Published in final edited form as:

J Biomed Inform. 2021 January ; 113: 103635. doi:10.1016/j.jbi.2020.103635.

Hybrid Collaborative Filtering Methods for Recommending Search Terms to Clinicians

Zhiyun Ren[a], **Bo Peng**[b], **Titus K. Schleyer**[c,d], **Xia Ning**[a,b,e]

[a]Department of Biomedical Informatics, The Ohio State University, 1800 Cannon Drive, Columbus, OH, 43210 USA

[b]Department of Computer Science and Engineering, The Ohio State University, 281 W Lane Ave, Columbus, OH, 43210 USA

[c]Regenstrief Institute, 1101 W 10th St, Indianapolis, IN, 46202 USA

[d]Indiana University School of Medicine, 340 W 10th St #6200, Indianapolis, IN 46202 USA

[e]Translational Data Analytics Institute, The Ohio State University, 1760 Neil Ave, Columbus, Ohio 43210

Abstract

With increasing and extensive use of electronic health records (EHR), clinicians are often challenged in retrieving relevant patient information efficiently and effectively to arrive at a diagnosis. While using the search function built into an EHR can be more useful than browsing in a voluminous patient record, it is cumbersome and repetitive to search for the same or similar information on similar patients. To address this challenge, there is a critical need to build effective recommender systems that can recommend search terms to clinicians accurately. In this study, we developed a hybrid collaborative filtering model to recommend search terms for a specific patient to a clinician. The model draws on information from patients' clinical encounters and the searches that were performed during them. To generate recommendations, the model uses search terms which are (1) frequently co-occurring with the ICD codes recorded for the patient and (2) highly relevant to the most recent search terms. In one variation of the model (Hybrid Collaborative Filtering Method for Healthcare, or HCFMH), we use only the most recent ICD codes assigned to the patient, and in the other (Co-occurrence Pattern based HCFMH, or cpHCFMH), all ICD codes. We have conducted comprehensive experiments to evaluate the proposed model. These

schleyer@regenstrief.org (T.K. Schleyer), ning.104@osu.edu (X. Ning).

Credit Author Statement

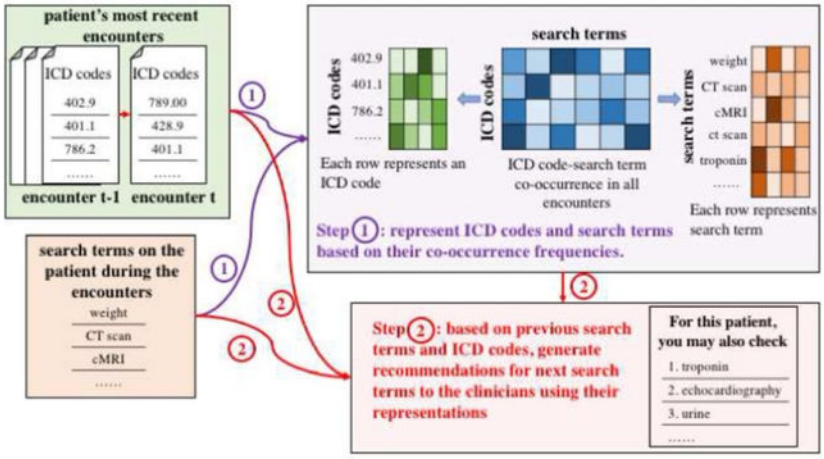
Zhiyun Ren: Data curation, Formal analysis, Methodology, Visualization, Writing - original draft. **Bo Peng**: Data curation, Formal analysis, Visualization, Writing – original draft. **Titus Schleyer**: Conceptualization, Funding acquisition, Investigation, Resources, Supervision, Validation, Project administration, Writing – review & editing. **Xia Ning**: Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Supervision, Project administration, Writing – original draft, review & editing.

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

experiments demonstrate that our model outperforms state-of-the-art baseline methods for top-N search term recommendation on different data sets.

Graphical Abstract



Keywords

collaborative filtering; search term recommendation; clinical decision support

1. Introduction

Electronic Health Records (EHRs) contain increasingly large and varied collections of health information about patients [24, 15, 17, 29]. However, given the limitations of today's user interfaces, it is difficult for clinicians to retrieve information from EHRs efficiently and effectively when they are providing care for patients in the clinic [4, 11, 26, 28]. Clinicians often operate under time pressure and must invest significant effort in retrieving information, such as demographics, prior findings and lab results, from EHRs in order to develop diagnoses and treatment plans [26, 10, 18, 25]. While conducting a search using an EHR's built-in search function can be a useful alternative to browsing through a patient record, searching for the same or similar information on similar patients is repetitive, time-consuming and cumbersome. Therefore, there is a critical need to build systems that can accurately recommend search terms to clinicians.

In this manuscript, we tackle the problem of recommending search terms for clinical decision-making. Given a patient's history of encounters and the search terms clinicians have used, the goal of these recommendations is to suggest information items to clinicians that are most relevant to the management of the patient at the time. Our method is intended to identify such information items proactively, and thus save time and effort that would be needed for manual searching/browsing. In addition, the suggestions may provide helpful reminders or hints to clinicians about potentially relevant information that they may have overlooked.

Outside of healthcare, online shoppers and other Internet users are familiar with recommender systems (RS) that suggest potentially relevant items, such as products, information or services, by learning user preferences from their previous interactions with items. Traditional RS techniques, such as collaborative filtering (CF) [23, 19], have been widely used to recommend the top-N most relevant items to users – the so-called top-N recommendation. In this manuscript, we adapt traditional CF techniques to information retrieval in healthcare. In an analogy to RSs in general, we consider clinicians as "users" and search terms as "items." The patient is the "context" of the search, much like "Thanksgiving season" might be a context for movie recommendations. Of note, recommendations are designed to be specific to a particular patient, their condition(s), time and other factors.

To develop our algorithm, we assumed that useful search term recommendations are strongly related to two characteristics of a patient: (1) the search terms clinicians had used previously for the patient, and (2) the patient's diagnoses (as represented by ICD codes). Based on this assumption, we developed a model that we named Hybrid Collaborative Filtering Method for Healthcare, denoted as HCFMH, to recommend search terms for a patient based on previous searches and diagnoses. This model first calculates the co-occurrence frequency between each ICD code and search term, given the recorded ICD codes and search terms for a patient. A search term "co-occurs" with an ICD code if it has been searched within three months from the time an ICD code was recorded for a patient. The HCFMH model then recommends terms that have high co-occurrence frequencies with the most *recent* ICD codes and are highly relevant to the most recent search terms. The model determines relevance between a recommendation candidate and the most recent search terms using latent factor models [14, 2]. A variation of the HCFMH model is based on Co-occurrence Pattern, denoted as cpHCFMH. Similar to the HCFMH, the cpHCFMH first calculates the co-occurrence frequency between each ICD code and search term. Unlike the HCFMH, the cpHCFMH recommends the search terms that have high co-occurrence frequencies with *all* (instead of the most *recent*) ICD codes of a patient. The experimental results show that the proposed models outperform all current state-of-the-art methods for top-N search term recommendation on a large real-world data set using four different cutoff dates for training and test data sets. We have also conducted comprehensive parameter studies that provide important insights on the use of previously recorded search terms and ICD codes.

2. Related Work

The work most relevant to the development of our method derives from research on recommender systems (RSs). Recent research in RS has focused significant attention on top-N recommendation problems, that is, how to recommend the N most relevant items [5, 6] given a specific user and context. One method, factorized personalized Markov chains (FPMC) [22], recommends items using Markov chains, which capture item-item transition relations. Hidasi *et al.* [9, 8] adapted deep learning techniques and used gated recurrent units (GRUs) to model the dynamics of users' preferences. Kang *et al.* [12] developed a self-attention-based sequential model (SASRec) to capture the most informative items in a user's action history for top-N recommendations. Tang *et al.* [27] created a convolutional sequence embedding recommendation model (Caser) by adapting multiple convolutional filters for the most recent portion of the action history of a user to model sequential features. Ma *et al.* [16]

developed a hierarchical gating network (HGN) that adapts gating mechanisms on users' action histories to identify important items and their features for top-N recommendations.

Very limited work has been conducted on recommending search terms for clinical applications. Earlier, our team developed several CF models [7] to tackle this problem. Among these models, the Transition-Involved Patient-Term-Similarity-based CF Scoring (TptCF) model recommends search terms for a patient to a clinician based on patient-patient similarities, term-term similarities and term-term transition relations. Major differences between TptCF and our new model are that our model generates top-N recommendations using both previous search terms and previous encounters, while the TptCF uses only previous search terms, and our model does not formally calculate patient-patient similarities, term-term similarities, etc., as TptCF does.

3. Definitions and Preprocessing

3.1. Definitions and Notations

Figure 1 and Table 1 show the data preprocessing protocols and key notations used in this manuscript, respectively. The key entities used in generating recommendations were patients; search terms and their sequences; patient encounters and their sequences; and the ICD codes associated with encounters. The terms searched on each patient were sorted chronologically. The sequence of patient p's sorted search terms (blue line in Figure 1) is denoted as S_p, and the subsequence of S_p from the i-th search to the j-th search is denoted as $S_p(i, j)$. For simplicity, we first generated an indexed collection of unique search terms for all patients and clinicians. S_p then stores only the indices of the search terms in the collection instead of the terms themselves.

Similarly, the encounters of each patient were sorted chronologically (orange line in Figure 1). The sequence of patient p's sorted encounters is denoted as C_p, and the subsequence of C_p from the i-th encounter to the j-th encounter is denoted as $C_p(i, j)$. For each patient, each search term was matched to the most recent prior encounter using timestamps (green arrows in Figure 1). Please note that such matching only indicates temporal proximity. It does not necessarily imply that the searches occurred during the matched encounters or that they were triggered by the encounters.

For each patient, one or more ICD codes may be associated with one or more encounters (light blue arrows in Figure 1). We denote the encounters of patient p that contain ICD code c as $C_p(c)$. Similarly, a term can be searched multiple times for a patient. We denote the encounters of patient p that each search term s is matched to as $C_p(s)$ (red arrows in Figure 1). In the sequences, we refer to ICD codes and search terms using indices.

3.2. Identifying Sessions

In our data set (discussed in Section 5.2), the time difference between consecutive searches may vary from minutes to years, but unfortunately, session information - the explicit start and end time of a set of cohesive physician interactions with the EHR systems for a specific patient - is not always logged. Therefore, we grouped searches into sessions based on their timestamps using a sliding window of three months (sessions are denoted by blue curly

brackets in Figure 1). If the time interval between two consecutive searches was less than three months, the search terms were grouped into one session. To generate recommendations, we only used search terms that are contained in the most recent session at the time. This approach reflects the scenario in which clinicians search for information about a patient within the context of the current or most recent visit.

4. Methods

Our methods have the following steps. First, we learn to numerically represent each ICD code and each search terms so that the representations can be used to calculate recommendation scores. Such representations encode how ICD codes and search terms co-occur in our data (Section 4.1). Then, we calculate a recommendation score for each term for each patient. The recommendation score is based on the previous search terms and encounters of all patients. We then calculate recommendation scores using the two methods we developed: HCFMH (Section 4.2) and cpHCFMH (Section 4.3). Finally, we rank the recommendation scores of all terms for each patient and recommended top-N terms to the clinician who is seeing the patient.

4.1. Learning from Previous Search Terms and Encounters

4.1.1. Constructing the ICD Code-Search Term Co-Occurrence Matrix—We assume that search terms are highly related to the patient's most recent encounters, that is, given the ICD codes that are assigned to a patient, terms that are related to the ICD codes are more likely to be searched next. For example, if a patient was assigned the ICD code "588.81: secondary hyperparathyroidism (of renal origin)" in a recent encounter, terms such as "potassium level," which is highly related to hyperparathyroidism, have high probability to follow. This is in contrast with, for instance, ICD code "786.2: Cough," for which "potassium level" would provide little information. Thus, co-occurrence frequencies between ICD codes and search terms are likely to provide useful information for predicting search terms. Given recent ICD codes assigned to a patient, terms with high co-occurrence frequencies with these ICD codes across all patients are more likely to be searched next and thus should be recommended.

Based on this assumption, we first calculate the frequency of co-occurrence between each ICD code and search term by counting how many times the term has been searched after the ICD code was assigned in all encounters of all patients. We construct a matrix $A \in \mathbb{R}^{n \times m}$ to store such co-occurrence frequencies, where n is the number of all unique ICD codes and m is the number of all unique search terms. We assume that clinicians tend to search information based on recent encounters of a patient. Thus, useful term recommendations are more likely to be generated from recent than past ICD codes. Based on this assumption, we emphasize information from recent encounters using a time-decay parameter and calculate the ICD code-search term co-occurrence frequencies a_{cs} as follows:

$$a_{CS} = \sum_{p=1}^{l} \sum_{e_c \in C_{p(c)}} \sum_{e_s \in C_{p(s)}} \lambda^{i(e_s) - i(e_c)} \mathbb{1}\left(i(e_s) \geq i(e_c)\right), \tag{1}$$

where e_s and e_c are two encounters; I is the total number of patients, $\lambda \in (0, 1)$ is the time-decay parameter (in our experiments, $\lambda = 0.5$); $\mathbb{1}(x)$ is the indicator function ($\mathbb{1}(x) = 1$ if x is true, otherwise, $\mathbb{1}(x) = 0$); $i(e_s)$ and $i(e_c)$ are the indices of encounter e_s and encounter e_c, respectively, in patient p's encounter sequence C_p. When calculating the co-occurrence frequencies between ICD code c and term s, we only consider cases in which term s has been searched during or after the encounter in which ICD code c was assigned to the patient (i.e., $\mathbb{1}(i(e_s) \geq i(e_c))$). Please note that a_{cs} is not a probability value and can have values greater than 1. A larger a_{cs} indicates a greater likelihood that ICD code c and search term s co-occur.

4.1.2. Representation Learning for ICD Codes and Search Terms—

Note that the co-occurrence matrix A as constructed above is typically sparse because most ICD codes do not co-occur with most search terms. In order to capture the underlying relations between each ICD code and search term that are not observed directly in A, we use a matrix factorization method [14] to learn the representations of ICD codes and search terms which together produce what we observe in matrix A and reconstruct what we do not observe in A. Specifically, we factorize A into two low-rank matrices, $U \in \mathbb{R}^{n \times d}$ and $V \in \mathbb{R}^{m \times d}$ $d < \min(n, m)$, representing ICD codes and search terms, respectively. Each row in matrix U, denoted as $\mathbf{u}_c$, represents the ICD code c, and each row in matrix V, denoted as $\mathbf{v}_s$, represents the search term s. Thus, all ICD codes and search terms are represented by size-d latent vectors that can be learned from matrix A. The co-occurrence "chance" between ICD code c and search term s can be estimated as follows:

$$\hat{a}_{cs} = \mathbf{u}_c \mathbf{v}_s^{\mathrm{T}}, \tag{2}$$

where $\hat{a}_{CS}$ is the estimation of a_{cs}. To learn the representations of each ICD code and search term, we formulate the following optimization problem:

$$\min_{U,V} \|A - UV^T\|_F^2 + \frac{\gamma}{2}\left(\|U\|_F^2 + \|V\|_F^2\right), \tag{3}$$

where $U = [\mathbf{u}_1; \mathbf{u}_2; \cdots; \mathbf{u}_n]$, $V = [\mathbf{v}_1; \mathbf{v}_2; \cdots; \mathbf{v}_m]\gamma$ is the weight for the regularization term; $\|\cdot\|_F$ is the Frobenius norm, and regularization on the Frobenius norm restricts large values in U and V. We solve this problem using an alternative gradient descent method [14]. Alternative representation learning methods include deep learning-based methods [27, 12], which typically require a lot of data for model training. Given the very sparse nature of our data (Section 5.2) we did not consider them appropriate. Instead, the matrix factorization-based method (Equation 2) has proven very effective in learning from sparse data [14].

4.2. HCFMH: Hybrid Collaborative Filtering Model for Healthcare

Our hybrid collaborative filtering model (HCFMH) using encounter information recommends search terms for a patient to clinicians using two factors: (1) search terms used on the patient and (2) past encounters of the patient. For each factor individually, we calculate a recommendation score for candidate terms (see Sections 4.2.1 and 4.2.2). The

two recommendation scores are then combined into a final recommendation score (Section 4.2.3).

4.2.1. Recommendation Score from Previous Search Terms—Studies from the RS field have shown that information about more recent items is more pertinent to generating appropriate recommendations than information about earlier items [27, 16]. Assuming that the same premise applies to searching for patient information, we generate recommendations using the most recent search terms on a patient. We aggregate information about the most recent m_s search terms in the current search session by calculating the mean values of their latent feature representations as follows:

$$\mathbf{m}_p = \frac{1}{m_s} \sum_{i \in S_p(n_p - m_s, n_p)} \mathbf{v}_i, \tag{4}$$

where n_p is the number of all search terms on patient p at the time a recommendation is to be made; m_s is the count of the most recent search terms that are used for recommendation (m_s is a fixed number in our experiments. We evaluate the effect of varying m_s in the parameter study described in Section 6.2.); and $\mathbf{m}_p \in \mathbb{R}^{1 \times d}$ is the aggregated representation of the previous m_s search terms on patient p. The recommendation score of term s for patient p is calculated as the dot-product similarity between $\mathbf{m}_p$ and $\mathbf{v}_s$ as follows:

$$x_{ps} = \mathbf{m}_p \mathbf{v}_s^T . \tag{5}$$

4.2.2. Recommendation Score from Previous Encounters—To calculate the recommendation score for each term, we also use the information from the most recent m_c encounters of each patient. Here, we assume that clinicians tend to search within the context of the most recent diagnoses and that ICD codes recently assigned to the patient are more likely to induce future searches than past ICD codes. Thus, recent ICD codes should be emphasized when recommending search terms. Based on this assumption, we calculate an importance weight for each ICD code c of each patient p. The importance weight is calculated as the normalized dot-product similarity between each ICD code and the most recent m_s search terms as follows:

$$w_{pc} = \frac{\exp\left(\mathbf{u}_c \mathbf{m}_p^T\right)}{\sum_{e' \in C_p(l_p - m_c, l_p)} \sum_{c' \in e'} \exp\left(\mathbf{u}'_c \mathbf{m}_p^T\right)} \tag{6}$$

where $\mathbf{m}_p$ is calculated as shown in Equation 4; l_p is the number of all encounters of patient p at the time the recommendation is to be made; m_c is the number of the most recent encounters that are used for recommendation (m_c is a fixed number in our experiments. We evaluate the effect of varying m_c in the parameter study described in Section 6.2.); e' is an encounter in $C_p(l_p - m_c, l_p)$; and c' is an ICD code in e'. The recommendation score of term s on patient p based on previous encounters is calculated as follows:

$$y_{ps} = \sum_{e \in C_p(l_p - m_c, l_p)} \sum_{c \in e} w_{pc} \mathbf{u}_c \mathbf{v}_s^T, \tag{7}$$

where e is an encounter in $C_p(l_p - m_c, l_p)$ and c is an ICD code in e.

4.2.3. Combination of Recommendation Scores—The recommendation scores of term s on patient p calculated as above are then combined into a final recommendation score as follows:

$$r_{ps} = \alpha\, x_{ps} + (1 - \alpha)\, y_{ps}, \tag{8}$$

where $\alpha \in [0, 1]$ is a predefined weight for the two factors. In Equation 8, α=1 indicates that only previous search terms are used for the recommendation, and α=0 indicates that only previous encounters are used for the recommendation. Terms are sorted by their recommendation scores and the terms with top-N scores are recommended. We evaluate the effect of varying α in the parameter study described in Section 6.2.

4.3. cpHCFMH: Co-occurrence Pattern-based HCFMH

A variation of HCFMH, the cpHCFMH, uses only ICD-search term co-occurrence patterns (Section 4.1) to recommend terms to clinicians. The difference between HCFMH and cpHCFMH is that HCFMH recommends terms that are most relevant to the most recent search terms and the most recent encounters, whereas cpHCFMH recommends terms that are most relevant to *all* previous encounters of a patient. Both methods represent ICD codes and search terms using the representation matrices, U and V, respectively, as learned based on Equation 3.

We aggregate information about all ICD codes from the patient's previous encounters to calculate the recommendation score for each search term. As above, we assume that more recent ICD codes are more likely to induce future searches than past ICD codes. Therefore, we emphasize recent encounters/ICD codes in generating recommendations using a time-decay parameter. The recommendation score of term s for patient p is calculated as follows:

$$r_{ps} = \sum_{e \in C_p(1, l_p)} \sum_{c \in e} \sigma^{i(e_s) - i(e)} \mathbf{u}_c \mathbf{v}_s^T, \tag{9}$$

where e is an encounter in $C_p(1, l_p)$, and c is an ICD code in e; e_s is the most recent encounter at the time the recommendation is to be made; $i(e_s)$ and $i(e)$ are the indices of encounter e_s and encounter e, respectively; and $\sigma \in (0, 1)$ is the time-decay parameter (in our experiments, σ=0.5). Note that, here, the time-decay parameter σ indicates how long ago each encounter occurred before the time of recommendation, whereas the time-decay weight λ in Equation 1 indicates the temporal proximity between an encounter and a search term. Thus, the two time-decay parameters represent different information in the model. Finally, terms are sorted by their recommendation scores and the terms with top-N scores are recommended.

5. Materials

5.1. Baseline Methods

We compared HCFMH and cpHCFMH to the state-of-the-art baseline methods listed below. These RS methods (except TptCF, which we designed for the healthcare context) operate on users and items. To recommend terms using these methods, we considered "patients" and "search terms" analogous to "users" and "items," respectively, in our experiments.

- **Hierarchical Gating Network (HGN)** [16]. HGN selects important items and their important features by adapting gating mechanisms [16] using physicians' action histories. To generate recommendations, HGN uses the identified important items and important features, along with users' general preferences, to calculate recommendation scores. HGN is a state-of-the-art sequential recommendation method.
- **Hybrid Associations Model (HAM)** [21]. HAM models users' long- and short-term preferences from their complete and most recent portion of their action histories, respectively. Short-term preferences contain both high-order and low-order association patterns among items. Both long-term and short-term preferences are used to generate recommendations. HAM has been demonstrated as the state-of-the-art algorithm for sequential top-N recommendation in terms of recommendation performance.
- **Transition-Involved Patient-Term-Similarity-based CF Scoring (TptCF)** [7]. The TptCF model generates recommendations for terms using two factors: (1) patients' similarities and search terms' similarities (similarity-based scoring), and (2) search term dynamic transitions (dynamics-based scoring). For each patient, TptCF calculates a recommendation score for each search term using similarity-based scoring and dynamics-based scoring, and it then recommends the search terms with the highest scores. The supplementary material includes a detailed explanation of the TptCF method.
- **Personalized top-N (PTN)**. The PTN model recommends terms based on the most frequent items in a user's action history. In our experiments, the terms are ranked based on how many times physicians use them to search on each patient, and a patient's top-N most frequently searched terms are recommended to the physicians.

5.2. Data Set

The data used in our experiments consist of searches conducted by physicians at Eskenazi Health in Marion County, Indiana, US, logged from 04/2013 to 05/2016. The data consist of 13,934 patients, their 1,377,381 encounters, 9,565 valid ICD 9 codes and 7,215 unique search terms. We preprocessed the search terms by removing irregular search terms, such as numbers and punctuation, and terms that appear only once, and mapping misspelled terms to terms most similar to them (e.g., "adimssion" to "admission"). We also removed patients who did not have at least two search terms and three encounters. Table 2 summarizes the characteristics of the data set after preprocessing.

The final data set contains 2,955 patients and 2,101 unique search terms. On average, each patient has 10.22 searches and 173.26 encounters, each term was searched 14.37 times over all patients and each encounter had 2.09 ICD codes. Our study was approved by the Indiana University Institutional Review Board (Protocol #1612682149 “Supporting information retrieval in the ED through collaborative filtering”).

Figure 2 shows the distribution of the lengths of the search sequences. The length of a search sequence is defined as the raw number of terms that were searched on for a unique patient in the data set. Figure 2 shows that there are many more short than long sequences. For instance, 96% of all sequences contain 10 or fewer terms. Figure 3 shows the distribution of the number of unique terms per patient. On average, each sequence contains 7.00 unique search terms. In addition, we grouped search terms into sessions as described in Section 3.1. There are 3,488 sessions in the data set. On average, each patient has 1.18 sessions, and each session has 8.66 search terms. It is notable that search sequences are typically very short and the number of unique search terms per patient is very small. This sparsity of data makes recommending terms difficult. Table 3 lists the most frequently searched terms, where frequency was calculated based on how many times each term was searched on all patients.

5.3. Experimental Protocols

We used the following experimental protocol to evaluate our methods on the data set. The search sequences were split into a training and a test set, respectively, before and after a specific cut-off date. The models were trained on the training set. For example, the co-occurrence matrix (Equation 1) was constructed only using search terms and encounters in the training set. This protocol is referred to as cut-off cross validation, denoted as CUTOFF. Figure 4 shows the CUTOFF experimental protocol.

We selected four cut-off dates: 10/01/2013, 04/01/2014, 10/01/2014 and 04/01/2015. These cut-off dates were selected to retain different numbers of search terms and encounters from the search sequences in the training and test sets, respectively. The statistics for training and test data given the four cut-off dates are presented in Table 4. In the CUTOFF setting, all data before a certain date was used to predict information after that date. For those sequences which include terms after the cut-off date, only the first term after the cut-off date will be predicted and evaluated. This approach makes the evaluation applicable to real-life scenarios. However, a shortcoming of CUTOFF is that many early search sequences do not have any terms in the testing set, and many late search sequences do not have any terms in the training set. Sequences that do not have testing terms are still used to train models. Sequences that do not have training terms are not used.

5.4. Evaluation Metrics

We used hit rate at k (HR@k) to compare the different recommendation methods. HR@k measures the proportion of patients whose ground-truth next search term is contained in the top-k recommendations. Higher HR@k values indicate better recommendation performance.

5.5. Parameter Tuning

We used grid search to tune four parameters: m_s (Equation 4), m_c (Equation 6, 7), α (Equation 8) and latent dimension d. We present the best results among different parameter settings in Section 6.

6. Results and Discussions

6.1. Overall Performance

We compared HGN, HAM, TptCF, PTN, cpHCFMH and HCFMH in our experiments. Tables 5, 7, 8 and 9 present the best performance of each method in terms of HR@k ($k \in \{1, 2, 3, 4, 5, 10, 20\}$) for the four cut-off dates, respectively. In Table 5, we present different parameter settings of each method that achieved the best HR@k results for each k value. For simplicity's sake, we only show the parameter setting of each method that achieved the best HR@5 in Tables 7, 8 and 9. We also show the improvement of HCFMH over the second best results in terms of all HR metrics in the tables. In addition, we added the experimental results for the HCFMH model when m_s equaled the number of all previous search terms in the current search session for each patient (i.e., the row for HCFMH with parameter "all" in Table 5).

6.1.1. Overall Performance on Cut-off Date 04/01/2015—Table 5 shows that the HCFMH model outperforms all baseline methods for HR@k with $k \in \{1, 2, 3, 4, 5, 10\}$ and performs second best for HR@20. HCFMH achieved HR@1 =0.0917, meaning the model recommended the next search term for about 9% of all patients correctly. Although this HR@1 value is not high in itself, it is substantially higher than random guessing, which has an expected $HR@1 = \frac{1}{1,915} = 5.22e^{-4}$ (given there are 1,915 search terms in the training set for the cut-off date 04/01/2015, shown in Table 4). That is, HCFMH is 175-fold better than random guessing on HR@1. HCFMH achieved HR@5=0.2569, meaning that the next search term was contained in the top-5 recommendations for about 26% of all patients. Again, this performance is substantially better than random guessing, which has an expected HR@5=5/1, 915=2.6e^{-3}. That is, HCFMH was 98-fold better than random guessing on HR@5.

The second best method is cpHCFMH as it achieved the second best results on HR@2, HR@3, HR@4, HR@5 and HR@10, and the best results on HR@20. The difference between HCFMH and cpHCFMH is that HCFMH uses the most recent search terms and most recent encounters to recommend the next search term, whereas cpHCFMH uses all previous encounters to do so. However, as Table 4 shows, each patient has 167.88 encounters on average. Since past encounters, especially if they occurred a long time ago, may not contain the information that the clinician is interested in at the time of the recommendation, recommendations generated by cpHCFMH are likely to be less accurate than those generated by HCFMH.

PTN performs slightly worse than the methods HCFMH and cpHCFMH. This is probably because PTN uses the most popular search terms in a patient's entire search history for recommendation, and terms that were searched frequently in the past may not be of interest

to the clinician at the time of the recommendation, given the progression of the patient's health condition.

TptCF performs slightly worse than PTN, probably due to data sparsity. As shown in Table 4, the data set contains 2,627 patients and 1,915 unique search terms in the training set. However, each patient had only an average of 8.22 search terms. Thus, data sparsity may have caused TptCF to learn inaccurate patient-patient similarities and term-term similarities, both of which can lead to inaccurate search term recommendations.

HAM and HGN performed worst in this experiment. HGN assumes that each item contributes differently to the next item to be recommended and therefore learns importance weights for items using gating mechanisms. However, data sparsity may cause HGN to learn inaccurate weights for items, leading to poor recommendation performance. HAM, on the other hand, generates recommendations using three factors: users' long-term preferences modeled from all previous items, high-order association patterns modeled from a number of most recent previous items and low-order association patterns modeled from few most recent previous items. Our data set may not contain long-term preferences and the association patterns, and thus HAM may generate less meaningful recommendations and thus have poor recommendation performance.

Table 5 also shows that HCFMH consistently improves over the second best method on all evaluation metrics except HR@20. In practical applications, good performance with small values k in HR@k is preferred since that means that the correct recommendation is among a small number of terms. This makes it easy for the user to select the term of interest. Therefore, reduced performance given large values of k is less of an issue than given small values of k. In addition, HCFMH performs more than 10% better than the second-best models for HR@4 and HR@5. This indicates that HCFMH can push the most relevant search terms to the very top of the recommendation list.

In addition, we conducted extensive experiments to calculate bootstrapped confidence intervals for HR results. For this set of experiments, we compared the proposed models, HCFMH and cpHCFMH, to the best baseline methods, TptCF and PTN, since these two models achieved the second best results for HR@1 and HR@5, respectively, as shown in Table 5. For a fair comparison, we choose only the parameter setting for each model that resulted in the best performance. We present the 95% confidence intervals for each metric in Table 6.

Table 6 shows that HCFMH outperforms all other models for HR@k with $k \in \{2, 3, 4, 5, 10\}$, and cpHCFMH performs second-best on HR@20. Moreover, although TptCF achieved the best HR@1, HCFMH outperforms TptCF with improvements of 10.35%, 31.38%, 22.96%, 24.47%, 28.59% and 5.58% in terms of HR@k with $k \in \{2, 3, 4, 5, 10, 20\}$, respectively. Thus, HCFMH significantly outperforms all other models in terms of most HR metrics for recommending terms.

6.1.2. Performance with Other Cut-off Dates—

Tables 7 and 8 present the performance of the different methods with cut-off dates of 10/01/2014 and 04/01/2014,

respectively. Performance patterns are similar to those shown in Table 5, with HCFMH outperforming the baseline methods for all HR metrics. In addition, all methods tend to score higher on HR with the cut-off date 04/01/2014 (Table 8) than for 10/01/2014 (Table 7) and 04/01/2015 (Table 5). This observation is probably due to the fact that the training set for cut-off date 04/01/2014 contains fewer search terms (1,065) than the other two (1,493 and 1,915, respectively) (see Table 4). Note that our models only recommend terms that have appeared in the training set and cannot recommend new terms. Recommending one of 1,065 search terms is more likely to hit the accurate term by chance than recommending one of 1,915 search terms. Thus, The HCFMH is more likely to recommend search terms accurately with a cut-off date of 04/01/2014 than with 04/01/2015.

Table 9 presents the performance of the different methods for cut-off date 10/01/2013. It shows that HCFMH had the best results for HR@3, HR@4, HR@5 and HR@10, and the second-best results for HR@1 and HR@20. In this experiment, PTN achieved the best HR@1, HR@2 and HR@3 results. Further, HAM performed best in terms of HR@20, and achieved the second best results in terms of HR@2, HR@3, HR@5 and HR@10. In this experiment, there are very few patients and search terms in the training and test sets, respectively, as shown in Table 2. Due to the resulting data sparsity, representations of search terms and ICD codes may not be well learned, causing both HCFMH and cpHCFMH to perform relatively poorly with only little improvement compared to the baseline methods. Table 9 also shows that PTN had identical values for all HR metrics. This observation is probably due to the fact that there are sessions in the test set that contain terms that are not contained in the corresponding training set. In consequence, PTN cannot recommend accurate search terms for these sessions regardless of the value of k.

Comparing Tables 5, 7, 8 and 9 further, we notice that HCFMH tends to significantly outperform the baseline methods when there is sufficient data for training, and does not perform very well when data is very sparse. Overall, for all three data sets, HCFMH consistently outperforms the baseline methods in terms of HR@3, HR@4, HR@5 and HR@10, and improves performance by at least 4.56%, 7.28%, 7.15% and 5.54%, respectively, compared to the second-best method.

6.2. Parameter Study

For the parameter study, we evaluated the models with the cut-off date of 04/01/2015 since it provides sufficient training and testing data. We chose the following set of parameters: m_s=6, m_c=2 and α=0.2, since HCFMH achieved the best HR@2, HR@3 and HR@5 with this set of parameters. To conduct the parameter study, we fixed two of three parameters and evaluated the model with different values of the third parameter. Tables 12, 11 and 10 present the results for the parameter study on HCFMH on m_s, m_c and α, respectively.

Table 10 shows that HCFMH performed best for HR@5 with $m_s \in [5, 9]$. HR@5 decreased when m_s was increased or decreased outside of that range. Recall that m_s is the number of previous search terms used to generate the recommendation. These results indicate that recommendation performance decreases when too few previous search terms do not provide sufficient information or too many previous search terms provide irrelevant information. Table 10 also shows that HCFMH produced the same HR@1 result with different m_s values.

This indicates that varying m_s values may not be sufficient to promote the most relevant search term to the top-ranked position. However, as illustrated by the results for HR@5, doing so may promote more relevant search terms into the top-5 when choosing a proper value of m_s.

Table 11 shows that HCFMH achieved the same HR@1 result with different m_c values, and the best HR@5 with both m_c =2 and 3. Recall that m_c is the number of previous encounters used to recommend the next search term. Table 11 also shows that both HR@5 and HR@10 decreased as m_c varied from 2 (or 3). This indicates that changing m_c values may not help HCFMH promote the most relevant search term to the top-ranked position. However, with m_c set to a small number (e.g., m_c =2), HCFMH performed best for HR@5 and HR@10.

Table 12 shows that when α (Equation 8) increased, HR@1 increased, and HR@10 tended to decrease. Furthermore, HCFMH achieved the best HR@5 results with α equal to 0.2, 0.4 and 0.8. Recall that with α set to 1.0, HCFMH only uses previous search terms to generate recommendations, and with α set to 0, HCFMH only uses previous encounters for recommendation. This indicates increasing the weight of previous search terms improves HR@1 performance but hurts HR@10 performance, whereas increasing the weight of previous encounters hurts HR@1 performance but improves HR@10 performance. Thus, using only previous search terms appears to help promote the most relevant search term to the top-ranked position since HCFMH performed best for HR@1 with m_s equal to 1.0. However, using only previous search terms disregards encounter information, which may be highly relevant to clinicians' searches. Therefore, HCFMH can promote more relevant search terms to the top-10 when m_s is equal to 0. Using both previous search terms and previous encounters, HCFMH was able to achieve the best HR@5 results.

6.3. Analysis of Length of Search Sessions

We evaluated HCFMH on sessions of different sequence lengths in order to better understand the influence of session lengths and information content on recommendation performance. To do so, we first divided sessions in the test set into five groups based on their lengths. Table 13 presents the statistics of the lengths of search sessions in our experiment. For each group, we present the min length, max length and average length of the sessions. In this experiment, we included all previous searches to recommend the next search term. Therefore, we set m_s to "all" in HCFMH. Given m_s set to "all", m_c =2 and α=0.8 allowed HCFMH to perform best, as shown in Table 5. Thus, we selected m_s="all", m_c =2 and α=0.8 as the parameters for our experiment. In addition, m_s=6, m_c =2 and α=0.2 in HCFMH achieve the best HR@2, HR@3 and HR@5 results, as shown in Table 5. For comparison purposes, we also added this set of parameters in our experiment. Finally, in order to understand the effects of parameter α, we added two parameter settings for HCFMH: (1) m_s="all", m_c =2 and α=0.2; and (2) m_s=6, m_c =2 and α=0.8, that is, varying α and keeping the other two parameters fixed according to the parameter settings above. We present the results of these four parameter settings in Figures 5a, 5b, 5c and 5d, respectively.

Figures 5a, 5b, 5c and 5d show the results in terms of HR@1, HR@5 and HR@10 for the five groups of search sessions. The four figures show similar patterns, that is the top-20%-40% longest sessions performed best, and the top-20% longest sessions worst. This

indicates that search terms from a long time ago may not represent the information that the clinicians intend to search at the time of the recommendation. Furthermore, groups with top-20% shortest sessions, top-20%-40% shortest sessions and mid-40%- 60% sessions achieved similar HR@1 and HR@10 values, which are much lower than for those with the top-20%-40% longest sessions. This indicates that using an insufficient number of search terms may limit recommendation quality. On the other hand, using the top-20%-40% longest sessions, that is, sessions with lengths ranging from 9 to 24 (Table13), can help make recommendations more accurate.

Moreover, HCFMH with m_s set to "all" and m_s=6 showed a similar pattern. Specifically, for sessions with length less or equal to 6 (i.e., the top-20% and top-20%-40% shortest sessions), HCFMH produced identical results with the two parameter settings. This is due to the fact that when m_s is larger than the number of all previous search terms in the current session, we use all search terms to recommend the next term. In this case, using m_s=6 produces the same results as using m_s="all". For sessions of mid-40%-60% lengths, given that their average length is 6.76 as shown in Table 5, the six most recent searches constitute 88.76% of a search session on average. Thus, using the six most recent searches in a session produces results similar to using all previous searches for these sessions. For the top-20%-40% longest sessions, the average length is 15.04 as shown in Table 5, and the six most recent searches constitute 39.90% of a search session on average. Furthermore, comparing Figures 5a and 5c, Figure 5b and 5d, we notice that using previous six searches achieves significant improvement compared to using all previous searches in term of HR@5 for top-20%-40% longest sessions. This indicates that although using the most recent six searches discards a lot of information in a session, the most recent, pertinent information is still retained and enables superior recommendation performance. For the top-20% longest sessions, the average length is 62.75 as shown in Table 5. For these sessions, the most recent six searches constitute, on average, 9.56% of a search session. In this scenario, using the most recent six searches to generate recommendations may discard too much information, resulting in poor performance analogous to using all previous searches, which may retain too much irrelevant information.

6.4. Case Study

Finally, we present a case study based on the results described in Section 6.3. Specifically, we extracted the top- 20%-40% longest sessions (as shown in Table 5) and divided the sessions into two groups: top-5 hit group and top-5 miss group, based on whether the test search terms were correctly recommended among the top 5 or not. It turns out that each group had 12 sessions. We examined the test search terms of each group and investigated the following two questions: (1) Was the test search term frequently searched? and (2) had the test search term appeared earlier in the same session? Table 14 presents the test search terms and how many times they were searched on all patients in our data set (i.e., training and test sets). Table 14 also shows the percentage of search sessions that had a corresponding test search term. Table 14 shows that for the top-5 hit group, all search terms in the test set were frequently searched (at least 200 times). However, for the top-5 miss group, only three search terms in the test set were searched over 200 times, and seven search terms were searched less than 100 times. This shows that if the test search term is frequently searched,

the model is more likely to generate accurate recommendations for this search term. On the other hand, if the test search term is infrequently searched, it is hard for the model to generate accurate recommendations. Moreover, in the top-5 hit group, 75.00% of the sessions had their test search terms appear earlier in the same sessions, whereas in the top-5 miss group, only 41.67% of the sessions had the test search terms appear earlier. This indicates that HCFMH is more likely to generate accurate recommendations when the test search term has appeared earlier in the same session.

7. Conclusions

In this manuscript, we developed a model named the Hybrid Collaborative Filtering Model for Healthcare, denoted as HCFMH, to recommend search terms to clinicians. Given that recommender systems are now fairly ubiquitous (as implemented by Netflix, Google, Facebook, Twitter and many other services [30], it is reasonable to explore their application in EHRs. Doing so is even more compelling since information retrieval from EHRs is, in general, inefficient and cumbersome [4, 11, 26, 28].

In terms of recommendation quality, the HCFMH model outperformed the baseline methods with improvements of at least 4.56%, 7.28%, 7.15% and 5.54% over the second-best results with regard to HR@3, HR@4, HR@5 and HR@10, respectively, on data set variations with different degrees of data sparsity. This means that a custom-built algorithm for healthcare produces non-trivial performance gains over current state-of-the-art RS methods.

The expected impact of methods such as HCFMH on the end user experience of retrieving information from EHRs is a key question for our research. Summarizing our evaluation results in general, HCFMH's performance on HR@1, HR@5 and HR@10 ranges from 0.0642 to 0.3237, 0.2569 to 0.4317, and 0.4220 to 0.4460, respectively. This performance compares very favorably with HR@10 values of about 0.200 (Netflix) and 0.330 (MovieLens) [20].

Practically speaking, an HR@5 of 0.4317 means that the probability that the term that a user will search for has more than 43% probability being among the top-five recommended items. In theory, this functionality could save 43% of the keystrokes necessary to search for a term. If achieved in practice, this would constitute a non-trivial reduction of the user effort for searching.

Of course, a key question is to what degree users search (as opposed to browse) for information in EHRs in the first place. The paucity of articles about searching in EHRs in the literature, as well as anecdotal evidence, indicate that browsing is far more predominant than searching. This has implications for the utility of RS algorithms: While it is difficult to generate recommendations with few searches, it is impossible without them.

Analyzing search goals and semantics could help improve our methods in the future. Currently, our methods rely on associations/co-occurrence among search terms and ICD codes to generate recommendations. Understanding *what* a clinician searches for and *why* could help us tailor recommendations more closely to the application scenario than currently possible. Understanding the semantics of a search will require natural processing (NLP)

techniques [1, 13]. NLP can automatically assign large numbers of search terms to categories such as medications and lab tests, a task infeasible to perform manually. Understanding the purpose of a search would require inferring goals from search terms, which could require a large and well-annotated data set. For example, if a clinician searches for blood pressure, we would have to understand whether she is interested in the general condition of the patient or suspects that blood pressure is related to a particular problem. Unfortunately, however, clinician's search goals are neither expressed nor captured. While understanding what clinicians search for and why would highly valuable, it is nontrivial. We plan to investigate this direction in future research.

A notable limitation of the developed methods is that they cannot generate recommendations for a patient who does not have any encounter information in the system. This is the "cold-start" problem described in the recommender systems literature [3]. In future research, we plan to address this problem using patient similarities calculated from demographic information to generate recommendations for cold-start patients. Furthermore, we also plan to implement our recommendation models in the clinic and gather clinician feedback to gain deeper insights into the quality and usefulness of our recommendations.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgements

This project was made possible, in part, by support from the National Science Foundation (grant numbers IIS-1855501 and IIS-1827472), the National Library of Medicine (grant number 1R01LM012605-01A1) and the Lilly Endowment, Inc. Physician Scientist Initiative. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect the views of the funding agencies.

References

[1]. Abacha AB, Zweigenbaum P, 2015. Means: A medical question-answering system combining nlp techniques and semantic web technologies. Information processing & management 51, 570–594.

[2]. Agarwal D, Chen BC, 2009. Regression-based latent factor models, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA. p. 19–28. URL: 10.1145/1557019.1557029, doi:10.1145/1557019.1557029.

[3]. Bobadilla J, Ortega F, Hernando A, Bernal J, 2012. A collaborative filtering approach to mitigate the new user cold start problem. Knowledge-based systems 26, 225–238.

[4]. Christensen T, Grimsmo A, 2008. Instant availability of patient records, but diminished availability of patient information: A multi-method study of GP's use of electronic patient records. BMC Medical Informatics and Decision Making 8. doi:10.1186/1472-6947-8-12.

[5]. Cremonesi P, Koren Y, Turrin R, 2010. Performance of recommender algorithms on top-n recommendation tasks, in: Proceedings of the Fourth ACM Conference on Recommender Systems, Association for Computing Machinery, New York, NY, USA. p. 39–46. URL: 10.1145/1864708.1864721, doi:10.1145/1864708.1864721.

[6]. Deshpande M, Karypis G, 2004. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst 22, 143–177. URL: 10.1145/963770.963776, doi:10.1145/963770.963776.

[7]. Fan Z, Burgun E, Schleyer T, Ning X, 2019. Improving information retrieval from electronic health records using dynamic and multi-collaborative filtering, in: 2019 IEEE International Conference on Healthcare Informatics (ICHI), IEEE. pp. 1–3.

[8]. Hidasi B, Karatzoglou A, 2018. Recurrent neural networks with top-k gains for session-based recommendations, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, ACM. pp. 843–852.

[9]. Hidasi B, Karatzoglou A, Baltrunas L, Tikk D, 2015. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939.

[10]. Hill RG Jr, Sears LM, Melanson SW, 2013. 4000 clicks: a productivity analysis of electronic medical records in a community hospital ed. The American journal of emergency medicine 31, 1591–1594. [PubMed: 24060331]

[11]. Howe JL, Adams KT, Hettinger AZ, Ratwani RM, 2018. Electronic health record usability issues and potential contribution to patient harm. JAMA 319, 1276. doi:10.1001/jama.2018.1171. [PubMed: 29584833]

[12]. Kang WC, McAuley J, 2018. Self-attentive sequential recommendation, in: 2018 IEEE International Conference on Data Mining (ICDM), IEEE. pp. 197–206.

[13]. Koopman B, Bruza P, Sitbon L, Lawley M, 2012. Towards semantic search and inference in electronic medical records: an approach using concept-based information retrieval. The Australasian medical journal 5, 482. [PubMed: 23115582]

[14]. Koren Y, Bell R, Volinsky C, 2009. Matrix factorization techniques for recommender systems. Computer 42, 30–37.

[15]. Kruse CS, Goswamy R, Raval Y, Marawi S, 2016. Challenges and opportunities of big data in health care: A systematic review. JMIR Medical Informatics 4, e38. doi:10.2196/medinform.5359. [PubMed: 27872036]

[16]. Ma C, Kang P, Liu X, 2019. Hierarchical gating networks for sequential recommendation. arXiv preprint arXiv:1906.09217.

[17]. Manor-Shulman O, Beyene J, Frndova H, Parshuram CS, 2008. Quantifying the volume of documented clinical information in critical illness. Journal of Critical Care 23, 245–250. doi:10.1016/j.crc.2007.06.003. [PubMed: 18538218]

[18]. Mazur LM, Mosaly PR, Moore C, Marks L, 2019. Association of the usability of electronic health records with cognitive workload and performance levels among physicians. JAMA Network Open 2, e191709–e191709. [PubMed: 30951160]

[19]. Ning X, Desrosiers C, Karypis G, 2015. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. Springer US, Boston, MA. pp. 37–76. URL: 10.1007/978-1-4899-7637-6_2, doi:10.1007/978-1-4899-7637-6_2.

[20]. Ning X, Karypis G, 2011. Slim: Sparse linear methods for top-n recommender systems, in: Proc. IEEE 11th Int. Conf. Data Mining, pp. 497–506. doi:10.1109/ICDM.2011.134.

[21]. Peng B, Ren Z, Parthasarathy S, Ning X, 2020. Ham: Hybrid associations model with pooling for sequential recommendation. arXiv preprint arXiv:2002.11890.

[22]. Rendle S, Freudenthaler C, Schmidt-Thieme L, 2010. Factorizing personalized markov chains for next-basket recommendation, in: Proceedings of the 19th international conference on World wide web, ACM. pp. 811–820.

[23]. Ricci F, Rokach L, Shapira B, 2011. Introduction to recommender systems handbook, in: Recommender systems handbook. Springer, pp. 1–35.

[24]. Ross M, Wei W, Ohno-Machado L, 2014. “big data” and the electronic health record. Yearbook of medical informatics 9, 97–104. doi:10.15265/IY-2014-0003. [PubMed: 25123728]

[25]. Ruppel H, Bhardwaj A, Manickam RN, Adler-Milstein J, Flagg M, Ballesca M, Liu VX, 2020. Assessment of electronic health record search patterns and practices by practitioners in a large integrated health care system. JAMA Network Open 3, e200512. doi:10.1001/jamanetworkopen.2020.0512. [PubMed: 32142128]

[26]. Smelcer JB, Miller-Jacobs H, Kantrovich L, 2009. Usability of electronic medical records. Journal of usability studies 4, 70–84.

[27]. Tang J, Wang K, 2018. Personalized top-n sequential recommendation via convolutional sequence embedding, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, ACM. pp. 565–573.

[28]. Vehko T, Hyppönen H, Puttonen S, Kujala S, Ketola E, Tuukkanen J, Aalto AM, Heponiemi T, 2019. Experienced time pressure and stress: electronic health records usability and information

technology competence play a role. BMC Medical Informatics and Decision Making 19. doi:10.1186/s12911-019-0891-z.

[29]. Wilkerson ML, Henricks WH, Castellani WJ, Whitsitt MS, Sinard JH, 2015. Management of laboratory data and information exchange in the electronic health record. Archives of Pathology & Laboratory Medicine 139, 319–327. doi:10.5858/arpa.2013-0712-so. [PubMed: 25724028]

[30]. Yang X, Guo Y, Liu Y, Steck H, 2014. A survey of collaborative filtering based social recommender systems. Computer communications 41, 1–10.

Highlights

- Searching for the same or similar information on similar patients is inefficient.
- Recommending search terms automatically could help alleviate this problem.
- We use previous searches and patient encounters to recommend search terms.
- Our new models outperform all current methods for top-N search term recommendation.
- Recommender systems could make information retrieval more efficient and effective.

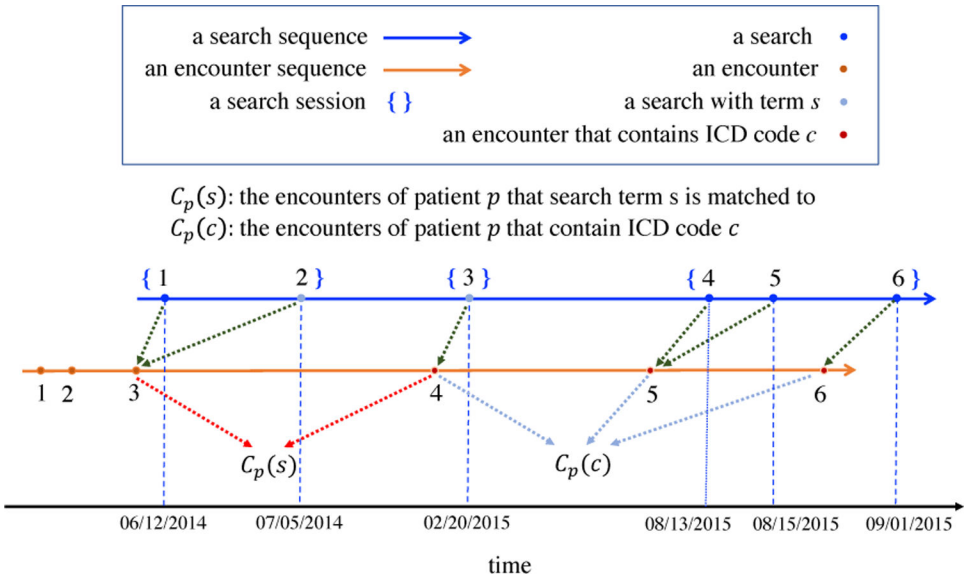


Figure 1:

Data preprocessing protocols illustrated using search terms, encounters and ICD codes for a patient

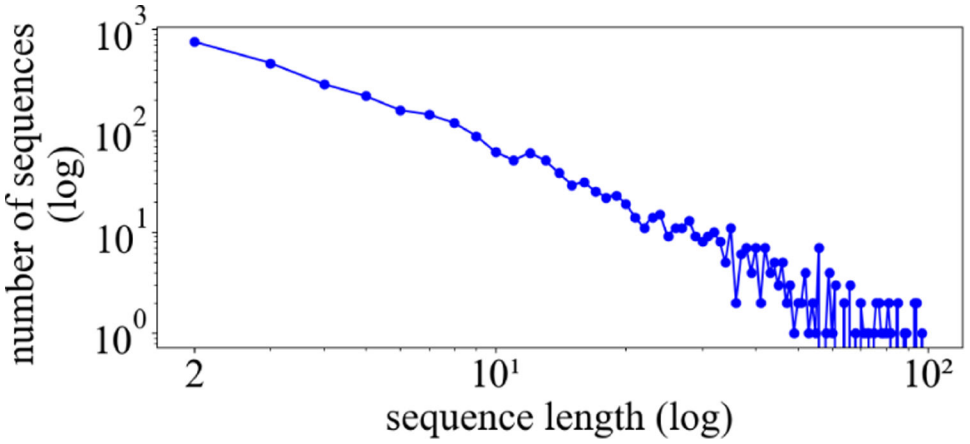


Figure 2:
Distribution of Search Sequence Lengths

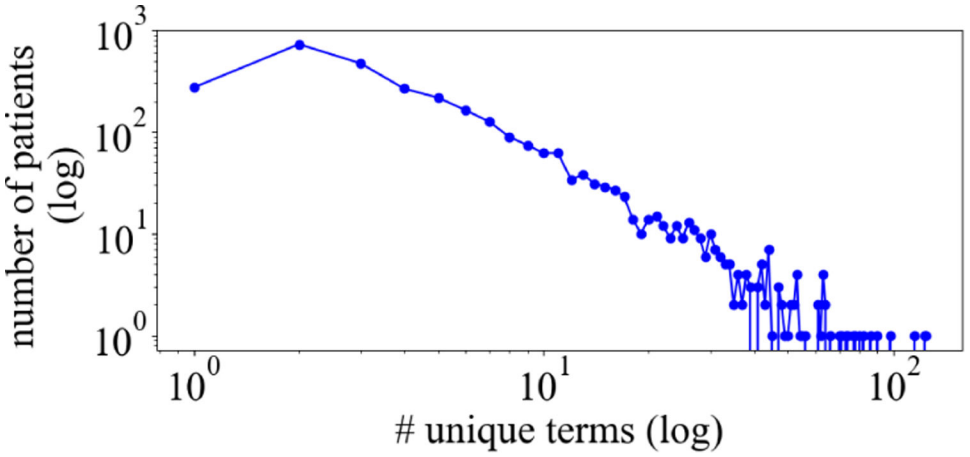


Figure 3:
Distribution of Unique Terms per Patient

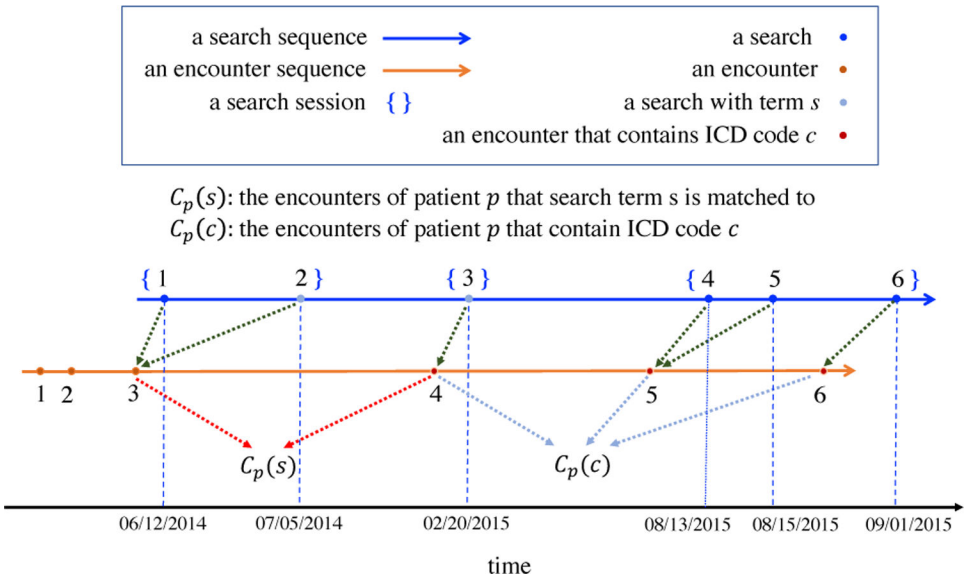
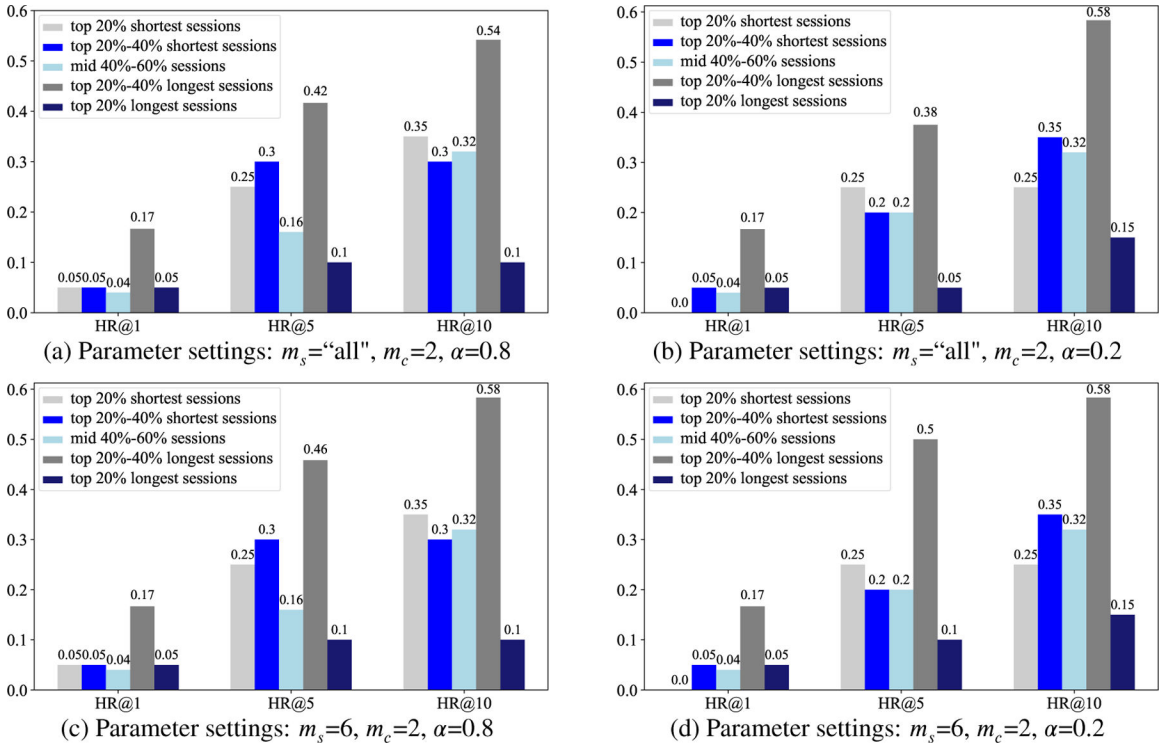


Figure 4:
CUTOFF Experimental Protocol



(a) Parameter settings: m_s="all", m_c=2, α=0.8

(b) Parameter settings: m_s="all", m_c=2, α=0.2

(c) Parameter settings: m_s=6, m_c=2, α=0.8

(d) Parameter settings: m_s=6, m_c=2, α=0.2

Figure 5:
Performance Comparison for Sessions with Different Lengths

Table 1

Notations

notation	meaning
S_p	the sequence of patient p's chronologically sorted searches
$S_p(i, j)$	subsequence of S_p from the i-th search to the j-th search
C_p	the sequence of patient p's chronologically sorted encounters
$C_p(i, j)$	subsequence of C_p from the i-th encounter to the j-th encounter
$C_p(c)$	the encounters of patient p that contain ICD code c
$C_p(s)$	the encounters of patient p that search term s is matched to
$n/m/l$	number of ICD codes/search terms/patients
d	the dimension of representations
n_p/l_p	the number of all search terms/encounters on patient p at the time when the recommendation is to be made
m_s/m_c	the number of previous search terms/encounters that are used for recommending a search term

Table 2

Dataset Statistics

Variables	Statistics
Number of	
patients	2,955
unique search terms	2,101
unique ICD 9 codes	7,027
encounters	511,987
sessions	3,488
Average number of	
searches per patient	10.22
unique search terms per patient	7.00
encounters per patient	173.26
sessions per patient	1.18
searches per term	14.37
previous encounters per term	228.89
search records per session	8.66
unique ICD 9 codes per encounter	2.09

Table 3

Most Frequent Search Terms

Term	Freq	Term	Freq	Term	Freq
hiv	2,144	culture	456	urine	284
a1c	854	inr	431	pap	279
creatinine	709	colonoscopy	409	height	270
weight	706	tsh	401	hgb	253
ekg	634	troponin	371	ct	224
cbc	542	ldl	352	urology	221
bmp	475	echo	316		

In this table, “Freq” represents frequency.

Table 4

Dataset Statistics for Different Cut-off Dates

Cut-off Date	P_t	P_e	T_t	T_e	S_t	T_t/S_t	E_t/P_t
04/01/2015	2,627	109	1,915	442	2,854	8.22	167.88
10/01/2014	2,086	20	1,493	56	2,224	6.30	167.37
04/01/2014	1,129	139	1,065	254	1,171	6.12	164.84
10/01/2013	248	11	373	32	249	4.57	178.84

In this table, P_t indicates the number of patients in the training set; P_e is the number of patients in the test set; T_t is the number of unique search terms in the training set; T_e is the number of unique search terms in the test set; S_t is the number of search term sessions in the training set; T_t/S_t is the average number of search terms per session in the training set; E_t/P_t is the average number of encounters per patient in the training set.

Table 5

Performance Comparison with Cut-off Date 04/01/2015

method	parameters			HR@1	HR@2	HR@3	HR@4	HR@5	HR@10	HR@20
HGN	5	1	-	0.0798	0.1063	0.1247	0.1329	0.1472	0.1963	0.3129
	20	3	-	0.0654	0.1002	0.1309	0.1391	0.1616	0.2188	0.3395
	10	2	-	0.0593	0.0941	0.1125	0.1309	0.1554	0.2025	0.3476
HAM	9	2	1	0.0675	0.0941	0.1145	0.1391	0.1616	0.1943	0.3211
	30	3	3	0.0818	0.0941	0.1022	0.1043	0.1125	0.1963	0.3190
	30	3	2	0.0736	0.1084	0.1247	0.1350	0.1411	0.1738	0.3292
	15	2	3	0.0736	0.0961	0.1288	0.1431	0.1472	0.1861	0.3374
	30	3	1	0.0613	0.0941	0.1125	0.1309	0.1411	0.2168	0.3354
	4	3	2	0.0716	0.0961	0.1145	0.1288	0.1391	0.1902	0.3517
TptCF	0.1	0.1	0.9	**0.0859**	0.1309	0.1595	0.1861	0.2045	0.2638	0.3722
	0.1	0.1	0.7	0.0736	0.1247	0.1575	0.1881	0.2106	0.2945	0.3742
	0.1	0.1	0.5	0.0736	0.1186	0.1493	0.1779	0.1984	0.3006	0.3763
	0.1	0.1	0.3	0.0716	0.1145	0.1472	0.1800	0.1984	0.2822	0.3804
PTN	-	-	-	0.0734	0.1193	0.1835	0.2110	**0.2294**	0.2844	0.3211
cpHCFMH	64	0.01	-	0.0642	0.1284	**0.2018**	**0.2202**	**0.2294**	0.3211	***0.4312**
	32	0.05	-	0.0642	0.1284	0.1835	0.2018	0.2202	**0.3303**	0.4128
	32	0.01	-	0.0734	**0.1376**	0.1927	0.2018	0.2202	0.3211	***0.4312**
HCFMH	6	2	0.2	0.0642	***0.1468**	***0.2110**	0.2294	***0.2569**	0.3394	0.3945
	6	2	0.0	0.0550	0.1376	0.1927	***0.2477**	0.2477	***0.3486**	0.3945
	4	1	1.0	***0.0917**	0.1376	0.1743	0.2294	***0.2569**	0.3211	**0.4220**
	all	2	0.8	0.0734	0.1376	0.1835	0.2110	0.2477	0.3303	0.3945
Improvement				6.75%	6.69%	4.56%	12.49%	11.99%	5.54%	−2.13%

The best performance under each metric over all the methods is **bold** with *.

The second best performance for under each metric over all the methods is **bold**. The “improvement” represents the percentage improvement from the best performance over the second best performance. The best performance within each method under each metric is underlined. The three parameters for TptCF are patient similarity threshold, term similarity threshold and weighting parameter [7]; the two

parameters for HGN are the number of previous purchases/ratings that are used for recommendation and the number of next purchases/ratings that are recommended for [16]; the three parameters for HAM are the number of items of high-order sequential association, the number of items of low-order sequential association, and the number of next purchases/ratings that are recommended for [21]; the two parameters for cpHCFMH are dimension of latent features and regularization weight; the three parameters for HCFMH are the number of previous search terms that are used for recommendation, the number of previous encounters that are used for recommendation and the weighting parameter (i.e., α). We didn't present latent dimension k in the table since it is equal to 32 for all the HCFMH results. The value "all" for parameter m_S indicating considering all previous search terms for recommendation.

Table 6

95% Confidence Interval on HR Results with Cut-off Date 04/01/2015

method	HR@1	HR@2	HR@3	HR@4	HR@5	HR@10	HR@20
TptCF	**0.0859±0.0026**	0.1304±0.0032	0.1584±0.0034	0.1855±0.0036	0.2043±0.0038	0.2634±0.0039	0.3712±0.0042
PTN	0.0737±0.0050	0.1222±0.0061	0.1861±0.0069	0.2131±0.0075	0.2328±0.0075	0.2885±0.0080	0.3244±0.0085
cpHCFMH	0.0657±0.0048	0.1272±0.0063	0.1993±0.0082	0.2187±0.0087	0.2287±0.0089	0.3215±0.0094	**0.4367±0.0090**
HCFMH	0.0648±0.0046	**0.1439±0.0072**	**0.2081±0.0084**	**0.2281±0.0087**	**0.2543±0.0089**	**0.3387±0.0096**	0.3919±0.0099

The best performance under each metric over all the methods is **bold**. For TptCF, we set patient similarity threshold as 0.1, term similarity threshold as 0.1 and weighting parameter as 0.9. For cpHCFMH, we set dimension of latent features as 64 and regularization weight as 0.01. For HCFMH, we set the number of previous search terms as 6, the number of previous encounters as 2 and the weighting parameter (i.e., α) as 0.2.

Table 7

Performance Comparison with Cut-off Date 10/01/2014

method	parameters			HR@1	HR@2	HR@3	HR@4	HR@5	HR@10	HR@20
HGN	20	3	-	0.0854	0.1250	0.1437	0.1604	0.1833	0.2354	0.3083
HAM	30	1	3	0.0583	0.0854	0.1229	0.1521	0.1854	0.2396	0.3208
TptCF	0.1	0.1	0.1	0.0750	0.1146	0.1479	0.1708	0.1979	0.2729	**0.3479**
PTN	-	-	-	***0.1500**	**0.1500**	**0.1500**	0.1500	0.1500	0.1500	0.1500
cpHCFMH	32	0.05	-	**0.1000**	**0.1500**	**0.1500**	**0.2000**	**0.2000**	**0.3000**	***0.4000**
HCFMH	10	1	0.0	***0.1500**	***0.2000**	***0.2000**	***0.2500**	***0.2500**	***0.3500**	***0.4000**
Improvement				50.00%	33.33%	33.33%	25.00%	25.00%	16.67%	14.98%

In this table, the best performance under each metric over all the methods is **bold** with *.

The second best performance for under each metric over all the methods is **bold**. The parameter columns of each method are corresponding to those in Table 5.

Table 8

Performance Comparison with Cut-off Date 04/01/2014

method	parameters			HR@1	HR@2	HR@3	HR@4	HR@5	HR@10	HR@20
HGN	2	3	-	0.1458	0.1852	0.2075	0.2281	0.2556	0.2916	0.3688
HAM	25	0	3	0.0755	0.1578	0.2007	0.2367	0.2590	0.3002	0.3585
TptCF	0.1	0.1	0.7	0.1407	0.2024	0.2264	0.2436	0.2676	**0.3276**	0.3928
PTN	-	-	-	0.2302	0.2374	0.2734	0.2950	0.3094	0.3094	0.3094
cpHCFMH	32	0.05	-	**0.3165**	**0.3453**	**0.3741**	**0.3957**	**0.4029**	*0.4460	**0.4820**
HCFMH	7	1	0.4	*0.3237	*0.3597	*0.4101	*0.4245	*0.4317	*0.4460	*0.5324
Improvement				7.20%	4.17%	10.42%	7.28%	7.15%	36.14%	10.46%

In this table, the best performance under each metric over all the methods is **bold** with *.

The second best performance for under each metric over all the methods is **bold**. The parameter columns of each method are corresponding to those in Table 5.

Table 9

Performance Comparison with Cut-off Date 10/01/2013

method	parameters			HR@1	HR@2	HR@3	HR@4	HR@5	HR@10	HR@20
HGN	25	3	-	0.0526	0.0643	0.0994	0.1053	0.1404	0.1637	0.2339
HAM	10	3	3	0.0468	**0.0994**	**0.1637**	0.1696	**0.1988**	**0.2456**	***0.3216**
TptCF	0.1	0.1	0.7	0.0175	0.0234	0.0292	0.0351	0.0468	0.0760	0.1345
PTN	-	-	-	***0.1818**	***0.1818**	***0.1818**	**0.1818**	0.1818	0.1818	0.1818
cpHCFMH	32	0.05	-	**0.0909**	0.0909	0.0909	0.0909	***0.2727**	***0.2727**	**0.2727**
HCFMH	10	1	0.0	**0.0909**	0.0909	***0.1818**	***0.2727**	***0.2727**	***0.2727**	**0.2727**
Improvement				-50.00%	−50.00%	11.06%	50.00%	37.17%	11.03%	−15.21%

In this table, the best performance under each metric over all the methods is **bold** with *.

The second best performance for under each metric over all the methods is **bold**. The parameter columns of each method are corresponding to those in Table 5.

Table 10

Parameter Study of HCFMH for m_s (m_c=2, α=0.2)

m_s	HR@1	HR@5	HR@10
1	0.0642	0.2110	**0.3486**
2	0.0642	0.2477	0.3303
3	0.0642	0.2202	0.3394
4	0.0642	0.2477	0.3394
[5, 9]	0.0642	**0.2569**	0.3394
10	0.0642	0.2477	0.3394
all	0.0642	0.2202	0.3394

In this table, the best performance for each metric is **bold**.

Table 11

Parameter Study of HCFMH for m_c (m_s=6, α=0.2)

m_c	HR@1	HR@5	HR@10
1	0.0642	0.2110	0.2844
2	0.0642	**0.2569**	**0.3394**
3	0.0642	0.2385	**0.3394**
4	0.0642	0.2294	0.3303

In this table, the best performance for each metric is **bold**.

Table 12

Parameter Study of HCFMH for α (m_s=6, m_c =2)

α	HR@1	HR@5	HR@10
0.0	0.0550	0.2477	**0.3486**
0.2	0.0642	0.**2569**	0.3394
0.4	0.0642	0.**2569**	0.3303
0.6	0.0734	0.2477	0.3211
0.8	0.0734	0.**2569**	0.3394
1.0	**0.0917**	0.2294	0.3119

In this table, the best performance for each metric is **bold**.

Table 13

Statistics on Lengths of Sessions

length of sessions	min	max	mean
top 20% shortest sessions	2	2	2.00
top 20%-40% shortest sessions	3	5	3.95
mid 40%-60% sessions	6	8	6.76
top 20%-40% longest sessions	9	24	15.04
top 20% longest sessions	26	202	62.75

Table 14

Test Search Terms and Frequencies

Top-5 Hit Group		Top-5 Miss Group	
Term	**Freq (%)**	**Term**	**Freq (%)**
hiv	2,144 (33.33)	cbc	542 (8.33)
a1c	854 (8.33)	troponin	371 (8.33)
creatinine	709 (8.33)	ct	224 (8.33)
weight	706 (8.33)	neurol	119 (8.33)
ekg	634 (8.33)	operation	101 (8.33)
cbc	542 (8.33)	culture & blood	79 (8.33)
bmp	475 (8.33)	hep	51 (8.33)
inr	431 (8.33)	cytol	45 (8.33)
pap	279 (8.33)	prealb	40 (8.33)
		tspot	31 (8.33)
		aldo	14 (8.33)
		form	5 (8.33)

In this table, Top-5 Hit Group is the group in which testing search terms are among top-5 recommendations; Top-5 Miss Group is the group in which testing terms are not among top-5 recommendations; “Freq” indicates how many times each term is searched on all the patients; “%” is the perentage of search sessions that have a corresponding testing search term among all the search sessions in each group.