Automating Medical Billing Code Generation Using NLP

Tony Cen Cen
tcen17@bu.edu

Andrew Juang
juanga@bu.edu

Department of Computer Science
Boston University
Boston, MA, USA

Abstract

Medical coding is an essential but labor-intensive task in healthcare administration, translating clinical documentation into standardized codes such as ICD-9, ICD-10, and CPT for medical insurance billing and record-keeping. This project explores the use of natural language processing (NLP) to automate ICD-9 code assignment from free-text clinical notes, using the MIMIC-III dataset as a training and evaluation resource.

We built an end-to-end pipeline that includes specialized text preprocessing, TF-IDF-based feature extraction, deep learning architectures (CNNs and BiLSTMs), and hierarchical code grouping to address class imbalance. Our experiments show that convolutional neural networks (CNNs) significantly outperform baseline logistic regression and BiLSTM models, achieving an accuracy of 39.45%—more than triple that of the statistical baseline.

While promising, challenges such as rare code prediction and multi-label classification remain. Future directions include leveraging domain-specific pretrained models like BioBERT and expanding the system to predict both diagnostic and procedural codes. This work demonstrates the potential of NLP to reduce administrative burden, improve coding accuracy, and accelerate medical billing workflows.

1 Introduction

Medical coding is a critical step in healthcare administration where clinical documentation is translated into standardized billing codes such as ICD-9, ICD-10, and CPT. These codes are essential for a range of operations: enabling insurance reimbursement, facilitating clinical research, supporting public health monitoring, and structuring hospital management data.

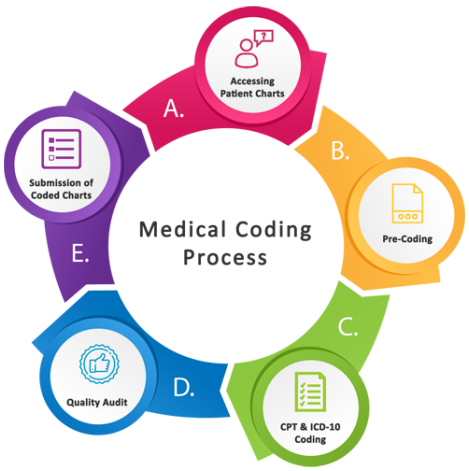

Figure 1: Traditional manual medical coding process

Despite its importance, medical coding remains a largely manual and error-prone process. Highly trained, specialized coders must carefully interpret free-text clinical notes, discharge summaries, and other complex documentation. These notes often vary widely in structure, use specialized medical jargon, and contain nonstandard syntax. Combined with the high volume of documentation produced daily, these challenges contribute to frequent coding errors, delayed reimbursements, compliance issues, and increased administrative costs for healthcare providers.

Recent advances in natural language processing (NLP) and deep learning offer promising tools to automate this task. Our project focuses on developing an NLP-based system that predicts appropriate ICD-9 codes from free-text clinical notes, using the MIMIC-III dataset of ICU patient records. By applying both statistical learning methods and deep neural models, we aim to reduce coding errors, streamline hospital billing processes, and lay the groundwork for scalable, automated medical coding systems.

2 Motivation

Our interest in this project stems from firsthand insights into the challenges of current billing workflows. Through conversations with individuals working at a medical insurance company, we learned how much time and labor is spent manually reviewing and correcting billing codes submitted by clinics. Inaccurate codes frequently led to claim rejections, delayed payments, and additional administrative work.

This experience highlighted an opportunity: if a model could automatically predict likely billing codes from clinical notes with reasonable accuracy, it could significantly reduce human workload and minimize costly errors.

Moreover, clinical discharge summaries often contain key diagnostic information organized into predictable sections, suggesting that well-designed NLP models could perform effectively with appropriate preprocessing. Motivated by these practical challenges and possibilities, we set out to build a system that could improve the accuracy, efficiency, and scalability of the medical coding process.

3 Problem Formulation

3.1 Input and Output

The objective of this project is to develop a system that can automatically assign appropriate ICD-9 billing codes to clinical notes.

Input: The system receives a single free-text clinical note, specifically discharge summaries from ICU hospitalizations. These notes contain diagnostic descriptions, clinical observations, procedures performed, and treatment plans.

Output: For each input note, the system predicts one ICD-9 code that best represents the primary diagnosis associated with the hospitalization. In this initial phase, we focus on single-label prediction, mapping each note to its most important diagnostic code.

Example:

- **Input Note:** "Patient presents with acute chest pain radiating to the left arm. ECG indicates ST-elevation myocardial infarction. Administered aspirin 325 mg and initiated thrombolytic therapy."
- **Predicted Code:** 410.9 (Acute myocardial infarction of unspecified site)

This focused approach allows us to establish a robust proof-of-concept system before expanding to more complex settings such as multi-label classification.

3.2 Scope and Assumptions

Our project operates under several simplifying assumptions to make the initial model feasible given our time constraints:

- **Typed Text Only:** We assume the input clinical notes are typed. Handwritten notes would require additional OCR processing, which is beyond the scope of this project.
- **Single Primary Diagnosis:** Each note is mapped to a single ICD-9 code, representing the most important diagnosis. In real clinical practice, multiple diagnoses may be relevant, but we limit ourselves to the primary label.
- **Discharge Summaries Only:** We focus specifically on discharge summaries, as they summarize the complete course of hospitalization and contain critical diagnostic information.
- **Code Mapping Exists:** We assume that each clinical note is already associated with one or more ICD-9 codes in the dataset (no missing label cases).
- **No Hand-Crafted Feature Engineering:** Except for basic NLP preprocessing (tokenization, abbreviation expansion, section tagging), we avoid manually engineering clinical features like lab values or vital signs.

These assumptions allow us to streamline model development and focus on validating the feasibility of automated ICD-9 prediction using NLP.

4 Data and Preprocessing

4.1 Data Source: MIMIC-III Dataset

We utilize the Medical Information Mart for Intensive Care III (MIMIC-III) database (Johnson et al., 2016), a large, publicly available dataset developed by the MIT Laboratory for Computational

Physiology. MIMIC-III contains de-identified health data from over 40,000 ICU patients at the Beth Israel Deaconess Medical Center between 2001 and 2012.

For this project, we specifically use two key tables from the MIMIC-III database:

- **NOTEEVENTS**: Contains unstructured clinical text notes, including discharge summaries, nursing notes, and radiology reports. We focus exclusively on discharge summaries for ICD-9 code prediction.
- **DIAGNOSES_ICD**: Provides structured mappings between hospital admissions (HADM_ID) and assigned ICD-9 diagnostic codes.

Although MIMIC-III contains many additional tables—such as LABEVENTS (lab results), PROCEDURES_ICD (procedural codes), and PRESCRIPTIONS (medication orders)—we restricted our analysis to NOTEEVENTS and DIAGNOSES_ICD. This selection was made because the other tables in MIMIC-III, while useful for broader clinical research, were not necessary for our goal of mapping free-text clinical notes to ICD-9 codes. The NOTEEVENTS and DIAGNOSES_ICD tables provided all the information needed to build a supervised learning framework focused specifically on discharge summary classification.

Each discharge summary can be linked to corresponding ICD-9 codes through the hospital admission ID (HADM_ID), allowing us to build supervised learning datasets.

Data Access: Because the dataset contains sensitive clinical information (even though it is de-identified), researchers are required to complete a credentialing process before access is granted. This includes taking approved training courses in human subjects research ethics (e.g., CITI Program certification) and formally applying for database access through PhysioNet. We successfully completed these steps to ensure compliance with privacy and data security standards.

Dataset Relevance: The MIMIC-III dataset is especially valuable for our task because it reflects real-world clinical documentation practices, including the variability, messiness, and richness of ICU patient notes.

4.2 Preprocessing Steps

To prepare the raw clinical notes for machine learning models, we implemented several preprocessing steps tailored to the specific challenges of medical text:

- **Tokenization:** We used the NLTK word tokenizer, customized with regular expressions to better handle domain-specific tokens such as medical abbreviations (e.g., ”IV”, ”mg/dL”) and numeric values. Proper tokenization ensures that medical phrases are preserved correctly rather than fragmented.
- **Stopword Removal:** We applied stopword filtering using NLTK’s standard stopword list, but carefully preserved medically important negations such as ”no”, ”not”, ”denies”, as these terms critically affect diagnostic meaning.
- **Lemmatization:** We used the WordNet lemmatizer to normalize words to their root forms (e.g., ”conditions” $\rightarrow$ ”condition”). However, we excluded clinical terms from aggressive lemmatization to prevent distortion of key medical terminology.
- **Abbreviation Expansion:** Many medical notes use standard abbreviations. We expanded common abbreviations (e.g., ”COPD” $\rightarrow$ ”chronic obstructive pulmonary disease”, ”HTN” $\rightarrow$ ”hypertension”) to ensure consistency and improve model interpretability.
- **Section Tagging:** Clinical notes are typically divided into sections like ”HISTORY OF PRESENT ILLNESS” and ”DISCHARGE DIAGNOSIS.” We detected and tagged these section headers, providing structural markers that models could later exploit to identify regions containing critical diagnostic information.

These preprocessing steps helped standardize the clinical notes by normalizing terminology, removing irrelevant or redundant information, and structuring the text in a way that made it easier for models to learn meaningful patterns. By reducing

noise, such as common stopwords and inconsistent abbreviations, and retaining important semantic signals like clinical conditions, medications, and section headers, we ensured that the input features preserved the key medical context necessary for accurate ICD-9 code prediction. This careful preprocessing was especially critical given the variability and complexity inherent in real-world clinical documentation.

5 Methods

5.1 Baseline Model: TF-IDF + Logistic Regression

As a starting point, we built a baseline classifier using TF-IDF feature extraction followed by a logistic regression model.

TF-IDF Vectorization: Each discharge summary was converted into a numerical vector using Term Frequency–Inverse Document Frequency (TF-IDF) weighting. TF-IDF emphasizes words that are important within a document but rare across the entire corpus—ideal for highlighting distinctive medical terminology while down-weighting common, uninformative words.

We generated TF-IDF features from both unigrams (single words) and bigrams (two-word phrases), as many medical conditions are expressed as phrases (e.g., "heart failure", "respiratory distress").

Logistic Regression Classifier: We trained a multinomial logistic regression model using these TF-IDF vectors as input features. Key choices included:

- **Solver:** We used the 'saga' (Stochastic Average Gradient Augmented) solver. SAGA is well-suited for very large and sparse datasets, such as TF-IDF representations of text, because it updates model parameters incrementally (on small batches of data) rather than recalculating gradients over the entire dataset each time. It supports both L1 and L2 regularization, making it highly flexible and efficient for our setting.
- **Class Weighting:** The ICD-9 codes in our dataset are heavily imbalanced, with some diagnoses (like pneumonia or heart failure) appearing much more frequently than rare conditions. To address this, we used the `class_weight="balanced"` option. This reweights the loss function during training so that rare classes have a higher penalty for misclassification, helping the model avoid always predicting the most common codes.
- **Regularization:** We applied L1 regularization (lasso), which encourages the model to assign zero weights to less important features. This promotes sparsity in the model, simplifying it and helping prevent overfitting.

Overall, these design choices aimed to make logistic regression competitive despite the extreme feature sparsity, label imbalance, and complexity of the clinical notes.

Motivation: This baseline approach was motivated by its simplicity, speed of training, and its historical success on large text classification tasks. Additionally, it provides an interpretable foundation against which to compare more sophisticated deep learning models later.

Although we expected the logistic regression baseline to struggle given the complexity and variability of clinical notes, it offered a useful sanity check and a quantifiable starting point for improvements.

5.2 Deep Learning Models

To move beyond the limitations of linear models like logistic regression, we experimented with two deep learning architectures designed to better capture the structure and semantics of clinical text: Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory networks (BiLSTMs).

Convolutional Neural Network (CNN)

Our CNN architecture was based on a simple and effective text classification model. Our goal was to detect **local patterns** (e.g., medical terms, diagnostic phrases) within clinical notes efficiently.:

- **Embedding Layer:** 100-dimensional trainable embeddings initialized randomly.
- **Convolutional Layer:** 128 filters with a window size of 5 tokens, allowing the network to detect important 5-word phrases commonly indicative of medical conditions. This means

the model looks at 5-word chunks at a time, scanning through the text to find important local patterns. In clinical notes, many diagnoses are described in short phrases (e.g., "acute respiratory failure", "chronic kidney disease"), making this window size well-suited.

- **Global Max Pooling:** After convolution, we applied **global max pooling**, which picks the highest activation from each filter across the entire document. This means the model **focuses on the single most important occurrence** of a pattern, rather than its location. This is especially useful when critical diagnostic information can appear anywhere in a discharge summary.

- **Dense Layer:** A fully connected layer followed by a softmax output for classification.

- **Training Setup:** Batch size of 64, sequence length truncated/padded to 500 tokens to ensure uniform input size.

Bidirectional LSTM (BiLSTM)

To better capture the sequential nature of clinical text, we implemented a Bidirectional Long Short-Term Memory (BiLSTM) network. BiLSTMs process input sequences in both forward and backward directions, allowing the model to capture dependencies from both past and future tokens.

- **Embedding Layer:** Each word was mapped to a **200-dimensional trainable embedding**.

- **BiLSTM Layer:** We used a **bidirectional LSTM** with **64 hidden units** in each direction (forward and backward). This setup enabled the model to gather information not only from previous words but also from upcoming words.

- **Dense Layer and Output:** The final hidden states from both directions were concatenated and passed through a **fully connected dense layer**, followed by a **softmax activation** to predict the ICD-9 code.

- **Training Setup:**
  - **Batch Size:** 32
  - **Sequence Length:** Clinical notes were truncated or padded to **1000 tokens** to accommodate longer documents typical of ICU discharge summaries.

The BiLSTM was designed to capture longer-range dependencies across the entire clinical note, which could be beneficial for notes where diagnoses are spread across different sections.

5.3 Label Simplification

One of the major challenges in medical coding is the extreme class imbalance: certain ICD-9 codes are very common, while many rare codes appear only once or twice. To mitigate this issue, we applied hierarchical code grouping.

Hierarchical Grouping Strategy:

- ICD-9 codes are structured hierarchically (e.g., code "428.21" for "Acute systolic heart failure" belongs to parent category "428" for "Heart failure").

- We grouped rare ICD-9 codes into their parent categories based on the first three digits, following standard clinical practice.

Thresholds:

- **Min-count=5:** Codes appearing fewer than 5 times were grouped into their parent categories.

- **Min-count=10:** We experimented with a stricter threshold, grouping codes appearing fewer than 10 times. This improved training stability and boosted performance metrics.

This label simplification process reduced the number of distinct classes, making the classification task more tractable and reducing noise from extremely rare codes.

6 Evaluation Metrics

To evaluate our models, we selected the following metrics:

- **Accuracy:** Measures the proportion of clinical notes for which the predicted ICD-9 code exactly matches the true code. It is a straightforward indicator of model performance but can be misleading in imbalanced datasets.

- **F1-Score (weighted):** Harmonic mean of precision and recall, weighted by class frequencies. This is particularly critical in our setting where common and rare codes are unevenly distributed, ensuring that model performance across all classes is fairly evaluated.

- **Hamming Loss:** Measures the fraction of labels that are incorrectly predicted. A lower Hamming Loss implies fewer errors per document, which is directly important for practical billing accuracy.

- **Cohen's Kappa:** Quantifies the level of agreement between the model's predictions and the true labels beyond chance. This prevents inflated performance scores that could occur from always predicting common codes, and it ensures that the model provides real clinical value.

These metrics together provide a balanced view of raw predictive accuracy, error rates, and model reliability, all of which are crucial for real-world deployment in medical billing systems.

7 Results

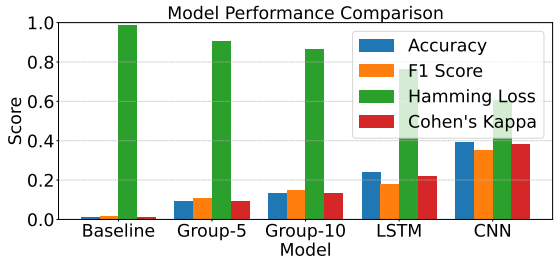

Figure 2: Performance comparison across models (Baseline Logistic Regression, BiLSTM, and CNN). Metrics include Accuracy, F1 Score, Hamming Loss, and Cohen's Kappa.

7.1 Baseline Performance

Our initial baseline model, using TF-IDF features and logistic regression, achieved very poor results on the test set:

- **Accuracy:** 1.16%

- **F1 Score (weighted):** 1.53%

- **Hamming Loss:** 98.84%

- **Cohen's Kappa:** 0.0114

These numbers indicate that the model performed only marginally better than random guessing. The high Hamming Loss and near-zero Kappa suggest that surface-level term frequency features were insufficient to capture the complexity of medical notes.

Applying hierarchical code grouping significantly improved baseline performance. Grouping rare codes (minimum frequency of 5) increased accuracy to 9.59%, and grouping codes with fewer than 10 occurrences further raised accuracy to 13.36%. This showed that reducing label sparsity makes learning feasible even for simple models.

7.2 Deep Model Performance

Switching to deep learning methods led to substantial improvements:

- **CNN Model:**
  - Accuracy: 39.45%
  - F1 Score (weighted): 35.01%
  - Hamming Loss: 60.55%
  - Cohen's Kappa: 0.3834

- **BiLSTM Model:**
  - Accuracy: 23.82%
  - F1 Score (weighted): 18.00%
  - Hamming Loss: 76.18%
  - Cohen's Kappa: 0.221

The CNN model outperformed both logistic regression and the BiLSTM by a wide margin across all metrics. It achieved over three times the baseline logistic regression accuracy, indicating the effectiveness of deep feature extraction and localized pattern detection.

7.3 Analysis: Why CNN Outperformed LSTM

While LSTM architectures are generally favored for sequential data, several task-specific factors explain the CNN's superior performance here:

- **Localized Diagnostic Information:** In discharge summaries, key diagnostic terms often appear in specific sections (e.g., "Discharge Diagnosis"), meaning the global document structure was less critical than detecting localized patterns.

- **Efficiency in Keyword Detection:** CNNs are highly effective at capturing small, discriminative n-gram patterns such as "myocardial infarction" or "pulmonary embolism," which are strong indicators for specific ICD codes.

- **Effective Preprocessing:** Our preprocessing pipeline preserved section headers and expanded abbreviations, creating a semi-structured input where CNNs could exploit regular phrasing and terminology patterns without needing full document memory.

- **Long Sequences Challenge for LSTM:** LSTMs struggled with the long sequence lengths (up to 1000 tokens), which could lead to vanishing gradients or overfitting on irrelevant text.

Thus, in this task, the CNN's ability to quickly identify important phrases proved more critical than the LSTM's sequence modeling capabilities.

8 Challenges and Limitations

Despite encouraging results, our project faced several challenges:

- **Single-label Limitation:** We modeled the problem as single-label classification, predicting only one ICD-9 code per note. In reality, many patients have multiple simultaneous diagnoses, requiring multi-label prediction.

- **Rare Code Challenge:** Some ICD-9 codes occur extremely infrequently, making it difficult for models to learn them. Even with hierarchical grouping, rare classes were often underrepresented or misclassified.

- **Computational Constraints:** Limited computational resources restricted the complexity of our models, the number of training epochs, and hyperparameter tuning. More training time and larger models could potentially yield further improvements.

- **Data Noise:** Clinical notes sometimes contain ambiguities, copy-paste artifacts, and contradictory information, which introduce unavoidable noise into the training process.

9 Future Work

Several promising directions could extend and improve this work:

- **Multi-label Prediction:** Future models should allow assigning multiple ICD-9 codes per clinical note to reflect real-world diagnostic complexity.

- **Domain-aware Pretrained Models:** Fine-tuning pretrained biomedical models like BioBERT or ClinicalBERT could better capture clinical nuances and improve performance, especially for rare or complex conditions.

- **CPT Code Prediction:** Expanding the system to also predict CPT codes (for procedures) would create a full-spectrum billing assistant covering both diagnoses and treatments.

- **Clinician-facing Tools:** Building prototype applications that integrate our prediction system into electronic health record (EHR) workflows could gather real-world feedback and improve adoption.

- **Explainability and Interpretability:** Adding attention mechanisms or saliency maps could make model predictions more interpretable for clinical users, increasing trust and usability.

10 Conclusion

This project demonstrated that natural language processing can substantially assist in automating the medical billing code assignment process from clinical notes. Using the MIMIC-III dataset, we showed that:

- Logistic regression with TF-IDF features provides a weak baseline.

- Hierarchical grouping of rare ICD codes improves performance.

- Deep learning models, particularly CNNs, significantly outperform traditional baselines.

Our best CNN model achieved a 39.45% accuracy, more than tripling the baseline's performance. While challenges remain, especially

around multi-label prediction and rare codes, the results are promising. Future research integrating domain-specific models and clinician-centered design could further enhance the feasibility and impact of automated medical coding systems in healthcare.

References

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3(1):160035.