Tạp chí Tin học và Điều khiển học, T.24, S.1 (2008), 62-74

MỘT THUẬT TOÁN LỌC CỘNG TÁC CHO TRƯỜNG HỢP ÍT DỮ LIỆU

NGUYỄN DUY PHƯƠNG, TỪ MINH PHƯƠNG

Học viện công nghệ bưu chính viễn thông; phuong.ptit@yahoo.com

Abstract. Collaborative filtering is a technique to predict the utility of items for a particular user by exploiting the behavior patterns of a group of users with similar preferences. This technique has been widely used for recommender systems and has a number of useful applications in e-commerce. In this paper, we present a collaborative filtering method based on an multi-task learning algorithm that was designed for pattern recognition . The method formulates the collaborative filtering problem as classification problems and performs classification for all users simultaneously by using a modified boosting algorithm. This allows sharing common features among different classification tasks and thus reduces the negative effect of data sparseness. Experimental results show the effectiveness of the proposed method in comparison with other methods, especially when data are sparse.

Tóm tắt. Lọc cộng tác là phương pháp dự đoán về mặt hàng mà người dùng quan tâm dựa trên thông tin từ những người dùng có cùng sở thích. Phương pháp này được sử dụng phổ biến cho các hệ hỗ trợ tư vấn và có nhiều ứng dụng trong thương mại điện tử. Trong bài báo này, chúng tôi đề xuất sử dụng một phương pháp lọc cộng tác dựa trên kỹ thuật học đa nhiệm đã được dùng trong nhận dạng ảnh. Đây là phương pháp sử dụng kỹ thuật tăng cường để thực hiện đồng thời việc phân loại cho nhiều người dùng khác nhau, qua đó cho phép chia sẻ những đặc trưng chung giữa các bài toán phân loại để giảm bớt ảnh hưởng của việc có ít dữ liệu. Kết quả thử nghiệm và so sánh với kỹ thuật lọc khác cho thấy phương pháp cho kết quả tốt, đặc biệt trong trường hợp ít dữ liệu.

1. MỞ ĐẦU

Người sử dụng Internet thường gặp khó khăn vì phải xử lý quá nhiều thông tin trước khi tìm được thông tin quan tâm. Một trong những giải pháp hỗ trợ người dùng trong trường hợp này là sử dụng các hệ hỗ trợ tư vấn (recommender systems). Hệ hỗ trợ tư vấn là những hệ thống có khả năng tự động lựa chọn và cung cấp cho người dùng những thông tin mà người đó quan tâm. Ở đây thông tin có thể là thông tin dưới dạng văn bản như trang web, bản tin, hay thông tin về hàng hoá, sản phẩm .v.v. Để đơn giản, trong phần trình bày tiếp theo, chúng tôi sẽ gọi chung thông tin hay hàng hoá là các mặt hàng (item). Hệ hỗ trợ tư vấn đã có nhiều ứng dụng thương mại thành công như tư vấn mua hàng trong website mua bán trực tuyến lớn nhất thế giới http://www. amazon.com, các website tư vấn lựa chọn đĩa nhạc và phim ảnh ...

Hệ hỗ trợ tư vấn được chia thành hai loại: hệ sử dụng phương pháp lọc cộng tác (collaborative filtering) và hệ lọc theo nội dung (content-based filtering) [2]. Lọc theo nội dung là phương pháp dựa trên việc so sánh nội dung của thông tin hay mô tả hàng hoá để tìm ra

những mặt hàng tương tự với những gì người dùng từng quan tâm và giới thiệu cho người dùng những mặt hàng này.

Khác với lọc theo nội dung, lọc cộng tác không dựa trên việc sử dụng nội dung thông tin. Thay vào đó, phương pháp này xác định những người có cùng sở thích với người dùng cần tư vấn, từ đó giới thiệu với người cần tư vấn những mặt hàng mà người cùng sở thích từng mua hoặc đánh giá cao [13]. Hai người được coi là có cùng sở thích nếu họ đã từng mua những loại hàng giống nhau hay cùng truy cập những thông tin giống nhau trong quá khứ. So với lọc theo nội dung, lọc cộng tác có một số ưu điểm như có thể sử dụng với mọi loại thông tin hay hàng hoá mà không cần có mô tả dưới dạng văn bản. Kết quả thử nghiệm cũng cho thấy, lọc cộng tác cho kết quả tư vấn tốt hơn trong nhiều trường hợp [1, 10]. Trong bài báo này, chúng tôi chỉ tập trung vào phương pháp lọc cộng tác.

Ngoài các ưu điểm so với lọc theo nội dung, một khó khăn khi sử dụng lọc cộng tác là mỗi người dùng thường chỉ đánh giá hoặc mua tương đối ít mặt hàng, hoặc khi xuất hiện mặt hàng mới chưa có đánh giá của ai cả. Trường hợp này được gọi là trường hợp có ít dữ liệu hoặc dữ liệu thưa thớt và sẽ được chú ý giải quyết trong nghiên cứu này.

Bài toán lọc cộng tác có thể phát biểu như bài toán phân loại tự động của học máy. Dựa trên đánh giá của người dùng về những mặt hàng khác nhau, với mỗi người dùng, một mô hình phân loại sẽ được xây dựng và huấn luyện, mô hình này sau đó được sử dụng để phân chia mặt hàng mới thành các loại khác nhau, ví dụ như loại "thích" và "không thích". Tương tự như vậy, có thể thay đổi vai trò giữa người dùng và mặt hàng và xây dựng bộ phân loại cho phép dự đoán một mặt hàng cụ thể sẽ được một người dùng "thích" hay "không thích". Phương pháp này đã được sử dụng trong một số nghiên cứu về lọc cộng tác [3, 5] và cho kết quả tốt so với phương pháp lọc cộng tác truyền thống.

Điểm đặc trưng nhất của phương pháp phân loại nói trên là mỗi người dùng hay mỗi mục sẽ được coi như một bài toán phân loại riêng, độc lập với những người khác. Tuy nhiên, dễ dàng nhận thấy, những người dùng khác nhau không hoàn toàn độc lập với nhau và có những mối tương quan nhất định trong sở thích. Việc chia sẻ thông tin tương quan giữa các bài toán phân loại cho những người dùng khác nhau có thể cho phép cải thiện hiệu quả phân loại, nhất là trong trường hợp từng bài toán phân loại có ít dữ liệu huấn luyện, tương ứng với việc mỗi người dùng mới có đánh giá cho rất ít mặt hàng. Phương pháp sử dụng thông tin từ bài toán phân loại này cho bài toán phân loại khác được gọi là học đa nhiệm (multitask learning) hay học chuyển giao (transfer learning) và đã được đề cập trong nhiều nghiên cứu, ví dụ [4, 7].

Trong bài báo này, chúng tôi đề xuất sử dụng một kỹ thuật học đa nhiệm cho lọc cộng tác. Khác với kỹ thuật phân loại trình bày trong [5], phương pháp học đa nhiệm tiến hành huấn luyện đồng thời bộ phân loại cho tất cả người dùng sử dụng kỹ thuật boosting kết hợp với gốc cây quyết định (decision stump). Việc huấn luyện đồng thời cho phép phát hiện đặc trưng chung cho những người dùng khác nhau. Đặc trưng chung đóng vai trò bổ sung thông tin giữa từng bài toán phân loại riêng lẻ và cho phép cải thiện hiệu quả phân loại. Phương pháp này đã được sử dụng khá thành công trong nhận dạng ảnh nhằm mục đích phát hiện những đặc trưng chung giữa các đối tượng cần phân loại [14].

Phương pháp đề xuất trong bài báo được thử nghiệm trên hai bộ dữ liệu thực về đánh giá của người dùng đối với phim. Kết quả thử nghiệm cho thấy việc học đa nhiệm và sử dụng đặc trưng chung cho kết quả lọc tốt hơn so với học riêng rẽ và phương pháp lọc cộng tác truyền thống dựa trên tương quan giữa người dùng trong trường hợp có ít dữ liệu.

2. LỌC CỘNG TÁC BẰNG PHÂN LOẠI

Để tiện cho việc trình bày phương pháp đề xuất, trong phần này chúng tôi sẽ nhắc lại tóm tắt phương pháp phân loại dùng cho lọc cộng tác của Billsus và Pazani [5].

Gọi $U = \{u_1, ..., u_K\}$ là tập hợp bao gồm K người dùng, $G = \{g_1, ..., g_N\}$ là tập gồm N mặt hàng. Đánh giá của người dùng u_i về mặt hàng g_j được ký hiệu bằng r_{ij}. Như vậy, đánh giá r_{ij} tạo thành ma trận dữ liệu của bài toán lọc cộng tác, trong đó các hàng tương ứng với người dùng và cột tương ứng với mặt hàng. Giá trị r_{ij} có thể được thu thập trực tiếp bằng cách hỏi ý kiến người dùng hoặc thu thập gián tiếp: ví dụ khi người dùng mua mặt hàng gì hay xem bộ phim gì thì đánh giá của người dùng sẽ là "thích". Trong trường hợp tổng quát, r_{ij} có thể nhận giá trị từ một tập các mức đánh giá có thứ tự. Để cho đơn giản, ở đây chúng tôi giả sử r_{ij} có thể nhận giá trị "thích" hoặc "không thích" hay $+1$ và -1. Thông thường, mỗi người dùng chỉ đánh giá một tập rất nhỏ các mặt hàng và do vậy đa số các giá trị r_{ij} được để trống $r_{ij} = \phi$. Bài toán đặt ra khi đó là dự đoán những giá trị còn trống này của ma trận r_{ij}.

Để xác định các giá trị còn trống, [5] coi việc dự đoán những đánh giá cho mỗi người dùng là một bài toán phân loại riêng lẻ. Như vậy phương pháp này đòi hỏi huấn luyện m bộ phân loại riêng biệt, mỗi bộ cho phép dự đoán giá trị trống trong một hàng của ma trận x_{ij}.

Xét ví dụ cho trong Bảng 1. Ví dụ này bao gồm 4 người dùng và 5 mặt hàng. Giả sử cần dự đoán đánh giá của người dùng 4 đối với mặt hàng 5. Do người dùng 4 đã có đánh giá với 3 mặt hàng 1, 2 và 3, những đánh giá này sẽ được dùng làm ví dụ huấn luyện bộ phân loại. Mỗi ví dụ huấn luyện có dạng một vectơ các đặc trưng, mỗi đặc trưng tương ứng với một người dùng khác người dùng 4, giá trị của đặc trưng là giá trị các ô của ma trận. Nhãn phân loại cho các ví dụ huấn luyện là các đánh giá tương ứng của người dùng 4 cho mặt hàng 1, 2 và 3.

Bảng 1

	G_1	g_2	g_3	g_4	g_5
u1	+1	−1		+1	+1
u2		+1	+1	−1	
u3	+1	+1	−1		−1
u4	+1	−1	+1		?

Với ví dụ huấn luyện như trên, bài toán phân loại có thể thực hiện bằng những phương pháp phân loại thông dụng, ví dụ mạng nơ ron nhân tạo, cây quyết định, support vector machines .v.v. Tuy nhiên, trước khi sử dụng trực tiếp dữ liệu để huấn luyện và phân loại, một vấn đề cần giải quyết là vấn đề trích chọn đặc trưng. Trong trường hợp trình bày ở đây, mỗi đặc trưng chính là đánh giá của một người dùng khác với người dùng đang xét (trong ví

dụ ở Bảng 1, bài toán phân loại cho người dùng 4 có 3 đặc trưng là đánh giá của người dùng 1, 2, và 3). Trên thực tế, số lượng đặc trưng rất lớn và không phải đặc trưng nào cũng liên quan tới đánh giá của người dùng đang xét. Việc sử dụng cả các đặc trưng không liên quan làm tăng độ phức tạp tính toán đồng thời làm giảm độ chính xác phân loại.

Để giải quyết vấn đề trích chọn đặc trưng, trong [5], các tác giả sử dụng phương pháp singular value decomposition (SVD). Phương pháp này phân tích ma trận x_{ij} thành tích của ma trận bao gồm các vectơ riêng và ma trận đường chéo bao gồm các giá trị riêng sau đó rút gọn kích thước ma trận bằng cách chỉ giữ lại những vectơ riêng tương ứng với những giá trị riêng lớn nhất. Nhờ vậy, những đặc trưng ban đầu được biến đổi thành đặc trưng mới. Đặc điểm của đặc trưng mới là số lượng đặc trưng ít hơn nhưng sau khi chiếu dữ liệu xuống đặc trưng mới sẽ cho phương sai lớn hơn so với khi chiếu xuống đặc trưng gốc và do vậy dễ phân loại dữ liệu hơn.

3. PHÂN LOẠI VỚI CÁC ĐẶC TRƯNG CHUNG

Với phương pháp trình bày ở trên, quá trình trích trọn đặc trưng và huấn luyện bộ phân loại cho người dùng u_i người dùng được thực hiện trên dữ liệu được tạo thành từ những mặt hàng mà người dùng này đã có đánh giá. Thông thường, mỗi người dùng chỉ đánh giá một tập rất nhỏ các mặt hàng, do vậy mỗi bộ phân loại chỉ được huấn luyện trên một lượng dữ liệu nhỏ. Đây là yếu tố dẫn tới hiệu quả phân loại thấp.

Để giải quyết nhược điểm nói trên, phần này sẽ trình bày phương pháp mới trong đó việc huấn luyện và trích trọn đặc trưng được thực hiện đồng thời cho tất cả người dùng thay vì cho từng người riêng rẽ như vừa mô tả. Việc huấn luyện đồng thời cho phép kết hợp thông tin và dữ liệu huấn luyện từ những người dùng khác, nhờ vậy giảm bớt yêu cầu có nhiều mặt hàng được đánh giá trước cho mỗi người dùng. Đây là một kỹ thuật thường được gọi là học đa nhiệm.

Việc trích trọn đặc trưng và huấn luyện đồng thời cho tất cả người dùng được thực hiện bằng thuật toán boosting kết hợp với gốc cây quyết định (decision stump) [9, 12, 14].

3.1. Phương pháp boosting

Boosting là phương pháp học máy cho phép tạo ra bộ phân loại có độ chính xác cao bằng cách kết hợp nhiều bộ phân loại có độ chính xác kém hơn (còn gọi là bộ phân loại yếu) [12]. Dựa trên nguyên tắc chung này, nhiều phiên bản khác nhau của kỹ thuật boosting đã được đề xuất và sử dụng [8, 9, 12]. Trong nghiên cứu này, chúng tôi sẽ sử dụng phiên bản Gentle AdaBoost (viết tắt là gentleboost) được đề xuất trong [9] do các ưu điểm của phương pháp này như đơn giản, ổn định, và cho kết quả phân loại tốt trong nhiều ứng dụng.

Phương pháp gentleboost cho trường hợp phân loại hai lớp có thể mô tả tóm tắt như sau. Cho dữ liệu huấn luyện bao gồm N ví dụ $(x_1, y_1), ..., (x_N, y_N)$ với x_i là vectơ các đặc trưng và y_i là nhãn phân loại: $y_i = +1$ hoặc -1 (tương ứng với "thích" và "không thích"). Bộ phân loại $F(x)$ được tạo thành bằng cách tổ hợp $F(x) = \sum_{m=1}^{M} f_m(x)$, trong đó $f_m(x)$ là bộ phân loại yếu có khả năng dự đoán nhãn phân loại cho vectơ đầu vào x. Kết quả phân loại cuối cùng được tạo ra bằng cách tính $sign(F(x))$. Thuật toán bao gồm M vòng. Tại vòng

thứ m, các ví dụ huấn luyện sẽ được đánh trọng số lại sao cho những ví dụ bị phân loại sai trong vòng trước nhận được trọng số cao hơn và do vậy cần được bộ phân loại chú ý hơn. Bộ phân loại $f_m(x)$ được huấn luyện trên dữ liệu có trọng số trong vòng thứ m. Thuật toán gentleboost được thể hiện trên hình dưới đây.

Thuật toán gentleboost

1. Khởi tạo các trọng số $w_i = 1/N$, $i = 1..N$, w_i là trọng số của ví dụ huấn luyện thứ i. Khởi tạo $F(x) = 0$
2. Lặp với $m = 1, 2, ..., M$
   a. Huấn luyện $f_m(x)$ sử dụng dữ liệu huấn luyện có trọng số
   b. Cập nhật $F(x) \leftarrow F(x) + f_m(x)$
   c. Cập nhật trọng số $w_i \leftarrow w_i e^{-y_i f_m(x_i)}$ và chuẩn tắc hoá trọng số
3. Trả về bộ phân loại $sign[F(x)] = sign[\sum_{m=1}^{M} f_m(x)]$

Tại bước (a) của mỗi vòng lặp, thuật toán lựa chọn $f_m(x)$ sao cho sai số phân loại dưới đây là nhỏ nhất:

$$J = \sum_{i=1}^{N} w_i (y_i - f_m(x_i))^2. \tag{1}$$

Để tìm được bộ phân loại cho phép cực tiểu hoá (1), cần xác định bộ phân loại yếu $f_m(x)$ cho phép cực tiểu hoá bình phương lỗi phân loại có tính tới trọng số. Ở đây, chúng tôi sẽ sử dụng gốc quyết định (stump) làm bộ phân loại yếu. Gốc quyết định là phiên bản đơn giản của cây quyết định (decision tree) với một nút duy nhất. Gốc quyết định lựa chọn một đặc trưng của ví dụ huấn luyện, sau đó tuỳ thuộc vào giá trị của đặc trưng để gán cho nhãn giá trị 1 hay -1. Quá trình xác định nhãn phân loại được biểu diễn bởi công thức

$$f_m(x) = a\delta(x^f > t) + b\delta(x^f \leq t) \tag{2}$$

trong đó $\delta(e) = 1$ nếu e đúng và $\delta(e) = 0$ nếu ngược lại, t là một giá trị ngưỡng, a và b là tham số, x^f là giá trị đặc trưng thứ f của vectơ x. Trong trường hợp dữ liệu đánh giá chỉ bao gồm giá trị 1 và 0 hoặc 1 và -1, có thể chọn ngưỡng $t = 0$. Như vậy, ngoài việc phân loại, gốc quyết định còn thực hiện trích trọn đặc trưng do mỗi gốc chỉ chọn một đặc trưng duy nhất. Quá trình huấn luyện để chọn ra gốc tốt nhất (cho phép cực tiểu hoá (1)) được thực hiện bằng cách thử tất cả đặc trưng f. Với mỗi giá trị của f, giá trị tối ưu của a và b được tính như sau (sử dụng kỹ thuật least square estimation mà bản chất là tính giá trị tham số tại điểm có đạo hàm bằng 0):

$$a = \frac{\sum_i w_i y_i \delta(x^f > 0)}{\sum_i w_i \delta(x^f > 0)}, \tag{3}$$

$$b = \frac{\sum_i w_i y_i \delta(x^f \leq 0)}{\sum_i w_i \delta(x^f \leq 0)}, \tag{4}$$

Giá trị f và giá trị a và b tương ứng, cho sai số dự đoán (1) nhỏ nhất sẽ được chọn để tạo ra bộ phân loại $f_m(x)$ cho vòng lặp thứ m. $f_m(x)$ sau đó được thêm vào bộ phân loại chính $F(x)$ (bước b).

Tại bước (c), gentleboost tăng trọng số cho những ví dụ bị phân loại sai [ví dụ có y_i khác dấu với $f_m(x_i)$ và do vậy $y_i f_m(x_i) < 0$] và giảm trọng số cho những ví dụ được phân loại đúng. Bằng cách này, thuật toán sẽ khiến bộ phân loại ở vòng sau chú ý hơn tới những ví dụ hiện đang bị phân loại sai.

3.2. Cơ sở lý thuyết của gentleboost

Thực chất, gentleboost là cách xây dựng bộ phân loại bằng cách sử dụng phương pháp Niutơn để cực tiểu hoá hàm lỗi khi phân loại (xem [9]). Dưới đây là các phân tích chứng minh cho nhận xét này.

Trong bài toán phân loại nói chung, giá trị hàm lỗi được tính bằng kỳ vọng của số lượng ví dụ bị phân loại sai, tức là bằng:

$$E[\delta(F(x) \neq y)] = \frac{1}{N} \sum_{i=1}^{N} \delta(F(x_i) \neq y_i), \tag{5}$$

trong đó $E[.]$ là kỳ vọng và được tính bằng tỷ lệ lỗi trên dữ liệu huấn luyện. Do hàm (5) không khả vi, nên thay vào đó các thuật toán boosting sử dụng hàm lỗi dưới dạng hàm mũ như sau

$$J = E[e^{-yF(x)}]. \tag{6}$$

Dễ dàng nhận thấy hàm (6) là giới hạn trên của (5) (do $e^{-yF(x)} > 1$ khi $F(x_i) \neq y_i$), vì vậy cực tiểu hoá (6) cho phép giảm giá trị hàm lỗi (5).

Thuật toán gentleboost xây dựng hàm phân loại cho giá trị lỗi J nhỏ nhất bằng cách cải thiện dần hàm phân loại. Tại bước thứ m, ta cần cải thiện hàm $F(x)$ bằng cách cập nhật $F(x) + f_m(x)$. Thuật toán sẽ lựa chọn $f_m(x)$ sao cho giá trị hàm lỗi J giảm nhiều nhất. Khai triển hàm J dưới dạng chuỗi Taylor bậc 2, ta có:

$$\begin{aligned} J(F(x) + f_m(x)) &= E[e^{-y(F(x)+f_m(x))}] \\ &\approx E[e^{-yF(x)}(1 - yf_m(x) + y^2 f_m(x)^2/2)] \\ &= E[e^{-yF(x)}(1 + (y - f_m(x))^2)/2)] \text{ do } y^2 = 1. \end{aligned}$$

Do hằng số và hệ số 1/2 không ảnh hưởng tới $f_m(x)$, ta có thể viết:

$$f_m(x) = \arg\min_{f_m} E[e^{-yF(x)}(y - f_m(x))^2]. \tag{7}$$

Thay kỳ vọng bằng giá trị trung bình trên dữ liệu huấn luyện và sử dụng ký hiệu trọng số $w_i = e^{-y_i F(x_i)}$, (7) được viết lại như sau:

$$f_m(x) = \arg\min_{f_m} \sum_{i=1}^{N} w_i(y_i - f_m(x_i))^2]. \tag{8}$$

Giá trị cần tối ưu chính là hàm lỗi (1) đã nói ở trên. Công thức này giải thích việc lựa chọn $f_m(x)$ bằng kỹ thuật least square estimation và cách thức cập nhật trọng số w_i tại bước m.

Tiếp theo, ta sẽ thấy việc cập nhật $F(x)$ sử dụng $f_m(x)$ theo (7) tương ứng với các bước trong phương pháp Niutơn để cực tiểu hoá hàm lỗi. Thật vậy, sử dụng một số biến đổi đơn giản, ta có:

$$\left.\frac{\partial J(F(x)+f_m(x))}{\partial(f_m(x))}\right|_{f_m(x)=0} = -E[e^{-yF(x)}y|, \tag{9}$$

$$\left.\frac{\partial^2 J(F(x)+f_m(x))}{\partial^2(f_m(x))}\right|_{f_m(x)=0} = E[e^{-yF(x)}y^2] = E[e^{-yF(x)}]. \tag{10}$$

Các bước cập nhật của phương pháp Niutơn khi đó được tính bởi:

$$F(x) \leftarrow F(x) + \frac{E[e^{-yF(x)}y|}{E[e^{-yF(x)}]}. \tag{11}$$

Thay trọng số $w_i = e^{-y_i F(x_i)}$ và xác định $f_m(x)$ từ (7) bằng cách cho đạo hàm bằng 0, ta có:

$$f_m(x) = \frac{E[e^{-yF(x)}y|}{E[e^{-yF(x)}]}. \tag{12}$$

Như vậy việc thêm $f_m(x)$ vào $F(x)$ tại bước m tương ứng với bước cập nhật của phương pháp Niutơn. Từ đây có thể suy ra tính hội tụ của thuật toán. Các chứng minh chi tiết hơn có thể xem trong tài liệu [9]. Trên thực tế, thuật toán gentleboost không cần chạy tới khi hội tụ. Theo các nghiên cứu trước đây cho thấy, $M = 200$ vòng lặp cho kết quả tốt. Chúng tôi cũng sử dụng số bước lặp này trong thử nghiệm của mình.

3.3. Boosting đồng thời cho nhiều bài toán phân loại

Trong phần này, chúng tôi đề xuất sử dụng phương pháp boosting cải tiến đã được trình bày trong tài liệu tham khảo [14] để thực hiện đồng thời cho nhiều bài toán phân loại, tương ứng với nhiều người dùng trong trường hợp lọc cộng tác.

Với tập K người dùng U; N mặt hàng G và giá trị đánh giá r_{ij} như đã cho trong phần 2, ta có tất cả K bài toán phân loại, bài toán thứ k, $k = 1, ..., K$ được cho bởi N ví dụ huấn luyện $(x_1^k, y_1^k), ..., (x_N^k, y_N^k)$, trong đó $y_j^k = r_j^k$ tức là đánh giá của người dùng k cho mặt hàng j, và $x_j^k = (r_{1j}, ..., r_{(k-1)j}, r_{(k+1)j}, ..., r_{Kj})$ tức là đánh giá của tất cả người dùng cho mặt hàng j trừ người dùng k. Cần lưu ý rằng, chỉ những cột có $r_{kj} \neq \phi$ mới được sử dụng làm ví dụ huấn luyện trong bài toán thứ k (cột của mặt hàng 1,2,3 trong ví dụ ở Bảng 1). Tuy nhiên, để cho đơn giản, ta vẫn liệt kê cả những ví dụ có $r_{kj} = \phi$. Những ví dụ này sau đó sẽ được gán trọng số bằng 0 và do vậy không ảnh hưởng tới kết quả huấn luyện.

Điểm khác biệt chủ yếu của thuật toán boosting cho nhiều bài toán đồng thời là tại mỗi vòng lặp, thuật toán sẽ tìm một đặc trưng cho phép giảm sai số dự đoán đồng thời cho một tập con các bài toán phân loại thay vì giảm sai số của một bài toán phân loại như mô tả ở trên. Đặc trưng này đóng vai trò đặc trưng chung cho tất cả các bài toán phân loại trong tập con được chọn. Các tập con được chọn trong những vòng khác nhau của thuật toán có thể giao nhau.

Cụ thể, thuật toán boosting sẽ được thay đổi như sau. Mỗi ví dụ huấn luyện thứ j sẽ có k trọng số w_j^k, $k = 1, ..., K$. Mỗi trọng số được sử dụng khi ví dụ đó được dùng với bộ phân

loại thứ k. $w_j^k = 0$ nếu $r_j^k = 0$ tức là ví dụ j không tham gia vào huấn luyện bộ phân loại k. Sai số phân loại được tính bằng tổng sai số cho tất cả K bộ phân loại:

$$J = \sum_{k=1}^{K} \sum_{i=1}^{N} w_i^k (y_i^k - f_m^k(x_i))^2. \tag{13}$$

Tại mỗi vòng lặp m, gọi $S(t)$ là tập con các bài toán. Thay vì xác định đặc trưng f tốt nhất cho từng bài toán riêng lẻ như ở phần trước, thuật toán cần xác định đặc trưng chung cho tất cả bài toán thuộc $S(t)$ và chọn gốc quyết định tương ứng sao cho sai số (13) là nhỏ nhất. Gốc cây quyết định sẽ có dạng như sau:

$$f_m^k(x,t) = \begin{cases} a_S \delta(x^f > 0) + b_S \delta(x^f \leq 0) & \text{khi } k \in S(t) \\ c^k & \text{khi } k \notin S(t) \end{cases} \tag{14}$$

Ở đây, giá trị gốc cây quyết định phụ thuộc vào việc tập con $S(t)$ được chọn là tập con nào và vì vậy ta ký hiệu hàm f_m là hàm của t. Ký hiệu $f_m^k(x,t)$ được hiểu là hàm phân loại yếu tại bước thứ m cho bài toán thứ k và hàm này chung cho tập con $S(t)$ các bài toán phân loại. Do giá trị hàm lỗi (13) cũng phụ thuộc vào tập con $S(t)$ nên hàm lỗi (13) cũng cần viết lại thành hàm của tham số t như sau:

$$J(t) = \sum_{k=1}^{K} \sum_{i=1}^{N} w_i^k (y_i^k - f_m^k(x_i,t))^2. \tag{15}$$

Điểm khác nhau cơ bản so với gốc quyết định ở phần trước là gốc quyết định (14) phân biệt trường hợp bài toán k thuộc tập con $S(t)$ và trường hợp không thuộc. Trong trường hợp k không thuộc $S(t)$, hàm $f_m(x)$ sẽ được đặt bằng hằng số c^k để tránh trường hợp lựa chọn bộ phân loại một cách tình cờ do chênh lệch số lượng giữa ví dụ huấn luyện 1 và -1 (chẳng hạn trong trường hợp quá nhiều ví dụ 1 thì có thể luôn dự đoán nhãn là 1 không cần quan tâm tới đặc trưng).

Với mỗi tập con $S(t)$, giải bài toán cực tiểu hoá sai số (13) sẽ cho kết quả sau:

$$a_S(f) = \frac{\sum_{k \in S(t)} \sum_i w_i^k y_i^k \delta(x_i^f > 0)}{\sum_{k \in S(t)} \sum_i w_i^k \delta(x_i^f > 0)}, \tag{16}$$

$$b_S(f) = \frac{\sum_{k \in S(t)} \sum_i w_i^k y_i^k \delta(x_i^f \leq 0)}{\sum_{k \in S(t)} \sum_i w_i^k \delta(x_i^f \leq 0)}, \tag{17}$$

$$c^k = \frac{\sum_i w_i^k y_i^k}{\sum_i w_i^k, \ k \notin S(t)}. \tag{18}$$

Tại mỗi bước lặp, thuật toán sẽ lựa chọn tập con $S(t)$ tốt nhất, tức là tập con cho giá trị hàm lỗi (13) nhỏ nhất và gốc quyết định tốt nhất cho tập con đó. Ký hiệu $F^k(x)$ là bộ phân loại chính xác cho bài toán phân loại thứ k, ta có thuật toán boosting mới được thể hiện như sau.

Thuật toán 2. Thuật toán boosting cải tiến sử dụng đặc trưng chung cho nhiều bài toán

1. Khởi tạo $w_j^k = 1$ nếu $r_j^k \neq \phi$ và $w_j^k = 0$ nếu $r_j^k = \phi$, $i = 1,..,N; k = 1,..,K$
   Khởi tạo $F^k(x) = 0$

2. Lặp với $m = 1, ..., M$

   a. Lặp với tập con các bài toán $S(t)$

      i. Tính tham số a_S, b_S, và c^k theo (16), (17), (18)

      ii. Tính sai số $J(t) = \sum_{k=1}^{K} \sum_{i=1}^{N} w_i^k (y_i^k - f_m^k(x_i, t))^2$

   b. Chọn tập $S(t)$ tốt nhất $t^* = \arg\min J(t)$

   c. Cập nhật $F^k(x) \leftarrow F^k(x) + f_m^k(x, t^*)$

   d. Cập nhật trọng số $w_i^k \leftarrow w_i^k e^{-y_i^k f_m(x_i, t^*)}$

3. Trả về bộ phân loại $sign[F^k(x)]$

Một chi tiết cần làm rõ đối với thuật toán trên là cách xác định tập con $S(t)$. Nếu liệt kê tất cả tập con $S(t)$ từ K bài toán thì số lượng tập con sẽ là $O(2^K)$. Do vậy, thay vì liệt kê, chúng tôi sử dụng phương pháp tìm kiếm tham lam. Trước tiên, xác định tập con t chỉ bao gồm 1 bài toán và có sai số (13) nhỏ nhất. Sau đó thêm một bài toán khác vào tập con t trước đó sao cho t mới có sai số nhỏ nhất. Tiếp tục như vậy cho đến khi thêm tất cả K bài toán vào t. Trong số các tập t như vậy lựa chọn t có sai số nhỏ nhất. Độ phức tạp tính toán khi đó sẽ là $O(K^2)$ cho mỗi bước lặp và $O(MK^2)$ cho toàn bộ thuật toán.

Phân tích lý thuyết

Các phân tích lý thuyết đối với thuật toán gentleboost ở Mục 3.1 vẫn đúng đối với Thuật toán 2 nếu thay hàm lỗi (6) bằng $\exp(-\sum_k y^k F^k(x))$ và thay kỳ vọng trong (7) và (11) bằng giá trị trung bình cho cả K bộ phân loại. Điểm khác nhau duy nhất ở đây là tại mỗi bước lặp, thuật toán có thể không tìm được hàm $f_m^k()$ cho sai số nhỏ nhất do phải xác định tập con $S(t)$ một cách tham lam và do vậy tốc độ hội tụ sẽ chậm hơn so trường hợp tìm được $f_m^k()$ tối ưu. Tuy nhiên, thuật toán vẫn cho phép giảm dần lỗi phân loại tại mỗi bước lặp và cho kết quả tốt trong các thử nghiệm.

Các kết quả liên quan

Như đã nói ở trên, phương pháp lọc cộng tác vừa trình bày dựa trên việc sử dụng thuật toán boosting với các đặc trưng chung do Torralba et al. đề xuất cho bài toán nhận dạng ảnh và được trình bày trong tài liệu [14]. Tuy nhiên, tính chất bài toán cũng như mục đích sử dụng đặc trưng chung trong hai trường hợp là khác nhau:

- Trong [14], thuật toán được đề xuất cho phân loại đối tượng ảnh. Bài toán phân loại các đối tượng ảnh trong [14] vốn là một bài toán phân loại nhiều lớp (multiclass classification), trong khi đó bài toán lọc cộng tác ở đây bao gồm nhiều bài toán phân loại hai lớp (binary classification). Sự khác nhau này đòi hỏi cách biểu diễn phù hợp để áp dụng thuật toán boosting vừa trình bày.

- Mục tiêu sử dụng boosting cải tiến trong [14] là phát hiện đặc trưng chung mà các đối tượng ảnh chia sẻ để tăng tốc độ (do không phải chia bài toán nhiều lớp thành nhiều bài toán hai lớp) và giảm dữ liệu huấn luyện, trong khi đó mục tiêu của lọc cộng tác là phát hiện những người dùng tương tự nhau và giải quyết vấn đề dữ liệu thưa thớt (nhiều dữ liệu thiếu). Việc tìm ra các tập con $S(t)$ (tương ứng với tập người dùng cùng sở thích) sẽ có ý nghĩa quan trọng đối với lọc cộng tác khi cần giải thích lý do đưa ra quyết định.

4. THỬ NGHIỆM VÀ KẾT QUẢ

Hiệu quả lọc cộng tác được xác định dựa trên khả năng thuật toán dự đoán chính xác đánh giá của khách hàng. Phương pháp trình bầy ở trên được đánh giá và so sánh với các phương pháp khác theo thủ tục mô tả dưới đây.

Trước tiên, toàn bộ khách hàng được chia thành hai phần, một phần U_{tr} được sử dụng làm dữ liệu huấn luyện, phần còn lại U_{te} được sử dụng để kiểm tra. Dữ liệu huấn luyện được sử dụng để xây dựng mô hình theo thuật toán mô tả ở trên. Với mỗi khách hàng thuộc tập dữ liệu kiểm tra u, các đánh giá (đã có) của khách hàng được chia làm hai phần O_u và P_u. O_u được coi là đã biết, trong khi đó P_u là đánh giá cần dự đoán từ dữ liệu huấn luyện và O_u.

Sai số dự đoán MAE_u với mỗi khách hàng u thuộc tập dữ liệu kiểm tra được tính bằng trung cộng sai số tuyệt đối giữa giá trị dự đoán và giá trị thực đối với tất cả mặt hàng thuộc tập P_u.

$$MAE_u = \frac{1}{|P_u|} \sum_{y \in P_u} |\hat{r}_y^u - r_y^u|.$$

Sai số dự đoán trên toàn tập dữ liệu kiểm tra được tính bằng trung bình cộng sai số dự đoán cho mỗi khách hàng thuộc U_{te}.

$$MAE = \frac{\sum_{u \in U_{te}} MAE_u}{|U_{te}|}.$$

Giá trị MAE càng nhỏ càng tốt, tức là phương pháp càng chính xác.

4.1. Dữ liệu thử nghiệm

Thuật toán lọc cộng tác được thử nghiệm trên hai bộ dữ liệu EachMovie [16] và MovieLens [17]. Đây là hai bộ dữ liệu thường được sử dụng để đánh giá các phương pháp lọc.

EachMovie được xây dựng bởi trung tâm nghiên cứu hệ thống thông tin của hãng Compaq. Bộ dữ liệu này gồm 72916 người dùng, 1628 bộ phim với 2811983 đánh giá, các mức đánh giá được cho từ 1 đến 6 (chính xác là 0.0, 0.2, 0.4, 0.6, 0.8, 1.0) , trung bình số lượng phim người dùng chưa đánh giá là 97.6%. Hai mức đánh giá cao nhất (0.8 và 1.0) được biến đổi thành "thích" (+1) và bốn mức đánh giá còn lại được biến đổi thành "không thích" (−1) (đây là phương pháp biến đổi được sử dụng trong [5]).

MovieLens là cơ sở dữ liệu được xây dựng bởi nhóm nghiên cứu GroupLens của trường đại học Minnesota. MovieLans có 6040 người dùng, 3900 bộ phim, 1000209 đánh giá, các mức đánh giá cho từ 1 đến 5, trung bình số lượng phim người dùng chưa đánh giá là 95.7%. Tương tự như trên, hai mức đánh giá cao nhất được biến đổi thành "thích", các mức còn lại thành "không thích".

Để hạn chế ảnh hưởng của tình trạng dữ liệu quá thưa thớt, chúng tôi chỉ lựa chọn những khách hàng có đánh giá ít nhất cho 20 bộ phim. Trong số những người dùng thoả mãn điều kiện này, chúng tôi chọn ra bộ dữ liệu thứ nhất từ EachMovie bao gồm 10000 người dùng. Bộ dữ liệu thứ hai từ MovieLens bao gồm đánh giá của 500 người dùng.

4.2. So sánh và kết quả

Phương pháp boosting với đặc trưng chung (ký hiệu là MC Boost) trình bày trong phần 3.2 được so sánh với những phương pháp sau:

- Phương pháp k hàng xóm gần nhất sử dụng độ tương quan Pearson (ký hiệu là KPC) [6]. Đây là phương pháp lọc cộng tác thông dụng nhất và thường được sử dụng như phương pháp tiêu chuẩn khi so sánh.

- Phương pháp boosting không sử dụng đặc trưng chung như trình bày trong Mục 3.1.

Cách thức tiến hành thử nghiệm và so sánh như sau:

Lần lượt 100, 200, và 300 người dùng được lựa chọn ngẫu nhiên từ bộ dữ liệu MovieLens và được dùng làm dữ liệu huấn luyện, 200 người dùng được lựa chọn ngẫu nhiên trong số còn lại để làm tập kiểm tra.

Đối với bộ dữ liệu EachMovies, lần lượt 1000, 2000, và 6000 người dùng được chọn làm tập huấn luyện, 4000 người dùng còn lại được dùng để kiểm tra.

Để thử nghiệm khả năng của phương pháp mới đề xuất so với những phương pháp khác trong trường hợp có ít dữ liệu, chúng tôi thay đổi số lượng đánh giá của mỗi người dùng trong tập kiểm tra sao cho số lượng đánh giá đã biết lần lượt là 5, 10 và 20, phần còn lại là những đánh giá cần dự đoán. Giá trị MAE khi thử nghiệm với hai bộ dữ liệu MovieLens và EachMovies được thể hiện trong Bảng 2 và Bảng 3. Như đã nói ở trên, giá trị MAE nhỏ hơn chứng tỏ phương pháp chính xác hơn.

Bảng 2. Kết quả thử nghiệm với MovieLens

Kích thước tập huấn luyện	Phương pháp	Số đánh giá cho trước của tập kiểm tra		
		5	10	20
100 người dùng	KPC	0.378	0.337	0.328
	GentleBoost	0.350	0.322	**0.291**
	MC Boost	**0.329**	**0.305**	0.292
200 người dùng	KPC	0.361	0.330	0.318
	GentleBoost	0.333	0.314	**0.284**
	MC Boost	**0.314**	**0.299**	0.289
300 người dùng	KPC	0.348	0.336	0.317
	GentleBoost	0.325	0.304	**0.279**
	MC Boost	**0.308**	**0.298**	0.283

Kết quả thử nghiệm cho thấy cả hai phương pháp lọc cộng tác bằng phân loại boosting đều cho kết quả tốt hơn so với phương pháp truyền thống sử dụng độ tương quan Pearson. Sai số của boosting với mọi kích thước dữ liệu huấn luyện và số lượng đánh giá cho trước của người dùng kiểm tra đều nhỏ hơn phương pháp tương quan Pearson.

Trong trường hợp đủ dữ liệu, cụ thể là khi biết trước nhiều đánh giá của người dùng trong tập kiểm tra, phương pháp GentleBoost cho kết quả tốt hơn so với MC Boost. Có thể giải thích kết quả này là do GentBoost cho phép chọn đặc trưng tối ưu hơn đối với từng bài toán phân loại trong khi MC Boost chỉ chọn được đặc trưng tối ưu cho cả nhóm bài toán phân loại.

Tuy nhiên, khi dữ liệu ít đi, cụ thể là khi chỉ biết trước 5 hoặc 10 đánh giá của người dùng kiểm tra thì trong đa số trường hợp, MC Boost cho sai số MAE nhỏ hơn so với GentleBoost. Lý do chủ yếu là do MC Boost cho phép kết hợp thông tin từ những người dùng tương tự

với người dùng kiểm tra thông qua các đặc trưng chung và do vậy giảm được ảnh hưởng của việc thiếu nhãn phân loại.

Bảng 3. Kết quả thử nghiệm với EachMovies

Kích thước tập huấn luyện	Phương pháp	Số đánh giá cho trước của tập kiểm tra		
		5	10	20
1000 người dùng	KPC	0.559	0.474	0.449
	GentleBoost	0.515	**0.455**	**0.421**
	MC Boost	**0.492**	0.460	0.429
2000 người dùng	KPC	0.528	0.450	0.422
	GentleBoost	0.495	0.424	**0.393**
	MC Boost	**0.484**	**0.419**	**0.393**
6000 người dùng	KPC	0.521	0.437	0.378
	GentleBoost	0.477	0.408	**0.362**
	MC Boost	**0.452**	**0.397**	0.365

5. KẾT LUẬN

Bài báo đã trình bày một phương pháp lọc cộng tác bằng cách sử dụng kỹ thuật phân loại boosting và gốc cây quyết định đồng thời cho nhiều người dùng. Đây là một cải tiến của thuật toán boosting, trong đó việc lựa chọn đặc trưng cho mỗi bộ phân loại yếu được thực hiện đồng thời trên một nhóm người dùng tương tự nhau. Ưu điểm chủ yếu của phương pháp này là việc phân loại đồng thời từng nhóm người dùng cho phép sử dụng thông tin từ những người dùng tương tự nhau và nhờ vậy cải thiện độ chính xác phân loại khi có ít dữ liệu, ví dụ khi người dùng cần dự đoán chỉ đánh giá rất ít mặt hàng trước đó. Kết quả thử nghiệm trên hai bộ dữ liệu MovieLens và EachMovies đã cho thấy phương pháp đề xuất cho kết quả tốt hơn hai phương pháp khác trong trường hợp ít dữ liệu.

TÀI LIỆU THAM KHẢO

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, *IEEE Transactions On Knowledge And Data Engineering* **17** (6) (2005).

[2] M. Balabanovic and Y. Shoham, Fab: content-based, collaborative recommendation, *Comm. ACM* **40** (3) (1997) , 66–72.

[3] C. Basu, H. Hirsh, W. Cohen, Recommendation as classification: Using social and content-based information in recommendation, *Proc. of 15 Nat. Conf. on Artificial Intelligence (AAAI-98)*, Madison, WI, USA, 1998 (714–720).

[4] J. Baxter, A model for inductive bias learning, *J. of Artificial Intelligence Research* **12** (2000) 149–198.

[5] D. Billsus and M. Pazzani, Learning collaborative information filters, *Proc. Int'l Conf. Machine Learning*, Madison, WI, USA, 1998.

[6] J. S. Breese, D. Heckerman, and C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, *Proc. of 14th Conf. on Uncertainty in Artificial Intelligence*, Madison, WI, USA, 1998 (43–52).

[7] R. Caruana, Multi-task learning, *Machine Learning* **28** (1997) 41–75.

[8] Y. Freund and R. Schapire, Experiments with a new boosting algorithm, In Machine Learning, *Proceedings of the Thirteenth International Conference*, Bari, Italy, 1996 (148–156).

[9] J. Friedman, T. Hastie, and R. Tibshirani, Additive logistic regression: a statistical view of boosting, *The Annals of Statistics* **38** (2) (April, 2000) 337–374.

[10] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl, An algorithmic framework for performing collaborative filtering, *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '99*, Berkeley, CA, USA, 1999 (230–237).

[11] J. Herlocker, L. G. Konstan, Terveen, and J. Riedl, Evaluating collaborative filtering recommender systems, *ACM Transactions on Information Systems* **22** (1) (2004) 5–53.

[12] R. E. Schapire, The boosting approach to machine learning: an overview, *Proc. MSRI Workshop Nonlinear Estimation and Classification* , Berkeley, CA, Mar. 2001.

[13] U. Shardanand and P. Maes, Social information filtering: algorithms for automating 'Word of Mouth', *Proc. Conf. Human Factors in Computing Systems*, Denver, Colorado, USA, 1995.

[14] A. Torralba, K. P. Murphy, and W. T. Freeman, Sharing visual features for multiclass and multiview object detection, *IEEE Trans. On Pattern Analysis And Machine Intelligence* **29** (5) (May 2007).

[15] G. R. Xue, C. Lin, Q. Yang, W. Xi, H. J. Zeng, Y. Yu, and Z. Chen, Scalable collaborative filtering using cluster-based smoothing, *Proc. of SIGIR*, Salvador, Brazil, 2005 (114–121).

[16] http://www.reserch.compaq.com/SRC/eachmovie/

[17] http://www.movielens.org

Nhận bài ngày 1 - 10 - 2007
Nhận lại sau sửa ngày 22 - 2 -2008