Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTMENT OF SYSTEMS AND COMPUTER ENGINEERING

Codification of clinical episodes in natural language

Project Report to fulfill the Master's degree in Informatics Engineering

Specialization in Intelligent Data Analysis

Author

Hugo Filipe da Fonseca e Silva

Supervisors

Mateus Mendes

Mário Macedo

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, May 2024

Resumo

A Classificação Internacional de Doenças, 10ª Revisão (ICD-10) tem sido largamente utilizada para classificar informação diagnóstica do paciente. A codificação de episódios clínicos em códigos ICD-10 é uma tarefa laboriosa, geralmente feita por médicos dedicados com formação específica. A codificação automática a partir de Registos de Saúde Eletrónicos (EHR) em códigos de diagnóstico, tem sido desafiante para a comunidade de Processamento de Linguagem Natural (NLP).

Este trabalho apresenta inicialmente uma revisão de literatura sobre o tema da codificação de episódios clínicos em linguagem natural, nomeadamente os problemas e barreiras principais que a afetam, uso de processamento de linguagem natural em paralelo com ontologias, trabalhos de NLP na área da saúde, codificação automática ICD-10, uso de Modelos de Linguagem Pré-treinados (PLM), bem como trabalhos desenvolvidos para resolver a problemática de abreviaturas clínicas e deteção de negação de sintomas clínicos.

Pretende também propor o método PLM-ICD-C baseado na similaridade do cosseno, para processar EHRs em linguagem natural, fornecendo sugestões de códigos ICD-10 para os codificadores, por forma a facilitar o processo. Para tal propõe-se uma técnica de múltiplos runs e estratégia de bucket category, aplicada ao dataset Medical Information Mart for Intensive Care (MIMIC)-IV. Os resultados mostram que a estratégia de utilização do conceito de bucket category melhora os resultados, ao mesmo tempo que fornece sugestões úteis, onde a Precisão tem um valor 5 vezes maior, enquanto há melhorias de 2 a 3 vezes na Recall e de 4 vezes no F1-score.

A metodologia anterior é combinada com o PLM-ICD, por forma a aumentar o número de sugestões prováveis úteis de códigos ICD-10. Os resultados mostram que o uso do PLM-ICD-C, que consiste no método do cosseno melhorado e PLM-ICD, melhora os resultados, aumentando o F1-score em 0.5%, mas mais importante, aumentando a Precisão de 46.3% para 50%, o que significa uma melhoria significativa nas sugestões de códigos dados aos médicos que executam funções de codificação.

Palavras-chave: Codificação Automática, Episódios Clínicos, ICD-10, NLP, PLM, PLM-ICD-C, Registo de Saúde Eletrónico, Similaridade do Cosseno

Abstract

The International Classification of Diseases, 10th Revision (ICD-10), has been widely used to classify patient diagnostic information. Encoding pathologies of clinical episodes into ICD-10 codes is a laborious task, usually done by dedicated physicians with specific training. Automatically classifying Electronic Health Records (EHR) from text into diagnostic codes has been challenging to the Natural Language Processing (NLP) community.

This work presents a literature review on the subject of coding clinical episodes in natural language, namely the main problems and barriers that affect it, the use of natural language processing in parallel with ontologies, use of NLP in the area of healthcare, automatic ICD-10 coding, use of Pretrained Language Models (PLM), as well as developed works to solve the clinical abbreviation problem, and detection of clinical symptom negation.

It also intends to propose the method PLM-ICD-C based on the cosine similarity, to process EHRs with natural language texts, in order to give useful suggestions of ICD-10 codes for the coders, aiming to facilitate the process. For that it is proposed a technique of multiple runs and a bucket category strategy, applied to the Medical Information Mart for Intensive Care (MIMIC)-IV dataset. The results show that the strategy of using the concept of bucket category improves the results, while providing useful suggestions, where Precision has a 5-fold improvement, while there are 2-3 fold improvements in Recall and 4-fold improvements in F1-score.

The previous methodology is combined with PLM-ICD, in order to increase the number of probably useful suggestions of ICD-10 codes. The results show that the use of PLM-ICD-C, consisting of the improved cosine method and PLM-ICD, improves the results, increasing the F1-score by 0.5%, but most important, by increasing the Precision from 46.3% to 50%, which means a significant improvement on the code suggestions given to the medical doctors performing encoding functions.

Keywords: Automatic Coding, Clinical Episodes, Electronic Health Record, ICD-10, NLP, PLM, PLM-ICD-C, Cosine Similarity

Epigraph

Do not go where the path may lead, go instead where there is no path and leave a trail.

Ralph Waldo Emerson

Acknowledgments

I would like to thank the supervisors of this project, in particular Professor Mateus for the guidance, patience and wisdom shared throughout this work. Without your guidance and encouragement, this project would not have been possible. I would also like to thank my friends, whose support and encouragement were fundamental in overcoming the challenges. To my family, my safe haven, for their unconditional love and support at all times. This work is dedicated to all of you, with deep gratitude.

Table of Contents

List of Tables

List of Figures

List of Acronyms

API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
CHUC	Coimbra Hospital and University Centre
CITI	Collaborative Institutional Training Initiative
CNN	Convolutional Neural Networks
CRF	Cibersecurity Regulatory Framework
DDR	Double Data Rate
EHR	Electronic Health Record
FN	False Negative
FP	False Positive
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
HIPAA	Health Insurance Portability and Accountability Act
HHS	Health and Human Services
HMS	Harvard Medical School
ICD	International Classification of Diseases
ICS	Improved Cosine Similarity
ICU	Intensive Care Unit
IDO	Infectious Disease Ontology
INCD	National Infrastructure of Distributed Computation
ISEC	Coimbra Institute of Engineering
LCS	Longest Common Subsequence
LSTM	Long Short-Term Memory
MIMIC	Medical Information Mart for Intensive Care
MLP	Multilayer Perceptron
NEC	Not Elsewhere Classified
NER	Named Entity Recognition
NLP	Natural Language Processing
OOV	Out Of Vocabulary
OWL	Web Ontology Language
PLM	Pretrained Language Model
POS	Part Of Speech
RDF	Resource Description Framework
REST	Representational State Transfer
RNN	Recurrent Neural Networks

SNOMED	Systematized Medical Nomenclature for Medicine
TF-IDF	Term Frequency-Inverse Document Frequency
TN	True Negative
TP	True Positive
UMLS	Unified Medical Language System
WHO	World Health Organization

1 Introduction

Clinical episodes are normally registered by physicians or other health professionals using natural language. For every clinical episode, an EHR is registered and stored in the patient's clinical history. Since the clinical descriptions are recorded in natural language, there may be ambiguities, different denominations for the same pathology, symptoms, or therapies. Additionally, natural text is non-structured, so information is more difficult to retrieve than in structured databases. In order to facilitate procedures of text mining and information retrieval in EHRs, an international encoding system was developed, where each pathology and symptom is assigned a unique code. This encoding system is the ICD. The ICD is a medical diagnostic and procedure coding system that was developed by the World Health Organization (WHO) in the 1970s. ICD has suffered several major revisions along the years, incorporating new scientific knowledge, expanding its scope and content, and improving its structure and format. In 2009, the U.S. Department of Health and Human Services (HHS) released a final ruling for the U.S. to adopt the ICD-10 code set. In 2014, HHS issued a final rule formally establishing October 1 2015 as the ICD-10 compliance date and requiring the continued use of ICD-9 through September 30 2015. The latest version of the ICD is ICD-11, and was adopted in 2019 by the 72^{nd} World Health Assembly, becoming effective on 1 January 2022 [1].

One dataset used in the present work, MIMIC-IV, does not provide ICD-11 codes, providing only ICD-9 and ICD-10 codes for the diagnosis and procedures of the patients, so the ICD-10 was used. ICD-10 is widely used to describe patient diagnostic information. Codification of EHR into ICD-10 codes is normally done and audited by medical doctors with proper training in the process. The doctors read the clinical texts and assign the EHR the corresponding ICD-10 codes, so that retrieval procedures are facilitated in the future.

1.1 Motivation

The process of manually assigning the codes is very time consuming and prone to errors. For each illness mentioned in the EHR, the coders need to find the corresponding ICD-10 code and assign it to the episode. Totally automatic encoding could save time, improve efficiency and the quality of the work. Nonetheless, despite this being an area of heavy research, total automation is not expected to be feasible in the coming years, due to the complexity of the process and the rigour that is demanded in the

task, where errors can have unexpected catastrophic consequences. Therefore, a more realistic and safer goal is to produce co-pilots that can help in the encoding process, leading the coders to the right codes with minimal errors and, despite the fact that many approaches were introduced to help the encoding process, only a few have been applied in practice, due to the difficulty and low precision.

This project was initially intended to be done in collaboration with the Infectious Diseases Unit of the Coimbra Hospital and University Centre (CHUC). In order to get a dataset in Portuguese, a request was submitted, through the Infectious Diseases Unit, to the Unit of Research and Development of CHUC. The request was started in October 2022, but unfortunately the procedure was not finished in time to proceed with the experimental work. Therefore, the work continued using a dataset in English, the MIMIC-IV available in PhysioNet [1]. In order to get access to the dataset, the author successfully completed a specific training through the Collaborative Institutional Training Initiative (CITI) Program [2].

1.2 Objectives

This project aims to achieve the following generic objectives:

- Conduct a state-of-the-art survey of methods and systems for coding clinical episodes and electronic health records;
- Survey the state-of-the-art in terms of medical ontologies available in Portuguese or English, choosing one or more for use in the project;
- Design, develop and test the prototype of an application for analyzing clinical episodes described in natural language and coding clinical episodes.

The research questions to be answered during the research are:

- Is there a way to improve the state-of-the-art in EHR coding, using the cosine text similarity method?
- Is it possible to improve the results of the PLM-ICD as a tool to aid in the EHR encoding process?
- Is cosine method appropriate to improve PLM-ICD metrics?

1.3 Structure

Besides the first introductory chapter, this thesis has eight more chapters. Chapter 2 presents related work on quality in clinical coding, use of NLP with ontologies, NLP in

[1] https://physionet.org/content/mimiciv/2.2/, accessed on 2024-01-28
[2] https://www.citiprogram.org/, accessed on 2024-01-28

the area of healthcare, works on automatic ICD-10 coding, use of pretrained language models, as well as developed works to solve the clinical abbreviation problem, and detection of clinical symptom negation. Chapter 3 aims to describe medical ontologies. In Chapter 4 are explained some theoretical fundamentals, namely some NLP state-of-the-art models, along with parsing, text transformations, and Named Entity Recognition (NER). In Chapter 5 are described the datasets used, as well as its exploratory analysis. Chapter 6 describes the methodology. In Chapter 7 are described the conducted experiments as well as the results. Chapter 8 aims to discuss the results, and finally, Chapter 9 presents the main conclusions and future work.

2 State-of-the-art

This section makes a literature review of related works, focusing on the importance of coding quality, the main problems and barriers that impact this quality, the use of NLP, allied to the use of ontologies and how it can benefit their use, the application of innovative techniques of natural language processing in the areas of health, work carried out in the area of automatic ICD coding, use of pretrained language models, as well as works developed aiming to solve the clinical abbreviation problem, and detection of clinical symptom negation.

2.1 Quality in clinical coding

Alonso *et al.* (2020) [2] explore the perception of medical coders regarding possible problems with clinical records that can impact the quality of coded data. The authors show that there are several problems in clinical records, which influence the quality of the coded data, namely the lack of clear documented information, the variability of the diagnosis description, "copy and paste", and the lack of solutions to solve these problems. Obstacles to good clinical coding are variations in the description of the diagnosis by clinicians, lack of clarity in records, lack of readability, incomplete documentation, use of synonyms and abbreviations, and lack of communication between health professionals and clinical coders.

Lucyk *et al.* (2017) [3] explore the potential barriers that exist to quality coding. Namely, the mapped documentation of clinical events from admission to medical discharge, for example through the use of abbreviations or acronyms, mapping organization and its conjugation, variability in the interpretation of mapped information, and high mapping quota of clinicians to encode in a period of time, that originates an additional pressure on coding, compromising its quality.

Tang *et al.* (2017) [4] identify a range of sources of errors in the coding process, including the recognition of new diagnoses. However, the adoption of new diagnoses within the ICD-10 system is slow, leading to confusion between physician notes and coders. In this sense, the authors recommend that health administrators and politicians be encouraged to adopt the most recent diagnostic systems and train physicians and coders in their use.

2.2 Ontology-based NLP

Abburu *et al.* (2017) [5] implement a mechanism to extract relevant information from heterogeneous semi-structured and structured documents, using NLP and representing the extracted information in Resource Description Framework (RDF) ontology format. The proposed method comprises three main steps: ontology-based NLP process, extraction and integration of information, and semantic query and visualization.

Liu *et al.* (2011) [6] review existing methodologies and developed systems, and discuss how existing methods can benefit ontology development. For tasks like NER, almost any kind of terminology can be used, but more complex tasks like concept identification require the representation of synonyms. Some NLP tasks require more complex relationships between concepts, such as word disambiguation, coreference resolution, and attribute and value extraction. Ontologies can be used for complex inferences and to obtain rules needed for semantic interpretation and question-answer systems. Methods and algorithms for information gathering and free text extraction in knowledge resources can be categorized into symbolic, statistical and hybrid approaches.

Ayadi *et al.* (2020) [7] present an ontology-based NLP approach to automatically extract and transfer data from text sources to database. This approach combines the use of NLP techniques and a domain ontology to automatically extract environmental exposure and hazards information, by classifying according to the ontology concepts, and automatically enrich the database. On one hand, the NLP techniques are used to intelligently analyze textual documents and identify pertinent information from documents. On the other hand, the role of the domain ontology is to semantically classify and categorize the extracted information according to its concepts, thus associating the information with their corresponding attributes within the database. As a domain ontology, the authors used the EXPOSEO ontology, for environmental exposure to engineered nanomaterials ontology.

2.3 NLP in healthcare

Kormilitzin *et al.* (2021) [8] introduce a NER model for natural language processing in the clinical area. The model was trained to recognize seven categories: drug name, route of administration, frequency, dosage, robustness, form and duration. The model was initially pretrained on the task of predicting the next word, using a library of 2 million patient text records from the MIMIC-III corpus, clinical database, followed by a fine-tuning on the NER task.

Li *et al.* (2022) [9] describe NLP methods in the healthcare area, emphasizing tasks like classification, prediction, word embedding and extraction. Architectures used in NLP, like Autoencoders, Convolutional Neural Networks (CNN), Recurrent Neural

Networks (RNN), and Sequence-to-Sequence models, are described.

2.4 ICD-10 automatic coding

Chen *et al.* (2021) [10] build a deep learning model for ICD-10 coding to automatically determine corresponding diagnosis and procedure codes based on free medical text notes, in order to reduce human effort and increase accuracy. The entire process consists of data processing, feature extraction and model building and training.

Chen *et al.* (2017) [11] present an improved approach based on Longest Common Subsequence (LCS) and semantic similarity for automatic diagnosis, mapping disease names given by clinicians, into ICD-10 disease names. The LCS algorithm comprises input from clinical diagnoses, word segmentation, LCS calculation between the name of the diagnosed disease and the name of the ICD-10 disease, and the measurement of similarity, originating the corresponding output.

Ning *et al.* (2016) [12] focus on the development of a method that automatically assigns an ICD-10 code to a clinical text, using a hierarchical method. Compared to the flat method which searches within more than 10,000 subcategory codes, the hierarchical method only scans the entire section code (262 codes), the category codes under a specific section code (less than 40 codes), and the subcategory codes under a specific category code (maximum 10). In this way, the number of codes that need to be examined is greatly reduced, improving the efficiency of the code assignment task.

2.5 Pretrained language models

Devlin *et al.* (2018) [13] present a language representation model, BERT (Bidirectional Encoder Representations from Transformers), that is designed to pretrain deep bidirectional representations from unlabeled text. This model can be fine-tuned with just one additional output layer.

Lee *et al.* (2020) [14] introduce BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining), a domain-specific language representation model that is pretrained on large-scale biomedical corpora.

In the same way Alsentzer *et al.* (2019) [15] present ClinicalBERT, Gu *et al.* (2021) [16] the PubMed-BERT, and Lewis *et al.* (2020) [17] the RoBERTa-PM. These models are pretrained on domain-specific, crawled and processed for improvement of the downstream performance. These PLMs improved performance on tasks like text mining, NER, relation extraction, and question answering.

Zhang *et al.* (2020) [18] propose an extension of BERT for ICD coding, BERT-XML. It was pretrained on a large corpora of EHRs with specific vocabulary. This model

handles long input text by splitting into chunks and perform predictions for each chunk with a label attention mechanism.

Huang *et al.* (2022) [19] propose a Pretrained Language Model to predict ICD codes (PLM-ICD) that tackles the challenges of previous pretrained models, namely the use of domain-specific pre-training, segment pooling for the long input sequence problem, and level attention for the large label set problem. Experiments were conducted over MIMIC-II and MIMIC-III datasets. PLM-ICD is the first Transformer-based pretrained language model that has a competitive performance on these MIMIC datasets.

Edin *et al.* (2023) [20] reproduce, compare, and analyze state-of-the-art machine learning models for automatic medical encoding, including PLM-ICD. The authors show that many models underperform due to poor configurations, poor train-test splits, and insufficient evaluation. The analysis shows that all models have difficulties with rare codes, while long documents have a negligible impact. They present the first results on the new released MIMIC-IV dataset.

2.6 Clinical abbreviations

Sánchez *et al.* (2018) [21] describe a method to extract abbreviations within a document using regular expressions. Some assumptions were taken into account, namely that the abbreviation could appear many times in the document, its length should be between two and eight characters, and have just one and single definition. The authors used the dictionary of medical acronyms as the main source for the definitions, in total 3386, stored in a database and exposed as a service in a Representational State Transfer (REST) Application Programming Interface (API). Definitions are returned as a list of key-value pairs, composed by the short form and the long form or definition. Some abbreviations could have more than one definition.

Jaber *et al.* (2021) [22] propose a method to get the right sense of abbreviations in clinical texts, by considering a Word Sense Disambiguation (WSD) task in NLP. The authors investigated four strategies that integrate pretrained word embeddings as features to train two supervised learning machine learning models, namely Support Vector Machines (SVM) and Naive Bayes (NB). The method has three main phases. First, dataset generation where different preprocessing tasks are performed to prepare and clean data. Second, training a machine learning model for classification (SVM and NB methods) using different types of features, and finally, testing the model over the test dataset.

Skreta *et al.* (2021) [23] present a method for medical acronyms that improves a model's ability to generalize through novel data augmentation techniques that use information from biomedical ontologies. The authors took the following three approaches: first, they used information from related medical concepts to create more balanced and rep-

resentative examples of training data. Second, they leveraged structural relationships in biomedical ontologies such as the Unified Medical Language System (UMLS) to pretrain the models by constraining medical concepts to be in the same vector space as their neighbors. Third, they defined a simple global context that combines medical knowledge from the entire note and used it in conjunction with the local context of an abbreviation to improve accuracy.

2.7 Clinical symptom negation

Dalloux *et al.* (2021) [24] present a work for detection of negation in the biomedical domain, for French and Brazilian Portuguese languages, and developed new corpora for those, which have been manually annotated for marking up the negation cues and their scope. For the automatic detection, the authors use automatic methods based on supervised machine learning approaches.

Tanushi *et al.* (2013) [25] present a work by comparing three different systems for negation detection in Swedish clinical text, namely NegEx, PyConTextNLP and SynNeg, which have different approaches for determining the scope of negation cues. NegEx uses the distance between the cue and the disease, PyConTextNLP relies on a list of conjunctions limiting the scope of a cue, and in SynNeg the boundaries of the sentence units, provided by a syntactic parser, limit the scope of the cues.

2.8 Summary

Table 2.1 shows a summary of the previous state-of-the-art articles regarding quality in clinical coding, where the approach and main results of each one can be consulted.

Table 2.1: Summary of state-of-the-art articles regarding quality in clinical coding

Title	Author	Year	Approach	Main Results
Health records as the basis of clinical coding: Is the quality adequate? A qualitative study of medical coders' perceptions	Alonso V, Santos J, Pinto M et al	2020	The authors explore the perceptions of medical coders regarding possible problems with health records that can impact the quality of coded data	There are several problems that influence coded data, namely the lack of or unclear document information, variability in diagnosis description, copy and paste, and lack of solutions to solve these problems. Obstacles for good clinical coding are the variations in the description of diagnosis, lack of clarity and eligibility, incomplete documentation, use of synonyms and abbreviations and lack of communication between health professionals and medical coders
Barriers to data quality resulting from the process of coding health information to administrative data: a qualitative study	Lucyk K, Tang K, Quan H	2017	The authors explore the potential barriers that exist for high quality data coding through qualitative inquiry into the roles and responsibilities of medical chart coders	The main barriers to data quality are the chart documentation of clinical events from admission to discharge, chart organization and assembly, variability in the interpretation of chart information, and high quota expectations
Coder perspectives on physician-related barriers to producing high-quality administrative data: a qualitative study	Tang, Karen L and Lucyk, Kelsey and Quan, Hude	2017	The authors recruited 28 coders that worked in healthcare, where semistructured interviews were performed, audio-recorded and transcribed. The questions were related to coder training, work environment, documentation and coding standards	The main barriers that affect coding of high-quality are that the coders are limited in add to, modify or interpret physician documentation, physician documentation is incomplete and nonspecific, chart information tends to be replete with errors and discrepancies, physicians and coders use different terminology to describe clinical diagnoses, and there is a barrier on the communication between coders and physicians

Table 2.2 shows a summary of the previous state-of-the-art articles regarding ontology-

based NLP, where the approach and main results of each one can be consulted.

Table 2.2: Summary of state-of-the-art articles regarding ontology-based NLP

Title	Author	Year	Approach	Main Results
Natural Language Processing methods and systems for biomedical ontology learning	Liu K, Hogan W, Crowley R	2011	The authors review existing methodologies and developed systems, and discuss how existing methods can benefit the development of biomedical ontologies, where these must achieve a high degree of coverage of the domain concepts and concept relationships	Any type of terminology can be used for simple NLP tasks, such as NER. More complex tasks such as identification of concepts, requires synonyms. Some NLP tasks require more complex relationships, such as word sense disambiguation, co-reference resolution, discourse reasoning, and extraction of attributes and values. For information retrieval and extraction from free-text knowledge resources, methods categorized as symbolic, statistical, and hybrid approaches, can be used. The symbolic approach uses linguistic information, as LSP, or internal syntactic structure of component terms. The statistical approach uses large corpora of text data, and can be categorized in clustering and machine learning methods. For extraction of synonyms and concepts, symbolic and statistical methods can be used. For symbolic methods, compound noun information and LSP can be used. For statistical methods, clustering and machine learning approaches are commonly applied. For the extraction of taxonomic and non-taxonomic relationships, again statistical methods as LSP can be used, and statistical methods like clustering and machine learning are applied
Ontology-based NLP information extraction to enrich nanomaterial environmental exposure database	Ayadi A, Aufan M, Rose J	2020	The authors present an ontology-based NLP approach to automatically extract and transfer data from text sources to database, combining NLP techniques and a domain ontology to automatically extract environmental exposure and hazards information, by classifying according to the ontology concepts	The architecture of the proposed ontology-based NLP approach, is composed by the preprocessing and NLP component, information extraction and classification component, and database enrichment component. The first component aims to transform raw data into understandable format, using different text preprocessing services such as normalization, stemming and lemmatization, sentence boundary detection, removal of stop words, tokenization, part-of-speech tagging, and sentence parsing. The second component ensures the task of Named Entity Recognition and their classification, while analyzing the nominal phrases provided by the previous component in order to identify and categorize relevant entities according to the semantic categories. First, a predefined dictionary was developed based on the concepts of the domain ontology to ensure the recognition process. Then it was used an algorithm for matching the entities composing the extracted noun phrases with their semantic categories provided by the dictionary. Using SQL and SPARQL, the algorithm migrates the classified entities from their semantic categories to their appropriate positions in the database. Were defined five main categories, where manually, 26 terms were wrongly identified and 26 entities were incorrectly classified. With the proposed approach, 33 terms were unambiguously identified and 32 entities were incorrectly classified. From a semantic point of view and synonymy of terms, the automatic approach is more efficient

Table 2.3 shows a summary of the previous state-of-the-art articles regarding NLP in healthcare, where the approach and main results of each one can be consulted.

Table 2.4 shows a summary of the previous state-of-the-art articles regarding ICD-10 automatic coding, where the approach and main results of each one can be consulted.

Table 2.5 shows a summary of the previous state-of-the-art articles regarding pretrained language models, where the approach and main results of each one can be consulted.

Table 2.6 shows a summary of the previous state-of-the-art articles regarding clinical abbreviations, where the approach and main results of each one can be consulted.

Table 2.7 shows a summary of the previous state-of-the-art articles regarding clinical symptom negation, where the approach and main results of each one can be consulted.

Table 2.3: Summary of state-of-the-art articles regarding NLP in healthcare

Title	Author	Year	Approach	Main Results
Med7: A transferable clinical natural language processing model for electronic health records	Kormilitzin A, Vaci N, Liu Q et al.	2021	The authors introduce a NER model for clinical natural language processing, where it was trained to recognise seven categories. The model was first pretrained to predict the next word, followed by fine-tuning on the NER task	The dataset for training was combined of gold and silver annotated corpora. The model development included self-supervised pretraining and hyperparameter optimization to achieve high accuracy and robust generalisation. Text cleaning and preprocessing steps were taken to standardise texts. The pretrained language model was used to initialise the weights of Convolutional Neural Network layers, rather than starting with random weights, being experimented various combinations of hyperparameters. The overall performance of the NER model across all seven categories was F1=0.957 (0.893), with Precision=0.982 (0.916) and Recall=0.933 (0.871) for lenient (strict) estimates. Self-supervised pretraining of deep learning models has shown its efficiency in many NLP tasks. The larger model outperformed the default one by a small margin. Another objective of this work was to estimate the degree of transferability of the developed information extraction model to another clinical domain. Leveraging the transfer learning approach, whereby re-using the pretrained Med7 model on two different datasets, resulted in higher accuracy comparable with training and testing in the same domain
Neural Natural Language Processing for unstructured data in electronic health records: A review	Li I, Pan J, Goldwasser J et al.	2022	The authors present a survey paper, where they summarize current neural NLP methods for EHR applications, focusing on tasks like classification and prediction, word embeddings, and extraction	In NLP, the basic network architectures commonly used are Autoencoders, Convolutional Neural Networks, Recurrent Neural Networks, and Sequence-to-Sequence models. Regarding the semantic of words, embeddings are used, where the word2vec method is used, later with FastText that modified word2vec by inputting n-gram character strings, and finally doc2vec is used to learn fixed-length dense embeddings. ELMo introduced the use of bidirectional LSTMs to consider a word context in both directions. BERT is another breakthrough combining benefits of bidirectional training with the Transformer model that uses a modified attention mechanism to represent text in a parallel fashion. Transfer learning has shown immense usefulness in NLP, where they are pretrained on gigantic corpora, and language models like ELMo and BERT can be easily adapted for classification tasks. Segmentation serves an important role in preprocessing, where it can find boundaries between sections of a text. Word sense disambiguation is used to assign the correct meanings to an ambiguous word given its context, which is particular useful in EHRs. Medical coding task attempts to map text from EHR to ICD codes, where deep learning-based methods can be applied. Various applications of clinical embeddings can be found, namely medical concept embeddings, visit embeddings, patient embeddings, and BERT-based embeddings. Finally, information extraction is the task of automatically identify important content in unstructured natural language text, encompassing several subtasks, incluiding NER, entity linking, relation and event extraction, and medication information extraction

Table 2.4: Summary of state-of-the-art articles regarding ICD-10 automatic coding

Title	Author	Year	Approach	Main Results
Automatic ICD-10 Coding and Training System: Deep Neural Network Based on Supervised Learning	Chen PF, Wang SM et al.	2021	The authors construct an automatic ICD-10 coding and training system based on NLP technology, attention mechanism, and DNN models, which are applied for extracting information from EHR data, highlighting the key points from the extracted features, and implementing ICD-10 classification task	For classification of ICD-10-CM codes, the best DNN classification model based on BERT embedding and FCNN with BiGRU could achieve an F1-score of 0.715 and Recall@20 of 0.873. For the ICD-10-PCS, for the same models, could achieve an F1-score of 0.618 and a Recall@20 of 0.887. Regarding the ICD-10 classification with attention, could achieve an F1-score of 0.86
Automatic ICD-10 coding algorithm using an improved longest common subsequence based on semantic similarity	YunZhi Chen, HuiJuan Lu, LanJuan Li	2017	The authors present an improved approach based on the Longest Common Subsequence (LCS) and semantic similarity for automatic Chinese diagnoses, mapping from the disease names given by clinician to the disease names in ICD-10. It is also compared with HowNet which is a network knowledge base, which describes the relationship between the concepts, and compared with n-grams which is a contiguous sequence of n items from a given sequence of text	The average similarity value of namely W-LCS, proposed by the authors, is the largest, which verified that this similarity measurement method can enlarge the value of calculation results. Compared with computing semantic similarity of words, the F1-score value of word matching method is greatly reduced, only 0.760, which means semantic similarity in clinical diagnosis of automatic coding is indispensable, and also illustrates the process of English language may not be entirely or directly applied to Chinese text. The F1-score in W-LCS algorithm is 0.811, higher than the F1-score in HowNet (0.783), which shows the coding accuracy of W-LCS algorithm is higher than that of HowNet. The accuracy of W-LCS algorithm based on Chinese ICD coding is slightly higher than that of the bigrams algorithm
A hierarchical method to automatically encode Chinese diagnoses through semantic similarity estimation	Wenxin Ning, Ming Yu, Runtong Zhang	2016	The authors focus on developing a method that automatically performs ICD-10 code assignment to Chinese diagnoses from the electronic medical records to support the medical coding process in Chinese hospitals, using a flat method that determines the subcategory code to a given diagnosis, and a proposed hierarchical method that determines the most suitable code until the subcategory code is obtained. One knowledge-based and two distributional approaches were applied for word-to-word semantic similarity estimation, where word similarity threshold was also introduced	In addition to the validity of introducing word similarity threshold, the results also reveal that hierarchical method outperforms flat method in terms of coding accuracy. Apart from coding accuracy, time complexity, the hierarchical method is much faster than flat method. Measuring semantic similarity between Chinese medical words using HowNet, a domain-independent knowledge base, achieves higher coding accuracy than using distributional approaches

Table 2.5: Summary of state-of-the-art articles regarding pretrained language models

Title	Author	Year	Approach	Main Results
Bert: Pretraining of deep bidirectional transformers for language understanding	Devlin, Jacob and Chang, Ming-Wei and Lee et al.	2018	The authors introduce a new language model called BERT (Bidirectional Encoder Representations from Transformers), designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers	BERT model can be finetuned with just one additional output layer for a wide range of tasks, without substantial architecture modifications. It achieves state-of-the-art results as pushing the GLUE score to 80.5%, MultiNLI accuracy to 86.7%, SQuAD v1.1 question answering Test F1 to 93.2%, and SQuAD v2.0 Test F1 to 83.1%
BioBERT: a pre-trained biomedical language representation model for biomedical text mining	Lee, Jinhyuk and Yoon, Wonjin and Kim et al.	2020	The authors investigate how the pretrained language model BERT can be adapted for biomedical corpora	BioBERT outperforms BERT and previous state-of-the-art models in several text mining tasks: biomedical Named Entity Recognitions (0.62% F1-score improvement), biomedical relation extraction (2.00% F1-score improvement), and biomedical question answering (12.24% MRR improvement)
Publicly available clinical BERT embeddings	Alsentzer, Emily and Murphy, John R and Boag et al.	2019	The authors address the need of exploring and releasing BERT models for clinical text, namely for generic clinical text and another for discharge summaries	It is demonstrated that using domain-specific model yields performance improvements on three common clinical NLP tasks as compared to nonspecific embeddings
BERT-XML: Large scale automated ICD coding using BERT pretraining	Zhang, Zachariah and Liu, Jingshu and Razavian, Narges	2020	The authors propose a model, BERT-XML, for large scale automated ICD coding, using unsupervised pretraining that have achieved state-of-the-art performance in NLP tasks	The train of a BERT model from scratch, outperform off-the-shelf models. BERT architecture is adapted for ICD coding with multi-label attention. The effectiveness of BERT-based models is demonstrated on the large scale ICD code classification task
PLM-ICD: automatic ICD coding with pretrained language models	Huang, Chao-Wei and Tsai, Shang-Chi et al.	2022	The authors analyze the causes of the underperformance and develop a framework for automatic ICD coding with pretrained language models	Three main issues are spotted through the experiments: large label space, long input sequences and domain mismatch between pretraining and fine-tuning. The proposed PLM-ICD, is a framework that tackles the challenges, showing that can overcome them and achieve state-of-the-art performance in terms of multiple metrics on the MIMIC data
Automated Medical Coding on MIMIC-III and MIMIC-IV: A Critical Review and Replicability Study	Edin, Joakim and Junge, Alexander and Havtorn et al.	2023	The authors reproduce, compare and analyze state-of-the-art automatic medical coding models	Several models underperform due to weak configuration, poorly sampled train-test splits and insufficient evaluation. A revised model is provided using stratified sampling and identical experimental setups, including hyperparameters and decision boundary tuning. The analysis confirms that all models struggle with rare codes, while long documents only have a negligible impact. The code, model parameters and new MIMIC-III and MIMIC-IV training and evaluation pipelines are released to accommodate fair future comparisons

Table 2.6: Summary of state-of-the-art articles regarding clinical abbreviations

Title	Author	Year	Approach	Main Results
A Simple Method to Extract Abbreviations Within a Document Using Regular Expressions	Sánchez, Christian and Martínez, Paloma	2018	The authors describe a method to extract abbreviations within a document using regular expressions	Were obtained results of F1-score of 50.16%, where some issues were identified that have impact on the results, like the definitions could appear in different forms, there are variants of some of the definitions, and some typos
Disambiguating Clinical Abbreviations using Pre-trained Word Embeddings	Jaber, Areej and Martínez, Paloma	2021	The authors have investigated four strategies, to determine the right sense of an abbreviation, that integrate pretrained word embedding as features to train two supervised machine learning models: Support Vector Machines (SVM) and Naive Bayes (NB)	The results show that SVM performs better than NB in all four strategies, where the highest accuracy is 97.08% using a pretrained model trained from Wikipedia, PubMed and PubMedCentral texts
Automatically disambiguating medical acronyms with ontology-aware deep learning	Skreta, Marta and Arbabi, Aryan and Wang et al.	2021	The authors present a method for disambiguating abbreviations, that improves a model's ability to generalize through novel data augmentation techniques that uses information from biomedical ontologies in the form of related medical concepts	The model is trained on a public dataset (MIMIC-III) and tested on automatically generated and hand-labelled datasets from different sources (MIMIC-III, CASI, i2b2). Together, these techniques boost the accuracy of abbreviation disambiguation by up to 17%

Table 2.7: Summary of state-of-the-art articles regarding clinical symptom negation

Title	Author	Year	Approach	Main Results
Supervised learning for the detection of negation and of its scope in French and Brazilian Portuguese biomedical corpora	Dalloux, Clément and Claveau et al.	2021	The authors work with languages which have been poorly addressed up, namely Brazilian Portuguese and French. First it was developed new corpora for these two languages, which have been manually annotated for marking up the negation. Second, the authors propose automatic methods based on supervised machine learning approaches for the automatic detection of negation	The methods show to be robust in both languages (Brazilian Portuguese and French) and in cross-domain (general and biomedical languages) contexts. The approach is also validated on English data from the state-of-the-art: it yields very good results and outperforms other existing approaches
Negation scope delimitation in clinical text using three approaches: NegEx, PyConTextNLP and SynNeg	Tanushi, Hideyuki et al.	2013	The authors compared three different systems for negation detection in Swedish clinical text (NegEx, PyConTextNLP and SynNeg), which have different approaches for determining the scope of negation cues. NegEx uses the distance between the cue and the disease, PyConTextNLP relies on a list of conjunctions limiting the scope of a cue, and in SynNeg the boundaries of the sentence units, provided by a syntactic parser, limit the scope of the cues	The three systems produced similar results, detecting negation with an F1-score of around 80%, but using a parser had advantages when handling longer, complex sentences or short sentences with contradictory statements

3 Medical Ontologies

This section aims to describe available ontologies, how they can be used, how they are made, and if they follow any kind of standardization.

3.1 International Classification of Diseases

The International Classification of Diseases is one of the main epidemiological tools in a doctor day-to-day life. It shows how many known diseases there are, which are they, and classifies them by creating a code for each one of them. It eases the communication between healthcare professionals, and has the main purpose of monitoring the incidence and prevalence of each disease [26].

With the standardization, it is possible to analyse and determine which are the diseases, the symptoms, the complaints, external causes, abnormal aspects, and social circumstances. In this way, there is a guarantee of a more efficient treatment, while helping doctors and other healthcare professionals.

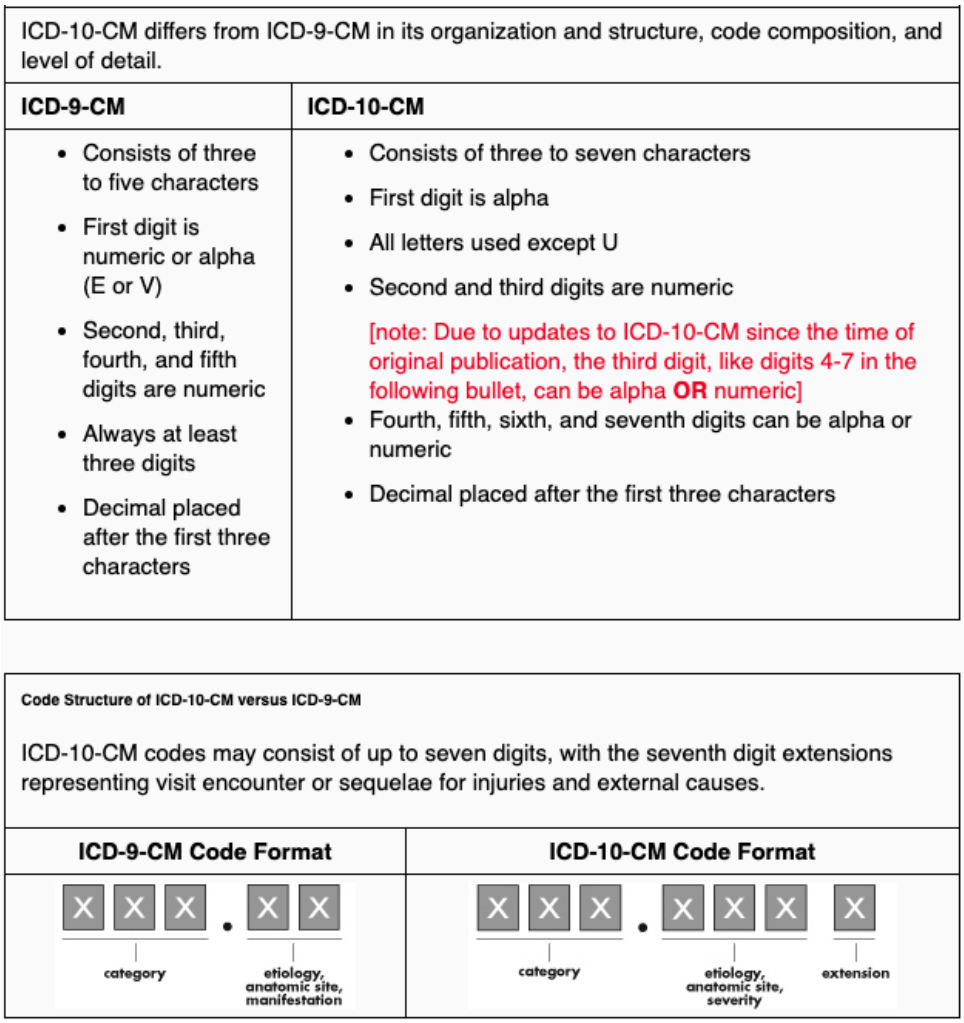
Diseases are grouped in 22 chapters, in which one of them is given an alphabet letter in ascending order, following A to Z. Inside those chapters, each pathology receives a different number, also in ascending order between 0 and 99. This standardization allows, in a general way, to monitor the quantity of cases inside each disease, and makes possible to create a broad overview of the global health situation.

ICD-10-CM (Clinical Modification) is used to classify diagnoses in inpatient and outpatient setting, while ICD-10-PCS (Procedure Coding System) is used to classify procedures in the inpatient setting [27].

ICD-10-CM, as with ICD-9-CM, is maintained by the National Center for Health Statistics. ICD-10-CM includes the level of details needed for morbidity classification and diagnostic specificity, and also provides code titles and language that complement accepted clinical practice.

ICD-10-CM consists of more than 68,000 codes, compared to approximately 13,000 ICD-9-CM codes. ICD-10-CM evidence more about the quality of care, and in this way, data can be used to improve the understanding of complications, improve the design of more robust algorithms, and improve the track of outcomes of care. It incorporates greater specificity and clinical detail to provide information for clinical decision making and research outcomes [28]. Figure 3.1 shows a comparison between ICD-9-CM

and ICD-10-CM.



ICD-10-CM differs from ICD-9-CM in its organization and structure, code composition, and level of detail.

ICD-9-CM	ICD-10-CM
• Consists of three to five characters • First digit is numeric or alpha (E or V) • Second, third, fourth, and fifth digits are numeric • Always at least three digits • Decimal placed after the first three characters	• Consists of three to seven characters • First digit is alpha • All letters used except U • Second and third digits are numeric [note: Due to updates to ICD-10-CM since the time of original publication, the third digit, like digits 4-7 in the following bullet, can be alpha **OR** numeric] • Fourth, fifth, sixth, and seventh digits can be alpha or numeric • Decimal placed after the first three characters

Figure 3.1: Comparing ICD-9-CM and ICD-10-CM[1]

3.1.1 ICD-10-CM structure

ICD-10-CM is similar to ICD-9-CM in terms of index and tabular list, however the ICD-10-CM index is much longer. For both, the index and tabular list is used as an indented format, being that categories, subcategories, and codes are contained in the tabular list.

The two parts of the ICD-10-CM index are the injury and external causes of injury. The table of drugs and chemicals and the neoplasm table are housed in the index to diseases and injury [29].

The first part of ICD-10-CM, the index of diseases and injury, consists of an alphabetical list of terms and respective codes. The sub-terms appear below the main term, namely:

- Alphabetical index of diseases and injuries;
- Alphabetical index of injury external causes;

[1]Source: https://www.cdc.gov/nchs/icd/icd10cm_pcs_background.htm, accessed on 2024-04-27

- Neoplasm table;
- Drug and chemical substances table.

The second part of ICD-10-CM, the tabular list of diseases and injuries, consists of a chronological list of codes divided into chapters, based on affected anatomical apparatus and types of conditions.

The codes can be formed by a minimum number of 3 characters until 4, 5, 6 or 7 characters. A subdivision of a category, is a subcategory, and the final subdivision is a code. The seven characters are codes and not subcategories [29].

This classification allows also, for some codes, for example:

1. Identify laterality, specifying if the condition occurs at left, right or bilateral. Examples:
   (a) H02401 - Unspecified ptosis of right eyelid;
   (b) H02402 - Unspecified ptosis of left eyelid;
   (c) H02403 - Unspecified ptosis of bilateral eyelids.
2. Identify the type of episode (initial, subsequent, ...). Examples:
   (a) M4842XA - Fatigue fracture of vertebra, cervical region, initial encounter for fracture.

The structure and format of diagnostic codes are:

- Minimum 3 characters;
- Maximum 7 characters;
- First character is always a letter (except U);
- The characters 2-7 can be letters or numbers (the second digit is numeric, and seventh digit is used only in some chapters);
- The characters do not have associated meaning;
- The characters are not case sensitive;
- Long descriptions until 250 characters;
- Short descriptions until 100 characters.

This structure allows a higher number of subcategories, and classification of laterality and bilaterality, as shown in Figure 3.2.

Table 3.1 shows the designations of each concept on the header of ICD-10-CM catalogue.

[2]Source: https://www.cdc.gov/nchs/icd/icd10cm_pcs_background.htm, accessed on 2024-04-27

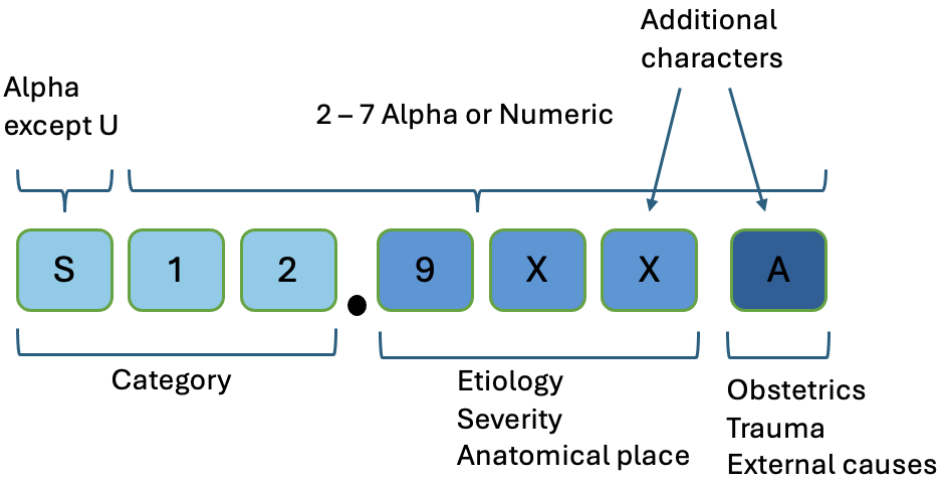


Figure 3.2: ICD-10-CM: structure and format[2]

Table 3.1: Catalogue structure of ICD-10-CM

Capítulo ICD-10-CM_Código	Identifies the range of codes of diagnostics of a Chapter
Capítulo ICD-10-CM_desc	Chapter description in English
Capítulo ICD-10-CM_desc_PT	Chapter description in Portuguese
Secção ICD-10-CM_Código	Identifies the range of codes of a Section of a Chapter
Secção ICD-10-CM_Desc	Description of a Section in English
Secção ICD-10-CM_Desc_PT	Description of a Section in Portuguese
Código ICD-10-CM	Code of diagnoses constituted by a minimum of 3 to 7 characters
Válido	Identifies "0" as aggregating codes not valid for record, and "1" as valid codes
Short_Descp ICD-10-CM	Short description in English
Long_Descp ICD-10-CM	Long description in English
Descrição PT_(Longa)	Long description of 2017 to 2022 versions translated to Portuguese
Descrição PT_(Curta)	Short description of 2017 to 2022 versions translated to Portuguese
Ano início	Year of the version in which the code was created
Ano fim	Year of the version in which the code was changed or eliminated. NOTE: the codes with end year, must not be chosen for record, because they may be in the system only to guarantee historic data
Versão	Indicates the month and year (mm_aaaa) where the description in Portuguese as received changes. NOTE: if this field is filled, it means that besides the code has not suffered changes, it occurred corrections or changes in the descriptions in Portuguese
Código versão anterior	Identifies the code that has been target of change
valido?	Identifies "0" the aggregating codes not valid for record, "1" as valid codes
Tipo Alteração	Identifies the type of change in a code (New; Changed description; Eliminated)
Código conversão 1	Identifies the new code(s) that came to replace the eliminated code
Código conversão N	Identifies the new code(s) that came to replace the eliminated code

3.1.2 Differences between ICD-10-CM and ICD-9-CM

The differences between ICD-10-CM and ICD-9-CM are in its organization and structure, code composition, and level of detail. ICD-10-CM codes may consist of up to seven digits, with the seventh digit extensions representing visit encounter or sequelae for injuries and external causes [30]. These differences are shown in Figure 3.1.

3.1.3 Organizational changes

ICD-10-CM and ICD-9-CM have the same type of hierarchical structure, but there are some differences in organization, including:

- ICD-10-CM consists of 21 chapters;
- Some chapters include the addition of a sixth character;
- ICD-10-CM includes full code titles for all codes (no references back to common fourth and fifth digits);
- V and E codes are no longer supplemental classifications;
- Sense organs have been separated from nervous system disorders;
- Injuries are grouped by anatomical site rather than injury category;
- Postoperative complications have been moved to procedure-specific body system chapter.

3.1.4 New features

ICD-10-CM has numerous new features to allow a greater level of specificity and clinical detail, including:

- Combination codes for conditions and common symptoms or manifestations;
- Combination codes for poisonings and external causes;
- Added laterality;
- Added extensions for episode of care;
- Expanded codes (injury, diabetes, alcohol/substance abuse, postoperative complications);
- Inclusion of trimester in obstetrics codes and elimination of fifth digits for episode of care;
- Expanded detail relevant to ambulatory and managed care encounters;
- Changes in timeframes specified in certain codes;
- External cause codes no longer a supplementary classification.

ICD-10-CM also includes added standard definitions for two types of exclusion notes. Excludes1 indicates not coded here. The code being excluded is never used with the code. The two conditions cannot occur together. For example, B06 Rubelia has an Excludes1 of congenital rubella (P35.0).

Excludes2 indicates not included here. The excluded condition is not part of the condition represented by the code. It is acceptable to use both codes together if the patient has both conditions. For example, J04.0, Acute laryngitis has an Excludes2 of chronic laryngitis (J37.0).

An additional feature is the expansion of codes for certain conditions. Two examples are diabetes mellitus and postoperative complication codes.

Diabetes mellitus codes are expanded to include the classification of the diabetes and the manifestation. The category for diabetes mellitus has been updated to reflect the current clinical classification of diabetes and is no longer classified as controlled or uncontrolled:

- E08.22, Diabetes mellitus due to an underlying condition with diabetic chronic kidney disease;
- E09.52, Drug or chemical induced diabetes mellitus with diabetic peripheral angiopathy with gangrene;
- E10.11, Type 1 diabetes mellitus with ketoacidosis with coma;
- E11.41, Type 2 diabetes mellitus with diabetic mononeuropathy.

ICD-10-CM provides 50 different codes for "complications of foreign body accidently left in body following a procedure", compared to only one code in ICD-9-CM. Examples include:

- T81.535, Perforation due to foreign body accidently left in body following heart catheterization;
- T81.530, Perforation due to foreign body accidently left in body following surgical operation;
- T81.524, Obstruction due to foreign body accidently left in body following endoscopic examination;
- T81.516, Adhesions due to foreign body accidently left in body following aspiration, puncture or other catheterization.

Code extensions (seventh character) have been added for injuries and external causes to identify the encounter: initial, subsequent, or sequels. The extensions are:

- A Initial encounter;
- D Subsequent encounter;

- S Sequelae.

For example, ICD-10-CM code S31.623A, Laceration with foreign body of abdominal wall, right lower quadrant with penetration into peritoneal cavity, initial encounter, shows an extension used with a laceration code. Note that in ICD-10-CM, the entire code description is written out.

Fracture codes require a seventh character that identifies if the fracture is open or closed for an initial encounter or if a subsequent encounter is for routine healing, delayed healing, nonunion, malunion, or sequelae. The fracture extensions are [31]:

- A Initial encounter for closed fracture;
- B Initial encounter for open fracture;
- D Subsequent encounter for fracture with routine healing;
- G Subsequent encounter for fracture with delayed healing;
- K Subsequent encounter for fracture with nonunion;
- P Subsequent encounter for fracture with malunion;
- S Sequelae.

An example is code S42.321A, Displaced transverse fracture of shaft of humerus, right arm, initial encounter for closed fracture.

Figure 3.3 shows ICD-10-CM code examples.

3.2 Systematized Medical Nomenclature for Medicine – Clinical Terminology

Systematized Medical Nomenclature for Medicine – Clinical Terminology (SNOMED CT) was developed to assign meaning or relevance to EHRs from health databases. It complements the existing terminologies as ICD-9 and ICD-10, that by nature represent only one dimension of meaning [32].

For example, a "Viral Pneumonia" in ICD-9 or ICD-10 is represented by J12 or J12.9, being classified as a respiratory disease, being this "respiratory disease" the only dimension represented in this record.

SNOMED CT adds several dimensions to this record through several relations like "is a", "finding site" and "causative agent". For example "viral pneumonia", "is an" infectious pneumonia, "is a" respiratory disease, "is an" infection, "finding site" in lungs, "causative agent" by virus, as it can be seen in Figure 3.4.

[3]Source: https://www.xn–tempo-semntico-jhb.pt/snomed-ct-nao-nos-podemos-dar-ao-luxo-nao-implementar/, accessed on 2024-04-27

ICD-10-CM Code Examples

ICD-10-CM consists of new features and greater specificity. Sample ICD-10-CM codes are outlined below to illustrate this increased detail.

Combination Codes for Conditions and Common Symptoms

- I25.110, Arteriosclerotic heart disease of native coronary artery with unstable angina pectoris
- K50.013, Crohn's disease of small intestine with fistula
- K71.51, Toxic liver disease with chronic active hepatitis with ascites

Combination Codes for Poisonings and the External Cause

- T39.011, Poisoning by aspirin, accidental (unintentional)
- T39.012, Poisoning by aspirin, intentional self harm
- T39.013, Poisoning by aspirin, assault
- T39.014, Poisoning by aspirin, undetermined

Laterality

- C50.212, Malignant neoplasm of upper-inner quadrant of left female breast
- H02.835, Dermatochalasis of left lower eyelid
- I80.01, Phlebitis and thrombophlebitis of superficial vessels of right lower extremity
- L89.213, Pressure ulcer of right hip, stage III

Figure 3.3: ICD-10-CM code examples[3]

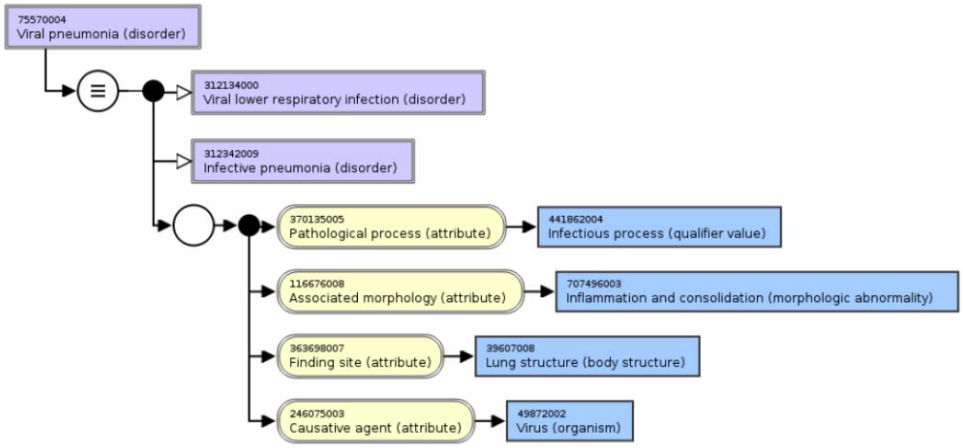


Figure 3.4: SNOMED example for viral pneumonia[4]

The adoption of the SNOMED CT terminology is specially beneficial regarding the ICD-9/10 classification. If there is already an ICD-9/10 classification, a mapping can be done, which can be one-to-one, or one-to-many. In the one-to-one mapping, the ICD-9 codes are identified and matching to each SNOMED CT concept, which allows a direct

[4]Source: https://www.xn–tempo-semntico-jhb.pt/snomed-ct-nao-nos-podemos-dar-ao-luxo-nao-implementar/, accessed on 2024-04-27

correspondence between two terminologies without having lost or misunderstandings of meanings. In the one-to-many mapping, it is not always possible to do a direct match between these two terminologies, although a NEC (Not Elsewhere Classified) map "catch-all" is made available.

The SNOMED CT Snapshot REST API [5] provides access to SNOMED CT content snapshot form. This API has been developed by SNOMED International to support lightweight deployments of read-only SNOMED CT terminology servers.

3.3 Open Biomedical Ontologies Foundry

Open Biomedical Ontologies Foundry, is a project that involves people connected to the development of ontologies in the biomedical science domain. In their site [6], there are more than 50 ontologies, which can be downloaded in Web Ontology Language (OWL) format. Still related to this work, there is the Infectious Disease Ontology (IDO) [7], a set of interoperable ontologies, which cover the domain of infectious diseases. Among other ontologies there are the brucellosis, flu, malaria, tuberculosis and vaccines.

3.4 Summary

ICD is an ontology used in doctors' day-to-day life. It is grouped in 22 chapters, which are given an alphabet letter in ascending order. Inside the chapters, each pathology is associated with a number, between 0 and 99, also in ascending order.

ICD-10-CM stands for Clinical Modification, used to classify diagnoses in both inpatients and outpatients. The differences between ICD-10-CM and ICD-9-CM are in the organization and structure, code composition, and level of detail.

SNOMED is another ontology, that was developed to assign meaning or relevance to EHRs from health databases. It is a complement of ICD-10 and ICD-9, that by nature represent only one dimension of meaning. If there is already an ICD-9/10 classification, a relation one-to-one, or one-to-many, can be done. SNOMED has a REST API, that provides access to content snapshot form.

Open Biomedical Ontologies Foundry is a project with more than 50 ontologies, in the biomedical science domain, that can be downloaded in OWL format. IDO is an ontology which covers the domain of infectious diseases.

[5] https://snomedctsnapshotapi.docs.apiary.io/, accessed on 2023-10-07
[6] https://obofoundry.org/, accessed on 2023-06-01
[7] https://bioportal.bioontology.org/ontologies/IDO, accessed on 2024-03-22

4 Theoretic fundamentals

In this chapter, some theoretic fundamentals are explained, namely state-of-the-art NLP machine learning models, along with parsing, text transformations, and NER.

4.1 Models

Regarding NLP basic architectures, the following are emphasized, described as the state-of-art in this field, namely Recurrent Neural Networks, Convolutional Neural Networks, Autoencoders, and Sequence-to-Sequence models.

4.1.1 Convolutional Neural Network

A Convolutional Neural Network, in the field of deep learning, is a class of deep neural networks, that are often used to analyze images, using a special technique called Convolution [33]. Convolution is an operation on two functions that originate a third function that traduces on how the shape of one is modified by the other, as can be seen in Figure 4.1.

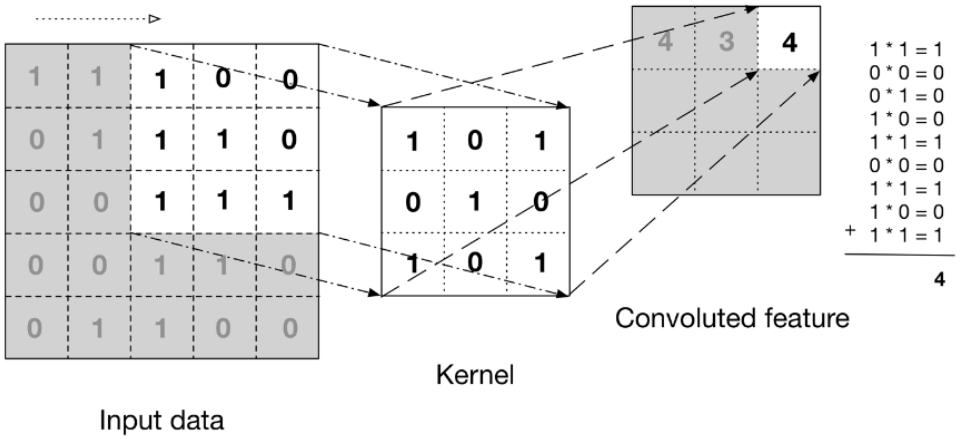


Figure 4.1: Representation of a Convolution[1]

The convolution represented in Figure 4.1, takes a filter/kernel (3x3 matrix) and applies it to the input image to get the convolved feature, where it is passed on to the next layer.

[1]Source: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/, accessed on 2024-04-26

The Pooling Layer, as the Convolutional Layer, has the responsibility of reducing the size of the Convolved Feature. This is done to reduce the computational power of the processing of data, while reducing the dimensions [33].

There are some limitations of CNN, such as when it comes to understanding the contents of an image.

4.1.2 Autoencoder

This architecture is classified as unsupervised neural network, compressing and encoding data, and learning the reconstruction of data from the reduced encoded representation, as much as possible close to the original input. This model, besides reducing data dimension, it also ignores the noise in it [34].

Autoencoders have four main components:

- **Encoder:** it learns the reducing of input dimensions, and compresses the input data into an encoded representation;
- **Bottleneck:** is responsible to hold the compressed representation of the input data;
- **Decoder:** the model learns how to reconstruct the data from the encoded representation, in a way that is intended to be as close as possible to the original input;
- **Reconstruction Loss:** measures how close the output is to the original input, and the training uses back propagation in order to minimize it.

Figure 4.2 shows an example of an autoencoder.

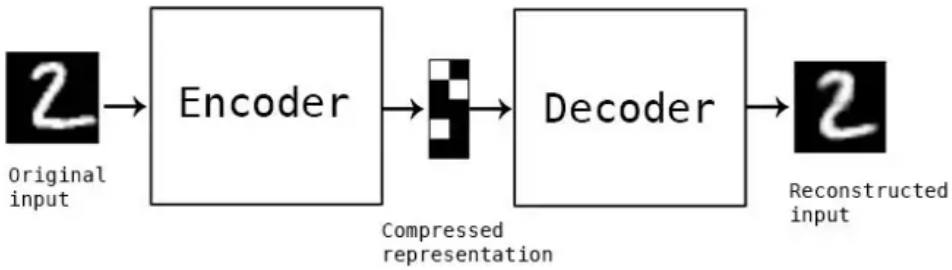


Figure 4.2: Example of an autoencoder[2]

4.1.3 Sequence-to-Sequence

Sequence-to-Sequence models, also know has Seq2Seq, is a special class of RNN architectures that are used to solve complex problems like machine translation, ques-

[2]Source: https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726, accessed on 2024-04-03

tion answering, creating chatbots, text summarization, etc. It can be used to solve any sequence-based problem, mainly when the inputs and outputs have different sizes and categories [35].

Encoder-Decoder is the most common architecture used to build Seq2Seq models. Both encoder and the decoder are Long Short-Term Memory (LSTM) models, or sometimes Gated Recurrent Unit (GRU) models.

The input sequence is read by the encoder and then it summarizes the information in the internal state vectors, where for the LSTM, they are called the hidden state and cell state vectors. The outputs of the encoder are discarded, keeping only the internal states in order to help the decoder to make accurate predictions.

On the other hand, decoder is an LSTM where the initial states are initialized to the final states of the encoder LSTM. With these initial states, the decoder starts generating the output sequence, and taken into consideration for future outputs. Each recurrent unit accepts an hidden state from the previous unit, producing an output as well as its own hidden state.

Figure 4.3 shows the overall encoder-decoder architecture.

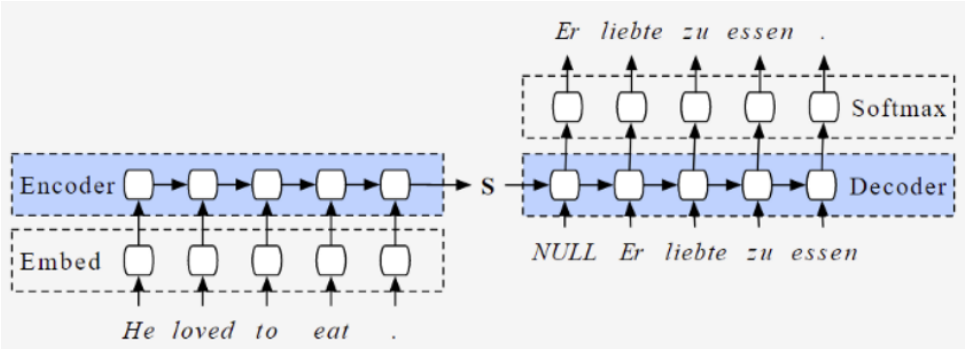


Figure 4.3: Overall encoder-decoder architecture [3].

In summary, during inference, one word at a time is generated, and the initial states of the decoder are defined to the final states of the encoder. The initial input to the decoder is always the START token, and at each step, the predicted output is fed as input in the next step. The loop stops when the END token is predicted by the decoder.

This architecture has two main drawbacks, regarding the length. The first is that this architecture has limited memory, and second, it is hard to train deep neural networks. For the case of RNNs, the longer the sequence is, the deeper the network is, resulting in vanishing gradients.

Furthermore, for more robust and lengthy sentences, there are models like Attention Models and Transformers.

[3]Source: https://www.analyticsvidhya.com/blog/2020/08/a-simple-introduction-to-sequence-to-sequence-models/, accessed on 2024-04-08

4.1.4 PLM-ICD

PLM-ICD [19] is a proposed framework to fine-tune pretrained language models for ICD coding. To overcome challenges of PLMs, mechanisms are proposed to tackle them, namely domain-specific pretraining for the domain mismatch problem, segment pooling for the long input sequence problem, and label attention for the large label set problem. Figure 4.4 presents the architecture of PLM-ICD.

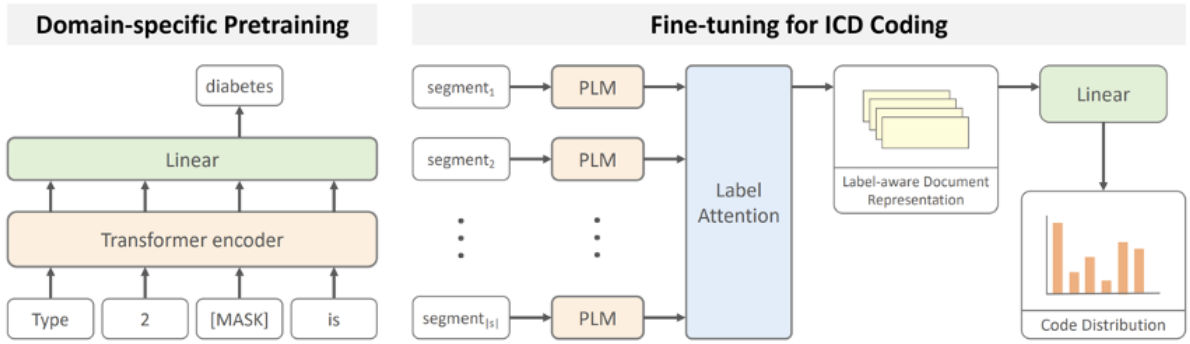


Figure 4.4: PLM-ICD architecture[4]

By default, the following PLMs can be used in PLM-ICD: BioLM [5], BioBERT [6] and PubMedBERT [7]. PLM-ICD is applied to the MIMIC-II and MIMIC-III datasets, being the first Transformer-based pretrained language model that achieves competitive performance on the MIMIC datasets.

4.2 Parsing

Parsing is the process of analyzing the sentence for its structure, content and meaning, i.e., to uncover the structure, articulate the constituents and the relation between the constituents of the input sentence. In other words, parsing is the process of breaking down the sentence into its constituent words in order to find out the grammatical type of each word or alternatively to decompose an input into more easily processed components [36].

English language provides eight part-of-speech, namely article, noun, pronoun, verb, adverb, adjective, preposition, and conjunction. For example in the sentence, "John is playing game" part-of-speech for each token is "noun" for "John" and "game", "verb" for "is" and "playing". Making a disambiguation decision means finding the correct part-of-speech for a word having multiple part-of-speeches, which give rise to "ambiguity". Ambiguity means having more that one interpretation of a word or a sentence.

[4]Source: https://github.com/MiuLab/PLM-ICD, accessed on 2024-04-09
[5]https://github.com/facebookresearch/bio-lm, accessed on 2024-04-08
[6]https://github.com/dmis-lab/biobert, accessed on 2024-04-08
[7]https://huggingface.co/microsoft/BiomedNLP-BiomedBERT-base-uncased-abstract, accessed on 2024-04-08

Example "book", it can be "noun" or "verb", depending upon its use, parsing is used to find the correct parse for a word or a sentence.

The structure of a sentence is: Sentence = S = Noun Phrase + Verb Phrase + Preposition Phrase (S = NP + VP + PP) [37]. According to English grammar rules, the different word groups that exist are:

- Noun Phrase (NP): Determiner + Nominal Nouns = DET + Nominal;
- Verb Phrase (VP): Verb + range of combinations;
- Prepositional Phrase (PP): Preposition + Noun Phrase = P + NP.

There are different forms and structure versions of the noun phrase, verb phrase, prepositional phrase and join in a sentence, that can be made. For instance, the sentence "The boy ate the pancakes", has the following structure:

- The boy: Noun Phrase;
- ate: Verb;
- the pancakes: Noun Phrase (determiner + Noun).

The preposition "under" is followed by the noun phrase "the door", being syntactically correct but not contextually.

Taking the same sentence in another way: "The boy ate the pancakes from the jumping table":

- The boy: Noun Phrase;
- ate: Verb;
- the pancakes: Noun Phrase (Determiner + Noun);
- from: preposition;
- jumping table: Verb Phrase.

Because of the preposition "from" followed by a verb phrase "jumping table", the sentence is syntactically incorrect.

Syntactic analysis or syntax analysis, are the other names for Parsing, where it checks the text for meaningfulness [37]. For instance, the sentence "Give me hot ice-cream", would be rejected by the syntactic analyzer. Parsing is the process of analyzing the strings in natural language, according to the rules of formal grammar, as it can be seen in Figure 4.5.

The relevance of parsing in NLP can be understood as follows:

- Parser is used to report any syntax error;

[8]Source: https://www.analyticsvidhya.com/blog/2022/03/syntactical-parsing-in-nlp/, accessed on 2024-04-09

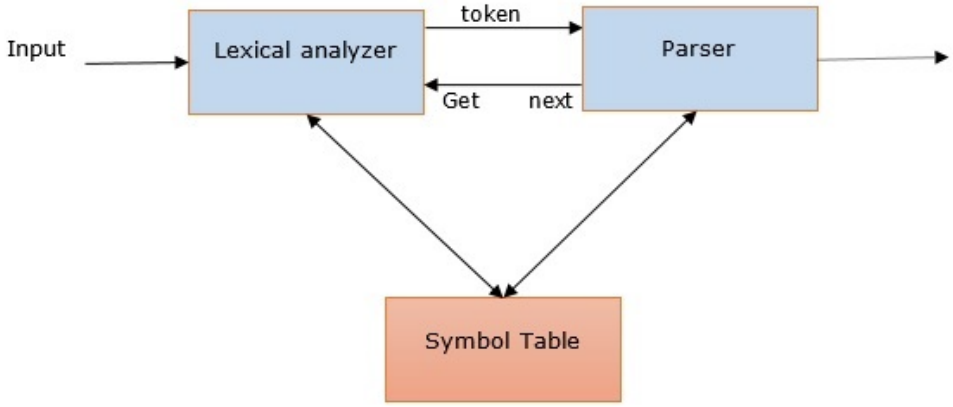


Figure 4.5: Parsing process [8]

- It helps to recover from commonly occurring error so that the processing of the remainder of the program can be continued;
- Parse tree is created with the help of a parser;
- Parser is used to create symbol table, which plays an important role in NLP;
- Parser is also used to produce intermediate representations (IR).

Table 4.1 shows a comparison between deep and shallow parsing.

Table 4.1: Comparison between deep and shallow parsing

Deep Parsing	Shallow Parsing
The search strategy, in deep parsing, will provide a complete syntactic structure to a sentence	It parses a limited part of the syntactic information
It is suitable for complex NLP applications	It can be used for less complex NLP applications
Examples where deep parsing is used, are the dialogue systems and summarising	Examples where shallow parsing is used, are the information extraction and text mining
It is also called full parsing	It is also called chunking

There are two imperative attributes of text syntax: Part of Speech tags and Dependency Grammar. Part of Speech tagging or POS tagging, defines the attribute of the word, being associated with a part of speech tag like noun, verb, adjective, adverb.

POS does not tag phrases, only individual words, hence is not enough to create a parse tree. These parse trees need that the noun phrase, verb phrase, or prepositional phrase, are in a particular order. Syntactic parsing refers to the analysis of words in a sentence for grammar and their ordering, in a manner that shows the relations between the words. Dependency grammar is part of syntactic text analysis, where it evidences the relationship between the words in a sentence. These relationships are represented in the form of a triplet, namely relation, governor, and dependent. A natural language parser is an application that finds out which group of words go together, and which words are the subject of a verb. This parser separates the text into smaller chunks based on the grammar rules. If a sentence cannot be parsed, that means it may have grammatical errors.

4.2.1 Various types of parsers

Parser can be defined as a procedural interpretation of grammar, that searches for an optimal tree of a given sentence, after searching the space of a variety of trees. The types of parsers are listed below [36].

Recursive descent parser

Recursive descent parser is one straightforward form of parsing:

- It follows a top-down process;
- It attempts to verify whether the syntax of the input stream is correct or not;
- It reads the input sentence from left to right;
- It needs to read characters from the input stream, and then it matches with the terminals from the grammar.

Shift-reduce parser

Shift-reduce parser has some important points:

- It follows a simple bottom-up process;
- Tries to search for a sequence of words and sentences that match the right-hand side of a grammar production, while replacing them with the left-hand side;
- The above attempt to find a sequence of word continues until the whole sentence is reduced;
- In summary, this parser starts with the input symbol and it builds the parser tree up to the start symbol.

Chart parser

Chart parser has the following important points:

- It is mainly useful or suitable for ambiguous grammars, including grammars of natural languages;
- It applies dynamic programming to the parsing problems;
- Because of dynamic programming, partial hypothesized results are stored in a structure called a chart;
- The chart can also be re-used.

Regexp parser

Regexp parsing is one of the mostly used parsing techniques. Following are some important points about Regexp parser:

- It uses a regular expression defined in the form of grammar on top of a POS-tagged string;
- It uses these regular expressions to parse the input sentences and generate a parse tree out of this.

Dependency parsing

Dependency parsing (DP), is a novel mechanism, where each linguistic unit relates to each other over a direct link. These direct links are actually linguistic dependencies. For example, the following Figure 4.6 shows dependency grammar for the sentence "John can hit the ball".

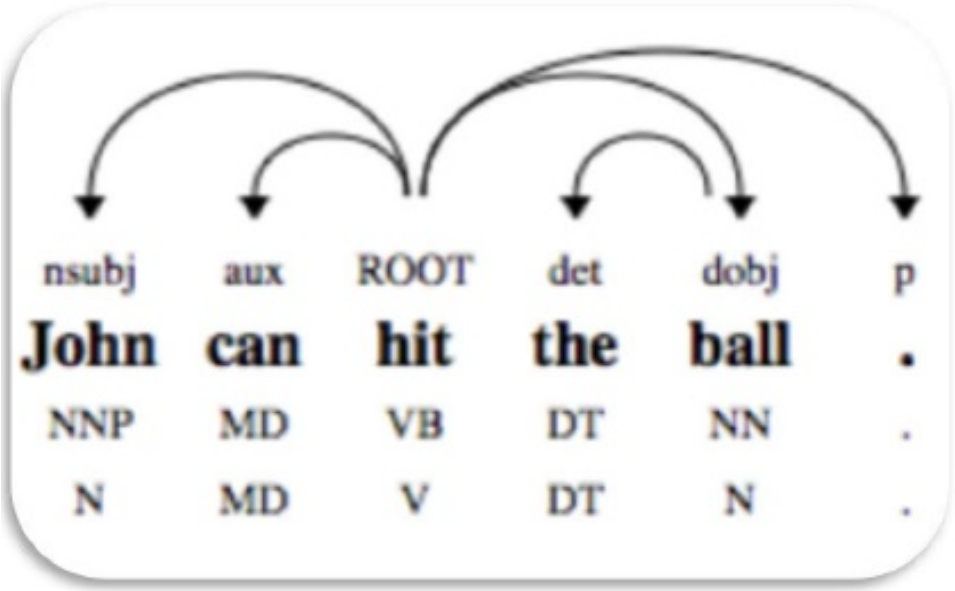


Figure 4.6: Dependency parsing[9].

4.3 Text transformations

The previously described model architectures only work with numeric tensors, so some important steps need to be done before applying the data to a model, namely tokenization, data preprocessing, and encoding.

4.3.1 Tokenization

The process of splitting a piece of text into smaller units, is called tokenization, that will be used to feed the network [38]. These tokens will be used to set up a vocabulary,

[9]Source:https://www.tutorialspoint.com/natural_language_toolkit/natural_language_toolkit_parsing.htm, accessed on 2024-04-10

which is a set of unique tokens in the corpus. One way to boost the performance of the NLP model, is to create a vocabulary out of the top-K frequently occurring words. Traditional approaches such as Count Vectorizer and Term Frequency-Inverse Document Frequency (TF-IDF) use vocabulary as features. Each word in the vocabulary is treated as a unique feature [38].

Tokenization can be classified into three types, namely word, character, and subword tokenization.

One of the most used tokenization algorithms is the word tokenization. In this algorithm, a piece of text is split into individual words, based on a delimiter. Examples are pretrained word embeddings like Word2Vec and Glove, but they come with some drawbacks. One issue is the fact on how to deal with Out Of Vocabulary (OOV) words, but it can be overcome with the formation of the vocabulary with the top-K frequent words and replace the rare words with unknown tokens (UNK). Another issue might arise with the size of the vocabulary, because building the vocabulary with all the unique words in large corpus, may explode the vocabulary. This opens the door to character tokenization.

Character tokenization splits a text into a group of characters, and removes the drawbacks of word tokenization, namely the OOV while keeping the information of the word. It splits the OOV word into characters representing the word in terms of these characters. Limits also the length of the vocabulary, by reducing to a unique group of characters. The limitations of character tokenization, besides solving the OOV problem, are the rapid increase of the length of the input and output sentences, because it represents a sentence as a sentence of characters. in this way, it turns challenging to learn the relationship between the characters to form words with meaning. This leads to another tokenization, known as subword tokenization, which is in between word and character tokenization.

Subword tokenization splits the text into subwords (or n-gram characters), where words like “lower”, are splitted as “low-er”, or “smartest” as “smart-est”. The Byte Pair Encoding (BPW), is one popular subword tokenization algorithm, where it overcomes the OOV, while segmenting OOV as subwords, representing the word in terms of these subwords. Compared to character tokenization, the length of input and output sentences are shorter.

4.3.2 Preprocessing

Another important text transformation is the data preprocessing, that includes dataset cleaning, lowercase conversion, removal of stop words, text normalization, augmentation, part-of-speech tagging, sentence parsing, and stemming/lemmatization [39].

For the lowercasing, it is one of the simplest and most effective forms of text preprocess-

ing, helping in cases where the size of the dataset is not very large, and also improves the consistence of the expected output [39].

Concerning stopword removal, stop words are a group of commonly used words. These stop words, in English, can be “a”, “the”, “is”, “are”. By removing stop words, it removes the low information and the focus stays on the important words instead.

The process of transforming a text into a standard form, is called text normalization. As an example, the word “gooood” and “gud” can be transformed to “good’. Other example is the map of near identical words like “stopwords”, “stop-words” and “stop words”, to just “stopwords”.

Augmentation involves augmenting the original text with information that did not exist before. It provides more semantics to the original text, thereby improving its predictive power and the depth of analysis that can be performed on the data.

Stemming is the process of the inflection reducing in words (e.g. troubled, troubles) into their root form (e.g. trouble). In this case, “root” may not be a real root word, being just a canonical form of the original word.

Lemmatization is very similar to stemming, and the purpose is to remove inflections, while mapping a word into its root form. Lemmatization tries to do it in the proper way, compared to stemming. It does not only chop things off, but it really transforms words to the actual root. As an example, the word “better” would map to “good”.

Part-of-speech tagging consists in identifying for each word its morphosyntactic class based on its context and lexical knowledge. This task provides more semantics to the original text and enables to get more granular information about its words.

Sentence parsing tries to define sub-components such as name entity and phrases in the sentence. It uses part-of-speech tagging as input and provides chunks as output. In this context, sentence parsing is very important because nanotoxicological entities are generally composed of words or noun phrases

4.3.3 Encoding

The last important text transformation is about encoding. Token encoding can be obtained with one-hot encoding, integer encoding, and word embeddings [40].

One-hot encoding creates binary vectors with length equal to the vocabulary, where each bit will be assigned to a specific word. The drawbacks of this approach are the sparse encoding, difficulty to handle large vocabularies, and it does not capture relationships between words.

On the other hand, integer encoding encodes each word with a unique number. The drawbacks are the dense encoding, it does not capture relationships between words, and the model may wrongly assume similarities/distances between words that do not

exist.

Finally, word embedding is a dense representation of words in the form of numeric vectors. It can be learned using a variety of language models. The word embedding representation is able to reveal many hidden relationships between words. For example, vector("cat") - vector("kitten") is similar to vector("dog") - vector("puppy"). The advantages of this method are the dense vector of floating values that represents a word, efficient dense representation, it encodes relationships between words, and the number of dimensions is a hyperparameter. It maps human language to a numerical geometric space, where synonyms should be embedded into similar vectors, and geometric distance should be related to semantic difference.

To obtain a word embedding, there are two possibilities. First, start with random vectors and learn word embeddings jointly with the main task, where in this case we need training data and time. Second, use a pretrained word embedding, where there are precomputed embedding spaces that capture generic aspects of language structure.

4.4 Named Entity Recognition

Named Entity Recognition is a supervised learning method. It identifies entities and data items with the support of a training model. For example, the architecture of Spacy's NER model is based on Convolutional Neural Networks with tokens represented as hashed Bloom embeddings of prefix, suffix and lemmatisation of individual words augmented with a transition-based chunking model.

NER is the task of determining whether tokens or spans in a text correspond to certain "named entities" of interest, such as medications and diseases. As a sub-task of NER, clinical concept extraction seeks to identify medical concepts, such as treatments and drug names. In earlier years, challenges for concept extraction tasks were primarily won by hybrid approaches, which combined rule-based and machine learning-based approaches. Similar to NER, people have framed clinical concept extraction as a sequence tagging problem and commonly use LSTMs and/or CRF (Cibersecurity Regulatory Framework) frameworks.

4.5 Summary

There are state-of-the-art NLP architectures in the field. CNN is a class of deep neural networks, using a special technique called Convolution. Autoencoder is an unsupervised neural network that compresses and encodes data, learning the reconstruction of data from the reduced encoded representation. Sequence-to-Sequence models are a special class of RNN architecture that solves language problems like machine translation, question answering, chatbots and text summarization. PLM-ICD is a framework

of Transformer-based pretrained language model that has competitive performance on the MIMIC datasets, that solves the domain mismatch problem, segment pooling for the long input sequence problem, and label attention for the large label set problem.

Parsing is a technique to analyze the structure of a sentence, content and meaning, articulating the constituents and the relation between them. There are different types of parsers, namely recursive descent parser, shift-reduce parser, chart parser, regex parser which is one of the most used parsing techniques.

Text transformations can be for example the tokenization which is the process of separating a piece of text into smaller units called tokens. Preprocessing is another important text transformation, that includes dataset cleaning, lowercase conversion, removal of stop words, text normalization, augmentation, part-of-speech tagging, sentence parsing and stemming/lemmatization. Finally, encoding is the last important text transformation that can be obtained with one-hot encoding, integer encoding and word embeddings.

NER identifies entities and data items with the support of a training model. In a clinical concept, it seeks to identify medical concepts, such as treatments and drug names.

5 Datasets used

The ICD-10 ontology was used in the present research. An unlabeled english dataset from Harvard Medical School was also used mostly for the developing phase of the project. For the main tests, a subset of MIMIC-IV dataset was used.

5.1 ICD-10

ICD is a medical coding system, also known as medical ontology, that is popular and used worldwide [41].

The 10^{th} revision of ICD was used in this work. ICD-10-CM is a seven-character, alphanumeric code. Every code starts with a letter, followed by two numbers. The first three characters of ICD-10-CM are the "category", and describes the general type of disease or injury, followed by a decimal point and the subcategory. Then it is followed by up to two subclassifications, that explain the cause, manifestation, location, severity, and type of disease or injury. The last character is the extension [42]. The extension describes the type of encounter, namely if it is the first time the healthcare has seen the patient, for the disease/injury/condition, it is named as the "initial encounter". After the first encounter, it is listed as "subsequent encounter". If the patient visit is related to a previous disease or injury, it is listed with term "sequela".

Figure 5.1 shows the format of ICD-10-CM.

Based on the type of disease or injury they document, ICD-10-CM is divided into ranges. Figure 5.2 shows the possible ranges and topics for ICD-10-CM.

Figure 5.3 shows an example of the levels of detail for the ICD-10 codes.

As an example, Chen *et al.* (2021)[10] used diagnosis records of the Taiwan University Hospital, applying NLP techniques, on the deep neural network architecture, using the attention mechanism to extract keywords from diagnoses, for ICD-10 encoding.

The main differences between ICD-10 and the latest version ICD-11, includes 28 chapters compared to the 22 chapters in ICD-10. The ICD-11 codes have a different, more detailed categorization and structure than ICD-10 codes. There are more than 55,000 ICD-11 codes versus the 14,000 in ICD-10. In ICD-11, extension codes replace the ICD-10 adjunct codes, that can be used to describe the dimensions of an injury or other ex-

[1]Source: https://www.medicalbillingandcoding.org/icd-10-cm/, accessed on 2024-04-17
[2]Source: https://www.medicalbillingandcoding.org/icd-10-cm/, accessed on 2024-04-17
[3]Source: https://www.medicalbillingandcoding.org/icd-10-cm/, accessed on 2024-04-17

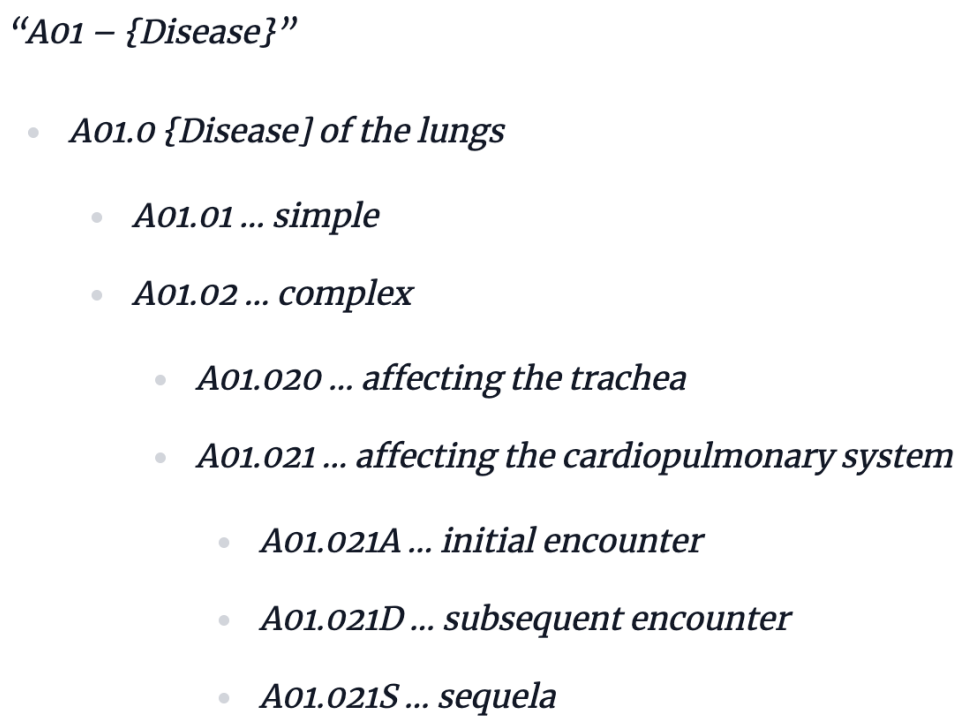


Figure 5.1: Format of ICD-10-CM[1]

ternal causes. ICD-11 also allows for cluster coding, which means that multiple codes can be combined to describe a diagnosis in detail.

5.2 Portuguese dataset

The initial idea of this work was to deal with a Portuguese dataset, due to the lack of investigation related to ICD codification in this language. Contacts were made with the Infectiology Unit of Coimbra Hospital and University Centre (CHUC). Due to the sensible and the privacy of data involved, some steps are required, namely the approval from the Ethical Commission of CHUC. The ethical and legal issues were solved and access was granted to the data. However, the whole process took more than six months and when the approval process was completed it was deemed too late to use the Portuguese data in the scope of the current project. Hence, the results presented were obtained with just the MIMIC dataset, described in Section 5.4.

5.3 Harvard Medical School dataset

Although the main goal was initially to work with a Portuguese dataset, we requested access to a dataset from the Harvard Medical School (HMS) [4], for investigation purposes, and this was granted. In order to test the algorithm on codification and translate

[4]Source: https://portal.dbmi.hms.harvard.edu/projects/n2c2-nlp/, accessed on 2023-10-10

RANGE	TOPIC
A00-B99	Certain infections and parasitic diseases
C00-D49	Neoplasms
D50-D89	Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism
E00-E89	Endocrine, nutritional and metabolic diseases
F01-F99	Mental, Behavioral and Neurodevelopmental disorders
G00-G99	Diseases of the nervous system
H00-H59	Diseases of the eye and adnexa
H60-H95	Diseases of the ear and mastoid process
I00-I99	Diseases of the circulatory system
J00-J99	Diseases of the respiratory system
K00-K95	Diseases of the digestive system
L00-L99	Diseases of the skin and subcutaneous tissue
M00-M99	Diseases of the musculoskeletal system and connective tissue
N00-N99	Diseases of the genitourinary system
O00- O9A	Pregnancy, childbirth, and puerperium
P00-P96	Certain conditions originating in the perinatal period
Q00-Q99	Congenital malformations, deformations and chromosomal abnormalities
R00-R99	Symptoms, signs, and abnormal clinical laboratory findings, not elsewhere classified
S00-T88	Injury, poisoning, and certain other consequences of external causes
V00-Y99	External causes of morbidity
Z00-Z99	Factors influencing health status and contact with health services

Figure 5.2: Ranges of ICD-10-CM[2]

it into metrics, 18 EHRs were sent to a coder doctor, Dr[a] Helena Alves, from CHUC. Samples from this HMS dataset were used at an early stage of the work. However, for more training of deep learning models a large dataset was required, thus the option was to try the MIMIC dataset, as described below in Section 5.4.

5.4 MIMIC-IV dataset

There are not many healthcare labeled datasets, in English, available for public use. Nonetheless, MIMIC-IV dataset was a viable solution. This dataset contains data of patients that were admitted to the Beth Israel Deaconess Medical Center Emergency Department or ICU between 2008 and 2019, annotated with ICD-9 and ICD-10 codes [43]. MIMIC-IV was released on 6 January 2023, and it is increasingly more popular for training and testing applications for automatic medical encoding. MIMIC-IV contains information for each patient (257,000 distinct patients), namely the hospital

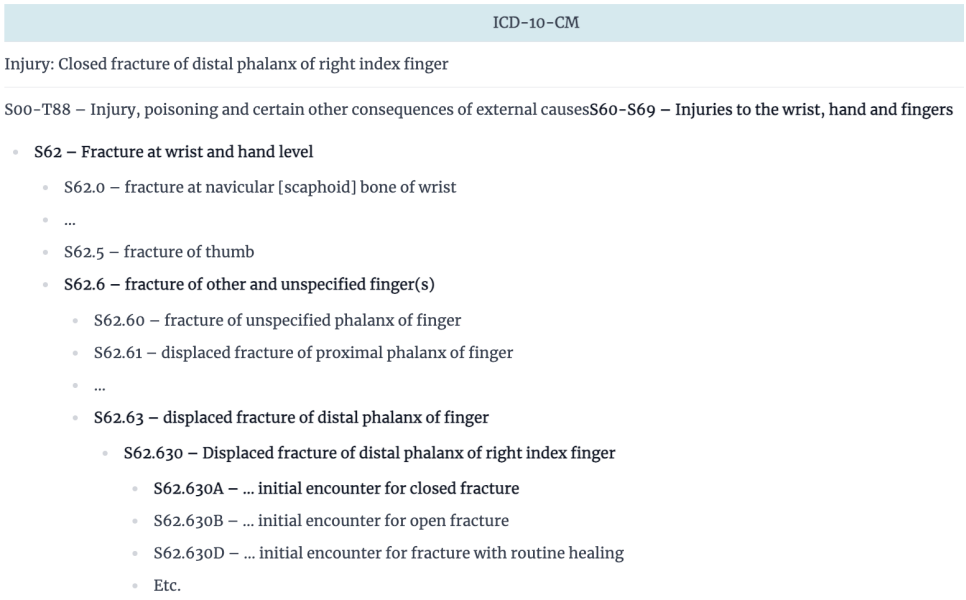


Figure 5.3: Example of levels which ICD-10 codes[3]

stay, laboratory measurements, medication, and vital signs. It is composed by 524,000 admission records [44].

MIMIC-IV adopts a relational structure with predefined relationships stored across tables and columns. Data are grouped into three modules: "hosp", "icu", and "note". Admission/discharge/transfer (ADT) records are placed in the "hosp" module, as well as other hospital data such as laboratory values, microbiology cultures, medication orders, and administrative data from hospital billing practices [43].

The "icu" module comprises data documented at the ICU bedside. Data includes intravenous infusions, patient outputs, charted observations, and documentation of ongoing procedures. The "note" module comprises discharge summaries and radiology reports.

In this dataset, subject id identifies a patient, while hadm id identifies an admission of a patient to the hospital. For example, the following text is part of a MIMIC-IV EHR, with subject id "1000017" and hadm id "22927623": "She describes feeling as though food gets stuck in her neck when she eats. She put herself on a pureed diet to address this over the last 10 days. When she has food stuck in the throat, she almost feels as though she cannot breath, but she denies trouble breathing at any other time. She does not have any history of food allergies or skin rashes." This excerpt shows an example of a text that is somehow vague concerning the diagnosis and negated terms (no food allergies or skin rashes). Those are common difficulties when coding.

To obtain MIMIC-IV dataset, first was visited the web page of Physionet [5]. Physionet is a web-based resource designed to support current research and stimulate new investi-

[5]https://physionet.org/content/mimiciv/2.2/, accessed on 2024-02-21

gations in the study of complex physiologic and clinical data.

As a requirement, a short course is mandatory to obtain the dataset, namely the training CITI Data or Specimens Only Research [6], which covers subjects like:

- Defining Research with Human Subjects;
- Privacy and Confidentiality;
- Assessing Risk;
- Research with Children;
- International Research;
- History and Ethical Principles;
- Regulations and Process;
- Social and Behavioural Research Methodologies in Biomedical Research;
- Genetics Research;
- Records-Based Research;
- Populations in Research Requiring Additional Considerations and/or Protections;
- Health Insurance Portability and Accountability Act and Human Subjects Research;
- Conflicts of Interest in Research Involving Human Subjects.

Before the training process, an account needs to be created in order to enroll to the course [7].

Table 5.1 shows a summary of characteristics and statistics of MIMIC-IV ICD-10 dataset [20]. The number of unique codes of MIMIC-IV ICD-10 (7942), evidences an imbalanced dataset, regarding over 69,000 ICD-10-CM diagnosis codes.

Table 5.1: Summary of characteristics and statistics of MIMIC-IV ICD-10 dataset

MIMIC-IV ICD-10	
Number of documents	122,279
Number of patients	65,659
Number of unique codes	7942
Codes pr. instance: Median (IQR)	14 (9 - 20)
Words pr. document: Median (IQR)	1492 (1147 - 1931)
Documents: train/val/test [%]	60/20/20

For the present work, MIMIC-IV dataset was split into a training set composed by 73,381 discharge summaries (60%), a validation set consisting of 24,461 records (20%),

[6] https://physionet.org/content/mimiciv/view-required-training/2.2/#1, accessed on 2024-02-21
[7] https://www.citiprogram.org/index.cfm?pageID=14&message=64, accessed on 2024-02-21

and a test set consisting of 24,461 records (20%), following the partition method proposed in [45].

For the sake of time, only a fraction of $\frac{1}{6}$ of the test set (4077 out of the 24,462) samples were used in the experiments.

5.5 Summary

ICD is a medical diagnostic and procedure coding system that was developed by WHO in the 1970s, which has undergone several major revisions along the years. The 10^{th} revision of ICD was used in this work. ICD-10-CM is a seven-character, alphanumeric code. Every code starts with a letter, followed by two numbers. The first three characters of ICD-10-CM are the "category", and describe the general type of disease or injury, followed by a decimal point and the subcategory. Then it is followed by up to two subclassifications, that explain the cause, manifestation, location, severity, and type of disease or injury. The last character is the extension. The extension describes the type of encounter this is.

The initial idea of this work was to deal with a Portuguese dataset, because there is lack of investigation related to ICD codification in this language. Contacts were made with the Infectiology Unit of CHUC, where the ethical and legal issues to have access to the dataset were solved. However, the whole process took more than six months and when the approval process was completed it was deemed too late to use the Portuguese data in the scope of the current project.

It was requested access to a dataset from HMS, for investigation purposes, and this was granted. In order to test the algorithm on codification and translate it into metrics, 18 EHRs were sent to a coder doctor, Dr[a] Helena Alves, from CHUC. However, for more training of deep learning models a large dataset was required, thus the option was to try the MIMIC dataset.

MIMIC-IV dataset contains data of patients that were admitted to the Beth Israel Deaconess Medical Center emergency department of ICU between 2008 and 2019, annotated with ICD-9 and ICD-10 codes. This dataset contains a number of 7942 codes, evidencing an imbalanced dataset, regarding over 69,000 ICD-10-CM diagnosis codes. MIMIC-IV dataset was split into a training set composed by 73,381 discharge summaries (60%), a validation set consisting of 24,461 records (20%), and a test set consisting of 24,461 records (20%). For the sake of time, only a fraction of $\frac{1}{6}$ of the test set (4077 out of the 24,462) samples were used in the experiments.

6 Methodology

This section describes the used methodology.

6.1 Software and hardware

The software used was the Python, version 3.9.12, programming environment with spaCy, an NLP library that provides support for different text processing operations. A non-relational database (Mongo) was used in the beginning of the work, to insert ICD-10 information, and make it available to be queried, but it turned out to be very low performant. For debug and code implementation it was used a MacBook Pro, 2.6 GHz 6-Core Intel Core i7 Processor, with 16 GB 2400 MHz DDR4 of memory. Also, Google Colaboratory, a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, was used for some intermediate calculations. Training and tests were performed on a cluster, made available by the National Infrastructure of Distributed Computation (INCD) [1]. The computer used was Stratus cloud computing service, which is an IaaS cloud infrastructure based on Openstack, which enables control of large pools of computing, storage, and networking resources.

6.2 Performance metrics

In this project, the evaluation metrics used were the micro-average Precision (P) (Equation 6.1), micro-average Recall (R) (Equation 6.2), and micro-average F1-score (F1) (Equation 6.3). The justification for the use of these metrics, is because they are appropriated for imbalanced datasets, where no class is more important than the others.

The equations refer to an example with two classes, class 1 and class 2, where TP stands for True Positive, FP stands for False Positive, and FN stands for False Negative.

$$\mathrm{P} = \frac{TP1 + TP2}{TP1 + TP2 + FP1 + FP2} \tag{6.1}$$

$$\mathrm{R} = \frac{TP1 + TP2}{TP1 + TP2 + FN1 + FN2} \tag{6.2}$$

[1] https://www.incd.pt/, accessed on 2023-12-03

$$\text{F1} = \frac{2 \times R \times P}{R + P} \tag{6.3}$$

6.3 Cosine text similarity method

While the exact origin date of the cosine similarity method is not well-documented, it has been a fundamental technique in text analysis and natural language processing for several decades.

Cosine similarity method measures the similarity between two or more vectors of an inner product space. It is given by Equation 6.4, that measures the cosine of the angle between two vectors , which are normally different from zero, where a and b are two vectors and θ the angle they form [46].

$$cos(\theta) = \frac{a.b}{||a||.||b||} \tag{6.4}$$

If the value of the cosine is 0, it means that the vectors form an angle of 90 degrees, and the documents have no match. As the cosine similarity measurement gets closer to 1, the smaller is the angle between the two vectors A and B. Figures 6.1 and 6.2 shows this in a clearer way.

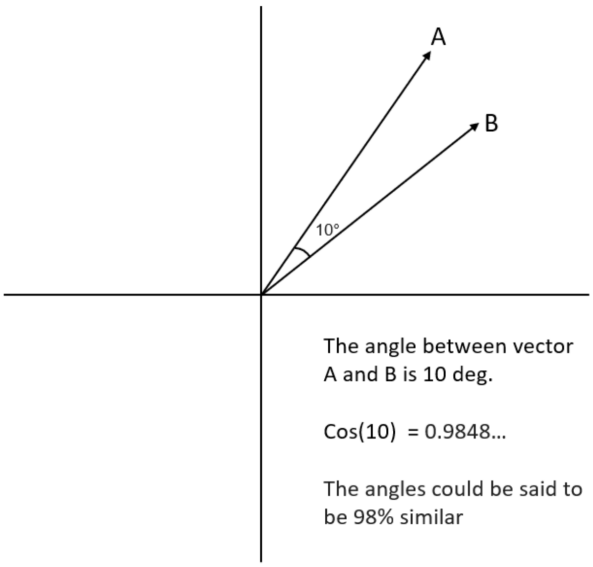


Figure 6.1: Representation of vectors A and B, with angle of 10º[2]

Cosine similarity is applied in many algorithms and applications. For example, the

[2]Source: https://builtin.com/machine-learning/cosine-similarity, accessed on 2024-04-20
[3]Source: https://builtin.com/machine-learning/cosine-similarity, accessed on 2024-04-20

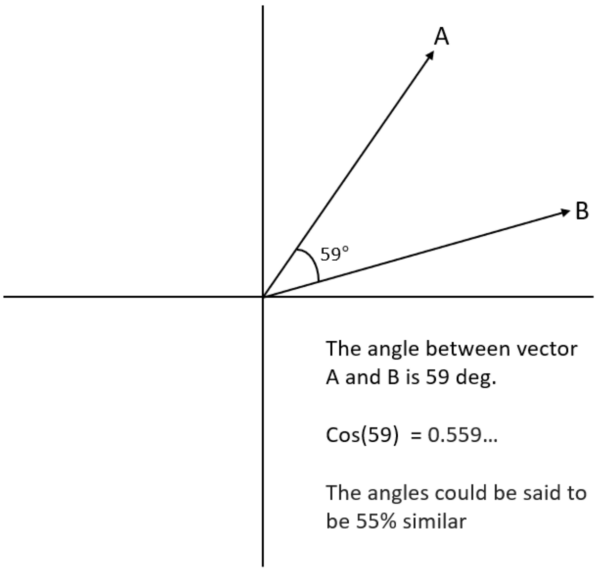


Figure 6.2: Representation of vectors A and B, with angle of 59º[3]

document similarity, is a good case that involves the requirement of identifying the similarity between pairs of documents. To find the quantification of the similarity between two documents, first is needed to convert the words or phrases within the document or sentence into a vectorized form of representation. The cosine similarity of 1 means that the two documents are exactly the same and a cosine similarity of 0 means that there are no similarities between the two documents.

There are other applications of cosine similarity, like recommendation systems, plagiarism detectors, data mining, and it can also be used as loss function when training neural networks.

In this work, our own version of the cosine similarity was implemented, but there are several available implementations.

6.4 Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency, is a method that allows to encode the text as vectors, and it tells how unique a word is across multiple text. The computation is based on the sum of TF-IDF of each query term, and is given by the equations 6.5, 6.6 and 6.7 [47].

$$TF = \frac{NW}{TW} \tag{6.5}$$

$$IDF = \log\left(\frac{TD}{ND}\right) \tag{6.6}$$

$$TF - IDF = TF \times IDF \tag{6.7}$$

In the equations, NW is the number of times the word appears in the document. TW is the total number of words in the document. TD is the number of documents in the corpus, and ND is the number of documents in the corpus that contain the term [47].

6.5 Text preprocessing for cosine method

The use of the cosine method involves the preprocessing of each EHR. Figure 6.3 shows the steps for the cosine method preprocessing.

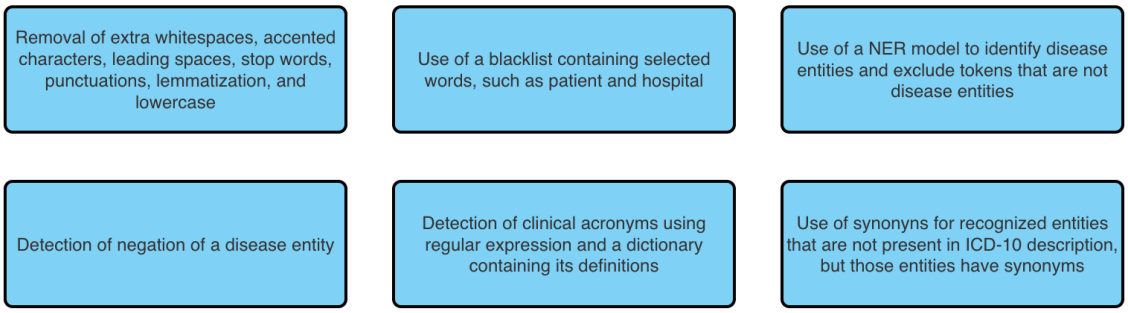


Figure 6.3: Preprocessing steps for cosine method

In detail, the preprocessing of cosine method includes:

- Removal of extra whitespaces, accented characters, leading spaces, stop words, punctuations, conversion to the base form (lemmatization), and lowercase;
- Removal of words in a blacklist containing selected words, such as "patient" and "hospital", because they have high prevalence in some texts and they are irrelevant for the codification. This is because some ICD-10 codes have for example the word "patient", such as "Y92230 – Patient room in hospital as the place of occurrence of the external cause", or "Z911 – Patient's noncompliance with medical treatment and regimen." The fact that those words are very prevalent in the EHRs, the cosine method encodes them and they interfere with classification, despite the absence of meaning for the task;

- Use of NER model "en ner bc5cdr md" [4], from scispaCy [5], a library of clinical and biomedical specific components that integrate with spaCy, to identify disease entities in EHR text and exclude tokens that are not disease entities;
- Detection of negation of a disease entity, using negspaCy, a spaCy pipeline component for negating concepts in a text;
- Detection of clinical acronyms, using a regular expression and a dictionary containing its definitions, and replacement by its long form. Taking the example of EHR with subject id "1000244" and hadm id "2132902", the short form word "afib" is present in the text, but not in any ICD-10 code description. After the long form replacement of "afib" by "atrial fibrillation", the cosine method could find the ICD-10 code "I480 - Paroxysmal atrial fibrillation". The dictionary of acronyms built contains 256 pairs of acronyms and their long form, where they can be consulted in Annex A 9;
- Use of synonyms because some EHRs could contain recognized entities that are not present in ICD-10 code descriptions, but those entities have synonyms, like the case of "dyslipidemia" that is an EHR disease entity that is not present in code description, but the synonym "hyperlipidemia" is. Hence, a dictionary was manually built in Python with entities and synonyms of those entities. This dictionary was created as a proof of concept, based on the observation that sometimes the keywords in the ICD-10 code description do not match the equivalent keywords in the EHR text, because synonyms are used. For example in the EHR with subject id "1000017" and hadm id "22927623", the ICD-10 code "K31819 - Angiodysplasia of stomach and duodenum without bleeding", the disease entity "angiodysplasia" was not present in the EHR text. The synonym "angioectasia" is present instead. For this reason, the cosine method could not properly detect the code. After implementation of the dictionary of disease synonyms, containing the pair "angiodysplasia/angioectasia", the method was able to correctly detect the "K31819" ICD-10 code. The dictionary of synonyms, as a proof of concept, contains only four pairs of entities and their synonyms, namely "angioectasia" as "angiodysplasia", "dyslipidemia" as "hyperlipidemia", "tobacco" as "nicotine", and "smoking" as "nicotine", and expansion is left as future work.

6.6 Improved Cosine Similarity implementation

This section describes the Improved Cosine Similarity (ICS) method, with the implementation of the multiple runs algorithm and the bucket category concept.

[4]https://allenai.github.io/scispacy/, accessed on 2023-10-10
[5]https://allenai.github.io/scispacy/, accessed on 2023-10-10

6.6.1 Multiple runs algorithm

A multiple runs algorithm was implemented, where between runs, the most common encoded words (≥ 2 occurrences) are included in the blacklist and allow to encode other disease entities present in the EHR. This algorithm was implemented in order to eliminate words that are frequent and superimpose to other important words. If the disease entity does not appear with frequency ≥ 2 occurrences in the top-5 ICD-10 codes, based on their similarity to a given EHR text, but appears in the top-5 of two consecutive runs, they are also included in the blacklist of words, thus allowing to encode other disease entities present in the EHR.

Let's consider, as an example, the case where for a single run, the top-5 similarity, between the ICD-10 codes and a given EHR text, is "J17 - Pneumonia in diseases classified elsewhere" (0.63 similarity), "R0602 - Shortness of breath" (0.62 similarity), "P23 - Congenital pneumonia" (0.58), "P239 - Congenital pneumonia, unspecified" (0.54) and "J1289 - Other viral pneumonia" (0.49). The word "pneumonia" has frequency ≥ 2 occurrences among the top-5, so it will be included in the blacklist, and in the next run it will not be used for similarity calculation. If the word was not removed, then "pneumonia" would superimpose, making it difficult to encode other ICD-10 codes which are still present but have lower similarity scores because of the influence of the word pneumonia. In the second run, after the removal of "pneumonia", the encoded top-5 is "R0602 - Shortness of breath", "J44 - Other chronic obstructive pulmonary disease", "J81 - Pulmonary edema", "J449 - Chronic obstructive pulmonary disease, unspecified" and "J811 - Chronic pulmonary edema". The next run may add the word "pulmonary" to the blacklist, thus making way to possibly find other ICD-10 codes.

On top of that, if the disease entity does not appear with frequency ≥ 2 occurrences in the top-5 ICD-10 codes, but appears in the top-5 of two consecutive runs, it is also included in a temporary blacklist. To exemplify this fact, consider the previous example, where "R0602 - Shortness of breath" appears in both runs. Its words are also included in the temporary blacklist, since it was already encoded. From the previous example, after the removal of "shortness of breath", the third run does not find this ICD-10 code, thus allowing other codes to surface: "R11 - Nausea and vomiting", "R112 - Nausea with vomiting, unspecified", "R0603 - Acute respiratory distress", "R110 - Nausea", "R1111 - Vomiting without nausea".

6.6.2 Bucket category concept

A new strategy was implemented that was named as bucket category selection, that extends and improves the multiple runs algorithm. In each run of the multiple runs algorithm, for each one of the top-5 ICD-10 codes retrieved, the parent ICD-10 category is also retrieved (one level above the ICD-10 code). Afterwards, the similarity is calcu-

lated between the EHR and all the ICD-10 codes that are under the ICD-10 code's parent category, forming what we call a bucket of codes. The codes in the bucket are then sorted by similarity and up to 40 of them are considered positive candidates. Hence, the bucket selection works like a way of lowering the similarity threshold to codes of a given category, based on the observation that neighbouring codes are often found in EHR.

Figure 6.4 shows the flowchart of the algorithm, with the bucket category concept.

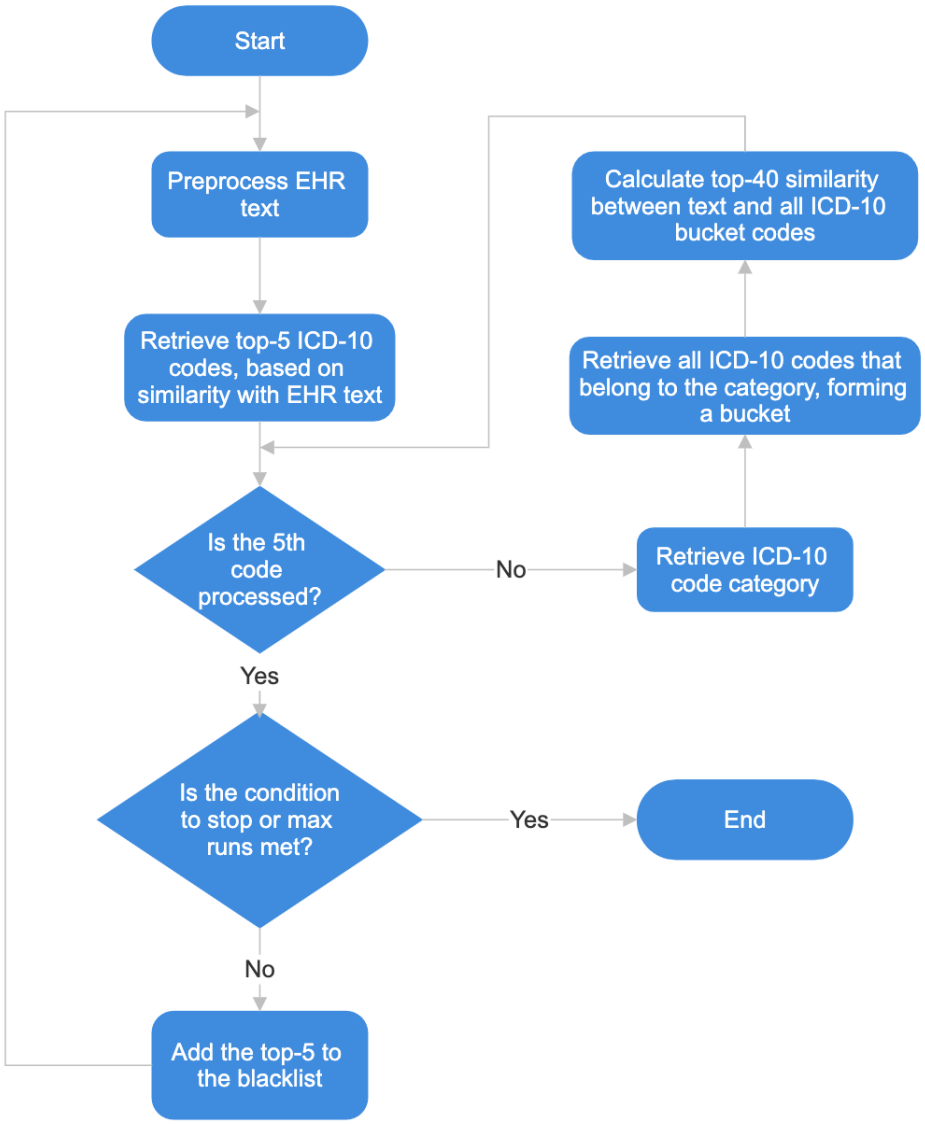


Figure 6.4: Flowchart of the algorithm developed, showing the bucket category concept

A true positive is considered if the ICD-10 label of the EHR exists in the bucket, and a false positive if it does not exist. To understand how the bucket category selection works, consider the following ICD-10 codes label of the EHR, "S72012A", "W010XXA", "Y93K1", "Y92480", "K219", "E7800", "I341", "G43909", "Z87891", "Z87442", "F419", "M810", "Z7901". After the calculation of the similarity between the preprocessed EHR text and ICD-10 code descriptions, "K21", "I341", "M2535" and "M21259", were

identified with the highest similarity. After this, the following ICD-10 categories of the previous codes were obtained, "K20-K31", "I30-I52" and "M20-25", forming the buckets with all the ICD-10 codes that belong to them. For the category "K20-K31", it is found that "K219", is included in the top-40 bucket similarity. And for category "I30-I52", code "I341" is also retrieved. In this case there are two extra true positives.

6.6.3 PLM-ICD

PLM-ICD [19] refers to Pretrained Language Model, and it is a deep-learning based model for automatic encoding of clinical texts that contain biomedical terms into ICD-10 codes.

General PLMs are pretrained on large texts that may or may not contain biomedical text. PLM-ICD uses BioBERT [14], PubMedBERT [16], and RoBERTA-PM [17] pretrained models. This model uses these domain-specific PLMs and fine-tunes them on the automatic ICD coding.

PLM-ICD also tackles the long input text problem, proposing segment pooling by splitting the whole document into segments shorter than the maximum length, and encodes them into segment representations. These segments are input into PLMs separately to compute hidden representations, and after concatenated to get the hidden representations of all tokens.

This model also tries to solve the problem of large datasets with the label-aware attention mechanism proposed by [48] to capture the important text fragments relevant to certain labels.

The PLM-ICD code and documentation are available at [6], only for the MIMIC datasets previous to MIMIC-IV. In order to train to this specific dataset, adaptations were performed. Preprocessing of the MIMIC-IV dataset was performed following the setting from [45], available at [7]. The PLM used was the BioLM RoBERTa-base-PM-M3-Voc-distill-align-hf, available at [8].

6.6.4 PLM-ICD-C

A model named PLM-ICD-C was used, in order to improve PLM-ICD results based on the cosine similarity method. Combining these two methods, the goal is to discover some more useful ICD-10 codes to suggest to the medical doctors that perform the classification.

[6] https://github.com/MiuLab/PLM-ICD/tree/master, accessed on 2023-12-03
[7] https://github.com/jamesmullenbach/caml-mimic, accessed on 2023-12-03
[8] https://github.com/facebookresearch/bio-lm, accessed on 2023-12-03

PLM-ICD-C rules

Three rules were defined for PLM-ICD-C. S_P stands for similarity of PLM-ICD, S_C stands for similarity of the Improved Cosine algorithm. PC is a code determined as probable candidate to positive, and NC stands for candidate to negative. T_1, T_2 and T_3 are similarity thresholds to be defined on each experiment. T_1 and T_2 are thresholds used for the PLM-ICD similarity levels, and $T_1 > T_2$. T_3 is a threshold used for the ICS similarity level. The rules are as follows:

1. IF $S_P \geq T_1 \longrightarrow PC$;
2. IF $T_2 \leq S_P < T_1$ AND $S_C \geq T_3 \longrightarrow PC$;
3. IF $S_P < T_2 \longrightarrow NC$.

Rule number 1 states that when PLM-ICD finds a code with high probability of being a positive, it is accepted as a positive candidate without further screening by the Improved Cosine method.

Rule number 2 brings ICS into play when PLM-ICD retrieves a code with just average confidence. That happens when the similarity is normally about 0.5, which means there is almost the same chance of a code being positive as it is of being negative. Hence, when S_P is between T_2 and T_1, then S_C is calculated for that code and if it is above T_3 then the code is proposed as PC, otherwise it is proposed as NC.

Rule number 3 states that when PLM-ICD assigns very low probability to a code, it is also not screened by ICS, because chances of it being a true positive are very low, so it is immediately accepted as a negative candidate.

Architecture

Figure 6.5 contains a diagram showing the flow of data and components of the PLM-ICD-C. The EHR is fed to the PLM-ICD, whose output is then fed to the Improved Cosine Similarity module, where it is further processed and an improved prediction is output.

The ICS module applies the rules described in Section 6.6.4. Therefore, if needed it will calculate the cosine similarity between the input EHR and all ICD codes, performing multiple runs and bucket category selection if applicable, in order to optimise the prediction that is output.

Implementation of rules

Table 6.1 clarifies how the truth values are determined for each code, according to the results of the PLM-ICD and ICS modules. Label is the desired result.

For the condition $S_P \geq T_1$, only the similarity of PLM-ICD, S_P, is considered—the

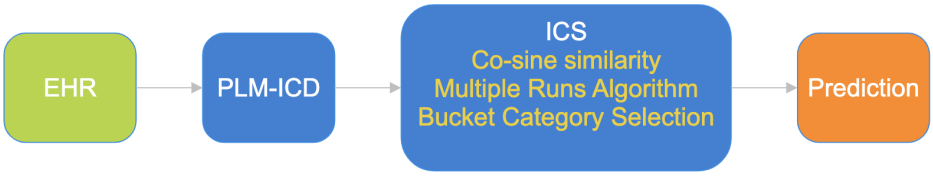


Figure 6.5: PLM-ICD-C architecture

prediction output is positive, based on PLM-ICD only. For the condition $S_P < T_2$, the prediction is also just based on S_P, but in this case it is assumed as negative.

When S_P is in the interval $T_2 \leq S_P < T_1$, then ICS is activated and S_C is also considered to determine the prediction. When $S_C \geq T_3$, a positive is predicted, otherwise a negative is predicted.

Table 6.1: Summary of the truth values resulting from applying the PLM-ICD-C rules

S_P	S_C	Prediction	Label	Result
$\geq T_1$	-	Positive	Positive	TP
			Negative	FP
$< T_1$	$\geq T_3$	Positive	Positive	TP
			Negative	FP
$\geq T_2$	$< T_3$	Negative	Positive	FN
			Negative	TN
$< T_2$	-	Negative	Positive	FN
			Negative	TN

6.7 Summary

This project used Python as software, and training and tests were performed on a cluster, namely a Stratus cloud computing service which is an IaaS cloud infrastructure based on Openstack.

The evaluation metrics used, taking into account an imbalanced dataset, were the micro-average Precision, micro-average Recall, and micro-average F1-score.

An implementation of the cosine similarity method was performed, which can be used for document similarity, as well as TF-IDF, that can tell how unique a word is across multiple text.

The cosine method involves the preprocessing of each EHR, namely removal of extra whitespaces, accented characters, leading spaces, stop words, punctuations, lemmatization, and lowercase. A blacklist was also used. It contains selected words, such as "patient" and "hospital", because they have high prevalence and are irrelevant for the codification. A NER model was used to identify disease entities and exclude tokens

that are not disease entities. For the case of negation of entities, negspaCy was used. Detection of clinical acronyms was implemented, using regular expression and a dictionary containing its definitions. Finally, it were used synonyms for entities that are not present in ICD-10 code descriptions, but those entities have synonyms. As a proof of concept, four pairs of entities and their synonyms were used.

An Improved Cosine Similarity method was implemented. It includes a novel multiple runs algorithm and bucket category concept. The multiple runs algorithm consists in including in a blacklist, the most common encoded words (≥ 2 occurrences), between runs. On the other hand, the bucket category concept extends and improves the multiple runs algorithm, where in each run of the multiple runs algorithm, for each one of the top-5 ICD-10 codes retrieved, the parent ICD-10 category is also retrieved. Afterwards, the similarity is calculated between the EHR and all the ICD-10 codes that are under the ICD-10 code's parent category, forming a bucket of codes. The codes in the bucket are then sorted by similarity and up to 40 of them are considered positive candidates.

PLM-ICD is a deep-learning based model for automatic encoding of clinical texts that contain biomedical terms into ICD-10 codes. It uses MIMIC datasets previous to MIMIC-IV. In order to train to this specific dataset, adaptations were performed, as well as preprocessing following the setting from [45].

A model named PLM-ICD-C was used, in order to improve PLM-ICD results based on the cosine similarity method. Combining these two methods, the goal is to discover some more useful ICD-10 codes to suggest to the medical doctors that perform the classifications. Three rules were defined for PLM-ICD-C. Rule number 1 states that when PLM-ICD finds a code with high probability of being a positive, it is accepted as a positive candidate without further screening by ICS. Rule number 2 brings ICS into play when PLM-ICD retrieves a code with just average confidence. That happens when the similarity is normally about 0.5, which means there is almost the same chance of a code being positive as it is of being negative. Finally, rule number 3 states that when PLM-ICD assigns very low probability to a code, it is also not screened by ICS, because chances of it being a true positive are very low, so it is immediately accepted as negative candidate.

7 Experiments and results

In this section, for the different datasets used, the results are described for the Improved Cosine Similarity method, PLM-ICD and PLM-ICD-C. Part of these results were published in [49].

7.1 Harvard Medical School dataset

A summary of work performed for the unlabeled Harvard Medical School dataset is given below.

- Information on excel format of ICD-10-CM-PCS was provided from the Portuguese Ministry of Health;
- ICD-10-CM information was inserted into a non-relational database (Mongo), for ease of manipulation and access;
- Python programming environment was set up, with spaCy, an NLP library that provides support for different text processing operations.

Once the experimental setup was created, experiments were performed in order to prepare the data and test some coding approaches. The first approach is described below.

- In order to process a clinical health record, the text is preprocessed as previously described;
- After the first step, a list of ICD-10 codes is retrieved for a health record, sorted by similarity. The top-5 codes, with the biggest value of cosine of similarity, is considered.

This first approach, using Mongo to store the ICD-10-CM information, showed poor performance, because of the time necessary to obtain the information from the repository. Different improvements were implemented step by step. At this stage, the programming code already includes:

- An improvement of performance, reading ICD-10-CM information from a JSON file, previously created in a proper format;
- Use of a blacklist containing selected words, such as “patient” and “hospital”. Results showed that some EHR were misclassified because of those words. The ICD-10 codes “Y92230 - Patient room in hospital as the place of occurrence of the

external cause" and "Z911 - Patient's noncompliance with medical treatment and regimen", were wrongly encoded because of the word "patient" present in the EHR. After including the word in the blacklist, the ICD-10 code "R0602 - Shortness of breath" was encoded, as the sentence "The patient at that time noted slight shortness of breath but was sent home anyway" is present in the EHR, as can be seen in Table 7.1 for EHR 2.xml.txt;

- Use of NER model "en ner bc5cdr md" [1]. It allows a more precise codification, because before using NER model, two ICD-10 codes detected were "G443 - Post-traumatic headache" and "O480 - Post-term pregnancy", which neither "headache" or "pregnancy" words were present in the EHR. After the use of the NER model, the two ICD-10 codes, "R0602 - Shortness of breath" and "T82855 - Stenosis of coronary artery stent", were correctly encoded because of the sentences "Ms. Bridge is a 74 y/o female admitted to outside hospital on 2015-10-21 with chest discomfort, vomiting and shortness of breath" and "CNIS revealed carotid stenosis and she ultimately underwent left carotid stenting by vascular surgery on 10-30", as can be seen in Table 7.2, for EHR 3.xml.txt;
- Detection of negation of a disease entity, using negspaCy, because before using negspaCy, two ICD-10 codes detected were "T79A0 - Compartment syndrome, unspecified" and "T79A0XS - Compartment syndrome, unspecified, sequela", because of the presence of the entity "compartment syndrome", but negated, as can be seen in the EHR sentence "Dermatology and plastic surgery were consulted who felt that symptoms were due to trauma; it was not felt that patient had compartment syndrome". After the use of negspaCy, the previous ICD-10 codes were no more encoded, giving the place of two new ICD-10 codes "F31 - Bipolar disorder" and "F318 - Other bipolar disorders", because of the presence of the sentence "Ms. Franklin is a 34-year-old woman with Bipolar disorder and group home resident who by way of EMS for apparent alprazolam overdose", as can be seen in Table 7.3, for EHR 23.xml.txt;
- Detection of clinical acronyms, because before using it, while having the acronym "COPD" which means Chronic Obstructive Pulmonary Disease, no ICD-10 code was detected containing these words. After using the detection of clinical acronyms, the two ICD-10 codes "J44 - Other chronic obstructive pulmonary disease" and "J449 - Chronic obstructive pulmonary disease, unspecified", were encoded, as can be seen in Table 7.4, for EHR 8.xml.txt;
- Implementation of multiple runs, because if disease entities were not included in the blacklist after multiple runs, the entity "pain" in the first run, with ICD-10 codes "R52 - Pain, unspecified" and "M545 - Low back pain", would superimpose and the following runs would have ICD-10 codes having this disease entity.

[1] https://allenai.github.io/scispacy/, accessed on 2023-10-10

After including the disease entities, namely "pain", in the blacklist of consecutive runs, new ICD-10 codes can be detected, namely "R11 - Nausea and vomiting" and "R112 - Nausea with vomiting, unspecified", because of the sentence "She presented with left upper quadrant pain as well as nausea and vomiting which is a long-standing complaint" as can be seen in Table 7.5, for EHR 1.xml.txt;

- Use of trigrams, because while not using them before the disease entity, namely the three words of the EHR text before the disease entity, the detected ICD-10 codes are "R52 - Pain, unspecified" and "M545 - Low back pain". After the use of trigrams before the disease entity, namely the three words of the EHR text before the disease entity, the detected ICD-10 codes are "R1011 - Right upper quadrant pain" and "R1012 - Left upper quadrant pain, because of the sentence "She presented with left upper quadrant pain as well as nausea and vomiting which is a long-standing complaint", allowing to give more context to the codification, as can be seen in Table 7.6, for EHR 1.xml.txt.

Table 7.1: Example results of encoding for EHR 2.xml.txt, before and after using blacklist

EHR 2.xml.txt		
	ICD-10 code	Top-5 Cosine Similarity
Encoding before using blacklist	Y92230 – Patient room in hospital as the place of occurrence of the external cause	0.33166
	Z911 – Patient's noncompliance with medical treatment and regimen	0.29775
Encoding after using blacklist	R0602 – Shortness of breath	0.20592
	Z836 – Family history of other diseases of the respiratory system	0.18909

Table 7.2: Example results of encoding for EHR 3.xml.txt, before and after using NER model

EHR 3.xml.txt		
	ICD-10 code	Top-5 Cosine Similarity
Encoding before using NER model	G443 – Post-traumatic headache	0.25540
	O480 – Post-term pregnancy	0.25359
Encoding after using NER model	R0602 – Shortness of breath	0.43599
	T82855 – Stenosis of coronary artery stent	0.34729

Table 7.3: Example results of encoding for EHR 23.xml.txt, before and after using negspaCy

EHR 23.xml.txt		
	ICD-10 code	Top-5 Cosine Similarity
Encoding before using negspaCy	T79A0 – Compartment syndrome, unspecified	0.65795
	T79A0XS – Compartment syndrome, unspecified, sequela	0.63173
Encoding after using negspaCy	F31 – Bipolar disorder	0.47625
	F318 – Other bipolar disorders	0.47625

Table 7.4: Example results of encoding for EHR 8.xml.txt, before and after expanding acronyms to long form

EHR 8.xml.txt		
	ICD-10 code	Top-5 Cosine Similarity
Encoding before expanding acronyms to long form	R0603 – Acute respiratory distress	0.63544
	P228 – Other respiratory distress of newborn	0.61513
Encoding after expanding acronyms to long form	J44 – Other chronic obstructive pulmonary disease	0.97870
	J449 – Chronic obstructive pulmonary disease, unspecified	0.96665

Table 7.5: Example results of encoding for EHR 1.xml.txt, after multiple runs, first and second runs

EHR 1.xml.txt		
	ICD-10 code	Top-5 Cosine Similarity
Encoding after multiple runs, first run	R52 – Pain, unspecified	0.46000
	M545 – Low back pain	0.41942
Encoding after multiple runs, second run	R11 – Nausea and vomiting	0.41084
	R112 – Nausea with vomiting, unspecified	0.40632

7.2 MIMIC-IV dataset

The programming code at this stage includes:

- Use of trigrams before and after the disease entity, within continuous grams, namely the three words of the EHR text before and after the disease entity, that allow to provide more context on the process of encoding, comparing with only using trigrams before the entity, where for subject id "10000084" and hadm id "23052089" it allows to encode one more ICD-10 code, namely "Z8546 - Personal history of malignant neoplasm of prostate", in a total of four correct ICD-10 codes, out of six in total, as can be seen in Table 7.7;

- Use of synonyms, as described previously, where for the EHR with subject id "1000017" and hadm id "22927623", the ICD-10 code "K31819 - Angiodysplasia of stomach and duodenum without bleeding", the disease entity "angiodysplasia" was not present in the EHR text. The synonym "angioectasia" is present instead. For this reason, the cosine method could not properly detect the code. After implementation of the dictionary of disease synonyms, containing the pair "angiodysplasia/angioectasia", the method was able to correctly detect the "K31819 - Angiodysplasia of stomach and duodenum without bleeding" ICD-10 code;

- Use of new strategy named as bucket category, where for subject id "10002443" and hadm id "21329021" it allows to encode one more ICD-10 code, namely "Z86718 - Personal history of other venous thrombosis and embolism", in a total of six correct ICD-10 codes, out of eleven in total, as can be seen in Table 7.8.

Table 7.6: Example results of encoding for EHR 1.xml.txt, before and after using trigrams, before the disease entity

EHR 1.xml.txt		
	ICD-10 code	Top-5 Cosine Similarity
Encoding before using trigrams, before the disease entity	R52 – Pain, unspecified	0.46000
	M545 – Low back pain	0.41942
Encoding after using trigrams, before the disease entity	R1011 - Right upper quadrant pain	0.43334
	R1012 - Left upper quadrant pain	0.40227

Table 7.7: Example results of encoding for subject id "10000084" and hadm id "23052089", using trigrams, before and after the disease entity

EHR subject id "10000084" and hadm id "23052089"		
	ICD-10 code	ICD-10 source of truth
Encoding using trigrams, before the disease entity	F0280 - Dementia in other diseases classified elsewhere without behavioral disturbance R441 - Visual hallucinations E785 - Hyperlipidemia, unspecified	G3183 - Dementia with Lewy bodies F0280 - Dementia in other diseases classified elsewhere without behavioral disturbance R441 - Visual hallucinations R296 - Repeated falls E785 - Hyperlipidemia, unspecified Z8546 - Personal history of malignant neoplasm of prostate
Encoding using trigrams, before and after the disease entity	F0280 - Dementia in other diseases classified elsewhere without behavioral disturbance R441 - Visual hallucinations E785 - Hyperlipidemia, unspecified Z8546 - Personal history of malignant neoplasm of prostate	

7.2.1 Improved Cosine Similarity method

This section describes the results obtained for the Improved Cosine Similarity method, using the multiple runs algorithm and the bucket category concept, while using samples from a fraction of $\frac{1}{6}$ of the test set (4077 out of 24,462), thus proving the usefulness of those improvements to reach the Improved Cosine method used.

Multiple runs algorithm

Table 7.9 shows a summary of the main results, obtained using the multiple runs algorithm, but without the strategy of bucket category. Three different approaches were tried. In the first approach, a maximum of 11 runs was defined. Stop conditions were also defined, namely the algorithm would stop if 60% of the top-5 codes have a similarity below 0.197. If achieved a state in the run where it has a number of detected ICD-10 codes less than 12% of the total encoded so far, and the maximum number of 11 runs is not yet reached, the run is also stopped. This is done to let prevail the codes found in the initial runs, which have a bigger similarity, thus a bigger probability of being correct, compared to codes in latter runs that are in less number. These values of percentages were defined empirically. More true positives may be found at a lower value of similarity, but at the same time, more possible false positives may be detected. On the other hand, if the value of the percentage is higher, it is likely that less true positives may be detected, but less false positives would be found. At the end it is a compromise between Precision and Recall.

Table 7.8: Results of encoding for subject id "10002443" and hadm id "21329021", before and after using the concept of bucket selection

EHR subject id "10002443" and hadm id "21329021"		
	ICD-10 code	ICD-10 source of truth
Encoding before using the concept of bucket selection	I309 - Acute pericarditis, unspecified I314 - Cardiac tamponade I480 - Paroxysmal atrial fibrillation E119 - Type 2 diabetes mellitus without complications M069 - Rheumatoid arthritis, unspecified	I309 - Acute pericarditis, unspecified J9602 - Acute respiratory failure with hypercapnia I314 - Cardiac tamponade I480 - Paroxysmal atrial fibrillation E119 - Type 2 diabetes mellitus without complications Z66 - Do not resuscitate I10 - Essential (primary) hypertension M069 - Rheumatoid arthritis, unspecified E785 - Hyperlipidemia, unspecified Z86718 - Personal history of other venous thrombosis and embolism Z87891 - Personal history of nicotine dependence
Encoding after using the concept of bucket selection	I309 - Acute pericarditis, unspecified I314 - Cardiac tamponade I480 - Paroxysmal atrial fibrillation E119 - Type 2 diabetes mellitus without complications M069 - Rheumatoid arthritis, unspecified Z86718 - Personal history of other venous thrombosis and embolism	

Table 7.9: Metric (micro-average) results for the cosine method, without bucket category concept

Micro-avg	Precision	Recall	F1-score
Approach 1 (11 runs)	0.045	0.111	0.064
Approach 2 (5 runs)	0.055	0.087	0.068
Approach 3 (11 runs, similarity threshold 0.2)	0.047	0.104	0.065

The second approach used the same configuration as the first approach, but instead of using up to 11 runs, it uses just 5 runs.

The third approach uses the same configuration, but with up to 11 runs, and excludes any ICD-10 codes that have a similarity below 0.2.

As the table shows, the values of Precision, Recall and F1 are very modest, showing that the cosine similarity with just these aids is insufficient to provide useful coding suggestions.

Bucket category

In order to improve the algorithm, a new strategy was designed, that was named as bucket category, as explained in Section 6.6.2. In this strategy, from the top-5 codes of each run, it gets the ICD-10 categories. Then a new similarity is calculated between the text of the run and all the codes of those categories, forming a bucket.

Table 7.10 shows a summary of the main results with the Improved Cosine Similarity method using the multiple runs algorithm and the bucket category concept, obtained using the three different approaches too. The first, second and third approaches, consider the bucket category concept, for 5, 7, and 11 runs, respectively.

Table 7.10 shows an improvement on the previous results. Precision had a 5-fold improvement, while there are 2-3 fold improvements in Recall and 4-fold improvements

Table 7.10: Metric (micro-average) results, for the cosine method with bucket category concept

Micro-avg	Precision	Recall	F1-score
Approach 1	0.291	0.206	0.241
Approach 2	0.271	0.239	0.254
Approach 3	0.247	0.265	0.256

in F1, using the concept of bucket category. Therefore, this combination of multiple runs and bucked category search was adopted in the remainder of the project and is called ICS method.

7.2.2 PLM-ICD

The run of the PLM-ICD was initially performed in Google Colaboratory, but the limitations of memory and time could not allow to train the model in this platform. After this, the resources from INCD were used, but yet initially the training was not possible because of the limited time that INCD imposes (maximum 4 days) for a job to run using Graphics Processing Unit (GPU). INCD then proposed to use of a virtual machine from Google, with A100 GPUs with nvlink interconnections, that allow to train the model for a wider time frame window.

Table 7.11 shows a summary of the results for PLM-ICD, obtained for test with threshold of 0.5, 0.4 and 0.2, and $\frac{1}{6}$ of the test dataset, the same as used for the cosine method.

Table 7.11: Metric (micro-average) results, for PLM-ICD test

Micro-avg	Precision	Recall	F1-score
Test (threshold 0.5)	0.65614	0.34544	0.45260
Test (threshold 0.4)	0.60596	0.38264	0.46907
Test (threshold 0.2)	0.46282	0.47680	0.46970

As the table shows, PLM-ICD performs better than the cosine method using the approaches described before. However, there is still room for improvement, specially considering the difficult trade-off between Precision and Recall. As the threshold decreases, PLM-ICD finds more codes, but many of them are false positives. So Recall increases but Precision is dramatically affected.

Edin, et al. (2023) [20] reported a better F1-score on the test set (58.5% compared to 46.9% in the present work), using PLM-ICD on MIMIC-IV with ICD-10 labels. They used a bigger test set consisting of 24,461 records, following the partition method proposed in [45]. In the present work, because of lack of computational resources, the volume of data was smaller and that may justify the different results.

7.2.3 PLM-ICD-C

PLM-ICD-C is a proposal to enhance the PLM-ICD model using the ICS method, with the multiple runs and bucket category algorithms. So the results are presented and compared below.

Results with different thresholds

Tables 7.12 and 7.13 show a summary of the results for PLM-ICD-C, with the values of T_1=0.5, $T_2 = 0.4$, $T_3 = 0.002$, which we call PLM-ICD-C (1), and $T_1 = 0.4$, $T_2 = 0.2$, $T_3 = 0.002$, which we call PLM-ICD-C (2), respectively. The threshold values T_1, T_2 and T_3 were determined empirically. Therefore, they may be subject to optimization in future work, even though, according to the experiments, they may be close to optimal.

Table 7.12: Summary of the correct and incorrect results for PLM-ICD-C (1)

S_P	S_C	Result	PLM-ICD	PLM-ICD-C
≥ 0.5	≥ 0.002	TP	20,280	16,969
		FP	10,628	7741
	< 0.002	FN	-	3311
		TN	-	2887
< 0.5	≥ 0.002	TP	2184	1659
		FP	3980	2801
≥ 0.4	< 0.002	FN	-	525
		TN	-	1179
< 0.4	-	FN	36,244	-
		TN	396,337,471	-

Table 7.13: Summary of the correct and incorrect results for PLM-ICD-C (2)

S_P	S_C	Result	PLM-ICD	PLM-ICD-C
≥ 0.4	≥ 0.002	TP	22,464	18,628
		FP	14,608	10,542
	< 0.002	FN	-	3836
		TN	-	4066
< 0.4	≥ 0.002	TP	5528	4140
		FP	17,882	11,987
≥ 0.2	< 0.002	FN	-	1388
		TN	-	5895
< 0.2	-	FN	30,716	-
		TN	396,319,589	-

Tables 7.14 and 7.15 show the comparison of results using PLM-ICD threshold 0.4 and PLM-ICD-C (1) with $T_1 \geq 0.5$ and $T_3 \geq 0.002$, and using PLM-ICD threshold 0.2 and PLM-ICD-C (2) with $T_1 \geq 0.4$ and $T_3 \geq 0.002$, respectively.

The results shown in Tables 7.14 and 7.15 show that PLM-ICD-C (2) performs better than PLM-ICD-C (1), because it filters more FP than looses TP.

Table 7.14: Comparison of results while using PLM-ICD threshold 0.4, and PLM-ICD-C (1) with $T_1 \geq 0.5$ and $T_3 \geq 0.002$
The results shown in Tables 7.14 and 7.15 show that PLM-ICD-C (2) performs better than PLM-ICD-C (1), because it filters more FP than looses TP.

Result	PLM-ICD	PLM-ICD-C	Difference
TP	20,280	16,969	Lost 3311
FP	10,628	7741	Filtered 2887

Table 7.15: Comparison of results while using PLM-ICD threshold 0.2, PLM-ICD-C (2) with $T_1 \geq 0.4$ and $T_3 \geq 0.002$

Result	PLM-ICD	PLM-ICD-C	Difference
TP	22,464	18,628	Lost 3836
FP	14,608	10,542	Filtered 4066

Tables 7.16 and 7.17 show the comparison of results using PLM-ICD threshold 0.4 and PLM-ICD (1) with T_1 <0.5, $T_2 \geq 0.4$, $T_3 \geq 0.002$, and using PLM-ICD threshold 0.2 and PLM-ICD (2) with T_1 <0.4, $T_2 \geq 0.2$, and $T_3 \geq 0.002$, respectively.

Again, the results show that PLM-ICD-C (2) in Table 7.17 has a better performance that PLM-ICD-C (1) in Table 7.16, because it filters more FP than looses TP.

Table 7.16: Comparison of results while using PLM-ICD threshold 0.4 and PLM-ICD (1) with T_1 <0.5, $T_2 \geq 0.4$ and $T_3 \geq 0.002$

Result	PLM-ICD	PLM-ICD-C	Difference
TP	2184	1659	Lost 525
FP	3980	2801	Filtered 1179

Figure 7.1 shows the dynamics of lost TP and filtered FP for samples with similarity S_P above T_1. The results were calculated for values of T_1 above 0.7, 0.5, 0.45 and 0.4.

Figure 7.2 also shows the dynamics of lost TP and filtered out FP, but now the samples where $T_2 < S_P < T_1$. Results are shown for different intervals between T_2 and T_1, namely 0.4 and 0.7, 0.4 and 0.5, 0.4 and 0.45, and 0.2 and 0.4.

Figure 7.1 shows that the number of filtered FP decreases slightly faster than the loss of TP. The use of PLM-ICD-C is more advantageous for the interval where having $T_2 < S_P \leq T_1$, *i.e.*, where PLM-ICD's confidence is lower, as can be seen in Figure 7.2. PLM-ICD-C can still let through some TP, while filtering out FP that PLM-ICD would be unable to filter out.

Table 7.17: Comparison of results while using PLM-ICD threshold 0.2 and PLM-ICD (2) with T_1 <0.4, $T_2 \geq 0.2$ and $T_3 \geq 0.002$

Result	PLM-ICD	PLM-ICD-C	Difference
TP	5528	4140	Lost 1388
FP	17,882	11,987	Filtered 5895

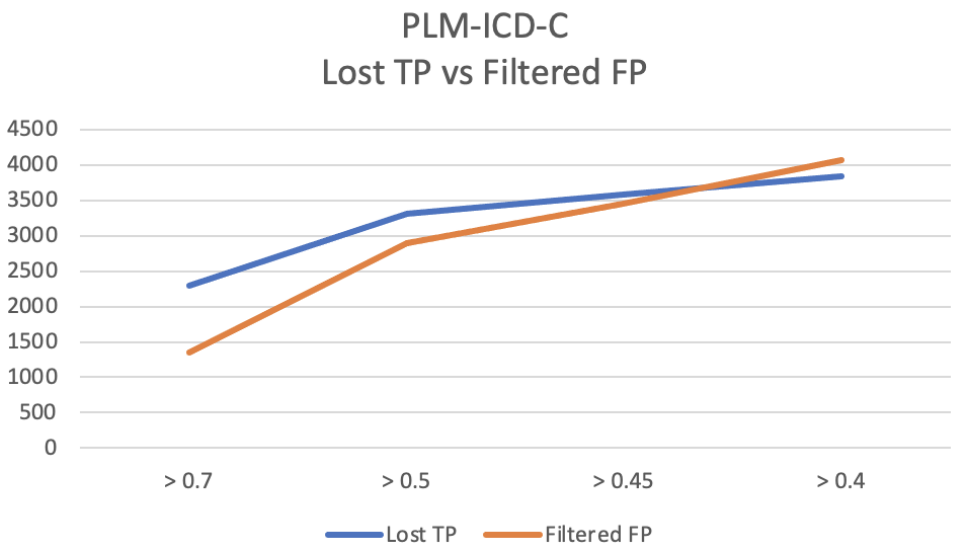


Figure 7.1: Dynamics of lost TP and filtered FP for PLM-ICD-C for samples with $S_P \geq T_1$, for different values of T_1

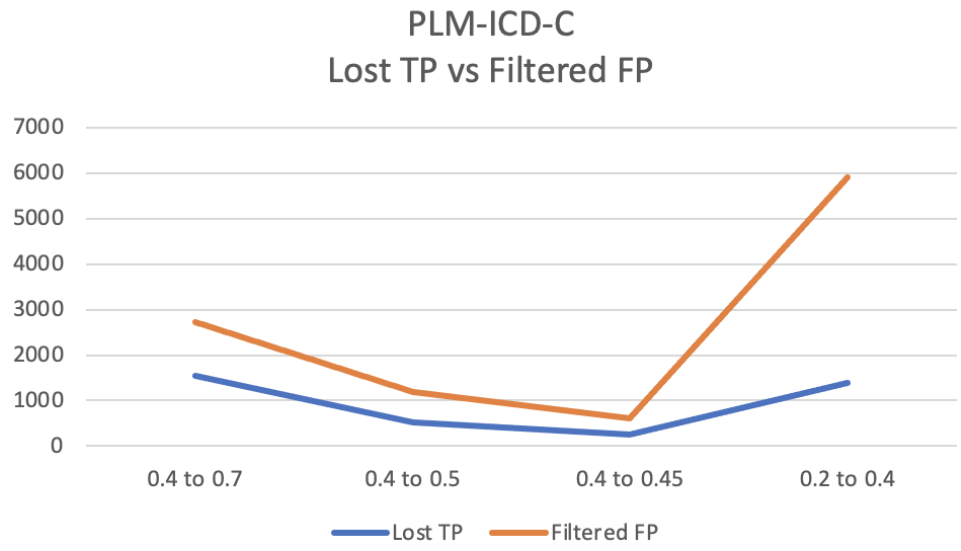


Figure 7.2: Dynamics of lost TP and filtered FP for PLM-ICD-C for samples with similarity in the range $T_2 < S_P < T_1$, for different values of T_2 and T_1

Performance metrics

Tables 7.18 shows the performance metrics for PLM-ICD and PLM-ICD-C.

For PLM-ICD, the threshold of 0.2 has a lower Precision (0.46282) than the threshold

Table 7.18: Micro-average results, for PLM-ICD thresholds of 0.4 and 0.2, and for PLM-ICD-C (1) and (2)

Micro-avg	Precision	Recall	F1-score
PLM-ICD (threshold 0.4)	0.60596	0.38264	0.46907
PLM-ICD (threshold 0.2)	0.46282	**0.47680**	0.46970
PLM-ICD-C (1)	**0.62031**	0.37370	0.46641
PLM-ICD-C (2)	0.50008	0.45316	**0.47547**

of 0.4 (0.60596), but a much better Recall (from 0.38264 to 0.47680), which translates into a slightly better F1-score.

For PLM-ICD-C, model (1) has a better Precision (0.62031) than (2) (0.50008), but (2) has a much better Recall (from 0.37370 to 0.45316), which translates into a better F1 for (2) (from 0.46641 to 0.47547).

In general, PLM-ICD-C (2) has the best F1-score (0.47547), in which it is slightly better than PLM-ICD threshold 0.2 (0.46970), but a much better Precision (from 0.46282 to 0.50008), which may improve the code suggestions.

The three rules defined for PLM-ICD-C, show that they improve the results. For rule number 1, if PLM-ICD finds a code with high probability of being a positive, there is no need to screen it again, being immediately accepted as a positive. For rule number 2, ICS is called to disambiguate, because PLM-ICD's confidence is just borderline between positive or negative. Finally, rule number 3 determines that when PLM-ICD gives a very low probability of being positive to a code, it is not screened by ICS, being accepted as a negative candidate.

7.3 Summary

While using the Harvard Medical School dataset, the work in terms of programming code includes an improvement of reading ICD-10-CM information from a JSON file instead of Mongo, the use of a blacklist containing selected words, the user of a NER model, detection of negation of a disease entity, detection of clinical acronyms, implementation of multiple runs, and use of trigrams before the disease entity to give more context to the codification. These strategies resulted in improvements to the codification results.

For the MIMIC-IV dataset, the work includes the use of trigrams before and after the disease entity, the use of a list of synonyms, and the use of a strategy of bucket concept. While running the Improved Cosine Similarity method, with the bucket category concept, over the test dataset of 4077 EHRs, resulted in a 5-fold improvement for the Precision, 2-3 fold improvement in Recall, and 4-fold improvement in F1-score, comparing to not using the bucket category concept.

The run of PLM-ICD over the MIMIC-IV dataset, was initially performed in Google Colaboratory, but the limitations of memory and time could not allow to train the model in this platform. Instead, the resources of INCD were used. The results show that PLM-ICD performs better than the ICS method.

A proposal to enhance the PLM-ICD model was made (PLM-ICD-C), that uses the ICS method including the multiple runs and bucket category algorithms. The three rules defined for PLM-ICD-C, show that they improve the results. For rule number 1, if PLM-ICD finds a code with high probability of being a positive, there is no need to screen it again, being immediately accepted as a positive. For rule number 2, ICS is called to disambiguate, because PLM-ICD's confidence is just borderline between positive or negative. Finally, rule number 3 determines that when PLM-ICD gives a very low probability of being positive to a code, it is not screened by ICS, being accepted as a negative candidate. In general, PLM-ICD-C has the best F1-score (0.47547), in which it is slightly better than PLM-ICD (0.46970), but a much better Precision (from 0.46282 to 0.50008), which may improve the code suggestions.

8 Discussion

This chapter discusses the results, regarding the use of the Improved Cosine method and PLM-ICD-C, the answers to the research questions, as well as the contributions to the state-of-the-art.

8.1 Data limitations

Regarding the datasets, namely the labeled MIMIC-IV dataset, the health records were produced from the emergency department of a single hospital [20]. For that reason it is a source of possible bias in data. Also, the process of coding is prone to errors, with a degree of subjectivity. Shorter summaries from outpatients are easier to code than for inpatients. Burns, et al. (2012) [50] found an overall median accuracy of 83.2%, and Searle, et al. (2020) [51] concluded that 35% of the common codes in MIMIC-III, were under-coded.

8.2 Cosine method

The cosine method is negatively impacted when there are few or no biomedical words in the text. This is the case for MIMIC-IV dataset, where many codes and their biomedical words are not present in the EHR text, making it difficult to obtain high similarity with this method. It is also very sensitive to words such as "patient" and "hospital", which are semantically empty for the purpose. With the strategies of blacklist of words, the novel implemented multiple runs and bucket category selection algorithms, these drawbacks were mitigated, leading to a novel algorithm called Improved Cosine Similarity method.

8.3 PLM-ICD-C

Tables 7.12 and 7.13 show that for samples where $S_P \geq T_1$, with values of T_1 0.7, 0.5, 0.45, the use of PLM-ICD-C results in a loss of TP greater than the FP filtered out, compared to the use of PLM-ICD alone, except for $T_1 = 0.4$, which is exemplified in detail in Figure 7.1.

On the other hand, in the interval between T_2 and T_1, with values of 0.7, 0.5, 0.45 for T_1, and 0.4 for T_2, the balance is positive, as it can be seen in Figure 7.2, where for the

use of PLM-ICD-C, there is a loss of TP inferior to filtered FP. For T_1 value of 0.4 and T_2 value of 0.2, the balance is positive with a bigger margin.

The use of ICS along with PLM-ICD for $T_2 \leq S_P < T_1$, translates into better performance metrics, because it can filter more FP than using PLM-ICD alone in that interval. In general, it was demonstrated that the use of PLM-ICD-C is more advantageous than using the PLM-ICD alone, due to the greatly improved Precision, from 46.3% to 50%.

8.4 Advantages and limitations

With the optimized values for the thresholds T_1 and T_2, the use of the Improved Cosine Similarity method coupled to PLM-ICD can be advantageous, allowing to reach more TP while filtering out some possible FP that would slip through PLM-ICD alone and therefore affect Precision.

The cosine method has one important characteristic, that it does not require training to specific data, for it is a deterministic model, so there is no learning bias. The multiple runs and bucket category methods are also deterministic, even though specific parameters can be adjusted for better performance.

In future work, it will be useful to use not only the NER model for diseases and chemicals, as was done, but for other biomedical terms too. The NER model used, "en_ner_bc5cdr_md" [1], only recognizes entities of diseases and chemicals. For example, the entity "pregnancy" is neither a disease nor a chemical. Nonetheless, it is present in some ICD-10 code descriptions and may be important for accurate code retrieval. It is not recognized by this NER model, thus not contributing to retrieve important ICD-10 codes when calculating the similarity between the EHR text and ICD-10 descriptions.

In the cosine method implementation, a few synonyms for diseases were considered, namely "angioectasia" as "angiodysplasia", "dyslipidemia" as "hyperlipidemia", "tobacco" as "nicotine", and "smoking" as "nicotine", making it limited in the vocabulary and consequently in the results. A more comprehensive list of synonyms will most probably lead to better results, but it was out of the scope of the current project.

In general, PLM-ICD-C can be useful to provide code suggestions to the medical coders, and suggest more of the codes that the clinical record contains than PLM-ICD alone.

The computation required to perform experiments with MIMIC-IV makes it difficult to optimize the values of the thresholds used. Additionally, the experiments described were just performed with one part of the whole MIMIC-IV dataset. Therefore, there may still be room for improvement.

[1] https://allenai.github.io/scispacy/, accessed on 2023-10-10

8.5 Answers to the research questions

As a result of this work, the research questions formulated in Section 1.2 can now be answered, as described below:

- Is there a way to improve the state-of-the-art in EHR coding, using the cosine text similarity method? It was demonstrated that the use of the Improved Cosine Similarity method, can improve the PLM-ICD method, although for a different dataset size, but using the same settings for its creation;
- Is it possible to improve the results of the PLM-ICD as a tool to aid in the EHR encoding process? With the results of PLM-ICD for the smaller dataset size, ICS method improved the F1-score by 0.5%, but more important it improved the Precision from 46.3% to 50%;
- Is the cosine method appropriate to improve PLM-ICD metrics? It was demonstrated that ICS method improves the PLM-ICD metrics, although there is space for optimization and improvement of the metrics, so it can be stated that cosine method is one of the possible and appropriate methods to improve PLM-ICD metrics.

8.6 Contributions to the state-of-the-art

The main contributions to the state-of-art are:

- Two publications, [52] and [53], in TecnoHospital [2], a National Technical Journal on the area of Engineering and Health Management;
- A third publication [49] was published in MDPI Algorithms Scientific Journal [3], and was selected by the editors as the cover story of the April 2024 issue;
- Novel implemented multiple runs and bucket category selection algorithms, leading to a novel algorithm called Improved Cosine Similarity;
- Improvement to the PLM-ICD model performance metrics, with the previous ICS algorithm, which was called PLM-ICD-C;
- Answered the proposed research questions, with the improvement of the state-of-the-art in EHR coding, with space for optimization and improvement of the metrics.

The first article, in the publication nº 118 of TecnoHospital, presents a literature review on the topic of coding clinical episodes in natural language. The main problems and barriers that affect it are mentioned. Previous work is presented, using natural language processing in parallel with ontologies and the benefit of their joint use. The use

[2] http://www.tecnohospital.pt/, accessed on 2024-05-09
[3] https://www.mdpi.com/journal/algorithms, accessed on 2024-05-09

of natural language processing techniques restricted to the health area is analyzed. Finally, ICD-10 automatic coding works are presented, with a description of the methodologies used. This literature review aims to comprehensively study the state-of-the-art techniques that have been applied in general in the health area, and aims to guide readers in the automatic coding techniques of clinical episodes with a view to developing of methods that facilitate the manual coding process through effective computational methods.

The second article, in the publication nº 121 of TecnoHospital, aims to develop a method to propose useful ICD-10 code suggestions to coders, aiming to facilitate the process. The methodology is based on the similarity between the clinical text and the description of the ICD-10 code, where the tests were carried out with part of the MIMIC-IV dataset. The results show that the strategy of using the bucket category concept improves results, by providing useful suggestions, with a micro-average F1-score of 25.6%.

The third article in the Volume 17, Issue 4, of MDPI Algorithms, proposes a method where cosine text similarity is combined with a pretrained language model, PLM-ICD, in order to increase the number of probably useful suggestions of ICD-10 codes, based on the MIMIC-IV dataset. The results show that a strategy of using multiple runs, and bucket category search, in the cosine method, improves the results, providing more useful suggestions. Also, the use of a strategy composed by the cosine method and PLM-ICD, PLM-ICD-C, provides better results than just the PLM-ICD.

9 Conclusion

The classification of EHRs from text into diagnostic codes has been challenging to the NLP community. The proposed methodology relies on a method based on cosine similarity between the clinical record and the ICD-10 code description, using a strategy of multiple runs algorithm along with the concept of bucket category.

Comparing the F1-score obtained with PLM-ICD for the test set used, it is inferior to the best obtained in the state-of-the-art (46.9% to 58.5%). Since no training or architectural changes were made to PLM-ICD in the current project, the only reasonable explanation for the different results is the difference in the test set used, because Edin et al. (2023) [20] use a bigger test set consisting of 24,461 records, in comparison to the 4077 used in this work because of limited computational resources, namely the fact that in the INCD used environment, there was a computational limitation of a task for a maximum duration of 4 days.

The results show that the use of PLM-ICD-C, consisting of the Improved Cosine Similarity method and PLM-ICD, improve the results, increasing the F1-score by 0.5%, but most important, by increasing the Precision from 46.3% to 50%, which means a significant improvement on the code suggestions given to the medical doctors performing encoding functions.

For values of S_P above T_1 (0.4) or below T_2 (0.2), the use of ICS is not advantageous where it is more reliable to use only the PLM-ICD. For values of S_P between T_2 and T_1 values, then PLM-ICD-C, with the use of ICS, is a better choice.

In future work, the Improved Cosine Similarity method can be applied to other datasets that overcome the limitations that MIMIC-IV may have. The dictionary of synonyms, where only four pairs were used, just as a proof of concept, shall also be expanded. A more complete NER model can also be used, to detect other biomedical terms, and not only diseases. Finally, in this project it was not taken in consideration the possibility of having orthographic errors that EHRs may contain. For this, in future work would be interesting to apply techniques that can identify and correct these possible errors.

References

[1] "International classification of diseases 11th revision," https://www.who.int/standards/classifications/classification-of-diseases [Accessed: (2024-03-10)].

[2] V. Alonso, J. V. Santos, M. Pinto, J. Ferreira, I. Lema, F. Lopes, and A. Freitas, "Health records as the basis of clinical coding: Is the quality adequate? a qualitative study of medical coders' perceptions," *Health Information Management Journal*, vol. 49, no. 1, pp. 28–37, 2020.

[3] K. Lucyk, K. Tang, and H. Quan, "Barriers to data quality resulting from the process of coding health information to administrative data: a qualitative study," *BMC health services research*, vol. 17, no. 1, pp. 1–10, 2017.

[4] K. L. Tang, K. Lucyk, and H. Quan, "Coder perspectives on physician-related barriers to producing high-quality administrative data: a qualitative study," *Canadian Medical Association Open Access Journal*, vol. 5, no. 3, pp. E617–E622, 2017.

[5] S. Abburu and S. B. Golla, "Ontology and nlp support for building disaster knowledge base," in *2017 2nd International Conference on Communication and Electronics Systems (ICCES)*. IEEE, 2017, pp. 98–103.

[6] K. Liu, W. R. Hogan, and R. S. Crowley, "Natural language processing methods and systems for biomedical ontology learning," *Journal of biomedical informatics*, vol. 44, no. 1, pp. 163–179, 2011.

[7] A. Ayadi, M. Auffan, and J. Rose, "Ontology-based nlp information extraction to enrich nanomaterial environmental exposure database," *Procedia Computer Science*, vol. 176, pp. 360–369, 2020.

[8] A. Kormilitzin, N. Vaci, Q. Liu, and A. Nevado-Holgado, "Med7: A transferable clinical natural language processing model for electronic health records," *Artificial Intelligence in Medicine*, vol. 118, p. 102086, 2021.

[9] I. Li, J. Pan, J. Goldwasser, N. Verma, W. P. Wong, M. Y. Nuzumlalı, B. Rosand, Y. Li, M. Zhang, D. Chang *et al.*, "Neural natural language processing for unstructured data in electronic health records: A review," *Computer Science Review*, vol. 46, p. 100511, 2022.

[10] P.-F. Chen, S.-M. Wang, W.-C. Liao, L.-C. Kuo, K.-C. Chen, Y.-C. Lin, C.-Y. Yang, C.-H. Chiu, S.-C. Chang, F. Lai *et al.*, "Automatic icd-10 coding and training system: deep neural network based on supervised learning," *JMIR Medical Informatics*, vol. 9, no. 8, p. e23230, 2021.

[11] Y. Chen, H. Lu, and L. Li, "Automatic icd-10 coding algorithm using an improved longest common subsequence based on semantic similarity," *PloS one*, vol. 12, no. 3, p. e0173410, 2017.

[12] W. Ning, M. Yu, and R. Zhang, "A hierarchical method to automatically encode chinese diagnoses through semantic similarity estimation," *BMC medical informatics and decision making*, vol. 16, no. 1, pp. 1–12, 2016.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[14] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "Biobert: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[15] E. Alsentzer, J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. McDermott, "Publicly available clinical bert embeddings," *arXiv preprint arXiv:1904.03323*, 2019.

[16] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon, "Domain-specific language model pretraining for biomedical natural language processing," *ACM Transactions on Computing for Healthcare* (*HEALTH*), vol. 3, no. 1, pp. 1–23, 2021.

[17] P. Lewis, M. Ott, J. Du, and V. Stoyanov, "Pretrained language models for biomedical and clinical tasks: understanding and extending the state-of-the-art," in *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, 2020, pp. 146–157.

[18] Z. Zhang, J. Liu, and N. Razavian, "Bert-xml: Large scale automated icd coding using bert pretraining," *arXiv preprint arXiv:2006.03685*, 2020.

[19] C.-W. Huang, S.-C. Tsai, and Y.-N. Chen, "Plm-icd: automatic icd coding with pretrained language models," *arXiv preprint arXiv:2207.05289*, 2022.

[20] J. Edin, A. Junge, J. D. Havtorn, L. Borgholt, M. Maistro, T. Ruotsalo, and L. Maaløe, "Automated medical coding on mimic-iii and mimic-iv: A critical review and replicability study," *arXiv preprint arXiv:2304.10909*, 2023.

[21] C. Sánchez and P. Martínez, "A simple method to extract abbreviations within a document using regular expressions," in *IberEval@ SEPLN*, 2018, pp. 297–301.

[22] A. Jaber and P. Martínez, "Disambiguating clinical abbreviations using pre-trained word embeddings," in *HEALTHINF*, 2021, pp. 501–508.

[23] M. Skreta, A. Arbabi, J. Wang, E. Drysdale, J. Kelly, D. Singh, and M. Brudno, "Automatically disambiguating medical acronyms with ontology-aware deep learning," *Nature communications*, vol. 12, no. 1, p. 5319, 2021.

[24] C. Dalloux, V. Claveau, N. Grabar, L. E. S. Oliveira, C. M. C. Moro, Y. B. Gumiel, and D. R. Carvalho, "Supervised learning for the detection of negation and of its scope in french and brazilian portuguese biomedical corpora," *Natural Language Engineering*, vol. 27, no. 2, pp. 181–201, 2021.

[25] H. Tanushi, H. Dalianis, M. Duneld, M. Kvist, M. Skeppstedt, and S. Velupillai, "Negation scope delimitation in clinical text using three approaches: Negex, pycontextnlp and synneg," in *19th Nordic Conference of Computational Linguistics (NODALIDA 2013), May 22-24, 2013, Oslo, Norway*. Linköping University Electronic Press, 2013, pp. 387–474.

[26] "Classificação internacional de doenças," https://revista.abrale.org.br/classificacao-internacional-de-doencas-cid/ [Accessed: (2023-07-10)].

[27] "What is the difference between icd-10-cm and icd-10-pcs," https://www.aapc.com/support/books/what-is-the-difference-between-icd-10-cm-and-icd-10-pcs [Accessed: (2024-03-20)].

[28] M. C. Libicki and I. T. Brahmakulam, "The costs and benefits of moving to the icd-10 code sets," 2004.

[29] M. M. Care, F. H. Kids, and M. Advantage, "Icd-10-cm official coding and reporting guidelines april 1, 2020, through september 30, 2020," 2020.

[30] "International classification of diseases, (icd-10-cm/pcs) transition – background," https://www.cdc.gov/nchs/icd/icd10cm_pcs_background.htm [Accessed: (2024-04-27)].

[31] "Icd-10 know the codes," https://www.ibx.com/documents/35221/56659/icd-10-know-the-codes.pdf [Accessed: (2024-01-03)].

[32] "Snomed ct – o que é, e porque não nos podemos dar ao luxo de não o implementar," https://www.xn--tempo-semntico-jhb.pt/snomed-ct-nao-nos-podemos-dar-ao-luxo-nao-implementar/ [Accessed: (2023-06-01)].

[33] M. Mandal, "Introduction to convolutional neural networks (cnn)," https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/ [Accessed: (2024-04-03)].

[34] W. Badr, "Auto-encoder: What is it? and what is it used for? (part 1)," https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726 [Accessed: (2024-04-03)].

[35] "Introduction to seq2seq models," https://www.analyticsvidhya.com/blog/2020/08/a-simple-introduction-to-sequence-to-sequence-models/ [Accessed: (2024-04-08)].

[36] "Natural language toolkit - parsing," https://www.tutorialspoint.com/natural_language_toolkit/natural_language_toolkit_parsing.htm [Accessed: (2024-04-09)].

[37] N. Seth, "Syntactical parsing in nlp," https://www.analyticsvidhya.com/blog/2022/03/syntactical-parsing-in-nlp/ [Accessed: (2024-04-09)].

[38] A. Pai, "What is tokenization in nlp? here's all you need to know," https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/ [Accessed: (2024-04-10)].

[39] K. Ganesan, "All you need to know about text preprocessing for nlp and machine learning," https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html [Accessed: (2024-04-10)].

[40] K. M. Rosaria Silipo, "Text encoding: A review," https://towardsdatascience.com/text-encoding-a-review-7c929514cccf [Accessed: (2024-04-10)].

[41] F. Teng, Y. Liu, T. Li, Y. Zhang, S. Li, and Y. Zhao, "A review on deep neural networks for icd coding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 5, pp. 4357–4375, 2022.

[42] "Icd-10-cm," https://www.medicalbillingandcoding.org/icd-10-cm/ [Accessed: (2024-04-17)].

[43] A. E. Johnson, L. Bulgarelli, L. Shen, A. Gayles, A. Shammout, S. Horng, T. J. Pollard, S. Hao, B. Moody, B. Gow *et al.*, "Mimic-iv, a freely accessible electronic health record dataset," *Scientific data*, vol. 10, no. 1, p. 1, 2023.

[44] M. Gupta, B. Gallamoza, N. Cutrona, P. Dhakal, R. Poulain, and R. Beheshti, "An extensive data processing pipeline for mimic-iv," in *Machine Learning for Health*. PMLR, 2022, pp. 311–325.

[45] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein, "Explainable prediction of medical codes from clinical text," *arXiv preprint arXiv:1802.05695*, 2018.

[46] "Understanding cosine similarity and its applications," https://builtin.com/machine-learning/cosine-similarity [Accessed: (2024-04-20)].

[47] "Tf-idf — term frequency-inverse document frequency," https://www.learndatasci.com/glossary/tf-idf-term-frequency-inverse-document-frequency/ [Accessed: (2024-04-20)].

[48] T. Vu, D. Q. Nguyen, and A. Nguyen, "A label attention model for icd coding from clinical text," *arXiv preprint arXiv:2007.06351*, 2020.

[49] H. Silva, V. Duque, M. Macedo, and M. Mendes, "Aiding icd-10 encoding of clinical health records using improved text cosine similarity and plm-icd," *Algorithms*, vol. 17, no. 4, p. 144, 2024.

[50] E. M. Burns, E. Rigby, R. Mamidanna, A. Bottle, P. Aylin, P. Ziprin, and O. Faiz, "Systematic review of discharge coding accuracy," *Journal of public health*, vol. 34, no. 1, pp. 138–148, 2012.

[51] T. Searle, Z. Ibrahim, and R. J. Dobson, "Experimental evaluation and development of a silver-standard for the mimic-iii clinical coding dataset," *arXiv preprint arXiv:2006.07332*, 2020.

[52] H. Silva, V. Duque, M. Macedo, and M. Mendes, "Inovação e novas tecnologias em saúde - parte i," *TecnoHospital*, no. 118, 2023.

[53] H. Silva, V. Duque, M. Macedo, and M. Mendes, "Fronteiras de impacto da inteligência artificial na saúde," *TecnoHospital*, no. 121, 2024.

Annexes

Annex A - Dictionary of Acronyms

A.A.R.O.M.: active assistive range of motion

AAROM: active assistive range of motion

AAC: augmentative and alternative communication

A.B.G: arterial blood gas

ABG: arterial blood gas

afib: atrial fibrillation

AFib: atrial fibrillation

Afib: atrial fibrillation

AKA: above-knee amputation or above-the-knee amputation

ALS: amyotrophic lateral sclerosis

AMA: against medical advice

A.R.O.M.: active range of motion

AROM: active range of motion

ASAP: as soon as possible

ASD: autism spectrum disorder

BKA : below-knee amputation

BMR : basal metabolism rate

BP : blood pressure

BR : bed rest

BS : breath sounds

C1 : first cervical vertebrae

C2 : second cervical vertebrae

CA : cardiac arrest

CABG : coronary artery bypass graft

CAD : coronary artery disease

CBC : complete blood count

CHF : congestive heart failure

CCU : coronary care unit

CHI : closed head injury

CMT : continuing medication and treatment

CN : cranial nerve

CNA : certified nursing assistant

CNS : central nervous system

COTA : certified occupational therapy assistant

COPD : chronic obstructive pulmonary disease CP : cerebral palsy

CPAP : continuous positive airway pressure

CPR : cardiopulmonary resuscitation

CRF : chronic renal failure

CRNP : certified registered nurse practitioner

CSF : cerebrospinal fluid

CT : computerized tomography

CV : cardiovascular

CVA : cerebral vascular accident

CXR : chest X-ray

DC : discharge

DM : diabetes mellitus

DM2 : diabetes mellitus type 2

DNR : do not resuscitate

DNT : did not test

DOA : dead on arrival

DOB : date of birth

DOE : dyspnea on exertion

ECC : electrocardiogram

EKG : electrocardiogram

ED : emergency department

EEG : electroencephalogram

EENT : eyes, ears, nose, throat

EMG : electromyogram

ENT : ears, nose, throat

ER : emergency room

ETOH : ethanol

FH : family history

FOB : foot of bed

FWB : full weight bearing

GB : gall bladder

GCS : Glasgow Coma Scale

GE : gastroenterology

GERD : gastro-esophageal reflux disease

G/E : gastroenteritis

G.I. : gastrointestinal

GI : gastrointestinal

GNA : geriatric nursing assistant

GP : general practitioner

GSW : gunshot wound

GTT : glucose tolerance test

GYN : gynecology

H/A : headache

HAV : hepatitis A virus

HB : heart block

HBP : high blood pressure

HEENT : head, eyes, ears, nose, throat

HEP : home exercise program

H2O : water

HIV : human immunodeficiency virus

H.I.V : human immunodeficiency virus

HLD : hyperlipidemia

HOB : head of bed

H&P : history and physical

HR : heart rate

HTN : hypertension

HVD : hypertensive vascular disease

ICCU : intensive coronary care unit

ICP : intracranial pressure

ICU : intensive care unit

I&O : intake and output

IPPB : intermittent positive pressure breathing

IV : intravenous

L2 : second lumbar vertebrae

L3 : third lumbar vertebrae

LBW : low birth rate

L.E. : lower extremities

LE : lower extremities

L.O.C. : loss of consciousness

LOC : loss of consciousness

LOS : length of stay

LP : lumbar puncture

LPN : licensed practical nurse

LUE : left upper extremity

L&W : living and well

MBC : maximum breathing capacity

MBSS : modified barium swallow study

MCA : middle cerebral artery

MD : muscular dystrophy

MG : myasthenia gravis

MI : myocardial infarction

MICU : medical intensive care unit

MRI : magnetic resonance imaging

MRSA : methicillin-resistant Staphylococcus aureus

MVA : motor vehicle accident

NAD : no abnormality detected

NG : nasogastric

NIC : neonatal intensive care

NICU : neonatal intensive care unit

NKA : no known allergies

NOS : not otherwise specified

NPO : nothing by mouth

NSA : no specific abnormality

NST : nonstress test

N&V : nausea and vomiting

NVD : nausea, vomiting, diarrhea

N&W : normal and well

NWB : non–weight bearing

NYD : not yet diagnosed

OA : osteoarthritis

OB : obstetrics

OBG : obstetrics

OB/GYN : obstetrics and gynecology

OBS : organic brain syndrome

ODD : oppositional defiant disorder

O/E : on examination

OH : occupational history

OHD : organic heart disease

O.M. : otitis media

OM : otitis media

O.M.E. : otitis media with effusion

OME : otitis media with effusion

OOB : out of bed

OTC : over-the-counter

O.T. : occupational therapy

OT : occupational therapy

OR : operating room

OSA : obstructive sleep apnea

PA : physician's assistant

PACU : post anesthesia care unit

PAF : paroxysmal atrial fibrillation

PA : posterior–anterior view on X-ray

PCP : pneumocystis pneumonia

PD : parkinson's disease

PDN : private duty nurse

PE : physical exam

PEEP : positive end-expiratory pressure

PEG : percutaneous endoscopic gastrostomy

PET : positron emission tomography

PH : past history

PI : present illness

PICU : pulmonary intensive care unit

PID : pelvic inflammatory disease

P.M : postmortem

PMH : past medical history

PMR : physical medicine and rehabilitation

PN : poorly nourished

P&N : psychiatry and neurology

PNA : pneumonia

PNI : peripheral nerve injury

PNX : pneuomothorax

PR : proctology

PROM : passive range of motion

PSH : past surgical history

PT : physical therapy

P.T. : physical therapy

PTA : prior to admission

PTA pulse : posterior tibial artery pulse

PUD : peptic ulcer disease

PVD : peripheral vascular disease

PVT : previous trouble

PWB%: partial weight bearing with percent

PX : physical examination

RA : rheumatoid arthritis

RBC : red blood cell

RCA : right coronary artery

RCU : respiratory care unit

RD : respiratory distress

RDS : respiratory distress syndrome

RE : reconditioning exercise

RF : rheumatic fever

RLAS : Rancho Los Amigos Scale

RLE : right lower extremity

RN : registered nurse

RND : radical neck dissection

RO : rule out

ROM : range of motion

ROS : review of symptoms

RT : radiation therapy

RUE : right upper extremity

RV : residual volume

RW : rolling walker

SCCA : squamous cell carcinoma

SCD : sudden cardiac death

SCI : spinal cord injury

SCU : special care unit

SGA : small for gestational age

SH : social history

SI : stroke index

SICU : surgical intensive care unit

SIDS : sudden infant death syndrome

SL : under the tongue

SLP : speech-language pathologist

SNF : skilled nursing facility

SOAP : subjective, objective, assessment, plan

SOB : shortness of breath

SS : social service

ST : speech therapy

STAT : immediately

STD : sexually transmitted disease

T&A : tonsils and adenoids

TAH : total abdominal hysterectomy

TB : tuberculosis

TBI : traumatic brain injury

THR : total hip replacement

TIA : transient ischemic attack

TKR : total knee replacement

TNM : tumor, nodes, and metastases

TO : telephone order

TPN : total parenteral nutrition

TPR : temperature, pulse, respiration

U/A : urinalysis

UCD : usual childhood diseases

UCHD : usual childhood diseases

UG : upward gaze

URD : upper respiratory disease

URI : upper respiratory infection

UTI: urinary tract infection

VA: visual acuity

VC: vital capacity

VD: venereal disease

VF: visual fields

VFSS: videofluoroscopic swallowing study

VN: visiting nurse

VO: verbal order

VS: vital signs

V.S.: vital signs

W/C: wheelchair

WBT: weight bearing tolerance

WFL: within functional limits

WNL: within normal limits

WP: whirlpool

Instituto Superior de Engenharia
Politécnico de Coimbra