ÁP DỤNG HỌC MÁY DỰA TRÊN LẬP TRÌNH DI TRUYỀN TRONG TÌM KIẾM WEB XUYÊN NGỮ

LEARNING TO RANK BASED ON GENETIC PROGRAMMING FOR CROSS-LANGUAGE WEB SEARCH

Lâm Tùng Giang[1], Võ Trung Hùng[2], Huỳnh Công Pháp[3]

[1]Văn phòng UBND thành phố Đà Nẵng; gianglt@gmail.com
[2]Đại học Đà Nẵng; vthung@dut.udn.vn
[3]Trường Cao đẳng Công nghệ thông tin, Đại học Đà Nẵng; phaphc@gmail.com

Tóm tắt - Hầu hết các nghiên cứu trong lĩnh vực truy vấn thông tin xuyên ngữ giới hạn xem xét các tài liệu văn bản và chú trọng xử lý vấn đề dịch thuật. Trong bài báo này, chúng tôi đề xuất áp dụng học xếp hạng dựa trên kỹ thuật lập trình di truyền nhằm tăng hiệu quả của hệ thống tìm kiếm web xuyên ngữ. Cụ thể, chúng tôi đề xuất 2 phương pháp xây dựng các hàm xếp hạng mới dưới dạng tổ hợp tuyến tính của các hàm xếp hạng cơ sở. Đồng thời, chúng tôi cũng đề xuất 2 mô hình xếp hạng lân cận, ứng dụng trong truy vấn xuyên ngữ. Trong thí nghiệm với một hệ thống tìm kiếm web xuyên ngữ Việt-Anh, điểm số MAP trung bình sử dụng phương pháp kiểm định 5-thư mục của các mô hình đề xuất là 0,4640 và 0,4585, vượt trội so với điểm MAP 0,3742 của cấu hình cơ sở - sử dụng bản dịch thủ công.

Từ khóa - tìm kiếm xuyên ngữ; lân cận; xếp hạng lại; học xếp hạng; lập trình di truyền; tìm kiếm web.

Abstract - Most studies in the field of Cross-Language Information Retrieval consider the documents as plain texts and mainly focus on translation problems. In this article, we follow the learning to rank approach based on Genetic Programming to improve ranking performance of a cross-language web search system. We also introduce 2 proximity models, applied in cross-language information retrieval. We propose linear combinations of weak rankers for re-ranking the retrieved documents. In our experiment with a Vietnamese - English cross-language web search system, the performance measured by the MAP score and reported by a 5-fold cross validation of proposed models is 0.4640 and 0.4585. These results outperform the MAP score of 0.3742 given by the baseline configuration, using the manual translation.

Key words - Cross Language Information Retrieval (CLIR); proximity; re-ranking; learning to rank; Genetic Programming; web search.

1. Đặt vấn đề

Với khối lượng khổng lồ các tài liệu trực tuyến trên World Wide Web và số lượng ngày càng tăng người sử dụng từ các quốc gia khác nhau, truy vấn thông tin xuyên ngữ (Cross-Language Information Retrieval hay CLIR) trở thành một công cụ hữu hiệu với vai trò giúp người sử dụng vượt qua các rào cản ngôn ngữ để truy cập thông tin được viết bằng các ngôn ngữ khác nhau. Trọng tâm nghiên cứu trong lĩnh vực CLIR là vấn đề dịch thuật [1]. Cách tiếp cận phổ biến trong CLIR là dịch câu truy vấn từ ngôn ngữ nguồn (ngôn ngữ câu truy vấn) sang ngôn ngữ đích (ngôn ngữ tài liệu) bằng các kỹ thuật khác nhau như sử dụng từ điển, kho ngữ liệu song song, sử dụng máy dịch hay dựa trên ontology. Bản dịch câu truy vấn sau đó được sử dụng để tìm kiếm các văn bản ở ngôn ngữ đích.

Do sự hạn chế của các kho ngữ liệu song song và các ontology, phương pháp sử dụng từ điển là cách tiếp cận phổ biến nhằm xây dựng các hệ thống truy vấn CLIR có liên quan tiếng Việt [2]. Phân tích các kết quả tại các nghiên cứu cho thấy một số vấn đề. Thứ nhất, mô-đun dịch thuật độc lập với hệ thống truy vấn. Thứ hai, các nghiên cứu giới hạn xem xét các tài liệu văn bản (plain text). Điều này không phản ánh thực tế là người sử dụng thường tìm kiếm các tài liệu web với cấu trúc HTML.

Trong bài này, chúng tôi đề xuất một cách tiếp cận mới nhằm tăng hiệu năng của các hệ thống tìm kiếm Web xuyên ngữ. Chúng tôi sử dụng các thông tin kết xuất trong quá trình dịch cũng như khai thác cấu trúc của các tài liệu Web để định nghĩa các hàm xếp hạng cơ sở. Trong số này, 2 hàm xếp hạng xấp xỉ lần đầu tiên được giới thiệu và áp dụng cho truy vấn thông tin xuyên ngữ. Chúng tôi đề xuất 2 mô hình học máy dựa trên lập trình di truyền nhằm xây dựng hàm xếp hạng kết quả tìm kiếm dưới dạng một tổ hợp tuyến tính của các hàm xếp hạng cơ sở.

Bài báo được tổ chức như sau: phần 2 trình bày cơ sở lý thuyết. Trong phần 3, chúng tôi trình bày cách xây dựng các hàm xếp hạng lân cận, 2 phương án học xếp hạng và thiết kế hệ thống tìm kiếm web xuyên ngữ. Phần 4 trình bày nội dung thử nghiệm và phần 5 tổng kết, đánh giá kết quả nghiên cứu.

2. Cơ sở lý thuyết

Trong phần này, chúng tôi trình bày cơ sở lý thuyết liên quan mô hình xếp hạng OKAPI BM25, lĩnh vực học máy, giới thiệu các kết quả nghiên cứu về mô hình xếp hạng lân cận và xếp hạng trang Web.

2.1. Mô hình xếp hạng OKAPI BM25

OKAPI BM25 là một mô hình truy vấn xác suất dựa trên mô hình nhị phân độc lập. Mô hình này sử dụng lần xuất hiện của từ khóa trong tài liệu, độ dài tài liệu để tính trọng số các từ khóa trong tài liệu và trong câu truy vấn. Với một từ khóa t_i trong tài liệu d, trọng số w_i tương ứng với t_i có thể được tính như sau [3]:

$$w_i = (k_1 + 1)\frac{tf_i}{K + tf_i} \quad (1)$$

trong đó:

$$K = k \cdot ((1 - b) + b \cdot \frac{l}{avdl})$$

với l là độ dài tài liệu, $avdl$ là độ dài trung bình của các tài liệu, b là hằng số (gán giá trị 0.9), k là hằng số (được gán

giá trị 2), *k1* là hằng số (được gán giá trị 1.2) và tf_i là tần suất xuất hiện của từ khóa t_i trong tài liệu *d*.

Với từ khóa t_i trong câu truy vấn *q*, một hàm khác được áp dụng:

$$qw_i = \frac{qtf_i}{k3 + qtf_i} \cdot \log\left(\frac{n - df_i}{df_i}\right) \quad (2)$$

trong đó qtf_i là tần suất xuất hiện của từ khóa t_i trong câu truy vấn, df_i là số tài liệu chứa từ khóa t_i, *n* là số tài liệu trong kho tài liệu và *k3* là hằng số (được gán giá trị 1000). Điểm số của tài liệu *d* đối với câu truy vấn *q* khi đó được tính bằng công thức sau:

$$score_{okapi}(d, q) = \sum_i w_i \cdot qw_i \quad (3)$$

2.2. Học máy

Các hàm xếp hạng đóng vai trò trung tâm trong các hệ thống truy vấn thông tin và chịu trách nhiệm gán điểm cho các tài liệu trong kho tài liệu, sau đó sắp xếp các tài liệu theo thứ tự giảm dần của điểm. Các mô hình xếp hạng phổ biến bao gồm TF-IDF, BM25, LSI. Cho trước một danh sách các mô hình xếp hạng cơ sở, xếp hạng tổng hợp là quá trình kết hợp các hàm xếp hạng cơ sở để xây dựng một hàm xếp hạng mới, cho kết quả xếp hạng tốt hơn. Các phương pháp truyền thống như phương pháp Borda hay CombMNZ kết hợp thứ tự xếp hạng của các hàm xếp hạng cơ sở [4].

Gần đây, xu hướng học xếp hạng (Learning-to-Rank hay L2R) xuất hiện, kết nối các kết quả nghiên cứu trong các lĩnh vực học máy, truy vấn thông tin và xử lý ngôn ngữ tự nhiên. Các phương pháp học máy sử dụng dữ liệu huấn luyện và xây dựng mô hình xếp hạng mới dựa trên dữ liệu huấn luyện. Một số thuật toán học xếp hạng có giám sát bao gồm PRank, Rank SVM, RankNet, LamdaRank, LamdaMart, ListNet với chi tiết có thể xem tại [5].

Lập trình di truyền, được giới thiệu tại [6], là một kỹ thuật tính toán dựa trên thuyết tiến hóa, cho phép tìm thấy chương trình máy tính tối ưu được tạo để giúp con người giải quyết một vấn đề cụ thể. Trong lập trình di truyền, mỗi giải pháp tiềm năng (ví dụ một hàm xếp hạng) là một cá thể trong một quần thể các giải pháp. Các phương pháp tái sinh, lai ghép, đột biến được áp dụng qua một số thế hệ tiến hóa. Quá trình tiến hóa sẽ giúp xác định cá thể với độ thích nghi cao nhất, được coi là giải pháp tối ưu. Một số nghiên cứu cho thấy lập trình di truyền có thể được áp dụng đối với bài toán học xếp hạng [7], [8].

2.3. Mô hình xếp hạng lân cận

Trong các mô hình truy vấn thông tin truyền thống, các tài liệu được biểu diễn như "túi từ" (bags of words) và được tính điểm dựa trên các chỉ số thống kê như: tần suất từ, tần suất nghịch đảo của tài liệu, độ dài tài liệu. Một hạn chế của các mô hình này là chúng không khai thác mối liên hệ giữa các từ khóa cùng xuất hiện trong câu truy vấn. Một cách cảm quan, nếu một tài liệu chứa các thuật ngữ truy vấn đứng gần nhau, tài liệu đó được xếp hạng trên một tài liệu khác cũng chứa các thuật ngữ này, nhưng với khoảng cách xa hơn.

Mô hình hóa xếp hạng lân cận là một xu hướng nghiên cứu trong truy vấn thông tin, tích hợp yếu tố lân cận của các từ khóa trong hàm xếp hạng. Hai xu hướng phổ biến bao gồm dựa trên đoạn (span-based) và dựa trên cặp từ (pair-based). Trong xu hướng thứ nhất, span được định nghĩa như một đoạn văn bản chứa tất cả các từ khóa truy vấn trong tài liệu. Điểm lân cận của một tài liệu tương ứng với một câu truy vấn tỷ lệ thuận với số span và tỷ lệ nghịch với độ dài của span [9]. Trong xu hướng thứ hai, các tác giả đưa ra các công thức khác nhau để tính điểm lân cận cho từng cặp từ trong tài liệu, sau đó tính điểm lân cận của tài liệu bằng cách cộng dồn các điểm lân cận của tất cả các cặp từ khóa truy vấn xuất hiện trong tài liệu [3]. Một mô hình lân cận có thể được áp dụng để xếp hạng lại các tài liệu truy vấn sau lần tìm đầu tiên, hoặc có thể được xây dựng trong quá trình chỉ mục hóa văn bản.

Các hàm xếp hạng lân cận đóng vai trò hỗ trợ, cho phép tăng thứ hạng của các tài liệu phù hợp, nếu trong nội dung chứa các từ khóa truy vấn có khoảng cách gần nhau. Trong bài báo này, chúng tôi đề xuất 2 hàm xếp hạng lân cận ứng dụng cho truy vấn xuyên ngữ và sử dụng như các hàm xếp hạng cơ sở để học hàm xếp hạng mới.

2.4. Xếp hạng trang Web

Truy vấn thông tin trên Web có sự khác biệt so với truy vấn thông tin truyền thống, sử dụng chủ yếu cho các hệ thống thư viện. Một tài liệu HTML chứa các thành phần khác nhau như tiêu đề, tóm tắt, nội dung. Nó cũng thường chứa các thành phần đặc biệt như liên kết, neo, thẻ meta. Các thành phần có ảnh hưởng khác nhau trong tìm kiếm. Ví dụ, tài liệu với từ khóa tìm kiếm xuất hiện trong tiêu đề thường được giả định là phù hợp hơn so với tài liệu có từ khóa chứa trong thân bài. Tại [10], các tác giả đã phân tích cấu trúc tài liệu HTML để xây dựng các chỉ mục riêng cho các thành phần tiêu đề, thân bài, neo. Tại [11], các thành phần khác nhau của một tài liệu HTML được định nghĩa dựa trên cách xuất hiện, nội dung và đề xuất một số tổ hợp tuyến tính của các điểm số. Tại [12], các tác giả định nghĩa các đặc tính của trang web, đồng thời đưa ra khái niệm *Class Importance Vector* để gán mức độ quan trọng đối với mỗi đặc tính và áp dụng phương pháp heuristic để kết hợp các hàm xếp hạng. Trong bài báo này, chúng tôi đề xuất sử dụng phương pháp học máy nhằm "học" tổ hợp tuyến tính kết hợp các hàm xếp hạng cơ sở.

3. Giải pháp đề xuất

Trong phần này, chúng tôi trình bày thiết kế của một hệ thống tìm kiếm web xuyên ngữ cho cặp ngôn ngữ tiếng Việt và tiếng Anh. Tại bài báo này, chúng tôi định nghĩa các hàm xếp hạng cơ sở và áp dụng kỹ thuật học xếp hạng để xây dựng một hàm xếp hạng mới.

3.1. Mô hình học xếp hạng

Trong bài báo, 2 mô hình học xếp hạng dựa trên lập trình di truyền được đề xuất nhằm "học" hàm xếp hạng dưới dạng tổ hợp tuyến tính của các hàm xếp hạng cơ sở. Mô hình thứ nhất sử dụng dữ liệu huấn luyện chứa điểm số gán cho các thành phần trong các tài liệu HTML và nhãn xác định tài liệu có phù hợp hay không so với câu truy vấn. Mô hình thứ hai chỉ sử dụng điểm số gán cho các thành phần trong các tài liệu HTML, sau đó so sánh thứ tự xếp hạng của các hàm ứng viên so với các hàm xếp

hạng cơ sở. Nội dung tiếp theo mô tả các thành phần của các mô hình học máy dựa trên lập trình di truyền.

3.1.1. Cá thể

Với một tập n hàm xếp hạng cơ sở $F_0, F_1,\ldots, F_n$, mỗi cá thể được xem xét có dạng một hàm tuyến tính f kết hợp các hàm xếp hạng cơ sở:

$$f(d) = \sum_{i=0}^{n} \alpha_i F_i(d) \quad (4)$$

Với α_i là các số thực, d là tài liệu cần gán điểm. Mục đích của chúng ta là xác định hàm f cho kết quả xếp hạng tốt nhất.

3.1.2. Hàm phù hợp

Hàm phù hợp (fitness function) xác định mức độ thích nghi của mỗi cá thể. Hàm phù hợp được sử dụng trong mô hình học xếp hạng có giám sát được đề xuất là giá trị MAP (Mean Average Precision) và được tính toán bằng thuật toán 1:

Thuật toán 1: tính độ phù hợp (có giám sát)

Input: Hàm ứng viên *f,* tập các câu truy vấn *Q*
Output: mức độ phù hợp của hàm *f*
n = 0; sap = 0;
for each câu truy vấn *q* **do**
n+=1;
tính điểm mỗi tài liệu bởi hàm xếp hạng *f;*
ap = độ chính xác trung bình cho hàm xếp hạng *f;*
sap += ap;
end
map = sap/n
return *map*

Hàm phù hợp được sử dụng trong mô hình học xếp hạng không giám sát được xây dựng dựa trên ý tưởng được trình bày tại [13] về sự thống nhất giữa các hàm xếp hạng. Gọi *r(i,d,q)* là thứ hạng của tài liệu *d* trong danh sách kết quả tìm kiếm bằng câu truy vấn *q*, sử dụng hàm xếp hạng F_i, *rf(d,q)* là thứ hạng của tài liệu *d* trong danh sách kết quả tìm kiếm bằng câu truy vấn *q*, sử dụng hàm xếp hạng *f,* thuật toán được trình bày như sau:

Thuật toán 2: tính độ phù hợp (không giám sát)

Input: Hàm ứng viên *f,* tập các câu truy vấn *Q*
Output: mức độ phù hợp của hàm *f*
s_fit = 0;
for each câu truy vấn *q* **do**
tính điểm mỗi tài liệu bởi hàm xếp hạng *f;*
D = tập hợp 200 tài liệu đứng đầu*;*
for each tài liệu *d* **in** *D* **do**
k+=1;d_fit = 0;
for *i=0 to n* **do**
d_fit +=distance(i,k,q)
end
s_fit += d_fit
end
end
return *s_fit*

Chúng tôi thử nghiệm 3 phương án của hàm *distance(i,k,q)* được sử dụng trong thuật toán 2 như sau:

Bảng 1. *Các phương án hàm distance*

	distance(i,k,q)
1	*abs(r(i,d,q)-rf(d,q))*
2	*abs(r(i,d,q)-rf(d,q))/log(k+1)*
3	*(r(i,d,q)-rf(d,q))/ k*

3.1.3. Quá trình huấn luyện

Quá trình huấn luyện được thực hiện như sau:

Thuật toán 3: Học xếp hạng

Input: *Ng* = số thế hệ**,** *Np:* kích thước quần thể**,** *Nc***:** tốc độ lai ghép**,** *Nm***:** tốc độ đột biến
Output:
Tạo lập quần thể đầu tiên, mỗi cá thể có dạng hàm tuyến tính của các hàm *F0, F1,...,Fn*
Thực hiện những tác vụ sau *Ng* thế hệ*:*
Với mỗi cá thể tính giá trị hàm phù hợp;
Chọn cá thể có giá trị hàm phù hợp tốt nhất;
Tạo quần thể mới bằng cách thực hiện các hàm tái sinh, lai ghép, đột biến với tốc độ tương ứng;
F_best = cá thể tốt nhất;
return *f_best*

Với mô hình học máy giám sát, cá thể tốt nhất có giá trị *map* trả về cao nhất; với mô hình học máy không giám sát, đó là cá thể có giá trị trả về *s_fit* nhỏ nhất.

3.2. Hàm xếp hạng lân cận

Chúng tôi định nghĩa hai hàm xếp hạng lân cận áp dụng cho truy vấn thông tin xuyên ngữ, được sử dụng như các hàm xếp hạng cơ sở phục vụ trong quá trình học xếp hạng mô tả tại mục 3.1.

3.2.1. Hàm xếp hạng CL-Rasolofo

Hàm xếp hạng lân cận *CL-Rasolofo* được đề xuất dựa trên ý tưởng trình bày tại [3]. Trong bài báo này, chúng tôi phát triển phiên bản hàm xếp hạng lân cận áp dụng cho truy vấn thông tin xuyên ngữ. Với một đoạn văn bản s và một cặp từ khóa (t_i,t_j), hàm khoảng cách cặp từ *tpi* được định nghĩa như sau:

$$tpi(t_i, t_j, s) = \frac{1}{dis(t_i, t_j, s)} \quad (5)$$

Ở đây, $dis(t_i, t_j, s)$ là khoảng cách giữa 2 từ khóa t_i và t_j trong đoạn văn bản s. Từ đây, với cặp từ v_1, v_2 trong câu truy vấn ở ngôn ngữ nguồn và đoạn văn bản s ở ngôn ngữ đích, chúng tôi đề xuất phiên bản xuyên ngữ của hàm khoảng cách cặp từ như sau:

$$tpi(v_1, v_2, s) = \sum_{t_i \in T(v_1), t_j \in T(v_2)} tpi(t_i, t_j, s) \quad (6)$$

Ở đây, $T(v_i)$ là tập hợp các phương án dịch của v_i.

Với tập hợp *Sents* bao gồm tất cả các câu trong tài liệu d, hàm lân cận của 2 từ khóa v_1 và v_2 được định nghĩa:

$$fn(v_1, v_2) = \sum_{s \in Sents} tpi(v_i, v_j, s) \quad (7)$$

Đối với tập $T(w)$ chứa tất cả các bản dịch ứng viên của thuật ngữ w, hàm $idf(T(w))$ được định nghĩa như sau:

$$idf(T(w)) = \log\left(\frac{N + 0.5}{n(T(w)) + 0.5}\right) \quad (8)$$

trong đó N là tổng số tài liệu, $n(T(w))$ là số tài liệu chứa ít nhất một từ trong tập hợp $T(w)$.

Cuối cùng, với câu truy vấn q ở ngôn ngữ nguồn và tài liệu d ở ngôn ngữ đích, hàm xếp hạng lân cận xuyên ngữ *CL-Rasolofo* được định nghĩa bằng công thức:

$$score_{CL-Rasolofo} = \sum_{v_i, v_j \in q; v_i \neq v_j} m(i,j) \frac{fn(v_i, v_j)(k1 + 1)}{K + fn(v_i, v_j)} \quad (9)$$

Ở đây, $m(i,j)$ là giá trị nhỏ nhất giữa hai giá trị $idf(T(v_i))$ và $idf(T(v_j))$. Các giá trị $k1$ và K có được tính toán tương tự như tại mục 2.1 của bài báo.

3.2.2. Hàm xếp hạng CL-HighDensity

Hàm xếp hạng lân cận *CL-HighDensity* được định nghĩa dựa trên việc xem xét các câu trong tài liệu chứa nhiều từ khóa truy vấn. Cụ thể, ký hiệu $S(text)$ là tập hợp các câu trong văn bản *text*, $S_{density}(text)$ là tập con của $S(text)$, bao gồm các câu chứa bản dịch của ít nhất 2 từ khóa truy vấn. $text_{density}$ là văn bản mới tạo bằng cách nối các câu trong $S_{density}(text)$. Hàm xếp hạng lân cận được định nghĩa bằng công thức:

$$score_{CL-HighDensity}(d, q) = score_{okapi}(text_{density}, q) \quad (10)$$

trong đó, $score_{okapi}$ được tính dựa trên mô hình xếp hạng OKAPI BM25 như ở công thức (3).

3.3. Thiết kế hệ thống

Hệ thống tìm kiếm web xuyên ngữ sử dụng kết quả trình bày tại [14]. Từ câu truy vấn tiếng Việt q_v, mô-đun dịch thuật tạo câu truy vấn tiếng Anh dưới dạng:

$$q_e = (e_{1,1}w_{1,1}\ OR \ldots OR\ e_{1,n1}w_{1,n1})\ AND\ \ldots\ (e_{n,1}w_{n,1}\ OR \ldots OR\ e_{n,nm}w_{1,nm})\ AND\ t_1w_1 \ldots t_nw_n \quad (11)$$

Ở đây, với mỗi giá trị i, $T(v_i) = \{e_{i,1}, \ldots, e_{i,ni}\}$ là tập hợp các phương án dịch của từ khóa v_i, được gán với trọng số $\{w_{i,1}, \ldots, w_{i,ni}\}$; $\{t_1, \ldots, t_m\}$ là tập hợp các từ khóa mở rộng với các trọng số $\{w_1, \ldots w_m\}$.

Chúng tôi sử dụng máy tìm kiếm đơn ngữ tiếng Anh trên nền tảng công cụ mã nguồn mở Solr, sử dụng mô hình xếp hạng TF-IDF và cho phép đánh chỉ mục trên nhiều trường. Mỗi tài liệu web trong kho tài liệu được bóc tách các thành phần tiêu đề (tương ứng thẻ <TITLE>) và nội dung (tương ứng thẻ <BODY>) của mỗi tài liệu. Thành phần tóm tắt của mỗi tài liệu được định nghĩa như phần văn bản tương ứng thẻ <H2>. Nếu không có thẻ này, 200 ký tự đầu tiên của nội dung được coi như phần tóm tắt.

Có 4 hàm xếp hạng cơ sở F_0, F_1, F_2, F_3 được tạo, gán điểm cho các tài liệu tương ứng với điểm của máy tìm kiếm đơn ngữ tiếng Anh cho các tài liệu khi thực hiện tìm kiếm giới hạn theo các trường khác nhau của trang web: toàn văn, tiêu đề, tóm tắt và nội dung. Bên cạnh đó, 3 mô hình xếp hạng khác cũng được áp dụng, bao gồm mô hình *BM25*, các mô hình xếp hạng lân cận *CL-Rasolofo* và *CL-HighDensity* được định nghĩa tại mục 3.1, áp dụng cho các thành phần của trang Web. Cụ thể, ta có các hàm F_4, F_5 và F_6 gán điểm cho tài liệu tương ứng điểm tìm kiếm theo tiêu đề, tóm tắt và nội dung sử dụng mô hình xếp hạng BM25; các hàm F_7, F_8, F_9 sử dụng điểm xếp hạng lân cận *CL-Rasolofo* và các hàm F_{10}, F_{11}, F_{12} sử dụng điểm xếp hạng lân cận *CL-HighDensity*. Tổng cộng, ta thu được 12 hàm xếp hạng cơ sở $F_0, \ldots, F_{12}$.

Với câu truy vấn có cấu trúc được tạo bởi mô-đun dịch, mô-đun *Querying-By-Fields* được thực hiện để tính toán giá trị các hàm F_0, F_1, F_2, F_3 và tải về danh sách 200 tài liệu xếp hạng cao nhất tương ứng là L_0, L_1, L_2, L_3. Một danh sách tài liệu Lm được tạo từ tất các danh sách này. Với các tài liệu trong L_m, mô-đun *Additional-Ranking* gán điểm cho các tài liệu, sử dụng mô hình BM25 và các mô hình xếp hạng lân cận *CL-Rasolofo* và *CL-HighDensity*. Mô-đun *Learning-To-Rank* học các tham số để tạo tổ hợp tuyến tính của các hàm xếp hạng cơ sở, từ đó tạo hàm xếp hạng cuối cùng, được sử dụng bởi mô-đun *Joint-Ranking* để sắp xếp lại các tài liệu.

4. Kết quả thử nghiệm

4.1. Cấu hình thử nghiệm

Thí nghiệm sau được triển khai nhằm đánh giá các mô hình học xếp hạng đề xuất. Đầu tiên, 24000 tài liệu tiếng Anh được thu thập từ Web. Các tài liệu được đánh chỉ mục như mô tả tại mục 3.3. Tổng cộng 50 câu truy vấn tiếng Việt với độ dài 8,73 từ được sử dụng để kiểm tra hiệu năng của hệ thống. Phương pháp *pooling* [15] được sử dụng để xây dựng bộ dữ liệu kiểm thử. Thuật toán học xếp hạng đề xuất được kiểm tra với các cấu hình sau:

Cấu hình ***baseline***: các câu truy vấn được dịch thủ công, điểm xếp hạng tương ứng với điểm xếp hạng khi tìm kiếm toàn văn.

Cấu hình ***google***: các câu truy vấn được dịch bằng cách sử dụng máy dịch Google, điểm xếp hạng tương ứng với điểm xếp hạng khi tìm kiếm toàn văn.

Cấu hình ***SQ***: sử dụng bản dịch có cấu trúc.

Cấu hình ***SC***: kết quả học xếp hạng có giám sát.

Các cấu hình ***UC1, UC2, UC3***: kết quả học xếp hạng không giám sát, tương ứng với 3 cấu hình hàm phù hợp định nghĩa tại Bảng 1.

4.2. Kết quả thử nghiệm

Bảng 2 mô tả kết quả thử nghiệm, trong đó cột 2 thể hiện trung bình điểm số MAP (Mean Average Prevision), sử dụng phương pháp kiểm định 5 thư mục (5-fold validation).

Bảng 2. *Kết quả thử nghiệm*

Cấu hình	Giá trị MAP
Baseline	0.3742
Google	0.3548
SQ	0.4307
SC	0.4640
UC1	0.4284
UC2	0.4394
UC3	0.4585

Cấu hình *SC* cho điểm MAP cao nhất 0.4640, bằng 124% so với cấu hình cơ sở. Tuy áp dụng phương pháp học máy không giám sát, cấu hình *UC3* cũng cho kết quả điểm MAP là 0,4585; cao hơn cấu hình *SQ* 6,4% và cấu hình *baseline* 17,4%.

5. Kết luận

Trong bài báo, chúng tôi đề xuất 2 mô hình học xếp hạng dựa trên lập trình di truyền nhằm xếp hạng lại kết quả tìm kiếm các trang Web trong hệ thống tìm kiếm Web xuyên ngữ. Đóng góp của bài báo là việc đề xuất 2 mô hình xếp hạng lân cận *CL-Rasolofo* và *CL-HighDensity* áp dụng trong tìm kiếm xuyên ngữ. Bên cạnh đó, chúng tôi cũng đề xuất bóc tách và đánh chỉ mục các thành phần nội dung trong trang web trong máy tìm kiếm nhằm định nghĩa tập hợp các hàm xếp hạng cơ sở, được sử dụng trong quá trình học xếp hạng dựa trên lập trình di truyền phục vụ tìm kiếm hàm xếp hạng mới dưới dạng tổ hợp tuyến tính của các hàm xếp hạng cơ sở.

TÀI LIỆU THAM KHẢO

[1] Dong Zhou, Mark Truran, Tim Brailsford, Vincent Wade, and Helen Ashman, "Translation techniques in cross-language information retrieval", *ACM Computing Surveys*, 45(1), 2012, pp. 1–44.

[2] Nguyen Han Doan, "Vietnamese-English Cross-language information retrieval (CLIR) using bilingual dictionary", *International Workshop on Advanced Computing and Applications Ho Chi Minh City*, 2007.

[3] Yves Rasolofo and Jacques Savoy, "Term Proximity Scoring for Keyword-Based Retrieval Systems", *Lecture Notes in Computer Science*, 2003, pp. 207–218.

[4] Sibel Adali, Brandeis Hill, and Malik Magdon-Ismail, "Information vs. Robustness in rank aggregation: Models, algorithms and a statistical framework for evaluation", *Journal of Digital Information Management*, 5(5), 2007, pp. 292–308.

[5] T. Y. Liu, *Learning to rank for information retrieval*. Springer, 2011.

[6] John R. Koza, "The genetic programming paradigm: Genetically breeding populations of computer programs to solve problems", *Dynamic, Genetic, and Chaotic Programming*, 1992, pp. 203–321.

[7] Li Wang, Weiguo Fan, Rui Yang, Wensi Xi, Ming Luo, Ye Zhou, and Edward a Fox, "Ranking Function Discovery by Genetic Programming for Robust Retrieval", *NIST Special Publication 500-255: The Twelfth Text REtrieval Conference (TREC 2003)*, 2003, pp. 828–836.

[8] Weiguo Fan Weiguo Fan, M. D. Gordon, P. Pathak, Wensi Xi Wensi Xi, and E. a. Fox, "Ranking function optimization for effective Web search by genetic programming: an empirical study", *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*, 2004, pp. 105–112.

[9] K. M. Svore, P. H. Kanani, and N. Khan, "How Good is a Span of Terms? Exploiting Proximity to Improve Web Retrieval", *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 2010, pp. 154–161.

[10] M. Cutler, Y. Shi, and W. Meng, "Using the Structure of HTML Documents to Improve Retrieval", *Proceedings of the USENIX Symposium on Internet Technologies and Systems: December 8--11, 1997, Monterey, California*, 1997, pp. 241–252.

[11] H. Yunhua, H. Yunhua, X. Guomao, X. Guomao, S. Ruihua, S. Ruihua, H. Guoping, and H. Guoping, "Title Extraction from Bodies of HTML Documents and its Application to Web Page Retrieval", *on Research and Development in Information Retrieval*, 2005, pp. 250–257.

[12] Manjit Singh, Dheerendra Singh, and Surender Singh, "Use of HTML Tags in Web Search", 8(2), 2015, pp. 8–14.

[13] Alexandre Klementiev, Dan Roth, and Kevin Small, "An Unsupervised Learning Algorithm for Rank Aggregation", *Proceedings of European Conference on Machine Learning*, 2007.

[14] Lam Tung Giang, Vo Trung Hung, and Huynh Cong Phap, "Improve Cross Language Information Retrieval with Pseudo-Relevance Feedback", *FAIR*, 2015, pp. 315–320.

[15] K. Sparck Jones and C. J. Van Rijsbergen, "Information retrieval test collections", *Journal of Documentation*, 32(1), 1976, pp. 59–75.

(BBT nhận bài: 09/12/2015, phản biện xong: 25/12/2015)