JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 6, No. 2, June 2012, pp. 151-160

A One-Size-Fits-All Indexing Method Does Not Exist: Automatic Selection Based on Meta-Learning

Antonio Jimeno-Yepes*, James G. Mork, Dina Demner-Fushman, and Alan R. Aronson
National Library of Medicine, Bethesda, MD, USA
antonio.jimeno@gmail.com, mork@nlm.nih.gov, ddemner@mail.nih.gov, alan@nlm.nih.gov

Abstract
We present a methodology that automatically selects indexing algorithms for each heading in Medical Subject Headings (MeSH), National Library of Medicine's vocabulary for indexing MEDLINE. While manually comparing indexing methods is manageable with a limited number of MeSH headings, a large number of them make automation of this selection desirable. Results show that this process can be automated, based on previously indexed MEDLINE citations. We find that AdaBoostM1 is better suited to index a group of MeSH hedings named Check Tags, and helps improve the micro F-measure from 0.5385 to 0.7157, and the macro F-measure from 0.4123 to 0.5387 (both $p < 0.01$).

Category: Convergence computing

Keywords: MeSH; MEDLINE; Text categorization; Automatic indexing; Meta-learning

1. INTRODUCTION

MEDLINE® citations are indexed using the Medical Subject Headings (MeSH) ® controlled vocabulary. This indexing is performed by a relatively small group of highly qualified indexing contractors and staff at the US National Library of Medicine (NLM). Their task is becoming more difficult, due to the yearly increase of MEDLINE, currently increasing by around 700 k articles per year [1].

The Medical Text Indexer (MTI) [2-4] is a support tool for assisting indexers as they add MeSH indexing to MEDLINE citations. MTI has two main components: MetaMap [5], and the PubMed® Related Citations (PRC) algorithm [6]. MetaMap analyzes citations, and annotates them with Unified Medical Language System (UMLS) ® concepts. Then, the mapping from UMLS to MeSH follows the Restrict-to-MeSH [7] approach, which is based primarily on the semantic relationships among UMLS concepts (MetaMap Indexing, MMI). The PRC algorithm is a modified k-nearest neighbor (k-NN) algorithm, which relies on document similarity to assign MeSH headings. The output of MMI and PRC are combined by linear combination of their indexing confidence. This method attempts to increase the recall of MTI, by proposing indexing candidates for MeSH headings that are not explicitly present in the title and abstract of the citation, but that are used in similar contexts. Finally, a post-processing step arranges the list of MeSH headings, and tailors the output to reflect NLM indexing policy.

We are studying the use of machine learning, to improve the MeSH heading assignment to MEDLINE citations performed by MTI. While comparing and selecting indexing methods is manageable with a limited number of MeSH headings, a large number of them make automation of this selection desirable.

Open Access http://dx.doi.org/10.5626/JCSE.2012.6.2.151 http://jcse.kiise.org

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Received February 18 2012, **Accepted** May 18 2012
*Corresponding Author

Copyright © 2012. The Korean Institute of Information Scientists and Engineers pISSN: 1976-4677 eISSN: 2093-8020

In this work, we present a methodology to automatically select an indexing algorithm for each MeSH heading. Experiments are performed on the whole set of MeSH headings, and on a set of MeSH headings known as Check Tags [8]. Check Tags are a special class of MeSH headings routinely considered for every article, which cover species, sex and human age groups, historical periods, and pregnancy. We show that this process can be automated, based on previously indexed MEDLINE citations.

II. RELATED WORK

We find that most of the existing MeSH indexing methods fit into either pattern matching methods, which are based on a reference terminology (like UMLS or MeSH), or machine learning approaches, which learn a model from examples of previously indexed citations.

In the machine learning community, the task of MeSH indexing has been considered as a text categorization problem. Publication of the OHSUMED collection [9], containing all MEDLINE citations in 270 medical journals over a five-year period (1987-1991) including MeSH indexing, provided a large body of data that enabled us to view MeSH heading assignment as a classification problem. The scope of the collection determined the subset of MeSH that can be explored. For example, Lewis et al. [10] and Ruiz and Srinivasan [11] used 49 categories related to heart diseases with at least 75 training documents, and Yetisgen-Yildiz and Pratt [12] expanded the number of headings to 634 disease categories. Poulter et al. [13] provide an overview of these and other studies of classification methods, applied to MEDLINE and MeSH subsets.

Among the pattern matching methods, we find MetaMap, as mentioned above, and an information retrieval approach by Ruch [14]. Pattern matching considers only the inner structure of the terms, but not the terms with which they co-occur. This means that if a document is related to a MeSH heading, but the heading does not appear in the reference source, it will not be suggested.

Currently, MeSH contains 26,142 main headings and over 172,000 entry terms, to assist the indexers in determining the appropriate main headings to assign to a MEDLINE citation. Small-scale studies with machine learning approaches already exist [12, 15]. But the presence of a large number of categories has forced machine learning approaches to be combined with information retrieval methods designed to reduce the search space. For instance, PRC and a k-NN approach by Trieschnigg et al. [16] look for similar citations in MEDLINE, and predict MeSH headings by a voting mechanism on the top-scoring citations. Experience with MTI shows that k-NN methods produce high recall, but low precision indexing. Other machine learning algorithms have been evaluated that rely on a more complex representation of the citations, e.g., learning based on Inductive Logic Programming [17]. In previous work [18, 19], we showed that MeSH headings have different behavior, depending on the indexing algorithm used.

The selection of the best indexing method is a challenging task, due to the number of available categories and methods. In this paper, we present a methodology that automates the selection of indexing algorithms based on meta-learning.

III. META-LEARNING

In machine learning, meta-learning [20, 21] applies automatic learning to machine learning experiments. In our work, the experimental data are indexing algorithm results, which are used to select the most appropriate algorithm.

Indexing methods have different performance, depending on the MeSH heading. To illustrate why this happens, we can place the citations in a two dimensional space, in which a + sign is a positive example, and a - sign is a negative example.

Fig. 1 shows two sets of instances represented in this vector space. In the left image, the positive and negative citations can be split into two sets, based on a separating hyperplane, supporting the use of a support vector machine (SVM) approach with linear kernel. In the right image, it is not possible to identify a hyperplane, so another kind of learning algorithm is required, e.g., k-NN or SVM with non-linear kernel.

Without previous experimentation, it is difficult to know how the positive and negative instances are distributed in the citation space. Experimentation with several learning algorithms allows for a better understanding of the problem being addressed.

We propose to collect indexing results based on machine learning and MTI experiments, and use them as input data for the meta-learning experiments. The representation of the citation will play a role in the model optimization as well. For instance, n-grams afford an appropriate representation, when word collocation is relevant for indexing.

With small sets, manual selection and optimization of the parameters can be managed efficiently. But when

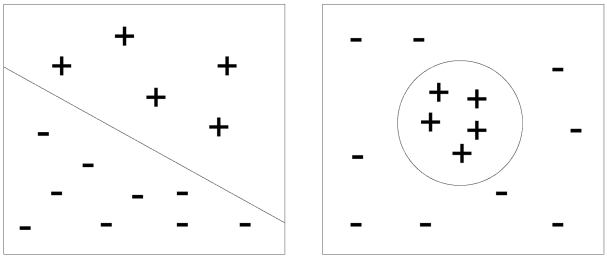

Fig. 1. Instance sets.

Table 1. F-measure for indexing methods on the Humans MeSH heading

Method	Average F-measure
Medical Text Indexer	0.72
Naïve Bayes	0.85
Support Vector Machine	0.88
AdaBoostM1	0.92

there is a large number of categories, meta-learning can play an important role. The optimization parameters are one level above traditional machine learning, since the objective is not to improve an existing learning algorithm, but to select the best algorithm, and its configuration for a given problem. In Table 1, we compare the performance of MTI and several standard machine learning algorithms for the Humans MeSH heading.

In this case, AdaBoostM1 outperforms all the other methods, and would be the method of choice for indexing citations with Humans MeSH heading.

IV. METHODS

In this section, we present how the framework is trained, and how it is used to index citations. Then, the base methods used for MeSH indexing are shown. The methods include MTI, a dictionary lookup approach, and several machine learning algorithms.

Experiments have been performed on a set of 300k citations from the 2011 MEDLINE Baseline and the 2011 MeSH vocabulary. The citations are sorted by date, so the first 200k citations are used for training, and the remaining 100k for testing.

A. Training

The outcome of the training is a mapping between a MeSH heading, and an indexing method to be used for that MeSH heading. The performance of each algorithm on each MeSH heading was collected and compared. In this work, we have used the F-measure as our indexing performance measurement, which is standard in text categorization, even though other measurements, like accuracy, could be considered as well.

Since machine learning algorithms require training, we have split the 200k training data into training and validation subsets. To increase confidence, several training and validation splits were evaluated, and the results averaged. We have run 5 times 2-fold cross validation. Statistical significance of the results was computed using a randomization version of the two sample *t*-test [22].

In each split, the steps to estimate the performance of each algorithm A for each MeSH heading M were:

- Step 1: If required, train algorithm A using the training subset. The positive examples are the citations indexed with the M MeSH heading, the rest are considered as negative examples. Note that MTI and dictionary lookup methods do not require training.
- Step 2: Use algorithm A to index the citations in the validation subset with M MeSH heading.
- Step 3: Compute the performance of algorithm A, i.e., the F-measure, comparing the indexing produced in step 2 to the original indexing for the validation set.

This process was repeated for each MeSH heading. The best method for each heading was selected and stored in a mapping table. For machine learning methods, the trained model for the best method was also stored in the table.

B. Indexing

During indexing time, the mapping table prepared during the training process is used to index citations. Given a new citation, for each MeSH heading M the corresponding method from the mapping table is selected, and used to determine if the citation should be indexed with M.

Several implementations could be considered to speed up the indexing. Batch indexing of the citations, and a post-processing of the outcome, could be considered to index the citations with predictions by MTI, filtering out the predictions for which MTI was not the preferred method. On the other hand, trained machine learning models could be applied in parallel, to determine the indexing. This would allow processing of a large number of citations with one method, instead of processing a single citation by all the methods. Again, the results would be post-processed, but this time to merge the results of each indexing method.

C. MeSH Indexing Methods

Most of the indexing algorithms we study here require a training phase, MTI and dictionary lookup being exceptions. MTI has already been described in the introduction, so we focus on the other methods used in our experiments.

Since the main focus of the paper is the meta-learning framework, only machine learning algorithms that we could train using a large number of examples and a large number of categories (MeSH headings) have been selected. AdaBoostM1 has been used only in the Check Tag experiments. We are planning to include more learning algorithms, as they are integrated into our system.

D. Dictionary Lookup

This method looks for mentions of the MeSH heading in the citation text, as they appear in MeSH. If the mention of a MeSH heading is matched in the citation text, the citation is indexed with this MeSH heading. The preferred term and its entry terms are included in the dictio-

nary. MeSH is turned into a list of terms and IDs.

Our dictionary lookup implementation is based on the monq.JFA package [23]. In addition to matching the dictionary terms to text, morphological changes are applied to the lexical items; e.g., the case of the first letter is normalized, hyphens are changed to spaces, and plural termination is normalized. Furthermore, the longest matched span is selected. For instance, the span of text "...quality of breast cancer care..." matches cancer and breast cancer. In this case, the match breast cancer is selected.

In our work, dictionary lookup was used to index a citation based on the title and abstract text (MeSH TIAB DL), and to index only the title (MeSH TI DL), which might provide higher precision, at the cost of recall.

E. Naïve Bayes

A citation C is indexed with a MeSH heading I, if the probability of indexing the citation with the MeSH heading is higher than the probability of not indexing it (NI):

$$P(I|C) > P(NI|C) \tag{1}$$

Using Bayes:

$$\frac{P(I)P(C|I)}{P(C)} > \frac{P(NI)P(C|NI)}{P(C)} \tag{2}$$

We can remove $P(C)$, without affecting the inequality.

As presented in Equation 3, the probability of a citation being indexed with a given MeSH heading is the product of the probabilities of each term t in the citation. The probability that a citation will not be indexed with the MeSH heading is estimated in the same way.

$$P(C|I) = \prod_{t \in C} P(t|I) \tag{3}$$

The probability of a term given a MeSH heading is estimated as shown in Equation 4, where N is the total number of citations, $cf_{t,I}$ is the number of citations where term t appears and the citation is indexed with the MeSH heading. V is the set of all tokens.

$$P(t|I) = \frac{cf_{t,I}}{\sum_{t_V \in V} cf_{t_V,I}} \tag{4}$$

We use a smoothed model based on Jelinek-Mercer [24], due to term sparsity. In our experiments, we have used a value for λ of 0.8.

$$\hat{P}(t|I) = \lambda P(t|I) + (1-\lambda) P(t|G) \tag{5}$$

Finally, the prior $P(I)$ is presented in Equation 6, where cf_I is the number of citations that have been indexed with the MeSH heading I.

$$P(I) = \frac{cf_I}{N} \tag{6}$$

We have also implemented a variant of Naïve Bayes (NB) based on term frequency - inverse document frequency (TF-IDF) [25], which has been shown to improve the performance of a traditional NB for text categorization.

We represent occurrences of the terms in the citations as binary features, so the frequency of a term in a document is not considered. We use a unigram model, so the relations of the terms in the citation are also not considered.

F. Rocchio

Usually used in query expansion in ad-hoc retrieval, Rocchio has been used as well for text categorization. A vector is calculated for each MeSH heading, by adding the mentions of the term t in the citations where the MeSH heading I and the term occur together, as we can see in Equation 7.

$$\vec{q} = \left\{ \frac{cf_{t_V,I}}{N}, \ldots, \frac{cf_{t_V,I}}{N} \right\} \tag{7}$$

Given a citation, MeSH headings are ranked by cosine similarity. From this ranked list, we take the top n MeSH headings. In our experiments, we have considered the top 20.

G. AdaBoostM1

AdaBoostM1 [26] is an ensemble learning algorithm, which samples iteratively from the training data, according to the performance of a base learner. In each iteration, a new model is produced. The final decision is based on the weighted sum of the models produced in the iterative process. The weights are estimated based on the performance of each model on the training data. In this work, 10 iterations were performed.

In our experiments, we used C4.5 as the base learner, since it has produced good results in the past [18], with a smaller set of MeSH headings. Our decision tree is an implementation of the C4.5 algorithm [27] with pruning, and with the minimum number of elements in leaf nodes set to 5. In our implementation, we consider binary features and 1-versus-all classification as well. This setup allows for optimizations in the information gain calculation that allow training this algorithm efficiently. We trained the learner on the random training set splits, as well as with oversampling of the positive examples, trying to overcome skewness in the distribution of positive and negative examples. In oversampling, examples are added to the minority category. In our experiments, we selected examples randomly from the minority category, till both categories had the same number of examples.

H. Voting

Combinations of methods have proved to increase performance of individual methods [28, 29]. Given a citation, for a given MeSH heading, the predictions for each

of the indexing methods presented above were collected. Then, the votes were counted, and if the sum of the votes was over a given threshold, the MeSH heading was predicted by this method. We have performed experiments with different voting values, based on the methods presented above.

V. RESULTS AND DISCUSSION

We have performed two experiments. In the first one, we have considered all the MeSH headings and trained algorithms, which can handle a large number of categories and features. In the second one, we evaluated a reduced set of MeSH headings, named check tags.

In both experiments, MTI annotation was considered the baseline method. Features for the machine learning algorithms were represented as the presence of tokens from the title and abstract of the citation; the frequency of the tokens in the citation was not used. The tokens were lowercased, but not stemmed.

A. Results with All MeSH Headings

This experiment was done on all MeSH headings. The experiment used all but the AdaBoostM1 method, due to the time it takes to train it. For 2,712 of the 26k MeSH headings, a different method from MTI was selected: either a single method or a voting combination of them. Only methods significantly better than MTI were selected. This means that if the methods had a similar performance, MTI was preferred. In Table 2, we only show the set of MeSH headings grouped by learning method, where MTI was outperformed by methods as selected by meta-learning. MTI is the best algorithm for the MeSH headings not reflected in this table.

Voting 3, in which at least three methods agreed on predicting the MeSH heading, seems to perform better than the individual methods tested in this work. Voting 4 has a larger increase in precision compared to the decrease in recall, so the F1 is higher compared to MTI. On the other hand, if five of the evaluated methods need to agree, only a small number of MeSH headings are affected, and the performance is lower, compared to MTI.

Surprisingly, dictionary lookup (MeSH TIAB DL) performs reasonably well in some cases, compared to MTI. Machine learning methods perform better only on a small set of MeSH headings; one of the problems could be the small number of positive examples available for most of

Table 2. Results for Medical Text Indexer (MTI) and meta-learning for the 2,712 MeSH headings (MHs)

Method	MH count	P	TP	FP	Micro P	Micro R	Micro F	Macro P	Macro R	Macro F
Vote 3	1,037	103,510	59,084	52,642	**0.5288**	0.5708	**0.5490**	**0.5747**	0.5774	**0.5760**
MTI	1,037	103,510	63,356	86,037	0.4241	**0.6121**	0.5010	0.4819	**0.6383**	0.5492
MeSH TIAB DL	701	53,140	28,550	29,373	0.4929	**0.5373**	**0.5141**	**0.5455**	**0.6118**	**0.5767**
MTI	701	53,140	23,989	24,503	**0.4947**	0.4514	0.4721	0.5347	0.5722	0.5528
MeSH TI DL	530	16,360	8,148	5,792	**0.5845**	0.4980	**0.5378**	**0.7712**	0.4989	**0.6059**
MTI	530	16,360	10,835	18,641	0.3676	**0.6623**	0.4728	0.5066	**0.6887**	0.5838
Vote 4	176	11,103	5,577	6,222	**0.8922**	0.5023	**0.6427**	**0.4400**	0.6479	**0.5241**
MTI	176	11,103	7,458	20,353	0.3658	**0.6717**	0.4737	0.1393	**0.9225**	0.2420
Rocchio	175	122,579	60,353	191,842	**0.2393**	**0.4924**	**0.3221**	0.1815	**0.5060**	**0.2672**
MTI	175	122,579	9,643	9,355	0.5076	0.0787	0.1362	**0.4498**	0.1048	0.1700
NBTFIDF	88	16,096	6,382	29,067	0.1800	**0.3965**	**0.2476**	0.1496	**0.4285**	**0.2218**
MTI	88	16,096	2,204	4,113	**0.3489**	0.1369	0.1967	**0.3957**	0.1236	0.1884
Naïve Bayes	3	141,448	108,214	48,534	0.6904	**0.7650**	**0.7258**	0.6644	**0.7505**	**0.7048**
MTI	3	141,448	68,255	7,577	**0.9001**	0.4825	0.6283	**0.8797**	0.4149	0.5639
Vote 5	2	85	34	21	**0.6182**	0.4000	**0.4857**	**0.8019**	0.3216	0.4591
MTI	2	85	70	151	0.3167	**0.8235**	0.4575	0.4049	**0.6564**	**0.5009**
Overall	MH count	P	TP	FP	Micro P	Micro R	Micro F	Macro P	Macro R	Macro F
Meta-learning	2,712	464,321	276,342	363,493	0.4319	**0.5952**	**0.5005**	**0.5690**	0.5589	**0.5639**
MTI	2,712	464,321	185,810	170,730	**0.5211**	0.4002	0.4527	0.4927	**0.5850**	0.5349

TI: title, AB: abstract, DL: dictionary lookup, NBTFIDF: Naïve Bayes term frequency inverse document frequency, P: precision, TP: true positive, FP: false positive, R: recall.

the MeSH headings.

Learning methods seemed to be more effective for the fewer MeSH headings that are more common in the indexing (e.g., Humans, Male, Female). These headings have more training data, and a more balanced proportion between positives and negatives.

NB has good performance on the most frequent MeSH headings (Humans, Male, and Female), which belong to the set of Check Tags. The modified NB (NBTFIDF) has better performance for a larger number of MeSH headings, compared to plain Naïve Bayes. Finally, Rocchio performs better in a larger number of MeSH headings, compared to the other two NB algorithms.

We can see that only a limited number of MeSH headings were affected by using the proposed approach. We have analyzed the results, and found that the improvements apply mostly to MeSH headings which had a higher indexing frequency. The large imbalance and variability between the training and testing might justify the results obtained with lower frequency MeSH headings. Another factor is that, since MTI is the current system, it has been left as the default, if the differences with MTI were not statistically significant.

B. Results with Chect Tags

In this experiment, we have included AdaBoostM1 as a learning algorithm. In Table 3, we present the evalua-

Table 3. Results for Medical Text Indexer (MTI) and meta-learning for the Check Tags set

MH	DUI	Method	P	TP	FP	Precision	Recall	F1
Adolescent	D000293	AdaBoostM1 OverSampling	8,156	2,638	2,489	0.5145	**0.3234**	**0.3972**
		MTI		1,275	410	**0.7567**	0.1563	0.2591
Adult	D000328	AdaBoostM1 OverSampling	19,362	11,516	7,396	**0.6089**	0.5948	**0.6018**
		MTI		2,407	2,874	0.4558	0.1243	0.1953
Aged	D000368	AdaBoostM1 OverSampling	13,389	7,509	5,357	0.5836	**0.5608**	**0.5720**
		MTI		875	249	**0.7785**	0.0654	0.1206
Aged, 80 and over	D000369	ROCCHIO.output	5,205	2,802	10,370	**0.2127**	**0.5383**	**0.3049**
		MTI		15	89	0.1442	0.0029	0.0057
Animals	D000818	AdaBoostM1 OverSampling	24,218	18,111	3,405	0.8417	**0.7478**	**0.7920**
		MTI		17,582	2,712	**0.8664**	0.7260	0.7900
Bees	D001516	MTI	59	46	19	0.7077	0.7797	0.7419
Cats	D002415	vote.3	233	153	18	**0.8947**	0.6567	**0.7574**
		MTI		196	107	0.6469	**0.8412**	0.7313
Cattle	D002417	AdaBoostM1 OverSampling	1,114	791	269	**0.7462**	**0.7101**	**0.7277**
		MTI		772	271	0.7402	0.6930	0.7158
Cercopithecus aethiops	D002522	MTI	206	62	56	0.5254	0.3010	0.3827
Chick Embryo	D002642	AdaBoostM1 OverSampling	92	55	57	0.4911	**0.5978**	**0.5392**
		MTI		28	9	**0.7568**	0.3043	0.4341
Child	D002648	MTI	6,082	3,501	2,122	0.6226	0.5756	0.5982
Child, Preschool	D002675	AdaBoostM1 OverSampling	3,302	1,495	1,448	**0.5080**	**0.4528**	**0.4788**
		MTI		23	62	0.2706	0.0070	0.0136
Cricetinae	D006224	AdaBoostM1 OverSampling	321	158	62	**0.7182**	0.4922	**0.5841**
		MTI		171	157	0.5213	**0.5327**	0.5270
Dogs	D004285	AdaBoostM1	633	461	70	**0.8682**	0.7283	**0.7921**
		MTI		483	134	0.7828	**0.7630**	0.7728
Female	D005260	AdaBoostM1 OverSampling	35,501	25,824	6,718	0.7936	**0.7274**	**0.7590**
		MTI		11,335	1,812	**0.8622**	0.3193	0.4660
Guinea Pigs	D006168	MTI	132	103	11	0.9035	0.7803	0.8374
History, 15th Century	D049668	AdaBoostM1 OverSampling	42	9	437	**0.0202**	**0.2143**	**0.0369**
		MTI		0	0	0.0000	0.0000	0.0000
History, 16th Century	D049669	AdaBoostM1	72	2	10	**0.1667**	**0.0278**	**0.0476**
		MTI		0	0	0.0000	0.0000	0.0000

Table 3. Continued

MH	DUI	Method	P	TP	FP	Precision	Recall	F1
History, 17th Century	D049670	AdaBoostM1	94	6	21	**0.2222**	**0.0638**	**0.0992**
		MTI		0	0	0.0000	0.0000	0.0000
History, 18th Century	D049671	AdaBoostM1	145	12	23	**0.3429**	**0.0828**	**0.1333**
		MTI		0	0	0.0000	0.0000	0.0000
History, 19th Century	D049672	AdaBoostM1 OverSampling	397	128	497	**0.2048**	**0.3224**	**0.2505**
		MTI		0	0	0.0000	0.0000	0.0000
History, 20th Century	D049673	AdaBoostM1 OverSampling	928	375	1097	**0.2548**	**0.4041**	**0.3125**
		MTI		0	0	0.0000	0.0000	0.0000
History, 21st Century	D049674	AdaBoostM1 OverSampling	476	97	730	**0.1173**	**0.2038**	**0.1489**
		MTI		0	0	0.0000	0.0000	0.0000
History, Ancient	D049690	AdaBoostM1 OverSampling	103	35	112	**0.2381**	**0.3398**	**0.2780**
		MTI		0	0	0.0000	0.0000	0.0000
History, Medieval	D049691	AdaBoostM1 OverSampling	59	10	64	**0.1351**	**0.1695**	**0.1504**
		MTI		0	0	0.0000	0.0000	0.0000
History of Medicine	D006666	MTI	27	1	3	0.25	0.0370	0.0645
Horses	D006736	MTI	229	182	37	0.8311	0.7948	0.8125
Humans	D006801	AdaBoostM1	71,484	66,429	5,985	**0.9174**	**0.9293**	**0.9233**
		MTI		48,318	4,360	0.9172	0.6759	0.7783
Infant	D007223	AdaBoostM1 OverSampling	2,569	1,144	1,224	**0.4831**	**0.4453**	**0.4634**
		MTI		668	841	0.4427	0.2600	0.3276
Infant, Newborn	D007231	AdaBoostM1 OverSampling	1,985	1,042	851	0.5504	**0.5249**	**0.5374**
		MTI		850	419	**0.6698**	0.4282	0.5224
Male	D008297	AdaBoostM1 OverSampling	34,463	24,664	7,107	0.7763	**0.7157**	**0.7448**
		MTI		8,602	1,405	**0.8596**	0.2496	0.3869
Mice	D051379	MTI	7,144	5,332	810	0.8681	0.7464	0.8026
Middle Aged	D008875	AdaBoostM1 OverSampling	18,709	12,275	6,351	**0.6590**	**0.6561**	**0.6576**
		MTI		56	500	0.1007	0.0030	0.0058
Pregnancy	D011247	AdaBoostM1 OverSampling	2,637	1,988	653	**0.7527**	0.7539	**0.7533**
		MTI		2,107	880	0.7054	**0.7990**	0.7493
Rabbits	D011817	MTI	531	418	58	0.8781	0.7872	0.8302
Rats	D051381	MTI	4,577	3,681	443	0.8926	0.8042	0.8461
Sheep	D012756	AdaBoostM1 OverSampling	249	196	78	**0.7153**	0.7871	**0.7495**
		MTI		199	125	0.6142	**0.7992**	0.6946
Swine	D013552	AdaBoostM1 OverSampling	767	581	212	**0.7327**	**0.7575**	**0.7449**
		MTI		479	187	0.7192	0.6245	0.6685
United States	D014481	AdaBoostM1 OverSampling	3,510	1,369	2,130	**0.3913**	**0.3900**	**0.3906**
		MTI		1,007	1,614	0.3842	0.2869	0.3285
Young Adult	D055815	ROCCHIO.output	8,527	3,561	10,388	**0.2553**	**0.4176**	**0.3169**
		MTI		12	186	0.0606	0.0014	0.0026

MH: MeSH heading, DUI: descriptor unique identifier, P: positive, TP: true positive, FP: false positive.

Table 4. Micro and macro results for the Check Tags set

Methods	Micro P	Micro R	Micro F	Macro P	Macro R	Macro F
Meta-learning	0.7151	**0.7157**	**0.7154**	**0.5549**	**0.5236**	**0.5387**
medical text indexer	**0.8283**	0.3989	0.5385	0.4884	0.3567	0.4123

tion results for the selected method on the test data. We show that in most of the cases, the AdaBoostM1 with oversampling is the selected method. In Table 4, we compare overall Check Tag results with the MTI results. The performance of MTI is largely improved by meta-learning methods. In particular, Middle Aged, Young Adult and history-related terms profit from the use of alternative methods, which have very low MTI performance.

These results are in agreement with the experiments performed with all of MEDLINE, in which high frequency MeSH headings show a larger improvement, based on meta-learning.

VI. CONCLUSIONS AND FUTURE WORK

We have presented a framework that allows comparison of alternative indexing strategies, and an automated way of deciding on an optimal strategy for use with a large scale categorizer, namely MTI. We plan to add classifiers like SVMs, and to experiment with a larger set of MeSH headings with AdaBoostM1. In addition, we would like to include techniques that could learn with very imbalanced datasets, to improve the performance in lower frequency MeSH headings. The software used for these experiments is available at [30].

We have considered only the text from the title and abstract. More information is available in MEDLINE metadata, which might be exploited; examples include the journal and author affiliations.

Other sampling techniques, like synthetic sampling, might overcome some of the problems of oversampling and undersampling.

We would like to work as well on the automatic combination of indexing methods. This may require a combination of features and models, in which genetic programming might play a relevant role.

ACKNOWLEDGMENTS

This work was supported in part by the Intramural Research Program of the National Institutes of Health, National Library of Medicine (NLM), and by the appointment of A. Jimeno-Yepes to the NLM Research Participation Program sponsored by the NLM, and administered by the ORISE.

REFERENCES

1. US National Library of Medicine, Key MEDLINE indicators, http://www.nlm.nih.gov/bsd/bsd_key.html.
2. US National Library of Medicine, Medical Text Indexer (MTI), http://ii.nlm.nih.gov/mti.shtml.
3. A. R. Aronson, O. Bodenreider, H. F. Chang, S. M. Humphrey, J. G. Mork, S. J. Nelson, T. C. Rindflesch, and W. J. Wilbur, "The NLM Indexing Initiative," *American Medical Informatics Association (AMIA) Annual Symposium Proceedings,* 2000, pp. 17-21.
4. A. R. Aronson, J. G. Mork, C. W. Gay, S. M. Humphrey, and W. J. Rogers, "The NLM Indexing Initiative's medical text indexer," *Proceedings of the 11th World Congress on Medical Informatics*, San Francisco, CA, 2004, pp. 268-272.
5. A. R. Aronson and F. M. Lang, "An overview of MetaMap: historical perspective and recent advances," *Journal of the American Medical Informatics Association*, vol. 17, no. 3, pp. 229-236, 2010.
6. J. Lin and W. J. Wilbur, "PubMed related articles: a probabilistic topic-based model for content similarity," *BMC Bioinformatics*, vol. 8, p. 423, 2007.
7. K. W. Fung and O. Bodenreider, "Utilizing the UMLS for semantic mapping between terminologies," *American Medical Informatics Association (AMIA) Annual Symposium Proceedings,* 2005, pp. 266-270.
8. US National Library of Medicine, Principles of MEDLINE subject indexing, http://www.nlm.nih.gov/bsd/disted/mesh/indexprinc.html.
9. W. Hersh, C. Buckley, T. J. Leone, and D. Hickam, "OHSUMED: an interactive retrieval evaluation and new large test collection for research," *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Dublin, Ireland, 1994, pp. 192-201.
10. D. D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka, "Training algorithms for linear text classifiers," *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 1996, pp. 298-306.
11. M. E. Ruiz and P. Srinivasan, "Hierarchical neural networks for text categorization (poster abstract)," *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, CA, 1999, pp. 281-282.
12. M. Yetisgen-Yildiz and W. Pratt, "The effect of feature representation on MEDLINE document classification," *American Medical Informatics Association (AMIA) Annual Symposium Proceedings,* 2005, pp. 849-853.

13. G. L. Poulter, D. L. Rubin, R. B. Altman, and C. Seoighe, "MScanner: a classifier for retrieving Medline citations," *BMC Bioinformatics*, vol. 9, p. 108, 2008.
14. P. Ruch, "Automatic assignment of biomedical categories: toward a generic approach," *Bioinformatics*, vol. 22, no. 6, pp. 658-664, 2006.
15. Y. Aphinyanaphongs, I. Tsamardinos, A. Statnikov, D. Hardin, and C. F. Aliferis, "Text categorization models for high-quality article retrieval in internal medicine," *Journal of the American Medical Informatics Association*, vol. 12, no. 2, pp. 207-216, 2005.
16. D. Trieschnigg, P. Pezik, V. Lee, F. de Jong, W. Kraaij, and D. Rebholz-Schuhmann, "MeSH Up: effective MeSH text classification for improved document retrieval," *Bioinformatics*, vol. 25, no. 11, pp. 1412-1418, 2009.
17. A. Neveol, S. E. Shooshan, and V. Claveau, "Automatic inference of indexing rules for MEDLINE," *BMC Bioinformatics*, vol. 9, no. Suppl 11, p. S11, 2008.
18. A. Jimeno-Yepes, B. Wilkowski, J. G. Mork, E. van Lenten, D. Demner-Fushman, and A. R. Aronson, "A bottom-up approach to MEDLINE indexing recommendations," *American Medical Informatics Association (AMIA) Annual Symposium Proceedings*, 2011, pp. 1583-1592.
19. A. Jimeno-Yepes, J. G. Mork, B. Wilkowski, D. Demner-Fushman, A. R. Aronson, "MEDLINE MeSH indexing: lessons learned from machine learning and future directions," *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, Miami, FL, 2012, pp. 734-742.
20. R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial Intelligence Review*, vol. 18, no. 2, pp. 77-95, 2002.
21. A. Kalousis, "Algorithm selection via meta-learning," Ph.D. dissertation, University of Geneva, Geneva, Switzerland, 2002.
22. P. R. Cohen, *Empirical Methods for Artificial Intelligence*, Cambridge, MA: MIT Press, 1995.
23. monq.jfa – Java Finite Automata, http://monqjfa.berlios.de/
24. C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*, New York, NY: Cambridge University Press, 2008.
25. J. Rennie, L. Shih, J. Teevan, and D. Karger, "Tackling the poor assumptions of Naïve Bayes text classifiers," *Proceedings of the 20th International Conference on Machine Learning*, Washington, DC, 2003, pp. 616-623.
26. Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," *Proceedings of the 13th International Conference in Machine Learning*, Bari, Italy, 1996, pp. 148-156.
27. J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81-106, 1986.
28. J. D. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii, "Overview of BioNLP'09 shared task on event extraction," *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, Boulder, CO, 2009, pp. 1-9.
29. L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia, "Overview of BioCreAtIvE: critical assessment of information extraction for biology," *BMC Bioinformatics*, vol. 6, no. Suppl 1, pp. S1, 2005.
30. US National Library of Medicine, MTI ML, http://ii.nlm.nih.gov/MTI_ML/index.shtml.

Antonio Jimeno-Yepes

Antonio Jimeno-Yepes is a post-doctoral fellow at the U.S. National Library of Medicine. He received his M.S. in Computer Science from the University Jaume I, Castellón de la Plana, Spain in 2001, M.S. in Intelligent Systems from the University Jaume I, Castellón de la Plana, Spain in 2008, and PhD in Computer Science from the University Jaume I, Castellón de la Plana, Spain in 2009. He previously worked at CERN from 2000 to 2006, and at the European Bioinformatics Institute from 2006 to 2010.

James G. Mork

Jim Mork received his B.S. degree in Computer Science from Central Michigan University, and M.S. degree in Computer Science from The Johns Hopkins University. In 1997, he started work on the Indexing Initiative project at the U.S. National Library of Medicine as a contractor, where he is the lead developer for the Medical Text Indexer program.

Dina Demner-Fushman

Dina Demner-Fushman is a staff scientist at the U.S. National Library of Medicine. Her interest in biomedical language processing stems from years of clinical practice (M.D. obtained from Kazan State Medical Institute in 1980) and clinical research (Doctorate (Ph.D.) in Medical Science earned from Moscow Medical and Stomatological Institute in 1989.) She earned her B.S degree in Computer Science from Hunter College, CUNY in 2000, and her MS and PhD in Computer Science from the University of Maryland, College Park in 2003 and 2006, respectively.

Alan R. Aronson

Alan (Lan) R. Aronson, PhD, is a principal investigator at the Lister Hill Center, U.S. National Library of Medicine. His research focuses on applying natural language techniques to biomedical text, for tasks ranging from indexing and retrieval of the biomedical literature, to mining clinical text. His research group is responsible for NLM's Medical Text Indexer (MTI), which assists in various indexing efforts, including MEDLINE indexing.