A System for Predicting ICD-10-PCS Codes from Electronic Health Records

Michael Subotin
3M Health Information Systems
Silver Spring, MD
msubotin@mmm.com

Anthony R. Davis
3M Health Information Systems
Silver Spring, MD
adavis4@mmm.com

Abstract

Medical coding is a process of classifying health records according to standard code sets representing procedures and diagnoses. It is an integral part of health care in the U.S., and the high costs it incurs have prompted adoption of natural language processing techniques for automatic generation of these codes from the clinical narrative contained in electronic health records. The need for effective auto-coding methods becomes even greater with the impending adoption of ICD-10, a code inventory of greater complexity than the currently used code sets. This paper presents a system that predicts ICD-10 procedure codes from the clinical narrative using several levels of abstraction. First, partial hierarchical classification is used to identify potentially relevant concepts and codes. Then, for each of these concepts we estimate the confidence that it appears in a procedure code for that document. Finally, confidence values for the candidate codes are estimated using features derived from concept confidence scores. The concept models can be trained on data with ICD-9 codes to supplement sparse ICD-10 training resources. Evaluation on held-out data shows promising results.

1 Introduction

In many countries reimbursement rules for health care services stipulate that the patient encounter must be assigned codes representing diagnoses that were made for and procedures that were performed on the patient. These codes may be assigned by general health care personnel or by specially trained medical coders. The billing codes used in the U.S. include International Statistical Classification of Diseases and Related Health Problems (ICD) codes, whose version 9 is currently in use and whose version 10 was scheduled for adoption in October 2014[1], as well as Current Procedural Terminology (CPT) codes. The same codes are also used for research, internal bookkeeping, and other purposes.

Assigning codes to clinical documentation often requires extensive technical training and involves substantial labor costs. This, together with increasing prominence of electronic health records (EHRs), has prompted development and adoption of NLP algorithms that support the coding workflow by automatically inferring appropriate codes from the clinical narrative and other information contained in the EHR (Chute et al., 1994; Heinze et al., 2001; Resnik et al., 2006; Pakhomov et al., 2006; Benson, 2006). The need for effective auto-coding methods becomes especially acute with the introduction of ICD-10 and the associated increase of training and labor costs for manual coding.

The novelty and complexity of ICD-10 presents unprecedented challenges for developers of rule-based auto-coding software. Thus, while ICD-9 contains 3882 codes for procedures, the number of codes defined by the ICD-10 Procedure Coding System (PCS) is greater than 70,000. Furthermore, the organization of ICD-10-PCS is fundamentally different from ICD-9, which means that the investment of time and money that had gone into writing auto-coding rules for ICD-9 procedure codes cannot be easily leveraged in the transition to ICD-10.

In turn, statistical auto-coding methods are constrained by the scarcity of available training data with manually assigned ICD-10 codes. While this problem will be attenuated over the years as ICD-10-coded data are accumulated, the health care

[1]The deadline was delayed by at least a year while this paper was in review.

industry needs effective technology for ICD-10 computer-assisted coding in advance of the implementation deadline. Thus, for developers of statistical auto-coding algorithms two desiderata come to the fore: these algorithms should take advantage of all available training data, including documents supplied only with ICD-9 codes, and they should possess high capacity for statistical generalization in order to maximize the benefits of training material with ICD-10 codes.

The auto-coding system described here seeks to meet both these requirements. Rather than predicting codes directly from the clinical narrative, a set of classifiers is first applied to identify coding-related concepts that appear in the EHR. We use General Equivalence Mappings (GEMs) between ICD-9 and ICD-10 codes (CMS, 2014) to train these models not only on data with human-assigned ICD-10 codes, but also on ICD-9-coded data. We then use the predicted concepts to derive features for a model that estimates probability of ICD-10 codes. Besides the intermediate abstraction to concepts, the code confidence model itself is also designed so as to counteract sparsity of the training data. Rather than train a separate classifier for each code, we use a single model whose features can generalize beyond individual codes. Partial hierarchical classification is used for greater run-time efficiency. To our knowledge, this is the first research publication describing an auto-coding system for ICD-10-PCS. It is currently deployed, in tandem with other auto-coding modules, to support computer-assisted coding in the 3M^{TM}360 Encompass^{TM}System.

The rest of the paper is organized as follows. Section 2 reviews the overall organization of ICD-10-PCS. Section 4.1 outlines the run-time processing flow of the system to show how its components fit together. Section 4.2 describes the concept confidence models, including the hierarchical classification components. Section 4.3 discusses how data with manually assigned ICD-9 codes is used to train some of the concept confidence models. Section 4.4 describes the code confidence model. Finally, Section 5 reports experimental results.

2 ICD-10 Procedure Coding System

ICD-10-PCS is a set of codes for medical procedures, developed by 3M Health Information Systems under contract to the Center for Medicare and Medicaid Services of the U.S. government. ICD-10-PCS has been designed systematically; each code consists of seven characters, and the character in each of these positions signifies one particular aspect of the code. The first character designates the “section” of ICD-10-PCS: 0 for Medical and Surgical, 1 for Obstetrics, 2 for Placement, and so on. Within each section, the seven components, or axes of classification, are intended to have a consistent meaning; for example in the Medical and Surgical section, the second character designates the body system involved, the third the root operation, and so on (see Table 1 for a list). All procedures in this section are thus classified along these axes. For instance, in a code such as *0DBJ3ZZ*, the *D* in the second position indicates that the body system involved is the gastrointestinal system, *B* in the third position always indicates that the root operation is an excision of a body part, the *J* in the fourth position indicates that the appendix is the body part involved, and the *3* in the fifth position indicates that the approach is percutaneous. The value *Z* in the last two axes means than neither a device nor a qualifier are specified.

Character	Meaning
1st	Section
2nd	Body System
3rd	Root Operation
4th	Body Part
5th	Approach
6th	Device
7th	Qualifier

Table 1: Character Specification of the Medical and Surgical Section of ICD-10-PCS

Several consequences of the compositional structure of ICD-10-PCS are especially relevant for statistical auto-coding methods.

On the one hand, it defines over 70,000 codes, many of which are logically possible, but very rare in practice. Thus, attempts to predict the codes as unitary entities are bound to suffer from data sparsity problems even with a large training corpus. Furthermore, some of the axis values are formulated in ways that are different from how the corresponding concepts would normally be expressed in a clinical narrative. For example, ICD-10-PCS uses multiple axes (root opreration, body part, and, in a sense, the first two axes as well) to encode what many traditional procedure terms (such as those ending in *-tomy* and *-plasty*) express by a single word, while the device axis uses generic categories where a clinical narrative would refer only to specific brand names. This drastically limits how much can be accomplished by matching code descriptions or indexes derived from them against the text of EHRs.

On the other hand, the systematic conceptual structure of PCS codes and of the codeset as a whole can be exploited to compensate for data sparsity and idiosyncracies of axis definitions by introducing abstraction into the model.

3 Related work

There exists a large literature on automatic classification of clinical text (Stanfill et al., 2010). A sizeable portion of it is devoted to detecting categories corresponding to billing codes, but most of these studies are limited to one or a handful of categories. This is in part because the use of patient records is subject to strict regulation. Thus, the corpus used for most auto-coding research up to date consists of about two thousand documents annotated with 45 ICD-9 codes (Pestian et al., 2007). It was used in a shared task at the 2007 BioNLP workshop and gave rise to papers studying a variety of rule-based and statistical methods, which are too numerous to list here.

We limit our attention to a smaller set of research publications describing identification of an entire set of billing codes, or a significant portion thereof, which better reflects the role of auto-coding in real-life applications. Mayo Clinic was among the earliest adopters of auto-coding (Chute et al., 1994), where it was deployed to assign codes from a customized and greatly expanded version of ICD-8, consisting of almost 30K diagnostic codes. A recently reported version of their system (Pakhomov et al., 2006) leverages a combination of example-based techniques and Naïve Bayes classification over a database of over 20M EHRs. The phrases representing the diagnoses have to be itemized as a list beforehand. In another pioneering study, Larkey & Croft (1995) investigated k-Nearest Neighbor, Naïve Bayes, and relevance feedback on a set of 12K discharge summaries, predicting ICD-9 codes. Heinze et al (2000) and Ribeiro-Neto et al (2001) describe systems centered on symbolic computation. Jiang et al (2006) discuss confidence assessment for ICD-9 and CPT codes, performed separately from code generation. Medori & Fairon (2010) combine information extraction with a Naïve Bayes classifier, working with a corpus of about 20K discharge summaries in French. In a recent paper, Perotte et al (2014) study standard and hierarchical classification using support vector machines on a corpus of about 20K EHRs with ICD-9 codes.

We are not aware of any previous publications on auto-coding for ICD-10-PCS, and the results of these studies cannot be directly compared with those reported below due to the unique nature of this code set. Our original contributions also include explicit modeling of concepts and the capability to assign previously unobserved codes within a machine learning framework.

4 Methods

4.1 Run-time processing flow

We first describe the basic run-time processing flow of the system, shown in Figure 1.

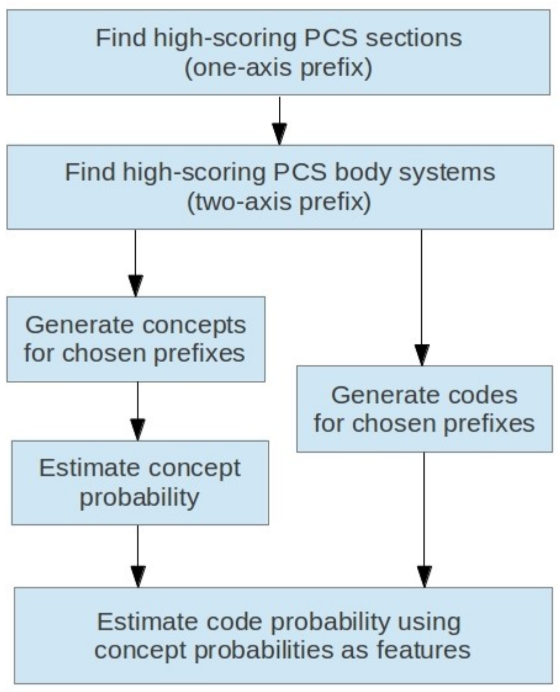
Find high-scoring PCS sections (one-axis prefix)
Find high-scoring PCS body systems (two-axis prefix)
Generate concepts for chosen prefixes
Generate codes for chosen prefixes
Estimate concept probability
Estimate code probability using concept probabilities as features

Figure 1: Run-time processing flow

In a naïve approach, one could generate all codes from the ICD-10-PCS inventory for each EHR[2] and estimate their probability in turn, but this would be too computationally expensive. Instead, the hypothesis space is restricted by two-

[2]We use the term EHR generically in this paper. The system can be applied at the level of individual clinical documents or entire patient encounters, whichever is appropriate for the given application.

level hierarchical classification with beam search. First, a set of classifiers estimates the confidence of all PCS sections (one-character prefixes of the codes), one per section. The sections whose confidence exceeds a threshold are used to generate candidate body systems (two-character code prefixes), whose confidence is estimated by another set of classifiers. Then, body systems whose confidence exceeds a threshold are used to generate a set of candidate codes and the set of concepts expressed by these codes. The probability of observing each of the candidate concepts in the EHR is estimated by a separate classifier. Finally, these concept confidence scores are used to derive features for a model that estimates the probability of observing each of the candidate codes, and the highest-scoring codes are chosen according to a thresholding decision rule.

The choice of two hierarchical layers is partially determined by the amount of training data with ICD-10 codes available for this study, since many three-character code prefixes are too infrequent to train reliable classifiers. Given more training data, additional hierarchical classification layers could be used, which would trade a higher risk of recall errors against greater processing speed. The same trade-off can be negotiated by adjusting the beam search threshold.

4.2 Concept confidence models

Estimation of concept confidence – including the confidence of code prefixes in the two hierarchical classification layers – is performed by a set of classifiers, one per concept, which are trained on EHRs supplied with ICD-10 and ICD-9 procedure codes.

The basis for training the concept models is provided by a mapping between codes and concepts expressed by the codes. For example, the code *0GB24ZZ* (Excision of Left Adrenal Gland, Percutaneous Endoscopic Approach) expresses, among other concepts, the concept *adrenal gland* and the more specific concept *left adrenal gland*. It also expresses the concept of *adrenalectomy* (surgical removal of one or both of the adrenal glands), which corresponds to the regular expression *0G[BT][234]..Z* over ICD-10-PCS codes. We used the code-to-concept mapping described in Mills (2013), supplemented by some additional categories that do not correspond to traditional clinical concepts. For example, our set of concepts included entries for the categories of *no device* and *no qualifer*, which are widely used in ICD-10-PCS. We also added entries that specified the device axis or the qualifier axis together with the first three axes, where they were absent in the original concept map, reasoning that the language used to express the choice of the device or qualifier can be specific to particular procedures and body parts.

For data with ICD-10-PCS codes, the logic used to generate training instances is straightforward. Whenever a manually assigned code expresses a given concept, a positive training instance for the corresponding classifier is generated. Negative training instances are sub-sampled from the concepts generated by hierarchical classification layers for that EHR. As can be seen from this logic, the precise question that the concept models seek to answer is as follows: given that this particular concept has been generated by the upstream hierarchical layers, how likely is it that it will be expressed by one of the ICD-10 procedure codes assigned to that EHR?

In estimating concept confidence we do not attempt to localize where in the clinical narrative the given concept is expressed. Our baseline feature set is simply a bag of tokens. We also experimented with other feature types, including frequency-based weighting schemes for token feature values and features based on string matches of Unified Medical Language System (UMLS) concept dictionaries. For the concepts of *left* and *right* we define an additional feature type, indicating whether the token *left* or *right* appears more frequently in the EHR. While still rudimentary, this feature type is more apt to infer laterality than a bag of tokens.

A number of statistical methods can be used to estimate concept confidence. We use the Mallet (McCallum, 2002) implementation of ℓ_1-regularized logistic regression, which has shown good performance for NLP tasks in terms of accuracy as well as scalability at training and run-time (Gao et al., 2007).

4.3 Training on ICD-9 data

In training concept confidence models on data with ICD-9 codes we make use of the General Equivalence Mappings (GEMs), a publicly available resource establishing relationships between ICD-9 and ICD-10 codes (CMS, 2014). Most correspondences between ICD-9 and ICD-10 procedure codes are one-to-many, although other mapping patterns are also found. Furthermore, a code in one set can correspond to a combination of codes from the other set. For example, the ICD-9 code for combined heart-lung transplantation maps to a set of pairs of ICD-10 codes, the first code in the pair representing one of three possible types of heart transplantation, and the other representing one of three possible types of bilateral lung transplantation.

A complete description of the rules underlying GEMs and our logic for processing them is beyond the scope of this paper, and we limit our discussion to the principles underlying our approach. We first distribute a unit probability mass over the ICD-10 codes or code combinations mapped to each ICD-9 code, using logic that reflects the structure of GEMs and distributing probability mass uniformly among comparable alternatives. From these probabilities we compute a cumulative probability mass for each concept appearing in the ICD-10 codes. For example, if an ICD-9 code maps to four ICD-10 codes over which we distribute a uniform probability distibution, and a given concept appears in two of them, we assign the probability of 0.5 to that concept. For a given EHR, we assign to each concept the highest probability it receives from any of the codes observed for the EHR. Finally, we use the resulting concept probabilities to weight positive training instances. Negative instances still have unit weights, since they correspond to concepts that can be unequivocably ruled out based on the GEMs.

4.4 Code confidence model

The code confidence model produces a confidence score for candidate codes generated by the hierarchical classification layers, using features derived from the output of the code confidence models described above. The code confidence model is trained on data with ICD-10 codes. Whenever a candidate code matches a code assigned by human annotators, a positive training instance is generated. Otherwise, a negative instance is generated, with sub-sampling. We report experiments using logistic regression with ℓ_1 and ℓ_2 regularization (Gao et al., 2007).

The definition of features used in the model requires careful attention, because it is in the form of the feature space that the proposed model differs from a standard one-vs-all approach. To elucidate the contrast we may start with a form of the feature space that would correspond to one-vs-all classification. This can be achieved by specifying the identity of a particular code in all feature names. Then, the objective function for logistic regression would decompose into independent learning subproblems, one for each code, producing a collection of one-vs-all classifiers. There are clear drawbacks to this approach. If all parameters are restricted to a specific code, the training data would be fragmented along the same lines. Thus, even if features derived from concepts may seem to enable generalization, in reality they would in each case be estimated only from training instances corresponding to a single code, causing unnecessary data sparsity.

This shortcoming can be overcome in logistic regression simply by introducing generalized features, without changing the rest of the model (Subotin, 2011). Thus, in deriving features from scores of concept confidence models we include only those concepts which are expressed by the given code, but we do not specify the identity of the code in the feature names. In this way the weights for these features are estimated at once from training instances for all codes in which these concepts appear. We combine these generalized features with the code-bound features described earlier. The latter should help us learn more specific predictors for particular procedures, when such predictors exist in the feature space.

While the scores of concept confidence models provide the basis for the feature space of the code confidence model, there are multiple ways in which features can be derived from these scores. The simplest way is to take concept identity (optionally specified by code identity) as the feature name and the confidence score as the feature value. We supplement these features with features based on score quantization. That is, we threshold each concept confidence score at several points and define binary features indicating whether the score exceeds each of the thresholds. For both these feature types, we generate separate features for predictions of concept models trained on ICD-9 data and concept models trained on ICD-10 data in order to allow the code confidence model to learn how useful predictions of concept confidence models are, depending on the type of their training data.

Both the concept confidence models and the code confidence model can be trained on data with ICD-10 codes. We are thus faced with the question of how best to use this limited resource. The simplest approach would be to train both types of models on all available training data, but there is a concern that predictions of the concept models on their own training data would not reflect their out-of-sample performance, and this would mislead the code confidence model into relying on them too much. An alternative approach, often called stacked generalization (Wolpert, 1992), would be to generate training data for the code confidence model by running concept confidence models on out-of-sample data. We compare the performance of these approaches below.

5 Evaluation

5.1 Methodology

We evaluated the proposed model using a corpus of 28,536 EHRs (individual clinical records), compiled to represent a wide variety of clinical contexts and supplied with ICD-10-PCS codes by trained medical coders. The corpus was annotated under the auspices of 3M Health Information Systems for the express purpose of developing auto-coding technology for ICD-10. There was a total of 51,082 PCS codes and 5,650 unique PCS codes in the corpus, only 76 of which appeared in more than 100 EHRs, and 2,609 of which appeared just once. Multiple coders worked on some of the documents, but they were allowed to collaborate, producing what was effectively a single set of codes for each EHR. We held out about a thousand EHRs for development testing and evaluation, each, using the rest for training. The same corpus, as well as 175,798 outpatient surgery EHRs with ICD-9 procedure codes submitted for billing by a health provider were also used to train hierarchical and concept confidence models.

We evaluated auto-coding performance by a modified version of mean reciprocal rank (MRR). MRR is a common evaluation metric for systems with ranked outputs. For a set of Q correct outputs with ranks $rank_i$ among all outputs, standard MRR is computed as:

$$\text{MRR} = \frac{1}{Q}\sum_{i=1}^{Q}\frac{1}{\text{rank}_i}$$

For example, a MRR value of 0.25 means that that the correct answer has rank 4 on average. This metric is designed for tasks where only one of the outputs can be correct. When applied directly to tasks where more than one output can be correct, MRR unfairly penalizes cases with multiple correct outputs, increasing the rank of some correct outputs on account of other, higher-ranked outputs that are also correct. We modify MRR for our task by ignoring correct outputs in the rank computations. In other words, the rank of a correct output is computed as the number of higher-ranked incorrect outputs, plus one. This metric has the advantage of summarizing the accuracy of an auto-coder without reference to a particular choice of threshold, which may be determined by business rules or research considerations, as would be the case for precision and recall.

One advantage of regularized logistic regression is that the value of 1 is often a near-optimal setting for the regularization trade-off parameter. This can save considerable computation time that would be required for tuning this parameter for each experimental condition. We have previously observed that the value of 1 consistently produced near-optimal results for the ℓ_1 regularizer in concept confidence models and for the ℓ_2 regularizer in the code confidence models, and we have used this setting for all the experiments reported here. For the code confidence model with ℓ_1-regularized logistic regression we saw a slight improvement with weaker regularization, and we report the best result we obtained for this model below.

5.2 Results

The results are shown in Table 2. The top MMR score of 0.572 corresponds to a micro-averaged F-score of 0.485 (0.490 precision, 0.480 recall) when the threshold is chosen to obtain approximately equal values for recall and precision[3]. The best result was obtained when:

- the concept models used bag-of-tokens features (with the additional laterality features described in Section 4.2);
- both concept models trained on ICD-9 data and those trained on ICD-10 data were used;
- the code confidence model was trained on data with predictions of concept models trained on all of ICD-10 data (i.e., no

[3]To put these numbers into perspective, note that the average accuracy of trained medical coders for ICD-10 has been estimated to be 63% (HIMSS/WEDI, 2013).

data splitting for stacked generalization was used);

- the code confidence model used all of the feature types described in Section 4.4;
- the code confidence model used logistic regression with ℓ_2 regularization.

We examine the impact of all these choices on system performance in turn.

Model	MRR
All data, all features, ℓ_2 reg.	**0.572**
Concept model training:	
Trained on ICD-10 only	0.558
Trained on ICD-9 only	0.341
Code model features:	
One-vs-all	0.519
No code-bound features	0.553
No quantization features	0.560
Stacked generalization:	
half & half data split	0.501
5-fold cross-validation	0.539
Code model algorithm:	
ℓ_1 regularization	0.528

Table 2: Evaluation results. Each row after the first correponds to varying one aspect of the model shown in the first row. See Section 5.3 for details of the experimental conditions.

5.3 Discussion

Despite its apparent primitive nature, the bag-of-token feature space for the concept confidence models has turned out to provide a remarkably strong baseline. Our experiments with frequency-based weighting schemes for the feature values and with features derived from text matches from the UMLS concept dictionaries did not yield substantial improvements in the results. Thus, the use of UMLS-based features, obtained using Apache ConceptMapper, yielded a relative improvement of 0.6% (i.e., 0.003 in absolute terms), but at the cost of nearly doubling run-time processing time. Nonetheless, we remain optimistic that more sophisticated features can benefit performance of the concept models while maintaining their scalability.

As can be seen from the table, both concept models trained on ICD-9 data and those trained on ICD-10 data contributed to the overall effectiveness of the system. However, the contribution of the latter is markedly stronger. This suggests that further research is needed in finding the best ways of exploiting ICD-9-coded data for ICD-10 auto-coding. Given that data with ICD-9 codes is likely to be more readily available than ICD-10 training data in the foreseeable future, this line of investigation holds potential for significant gains in auto-coding performance.

For the choice of features used in the code confidence model, the most prominent contribution is made by the feature that generalize beyond specific codes, as discussed in Section 4.4. Adding these features yields a 10% relative improvement over the set of features equivalent to a one-vs-all model. In fact, using the generalized features alone (see the row marked "no code-bound features" in Table 2) gives a score only 0.02 lower than the best result. As would be expected, generalized features are particularly important for codes with limited training data. Thus, if we restrict our attention to codes with fewer than 25 training instances (which account for 95% of the unique codes in our ICD-10 training data), we find that generalized features yielded a 25% relative improvement over the one-vs-all model (0.247 to 0.309). In contrast, for codes with over 100 training instances (which account for 1% of the unique codes, but 36% of the total code volume in our corpus) the relative improvement from generalized features is less than 4% (0.843 to 0.876). These numbers afford two further observations. First, the model can be improved dramatically by adding a few dozen EHRs per code to the training corpus. Secondly, there is still much room for research in mitigating the effects of data sparsity and improving prediction accuracy for less common codes. Elsewhere in Table 2 we see that quantization-based features contribute a modest predictive value.

Perhaps the most surprising result of the series came from investigating the options for using the available ICD-10 training data, which act as training material both for concept confidence models and the code confidence model. The danger of training both type of models on the same corpus is intuitively apparent. If the training instances for the code model are generated by concept models whose training data included the same EHRs, the accuracy of these concept predictions may not reflect out-of-sample performance of the concept models, causing the code model to rely on them excessively.

The simplest implementation of Wolpert's stacked generalization proposal, which is intended to guard against this risk, is to use one part of the corpus to train one predictive layer and use its predictions on the another part of the corpus to train the other layer. The result in Table 2 (see the row marked "half & half data split") shows that the resulting increase in sparsity of the training data for both models leads to a major degradation of the system's performance, even though at runtime concept models trained on all available data are used. We also investigated a cross-validation version of stacked generalization designed to mitigate against this fragmentation of training data. We trained a separate set of concept models on the training portion of each cross-validation fold, and ran them on the held-out portion. The training set for the code confidence model was then obtained by combining these held-out portions. At runtime, concept models trained on all of the available data were used. However, as intuitively compelling as the arguments motivating this procedure may be, the results were not competitive with the baseline approach of using all available training data for all the models.

Finally, we found that an ℓ_2 regularizer performed clearly better than an ℓ_1 regularizer for the code confidence model, even though we set the ℓ_2 trade-off constant to 1 and tuned the ℓ_1 trade-off constant on the development test set. This is in contrast to concept confidence models, where we observed slightly better results with ℓ_1 regularization than with ℓ_2 regularization.

6 Conclusion

We have described a system for predicting ICD-10-PCS codes from the clinical narrative contained in EHRs. The proposed approach seeks to mitigate the sparsity of training data with manually assigned ICD-10-PCS codes in three ways: through an intermediate abstraction to clinical concepts, through the use of data with ICD-9 codes to train concept confidence models, and through the use of a code confidence model whose parameters can generalize beyond individual codes. Our experiments show promising results and point out directions for further research.

Acknowledgments

We would like to thank Ron Mills for providing the crosswalk between ICD-10-PCS codes and clinical concepts; Guoli Wang, Michael Nossal, Kavita Ganesan, Joel Bradley, Edward Johnson, Lyle Schofield, Michael Connor, Jean Stoner and Roxana Safari for helpful discussions relating to this work; and the anonymous reviewers for their constructive criticism.

References

Sean Benson. 2006. Computer-assisted Coding Software Improves Documentation, Coding, Compliance, and Revenue. *Perspectives in Health Information Management, CAC Proceedings*, Fall 2006.

Centers for Medicare & Medicaid Services. 2014. *General Equivalence Mappings. Documentation for Technical Users*. Electronically published at cms.gov.

Chute CG, Yang Y, Buntrock J. 1994. An evaluation of computer assisted clinical classification algorithms. *Proc Annu Symp Comput Appl Med Care.*, 1994:162–6.

Jianfeng Gao, Galen Andrew, Mark Johnson, Kristina Toutanova. 2007. A Comparative Study of Parameter Estimation Methods for Statistical Natural Language Processing. *ACL 2007.*

Daniel T. Heinze, Mark L. Morsch, Ronald E. Sheffer, Jr., Michelle A. Jimmink, Mark A. Jennings, William C. Morris, and Amy E. W. Morsch. 2000. LifeCode$^{\text{TM}}$– A Natural Language Processing System for Medical Coding and Data Mining. *AAAI Proceedings.*

Daniel T. Heinze, Mark Morsch, Ronald Sheffer, Michelle Jimmink, Mark Jennings, William Morris, and Amy Morsch. 2001. LifeCode: A Deployed Application for Automated Medical Coding. *AI Magazine*, Vol 22, No 2.

HIMSS/WEDI. 2013. *ICD-10 National Pilot Program Outcomes Report*. Electronically published at himss.org.

Yuankai Jiang, Michael Nossal, and Philip Resnik. 2006. How Does the System Know It's Right? Automated Confidence Assessment for Compliant Coding. *Perspectives in Health Information Management, Computer Assisted Coding Conference Proceedings*, Fall 2006.

Leah Larkey and W. Bruce Croft. 1995. Automatic Assignment of ICD9 Codes To Discharge Summaries. *Technical report, Center for Intelligent Information Retrieval at University of Massachusetts.*

Andrew Kachites McCallum. 2002. MALLET: A Machine Learning for Language Toolkit. *http://mallet.cs.umass.edu*

Medori, Julia and Fairon, Cédrick. 2010. Machine Learning and Features Selection for Semi-automatic ICD-9-CM Encoding. *Proceedings of the NAACL HLT 2010 Second Louhi Workshop on Text and Data Mining of Health Documents*, 2010: 84–89.

Ronald E. Mills. 2013. Methods using multi-dimensional representations of medical codes. *US Patent Application* US20130006653.

S.V. Pakhomov, J.D. Buntrock, and C.G. Chute. 2006. Automating the assignment of diagnosis codes to patient encounters using example-based and machine learning techniques. *J Am Med Inform Assoc*, 13(5):516–25.

Adler Perotte, Rimma Pivovarov, Karthik Natarajan, Nicole Weiskopf, Frank Wood, Noémie Elhadad . 2014. Diagnosis code assignment: models and evaluation metrics. *J Am Med Inform Assoc*, 21(2):231–7.

Pestian, JP, Brew C, Matykiewicz P, Hovermale DJ, Johnson N, Bretonnel Cohen K, and Duch W. 2007. A shared task involving multi-label classification of clinical free text. *Proceedings ACL: BioNLP*, 2007:97–104.

Philip Resnik, Michael Niv, Michael Nossal, Gregory Schnitzer, Jean Stoner, Andrew Kapit, and Richard Toren. 2006. Using intrinsic and extrinsic metrics to evaluate accuracy and facilitation in computer-assisted coding.. *Perspectives in Health Information Management, Computer Assisted Coding Conference Proceedings*, Fall 2006.

Berthier Ribeiro-Neto, Alberto H.F. Laender and Luciano R.S. de Lima. 2001. An experimental study in automatically categorizing medical documents. *Journal of the American Society for Information Science and Technology*, 52(5): 391–401.

Mary H. Stanfill, Margaret Williams, Susan H. Fenton, Robert A. Jenders, and William R. Hersh. 2010. A systematic literature review of automated clinical coding and classification systems. *J Am Med Inform Assoc.*, 17(6): 646–651.

Michael Subotin. 2011. An exponential translation model for target language morphology. *ACL 2011*.

David H. Wolpert. 1992. Stacked Generalization. *Neural Networks*, 5:241–259.