Automatic algorithm selection for MeSH Heading indexing based on meta-learning

Antonio Jimeno-Yepes
National Library of Medicine
8600 Rockville Pike
Bethesda, 20894, MD, USA
antonio.jimeno@gmail.com

James G. Mork
National Library of Medicine
8600 Rockville Pike
Bethesda, 20894, MD, USA
mork@nlm.nih.gov

Dina Demner Fushman
National Library of Medicine
8600 Rockville Pike
Bethesda, 20894, MD, USA
ddemner@mail.nih.gov

Alan R. Aronson
National Library of Medicine
8600 Rockville Pike
Bethesda, 20894, MD, USA
alan@nlm.nih.gov

Abstract

We present a methodology that automatically selects indexing algorithms for each heading in MeSH, NLM's vocabulary for indexing MEDLINE. While manually comparing indexing methods is manageable with a limited number of MeSH headings, a large number of them makes automation of this selection desirable. Results show that this process can be automated based on previously indexed MEDLINE records. We find that AdaBoostM1 is better suited to index a group of MeSH headings 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 $p < 0.01$).

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 ever increasing size of MEDLINE, currently increasing by around 700k articles per year[1].

The Medical Text Indexer (MTI)[2] (Aronson et al., 2000; Aronson et al., 2004) is a support tool for assisting indexers as they add MeSH indexing to MEDLINE. MTI has two main components: MetaMap (Aronson and Lang, 2010) and the PubMed® Related Citations (PRC) algorithm (Lin and Wilbur, 2007). 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* (Fung and Bodenreider, 2005) approach which is based primarily on the semantic relationships among UMLS concepts (MMI). The PRC algorithm is a modified k-Nearest Neighbors (k-NN) algorithm which relies on document similarity to assign MeSH headings (MHs). 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 MHs which are not explicitly present in the title and abstract of the citation but which are used in similar contexts.

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

In this work, we present a methodology to select an indexing algorithm for each MeSH heading automatically. Experiments are performed on the whole set of MeSH headings and on a set MeSH headings known as Check Tags (CTs)[3]. Check Tags are a special class of MeSH Headings considered routinely for every article, which cover species, sex and human age groups, historical periods and pregnancy. We show that this

[1] http://www.nlm.nih.gov/bsd/bsd_key.html

[2] http://ii.nlm.nih.gov/mti.shtml

[3] http://www.nlm.nih.gov/bsd/disted/mesh/indexprinc.html

process can be automated based on previously indexed MEDLINE records.

2 Related work

We find that most of the existing MeSH indexing methods fit either into 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.

The task of MeSH indexing has been considered as a text categorization problem in the machine learning community. Publication of the OHSUMED collection (Hersh et al., 1994) containing all MEDLINE citations in 270 medical journals over a five-year period (1987-1991) including MeSH indexing, provided for a large body of data that enabled us to view MH assignment as a classification problem. The scope of the collection determines the subset of MeSH that can be explored. For example, (Lewis et al., 1996) and (Ruiz and Srinivasan, 1999) used 49 categories related to heart diseases with at least 75 training documents, and (Yetisgen-Yildiz and Pratt, 2005) expanded the number of headings to 634 disease categories. (Poulter et al., 2008) provides 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, 2006). Pattern matching considers only the inner structure of the terms but not the terms with which they cooccur. 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 MeSH headings and over 172,000 entry terms to assist the indexers in determining the appropriate MeSH headings to assign to a MEDLINE citation. Small scale studies with machine learning approaches already exist (Aphinyanaphongs et al., 2005; Yetisgen-Yildiz and Pratt, 2005). 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., 2009) 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 which rely on a more complex representation of the citations, e.g., learning based on Inductive Logic Programming (Névéol et al., 2008).

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 which automates the selection of indexing algorithms based on meta-learning.

3 Meta-learning

In machine learning, meta-learning (Vilalta and Drissi, 2002; Kalousis, 2002) 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.

Figure 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.

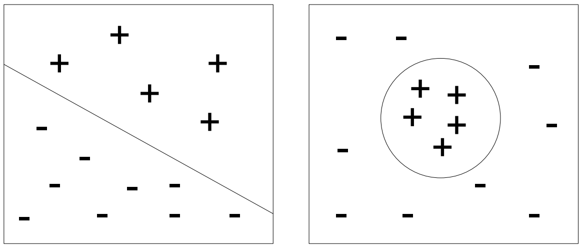

Figure 1: Instance sets

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.

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* MH.

Method	Average F-measure
MTI	0.72
Naïve Bayes	0.85
Support Vector Machine	0.88
AdaBoostM1	0.92

Table 1: F-measure for indexing methods on the *Humans* MeSH Heading

4 Methods

In this section, we present the training the framework, 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 indexing and 2011 MeSH. The citations are sorted by date so the first 200k citations are used for training and the remaining 100k for test.

4.1 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 is 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 are evaluated and the results averaged. We have considered 5 times 2-fold cross validation. Statistical significance of the results is computed using a randomization version of the two sample t-test (Cohen, 1995).

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

- 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.

- 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 is repeated for each MeSH heading. The best method for each heading is selected and stored in a mapping table. In the case of machine learning methods, the trained model for the best method is also stored into the table.

4.2 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.

4.3 MeSH indexing methods

Most of the indexing algorithms we study here require a training phase. MTI and dictionary lookup do not. 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. Only AdaBoostM1 has been used in a reduced set of MeSH headings. We are planning to include more learning algorithms as they are integrated in our system.

4.4 Dictionary lookup

This method looks for mentions of the MeSH heading terms 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 dictionary. MeSH is turned into a list of terms and IDs.

Our dictionary lookup implementation is based on the monqJFA package[4]. 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 is 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.

4.5 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 being indexed with the MeSH heading 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 the following equation, 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 of a citation not 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 the following equations, 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 (Manning et al., 2008) due to term sparsity. In our experiments, we have used a value for λ of 0.8.

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

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

[4] http://monqjfa.berlios.de

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

We have implemented, in addition, a variant of Naïve Bayes (NB) based on TF-IDF (Rennie et al., 2003) which has shown to improve the performance on traditional NB for text categorization.

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

4.6 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 MH I and the term occur together as we can see in the following formula.

$$\vec{q} = \{\frac{cf_{t_1,I}}{N}, \ldots, \frac{cft_V, I}{N}\} \tag{7}$$

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

4.7 AdaBoostM1

AdaBoostM1 (Schapire et al., 1996) 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 are performed.

In our experiments, we use C4.5 as the base learner since it produced good results in the past (Jimeno-Yepes et al., 2011) with a smaller set of MeSH headings. Our decision tree is an implementation of the C4.5 algorithm (Quinlan, 1986) with pruning and 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.

4.8 Voting

Combinations of methods have proved to increase performance of individual methods (Kim et al., 2009; Hirschman et al., 2005).

Given a citation, for a given MeSH heading the predictions for each of the indexing methods presented above are collected. Then, the number of votes are counted and if the sum of the votes is over a given threshold, the MeSH heading is predicted by this method.

5 Results and Discussion

We have performed two experiments. In the first one, we have considered all the MHs and trained algorithms which can handle a large number of categories and features. In the second one, we show results for the reduced set of MHs named Check Tags.

In both experiments, MTI annotation is considered the baseline method. Features for the machine learning algorithms are represented as the presence of tokens from the title and abstract of the citation, no frequency of the tokens in the citation text is used. Tokens are lowercased but not stemmed.

5.1 Results with all MeSH headings

This experiment is done on all MeSH headings. These experiments 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 than MTI is selected; either a single method or a voting combination of them. In table 2, we only show the set of MeSH headings grouped by learning method where MTI is outperformed by methods as selected by meta-learning. MTI is the best algorithms for the MeSH headings not included in this table.

Voting 3, at least three methods agree on predicting the MeSH heading, seems to perform better than the individual methods tested in this work. Surprisingly, dictionary lookup (MeSH TIAB DL) is performing reasonably well in some cases compared to MTI. Machine learning methods perform better only on a small set of MHs; one of the problems could be the small number of positive examples available for most of the MeSH headings. NB has good performance on the most frequent MeSH headings (*Humans*, *Male* and *Female*), which belong to the set of CTs. The modified NB (NBTFIDF) has a 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 appear significantly on MeSH headings which had a higher indexing frequency. The large imbalance and variability between the training and test might justify the results obtained with lower frequency MeSH headings. Another factor is that, since the MTI is the current system, it has been left as default if the differences with MTI were not statistically significant.

5.2 Results with Check Tags

In this experiment, we have included AdaBoostM1 as a learning algorithm. In table 3, we evaluate 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 the overall Check Tags 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 MHs show a larger improvement based on meta-learning.

6 Conclusions and Future Work

We have presented a framework which allows comparing alternative indexing strategies and an automated way of deciding on an optimal strategy a large scale categorizer, namely MTI. We plan to add classifiers like Support Vector Machines and to experiment with a larger set of MHs with AdaBoostM1. In addition, we would like to include techniques which could learn with very imbalanced datasets to improve the performance in lower frequency MeSH headings.

We have considered only the text from the title and abstract. More information is available in MEDLINE meta-data 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 the 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 NIH, NLM and by an appointment of A. Jimeno-Yepes to the NLM Research Participation Program sponsored by the NLM and administered by the ORISE.

References

Y. Aphinyanaphongs, I. Tsamardinos, A. Statnikov, D. Hardin, and C.F. Aliferis. 2005. Text categorization models for high-quality article retrieval in internal medicine. *Journal of the American Medical Informatics Association*, 12(2):207–216.

A.R. Aronson and F.M. Lang. 2010. An overview of MetaMap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229.

A.R. Aronson, O. Bodenreider, H.F. Chang, S.M. Humphrey, JG Mork, SJ Nelson, TC Rindflesch, and WJ Wilbur. 2000. The NLM Indexing Initiative. In *Proceedings of the AMIA Symposium*, page 17. American Medical Informatics Association.

A.R. Aronson, J.G. Mork, C.W. Gay, S.M. Humphrey, and W.J. Rogers. 2004. The NLM Indexing Initiative's Medical Text Indexer. In *Medinfo 2004: proceedings of the 11th World Conference on Medical Informatics*, page 268. OCSL Press.

Paul R. Cohen. 1995. *Empirical methods for artificial intelligence*. MIT Press, Cambridge, MA, USA.

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
Naive 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

Table 2: Results for MTI and Meta-learning for the 2,712 MHs

K.W. Fung and O. Bodenreider. 2005. Utilizing the UMLS for semantic mapping between terminologies. American Medical Informatics Association.

W. Hersh, C. Buckley, TJ Leone, and D. Hickam. 1994. OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201. Springer-Verlag New York, Inc.

L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia. 2005. Overview of biocreative: critical assessment of information extraction for biology. *BMC bioinformatics*, 6(Suppl 1):S1.

A. Jimeno-Yepes, B. Wilkowski, J.G. Mork, E. Van Lenten, D. Demner Fushman, and A.R. Aronson. 2011. A bottom-up approach to MEDLINE indexing recommendations. American Medical Informatics Association.

A. Kalousis. 2002. Algorithm selection via meta-learning. *University of Geneve, PhD Thesis*.

J.D. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii. 2009. Overview of bionlp'09 shared task on event extraction. In *Proceedings of the Workshop on BioNLP: Shared Task*, pages 1–9. Association for Computational Linguistics.

D.D. Lewis, R.E. Schapire, J.P. Callan, and R. Papka. 1996. Training algorithms for linear text classifiers. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 298–306. ACM.

J. Lin and W.J. Wilbur. 2007. PubMed related articles: a probabilistic topic-based model for content similarity. *BMC bioinformatics*, 8(1):423.

C.D. Manning, P. Raghavan, and H. Schütze. 2008. Introduction to information retrieval.

A. Névéol, S. Shooshan, and V. Claveau. 2008. Automatic inference of indexing rules for MEDLINE. *BMC bioinformatics*, 9(Suppl 11):S11.

G.L. Poulter, D.L. Rubin, R.B. Altman, and C. Seoighe. 2008. MScanner: a classifier for retrieving Medline citations. *BMC bioinformatics*, 9(1):108.

J.R. Quinlan. 1986. Induction of decision trees. *Machine learning*, 1(1):81–106.

J.D. Rennie, L. Shih, J. Teevan, and D. Karger. 2003. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 616–623.

P. Ruch. 2006. Automatic assignment of biomedical categories: toward a generic approach. *Bioinformatics*, 22(6):658.

M.E. Ruiz and P. Srinivasan. 1999. Hierarchical neural networks for text categorization (poster abstract). In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 281–282. ACM.

R.E. Schapire, Y. Freund, and R. Schapire. 1996. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156.

D. Trieschnigg, P. Pezik, V. Lee, F. De Jong, W. Kraaij, and D. Rebholz-Schuhmann. 2009. MeSH Up: effective MeSH text classification for improved document retrieval. *Bioinformatics*, 25(11):1412.

R. Vilalta and Y. Drissi. 2002. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95.

M. Yetisgen-Yildiz and W. Pratt. 2005. The effect of feature representation on MEDLINE document classification. In *AMIA Annual Symposium Proceedings*, volume 2005, page 849. American Medical Informatics Association.

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
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

Table 3: Results for MTI and Meta-learning for the CTs 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
MTI	0.8283	0.3989	0.5385	0.4884	0.3567	0.4123

Table 4: Micro and macro results for the CTs set