Recommending MeSH terms for annotating biomedical articles

Minlie Huang,[1,2] Aurélie Névéol,[2] Zhiyong Lu[2]

[1]State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing, PR China
[2]National Center for Biotechnology Information (NCBI), US National Library of Medicine, National Institutes of Health, Bethesda, Maryland, USA

Correspondence to
Dr Zhiyong Lu, National Center for Biotechnology Information (NCBI), National Library of Medicine (NLM), National Institutes of Health (NIH), Building 38A, Room 1003A, 8600 Rockville Pike, Bethesda, MD 20894, USA;
zhiyong.lu@nih.gov

Received 21 December 2010
Accepted 8 April 2011
Published Online First
25 May 2011

ABSTRACT

Background Due to the high cost of manual curation of key aspects from the scientific literature, automated methods for assisting this process are greatly desired. Here, we report a novel approach to facilitate MeSH indexing, a challenging task of assigning MeSH terms to MEDLINE citations for their archiving and retrieval.

Methods Unlike previous methods for automatic MeSH term assignment, we reformulate the indexing task as a ranking problem such that relevant MeSH headings are ranked higher than those irrelevant ones. Specifically, for each document we retrieve 20 neighbor documents, obtain a list of MeSH main headings from neighbors, and rank the MeSH main headings using ListNet—a learning-to-rank algorithm. We trained our algorithm on 200 documents and tested on a previously used benchmark set of 200 documents and a larger dataset of 1000 documents.

Results Tested on the benchmark dataset, our method achieved a precision of 0.390, recall of 0.712, and mean average precision (MAP) of 0.626. In comparison to the state of the art, we observe statistically significant improvements as large as 39% in MAP (p-value <0.001). Similar significant improvements were also obtained on the larger document set.

Conclusion Experimental results show that our approach makes the most accurate MeSH predictions to date, which suggests its great potential in making a practical impact on MeSH indexing. Furthermore, as discussed the proposed learning framework is robust and can be adapted to many other similar tasks beyond MeSH indexing in the biomedical domain. All data sets are available at: http://www.ncbi.nlm.nih.gov/CBBresearch/Lu/indexing.

INTRODUCTION

As one of the largest repositories for biomedical articles, PubMed comprises more than 20 million documents to date. The current volume of the biomedical literature and its rapid growth pose great challenges for service providers in terms of management, searching capabilities, and indexing.[1] To facilitate these processes, the US National Library of Medicine (NLM) developed the Medical Subject Headings (MeSH)[i], a controlled vocabulary for describing various biomedical topics such as diseases, chemicals, and drugs, to index articles in MEDLINE. MeSH indexing has been shown to greatly facilitate document retrieval,[2 3] document clustering,[4] and bioinformatics research.[5]

Manually assigning MeSH terms to biomedical articles is a complex, subjective, and time-consuming task that requires human comprehension of the articles and familiarity with the MeSH controlled vocabulary. As a result, indexing consistency between different indexers varies depending on the type and category of indexing terms. For instance, Funk and Reid (1983)[6] reported a consistency of 48.2% for MeSH main heading assignment. Furthermore, it is difficult to assign MeSH terms to citations immediately after they become searchable online. According to the NLM customer service, time to index varies greatly between all of the different works that MEDLINE indexes. According to their recent statistical analysis, 25% of the citations are completed within 30 days of receipt, 50% within 60 days, and 75% within 90 days.

In addition, manual indexing is expensive. As pointed out by Aronson *et al* (2000),[7] 'the total cost of indexing at the NLM includes data entry, NLM staff indexing and revising, contract indexing, equipment, and telecommunications costs.' Hence, to help improve the consistency and timely availability of MeSH indexing and cope with the increasing cost of human indexing in the era of flat/reduced NIH budgets, much effort has been devoted to developing tools that automatically produce MeSH terms for biomedical articles. These tools typically rely on one or more of the following techniques. (1) Selecting MeSH terms from the *k*-nearest neighbor documents as recommendations for the target document.[3 8 9] This technique is based on the assumption that documents similar in content would share similar MeSH term annotations. (2) Using probabilistic models or machine learning methods to learn the association between the document text and a MeSH term.[10–12] (3) Using domain-specific knowledge resources such as MetaMap[13] and trigram.[2] In 2000, the NLM launched its own indexing initiative project[7] to investigate automatic indexing methods for biomedical documents, which led to the development of the Medical Text Indexer (MTI).[14] MTI can assist human annotators with indexing recommendations in the form of MeSH main headings and more recently, main heading/subheading pairs.[15] MTI currently relies on a combination of techniques in both (1) and (3).

In line with the indexing initiative, the goal of this work is to further investigate automatic indexing methods to assist manual curation with MeSH indexing recommendations. Building on the aforementioned indexing technique (1), we hypothesize that the application of a ranking algorithm can significantly improve the quality of automatic indexing recommendations, which would be useful in daily indexing practice. Like

[i]A history of MeSH is available at: http://www.nlm.nih.gov/mesh/mesh_at_50/history_of_mesh.html.

methods based on the neighbor documents,[8 9] our approach obtains from neighbor documents an initial list of MeSH terms for each target article because we observe that over 85% of the gold-standard MeSH annotations for a target document are present in its nearest 20 neighbors. Instead of simply summing the affinity scores between the target document and its neighbors, we approach this task as a ranking problem and adopt a learning-to-rank algorithm to address it. Learning-to-rank algorithms have been studied extensively in document information retrieval and text mining communities.[16] We adopted ListNet[17] in our approach because it fits our problem naturally (detailed in the 'Problem formulation and the learning algorithm' section). Furthermore, we created a set of novel features for the learning algorithm such as the language translation probability features and the query likelihood features.

METHODS

Overview of our approach

There are three steps in our approach, as shown in figure 1. First, we adapt the PubMed Related Articles algorithm[18] to retrieve from the MEDLINE database *k*-nearest neighbors for each target document with a modification so that MeSH terms are not used in the computation of the neighbor documents. In other words, the similarity between documents is solely based on the words they have in common. In this regard, each common word is assigned a numerical weight to represent its importance in relation to the two documents in comparison. The similarity between two documents is computed as the summed weights of all of the terms the two documents have in common. Thus, for a given document, its neighbors are identified as the most similar (ie, with highest summed weights) documents found. We refer interested readers to the online explanation[ii] and Lin *et al*[18] for further information on this algorithm. We do so because such information would not be available for those to-be-indexed target articles in realistic circumstances. The optimal selection of parameter *k* is explained in our experiment.

The second step is to collect all the MeSH terms assigned to those *k*-nearest neighbor documents obtained in the previous step. To compare with prior studies,[9 14] we only considered the main headings and removed subheadings attached to the main headings. Let us see an example in figure 2: the three MeSH terms: 'Anti-Inflammatory Agents, Non-Steroidal/administration & dosage', 'Anti-Inflammatory Agents, Non-Steroidal/chemistry', and 'Anti-Inflammatory Agents, Non-Steroidal/therapeutic use' condensed into a single MeSH main heading 'Anti-Inflammatory Agents, Non-Steroidal' after discarding the three respective subheadings 'administration & dosage', 'chemistry', and 'therapeutic use'. Note that MeSH terms mentioned later in this paper are always referring to main headings as exemplified.

In the third step, each main heading in the initial list is assigned a score by the ranking algorithm. The top N ranked main headings are considered relevant to the target article and can be subsequently recommended to human indexers. To follow the lead of MTI,[14] we set the number N to be 25.

Problem formulation and the learning algorithm

We approach the task of MeSH term indexing as a ranking problem. Given a target article D, we first obtain an initial list of MeSH main headings $\{MH_1,MH_2,\ldots,MH_n\}$ from its neighbor documents. Each main heading is then represented as a feature vector as $x_i = (x_1^i, x_2^i, \ldots, x_m^i)$, where m is the number of features.

[ii]http://www.ncbi.nlm.nih.gov/books/NBK3827/#pubmedhelp.Computation_of_Related_Citations.

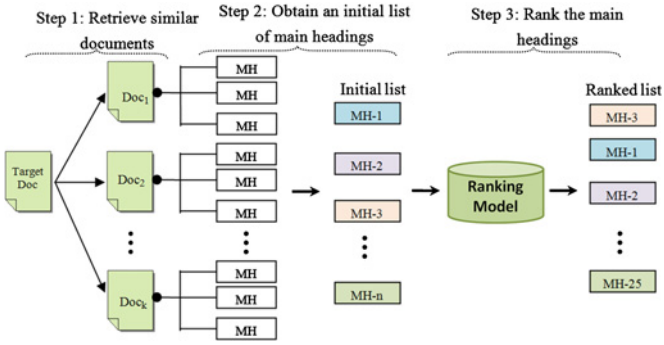

Figure 1 An overview of our approach. MH, main heading.

The learning objective is to find a ranking function $f(x)$ which can assign a score to each main heading based on the feature vector and subsequently use the scores to rank relevant main headings of the target document ahead of those irrelevant ones. We chose to use ListNet,[17] a newly proposed learning-to-rank algorithm that sorts results based on a list of scores, to learn such a function as follows.

First, for the learning purpose we obtained a training set comprising biomedical articles with human assigned MeSH terms from MEDLINE (used as the gold standard). For each target article in the training set, we obtain the corresponding initial list $\{MH_1,MH_2,\ldots,MH_n\}$ from its neighbors and label a main heading 1 if it was manually assigned to the target article, 0 otherwise. As a result, for the list of main headings from its neighbor documents, we obtain a corresponding list $\{y_1, y_2,\ldots,y_n\}$, where $y_i \epsilon \{0,1\}$.

Meanwhile, the ranking function $f(x)$ can be learned to assign a second score list $F=\{f_1,f_2,\ldots,f_n\}$ to these main headings. Each score in Y and F measures the likelihood of assigning a specific MeSH term to an article by human indexers and the ranking function, respectively. Thus, the probability that a main heading MH_i can be placed at the first position of a ranked list can be quantified, according to the two scoring schemas, respectively, as: $\Pr(y_i) \propto y_i$ and $\Pr(f_i) \propto f_i$. To avoid zero probabilities (y_i or f_i might be zero), the exponential function is used:

$$\Pr(y_i) \propto \frac{\exp(y_i)}{\sum_{j=1}^{n}\exp(y_j)}, \quad \Pr(f_i) \propto \frac{\exp(f_i)}{\sum_{j=1}^{n}\exp(f_j)}$$

$\Pr(y_1),\Pr(y_2),\ldots,\Pr(y_n)$ form a probability distribution as their sum equals 1. This distribution derives from the gold standard.

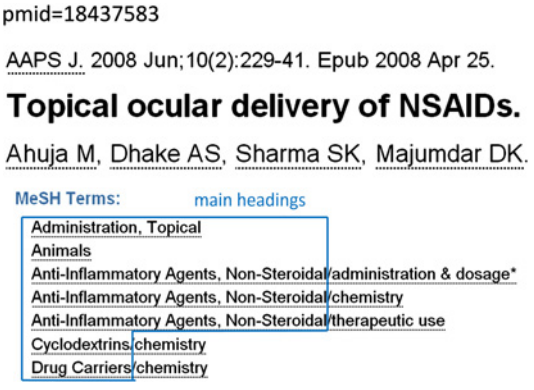

Figure 2 Sample MeSH terms assigned to a MEDLINE article. The terms inside the blue box are main headings, and those outside the blue box are subheadings.

On the other hand, $\Pr(f_1),\Pr(f_2),\ldots,\Pr(f_n)$ forms another distribution, predicted by the ranking function. The goal of the learning process is to do optimization such that the machine predictions are maximally aligned with the gold standard. To do that, we minimize the cross entropy between the two probability distributions as it measures the difference between two distributions:

$$L(Y,F) = -\sum_{j=1}^{n} \Pr(y_j) * \log \Pr(f_j) \quad (1)$$

In ListNet, the ranking function $f(x)$ is defined as a simple linear function, as follows:

$$f_w(x_i) = w^T x_i = \sum_{l=1}^{m} (w_l * x_l^i) \quad (2)$$

Then the gradient of loss function $L(Y,F)$ with respect to the parameter vector w can be calculated as follows:

$$\Delta w = \frac{\partial L(Y,F)}{\partial w} = -\sum_{j=1}^{n} \Pr(y_j) \frac{\partial f_w(x_j)}{\partial w} + \frac{1}{\sum_{k=1}^{n} \exp(f_w(x_k))} \sum_{j=1}^{n} \exp(f_w(x_j)) \frac{\partial f_w(x_j)}{\partial w} \quad (3)$$

Formula (3) defines the gradient for only one list of features vectors $\{x_1,x_2, \ldots,x_n\}$. The gradient for many lists of feature vectors can be simply summed up over all those lists. During training, after initialized to zeros, the w parameter vector is updated with a gradient descent method: $w=w-\eta^*\Delta w$ where η is the learning rate.

From the above formulations, we can see that the learning is performed on a set of lists of samples. The scores for a list of samples, coming from either the gold standard (human annotation) or prediction of the ranking function, are transformed to a probability distribution. Thus the relationship among samples in a list is encoded into a probability distribution. In the context of MeSH indexing, each document comes with a corresponding list of MeSH term candidates (see figure 1) to be ranked. The gold standard of each article is represented as a list of relevant MeSH terms (as opposed to only one relevant label for each instance in many other learning problems). Thus, our selection of the list-wise learning-to-rank algorithm (ListNet) fits squarely to this problem.

Features

We developed various novel features which can be categorized into several groups. When we computed all these features, both the MeSH term and the source text (the title and abstract) were preprocessed by a number of natural language processing techniques. The preprocessing includes tokenization (segmenting sentences into words), regularization (removing punctuations and digits), and normalization (converting words into base forms).

Neighborhood features

In this work, we computed two kinds of neighborhood features: the first counted the number of neighbor documents in which a candidate MeSH term appears. For the second feature, instead of counting documents, we summed document similarity scores. The two features are formulated as follows:

$$freq(MH|D) = |\{D_i | MH \in D_i, D_i \in \Omega_k\}| \quad (4)$$

$$sim(MH|D) = \sum_{MH \in D_i;\, D_i \in \Omega_k} sim(D, D_i) \quad (5)$$

where Ω_k is the k-nearest neighbors for a target document D and $\mathrm{sim}(D_i,D_j)$ is the similarity score between a target document and its neighbor document. The motivation for constructing these features is twofold. First, if a MeSH term appears in more neighbor documents, it is more likely to be assigned to the target document. Second, if a MeSH term appears in documents that are more similar to the target document, it is more likely to be relevant to the target document. As described in the 'Comparison to other methods' section, when used alone these two features represented two baseline ranking strategies and they in fact showed very strong performance.

Word unigram/bigram overlap features

We counted the number of unigrams/bigrams overlapping between the MeSH term and the title (or the abstract), dividing by the total number of unigrams or bigrams in the MeSH term. A unigram consists of a single word and a bigram two sequential words. Accordingly, we generated two features: one counted unigram overlap with the title, and the other counted bigram overlap with the title and abstract. These two features give a direct way of measuring the surface similarity between the MeSH term and the document text.

Translation probability features

The IBM translation model[19] was used to compute the translation probability feature—the probability of translating the title or abstract into a set of MeSH terms. The motivation behind this is that the article was written in the author's language, while the set of MeSH terms was selected from the indexing perspective using MeSH as a controlled vocabulary. Thus using statistical language translation models may bridge the gap between the two types of 'languages' (authors vs indexers). We collected 13 999 pairs of (MHs, Title) and (MHs, Abstract), respectively, where MHs is the list of main headings assigned to an article. Then we used the expectation maximization algorithm to estimate the translation probability $\Pr(t|s)$, where t is a single word in the main headings (as a target language) and s a single word in the article title and abstract (as a source language). This translation probability was also used in computing query-likelihood features (see below). Finally, the following formula gives the probability of translating a piece of text into a MeSH term using the estimated $\Pr(t|s)$:

$$\Pr(MH|Text) = \frac{1}{n^m} \prod_{t_i \in MH;\, i=1}^{m} \sum_{s_j \in Text;\, j=1}^{n} \Pr(t_i|s_j) \quad (6)$$

where t_i and s_i are single words in a MeSH term and text, respectively. Text represents the title or abstract, respectively, so that we have two separate translation probabilities. Note that a MeSH term may contain multiple words.

Query-likelihood features

This class of features computes likelihood scores between a MeSH main heading and the title and abstract of an article (D in the formulas below) when using the MeSH term as a query (Q in the formulas). The advantage of using such

query-likelihood scores is that they give a probability of whether a MeSH term should be assigned to the article, instead of only binary judgment. In most cases, there is only indirect evidence for mapping a main heading to an article. We used two genres of query models: translation-based query models[20] and the very classic Okapi model,[21] as follows:

$$\Pr(Q|D) = \prod_{q\in Q}\left(\sum_{w\in D}\Pr(q|w)\left((1-\lambda)\frac{tf(w,D)}{|D|} + \lambda * P_c(w)\right)\right) \quad (7)$$

$$\Pr(Q|D) = \prod_{q\in Q}\left((1-\beta)\frac{tf(q,D)}{|D|} + \beta\sum_{w\in D}\Pr(q|w)\frac{tf(w,D)}{|D|}\right) \quad (8)$$

$$okapi(Q,D) = \sum_{q\in Q}\frac{tf(q,D)\log((N-df(q)+0.5)/(df(q)+0.5))}{0.5+1.5*(|D|/avg(|D|))+tf(q,D)} \quad (9)$$

where $\Pr(q|w)$ is the probability of translating word w in the source text into the target language (main headings), estimated with the IBM model; $tf(w,D)$ is the count of w occurring in document D; $|D|$ is the total counts of single words in document D; $P_c(w)$ is the probability of word w occurring in a background corpus; this is obtained from a unigram language model that was estimated on the 13 999 articles; $\mathrm{df}(q)$ is the number of documents containing word q; $\mathrm{avg}(|D|)$ is the average length of documents in the training corpus; and N is the total number of documents (13 999).

Computing $\mathrm{df}(q)$ and $\mathrm{avg}(|D|)$ requires a particular corpus. We used the same 13 999 documents as those used in the translation model. The parameters, λ and β, in formula (7) and formula (8), respectively, were empirically set to be 0.2.

Synonym features

The MeSH thesaurus provides synonyms for each main heading. Such synonyms are known as entry terms. We designed two binary features (the value is either 0 or 1): one judges whether one of the entry terms can be exactly matched to the document text (title and abstract); and the other judges whether there exists an entry term whose unigram words have all been observed in the document text.

Evaluation metrics

We use precision, recall, F score, and mean average precision (MAP) to evaluate the ranking performance. Given a ranking list $H_1^N = h_1h_2\ldots h_N$ with top N items, the four metrics are defined as follows:

$$\text{Precision} = \sum_D c(N,D,H_1^N)/\sum_D N$$

$$\text{Recall} = \sum_D c(N,D,H_1^N)/\sum_D AN(D)$$

$$\text{F-score} = \frac{2*precision*recall}{precision+recall}$$

$$\mathrm{AP(D)} = \frac{1}{AN(D)}\sum_r I(h_r)*\frac{c(r,D,H_1^r)}{r}$$

$$\mathrm{MAP}(\Omega) = \frac{1}{|\Omega|}\sum_{D\in\Omega} AP(D)$$

where $c(r,D,H_1^r)$ is the number of correct main headings among the top r ranked main headings; $AN(D)$ is the total number of gold-standard main headings assigned to document D; $I(h_r)$ is an indicator function, whose value is 1 if the r-th main heading h_r was assigned to the document and 0 otherwise; and Ω is the document collection of the test dataset. AP(D) is the average precision for document D, which computes all main headings in the list (not limited to top N items only). MAP(Ω) is the MAP over all test documents in collection Ω. AP measures the quality of a ranking list: perfect ranking corresponds to an AP of 1.0. N is set to be 25 in this paper.

RESULTS

Datasets

To train the ranking algorithm, we randomly selected a set of 200 MEDLINE documents where their corresponding MeSH terms were recently assigned (from 2002 to 2009), named Small200. We used two datasets of different sizes for testing. The first test dataset is named NLM2007 and was obtained from the NLM indexing initiative.[14] We selected NLM2007 because it has also been used in benchmarking the same task by other methods in recent studies.[9 14] The second test dataset contains 1000 randomly selected MEDLINE documents, and we named it L1000. To gain a better understanding of how our approach performs on a more general dataset, we chose to use L1000 because it consists of a larger number of citations. In addition, there is a longer time span for citations in L1000 with respect to when the MeSH main headings were added to the corresponding citations (ie, spanning over 48 years from 1961 to 2009). Title and abstract are available for every document in these datasets. Full text was not considered in this study because information in full text was found to be of limited use for the automatic production of MeSH indexing recommendations.[22] In addition, free access to full text is still limited in scope. The statistics of the three datasets are listed in table 1. We can see that the average number of main headings is relatively stable across the three datasets.

As described in the 'Features' section, in addition to the above datasets, we collected 13 999 documents from the MEDLINE database to train the translation model ($\Pr(q|w)$ in formulas 7–9) and the background language model ($P_c(w)$ in formula 7). This dataset was also used to compute the average length of documents ($\mathrm{avg}(|D|)$ in formula 9) and the document frequency for each word ($\mathrm{df}(q)$ in formula 9). These probabilities and values were computed before we trained the learning algorithm. These 13 999 documents have no overlap with any of the three datasets.

Comparison to other methods

We trained the ranking algorithm on Small200 and tested it on NLM2007. Since NLM2007 has only 200 documents, we further

Table 1 Detailed information about the three datasets used in this study

Dataset	Number of citations	Total number of main headings	Average number of main headings	When main headings were curated	Data availability
Small200	200	2736	13.7	2002–2009	Freely available
NLM2007	200	2737	13.7	1997–2001	–
L1000	1000	12145	12.1	1961–2009	–

Table 2 Precision, recall, F-score, and MAP for different methods

Method	Precision	Recall	F score	MAP
MTI system	0.318	0.574	0.409	0.450
Reflective random indexing†	0.372	0.575	0.451	N/A
Neighborhood frequency	0.369	0.674	0.476	0.598*
Neighborhood familiarity	0.376	0.677	0.483	0.604*
Learning-to-rank algorithm	0.390	0.712	0.504	0.626

The comparison was performed on the NLM2007 dataset. Statistical significance tests were performance on mean average precision to compare the two baseline ranking strategies with our learning-to-rank algorithm.
*$p<0.001$, indicating that the performance of both baseline strategies was significantly lower than the learning-to-rank algorithm.
†We directly used the best results from the paper of Vasuki and Cohen (2009).[9]
MAP, mean average precision; MTI, Medical Text Indexer.

tested the learned model on L1000 (1000 documents). The learning rate η and the number of iterations in the model were empirically set to be 0.01 and 100, respectively. All the feature values were normalized to (0,1) using the maximum values of each feature. For comparison with prior studies, the number of neighbors was set to be 20. The optimal number of neighbor documents will be further discussed in the next section.

We compared our approach to four methods, as listed in table 2. The first system we compared with is NLM's MTI system.[14] The second system used reflective random indexing[9] to find similar documents. The third and fourth ranking strategies are based on neighborhood features: neighborhood frequency as defined by formula 4, and neighborhood similarity as defined by formula 5. The reason we used the two neighborhood ranking strategies for comparison is that we considered them as strong baseline methods.

As shown in table 2, the last three ranking strategies show substantial improvements over MTI and reflective random indexing, while MTI and reflective random indexing have comparable performance to each other. Moreover, the MAP of our method (0.626) represents a 39.1% improvement over that of MTI (0.450). Note that the MTI's results in table 2 are slightly different from those on the web page (overall precision 0.335, overall recall 0.559)[iii]. This is because we reprocessed the documents in the set using MeSH 2010 (vs MeSH 2007 on the web page).

To demonstrate whether the learning algorithm has significant improvements over the neighborhood ranking criteria, we conducted several significance tests: binomial test,[23 24] the paired t test,[25] and Wilcoxon signed rank test.[26] All the significance tests demonstrate that the learning algorithm significantly outperforms the two neighborhood ranking strategies, with a p value of less than 0.001. These tests were performed with respect to MAP.

To further assess the learning algorithm on a larger and more general dataset, we evaluated the approach on L1000. We trained the model on the same Small200 set (200 documents) and tested it on L1000 (1000 documents). We also obtained MTI's results on L1000. Comparative results, as shown in table 3, demonstrate significant differences ($p<0.001$ with all the three statistical hypothesis tests) between the learning algorithm and the neighborhood ranking strategies. Again, both baseline ranking strategies and our learning-to-rank algorithm achieved substantially better performance over the MTI system. And notably, the relative difference between our approach and MTI is steady across the two datasets.

[iii] http://ii.nlm.nih.gov/Eval_Analysis/Eval_2007/summary.shtml.

Table 3 Precision, recall, F score, and MAP for different methods

Method	Precision	Recall	F score	MAP
MTI system	0.302	0.583	0.398	0.462
Neighborhood frequency	0.329	0.679	0.443	0.584*
Neighborhood similarity	0.333	0.687	0.449	0.591*
Learning-to-rank algorithm	0.347	0.714	0.467	0.615

The comparison was performed on the L1000 dataset. Statistical significance tests were performance on mean average precision to compare the two baseline ranking strategies with our learning-to-rank algorithm.
*$p<0.001$, indicating that performance of both baseline strategies was significant lower than the learning-to-rank algorithm.
MAP, mean average precision; MTI, Medical Text Indexer.

Choosing the number of *k*-nearest neighbors

We demonstrate here how many neighbor documents are optimal for this task. In principle, the more neighbors were chosen, the more gold-standard MeSH terms of a target document would be found available in its neighbors' annotations, resulting in a higher recall upper bound for our method. However, with more neighbors to be included, there is a tradeoff. That is, the number of to-be-ranked MeSH term candidates will increase significantly as we consider more neighbors. The statistics shown in table 4 demonstrate this. We see from both datasets that with 20 neighbors a fairly high upper-bound recall can be observed (about 85% of gold-standard annotations were available in all the main heading candidates), and the average number of main headings to be ranked is about 100.

To investigate the effect of the number of neighbors on the performance, we experimented with different numbers of neighbor documents. We trained a model on Small200 and tested it on NLM2007. The performance curves are presented in figure 3. As illustrated, the performance becomes relatively steady when 20 or more neighbors are used. Although the best performance (precision=0.392, recall=0.717) is observed when 40 neighbors were used (with over 160 main headings to be ranked on average), the performance is not significantly different to that with 20 neighbors (precision=0.390, recall=0.712).

Feature study

To investigate the impact of different features, we performed a feature ablation study. The features used in this study were divided into five groups. For each round of this study, we removed one group of features from the entire feature set, trained the model on Small200, and tested the performance on NLM2007.

The experimental results with different features are shown in table 5. When the neighborhood features were removed (the third row), the performance dropped remarkably, indicating that the two features are critical in maintaining high performance. Hence, we experimented with only the two neighborhood features and present the results in the last row. Statistical significance tests show that there is a significant difference in performance ($p<0.001$ with all the three statistical hypothesis tests) using all features versus using only neighborhood features. This result shows that although removing other groups of features did not result in a significant decrease in performance, the best performance was achieved only when all features were combined. In other words, combining all other non-dominant features indeed contributes to significant improvements.

It must be mentioned that these features are not independent. For example, the four query likelihood features correlate with the translation probability features (see formulas 6 and 7) and the unigram/bigram overlap features. Thus the study presented

Table 4 The upper-bound recall and average number of main headings with different number of neighbor documents

Dataset	Measure	Number of neighbor documents							
		5	10	15	20	25	30	35	40
NLM 2007	Upper-bound recall	0.704	0.793	0.832	**0.856**	0.871	0.882	0.891	0.898
	Number of main heading candidates	38.8	64.1	83.6	**102.2**	119.7	136.4	151.7	166.4
L1000	Upper-bound recall	0.702	0.786	0.825	**0.853**	0.870	0.882	0.891	0.899
	Number of main heading candidates	37.3	60.9	81.5	**99.8**	117.2	133.5	148.8	163.6

Both NLM2007 and L1000 datasets were used in the experiments.

here (by simply removing one group of these features) ignores the correlation among those features. Further investigation on feature correlation might be considered in future studies.

DISCUSSION

Comparison with other systems

MTI[14] relies on two parallel paths to rank and recommend MeSH terms: (1) MetaMap indexing which maps phrases in the text to UMLS (Unified Medical Language System) concepts and restricts those concepts to MeSH terms using synonyms, associated terms, and inter-concept relationships[27]; and (2) PubMed Related Articles which uses related documents to rank MeSH terms.[18] Sophisticated filtering strategies are applied to combine the two paths. The reflective random indexing system[9] retrieves neighbor documents using a method similar to the PubMed Related Citations path employed by MTI. Main heading candidates from neighbor documents are then scored by summing document similarity scores where low scored candidates are filtered.

Different from MTI and reflective random indexing, our method is a supervised machine learning algorithm which combines various statistical features together. Although both MTI and reflective random indexing have a similar *k*-nearest neighbor component, they are not as competitive as the neighborhood ranking strategies. We conjecture that this may be partially due to the filtering strategies they employed, or possibly because of the lack of a learning process. Instead, we ranked all the main headings in the initial lists with the learning algorithm. We did not filter any candidates in neighbor documents.

Speaking of using neighbor documents for MeSH term recommendation, Delbecque and Zweigenbaum[28] recently investigated computing neighbor documents based on authors' other prior publications and the referenced citations. The upper-bound recall of their method seems to be comparable to ours.

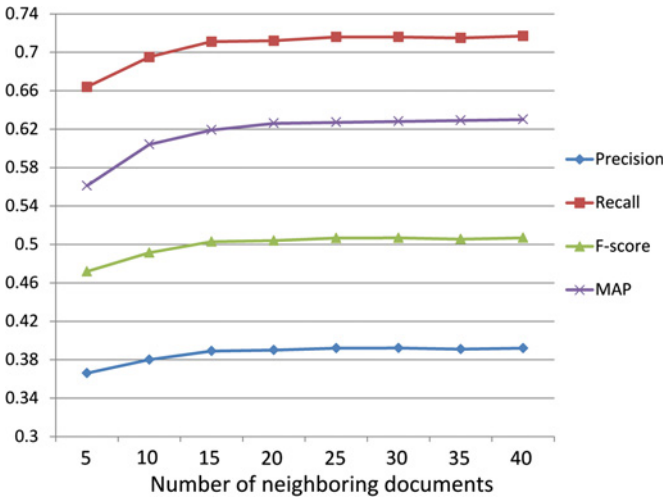

Figure 3 The ranking performance (y-axis) varies with different number of neighbor documents (x-axis). MAP, mean average precision.

However, their results are not directly comparable to ours (or others such as MTI) because their assessment was not carried out on the commonly used NLM2007 data set.

Performance ceilings and limitations

All methods that rely on neighbor documents have performance ceilings. The average number of annotations per document in our datasets is about 13. Therefore, when recommending 25 main headings for a given document, the upper-bound precision is about 0.520. As shown in table 4, for the NLM2007 dataset, the upper bound precision and recall are 0.547 and 0.856, respectively, with 20 neighbor documents. Our best performance yielded a precision and recall of 0.390 and 0.712, respectively. For the L1000 dataset, the upper bound precision and recall are 0.485 and 0.853, respectively, while our best performance gives 0.347 and 0.714 accordingly. Although precision is lower than recall, 'this corresponds to the observation that indexers will tolerate some inappropriate terms as long as many useful are presented to them.'[22] There is still space to improve, but not surprisingly, improving the performance with fewer recommended MeSH terms will be even more challenging.

Our approach shares the same limitations with MTI's Related Citations component[14] and the reflective random indexing approach,[9] as all these approaches rely on neighbor documents' annotations. As a result, this genre of methods is limited as regards recommending MeSH terms that have been recently added to the MeSH vocabulary as such terms may not yet have been assigned to any document. MTI compensates for the shortcomings of the *k*-nearest neighbor approach by combining it with a natural language processing approach, which specifically boosts the suggestion of terms newly added in MeSH. For comparability with both MTI and reflective random indexing results, the top 25 recommended terms were considered in the evaluations. As shown in table 5, this choice results in an upper bound in recall.

Implications of this research

Impact on indexing practice at NLM

The objective of this work is to investigate automatic indexing methods to enhance current indexing practices. Since 2002,

Table 5 Feature ablation study

Feature set	Precision	Recall	F score	MAP
All features	0.390	0.712	0.504	0.626
Neighborhood features	0.315*	0.575*	0.407*	0.435*
Unigram/bigram features	0.389	0.711	0.503	0.626
Translation probability features	0.389	0.711	0.503	0.626
Query likelihood features	0.385	0.704	0.498	0.626
Synonym features	0.385	0.703	0.497	0.618
Only neighborhood features	0.370*	0.677*	0.478*	0.602*

In rows starting with a minus sign (−), we trained and tested the learning algorithm using all but the given set of features. Those marked with asterisks are significant worse than the accordant measures using all features ($p<0.001$ with all the three statistical significance tests).

MeSH main heading recommendations automatically produced by MTI have been made available to NLM indexers in the form of a pick list. This recommendation list mainly serves two purposes. First, it is used as an educational tool. When used in training, the list is shown to teach novice indexers how to select indexing terms appropriately. Second, it is used as an indexing assistant. The list allows indexers to select desired MeSH terms with only one click per term without having to type or look up individual terms. Hence, improving the quality of the MTI recommendation list has been an on-going effort at NLM, involving close collaboration between researchers and indexers. Improvements to the recommendation list shown to benefit the indexing practice in research studies are deployed in production. For instance, subheading attachment recommendations were deployed in 2008.[15] As shown by comparison results, our approach holds great potential to be used for generating the recommendation list in practice and to subsequently improve the indexing quality and productivity. Another practical implication would be to adapt our approach for use in the current MTI system with the aim of improving it. In particular, the proposed supervised learning-to-rank algorithm might be a better approach for MTI than its current filtering strategies in integrating results from both MetaMap and PubMed Related Citations.

Other applications of the learning-to-rank method

Generally speaking, ranking-based approaches have advantages over classification-based approaches because in certain tasks[29 30] it is difficult to choose negative instances when only positive instances are labeled. One example of such a task is to identify a database record for a given gene/protein name. As detailed in Lu *et al*,[31] gene/protein names are notoriously ambiguous, and thus one name usually maps to multiple database records. Traditional classification-based methods attempt to predict each possible matched record to be either relevant or irrelevant to the target gene/protein mention. By contrast, the learning-to-rank method approaches the same issue as a ranking problem: to sort all matched records in a list and rank the relevant record to be at the top position on the list. As demonstrated by Huang *et al*,[29] the learning-to-rank approach shows favorable performance over other classification-based methods.

CONCLUSION AND FUTURE WORK

We presented a ranking-based method to recommend MeSH terms to annotate biomedical articles. The method retrieved *k*-nearest neighbor documents, and then it obtained an initial list of MeSH term candidates from the neighbors. The candidates were ranked and the document was annotated with top recommended MeSH terms. The method was assessed with large-scale experiments; significant improvements over the state of the art were observed on two representative datasets. The impact of different features was also investigated.

Future work will focus on improving the precision while maintaining the recall, with fewer recommended annotations. In this paper we recommend 25 MeSH main headings for each article. However, the average number of main headings is actually about 13. Thus, fewer recommendations would be more attractive for real use. Another interesting perspective would be to assess the practical potential of our approach. Specifically, the system could be evaluated on a corpus of recent citations in order to assess the impact on the recommendation of recent MeSH terms. Finally, although we have received some preliminary positive feedback when presenting the recommendation list obtained from our method compared to MTI, more experiments are needed to warrant the use of our approach in the practical setting for MeSH indexing. Specifically, feedback could be sought from indexers through a controlled experiment where one group of participating indexers would be shown indexing recommendations produced by the current version of MTI, while another group would be shown indexing recommendations produced by our method. In this setting, we could assess time spent on indexing articles, the number of recommendations actually used by the indexers as well as qualitative comments from the indexers. A subsequent analysis of these measures would then suggest which method offers better productivity, accuracy, and indexer satisfaction.

Acknowledgments We would like to thank Dr W John Wilbur and Dr Won Kim for providing the L1000 dataset and for retrieving neighbor documents, and thank Dr Alan Aronson and James G Mork for valuable discussions. We would also like to thank Professor Xiaoyan Zhu for her generous support when the first author was visiting NCBI.

Funding This work was supported by the Intramural Research Program of the National Institutes of Health, National Library of Medicine. The first author was also supported by the Chinese Natural Science Foundation under grant no. 60803075.

Competing interests None.

Provenance and peer review Not commissioned; externally peer reviewed

REFERENCES

1. **Baumgartner WA,** Cohen KB, Fox LM, *et al*. Manual curation is not sufficient for annotation of genomic databases. *Bioinformatics* 2007;**23**:i41—8.
2. **Kim W,** Aronson AR, Wilbur WJ. Automatic MeSH term assignment and quality assessment. *Proc AMIA Symp* 2001:319—23.
3. **Trieschnigg D,** Pezik P, Lee V, *et al*. MeSH Up: effective MeSH text classification for improved document retrieval. *Bioinformatics* 2009;**25**:1412—18.
4. **Zhu S,** Zeng J, Mamitsuka H. Enhancing MEDLINE document clustering by incorporating MeSH semantic similarity. *Bioinformatics* 2009;**25**:1944—51.
5. **Djebbari A,** Karamycheva S, Howe E, *et al*. MeSHer: identifying biological concepts in microarray assays based on PubMed references and MeSH terms. *Bioinformatics* 2005;**21**:3324—6.
6. **Funk ME,** Reid CA. Indexing consistency in MEDLINE. *Bull Med Libr Assoc* 1983;**71**:176—83.
7. **Aronson AR,** Bodenreider O, Chang HF, *et al*. The NLM Indexing Initiative. *Proc AMIA Symp* 2000:17—21.
8. **Yang Y,** Chute CG. An application of expert network to clinical classification and MEDLINE indexing. *The Eighteenth Annual Symposium on Computer Applications in Medical Care*. Bethesda, MD: American Medical Informatics Association 1994:157—61.
9. **Vasuki V,** Cohen T. Reflective random indexing for semiautomatic indexing of the biomedical literature. *AMIA Annu Symp Proc* 2009.
10. **Yang Y,** Chute CG. A linear least square fit mapping method for information retrieval from natural language texts. *Proceedings of the 14th International Conference on Computational Linguistics*. Stroudsburg, PA: Association for Computational Linguistics 1992;**2**:447—53.
11. **Sohn S,** Kim W, Comeau DC, *et al*. Optimal training sets for bayesian prediction of MeSH® assignment. *J Am Med Inform Assoc* 2008;**15**:546—53.
12. **Ruch P.** Automatic assignment of biomedical categories: toward a generic approach. *Bioinformatics* 2006;**22**:658—64.
13. **Aronson AR.** Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *Proc AMIA Symp* 2001:17—21.
14. **Aronson AR,** Mork JG, Gay CW, *et al*. The NLM Indexing Initiative's Medical Text Indexer. *Stud Health Technol Inform* 2004;**107**:268—72.
15. **Névéol A,** Shooshan SE, Humphrey SM, *et al*. A recent advance in the automatic indexing of the biomedical literature. *J Biomed Inform* 2009;**42**:814—23.
16. **Liu TY.** *Learning to Rank for Information Retrieval*. Foundations and Trends® in Information Retrieval, 2009;**3**:225—331. http://dx.doi.org/10.1561/1500000016.
17. **Cao Z,** Qin T, Liu TY, *et al*. Learning to rank: from pairwise approach to listwise approach. *International Conference on Machine Learning*. New York, NY: ACM, 2007:129—36.
18. **Lin JJ,** Wilbur WJ. PubMed related articles: a probabilistic topic-based model for content similarity. *BMC Bioinformatics* 2007;**8**:423.
19. **Brown PF,** Pietra VJD, Pietra SAD, *et al*. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* 1993;**19**:263—311.
20. **Berger A,** Lafferty J. Information retrieval as statistical translation. *The 1999 ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY: ACM, 1999:222—9.
21. **Robertson SE,** Walker S, Jones S, *et al*. Okapi at TREC-3. *Proceedings of the Third Text REtrieval Conference 1994*. Gaithersburg, MD: National Institute of Standards and Technology, 1994.

22. **Gay CW,** Kayaalp M, Aronson AR. Semi-automatic indexing of full text biomedical articles. *AMIA Annu Symp Proc* 2005:271—5.
23. **Salzberg SL.** *On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach*. Data Mining and Knowledge Discovery. New York, NY: Springer, 1997;**1**:317—27.
24. **Yang Y,** Liu X. A re-examination of text categorization methods. *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY: ACM, 1999:42—9.
25. **Goulden CH.** *Methods of Statistical Analysis*. 2nd edn. New York: Wiley, 1956:50—5.
26. **Wilcoxon F.** Individual comparisons by ranking methods. *Biometrics* 1945;**1**:80—3.
27. **Fung KW,** Bodenreider O. Utilizing the UMLS for semantic mapping between terminologies. *AMIA Annu Symp Proc* 2005:266—70.
28. **Delbecque T,** Zweigenbaum P. Using Co-Authoring and Cross-Referencing Information for MEDLINE Indexing. *AMIA Annu Symp Proc* 2010:147—51.
29. **Zheng ZC,** Li FT, Huang ML, *et al*. *Learning to Link Entities with Knowledge Base*. Proceedings of The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Stroudsburg, PA: Association for Computational Linguistics, 2010.
30. **Huang ML,** Liu JC, Zhu X. GeneTUKit: a software for document-level gene normalization. *Bioinformatics* 2011;**27**:1023—3.
31. **Lu Z,** Wilbur WJ. Overview of BioCreative III Gene Normalization. *Proceedings of the BioCreative III Workshop*. Newark, DE: University of Delaware Press, 2010.