \lambda_1 P(w_i|w_{i-2}w_{i-1}) + \lambda_2 P(w_i|w_{i-1}) + \lambda_3 P(w_i) \text{ với } \sum_i \lambda_i = 1$$

Trong công thức trên, tổng các hệ số phải bằng 1 để đảm bảo tổng xác suất của các trigram không đổi (vẫn bằng 1).

Trong thực tế, các tham số trong công thức nội suy không phải là hằng số cố định mà được thay bằng hàm tham số ứng với các n-gram. Các hàm tham số được lựa chọn dựa trên bộ dữ liệu chọn lọc: chọn hàm tham số sao cho kết quả ước lượng dựa trên bộ dữ liệu chọn lọc

Tuy nhiên, cũng có thể chọn các tham số λ như là một hàm của Ngram:

$\lambda_1 = \lambda_1(w_{i-2}w_{i-1}w_i)$, $\lambda_2 = \lambda_2(w_{i-1}w_i)$ và $\lambda_3 = \lambda_3(w_i)$

1.2.2.4 Phương pháp làm mịn Kneser - Ney:

Thuật toán Kneser-Ney xây dựng theo hai mô hình: truy hồi và nội suy, tuy nhiên trong thuật toán này không cần phải áp dụng các thuật toán chiết khấu trước khi áp dụng công thức truy hồi.

- **Mô hình truy hồi:**

$$P_{BKN}(w_i|w_{i-n+1}..w_{i-1}) = \begin{cases} \dfrac{C(w_{i-n+1}...w_i) - D}{C(w_{i-n+1}...w_{i-1})} & \text{nếu } C(w_{i-n+1}...w_i) > 0 \\ \alpha(w_{i-n+1}...w_{i-1})P_{BKN}(w_i|w_{i-n+2}...w_{i-1}) & \text{nếu } C(w_{i-n+1}...w_i) = 0 \end{cases}$$

Trong đó:

$P_{BKN}(w_i) = \dfrac{N(vw_i) - D}{\sum_w N(vw)}$ với N(vw) là số lượng từ v khác nhau xuất hiện trước w trong tập huấn luyện

$$\alpha(w_{i-n+1}..w_{i-1}) = \frac{1 - \dfrac{\sum_{w:C(w_{i-n+1}..w_{i-1}w)>0} C(w_{i-n+1}..w_{i-1}w) - D}{C(w_{i-n+1}..w_{i-1})}}{1 - \sum_{w:C(w_{i-n+1}..w_{i-1}w>0)} P_{BKN}(w|w_{i-n+2}..w_{i-1})}$$

Như vậy:

$$P_{BKN}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \dfrac{C(w_{i-2}w_{i-1}w_i) - D}{C(w_{i-2}w_{i-1})} & \text{nếu } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha(w_{i-2}w_{i-1})P_{BKN}(w_i|w_{i-1}) & \text{nếu } C(w_{i-2}w_{i-1}w_i) = 0 \end{cases}$$

$$P_{BKN}(w_i|w_{i-1}) = \begin{cases} \dfrac{C(w_{i-1}w_i) - D}{C(w_{i-1})} & \text{nếu } C(w_{i-1}w_i) > 0 \\ \alpha(w_{i-1})P_{BKN}(w_i) & \text{nếu } C(w_{i-1}w_i) = 0 \end{cases}$$

$$P_{BKN}(w_i) = \frac{N(vw_i) - D}{\sum_w N(vw)}$$

- **Mô hình nội suy:**

$$P_{IKN}(w_i|w_{i-n+1}..w_{i-1}) = \frac{C(w_{i-n+1}..w_i) - D}{C(w_{i-n+1}..w_{i-1})} + \lambda(w_{i-n+1}..w_{i-1})P_{IKN}(w_i|w_{i-n+2}..w_{i-1})$$

Trong đó: . .

$\lambda(w_{i-n+1}..w_{i-1}) = \dfrac{D\,N(w_{i-n+1}..w_{i-1}v)}{C(w_{i-n+1}..w_{i-1})}$ với $N(w_{i-n+1}..w_{i-1}v)$ là số lượng từ v khác nhau xuất hiện liền sau cụm $w_{i-n+1}..w_i$ trong tập huấn luyện

$P_{IKN}(w_i) = \dfrac{N(vw_i) - D}{\sum_w N(vw)} + \lambda \dfrac{1}{V}$ với N(vw) là số lượng từ v khác nhau xuất hiện liền trước từ w trong tập huấn luyện.

$$\lambda = \frac{D\,N(v)}{\sum_w N(vw)}$$

Như vậy:

$$P_{IKN}(w_i|w_{i-2}w_{i-1}) = \frac{C(w_{i-2}w_{i-1}w_i) - D}{C(w_{i-2}w_{i-1})} + \lambda(w_{i-2}w_{i-1})P_{IKN}(w_i|w_{i-1})$$

$$P_{IKN}(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i) - D}{C(w_{i-1})} + \lambda(w_{i-1})P_{IKN}(w_i)$$

$$P_{IKN}(w_i) = \frac{N(vw_i) - D}{\sum_w N(vw)} + \lambda \frac{1}{V}$$

Trong cả 2 mô hình nội suy và truy hồi, D được chọn: $D = \frac{N_1}{N1 + 2N_2}$

1.2.2.5 Phương pháp làm mịn Kneser - Ney cải tiến bởi Chen - GoodMan:

Công thức tính toán của thuật toán Kneser-Ney cải tiến bởi Chen và GoodMan giống công thức của thuật toán Kneser-Ney, tuy nhiên hằng số D bị thay đổi.

Chen và GoodMan chọn D như sau:

$$D = \begin{cases} 0 \text{ nếu } c(w_{i-n+1}..w_i) = 0 \\ D_1 \text{ nếu } c(w_{i-n+1}.. w_i) = 1 \\ D_2 \text{ nếu } c(w_{i-n+1}.. w_i) = 2 \\ D_3 \text{ nếu } c(w_{i-n+1}.. w_i) >= 3 \end{cases}$$

Với $Y = \frac{N_1}{(N_1 + 2N_2)}$

$$D_1 = 1 - 2Y \frac{N_2}{N_1}$$

$$D_2 = 1 - 3Y \frac{N_3}{N_2}$$

$$D_3 = 1 - 4Y \frac{N_4}{N_3}$$

Trong đó: N_i là số lượng cụm N-gram có số lần xuất hiện bằng i

Chú ý rằng: với mỗi bậc của N-gram ta lại có một bộ 3 hằng số trên. Điều đó có nghĩa là: unigram, bigram, ... có các hằng số trên là khác nhau.

1.3 Kỹ thuật làm giảm kích thước dữ liệu

Khi kích thước tập văn bản huấn luyện lớn, số lượng các cụm Ngram và kích thước của mô hình ngôn ngữ cũng rất lớn. Nó không những gây khó khăn trong việc lưu trữ mà còn làm tốc độ xử lý của mô hình ngôn ngữ giảm xuống do bộ nhớ của máy tính là hạn chế. Để xây dựng mô hình ngôn ngữ hiệu quả, chúng ta phải giảm kích thước của mô hình ngôn ngữ mà vẫn đảm bảo độ chính xác

Các kỹ thuật này làm giảm kích thước của mô hình ngôn ngữ. Mặc dù đều có chung một mục tiêu, nhưng mỗi kỹ thuật lại có hiệu quả khác nhau. Có ba kỹ thuật chính, bao gồm:

- Pruning (loại bỏ): làm giảm số lượng các cụm Ngram trong mô hình ngôn ngữ bằng cách loại bỏ các cụm Ngram không quan trọng
- Quantization (lượng tử hóa): thay đổi cấu trúc thông tin của mỗi cụm Ngram trong mô hình ngôn ngữ.
- Compression (nén): nén cấu trúc dữ liệu sử dụng trong việc lưu trữ các cụm Ngram trong mô hình ngôn ngữ

1.3.1 Loại bỏ (pruning):

Số lượng các cụm Ngram xuất hiện vài lần trong tập huấn luyện thường là lớn so với tổng số các cụm Ngram. Các cụm Ngram đó thường là lỗi ngữ pháp trong tập huấn luyện, hoặc là một số dạng đặc biệt như: tên riêng, từ viết tắt,. .. Những cụm Ngram này thường rất ít sử dụng trong thực tế, do đó việc tồn tại của chúng có thể làm ảnh hưởng đến độ chính xác của mô hình ngôn ngữ. Chính vì lý do đó, kỹ thuật pruning tập trung vào việc loại bỏ các cụm Ngram như vậy. Có 2 phương pháp chính:

- Cut-off (cắt bỏ) : phương pháp này tập trung vào việc loại bỏ các cụm Ngram có tần số thấp trong tập huấn luyện
- Weighted difference : phương pháp này tập trung vào việc đánh giá và loại bỏ các cụm Ngram không hiệu quả dựa vào xác suất của các cụm Ngram trước và sau khi làm mịn theo phương pháp truy hồi.

1.3.1.1 Cắt bỏ (cut-off)

Phương pháp cut-off hoạt động như sau: Nếu cụm Ngram xuất hiện ít hơn k lần trong tập văn bản huấn luyện thì cụm Ngram đó sẽ bị loại bỏ ra khỏi mô hình ngôn ngữ. Khi tính toán, nếu gặp lại các cụm Ngram này, thì tần số và xác suất của chúng sẽ được tính toán thông qua các phương pháp làm mịn đã trình bày ở trên.

Trong một mô hình ngôn ngữ, chúng ta có thể sử dụng các tham số k khác nhau với các cụm Ngram có độ dài khác nhau. Ví dụ: với unigram thì sử dụng k = 10, với bigram thì k = 1, và trigram thì k =5

Như vậy, việc chọn tham số k cho phương pháp cut-off chính là vấn đề chính của kỹ thuật này. Nếu k quá lớn, chúng ta sẽ bỏ sót thông tin về một số cụm Ngram, hiệu suất của ứng dụng cũng bị giảm. Nhưng ngược lại, nếu k quá nhỏ, thì kích

thước của mô hình ngôn ngữ cũng giảm không đáng kể. Có 2 cách để chọn k: chọn k theo phương pháp chạy thử nhiều lần hoặc chọn k theo tỉ lệ phần trăm số lượng các cụm Ngram.

Chọn k theo phương pháp chạy thử nhiều lần nghĩa là ta dùng phương pháp cut-off cho mô hình ngôn ngữ với nhiều giá trị k khác nhau rồi đánh giá độ hỗn loạn thông tin(perplexity) của tập văn bản đầu vào sau khi sử dụng phương pháp cut-off. Sau khi có kết quả, ta sẽ chọn tham số k sao cho mô hình ngôn ngữ là hiệu quả nhất (độ hỗn loạn thông tin của tập văn bản huấn luyện và kích thước mô hình ngôn ngữ đều thấp). Kỹ thuật này giúp chúng ta chọn được k phù hợp, tuy nhiên rất mất thời gian do phải chạy thử với rất nhiều giá trị của k. Tuy nhiên, để đạt được một mô hình ngôn ngữ hiệu quả thì đây là một phương pháp tốt.

Phương pháp thứ hai, chọn k dựa theo tỷ lệ phần trăm của số lượng các cụm Ngram phải bảo đảm rằng số cụm Ngram xuất hiện không quá k lần chiếm h% so với tổng số các cụm Ngram. Ví dụ: nếu h=50, thì chọn k sao cho số lượng các cụm Ngram xuất hiện không quá k lần (sẽ bị loại bỏ) chiếm 50% tổng số các cụm Ngram đã thống kê. Phương pháp này tuy nhanh hơn nhưng độ chính xác không cao bằng phương pháp thứ nhất đã đề cập ở trên

1.3.1.2 Sự khác biệt trọng số (Weighted difference):

Phương pháp cut-off chỉ quan tâm đến việc loại bỏ các cụm Ngram có tần số thấp, trong khi phương pháp weighted difference (sự khác biệt trọng số) thì quan tâm đến nhiều thông tin trong mô hình ngôn ngữ hơn như mối quan hệ giữa các cụm Ngram, xác suất của từng cụm Ngram,. ..Như đã trình bày ở các phần trên, nếu một cụm Ngram không xuất hiện trong tập huấn luyện, thì xác suất của nó được ước lượng thông qua xác suất của các cụm Ngram ngắn hơn (phương pháp làm mịn kiểu truy hồi) Do đó, nếu xác suất thực tế của một cụm Ngram xấp xỉ với xác suất có được theo công thức truy hồi, thì chúng ta chẳng cần lưu trữ cụm Ngram đó làm gì nữa. Đó chính là ý tưởng của phương pháp weighted difference. Sự khác biệt trọng số của một cụm Ngram được định nghĩa bằng:

w.d.factor = K * log((xác suất ban đầu) - log(xác suất truy hồi))

K chính là tham số sử dụng trong phương pháp làm mịn Good Turing. Dựa vào nhân tố w.d.factor ở trên, chúng ta sẽ biết nên giữ lại hay loại bỏ một cụm Ngram. Nếu w.d.factor nhỏ hơn một ngưỡng nhất định, thì cụm Ngram đó sẽ bị loại bỏ khỏi mô hình ngôn ngữ. Và ngưỡng nhất định đó chúng ta có thể bằng cách tìm theo phương pháp thử sai hoặc đặt nó bằng một giá trị hằng số.

Trong thực tế, phương pháp này mất nhiều thời gian hơn phương pháp cut-off do phải tính toán hệ số w.d.factor cho tất cả các cụm Ngram trong mô hình ngôn ngữ. Và sự khác biệt lớn nhất giữa 2 phương pháp loại bỏ này chính là phương pháp weighted different chỉ hoạt động trong mô hình ngôn ngữ kiểu truy hồi, còn phương pháp cut-off thì chỉ hoạt động trong mô hình ngôn ngữ lưu trữ dữ liệu dưới dạng tần số.

1.3.2 Đồng hóa (Quantization)

Thuật toán quantization (đồng hóa) làm giảm số lượng bit dùng để lưu trữ các biến trong mô hình ngôn ngữ. Thuật toán này gồm hai bước chính

Bước thứ nhất, liệt kê và lưu trữ tất cả các tần số của các cụm Ngram vào một bảng. Sau đó, thay thế tần số của các cụm Ngram trong mô hình ngôn ngữ bằng chỉ số của tần số trong bảng. Như vậy, thay vì sử dụng $b = \log_2$(tần số lớn nhất) bit để lưu trữ tần số của một cụm Ngram, thì chúng ta chỉ cần sử dụng $b' = \log_2$(kích thước của bảng) bit cho mỗi cụm Ngram. Do kích thước của bảng nhỏ hơn nhiều so với giá trị tần số lớn nhất của các cụm Ngram nên $b' < b$, tức là kích thước mô hình ngôn ngữ đã giảm so với cách lưu trữ ban đầu.

Tuy nhiên, để tăng tính hiệu quả, ở bước thứ hai, thuật toán này đồng hóa một số giá trị trong bảng tần số. Điều đó có nghĩa là, một số giá trị trong bảng có giá trị gần với nhau sẽ được thay thế bằng một con số chung. Sau bước này, chúng ta sẽ thu được một bảng tần số với ít giá trị hơn, cũng tức là đã làm giảm kích thước của mô hình ngôn ngữ đi một lần nữa.

1.3.3 Nén (Compression)

Mô hình ngôn ngữ nào cũng có một cấu trúc dữ liệu. Do đó nếu cấu trúc dữ liệu đó được nén lại bằng các thuật toán nén, thì kích thước của mô hình ngôn ngữ tất nhiên là giảm. Tuy nhiên, khi một mô hình ngôn ngữ bị nén, thì độ chính xác và tốc độ của mô hình ngôn ngữ đều giảm (do phải giải nén, hoặc bị mất dữ liệu do thuật toán nén chưa tốt). Do không hiệu quả nên kỹ thuật này hiện nay không còn phổ biến như hai kỹ thuật trên, tuy vẫn được sử dụng bởi Microsoft (trong modul kiểm lỗi chính tả của Microsoft Office 2007).

1.4 Đánh giá mô hình ngôn ngữ

1.4.1 Entropy – Độ đo thông tin

Entropy là thước đo thông tin, có giá trị rất lớn trong xử lý ngôn ngữ. Nó thể hiện mức độ thông tin trong ngữ pháp, thể hiện sự phù hợp của một câu với một

ngôn ngữ, và dự đoán được từ tiếp theo trong cụm Ngram. Entropy của một biến ngẫu nhiên X được tính theo công thức**:**

$$H(X) = - \sum_{x \in X} p(x)\log_2 p(x)$$

Xét các câu gồm hữu hạn m từ $W = (w_1, w_2,.. .., w_m)$ trong ngôn ngữ L. Ta có công thức tính entropy như sau:

$$H(w_1, w_2,.. .., w_m) = - \sum_{W \in L} p(w_1, w_2,.. .., w_m)\log_2 p(w_1, w_2,.. .., w_m)$$

Từ công thức trên, ta có thể đưa ra công thức tính tỉ lệ entropy trên các từ như sau:

$$\frac{1}{m} H(w_1, w_2,.. .., w_m) = - \frac{1}{m} \sum_{W \in L} p(w_1, w_2,.. .., w_m)\log_2 p(w_1, w_2,.. .., w_m)$$

Thực tế thì tỉ lệ entropy trên các từ thường được sử dụng vì giá trị của nó không phụ thuộc vào độ dài các câu. Tuy nhiên, để tính được entropy của một ngôn ngữ L theo công thức trên thì ta phải xét tới các câu dài vô hạn (tất cả các câu có thể có trong ngôn ngữ L), đó là điều không thể. Do đó, ta có thể tính xấp xỉ tỉ lệ entropy trên các từ theo công thức sau:

$$H(L) = - \lim_{m \to \infty} \frac{1}{m} H(w_1, w_2,.. .., w_m)$$

$$= - \lim_{m \to \infty} \frac{1}{m} \sum_{W \in L} p(w_1, w_2,.. .., w_m)\log_2 p(w_1, w_2,.. .., w_m)$$

Định lý Shannon-McMillan-Breiman đã chỉ ra rằng nếu ngôn ngữ ổn định (chứa các câu gồm các từ với cấu trúc thông dụng) thì công thức trên có thể biến đổi thành:

$$H(L) = - \lim_{m \to \infty} \frac{1}{m} \log p(w_1, w_2,.. .., w_m)$$

Với công thức trên, ta có thể sử dụng công thức Bayes và xác suất của các n-gram để tính $p(w_1, w_2,.. .., w_n)$:

$$H(L) = - \lim_{m \to \infty} \frac{1}{m} \log [\ p(w_n|w_1w_2..w_{n-1}) * p(w_{n+1}|w_2w_3.. w_n) *. .. * p(w_m |w_{m-n+1}...w_{m-1})\ ]$$

Công thức trên đã được biến đổi qua nhiều bước với các xấp xỉ gần đúng, do vậy để tăng tính chính xác khi sử dụng độ đo entropy thì câu kiểm tra cần phải đủ dài và tổng quát (phân tán rộng) để tránh tập trung vào các xác suất lớn (chỉ chứa các cụm thông dụng).

Các bước biến đổi gần đúng công thức trên khiến giá trị H(L) tính theo công thức cuối cùng sẽ lớn hơn giá trị H(L) gốc. Do vậy, khi tính H(L) của các mô hình ngôn ngữ khác nhau trên ngôn ngữ L, mô hình nào cho H(L) nhỏ hơn thì mô hình ngôn ngữ đó thể hiện chính xác ngôn ngữ L hơn.

1.4.2 Perplexity – Độ hỗn loạn thông tin

Sau khi LM đã được huấn luyện, chúng ta cần phải đánh giá chất lượng của mô hình. Cách đánh giá chính xác nhất một mô hình ngôn ngữ là kiểm tra trong thực tế. Ví dụ trong nhận dạng tiếng nói, chúng ta có thể so sánh hiệu quả của hai mô hình ngôn ngữ bằng cách chạy bộ nhận dạng ngôn ngữ hai lần, mỗi lần với một mô hình và xem mô hình nào cho kết quả chính xác hơn. Nhưng cách này lại rất tốn thời gian, vì thế, chúng ta cần một công cụ mà có thể nhanh chóng đánh giá hiệu quả của một mô hình. Perplexity (PP) là thước đo thường được dùng cho công việc này.

Perplexity thực chất là một dạng biến đổi của entropy chéo (cross entropy) của mô hình. Entropy chéo là cận trên của entropy. Entropy là một khái niệm cơ bản trong thuyết thông tin, đánh giá lượng thông tin của dữ liệu bằng độ đo sự không chắc chắn. Nếu một biến ngẫu nhiên x tồn tại trong khoảng X của thông tin đang được đánh giá với phân phối xác suất là p, thì khi đó entropy của x được định nghĩa là:

$$H(x) = -\sum_{x \in X} p \log_2 p$$

Ví dụ khi tung một đồng xu, x chỉ có thể là mặt ngửa hoặc mặt sấp và xác suất $p = 0.5$ trong cả hai trường hợp. Nhưng khi tung một hột xúc xắc sáu mặt, khoảng giá trị có thể của kết quả rộng hơn, và các xác suất là $p = \frac{1}{6}$. Vì hành động tung xúc xắc có độ đo không chắc chắn lớn hơn, nên entropy của nó cũng cao hơn hành động tung đồng xu.

Entropy chéo của một mô hình là độ đo thông tin giữa hai phân phối xác suất. Đối với một phân phối xác suất q nào đó mà chúng ta sử dụng để mô hình hóa phân phối xác suất p, entropy chéo được định nghĩa là:

$$H(p,q) = -\sum_{x \in X} p \log_2 q$$

Định lý Shannon-McMillan-Breiman chỉ ra rằng đối với cả entropy và entropy chéo chúng ta đều có thể bỏ đi thành phần p nếu chuỗi giá trị x đủ dài. Nếu chúng ta cần tính entropy cho từng từ thì chỉ việc chia cho tổng số từ:

$$H(p,q) = -\frac{1}{n}\sum_{x} \log_2 q(x) = -\frac{1}{n} \log_2 q(x_n^1)$$

Perplexity được định nghĩa là $PP = 2^{H(p,q)}$. Do entropy chéo là cận trên của entropy, $H(p,q) \geq H(p)$, chúng ta sử dụng entropy chéo trong Perplexity để không bao giờ đánh giá thấp entropy thực sự của mô hình. Perplexity của một mô hình được đánh giá trên tập kiểm tra. Trong thực tế, Perplexity là thước đo đầu tiên để đánh giá một mô hình ngôn ngữ, và có thể được coi là hàm của cả cả ngôn ngữ và mô hình. Trên phương diện là hàm của mô hình, nó đánh giá một mô hình mô phỏng ngôn ngữ chính xác đến mức độ nào. Còn trên phương diện là hàm của ngôn ngữ, nó đo tính phức tạp của ngôn ngữ.

1.4.3 MSE - Lỗi trung bình bình phương

Các mô hình LM có mất mát không đảm bảo xác suất chính xác vì nó lưu trữ dữ liệu không đầy đủ, do đó làm biến dạng phân phối xác suất thông thường. Chính vì lý do này mà ta không thể sử dụng các phương pháp đo dựa trên Entropy như Perplexity để đánh giá chất lượng của mô hình. Tuy nhiên chúng ta vẫn có thể sử dụng một mô hình đảm bảo phân phối xác suất thông thường làm chuẩn mực để so sánh xem các lossy LM khác biệt như thế nào so với mô hình này. Điều này có thể được thực hiện bằng cách sử dụng Lỗi trung bình bình phương (Mean Square Error - MSE) của lossy LM và lossless LM, đều được huấn luyện và kiểm tra sử dụng các tập ngữ liệu giống nhau.

$$MSE = \frac{1}{n-1}\sum_{i}^{n} (X_i - X'_i)^2$$

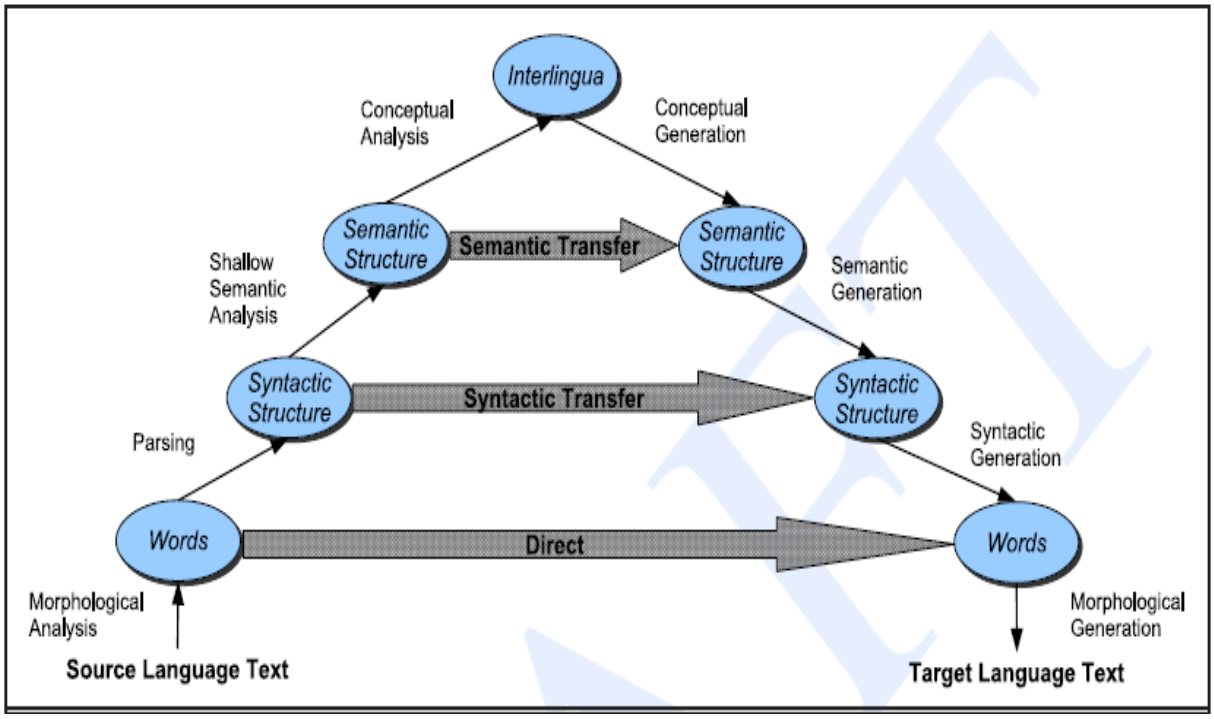
trong đó X là xác suất sự kiện i trong lossless LM và X’ là xác suất của cùng sự kiện đó trong lossy LM.

CHƯƠNG 2: ỨNG DỤNG CỦA MÔ HÌNH NGÔN NGỮ TRONG DỊCH MÁY THỐNG KÊ

2.1 Dịch máy

Dịch máy (Machine Translation - MT) là một hướng phát triển có lịch sử lâu đời từ thập kỷ 50 và được phát triển mạnh mẽ từ thập kỷ 80 cho đến nay. Hiện tại, trên thế giới có rất nhiều hệ dịch máy thương mại nổi tiếng trên thế giới như Systrans, Kant, … hay những hệ dịch máy mở tiêu biểu là hệ dịch của Google, hỗ trợ hàng chục cặp ngôn ngữ phổ biến như Anh-Pháp, Anh-Trung, Anh-Nhật, Hoa-Nhật, …

Các cách tiếp cận cổ điển cho hệ dịch máy: ba lớp chính là dịch trực tiếp (direct), dịch dựa trên luật chuyển đổi (transfer), dịch liên ngữ (interlingua), và hiện nay, tiếp cận dịch dựa vào thống kê (statistical MT) đang được nghiên cứu rộng rãi.



Hình 2.1: Các tiếp cận cổ điển cho hệ dịch máy

Phương pháp dịch dựa trên luật chuyển đổi và dịch liên ngữ chủ yếu dựa vào cú pháp, đã có thời gian phát triển khá dài và vẫn còn được sử dụng phổ biến trong nhiều hệ dịch thương mại. Các hệ dịch máy loại này này đã đạt được kết quả khá tốt với những cặp ngôn ngữ tương đồng nhau về cú pháp như Anh-Pháp, Anh-Tây Ban Nha,… nhưng còn gặp nhiều hạn chế đối với các cặp ngôn ngữ có cú pháp khác nhau như Anh-Trung, Anh-Nhật,…

Ở Việt Nam, dịch Anh-Việt, Việt-Anh cũng vấp phải những khó khăn tương tự do sự khác biệt về mặt cấu trúc ngữ pháp và tính nhập nhằng của ngữ nghĩa. Các hệ thống dịch Anh-Việt dựa trên luật chuyển đổi được thương mại hóa đầu tiên ở Việt Nam là EVTran, MTD của Lạc Việt. Hiện nay, nhiều nghiên cứu với mong muốn tăng chất lượng dịch vẫn đang được thực hiện thích nghi với đặc điểm của các cặp ngôn ngữ khác nhau.

2.2 Dịch máy thống kê

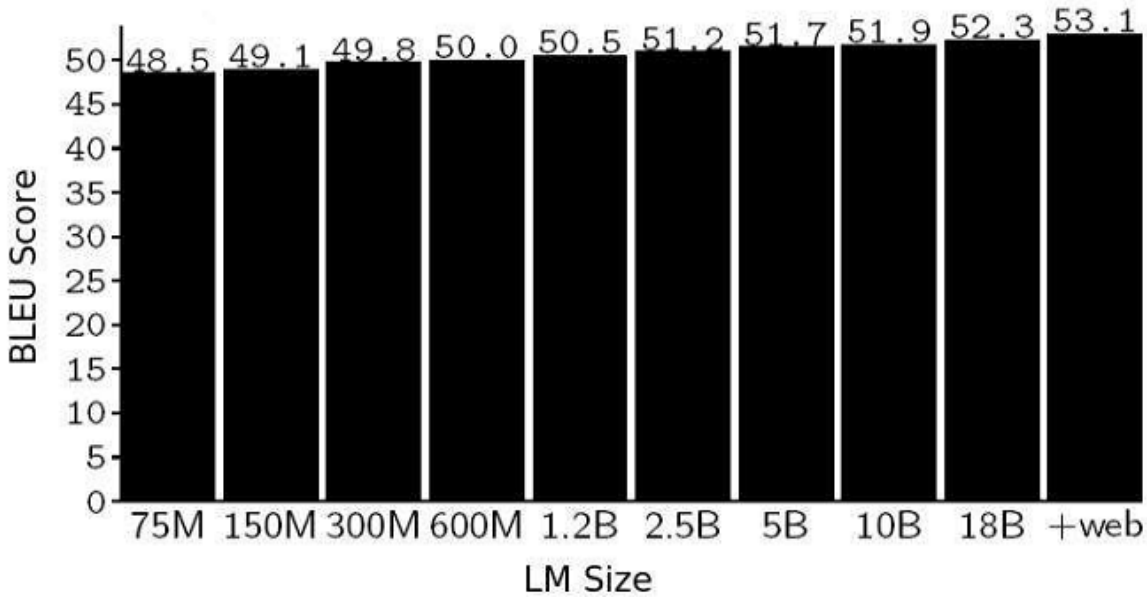
2.2.1 Giới thiệu

Dịch máy bằng phương pháp thống kê (Statistical Machine Translation) đã chứng tỏ là một hướng tiếp cận đầy đầy tiềm năng bởi những ưu điểm vượt trội so với các phương pháp dịch máy dựa trên cú pháp truyền thống qua nhiều thử nghiệm về dịch máy. Thay vì xây dựng các từ điển, các luật chuyển đổi bằng tay, hệ dịch này tự động xây dựng các từ điển, các quy luật dựa trên kết quả thống kê có được từ dữ liệu. Chính vì vậy, dịch máy dựa vào thống kê có tính khả chuyển cao, có khả năng áp dụng được cho cặp ngôn ngữ bất kỳ. Hệ thống SMT được đề xuất lần đầu tiên bởi Brown năm 1990 sử dụng mô hình kênh nhiễu và đã phát triển áp đảo trong ngành MT nhiều năm trở lại đây.

Trong phương pháp dịch trực tiếp, từng từ được dịch từ ngôn ngữ nguồn sang ngôn ngữ đích. Trong dịch dựa trên luật chuyển đổi, đầu tiên chúng ta cần phải phân tích cú pháp của câu vào, rồi áp dụng các luật chuyển đổi để biến đổi cấu trúc câu này ở ngôn ngữ nguồn sang cấu trúc của ngôn ngữ đích; cuối cùng ta mới dịch ra câu hoàn chỉnh. Đối với dịch liên ngữ, câu vào được phân tích thành một dạng biểu diễn trừu tượng hóa về ngữ nghĩa, được gọi là "interlingua", sau đó ta tìm cách xây dựng câu đích phù hợp nhất với "interlingua" này. Dịch máy thống kê có cách tiếp cận hoàn toàn khác, khả năng dịch có được là dựa trên các mô hình thống kê được huấn luyện từ các ngữ liệu song ngữ. Kiến trúc chung của một hệ thống SMT được thể hiện trong hình 2.3.

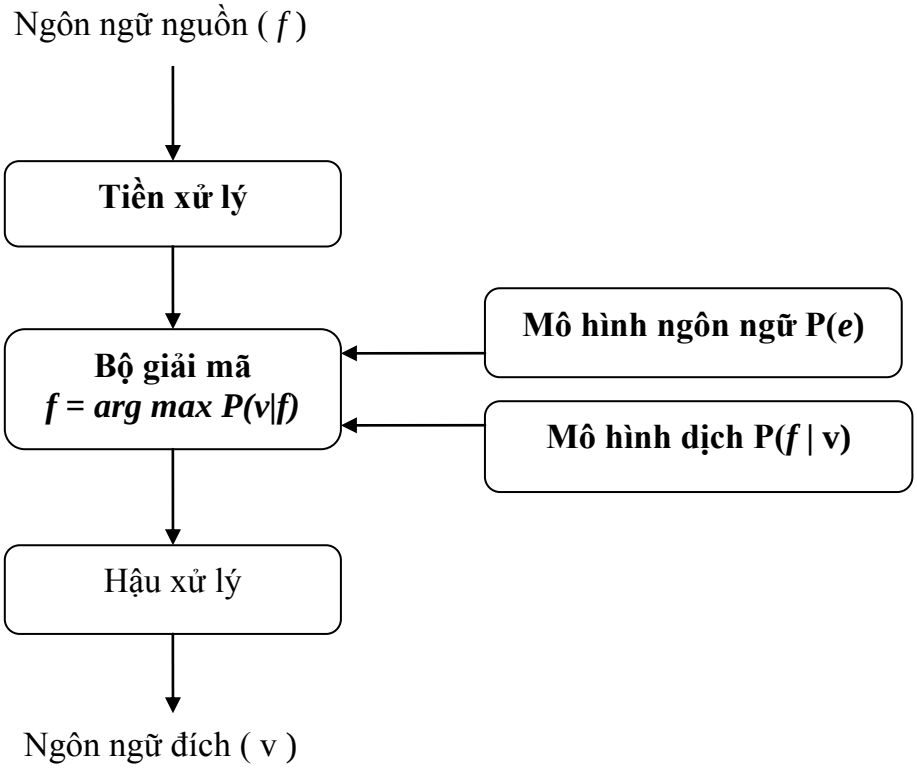
Mô hình của Brown (hay còn gọi là mô hình IBM) biểu diễn quá trình dịch bằng một mô hình kênh nhiễu (noisy channel model) bao gồm ba thành phần: một mô hình dịch (translation model), có nhiệm vụ liên hệ các từ, cụm từ tương ứng của các ngôn ngữ khác nhau; một mô hình ngôn ngữ (LM), đại diện cho ngôn ngữ đích; một bộ giải mã (decoder), kết hợp mô hình dịch và mô hình ngôn ngữ để thực hiện nhiệm vụ dịch.

Thường thì LM được gán trọng số cao hơn các thành phần khác trong hệ thống dịch, bởi vì ngữ liệu đơn ngữ dùng để huấn luyện LM lớn hơn nhiều ngữ liệu song ngữ, do đó có độ tin cậy lớn hơn. Ta thấy rằng việc tăng kích cỡ của LM cải thiện điểm BLEU – tiêu chuẩn phổ biến để đánh giá chất lượng dịch máy. Hình 2.2, cho thấy sự cải thiện chất lượng dịch khi tăng kích cỡ LM.



Hình 2.2 : Tăng kích cỡ LM cải thiện điểm BLEU

Trong mô hình đầu tiên của Brown, mô hình dịch dựa trên kiểu từ-thành-từ và chỉ cho phép ánh xạ một từ trong ngôn ngữ nguồn đến một từ trong ngôn ngữ đích. Nhưng trong thực tế, ánh xạ này có thể là một-một, một-nhiều, nhiều-nhiều hoặc một-không. Thế nên nhiều nhà nghiên cứu đã cải tiến chất lượng của SMT bằng cách sử dụng dịch dựa trên cụm (phrase-based translation).



Hình 2.3 : Kiến trúc của một hệ thống SMT

2.2.2 Nguyên lý và các thành phần

Cho trước câu ngôn ngữ nguồn f, mục tiêu của mô hình dịch máy là tìm ra câu e của ngôn ngữ đích sao cho xác suất P(e|f) là cao nhất.

Có nhiều cách tiếp cận để tính được xác suất P(e|f), tuy nhiên cách tiếp cận trực quan nhất là áp dụng công thức Bayes:

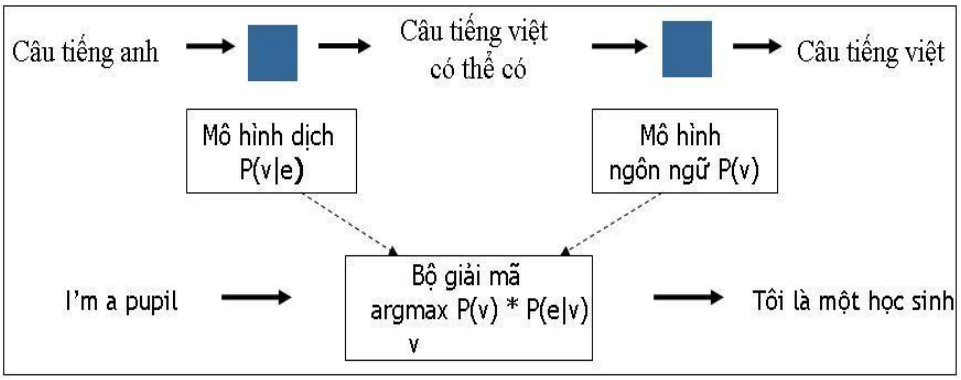
$$P(e|f) = \frac{P(e)P(f|e)}{P(f)}$$

Trong đó P(f|e) là xác suất câu ngôn ngữ nguồn là bản dịch của câu ngôn ngữ đích, còn P(e) là xác suất xuất hiện câu e trong ngôn ngữ. Việc tìm kiếm câu e* phù hợp chính là việc tìm kiếm e* làm cho giá tri P(e*)P(f|e*) là lớn nhất.

Để mô hình dịch là chính xác, thì công việc tiếp theo là phải tìm ra tất cả các câu e* có thể có trong ngôn ngữ đích từ câu ngôn ngữ nguồn f. Thực hiện công việc tìm kiếm hiệu quả chính là nhiệm vụ của bộ giải mã (decoder). Như vậy, một mô hình dịch máy bao gồm 3 thành phần:

- Mô hình ngôn ngữ: Tính toán được xác suất của câu ngôn ngữ nguồn. Thành phần này chính là mô hình ngôn ngữ đã được mô tả ở chương 1 của luận văn
- Mô hình dịch: Cho biết xác suất của câu ngôn ngữ nguồn là bản dịch từ câu ngôn ngữ đích.
- Bộ giải mã: Tìm kiếm tất cả các câu ngôn ngữ đích e có thể có từ câu ngôn ngữ nguồn f.

Mô hình dịch từ tiếng Anh sang tiếng Việt có thể hình dung thông qua biểu đồ dưới đây:



Hình 2.4 : Mô hình dịch máy thống kê từ tiếng Anh sang tiếng Việt

Mô hình dịch của mô hình ngôn ngữ đã được trình bày ở chương trước của luận văn. Ở phần này, luận văn chỉ đề cập đến hai thành phần còn lại của mô hình dịch máy thống kê.

2.2.3 Mô hình dịch

Mô hình dịch có 3 hướng tiếp cận chính:

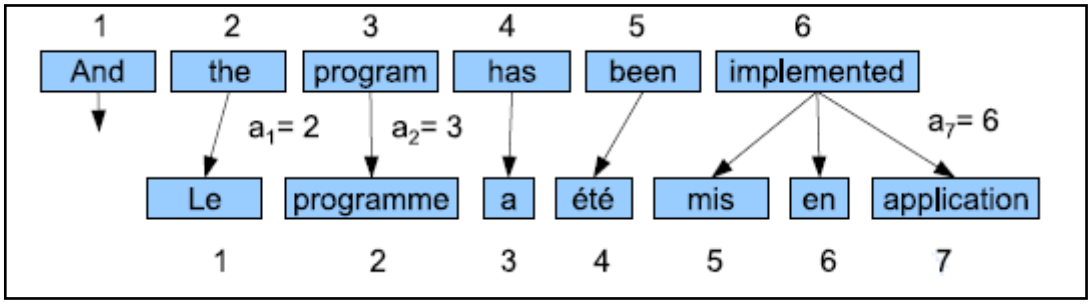
- Mô hình dịch dựa trên từ (word-based)
- Mô hình dịch dựa trên cụm từ (phrase-based)
- Mô hình dịch dựa trên cú pháp (syntax-based)

Cả 3 hướng tiếp cận trên đều dựa trên một tư tưởng. Đó là sự tương ứng giữa hai câu (alignment)

2.2.3.1 Sự gióng hàng (alignment)

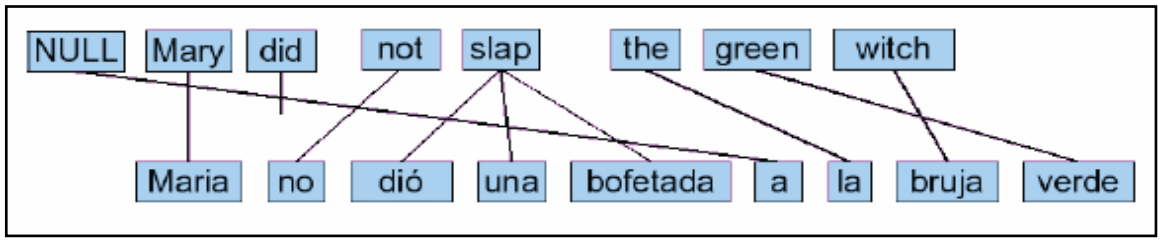
Tất cả các mô hình dịch thống kê đều dựa trên sự tương ứng của từ. Sự tương ứng của từ ở đây chính là một ánh xạ giữa một hay nhiều từ của ngôn ngữ nguồn với một hay nhiều từ của ngôn ngữ đích trong tập hợp các câu văn bản song ngữ.

Theo nguyên tắc, chúng ta có thể có mối liên hệ tùy ý giữa các từ của ngôn ngữ nguồn với các từ của ngôn ngữ đích. Tuy nhiên, để cho đơn giản, mô hình dịch máy dựa trên từ (word-based) đưa ra một giả định: mỗi từ của ngôn ngữ đích chỉ tương ứng với một từ của ngôn ngữ nguồn. Nếu áp dụng giả định này, chúng ta có thể biểu diễn một sự tương ứng từ bằng chỉ số của các từ trong ngôn ngữ nguồn tương ứng với từ trong ngôn ngữ đích. Như trong ví dụ ở hình 2.5 dưới đây có thể biểu diễn một tương ứng từ giữa tiếng Pháp và tiếng Anh bởi một dãy các chỉ số như sau: A = 1,2, 3, 4, 5, 6.



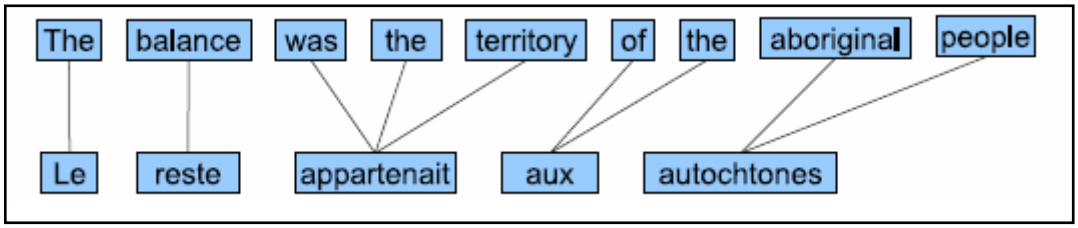
Hình 2.5: Sự tương ứng một - một giữa câu tiếng Anh và câu tiếng Pháp

Trong thực tế, có rất nhiều từ ở ngôn ngữ đích không tương ứng với từ nào trong ngôn ngữ nguồn. Để cho tổng quát, ta thêm một từ vô giá trị (null) vào đầu câu ngôn ngữ nguồn và những từ ở ngôn ngữ đích không tương ứng với từ nào sẽ được ánh xạ với từ vô giá trị đó. Hình 2.6 ở dưới thể hiện một tương ứng từ giữa hai câu tiếng Anh và tiếng Tây Ban Nha khi cho thêm từ vô giá trị vào đầu câu tiếng Anh.

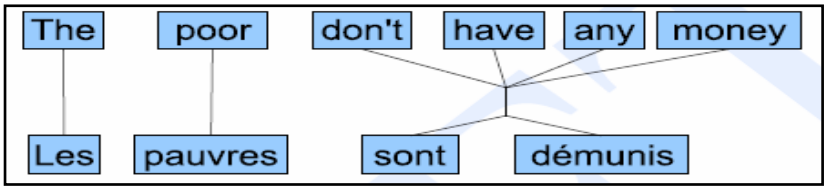


Hình 2.6: Sự tương ứng giữa câu tiếng Anh với câu tiếng Tây Ban Nha khi cho thêm từ vô giá trị (null) vào đầu câu tiếng Anh

Trong khi mô hình dịch dựa trên từ (word-based) chỉ giải quyết trường hợp một từ của ngôn ngữ đích chỉ tương ứng bởi một từ của ngôn ngữ nguồn, thì mô hình dịch dựa trên cụm từ (pharse-based) có thể giải quyết cả hai trường hợp còn lại là: một từ của ngôn ngữ này tương ứng với nhiều từ của ngôn ngữ kia và nhiều từ của ngôn ngữ này tương ứng với nhiều từ của ngôn ngữ kia. Hình 2.7 và 2.8 ở dưới minh họa các tương ứng nói trên.



Hình 2.7 : Sự tương ứng một - nhiều giữa câu tiếng Anh với câu tiếng Pháp



Hình 2.8 : Sự tương ứng nhiều - nhiều giữa câu tiếng Anh với câu tiếng Pháp.

2.2.3.2 Mô hình dịch dựa trên từ (Word-based)

Mô hình dịch dựa trên từ là thế hệ đầu tiên của mô hình dịch máy thống kê và được nghiên cứu và phát triển bởi IBM. Như đã trình bày ở phần trước**,** mô hình dịch này dựa trên sự tương ứng của các từ theo tương ứng một một (một từ của ngôn ngữ này chỉ tương ứng với một từ của ngôn ngữ kia và ngược lại). Cụ thể hơn, giả sử câu ngôn ngữ nguồn là $e_1e_2...e_n$ và câu ngôn ngữ đích là $f_1f_2...f_m$, khi đó mỗi từ f_j chỉ tương ứng với 1 và chỉ 1 từ trong câu ngôn ngữ nguồn hoặc là không tương ứng với từ nào. Do đó, một sự tương ứng giữa các từ của câu ngôn ngữ nguồn và câu ngôn ngữ đích có thể biểu diễn bằng một dãy m số: $\{a_1, a_2,.. .. a_m\}$ trong đó a_j là chỉ số của từ trong ngôn ngữ nguồn tương ứng với từ f_j của ngôn ngữ đích(a_j nhận các giá trị từ 1 đến l). Với mô hình IBM thứ nhất, giả định rằng mỗi biến a_j là độc lập, khi đó tương ứng tối ưu nhất chính là:

$$a = \underset{a_1^m}{\arg\max} \prod_{i=1}^{i=m} p(a_i)*p(f_i|e_{ai})$$

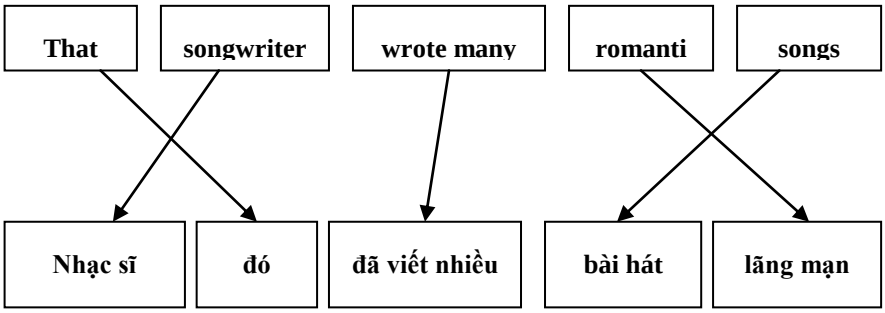
Như vậy, theo mô hình IBM thứ nhất, chúng ta có thể tính xác suất P(f|e) theo công thức sau:

$$P(f|e) = \prod_{i=1}^{i=m} \sum_{a_i=0}^{ai=n} p(a_i)*p(f_i|e_{ai})$$

Tuy nhiên trên thực tế, mô hình IBM thứ nhất này có chất lượng dịch không cao. Ở các mô hình IBM tiếp theo, người ta cải tiến các công thức và đưa ra những tương ứng, cũng như tính lại xác suất P(f|e) một cách tốt hơn. Tuy nhiên, do tiếp cận theo hướng tương ứng một một giữa các từ, nên mô hình dịch dựa trên từ nói

chung và các mô hình dịch IBM nói riêng đã không còn phổ biến. Hiện nay, các mô hình dịch theo hướng cụm từ được sử dụng rộng rãi và dần trở nên phổ biến hơn.

2.2.3.3 Mô hình dịch dựa trên cụm từ (Phrase-based)



Hình 2.9: Minh họa dịch máy thống kê dựa vào cụm từ

Trong dịch dựa trên cụm, một chuỗi các từ liên tiếp (cụm) được dịch sang ngôn ngữ đích, với độ dài cụm ngôn ngữ nguồn và đích có thể khác nhau. Hình 2.9 minh họa phương pháp dịch cụm: câu vào được chia thành một số cụm; từng cụm một được dịch sang ngôn ngữ đích; và sau đó các cụm được đảo trật tự theo một cách nào đó rồi ghép với nhau. Cuối cùng ta thu được câu dịch trong ngôn ngữ đích.

Giả sử ta gọi ngôn ngữ nguồn là f và ngôn ngữ đích là e, chúng ta sẽ cố gắng tối đa hóa xác suất $\Pr(f \mid e)$ với mong muốn có được bản dịch tốt nhất. Thực tế là tồn tại rất nhiều bản dịch đúng cho cùng một câu, mục đích của ta là tìm ra câu ngôn ngữ e phù hợp nhất khi cho trước câu ngôn ngữ nguồn f. Dịch dựa vào cụm sử dụng mô hình kênh nhiễu, áp dụng công thức Bayes ta có:

$$\arg\max_e \mathrm{P}(e \mid f) = \frac{\arg\max_e \mathrm{P}(f \mid e)\mathrm{P}(e)}{\mathrm{P}(f)}$$

Do P(f) là không đổi đối với e, vấn đề trở thành việc tìm câu e nhằm tối đa hóa P(f\e)P(e). Việc xây dựng mô hình ngôn ngữ cần sử dụng một ngữ liệu đơn ngữ lớn, trong khi đó mô hình dịch lại cần đến ngữ liệu song ngữ tốt. Bộ giải mã được sử dụng để chia câu nguồn thành các cụm và sinh ra các khả năng dịch có thể cho mỗi cụm nhờ sự trợ giúp của bảng cụm từ (phrase table).

Để sinh ra được câu dịch, câu nguồn được chia thành I cụm liên tiếp f_1^I. Chúng ta giả sử rằng phân phối xác suất là như nhau đối với các cụm này. Mỗi cụm f_i trong f_1^I được dịch thành cụm tương ứng trong ngôn ngữ đích e_i. Các cụm trong ngôn ngữ đích có thể đảo ví trí cho nhau. Quá trình dịch cụm được mô hình hóa bởi phân phối xác suất $\phi(f_i \mid e_i)$.

Việc đảo ví trí (reodering) của các cụm đầu ra được mô hình bởi phân phối xác suất $d(a_i - b_{i-1})$, trong đó a_i đại diện cho vị trí bắt đầu của cụm trong câu nguồn được dịch thành cụm thứ i trong câu đích, và b_{i-1} là ký hiệu chỉ vị trí kết thúc của cụm trong câu nguồn được dịch thành cụm (i-1) trong câu đích. Ở đây chúng ta sử dụng mô hình đảo cụm rất đơn giản như sau:

$$d(a_i - b_{i-1}) = \alpha^{|a_i - b_{i-1} - 1|}$$

với giá trị thích hợp cho tham số α.

Để xác định độ dài thích hợp của câu dịch, chúng ta đưa thêm vào thừa số ω khi sinh ra câu trong ngôn ngữ đích. Thừa số này sẽ được tối ưu qua quá trình tìm kiếm câu dịch tối ưu. Thừa số này càng lớn hơn 1 thì độ dài của câu trong ngôn ngữ đích càng dài.

Nói tóm lại, câu dịch tốt nhất e_{best} được sinh ra từ câu nguồn theo là:

$$e_{best} = \arg\max_e P(e \mid f) = \arg\max_e P(f \mid e) P_{LM}(e)\ \omega^{length(e)}$$

ở đâyP(f\e) được phân tích thành:

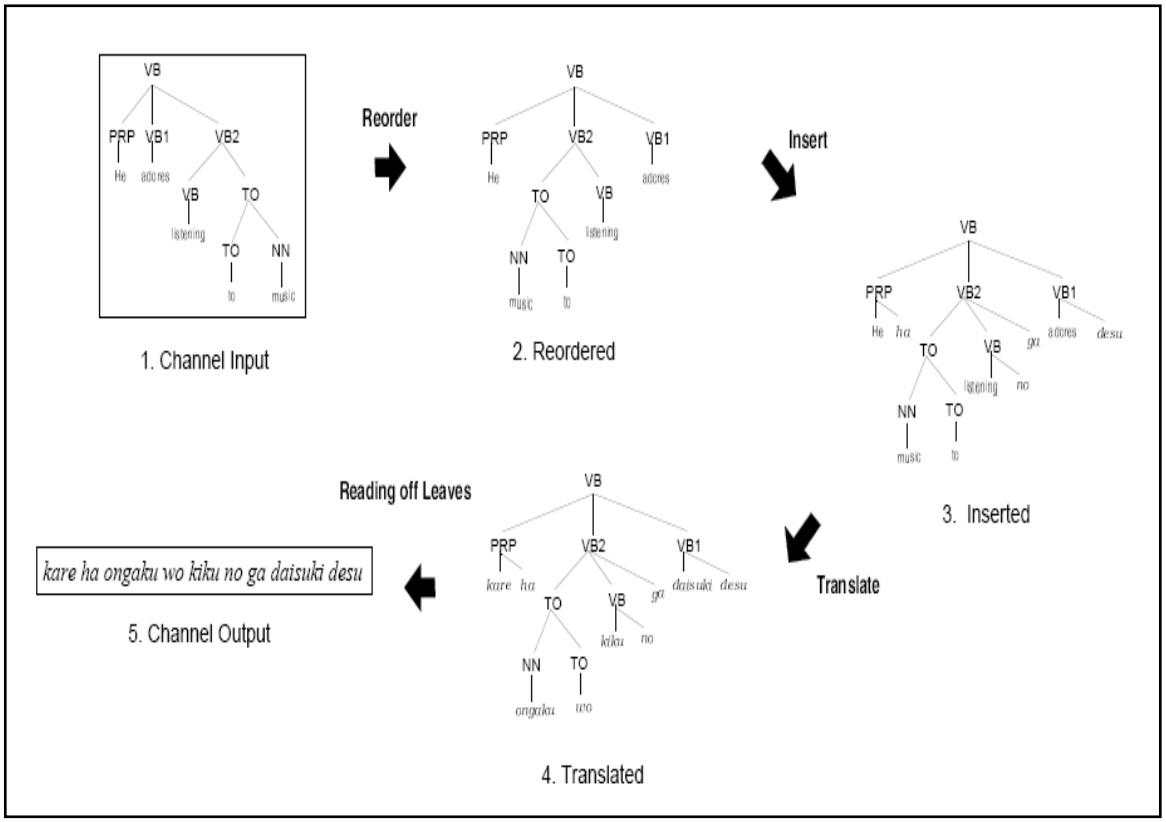
$$P(f_1^I \mid e_1^I) = \prod_{i=1}^{I} \varphi(f_i \mid e_i)\ d(a_i - b_{i-1})$$

2.2.3.4 Mô hình dịch dựa trên cú pháp (Syntax-based)

Cả 2 mô hình dịch dựa trên từ và cụm từ đều chỉ quan tâm đến sự tương ứng và ngữ nghĩa của từng từ trong câu ngôn ngữ nguồn và đích mà không quan tâm tới ngữ pháp, hình thái của cả hai câu. Mô hình dịch dựa trên cú pháp không chỉ quan tâm tới ngữ nghĩa của từng từ mà còn chú trọng tới cú pháp của câu.

Với mô hình dịch này, một câu ngôn ngữ nguồn e sẽ được phân tích thành cây cú pháp. Cây cú pháp này sẽ được sắp xếp lại để phù hợp với cú pháp của câu ngôn ngữ đích. Sau đó, một số từ mới có thể được chèn vào cây hiện tại cho phù hợp hơn với cú pháp của ngôn ngữ đích. Cuối cùng, các từ trong cây cú pháp của câu ngôn ngữ nguồn sẽ được dịch sang ngôn ngữ đích và ta thu được câu ngôn ngữ đích từ cây cú pháp trên.

Hình 2.10 dưới đây mô tả các bước làm việc của một mô hình dịch dựa trên cú pháp từ tiếng Anh sang tiếng Nhật.



Hình 2.10: Mô hình dịch dựa trên cây cú pháp

2.2.4 Bộ giải mã

Như đã trình bày ở các phần trên, nhiệm vụ của bộ giải mã chính là: cho trước câu ngôn ngữ nguồn f, tìm câu ngôn ngữ đích e tốt nhất được dịch từ f. Câu ngôn ngữ đích e tốt nhất chính là câu làm cho giá trị P(f|e)*P(e) là lớn nhất.

Bộ giải mã được phát triển đầu tiên cho mô hình dịch cụm từ được giới thiệu bởi Marcu và Wong, sử dụng các phương pháp leo đồi. Do không gian tìm kiếm là rất lớn, nên bộ giải mã trong mô hình dịch máy thống kê thường áp dụng các thuật toán tìm kiếm tối ưu.

Thuật toán mà bộ giải mã thường áp dụng có tên là A*, là một trong các phương pháp tìm kiếm tốt nhất - đầu tiên.Giải thuật A* có thể tóm tắt như sau: tại mỗi bước mở rộng không gian tìm kiếm, ta sử dụng các hàm ước lượng, đánh giá trọng số để kết quả tìm được luôn là tốt nhất có thể và là kết quả tìm thấy đầu tiên.

Ngữ liệu sau khi qua 2 mô hình ngôn ngữ và mô hình dịch ta được bảng xác suất cho từng thông số tương ứng. Vấn đề tìm ra tích số P(e)P(v|e) lớn nhất.

Có hai thuật giải và một thuật toán tối ưu cho mô hình tìm kiếm: thuật giải tìm kiếm tham lam, thuật giải tìm kiếm dựa trên ngăn xếp, và thuật toán tìm kiếm

theo chu trình Hamilton tối ưu. Hai thuật giải có thời gian nhanh xử lý nhanh hơn thuật toán nhưng kết quả thấp hơn thuật toán.

Bộ giải mã thực hiện một cái tìm kiếm theo chùm (beam search) tương tự công việc của Tillmann và Och. Bắt đầu bằng việc định nghĩa các khái niệm cơ bản của các lựa chọn dịch mô tả cơ chế hoạt động của beam search và các thành phần cần thiết của nó và các ước lượng giá trị tương lai và các khái niệm về sinh danh sách n-best.

2.3 Các phương pháp đánh giá bản dịch

Đánh giá độ chính xác của hệ thống dịch máy là một nhiệm vụ rất vất vả và khó khăn. Để đánh giá độ chính xác của bản dịch, ta có thể đánh giá trực tiếp thông qua người dùng hoặc đánh giá tự động bằng máy tính.

2.3.1 Đánh giá trực tiếp bằng con người

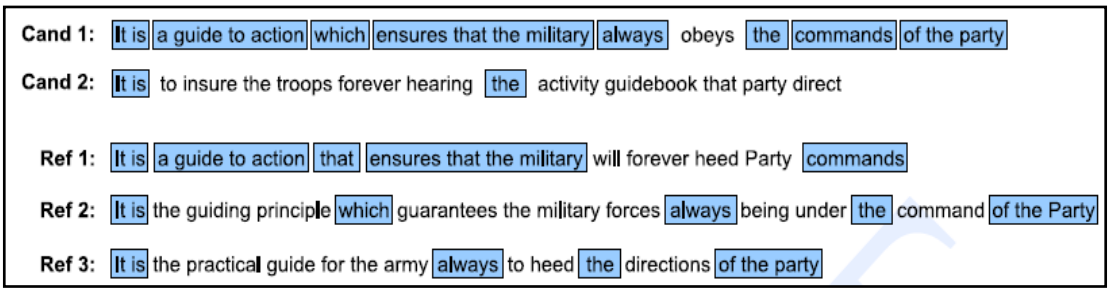
Để đánh giá độ chính xác của hệ thống dịch máy, ta có thể để con người trực tiếp đánh giá. Chúng ta có thể đưa ra một thước đo cho độ trôi chảy của bản dịch (ví dụ từ 1 đến 5 hay từ 1 đến 10 tùy thuộc vào độ trôi chảy của bản dịch), sau đó cho những người tham gia đánh giá đánh giá các câu trong bản dịch theo thang điểm đó. Như vậy, văn bản nào có điểm trung bình càng cao, thì chất lượng bản dịch đó càng tốt. Ngoài ra, cũng có thể đánh giá độ trôi chảy, độ chính xác của bản dịch thông qua thời gian mà người đọc đọc hiểu được bản dịch đó. Rõ ràng, bản dịch nào mà người đọc đọc hiểu càng nhanh, thì bản dịch đó càng chính xác.

Phương án đánh giá bản dịch bằng chính con người tuy rất dễ thực hiện, nhưng chi phí thì rất lớn, và nếu bản dịch có kích thước càng lớn thì phương pháp này càng kém hiệu quả. Ngày nay, các mô hình dịch máy đều áp dụng phương pháp đánh giá tự động, chi phí thấp nhưng hiệu quả cũng khá là cao.

2.3.2 Đánh giá tự động: phương pháp BLEU

BLEU(Bilingual Evaluation Understudy) là một thuật toán để đánh giá chất lượng văn bản đã được máy dịch từ một ngôn ngữ tự nhiên khác. Ý tưởng chính của phương pháp này là so sánh kết quả bản dịch tự động bằng máy với các bản dịch mẫu của con người, bản dịch máy nào càng giống với bản dịch mẫu của con người thì bản dịch đó càng chính xác. Việc so sánh trên được thực hiện thông qua việc thống kê sự trùng khớp của các từ trong hai bản dịch có tính đến thứ tự của chúng trong câu (phương pháp n-grams theo từ).

Trong ví dụ như hình 2.10, có hai bản dịch bằng máy được đem so sánh với ba bản dịch mẫu của con người. Có thể thấy rằng, bản dịch thứ máy nhất có nhiều từ chung (đóng khung) với các bản dịch mẫu hơn bản dịch máy thứ hai, nên theo phương pháp này có thể kết luận : bản dịch máy thứ nhất chính xác hơn bản dịch máy thứ hai



Hình 2.11: Sự trùng khớp của các bản dịch máy với bản dịch mẫu

Với một bản dịch máy và bản dịch mẫu thứ n, phương pháp BLEU trước tiên thống kê số lần tối thiểu của cụm Ngram xuất hiện trong từng cặp câu (câu dịch máy và câu dịch mẫu), sau đó đem tổng trên chia cho tổng số cụm Ngram trong toàn bản dịch máy. Tỉ lệ trùng khớp của một bản dịch máy và bản dịch mẫu thứ n được tính theo công thức :

$$P_n = \frac{\sum_{s \in \text{bản dịch máy}} \sum_{w \in s} \text{số lượng tối thiểu cụm w có trong bản mẫu}}{\sum_{s \in \text{bản dịch máy}} \sum_{w \in s} \text{số lượng cụm n-gram w trong bản dịch máy}}$$

Trong đó s là các câu trong bản dịch máy, w là các cụm ngram có trong câu s.

Điểm BLEU đánh giá một bản dịch máy với n bản dịch mẫu được tính theo công thức :

$$BLEU = BP * (\frac{1}{N} \sum_{i=1}^{i=N} \log p_i)$$

Trong đó :

$$BP = \begin{cases} 1 \text{ nếu } c>r \\ e^{(1-r/c)} \text{ với } c<=r \end{cases}$$ với c là độ dài của bản dịch máy, r là độ dài lớn nhất của các bản dịch mẫu

N là số lượng các bản dịch mẫu

Từ công thức trên, có thể thấy giá trị BLEU nằm trong khoảng 0 đến 1. Bản dịch nào có điểm BLEU càng cao, chứng tỏ độ trùng khớp giữa bản dịch máy và bản dịch mẫu càng nhiều, thì bản dịch đó càng chính xác.

CHƯƠNG 3: THỰC NGHIỆM

Để thực nghiệm, đề tài đã sử dụng bộ công cụ mã nguồn mở SRILM để xây dựng mô hình ngôn ngữ cho tiếng Việt.và hệ mã ngồn mở MOSES là bộ công cụ xây dựng mô hình dịch máy thống kê.

SRILM là bộ công cụ để xây dựng và áp dụng các mô hình ngôn ngữ thống kê, chủ yếu là để sử dụng trong nhận dạng tiếng nói, gắn thẻ thống kê và phân khúc, và dịch máy thống kê.

Moses là một hệ thống dịch máy thống kê cho phép người dùng xây dựng các mô hình dịch cho bất kỳ cặp ngôn ngữ nào với đầu vào là một tập hợp các văn bản song ngữ, được nhiều trường đại học, nhóm nghiên cứu nổi tiếng về xử lý ngôn ngữ tự nhiên và dịch máy thống kê như Edinburg (Scotland), RWTH Aachen (Germany),. .. tham gia phát triển. Đây là phần mềm có chất lượng khá tốt, khả năng mở rộng cao được dùng để xây dựng nhiều hệ thống dịch thử nghiệm cho nhiều cặp ngôn ngữ như Anh-Czech, Anh-Trung, Anh-Pháp,. .. Bộ công cụ Moses cho phép người dùng tạo ra một hệ thống dịch máy thống kê dựa trên cây cú pháp kết hợp với cụm từ một cách khá đơn giản.

3.1 Cài đặt hệ thống

Để cài đặt được chương trình này chúng tôi đã mất rất nhiều thời gian. Sau đây tôi xin tóm tắt quá trình cài đặt hệ thống.

3.1.1 Cấu hình và hệ điều hành.

- CPU Dual Core 1.73 GHz
- RAM 1G
- Hệ điều hành Ubuntu 11.04

3.1.2 Các công cụ sử dụng.

- Công cụ xây dựng mô hình ngôn ngữ: SRILM
- Công cụ gióng hàng từ: GIZA++
- Hệ dịch máy thống kê MOSES

3.1.3 Các bước huấn luyện dich và kiểm tra.

- Chuẩn hóa dữ lilệu
- Xây dựng mô hình ngôn ngữ
- Xây dựng mô hình dịch
- Hiệu chỉnh trọng số
- Dịch máy
- Đánh giá kết quả dịch

3.1.4 Chuẩn hóa dữ liệu.

- Tách từ
- Tách câu
- Chuyển sang chữ thường

3.1.5 Xây dựng mô hình ngôn ngữ.

- Sử dụng công cụ SRILM để xây dựng mô hình ngôn ngữ. Kết quả sau khi xây dựng mô hình ngôn ngữ tri-gam:

  \data

  ngram 1=18103

  ngram 2=219098

  ngram 3=144559

  \1-grams:

  -2.684004 ! -0.9555301

  -3.619983 " -0.2499142

  -4.59556 $ -0.1590119

3.1.6 Xây dựng mô hình dịch.

- Sử dụng GIZA++ là công cụ để gióng hàng để xây dựng mô hình dịch và dùng mkcls để ước lượng giá trị cực đại cho mỗi mô hình.

  Kết quả trong file extract

  i ||| tôi ||| 0-0

i have a sneaking suspicion ||| tôi nghi trong bụng ||| 0-0 1-1 1-3 2-3 3-3 1-4

i have a sneaking suspicion that ||| tôi nghi trong bụng là ||| 0-0 1-1 1-3 2-3 3-3 1-4 4-5

i have a sneaking suspicion that he ||| tôi nghi trong bụng là nó ||| 0-0 1-1 1-3 2-3 3-3 1-4 4-5 5-6

have a sneaking suspicion ||| nghi trong bụng ||| 0-0 0-2 1-2 2-2 0-3

have a sneaking suspicion that ||| nghi trong bụng là ||| 0-0 0-2 1-2 2-2 0-3 3-4

Kết quả trong file phrase-table

! ' !. ||| !. ||| 0.00389105 4.20299e-07 0.8 0.936597 2.718 ||| ||| 1028 5

! ' !. ||| ' !. ||| 1 0.0343045 0.2 0.75592 2.718 ||| ||| 1 5

! ' ! ||| ! ||| 0.00335852 4.21003e-07 0.8 0.972332 2.718 ||| ||| 1191 5

! ' ! ||| ' ! ||| 1 0.0343619 0.2 0.784762 2.718 ||| ||| 1 5

! ' ' indeed ? |||! ' ' thật không ? ||| 1 1.09941e-05 1 0.000411042 2.718 ||| ||| 1 1

3.1.7 Hiệu chỉnh trọng số.

- Sau khi xây dựng mô hình dịch ta có được bản Binary Phrase Table.
- Sau khi có được mô hình dịch và trọng bộ số tương ứng, tiến hành hiệu chỉnh trọng bộ số.Bước này mất rất nhiều thời gian.

3.1.8 Dịch máy.

- Bộ dữ liệu song ngữ Anh-Việt.
- Dịch câu tiếng Anh sang câu tiếng Việt.

3.1.9 Đánh giá kết quả dịch

- Chỉ số BLEU.

Evaluation of any-to-en translation using:

src set "conversation" (1 docs, 672 segs)

ref set "conversation" (1 refs)

tst set "conversation" (1 systems)

BLEU score = 0.1297 for system "ref"

- Chỉ số BLEU:Là chỉ số đánh giá chất lượng dịch của máy dịch thống kê từ ngôn ngữ này sang ngôn ngữ khác. Nếu kết quả gần giống với cách hiểu tự nhiên thì chất lượng dịch càng tốt. Điểm BLEU được tính bằng cách so sánh những câu cần dịch với một tập hợp các tham chiếu dịch tốt. Sau đó lấy ra giá trị trung bình tương ứng điểm số riêng lẻ này. Chỉ số này nằm trong khoảng 0 đến 1. Nếu càng gần1 thì chất lượng dịch càng tốt (sát nghĩa).

3.2 Bộ công cụ xây dựng mô hình ngôn ngữ - SRILM:

SRILM là bộ công cụ để xây dựng và áp dụng các mô hình ngôn ngữ thống kê, chủ yếu là để sử dụng trong nhận dạng tiếng nói, gắn thẻ thống kê và phân khúc, và dịch máy thống kê. Bộ công cụ này được phát triển bởi "Phòng thí nghiệm và nghiên cứu công nghệ giọng nói SRI" từ năm 1995, có thể chạy trên nền tảng Linux cũng như Windows.

SRILM bao gồm các thành phần sau:

- Một tập hợp các thư viện C++ giúp cài đặt mô hình ngôn ngữ, hỗ trợ cấu trúc dữ liệu và các chức năng tiện ích nhỏ.
- Một tập hợp các chương trình thực thi thực hiện nhiệm vụ xây dựng mô hình ngôn ngữ, đào tạo và thử nghiệm mô hình ngôn ngữ trên dữ liệu, gắn thẻ hoặc phân chia văn bản,...

Bộ công cụ SRILM có rất nhiều chương trình con, để xây dựng mô hình ngôn ngữ ta sử dụng 2 chương trình chính sau:

3.2.1 Ngram-count:

Chương trình Ngram-count thống kê tần số xuất hiện của các cụm Ngram. Kết quả của việc thống kê được ghi lại vào một tệp hoặc sử dụng chúng để xây dựng mô hình ngôn ngữ. Kết quả của việc thống kê được ghi lại theo định dạng sau:

\data\

ngram 1=*n1*

ngram 2=*n2*

...

ngram *N=nN*

\1-grams:

P w [bow]

...

\2-grams:

P w1 w2 [bow]

...

N-grams:

p w1. .. wN

...

\end\

Văn bản trên bắt đầu với một tiêu đề giới thiệu số lượng các cụm Ngram với chiều dài là 1, 2,. ..n. Tiếp theo, là từng đoạn chứa thông tin về các cụm Ngram có độ dài từ 1 đến n, mỗi đoạn bắt đầu bằng từ khóa *N*-grams: trong đó N là độ dài của các cụm Ngram được liệt kê ở bên dưới. Mỗi dòng tiếp theo của từng đoạn bắt đầu bằng một số thực là logarit cơ số 10 xác suất của cụm Ngram, tiếp theo là n từ w1, w2,. .. wn của cụm Ngram đó, và cuối cùng là trọng số truy hồi của cụm Ngram đó (có thể có)

Chương trình Ngram-count có một số tùy chọn chính sau:

- **text** *textfile*: thống kê tần số các cụm Ngram từ tệp văn bản đầu vào *textfile*. Tệp văn bản này có thể chứa mỗi câu trên một dòng. Kí hiệu kết thúc và bắt đầu dòng mới sẽ được tự động thêm vào nếu trong tệp đầu vào chưa có. Các dòng trống trong tệp này cũng bị loại bỏ.
- **order** *n* : thiết lập độ dài lớn nhất của các cụm Ngram sẽ thống kê bằng n. Giá trị mặc định nếu không thiết lập tham số này là n = 3
- **memuse:** hiển thị thông tin bộ nhớ mà chương trình sử dụng
- **lm** *lmfile*: xây dựng mô hình ngôn ngữ truy hồi từ các tần số vừa thống kê, sau đó ghi lại vào tệp *lmfile* theo định dạng ở trên.
- **gt*n*min** *count*: với *n* nhận các giá trị là 1, 2, 3, 4, 5, 6, 7, 8, hoặc 9. Tham số này thiết lập giá trị tần số nhỏ nhất với các cụm Ngram có độ dài là n. Tất cả các cụm ngram có độ dài là n, có tần số nhỏ hơn count sẽ bị loại bỏ khỏi mô hình ngôn ngữ

Và dưới đây là một số tham số thiết lập phương pháp làm mịn cho mô hình ngôn ngữ. Nếu không tham số nào dưới đây được thiết lập thì chương trình sẽ sử dụng phương pháp làm mịn Good-Turing.

- **wbdiscount*n*:** với *n* nhận các giá trị là 1, 2, 3, 4, 5, 6, 7, 8, hoặc 9. Sử dụng phương pháp làm mịn Witten-Bell cho N-gram với độ dài là *n*.
- **kndiscount*n*:** với *n* nhận các giá trị là 1, 2, 3, 4, 5, 6, 7, 8, hoặc 9. Sử dụng phương pháp làm mịn của Kneser-Ney được thay đổi bởi Chen và GoodMan cho N-gram với độ dài là *n*.
- **ukndiscount*n*:** với *n* nhận các giá trị là 1, 2, 3, 4, 5, 6, 7, 8, hoặc 9. Sử dụng phương pháp làm mịn của Kneser-Ney với độ dài là *n*.
- **addsmooth*n*** *delta*: với *n* nhận các giá trị là 1, 2, 3, 4, 5, 6, 7, 8, hoặc 9. Làm mịn bằng cách thêm một lượng *delta* vào tần số của tất cả các cụm Ngram với độ dài là *n*.
- **interpolate*n*:** với *n* nhận các giá trị là 1, 2, 3, 4, 5, 6, 7, 8, hoặc 9. Tính toán tần số của các cụm Ngram có độ dài là n bằng cách nội suy từ các cụm Ngram có độ dài nhỏ hơn.

3.2.2 Ngram:

Ngram là chương trình áp dụng mô hình ngôn ngữ để tính xác suất của một câu, tính toán độ hỗn loạn thông tin của văn bản, hay dùng để sinh các câu tiếp theo của một văn bản.

Chương trình Ngram có một số tùy chọn chính sau:

- **-order** *n* : thiết lập độ dài lớn nhất của các cụm Ngram sẽ thống kê bằng n. Giá trị mặc định nếu không thiết lập tham số này là n = 3
- **-memuse:** hiển thị thông tin bộ nhớ mà chương trình sử dụng
- **-lm** *file*: đọc mô hình ngôn ngữ từ tệp *file*. Tham số này là tham số bắt buộc, trừ khi tham số **-null** được chọn.
- **-null:** không sử dụng mô hình ngôn ngữ đọc từ tệp, mà sử dụng một mô hình ngôn ngữ đặt biệt (xác suất bằng 1 cho tất cả các từ). Tham số này thường được sử dụng trong việc gỡ lỗi.
- **-ppl** *textfile:* tính toán điểm(logarit cơ số 10 của xác suất) và độ hỗn loạn thông tin của tất cả các câu trong tệp *textfile*, mỗi câu viết trên một dòng.
- **-gen** *number*: sinh ngẫu nhiên *number* câu từ mô hình ngôn ngữ.

3.3 Bộ công cụ xây dựng mô hình dịch máy thống kê – MOSES:

Moses là một hệ thống dịch máy thống kê cho phép người dùng xây dựng các mô hình dịch cho bất kỳ cặp ngôn ngữ nào với đầu vào là một tập hợp các văn bản song ngữ, được nhiều trường đại học, nhóm nghiên cứu nổi tiếng về xử lý ngôn ngữ tự nhiên và dịch máy thống kê như Edinburg (Scotland), RWTH Aachen (Germany),. .. tham gia phát triển. Đây là phần mềm có chất lượng khá tốt, khả năng mở rộng cao được dùng để xây dựng nhiều hệ thống dịch thử nghiệm cho nhiều cặp ngôn ngữ như Anh-Czech, Anh-Trung, Anh-Pháp,. ..

Bộ công cụ Moses cho phép người dùng tạo ra một hệ thống dịch máy thống kê dựa trên cây cú pháp kết hợp với cụm từ một cách khá đơn giản. Hệ thống dịch máy thống kê này chứa một số thành phần sau:

- tệp **phrase-table:** tệp này chứa các cụm song ngữ theo định dạng:
- "cụm từ ở ngôn ngữ đích | cụm từ ở ngôn ngữ nguồn | xác suất"

Ví dụ:

! ' ' thật không ? ||| ! ' ' indeed ? ||| 1 1.09941e-05 1 0.000411042 2.718 ||| ||| 1 1

! ' ' thật ||| ! ' ' indeed ||| 1 6.44854e-05 1 0.000756693 2.718 ||| ||| 1 1

! ' ' vớ_vẩn ! họ ||| ! ' ' rot ! they ||| 1 5.64331e-05 1 0.0138612 2.718 ||| ||| 1 1

! ' ' vớ_vẩn ! ||| ! ' ' rot ! ||| 1 7.61009e-05 1 0.0228093 2.718 ||| ||| 1 1

! ' ' vớ_vẩn ||| ! ' ' rot ||| 1 7.96584e-05 1 0.0234583 2.718 ||| ||| 1 1

! ' ' ||| ! ' ' ||| 1 0.00135419 1 0.633375 2.718 ||| ||| 5 5

! ' ' Đó là một ||| ! ' ' it ' s a ||| 1 9.97436e-07 1 0.00306515 2.718 ||| ||| 1 1

- tệp **moses.ini** chứa các tham số cho bộ giải mã như: đường dẫn đến tệp **phrase-table,** đường dẫn đến tệp chứa mô hình ngôn ngữ, số lượng tối đa cụm từ của ngôn ngữ đích được dịch bởi một cụm từ của ngôn ngữ nguồn,. ...

Để xây dựng được mô hình dịch thống kê, ta có thể sử dụng script: **train-model.perl** với một số tham số sau:

- **--root-dir --** cài đặt thư mục gốc nơi lưu trữ các tệp đầu ra
- **--corpus --** tên của tệp văn bản huấn luyện (bao gồm cả 2 ngôn ngữ nguồn và đích)
- **--e --** đuôi mở rộng của tệp văn bản huấn luyện ngôn ngữ đích

- **--f --** đuôi mở rộng của tệp văn bản huấn luyện ngôn ngữ nguồn
- **--lm -- language model: <factor>:<order>:<filename>** : thiết lập file cấu hình mô hình ngôn ngữ theo định dạng đã trình bày trong phần **4.3.1**
- **--max-phrase-length --** độ dài lớn nhất của các cụm từ lưu trữ trong tệp **phrase-table**

Ví dụ, để xây dựng một mô hình dịch máy thống kê có chứa các tệp cấu hình tại thư mục hiện tại, tệp văn bản huấn luyện tên là "**corpus**", đuôi mở rộng của tệp ngôn ngữ nguồn tiếng Anh là. **en** còn đuôi mở rộng của tệp ngôn ngữ tiếng Việt là. **vn,** tệp chứa các thông số của mô hình ngôn ngữ là tệp "**lm.txt**", ta có thể sử dụng câu lệnh sau:

train-model.perl --root-dir. --f en --e vn --corpus corpus -lm 0:3:lm.txt

Sau khi sử dựng script trên để xây dựng mô hình dịch, ta sẽ có được các tệp cấu hình đã trình bày ở trên. Khi đó, để dịch một câu từ ngôn ngữ nguồn sang ngôn ngữ đích, ta có thể sử dụng câu lệnh như sau:

echo 'câu tiếng anh cần dịch" | moses -f moses.ini

Ví dụ dưới đây minh họa câu lệnh dùng để dịch câu "i love you" từ tiếng Anh sang tiếng Việt.

echo 'i love you' | moses -f moses.ini

Câu lệnh trên sẽ cho ra output trên màn hình là: "tôi yêu bạn". Để hiển thị rõ việc dịch các cụm từ, ta có thể thêm tham số -t vào câu lệnh trên, cụ thể:

echo "i love you' | moses -f moses.ini -t

Sau khi thực hiện câu lệnh trên, màn hình sẽ hiển thị đầu ra của quá trình dịch là: "tôi |0-0| yêu |1-1|bạn |2-2| ". Kết quả trên có thể hiểu là: cụm từ "i" ở vị trí 0(vị trí đầu tiên) trong câu được dịch thành cụm từ "tôi", cụm từ "love" ở vị trí 1 được dịch thành cụm từ "yêu", cụm từ "you" ở vị trí 2 được dịch thành cụm từ "bạn".

Muốn thực hiện việc dịch cho tệp văn bản, ta có thể thêm tham số -input-file tên_file_đầu_vào để thực hiện việc dịch tệp văn bản đó. Ngoài ra chương trình còn có rất nhiều tham số và chức năng khác, nhưng do giới hạn của luận văn nên chưa được trình bày ở đây.

3.4 Kết quả thực nghiệm khi đánh giá N-gram trong ứng dụng SMT

Sau khi xây dựng được mô hình Ngram với các phương pháp làm mịn khác nhau, chúng tôi sử dụng các mô hình Ngram đó vào mô hình dịch máy thống kê dịch từ tiếng Anh sang tiếng Việt. Bằng cách sử dụng các mô hình dịch máy thống kê đó dịch một đoạn văn bản tiếng Anh sang tiếng Việt sau đó tính điểm BLEU cho bản dịch, chúng tôi biết được phương pháp làm mịn nào là tốt nhất khi áp dụng trong mô hình dịch máy thống kê

Dữ liệu huấn luyện mô hình sử dụng tập văn bản song ngữ Anh-Việt với 54998 câu.

Dữ liệu dùng để điều chỉnh tham số sử dụng tập văn bản song ngữ Anh-Việt với 54998 câu.

Dữ liệu để kiểm tra sử dụng tập văn bản song ngữ Anh-Việt với 672 câu.

Dữ liệu để huấn luyện mô hình ngôn ngữ là tập văn bản đơn ngữ tiếng Việt với 7464 câu.

Sau khi xây dựng mô hình ngôn ngữ ta thu đươc bảng :

N-gram	Thông kê các cụm N-gram				
	Add-One	Witten Bell	Good Turing	Nội suy Kneser-Ney	Truy hồi Kneser-Ney
1-gram	6773	6773	6773	6773	6773
2-gram	162284	162284	162284	162284	162284
3-gram	92846	92846	92846	92846	92846

Bảng 3.1:Thống kê các cụm N-gram với các phương pháp làm mịn

N-gram	Độ đo BLEU trên các phương pháp làm mịn				
	Add-One	Witten Bell	Good Turing	Nội suy Kneser-Ney	Truy hồi Kneser-Ney
1-gram	0.0945	0.0957	0.0962	0.0965	0.0967
2-gram	0.0146	0.0159	0.0157	0.0158	0.0161
3-gram	0.0053	0.0063	0.0072	0.0075	0.0073

Bảng 3.2: Kết quả theo độ đo BLEU khi đánh giá SMT với các mô hình N-gram khác nhau

Từ kết quả ở bảng 3.2 chúng ta có thể rút ra một số nhận xét sau:

- Đối với các phương pháp chiết khấu sử dụng trong N-gram thì phương pháp Good-Turing cho kết quả BLEU tốt nhất đối với cả 1-gram, 2-gram, và 3-gram. Phương pháp Witten-Bell cho kết quả xấp xỉ Good-Turing, và vượt trội hẳn so với Add-One.
- Chúng ta không thấy sự khác biệt lớn của độ đo BLEU khi hệ thống SMT sử dụng các phương pháp *N*-gram đối với Good-Turing, Nội suy Kneser-Ney, và Truy hồi Kneser-Ney. Tuy vậy chúng ta cũng thấy kết quả tốt nhất đạt được với 3-gram và sử dụng Nội suy Kneser-Ney. Điều này phù hợp với các thực nghiệm N-gram độc lập rằng Nội suy Kneser-Ney cho kết quả tốt nhất.
- Phương pháp làm mịn GoodTuring là tốt nhất khi áp dụng cho mô hình ngôn ngữ sử dụng trong dịch máy thống kê.
- Có thể thấy rằng các kết quả này cho phép chúng ta kết luận các mô hình N-gram khác nhau có ảnh hưởng khác nhau tới chất lượng dịch của hệ SMT. Trong đó Good-Turing là phương pháp đơn giản nhưng đủ tốt để áp dụng cho SMT.

KẾT LUẬN

Trên đây chúng tôi đã khảo sát trên cả mặt lý thuyết và thực nghiệm đối với xây dựng mô hình ngôn ngữ cho tiếng Việt. Luận văn hướng tới mục tiêu xây dựng mô hình ngôn ngữ Ngram cho tiếng Việt. Trong khoảng thời gian nhất định dành cho thực hiện đề tài, nên một số vấn đề vẫn chưa hoàn chỉnh. Tuy nhiên, luận văn cũng đạt được một số kết quả:

- Về lý thuyết: Tìm hiểu, nghiên cứu mô hình ngôn ngữ, tìm hiểu các khó khăn còn tồn tại và phương pháp khắc phục, trong đó trọng tâm nghiên cứu các phương pháp làm mịn.
- Về thực nghiệm: Sử dụng bộ công cụ mã nguồn mở SRILM để xây dựng mô hình ngôn ngữ cho tiếng Việt. Sử dụng công cụ Moses để xây dựng mô hình dịch máy thống kê.

Do thời gian có hạn, nên hiện tại luận văn mới chỉ nghiên cứu được độ tin cậy của các phương pháp làm mịn trong mô hình ngôn ngữ khi áp dụng cho tiếng Việt và mô hình dịch máy thống kê từ tiếng Anh sang tiếng Việt. Trong tương lai, chúng tôi sẽ tiếp tục có những thống kê đầy đủ hơn.

TÀI LIỆU THAM KHẢO

Tài liệu tham khảo Tiếng Việt

[1]. **Huy Nguyễn Thạc.** *Tìm hiểu mô hình ngôn ngữ sử dụng phương pháp Bloom Filter.* Hà Nội : s.n., 2010.

[2]. **Lê Anh Cường, Cao Văn Việt, Nguyễn Việt Hà.** Xây dựng mô hình ngôn ngữ tiếng Việt và ứng dụng, Trường ĐH Công Nghệ, ĐHQG Hà Nội.

[3]. **Thắng Tô Hồng.** NGRAM. Trường đại học Công Nghệ, 2007.

[4]. **Cao Văn Việt,** Xây dựng mô hình ngôn ngữ cho tiếng Việt**,** Đồ án tốt nghiệp, Trường ĐH Công Nghệ, ĐHQG Hà Nội, 2009.

Tài liệu tham khảo Tiếng Anh

[1]. **Thắng Tô Hồng.** Building language model for vietnamese and its application, graduation thesis. 2008.

[2]. **Brown, P. F, Cocke J., Della Pietra V., Della Pietra S., Jelinek F., Lafferty J. D., Mercer R. L., and Roossin P. S.** *A statistical approach to machine translation.* s.l. : Computational Linguistics, 1990.

[3]. **Chen, S. and Goodman, J.** *An empirical study of smoothing tech-niques for language modeling.* s.l. : Computer Speech & Language, 1999.

[4]. **Kneser, R. and Ney, H.** *Improved backing-off for m-gram language modelling.* s.l. : In Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing, 1995.

[5]. **MacCartney, Bill.** *NLP Lunch Tutorial: Smoothing.* 21 April 2005.

Tài liệu tham khảo trực tuyến

[1]. BLEU. [Online] http://en.wikipedia.org/wiki/BLEU.

[2]. Moses. [Online] http://www.statmt.org/moses/.