An empirical evaluation of supervised learning approaches in assigning diagnosis codes to electronic medical records

Ramakanth Kavuluru[a,b], Anthony Rios[b], Yuan Lu[b]

[a]*Division of Biomedical Informatics, Department of Biostatistics, University of Kentucky, 230E MDS Building, 725 Rose Street, Lexington KY 40536, USA*
[b]*Department of Computer Science, University of Kentucky, Davis Marksbury Building, 329 Rose Street Lexington, KY 40506, USA*

Abstract

Background: Diagnosis codes are assigned to medical records in healthcare facilities by trained coders by reviewing all physician authored documents associated with a patient's visit. This is a necessary and complex task involving coders adhering to coding guidelines and coding all assignable codes. With the popularity of electronic medical records (EMRs), computational approaches to code assignment have been proposed in the recent years. However, most efforts have focused on single and often short clinical narratives, while realistic scenarios warrant full EMR level analysis for code assignment.

Objective: We evaluate supervised learning approaches to automatically assign international classification of diseases (ninth revision) - clinical modification (ICD-9-CM) codes to EMRs by experimenting with a large realistic EMR dataset. The overall goal is to identify methods that offer superior performance in this task when considering such datasets.

Methods: We use a dataset of 71,463 EMRs corresponding to in-patient visits with discharge date falling in a two year period (2011–2012) from the University of Kentucky (UKY) Medical Center. We curate a smaller subset of this dataset and also use a third gold standard dataset of radiology reports. We conduct experiments using different problem transformation approaches with feature and data selection components and employing suitable label calibration and ranking methods with novel features involving code co-occurrence frequencies and latent code associations.

Results: Over all codes with at least 50 training examples we obtain a micro F-score of 0.48. On the set of codes that occur at least in 1% of the two year dataset, we achieve a micro F-score of 0.54. For the smaller radiology report dataset, the classifier chaining approach yields best results. For the smaller subset of the UKY dataset, feature selection, data selection, and label calibration offer best performance.

Conclusions: We show that datasets at different scale (size of the EMRs, number of distinct codes) and with different characteristics warrant different learning approaches. For shorter narratives pertaining to a particular medical subdomain (e.g., radiology, pathology), classifier chaining is ideal given the codes are highly related with each other. For realistic in-patient full EMRs, feature and data selection methods offer high performance for smaller datasets. However, for large EMR datasets, we observe that the binary relevance approach with learning-to-rank based code reranking offers the best performance. Regardless of the training dataset size, for general EMRs, label calibration to select the optimal number of labels is an indispensable final step.

Keywords: Multi-label text classification, learning to rank, label calibration, diagnosis code assignment

1. Introduction

Assigning codes from standard terminologies is a regular and indispensable task often encountered in medical and healthcare fields. Diagnosis codes, procedure codes, cancer site and morphology codes are all manually extracted from patient records by trained human coders. The extracted codes serve multiple purposes including billing and reimbursement, quality control, epidemiological studies, and cohort identification for clinical trials. In this paper we focus on assigning international classification of diseases, clinical modification, 9th revision (ICD-9-CM) diagnosis codes to electronic medical records (EMRs), and the application of supervised multi-label text classification approaches to this problem.

Diagnosis codes are the primary means to systematically encode patient conditions treated in healthcare facilities both for billing purposes and for secondary data usage. In the US, ICD-9-CM (just ICD-9 henceforth) is the coding scheme still used by many healthcare providers while they are required to comply with ICD-10-CM, the next and latest revision, by October 1, 2015. Regardless of the coding scheme used, both ICD code sets are very large, with ICD-9 having over 14,000 diagnoses while ICD-10 has nearly 68,000 diagnosis codes [1] and as will be made clear in the rest of the paper, our methods will also apply to ICD-10 coding tasks. ICD-9 codes contain 3 to 5 digits and are organized hierarchically: they take the form `abc.xy` where the first three character part before the period `abc` is the main disease category, while the `x` and `y` components represent subdivisions of the `abc` category. For example, the code `530.12` is for the condition *reflux esophagitis* and its parent code `530.1` is for the broader condition of *esophagitis* and the three character code `530` subsumes all *diseases of esophagus*. Any allowed code assignment should at least assign codes at the category level (that is, the first three digits). At the category levels there are nearly 1300 different ICD-9 codes.

The process of assigning diagnosis codes is carried out by trained human coders who look at the entire EMR for a patient visit to assign codes. Majority of the artifacts in an EMR are textual documents such as discharge summaries, operative reports, and progress notes authored by physicians, nurses, or social workers who attended the patient. The codes are assigned based on a set of guidelines [2] established by the National Center for Health Statistics and the Centers for Medicare and Medicaid Services. The guidelines contain rules that state how coding should be done in specific cases. For example, the signs and symptoms (`780-799`) codes are often not coded if the underlying causal condition is determined and coded. Given the large set of possible ICD-9 codes and the need to carefully review the entire EMR, the coding process is a complex and time consuming process. Hence, several attempts

Email addresses: `ramakanth.kavuluru@uky.edu` (Ramakanth Kavuluru), `anthony.rios1@uky.edu` (Anthony Rios), `yuan.lu@uky.edu` (Yuan Lu)

have been made to automate the coding process. However, computational approaches are inherently error prone. Hence, we would like to emphasize that automated medical coding systems, including our current attempt, are generally not intended to replace trained coders but are mainly motivated to expedite the coding process and increase the productivity of medical record coding and management.

The main focus of this paper is supervised text classification approaches to automatically assign[1] ICD-9 codes to clinical narratives using a large EMR dataset of patients discharged during a two year period (2011–2012) from the University of Kentucky (UKY) Medical Center. We will refer to this dataset as UKLarge in the rest of the paper. The correct codes for the EMRs in this dataset are those assigned by one of the trained coders in the UKY medical records office. To study the affect of scale, we also use two other smaller datasets: a gold standard dataset created for the BioNLP 2007 shared task [3] by researchers affiliated with the Computational Medicine Center (CMC: `http://computationalmedicine.org/`) and a subset of the UKLarge dataset, called UKSmall, that is comparable to the number of records used in BioNLP dataset. The CMC dataset is a high quality, but relatively small and simple dataset of clinical reports that covers the pediatric radiology domain. The gold standard correct codes are provided for each report. To experiment with a more general dataset with a comparable number of records to the CMC dataset, we curated UKSmall, a subset of the UKLarge dataset.

In supervised classification, multi-label problems are generally transformed into several multi-class (where each object belongs to a single class) or binary classification problems. In this effort, we explore these different transformation techniques with different base classifiers (support vector machines, naive Bayes, and logistic regression). Large label sets, high label cardinality (number of labels per instance), class imbalance, inter-class correlations, large feature sets are some of the factors that negatively affect performance in multi-label classification problems. We experiment with different state-of-the-art feature selection, training data selection, classifier chaining, and label calibration approaches to address some of these issues. We achieve comparable results to the state-of-the-art on the CMC dataset. Our experiments reveal how the differences in the nature of our three datasets significantly affect the performance of different supervised approaches. To our knowledge, our current contribution is the first of its kind in terms of the total number of codes considered and the realistic nature of the multi-document EMRs (Section 3). We design a sophisticated end-to-end pipeline involving novel features that depend on output code co-occurrences to rerank codes and to predict the correct number of codes per EMR (Sections 4.6 and 4.7). While our current task is diagnosis code assignment, our methods can also be used by the AI community to situations where coded biomedical information needs to be extracted from textual narratives.

The rest of the paper is organized as follows: In Section 2 we discuss related work on automated diagnosis code assignment and provide background on multi-label classification approaches. We elaborate on the statistics and characteristics of the three different datasets

[1]Throughout the article we use the verb *assign* to indicate the prediction of diagnosis codes by multi-label classification of the EMR narratives because the codes (or the corresponding English names) may not be present in the EMR for straightforward extraction through named entity recognition.

used in Section 3. In Section 4, we present details of different multi-label text classification learning components used in our experiments. After a brief discussion of evaluation measures in Section 5, we present our results in Section 6. We present a qualitative error analysis of our results and identify opportunities for improvement in Section 7.

2. Related Work and Background

In this section we discuss related work and prior efforts in assigning ICD-9 codes and present a brief general background for multi-label classification techniques.

Several attempts have been made to automatically assign ICD-9 codes to clinical documents since the 1990s. Advances in natural language and semantic processing techniques contributed to a recent surge in automated coding. de Lima et al. [4] use a hierarchical approach utilizing the alphabetical index provided with the ICD-9-CM resource. Although completely unsupervised, this approach is limited by the index not being able to capture all synonymous occurrences and also the inability to code both specific exclusions and other condition specific guidelines. Gunderson et al. [5] extracted ICD-9 codes from short free text diagnosis statements that were generated at the time of patient admission using a Bayesian network to encode semantic information. However, in the recent past, concept extraction from longer documents such as discharge summaries has gained interest. Especially for ICD-9 code assignment, recent results are mostly based on the systems and the CMC dataset developed for the BioNLP workshop shared task on multi-label classification of clinical texts [3] in 2007.

The CMC dataset consists of 1954 radiology reports arising from outpatient chest x-ray and renal procedures and is observed to cover a substantial portion of pediatric radiology activity. The radiology reports are also formatted in XML with explicit tags for *history* and *impression* fields. Finally, there are a total of 45 unique codes and 94 distinct combinations of these codes in the dataset. The dataset is split into training and testing sets of nearly equal size where example reports for all possible codes and combinations occur in both sets. This means that all possible combinations that will be encountered in the test set are known ahead of time. The top system obtained a micro-average F-score of 0.89 and 21 of the 44 participating systems scored between 0.8 and 0.9. Next we list some notable results that fall in this range obtained by various participants and others who used the dataset later. The techniques used range from completely handcrafted rules to fully automated machine learning approaches. Aronson et al. [6] adapted a hybrid medical subject heading (MeSH) assignment program called the medical text indexer (MTI) that is in use at the National Library of Medicine (NLM) and included it with SVM and k nearest neighbor classifiers for a hybrid *stacked* model. Goldstein et al. [7] applied three different classification approaches - traditional information retrieval using the search engine library Apache Lucene, Boosting, and rule-based approaches. Crammer et al. [8] use an online learning approach in combination with a rule-based system. Farkas and Szarvas [9] use an interesting approach to induce new rules and acquire synonyms using decision trees. Névéol et al. [10] also model rules based on indexing guidelines used by coders using semantic predications to assign MeSH heading-subheading pairs to indexing biomedical articles. A recent attempt [11] also exploits the hierarchical nature of the ICD-9-CM terminology to improve the performance achieving comparable performance to the best scores achieved during the competition. Although not applicable/practical in all situations, researchers have also tried to predict codes purely based on structured resources in the EMR [12].

In terms of dataset size, there are two recent efforts similar to ours. The first is by Ruch et al. [13] who worked on French EMRs. They used a k-NN approach to obtain codes from training documents. However, their results are very inconclusive as they focused on example-based recall R20, the recall at top 20 codes, without the accompanying precision values. Instead, they report P0, which is precision of the top most code. The second effort is by Perotte et al. [14] conducted independently around the same time our research was conducted. They use flat and hierarchical SVM approaches with a dataset of nearly 22,000 discharge summaries of intensive care unit patients and the corresponding diagnosis codes. They achieve an F-score of 39.5% over a set of 5000 codes where the definitions of true positives and false negatives are relaxed based on the ICD-9 hierarchy in a way that is broader than our treatment of codes at the fourth digit level (see Section 3). Their effort also considers single summaries while we consider full EMRs based on in-patient visits. Hence our findings are not directly comparable with their results.

Next, we provide a brief review of the background and state-of-the-art in multi-label classification, the general problem of choosing multiple labels among a set of possible labels for each object that needs to be classified. A class of approaches called problem transformation approaches convert the multi-label problem into multiple single-label classification instances. A second class of methods adapts the specific algorithms for single-label classification problems to directly predict multiple labels. Both problem transformation and algorithm adaptation techniques are covered in this recent survey by [15]. Recent attempts in multi-label classification also consider label correlations [16, 17, 18] when building a model for multi-label data. An important challenge in problems with a large number of labels per document is to decide the number of candidates after which candidate labels should be ignored, which has been recently addressed by calibrated ranking [19] and probabilistic thresholding [20]. Feature selection is an important aspect when building classifiers using machine learning. We request the readers to refer to Forman [21] for a detailed comparative analysis of feature selection methods. Combining the scores for each feature using different feature selection methods has also been applied to multi-label classification [22]. When dealing with datasets with class imbalance, methods such as random under/over-sampling, synthetic training sample generation, and cost-sensitive learning were proposed (see [23] for a survey). In contrast with these approaches, Sohn et al [24] propose an alternative Bayesian approach to curate customized optimal training sets for each label. In the next section, we discuss the characteristics of the three datasets used in this paper.

3. Datasets

We already introduced the CMC dataset in Section 2 when discussing related work. Here we will give some additional details to contrast it with our private datasets. From the 1954 reports in the CMC dataset, 978 are included in the training dataset with their corresponding ICD-9 codes; the remaining documents form the testing set. All labels sets that occur in the testing set occur at least once in the training dataset. At least 75% of the codes appear less than 50 times in the training set. Almost 50% of the 45 labels appear less than ten times. For each instance in the CMC dataset there are two fields that contain textual data, ‘clinical history’ and ‘impression’. Clinical history field contains textual information entered by a physician before a radiological procedure about the patient’s history. The impression field contains the textual narrative entered by a radiologist after the radiological procedure about his/her observations of the patient’s condition as obtained from the procedure. Many of the textual entries contained in these two fields are very short. An example of the clinical history field is “22 month old with cough.” The corresponding impression is just one word “normal.” The average size of a report is 21 words.

We created two datasets from EMRs of the UKY medical center in-patient visits with discharge dates in the 2011–2012 two year period. They have been approved by the UKY IRB for use in research projects (protocol #12-0139-p3h). The first dataset UKLarge is the largest and consists of all in-patient visits during this time period. We also collected the ICD-9 codes for these EMRs assigned by one of the trained coders at the UKY medical records office. There are a total of 71,461 EMRs that have a total of 7,485 unique ICD-9 codes. The average number of codes is 9.76 per EMR with a median of 8 codes. For each in-patient visit, the original EMR consisted of several documents, some of which are not conventional text files but are stored in the RTF format. There were approximately 921,000 physician authored documents in the entire dataset, so an average of 13 documents per EMR. Since many of the codes in the UKLarge have very few examples, we decided to consider predicting codes at the fourth digit level. That is, all codes of the form `abc.xy` for different ‘`y`’ are mapped to the four digit code `abc.x`, but codes that were already coded at the fourth or third digit level were retained in that form. With this mapping we had 4,723 unique codes. The average size of each EMR (that is, of all textual documents in it) in the UKLarge dataset is 5303 words. Even when truncated to 4 digits, there were still many codes that had too few examples to apply supervised methods. Hence we resorted to using 1231 codes (at the fourth digit level) that had at least 50 EMRs in the dataset. That is, our predictions are going to be only on these 1231 codes, and not all of the 4,723 codes. This subset of codes accounts for 76% of the total diagnoses in the dataset. There are 60,238 unique combinations of these 1231 codes in the data.

Label-cardinality is the average number of codes per report (or EMR in the UKY dataset). To use consistent terminology we refer to the single reports in the CMC dataset as EMRs that consist of just one document. Let m be the total number of EMRs and $\mathbf{Y}_i$ be the set of labels for the i-th EMR. Then we have

$$\text{Label-Cardinality} = \frac{1}{m}\sum_{i=1}^{m} |\mathbf{Y}_i|.$$

The CMC training dataset has label cardinality 1.24 and the UKLarge dataset has label cardinality 7.4. Another useful statistic that describes the datasets is *label-density*, which divides the average number of codes per EMR by the total number of unique codes q. We have

$$\text{Label-Density} = \frac{1}{q} \cdot \frac{1}{m}\sum_{i=1}^{m} |\mathbf{Y}_i|.$$

The label-density for the CMC dataset is 0.03 and for UKLarge is 0.006. Unlike label-cardinality, label-density also takes into account the number of unique labels possible. Two datasets with the same label cardinality can have different label densities and might need different learning approaches tailored to the situations. Intuitively, in this case, the dataset with the smaller density is expected to have fewer training examples per label.

Table 1: Comparison of datasets

	CMC	UKSmall	UKLarge
#EMRs	1954	1000	71,463
#Codes	45	56	1231
Label Cardinality	1.24	2.8	7.4
Label Density	0.03	0.06	0.006
EMR Size (avg #words)	21	2088	5303
#Code Combinations	94	554	60,238

As we can see, the datasets have significant differences: the CMC data set is coded by three different coding companies and final codes were consolidated from these three different assignments. As such, it is of higher quality compared to UKLarge, which is coded by only one of the eight trained coders from the UKY medical records office. On the other hand, the CMC dataset does not have the broad coverage of the UKLarge, which models a more realistic dataset at the EMR level. The CMC dataset only includes radiology reports and has 45 codes with 94 code combinations and has on an average 21 words per EMR. In contrast, even with the final set of 1231 codes (at the four digit level) that have at least 50 examples that we use for our experiments, the number of combinations for the UKY dataset is 60,238 with the average EMR size two orders of magnitude more than the average for the CMC dataset.

We created a subset of UKLarge, called UKSmall, with 1000 EMRs corresponding to a randomly chosen set of 1000 in-patient visits from February, 2012. Although in terms of the nature of the EMR documents this dataset shares the same traits as the UKLarge dataset, it has fewer codes and also fewer examples per code when considering absolute count. We also removed all EMR documents whose names did not contain any of the words “report”, “summary”, “note”, or “portion” based on coding specific knowledge that important notes for coding tasks contain these words. This halves the size of our EMRs in this dataset and from an automatic code extraction perspective has been shown to be detrimental for code recall (more in Section 7). For this dataset, we made prediction only for those codes that had at least 20 examples; although this number is less than the minimum requirement of 50 examples for the UKLarge dataset, considering the size of the datasets, 20 examples out of a total of 1000 examples constitute a 2% presence. However, 50 examples in a dataset of over 71,000 examples, constitute only 0.07% of the UKLarge dataset, which is an order of magnitude fewer examples than that for the UKSmall dataset. This is also reflected by the label density for the two datasets as shown in the summary of all three datasets in Table 1. This crucial difference translates to high class imbalance in the UKLarge dataset compared with the other two datasets. This was the original dataset used in the conference version of the paper [25] and offers a different perspective on suitability of different learning components and performance variations based on differences in size and nature of the datasets available (Section 7).

4. Components of the Code Assignment Framework

The main motivation for our effort is to assess the suitability of feature selection, data selection, label calibration, and different problem transformation methods for extracting diagnosis codes from EMRs on datasets at different scale (the three datasets in Section 3). The learning components we use fall into these four categories that are arranged in the order they are used.

1. First, we use a problem transformation approach and transform the multi-label classification problem into multiple binary classification problems. We also use different approaches that take into account label correlations expressed in the training data (Section 4.3).
2. After problem transformation, we utilize feature selection and training data selection as additional components of the binary classifiers (Sections 4.4 and 4.5).
3. Using the binary classifier outputs for each label, we use the learning-to-rank approach to rerank top label predictions from the binary classifiers based on output code co-occurrences (Section 4.6).
4. Finally, we also use label calibration methods to predict an appropriate number of labels for each EMR instead of a choosing a constant number for all (Section 4.7).

We used the Java based packages Weka [26] and Mulan [27] for our classification experiments with the smaller datasets. For the UKLarge dataset, we used Scikit-Learn [28] for binary classification and RankLib (`http://sourceforge.net/p/lemur/wiki/RankLib/`) for reranking the binary classifier predictions. Before going over components in each of the four categories in the prediction pipeline, we first outline the text features used.

4.1. EMR Text Features

We used unigram and bigram counts as the common features in all experiments. Stop words (determiners, prepositions, and so on) were removed from the unigram features. For the CMC dataset where the report size is very short, we used the binary features, but for the other two datasets we used the popular *tf-idf* transformation [29, Section 4] of word counts. In addition to these syntactic features, we also used semantic features such as *named entities* and binary relationships (between named entities) extracted from text, popularly called *semantic predications*, as features for the smaller datasets. Due to computational resource availability constraints, we could not use named entities or relationships as features for the UKLarge dataset. To extract named entities and semantic predications, we used software made available through the Semantic Knowledge Representation (SKR) project by the NLM. The two software packages we used were MetaMap and SemRep. MetaMap [30] is a biomedical named entity recognition (NER) program that identifies concepts from the Unified Medical Language System (UMLS) Metathesaurus, an aggregation of over 160 biomedical terminologies. When MetaMap outputs different named entities, it associates a confidence score in the range from 0 to 1000. We only used concepts with a confidence score of at least 700 as features. Each of the concepts extracted by MetaMap also contains a field

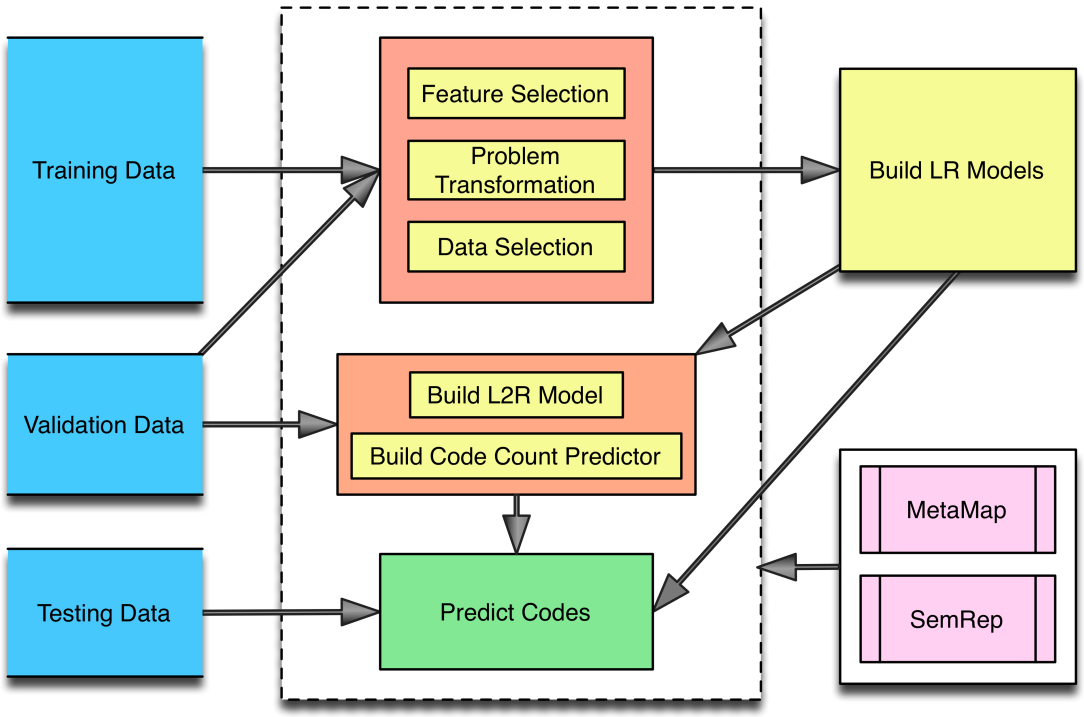

Figure 1: Diagnosis code assignment pipeline and constituent learning components

specifying if the concept was negated (e.g., "no evidence of hypertension"). We used negated concepts that capture the absence of conditions/symptoms as different features from the original concepts. We used SemRep, a relationship extraction program developed by Thomas Rindflesch [31] and team at the NLM that extracts semantic predications of the form $C1 \rightarrow relationType \rightarrow C2$ where $C1$ and $C2$ are two different biomedical named entities and $relationType$ expresses a relation between them (e.g., "Tamoxifen *treats* Breast Cancer"). If there is more than one document in an EMR, features are aggregated from all documents authored by a physician.

4.2. Diagnosis Code Assignment Pipeline

With this introduction, we present the overall code prediction pipeline in Figure 1. As shown in the figure, we first build the best models using problem transformation, feature selection, and data selection methods. Using the top ranked codes from these models for each EMR, we learn a ranking function that uses several novel features (Section 4.6) including the classifier scores based on a validation dataset. Subsequently, we also use the validation dataset to learning a function that predicts the correct number of codes for each EMR, which is used to select case specific top few codes after the learning-to-rank component reranks the original binary classifier based positions. NLM tools MetaMap and SemRep are not only used as features for the binary classifiers but also as features and filters in ranking the codes and predicting the correct number of labels.

4.3. Problem Transformation Approaches for Multi-Label Classification

We initially experimented with three base classifiers on codes in the UKSmall dataset: Support Vector Machines (SVMs), Logistic Regression (LR), and Multinomial Naive Bayes (MNB). We used the MNB classifier that is made available as part of the Weka framework. For LR, we used LibLINEAR [32] implementation in Weka/Scikit-Learn and for SVMs we used LibSVM [33] in Weka. We experimented with three different multi-label problem transformation methods: *binary relevance (BR), copy transformation*, and *ensemble of classifier chains*.

Let T be the set of labels and let $q = |T|$. BR learns q binary classifiers, one for each label in T. It transforms the dataset into q separate datasets. For each label T_j, we obtain the dataset for the corresponding binary classifier by considering each document–label-set pair $(D_i, \mathbf{Y}_i)$ and generating the document-label pair (D_i, T_j) when $T_j \in \mathbf{Y}_i$ and generating the pair $(D_i, \neg T_j)$ when $T_j \notin \mathbf{Y}_i$. When predicting, the labels are ranked based on their score output by the corresponding binary classifiers and the top k labels are considered as the predicted set for a suitable k. The copy transformation transforms multi-label data into single-label data. Let $T = \{T_1, \ldots, T_q\}$ be the set of q possible labels for a given multi-label problem. Let each document $D_j \in D$, $j = 1, \ldots, m$, have a set of labels $\mathbf{Y}_j \subseteq T$ associated with it. The copy transformation transforms each document–label-set pair $(D_j, \mathbf{Y}_j)$ into $|\mathbf{Y}_j|$ document–label pairs (D_j, T_s), for all $T_s \in \mathbf{Y}_j$. After the transformation, each input pair for the classification algorithm will only have one label associated with it and one can use any single-label method for classification. The labels are then ranked based on the score given from the classifier when generating predictions. We then take the top k labels as our predicted label set.

One of the main disadvantages of the BR and copy transformation methods is that they assume label independence. In practical situations, there can be dependence between labels where labels co-occur very frequently or where a label occurs only when a different label is also tagged. Classifier chains [16], based on the BR transformation, try to account for these dependencies that the basic transformations cannot. Like BR, classifier chains transform the dataset into q datasets for binary classification per each label. But they differ from BR in the training phase. Classifier chains loop through each dataset in some order, training each classifier one at a time. Each binary classifier in this order will add a new Boolean feature to the subsequent binary classifier datasets to be trained next. For further details of the chaining approach and the ensemble of chains modification that overcomes dependence on the chaining order, we request the reader to refer to the paper by Read et al. [16].

4.4. Feature Selection

An important issue in classification problems with a large number of classes and features is that the features most relevant in classifying one class from the rest might not be the same for every class. Furthermore, many features are either redundant or irrelevant to the classification tasks at hand. In the domain of text classification, Bi-Normal Separation (BNS) score was observed to result in best performance in most situations among a set of 12 feature selection methods applied to 229 text classification problems [21]. We employ this feature scoring method for our current effort. Let $T = \{T_1, \ldots, T_q\}$ be the set of q possible labels. For each label $T_i \in T$ and for each feature, we calculate the number of true positives (tp) and false positives (fp) with respect to that feature – tp is the number of training EMRs (with label T_i) in which the feature occurs. Similarly, fp is the number of negative examples for T_i in which the feature occurs. Let pos and neg be the total number of positive and negative examples for T_i, respectively. With F^{-1} denoting the inverse cumulative probability function for the standard normal distribution, we define the BNS score of a given feature for a particular class as

$$BNS = |F^{-1}(tpr) - F^{-1}(fpr)|$$

where

$$tpr = \frac{tp}{pos} \text{ and } fpr = \frac{fp}{neg}.$$

Because $F^{-1}(0)$ is undefined, we set tpr and fpr to 0.0005 when tp or fp are equal to zero. For each of the q binary classification problems, we pick an appropriate top k ranked features for each label. In our experiments $k = 8000$ gave the best performance out of a total of 68,364 features for the UKSmall dataset. The number of features was very small for the CMC dataset, a total of 2296 features, feature selection did not improve the performance. The UKLarge dataset had around a million features and feature selection did not improve performance.

4.5. Greedy 'Optimal' Training Data Selection

In the case of multi-label problems with a large number of classes, the number of negative examples is overwhelmingly larger than the number of positive examples for most labels. We experimented with the synthetic minority oversampling approach [34] for the positive examples which did not prove beneficial for our task. Deviating from the conventional random under/over-sampling approaches, we adapted the 'optimal' training set (OTS) selection approach used for medical subject heading (MeSH terms) extraction from biomedical abstracts by Sohn et al. [24]. The OTS approach is a greedy Bayesian approach that under-samples the negative examples to select a customized dataset for each label. The greedy selection is not technically optimal but we stick with the terminology in [24] for clarity. Our adaptation is described in detail in the conference version of our current contribution [25, Section IV(D)]. Essentially, the method ranks negative examples according to their content similarity to positive examples and iteratively selects negative examples according to this ranking and finally selects the negative instance subset that offers the best performance on a validation set for that label. The intuition behind this approach is that the negative examples that are 'closest' to the positive examples in terms of their content are the hardest to classify. Hence, choosing a subset that is the harder to distinguish from the positive examples provides the best outcome instead of using all the negative examples that could be easily distinguished but nevertheless add significant noise in the training process.

4.6. Learning-to-Rank to Rerank Code Candidates

As discussed in Section 4.3, the BR approach of building a binary classifier for each label does not take label correlations into account. Although the classifier chain transformation we discussed in that section helps account for label dependence, it is impractical with a large number of labels owing to the exponential number of chaining permutations for the ensemble. Other approaches such as ensemble of pruned label sets (EPS) [18] consider sets of labels as new special labels. Even this approach leads to combinatorial explosion for large label spaces. From Table 1 we have over 60,000 unique label combinations for UKLarge. Hence we need a different and computationally feasible way of accounting for label dependencies to rerank the original ranking of codes that is based on the binary classifiers. Originally, this reranking was done in an unsupervised way where empirical evidence is used to manually assign weights to different features that are pertinent to the ranking task. This is tedious and also impractical if the number of features used in the ranking is large. To handle this, as a collaboration between information retrieval and machine learning communities, learning-to-rank [35] has emerged as an automated way of learning functions that can rank a list of documents in response to an input query based on different query-specific features extracted from the documents. Although the original motivation was to rank documents returned from a search engine, this approach can be adapted to our current situation in a straightforward way. We first outline the approach in the information retrieval context.

4.6.1. Learning-to-Rank Basics

A learning-to-rank algorithm follows a supervised approach and in its training phase, takes as input a training dataset of queries and the corresponding ranked lists of documents:

$$\{(Q_i, \mathcal{R}(D_i)) : i = 1, \ldots, n\}, \tag{1}$$

where Q_i is a query, D_i is the set of documents associated with Q_i, $\mathcal{R}(D_i)$ is the ground truth ranking on the documents for Q_i, and n is the size of the training dataset. The algorithm then learns a ranking model that minimizes an appropriate loss function that pertains to the ranking. We note that the training process actually extracts features $f_1(D_i^r), f_2(D_i^r), \ldots,$ $r = 1, \ldots, t(i)$, for each document $D_i^r \in D_i$ where $t(i)$ is the number of documents provided to the i-th query instance for training and j is an index for the particular features used. Note that features extracted from the documents are heavily based on the specific query to tightly constrain the ranking based on information available in the query. Finally, given a new query and a specific set of documents as input, the learned model imposes a ranking on the document set based on query specific document features. This is a general description of the problem; for a more detail discussion of the *pointwise, pairwise,* and *listwise* variants see [35, Section 1.3.3].

Next, we map our problem of ranking ICD-9 codes to a listwise variant based on random forests that is implemented in the RankLib library, an open source collection of learning-to-rank implementations part of the Lemur project: `http://sourceforge.net/p/lemur/wiki/RankLib/`. We experimented with several of the algorithms in RankLib and chose the random forests based implementation that gave the best results on a validation dataset with normalized discounted cumulative gain (NDCG) [36] as the optimization metric. In Equation (1), our queries Q_i are the EMRs (the text from the EMRs) and the documents D_i are the top 200 candidate ICD-9 codes when ranked solely based on the binary classifier rankings. We chose 200 candidates because the top 200 terms had close to 90% of all possible correct terms on a validation dataset for the UKLarge dataset. Next we discuss the features for the the learning-to-rank.

4.6.2. Learning-to-Rank Candidate Code Features

An obvious feature of a candidate code c is

$$f^b(c, E) = \text{score } (\in [0, 1]) \text{ output by the corresponding binary classifier for } c$$

for EMR E. Next, we introduce a Boolean feature based on NER.

As introduced in Section 4.1, MetaMap is an NER tool that identifies biomedical concepts from over 160 source terminologies incorporated in the UMLS Metathesaurus (see NLM resource: `http://www.ncbi.nlm.nih.gov/books/NBK9684/`). ICD-9-CM is one of these terminologies and hence concepts in ICD-9-CM also have a concept unique identifier (CUI) in the Metathesaurus. As part of its output, for each concept extracted, MetaMap also gives the source vocabulary and the string representation of the code in the source terminology. Thus when MetaMap is run on an EMR text document set, we also obtain a set of ICD-9 codes, say $\mathcal{N}(E)$, by filtering the non-negated concepts to only those that are from the ICD-9-CM terminology. Note that codes in $\mathcal{N}(E)$ may not all be coded for the EMR by trained coders because mere mentions in the EMR text does not necessarily warrant inclusion. These NER extracted codes can nevertheless be included in learning-to-rank as a Boolean feature

$$f^n(c, E) = \begin{cases} 1 & \text{if } c \in \mathcal{N}(E); \\ 0 & \text{otherwise.} \end{cases}$$

Besides these two features, we also exploit ICD-9 code sets from historical records at the UKY medical center to obtain two additional features. From the time when the UK medical center moved to electronic record keeping to late 2011, we have a total of around 2 million visits and each visit has a set of diagnosis codes. From this, we can obtain code co-occurrence counts independent of the actual EMR texts. Intuitively, a predicted candidate code that highly co-occurs with the ICD-9 codes in $\mathcal{N}(E)$ (the ones extracted from EMR text using NER) is probably a more relevant code for the EMR over a predicted code with low co-occurrence with codes extracted using NER. We demonstrated that this co-occurrence based score is useful for unsupervised multi-label classification in the context of indexing biomedical citations with MeSH terms [37]. Here we use the co-occurrence score of a candidate term with the contextual codes extracted using NER from the EMR text as a feature for learning-to-rank. The co-occurrence score is computed by first building a square matrix

$$\mathcal{M}[i][j] = \frac{\text{number of code sets containing both } i\text{-th and } j\text{-th ICD-9 codes}}{\text{number of code sets containing the } i\text{-th code}}$$

that holds the normalized co-occurrence scores with size equal to the number of all unique ICD-9 codes that occurred in the 2 million code sets. Note that $\mathcal{M}[i][i] = 1$ because the numerator would be equal to the denominator. With this definition, $\mathcal{M}[i][j]$ is an estimate of the probability $P(j\text{-th code}|i\text{-th code})$. Finally, we introduce the co-occurrence frequency based feature for a candidate code c as

$$f^F(c, E) = \sum_{t \in \mathcal{N}(E)} \mathcal{M}[t][c].$$

Our final feature is based on measuring relatedness between any given pair of codes using distributional semantics [38], a well known methodology traditionally used to identify implicit relatedness between words in a corpus of documents. Although the co-occurrence based feature $f^F(c, E)$ captures direct association between the candidate code and the context codes of a test set EMR, it does not capture latent or implicit associations between codes that might not have co-occurred frequently in historical data but are nevertheless strongly associated from a distributional semantics perspective. We map each of the 2 million code sets to a document and the constituent ICD-9 codes to terms to build a distributional semantic index using the term based reflective random indexing (TRRI) approach by Cohen et al. [39], which was shown to successfully predict implicit associations between words. Finally, the TRRI feature for a candidate code c is defined as

$$f^R(c, E) = \sum_{t \in \mathcal{N}(E)} \mathcal{R}(t, c),$$

where $\mathcal{R}(t, c)$ is the TRRI index based similarity score of t with c. We used the semantic vectors package [40] to construct the MeSH term vectors using TRRI. Both the co-occurrence and TRRI based features when combined helped us obtain improved performance in MeSH term prediction for indexing biomedical citations over using only one of them or neither of them. Hence we also chose to conduct experiments with both these features for our current task of code assignment.

4.7. Predicting Number of Labels

The conventional approach in binary classification using an LR model is to predict a positive response if the value of the logistic function (whose input is a function of the input variables) is greater than 0.5. However, in the BR model, it is possible to have testing instances where too many or too few labels are predicted with the 0.5 threshold when compared with the correct number of labels. Thus in multi-label classification, it is also important to consider the effect of the number of labels predicted per document on the performance of the approaches used. A quick fix that many employ is to pick a threshold of top r (r-cut) labels (when ranked either using just the LR classifier scores or using more sophisticated ranking approaches discussed in Section 4.6) where the r picked is the one that maximizes the example-based F-score on the training data. However, using this method always results in the same number of labels for each document in the testing set. We used an advanced thresholding method, Multi Label Probabilistic Threshold Optimizer [20] (MLPTO), for choosing a different number of labels per EMR that changes for each EMR instance. The optimizer we employed uses $1 -$ (Example-Based-F-score) as the loss function and finds the r that minimizes the expected loss function across all possible example-based contingency tables for each instance. For specific details of this strategy, please see [20]. This strategy is computationally expensive and hence we also experimented with a simpler approach of predicting the number of labels based on linear regression. The idea is to use a validation dataset where the numerical output is the correct number of labels for the EMRs. We used two numerical features: 1. the number of labels for the EMR for which the corresponding binary classifiers output a probability at least 0.1; 2. the number of ICD-9 codes identified in the EMR text by named entity recognition (NER) using MetaMap. We also tried other features including the size of the EMR (numbers of documents and words in the EMR) which did not seem to predict well on the validation dataset.

5. Evaluation Measures

Before we discuss our methods and results, we establish notation to be used for evaluation measures. Since the task of assigning multiple codes to an EMR is the multi-label classification problem, there are multiple complementary methods [41] for evaluating automatic approaches for this task. Recall that $\mathbf{Y}_i$, $i = 1, \ldots, m$, is the set of correct labels (here codes) in the dataset for the i-th EMR, where m is the total number of EMRs. Let $\mathbf{Z}_i$ be the set of predicted labels for the i-th EMR. The example-based precision, recall, and F-score are defined as

$$P_{ex} = \frac{1}{m}\sum_{i=1}^{m}\frac{|\mathbf{Y}_i \cap \mathbf{Z}_i|}{|\mathbf{Z}_i|},\ R_{ex} = \frac{1}{m}\sum_{i=1}^{m}\frac{|\mathbf{Y}_i \cap \mathbf{Z}_i|}{|\mathbf{Y}_i|},$$

$$\text{and } F_{ex} = \frac{1}{m}\sum_{i=1}^{m}\frac{2|\mathbf{Y}_i \cap \mathbf{Z}_i|}{|\mathbf{Z}_i| + |\mathbf{Y}_i|},\ \text{respectively.}$$

For each label T_j in the set of labels T being considered, we have label-based precision $P(T_j)$, recall $R(T_j)$, and F-score $F(T_j)$ defined as

$$P(T_j) = \frac{TP_j}{TP_j + FP_j},\ R(T_j) = \frac{TP_j}{TP_j + FN_j},$$

$$\text{and } F(T_j) = \frac{2P(T_j)R(T_j)}{P(T_j) + R(T_j)},$$

where TP_j, FP_j, and FN_j are true positives, false positives, and false negatives, respectively, of label T_j. Given this, the label-based macro average F-score is

$$\text{Macro-F} = \frac{1}{|T|}\sum_{j=1}^{|T|} F(T_j).$$

The label-based micro precision, recall, and F-score are defined as

$$P^{mic} = \frac{\sum_{j=1}^{|T|} TP_j}{\sum_{j=1}^{|T|}(TP_j + FP_j)},\ R^{mic} = \frac{\sum_{j=1}^{|T|} TP_j}{\sum_{j=1}^{|T|}(TP_j + FN_j)},$$

$$\text{and}\quad \text{Micro-F} = \frac{2P^{mic} \cdot R^{mic}}{P^{mic} + R^{mic}},$$

While the macro measures consider all labels as equally important, micro measures tend to give more importance to labels that are more frequent. This is relevant for our dataset because we have a very unbalanced set of label counts and in such cases micro measures are considered more important.

6. Experiments and Results

We note that not every component of the automated assignment framework outlined in Section 4 is suitable for all datasets as will be observed in our results. For clarity we do not present performance measures for all possible combinations we experimented with for each dataset, but only show an appropriate subset especially the including top performing configuration. We will first present our results for the CMC dataset.

6.1. CMC Dataset Results

From the 1954 reports in the CMC dataset, 978 are included in the training dataset with their corresponding ICD-9 codes; the remaining documents form the testing set. All labels sets that occur in the testing set occur at least once in the training dataset. The best results on the CMC dataset were obtained using unigrams and named entity counts (without any weighting) as features and SVMs as the base classifiers. Here, we present the results when using BR and ensemble of classifier chains (ECC). We also used SVMs with bagging to compare BR with the more complex ensemble approaches, ECC and EPS [18], that take label dependencies into account. In Table 2, we only show the basic BR configuration and the best performer ECC. For the CMC competition, the micro F-score was the measure used for comparing relative performances of contestants. The mean score in the competition was 0.77 with a standard deviation of 0.13. The best performing method was able to achieve a 0.90 micro F-score.

Table 2: CMC test set scores

Learner	Example-Based			Micro			Macro		
	P	R	F	P	R	F	P	R	F
$BR + SVM$	0.82	0.82	0.81	0.86	0.81	0.83	0.54	0.48	0.49
$ECC + SVM$	0.84	0.84	0.83	0.88	0.82	0.85	0.54	0.44	0.47

There were many instances in the CMC dataset where our methods did not predict any codes. We experimented with using an unsupervised approach to generate predictions for those examples: we generated named entities using MetaMap for each of these documents that did not have any predictions. We mapped these entities to ICD-9-CM codes via a knowledge-based mapping approach [42] that exploits the graph of relationships in the UMLS Metathesaurus (which also includes ICD-9-CM). If MetaMap generated a concept that got mapped to an ICD-9 code we trained on, we used that ICD-9 code as our prediction. We were able to increase our best F-Score from using ECC from 0.85 to 0.86 using this method. Also, we don't report the results when including semantic predications as features because the results were comparable with no major improvements. Feature selection, optimal training sets, and probabilistic thresholding did not make significant improvements for the CMC dataset.

6.2. UKSmall Dataset Results

In the UKSmall dataset of 1000 EMRs, we removed those that are not coded with at least one of the 56 codes that had at least 20 examples. That left us with 827 EMRs out of which 100 were used for testing and the remaining for training examples. Table 3 shows the results on the testing set for this dataset.

In our experiments, we first tried our best performing combination from the CMC dataset and noticed that ECC did not perform well on this larger dataset; there seem to have been very few label dependencies in this dataset – recall that there were over 500 unique label sets of the 56 labels used for training in UKSmall. Also, on this dataset we achieved the best

Table 3: UKSmall test set scores (with LR as base classifier and BR as transformation)

Learner, Ranker, and Calibrator	Example-Based			Micro			Macro		
	P	R	F	P	R	F	P	R	F
BNS	0.38	0.21	0.24	0.64	0.16	0.25	0.21	0.13	0.15
$BNS + OTS + RCut = 3$	0.40	**0.49**	0.38	0.40	**0.41**	0.41	0.34	**0.31**	0.29
Best : $BNS + OTS + MLPTO$	**0.59**	0.42	**0.45**	**0.54**	0.37	**0.44**	**0.40**	0.29	**0.32**
$Best - OTS$	0.60	0.36	0.40	0.57	0.29	0.39	0.42	0.25	0.29
$Best - MLPTO$	0.52	0.34	0.37	0.61	0.32	0.42	0.43	0.26	0.30
$Best - BNS$	0.30	0.11	0.15	0.31	0.10	0.15	0.05	0.04	0.04

results using LR instead of an SVM as our base classifier. Because there was much more text available per example, we were able to take advantage of tf-idf weighting and used bigrams, unigrams, and named entities. For all models in Table 3 with the exception of the last row model, we used BR for the transformation, LR as the base classifier, and BNS for feature selection. In the second row we show the results of additionally using OTS (Section 4.5) to handle class imbalance in the binary classifiers and always choosing the top 3 labels for label calibration (RCut=3). RCut is determined based on the best number of labels to pick that maximizes example-based F-score on the training dataset. In our experiments we observed it to be close to the label cardinality (2.8 for this dataset). Since these methods use RCut 3, that is, three labels are always picked for each EMR, we can see how the recall gain also introduced many false positives. As seen in the next row, this issue is rectified by using MLPTO (Section 4.7) instead of Rcut and picking a different number of labels per EMR. Finally, we can see from the first row, where only BNS is used without any other learning components, the performance is significantly compared to the best results in the third row.

In rows 4–6 of Table 3, we also show the results of learning component ablation on our best classifier. Each of these rows show the effect on performance if one of the learning components is not used. While removal of MLPTO and OTS caused losses of up to 8% in F-scores, dropping feature selection with BNS caused a drop of 30% in F-score, which clearly demonstrates the importance of appropriate feature selection in these combinations. However, interestingly, we can see that just using BNS alone without MLPTO and OTS did not result in major performance gains. Next, we present our results on the UKLarge dataset.

6.3. UKLarge Dataset Results

From a total of 71,463 EMRs from the 2011–2012 two year period, we randomly selected 2000 EMRs for validation and 3000 EMRs for testing, and all other records were used for training. Unlike for the smaller datasets, we needed a validation dataset for certain components that gave us the best performance. We used unigrams and bigrams with tf-idf weighting as features after ignoring those features that did not occur in at least 10 EMRs in the training dataset. Most components that performed well on the CMC and UKSmall datasets were either not practical or did not show any performance improvements. Feature selection using BNS and careful selection of negative examples to handle class imbalance using OTS did not improve the performance. This dataset has 1231 labels and over 60,000 unique combinations of these labels, both at least an order of magnitude larger than the corresponding counts in the smaller datasets. Hence the ECC transformation approach that relies on building multiple classier chains turned out to be impractical. The implementation of MLPTO that calibrates the appropriate number of labels specific to each testing instance also did not scale to this larger training dataset. Thus we used a different simpler strategy of estimating the correct number of labels to be coded for an EMR using linear regression based on the number of predicted codes that exceed a LR classifier score threshold of 0.1 and the number of codes identified just with NER. We also used learning-to-rank (Section 4.6) to rerank the preliminary rankings from LR classifiers for all codes before choosing the predicted number of codes from the linear regression function. We also noticed that for 140 codes (out of 1231), a simple classifier that predicts a code if it can be extracted using NER on the EMR text does better than LR classifiers. So after choosing the predicted number of labels from the linear regression model, we added these NER classifier (NERC) based codes to the predicted code set for each test set EMR (we chose to trade-off some precision for recall in this constrained fashion). The linear regression function, the learning-to-rank (L2R) ranking function, and the identification of codes where NERC outperforms LR is all done over the validation set of 2000 EMRs that is not part of the training or testing datasets.

Table 4: UKLarge test set scores (with BR as transformation)

Calibrator	Learner and Ranker	Example-Based			Micro			Macro		
		P	R	F	P	R	F	P	R	F
RCut=8	LR	0.434	0.489	0.399	0.434	0.387	0.410	0.824	0.181	0.176
	$LR + L2R$	0.443	0.501	0.408	0.443	0.395	0.417	0.799	0.190	0.189
	$LR + L2R + NERC$	0.440	0.511	0.414	0.444	0.409	0.426	0.779	0.224	**0.211**
Linear Regression	LR	0.482	0.461	0.442	0.513	0.421	0.463	0.859	0.176	0.180
	$LR + L2R$	0.491	0.469	0.450	0.524	0.430	0.472	0.844	0.184	0.188
	LR + L2R + NERC	0.490	0.480	**0.455**	0.522	0.443	**0.479**	0.822	0.218	**0.211**
Correct Count	LR	0.484	0.481	0.481	0.482	0.482	0.482	0.835	0.197	0.198
	$LR + L2R$	0.486	0.486	0.486	0.495	0.495	0.495	0.821	0.211	0.212
	$LR + L2R + NERC$	0.485	0.496	0.490	0.493	0.505	0.499	0.801	0.241	0.230

The results of our experiments on the UKLarge dataset are presented in Table 4. We show up to three digits after the decimal for these scores since the differences are not as substantial as in the case of the smaller datasets. The first three rows of the table show the evaluation scores if the calibrator was simply selecting the top 8 (label cardinality is 7.4) codes. Rows 4–6 contain results when we apply our regression function as the calibrator for number of labels. Our best performer is row 6 indicated in bold font, which has the best example-based, micro, and macro F-scores when applying both L2R and NERC. Since the linear regression model does not perfectly predict (our r^2 was 0.7) the correct number of codes for an EMR, we obtained scores assuming a perfect calibrator, that is, assuming we know the correct number of labels for each EMR. These scores are shown in rows 7–9 of the table.

Regardless of the calibrator, we see that using L2R and NERC always improves all three F-scores. L2R and NERC lead to a 1.6 point (or a 3.4% relative improvement) in the micro F-score although having a better calibrator could add an additional 2 point improvement. The values in columns for example-based and micro F-scores are in ascending order indicating gains with calibrators and with additional learning components for a fixed calibrator. In particular, our best micro F-score in row 6 is still slightly lower than the corresponding value in row 7, where LR based ranking with a perfect calibrator is used. An accurate calibrator is a clearly a crucial component here that warrants further exploration. Example-based and micro F-score are close to each other, but the macro F-score is substantially lower owing to the very low code level recall for several codes with very few training examples.

7. Discussion

In Section 4, depending on the scale and characteristics of the datasets, we observed that some methods perform better over the others and as is usually the case there is no sliver bullet for assigning diagnosis codes. It is important to note that although more sophisticated methods did not improve the results, they did not worsen the scores in general. While more training data usually leads to better predictive power, efficient counterparts of techniques that work extremely well on smaller datasets are essential. In this section, we contrast the results on the three datasets and further analyze the UKLarge dataset results as that dataset provides a large representation of EMRs.

7.1. CMC Vs UKSmall

The CMC and the UKSmall datasets share similarities in the number of codes. However, the number of unique code combinations is 10 times the number of codes in UKSmall and it is twice the number of codes in the CMC dataset. We believe this is because each EMR in the CMC dataset is essentially a short radiology report dealing with chest x-ray or renal procedures and hence codes have high correlation. This could be the reason why the classifier chain approach worked well for the CMC dataset but not for the UKSmall dataset. The UKSmall EMR size is two orders of magnitude larger than the CMC reports. Hence feature selection, training data selection, and label calibration that helped with UKSmall predictions did not improve for the CMC predictions as the corresponding number of text features and label cardinality were much smaller. However, the scores were much better on the CMC dataset due to the more focused and simpler nature of the reports.

7.2. UKSmall Vs UKLarge

The UKLarge and UKSmall datasets differ substantially with the former having 21 times more codes to predict than the latter. There are fewer examples to learn from in the UKSmall dataset. However, the class imbalance is much higher in UKLarge since there are several codes each of which accounts for only 0.07% of the dataset, while in UKSmall each of the most infrequent codes constitute at least 2% of the dataset. Thus UKLarge had better micro recall and slightly lower micro precision compared with UKSmall. However, the macro recall for UKLarge is much worse than the macro recall for UKSmall because most of the recall gains for UKLarge come from a small set of high frequency codes (so have a large number of training examples). The ones that have very few examples form a large set and suffer significant recall loss leading to an overall low macro recall for UKLarge. To examine this, we took the best results we had and aggregated the scores for code subsets that constitute at least 2% or 1% of the entire dataset. However, the minimum requirement of 50 EMRs per code leads to codes with as low as 0.07% representation in the dataset. So we also aggregated scores for code subsets with at most 0.1% and 0.5% representation in UKLarge. The results are shown in Table 5. As we can see the micro recall for those codes with at least 2% representation in the dataset is 0.699, but the micro recall for the 874 codes (so 80% of the 1231 code set) with at most 0.5% representation is only 0.124. For the set of 250 codes each of which have $\leq 0.1\%$ representation, the micro recall further comes down to 0.05. Thus we can see that most of the recall gains come from the top few codes. Similar major difference can also be seen in macro recall. The unusually high example-based recall in the fourth row of the table is because of the absence of very low frequency codes in most EMRs.

Table 5: UKLarge test set scores for code subsets

Representation in UKLarge	Example-Based			Micro			Macro		
	P	R	F	P	R	F	P	R	F
$\geq$ 2% (92 codes)	0.498	0.699	0.520	0.531	0.634	0.578	0.547	0.533	0.513
$\geq$ 1% (190 codes)	0.493	0.622	0.500	0.524	0.557	0.540	0.563	0.404	0.395
$\leq$ 0.5% (874 codes)	0.870	0.387	0.361	0.536	0.124	0.202	0.912	0.167	0.161
$\leq$ 0.1% (250 codes)	0.984	0.844	0.831	0.367	0.052	0.091	0.959	0.187	0.167

Another difference is that UKSmall did not have all physician authored documents because we removed less relevant documents such as flow sheets, vital signs sheets, discharge instructions. This led to UKLarge EMRs having on an average over 2000 more words. Upon examination, we found out that some of these less relevant documents also contain important information albeit that should already be present in more important documents such as discharge summaries and progress notes. For example, a discharge instruction note mostly talks about doctors suggestions to the patient about when and how to follow-up, diet/exercise, and other recommendations. However, it also has information on what the patient was treated for and which conditions were resolved, evidence relevant to diagnosis coding. Although, this information is also recorded in discharge summaries and progress notes, having evidence in multiple notes (that may not be relevant for code assignment) may help the machine learning algorithms. Similarly, named entity recognition tools are better at extracting entities that are expressed in text in certain simple ways. If the condition is not expressed in an NER-friendly way in the discharge summary, it may be expressed in a concise NER-friendly format in a different not so relevant note.

7.3. Recall and Precision Errors in UKLarge

We also analyzed codes that have a substantial representation in UKLarge but suffered major recall or precision errors. We first ranked the top 100 frequent codes in UKLarge in the descending order of their precision and looked at the bottom ten in this list. All these codes had precision between 0.2 and 0.3. Six of these ten are the symptom codes (category level `780--799`) which are generally not coded as per the guidelines if a cause is ascertained during the course of the hospital stay in which case the symptom causing condition gets coded. In UKLarge, symptom codes account for only 6% of all diagnoses. However, symptoms are discussed for obvious reasons in the EMRs as they need to document patient's progress. The six symptoms we found here are: dyspnea and respiratory abnormalities, other symptoms involving digestive system, abdominal pain, tachycardia (unspecified), other general symptoms, fever and other physiologic disturbances of temperature regulation. As we can see, they are either generic or vague but probably discussed often in the EMRs. Two other low precision codes that are not symptom codes are anaemia (unspecified) and backache (unspecified). These are also generic conditions for which more specific codes exist.

Next we ranked the top 100 frequent codes in the descending order of their recall and identified the last ten in this list. For four of these codes, once we examined the corresponding predicted code sets, we noticed that in over 25% of recall errors, a sibling code (at the fourth digit level) of the correct code was predicted. Although, this is not desirable but it is a plausible explanation due to the similarities of sibling code conditions. One of the examples was the code `427.8` (other specified cardiac disrhythmias). In this particular case, 40 times out of the 121 times `427.8` was missed, we also predicted a code with prefix `427` (the general category of cardiac disrhythmias), which then also counts toward a precision error for example-based measures. Two other low recall codes are abnormal glucose and hypotension, both generally extracted based on numerical measures reported in the EMR text, situations where textual features are not useful.

7.4. Machine Learning Vs NER in UKLarge

Recall from Section 6.3 that for 140 out of the 1231 codes, a classifier based on whether the code was extracted using MetaMap NER tool outperformed machine learning based classifier. We found these 140 codes based on our findings on the validation dataset which has 2000 EMRs. Since very low frequency codes have extremely small representation in these 2000 EMRs, we wanted to observe the degree of difference in performance based on F-scores, for codes with at least 2% representation in the full dataset. We found at least ten such codes where the F-score was different by more than 10 points. These ten codes represent half of the 140 codes that also have at least 2% representation in the dataset.

8. Conclusion

In this paper, we used supervised multi-label text classification approaches to assign ICD-9-CM diagnosis codes to EMRs from three datasets that differ in scale with regards to number of codes and average size of the EMRs. Using a combination of problem transformation approaches with feature selection, training data selection, label calibration, and learning-to-rank, we compared our results with the basic approaches to assess the contribution of these additional learning components in the task of diagnosis code assignment.

We first showed that on a gold standard dataset with short reports, classifier chains result in comparable performance to the state of the art. We experiment with a big dataset, UKLarge, of EMRs of in-patients discharged at the University of Kentucky during the two year period 2011–2012. At the time of this writing, this is the first study in diagnosis code assignment on a generalized and large set of EMRs. First we curated a small subset of 1000 records of this dataset with label set size comparable to the CMC dataset but with larger EMR size and fewer code correlations. On this dataset, UKSmall, we show the feature selection, data selection, and label calibration provide significant gains in performance. However, feature selection and data selection do not improve performance in experiments with the larger dataset. We used a validation dataset in this case and applied a combination of NER based and machine learning based classifiers with a reranking approach based on classifier scores and output code correlations captured through code co-occurrences and implicit connections.

In our experiments on codes with at least 2% representation in corresponding datasets, in UKSmall we achieve a micro F-score of 0.44 and for UKLarge we achieve an F-score of 0.57. Over all the 1231 codes even with 80% of the codes have $\leq 0.5\%$ representation in UKLarge, we obtain a micro F-score of 0.47. However, most of gains in F-score come from a high recall for a selected few high frequency codes. Hence, the macro F-score of UKSmall is 0.32, which is 11 points more than that for UKLarge. These experiments make it clear that a larger dataset improves micro scores over larger code sets even if most codes have very few examples. However, better named entity recognition and more accurate approaches for label calibration are required to make further progress in automated code assignment especially for those labels with very few training examples compared with the size of the entire dataset. We will continue to explore these opportunities in our ongoing research efforts in the general area of computational code prediction.

Acknowledgements

Many thanks to anonymous reviewers for their detailed comments and suggestions that helped improve the content and presentation of this paper. This publication was supported by the National Center for Research Resources and the National Center for Advancing Translational Sciences, US National Institutes of Health (NIH), through Grant UL1TR000117. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

[1] Centers for Medicare and Medicaid Services, ICD-10 implementation guide, `https://implementc10.noblis.org/understand_comparison/`, Accessed: April 8, 2015.

[2] National Center for Health Statistics and the Centers for Medicare and Medicaid Services, International classification of diseases, ninth revision, clinical modification (ICD-9-CM), `http://www.cdc.gov/nchs/icd/icd9cm.htm`, Accessed: April 8, 2015.

[3] J. P. Pestian, C. Brew, P. Matykiewicz, D. J. Hovermale, N. Johnson, K. B. Cohen, W. Duch, A shared task involving multi-label classification of clinical free text, in: Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing, Association for Computational Linguistics, 2007, pp. 97–104.

[4] L. R. S. de Lima, A. H. F. Laender, B. A. Ribeiro-Neto, A hierarchical approach to the automatic categorization of medical documents, in: Proceedings of the 7th International Conference on Information Management, CIKM, ACM, 1998, pp. 132–139.

[5] M. L. Gundersen, P. J. Haug, T. A. Pryor, R. van Bree, S. Koehler, K. Bauer, B. Clemons, Development and evaluation of a computerized admission diagnosis encoding system, Computers and Biomedical Research 29 (5) (1996) 351–372.

[6] A. R. Aronson, O. Bodenreider, D. Demner-Fushman, K. W. Fung, V. K. Lee, J. G. Mork, A. Névéol, L. Peters, W. J. Rogers, From indexing the biomedical literature to coding clinical text: experience with MTI and machine learning approaches, in: Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing, Association for Computational Linguistics, 2007, pp. 105–112.

[7] I. Goldstein, A. Arzumtsyan, O. Uzuner, Three approaches to automatic assignment of ICD-9-CM codes to radiology reports, in: Proceedings of AMIA Symposium, American Medical Informatics Association, 2007, pp. 279–283.

[8] K. Crammer, M. Dredze, K. Ganchev, P. Pratim Talukdar, S. Carroll, Automatic code assignment to medical text, in: Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing, BioNLP, 2007, pp. 129–136.

[9] R. Farkas, G. Szarvas, Automatic construction of rule-based ICD-9-CM coding systems, BMC Bioinformatics 9 (Suppl 3) (2008) S10.

[10] A. Névéol, S. E. Shooshan, S. M. Humphrey, J. G. Mork, A. R. Aronson, A recent advance in the automatic indexing of the biomedical literature, Journal of biomedical informatics 42 (5) (2009) 814–823.

[11] Y. Zhang, A hierarchical approach to encoding medical concepts for clinical notes, in: Proceedings of the ACL-08: HLT Student Research Workshop, Association for Computational Linguistics, Columbus, Ohio, 2008, pp. 67–72.

[12] J. Ferrao, F. Janela, H. Martins, Using structured EHR data and SVM to support ICD-9-CM coding, in: Healthcare Informatics (ICHI), 2013 IEEE International Conference on, IEEE, 2013, pp. 511–516.

[13] P. Ruch, J. Gobeill, I. Tbahriti, A. Geissbühler, From episodes of care to diagnosis codes: automatic text categorization for medico-economic encoding, in: AMIA Annual Symposium Proceedings, Vol. 2008, American Medical Informatics Association, p. 636.

[14] A. Perotte, R. Pivovarov, K. Natarajan, N. Weiskopf, F. Wood, N. Elhadad, Diagnosis code assignment: models and evaluation metrics, Journal of the American Medical Informatics Association (2013) 231–237.

[15] A. C. P. L. F. de Carvalho, A. A. Freitas, A tutorial on multi-label classification techniques, in: Foundations of Computational Intelligence (5), Vol. 205, 2009, pp. 177–195.

[16] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Machine Learning 85 (3) (2011) 335–359.

[17] M.-L. Zhang, K. Zhang, Multi-label learning by exploiting label dependency, in: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '10, ACM, 2010, pp. 999–1008.

[18] J. Read, B. Pfahringer, G. Holmes, Multi-label classification using ensembles of pruned sets, in: Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on, 2008, pp. 995 –1000.

[19] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, K. Brinker, Multilabel classification via calibrated label ranking, Machine Learning 73 (2) (2008) 133–153.

[20] J. Quevedo, O. Luaces, A. Bahamonde, Multilabel classifiers with a probabilistic thresholding strategy, Pattern Recognition 45 (2) (2012) 876–883.

[21] G. Forman, An extensive empirical study of feature selection metrics for text classification, Journal of Machine Learning Research 3 (2003) 1289–1305.

[22] J. S. Olsson, Combining feature selectors for text classification, in: Proceedings of the 15th ACM international conference on Information and knowledge management, ACM, 2006, pp. 798–799.

[23] H. He, E. A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering 21 (9) (2009) 1263–1284.

[24] S. Sohn, W. Kim, D. C. Comeau, W. J. Wilbur, Optimal training sets for bayesian prediction of mesh term assignment., Journal of the American Medical Informatics Association 15 (4) (2008) 546–553.

[25] A. Rios, R. Kavuluru, Supervised extraction of diagnosis codes from EMRs: Role of feature selection, data selection, and probabilistic thresholding, in: Healthcare Informatics (ICHI), 2013 IEEE International Conference on, IEEE, 2013, pp. 66–73.

[26] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.

[27] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: A java library for multi-label learning, Journal of Machine Learning Research 12 (2011) 2411–2414.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[29] P. D. Turney, P. Pantel, et al., From frequency to meaning: Vector space models of semantics, Journal of Artificial Intelligence Research 37 (1) (2010) 141–188.

[30] A. R. Aronson, F.-M. Lang, An overview of metamap: historical perspective and recent advances, Journal of the American Medical Informatics Association 17 (3) (2010) 229–236.

[31] T. C. Rindflesch, M. Fiszman, The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text, Journal of Biomedical Informatics 36 (6) (2003) 462–477.

[32] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, Liblinear: A library for large linear classification, Journal of Machine Learning Research 9 (2008) 1871–1874.

[33] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm` (Accessed: April 8, 2015).

[34] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.

[35] T.-Y. Liu, Learning to rank for information retrieval, Foundations and Trends in Information Retrieval 3 (3) (2009) 225–331.

[36] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of IR techniques, ACM Transactions on Information Systems 20 (4) (2002) 422–446.

[37] R. Kavuluru, Z. He, Unsupervised medical subject heading assignment using output label co-occurrence statistics and semantic predications, in: Natural Language Processing and Information Systems, NLDB, Springer, 2013, pp. 176–188.

[38] T. Cohen, D. Widdows, Empirical distributional semantics: Methods and biomedical applications, Journal of Biomedical Informatics 42 (2) (2009) 390–405.

[39] T. Cohen, R. Schvaneveldt, D. Widdows, Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections, Journal of Biomedical Informatics 43 (2) (2010) 240–256.

[40] D. Widdows, T. Cohen, The semantic vectors package: New algorithms and public tools for distributional semantics, in: Semantic Computing (ICSC), 2010 IEEE Fourth International Conference on, IEEE, 2010, pp. 9–15.

[41] G. Tsoumakas, I. Katakis, I. P. Vlahavas, Mining multi-label data, in: Data Mining and Knowledge Discovery Handbook, 2010, pp. 667–685.

[42] O. Bodenreider, S. Nelson, W. Hole, H. Chang, Beyond synonymy: exploiting the UMLS semantics in mapping vocabularies, in: Proceedings of AMIA Symposium, American Medical Informatics Association, 1998, pp. 815–819.