Imperial College London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Towards Real-Time ICD Coding: Predictive Temporal Modelling of Electronic Health Records

Author:
Mireia Hernandez Caralt

Supervisor:
Dr. Marek Rei
Clarence Boon Liang Ng

Submitted in partial fulfillment of the requirements for the MSc degree in MSc Artificial Intelligence of Imperial College London

June 2023

Abstract

This thesis explores real-time ICD code prediction through temporal modelling of Electronic Health Records (EHR). Automated ICD coding consists in predicting standardized disease and procedure labels from clinical documents. While EHRs contain a wealth of notes generated during a patient's stay, existing methods focus on classifying the last document, known as the discharge summary. Consequently, these models cannot be employed before a patient's discharge for real-time tracking of diagnoses and forecasting future procedures. To bridge this gap, we introduce a novel task: real-time ICD code prediction, and propose a model architecture, based on hierarchical transformers and masked attention, that can predict diagnoses and procedures at any point throughout the hospital stay using the notes generated thus far. We evaluate the model with a cut-off based framework, and examine the predictive power of early documents for ICD-9 code forecasting across various dataset configurations. Additionally, we show that early ICD code prediction can be improved by incorporating auxiliary tasks during training to refine document embeddings with domain-specific knowledge. Furthermore, we develop an algorithm to address the limitations of current hierarchical models in handling the complete EHR sequence. This algorithm enables predictions based on extensive amounts of text, resulting in significant performance improvements for temporal ICD coding. Finally, we conduct an interpretability analysis on our model, shedding light on the types of documents, apart from the discharge summary, that offer the most value for early diagnosis.

Keywords: ICD coding, Clinical Notes, Electronic Health Records, Transformers, Hierarchical Transformers, Multi-Objective Learning, Transformers For Long Text, Natural Language Processing

Acknowledgments

First, I would like to express my gratitude to my supervisor, Dr. Marek Rei, for his invaluable guidance throughout this project. I want to extend my thanks for his constant support and encouragement during this challenging journey, and for generously sharing many new research ideas that helped me advance this project further.

I also want to thank Clarence Ng, who was always available to assist me with topics related to his previous work. He consistently met with me every week, offering valuable insights and suggestions.

Lastly, I want thank my family for their unconditional love and support throughout my studies.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Aims . . . 2
- 1.3 Contributions . . . 3
- 1.4 Ethical considerations . . . 3

2 Background **5**
- 2.1 Clinical Diagnosis Prediction . . . 5
  - 2.1.1 ICD-code prediction problem . . . 5
  - 2.1.2 MIMIC dataset . . . 6
- 2.2 Transformers . . . 6
  - 2.2.1 Architecture . . . 7
  - 2.2.2 Usage methods . . . 9
  - 2.2.3 Unsupervised pre-training . . . 10
  - 2.2.4 Adapting to long text sequences . . . 11
- 2.3 Models For ICD-Code Prediction . . . 12
  - 2.3.1 Label-wise attention network . . . 12
  - 2.3.2 CNN and LSTM-based methods . . . 13
  - 2.3.3 Transformer-based methods . . . 14
  - 2.3.4 Document-level methods . . . 16
- 2.4 Research on the full EHR Sequence . . . 20
  - 2.4.1 Readmission prediction . . . 21
  - 2.4.2 Discharge summary generation . . . 21

3 Experimental set-up **22**
- 3.1 Dataset Access . . . 22
- 3.2 Dataset Preprocessing . . . 23
- 3.3 Exploratory Analysis . . . 24
  - 3.3.1 Qualitative discussion . . . 24
  - 3.3.2 EHR note distribution . . . 25
- 3.4 Temporal ICD-9 Coding Task . . . 27
  - 3.4.1 Task specification . . . 27
  - 3.4.2 Evaluation framework . . . 29
  - 3.4.3 Evaluation metrics . . . 30
- 3.5 Task Set-ups . . . 31
  - 3.5.1 “Latest” set-up . . . 31

3.5.2 "Random" set-up . . . 32
3.5.3 "Limited DS" set-up . . . 33

4 Model Design **34**
4.1 Non-Temporal Model . . . 34
4.1.1 Chunk splitting . . . 34
4.1.2 Pre-trained transformer encoder . . . 35
4.1.3 Label-wise attention . . . 35
4.1.4 Binary cross-entropy loss . . . 36
4.2 Temporal Model: MMULA . . . 36
4.2.1 Chunk representation . . . 37
4.2.2 Masked hierarchical transformer . . . 37
4.2.3 Masked multi-head label-wise attention . . . 38

5 Enhancing early prediction **40**
5.1 Multi-Objective Training . . . 40
5.1.1 Document Category prediction . . . 40
5.1.2 Document Embedding prediction . . . 42
5.2 Extended-Length Algorithm . . . 43
5.2.1 Complete EHR distribution set-up . . . 43
5.2.2 Algorithm specification . . . 44

6 Evaluation **47**
6.1 Temporal evaluation of MMULA . . . 47
6.1.1 Model tuning . . . 47
6.1.2 Set-up experiments . . . 48
6.1.3 Analysis of temporal predictions . . . 49
6.2 Evaluation of enhancement proposals . . . 50
6.2.1 Auxiliary task comparison . . . 51
6.2.2 Extended-length algorithm (ELA) evaluation . . . 52
6.3 Ablation studies . . . 53
6.3.1 Effect of document embedding generator . . . 53
6.3.2 Effect of random sampling in ELA . . . 53
6.4 Test set evaluation . . . 54
6.5 Model Interpretability . . . 55
6.5.1 Explaining local predictions . . . 56
6.5.2 Attention across document categories . . . 57

7 Conclusion and Future Works **59**
7.1 Conclusion . . . 59
7.2 Limitations and future work . . . 60

Chapter 1

Introduction

1.1 Motivation

This thesis proposes a novel task of real-time ICD code prediction and explores temporal modelling of EHR (Electronic Health Record) sequences. The International Classification of Diseases (ICD) is a classification system developed by the World Health Organization (WHO) to standardize the collection of disease and procedure information [1]. The ICD coding tasks refers to the prediction of diagnoses and procedures based on clinical documents from the Electronic Health Record. When done manually, this task is time-consuming and error-prone due to the high specificity and technicality of the ICD taxonomy. For this reason, research efforts have focused on developing automated NLP approaches for ICD coding, which aim to reduce the medical burden placed on hospital workers [2].

Current state-of-the-art models are usually based on hierarchical transformer models and rely on classifying the discharge summary document, which is generated after the patient's discharge and contains a concise overview of the hospital course and the corresponding diagnoses and procedures [2]. However, the complete EHR sequence is much longer and contains numerous notes that are generated throughout the hospital stay, including nursing reports, and specialized notes in radiology, electrocardiograms and echographies, among others. These notes provide live updates of the patient's condition and can be used to develop a system that can make real-time predictions of ICD codes. For this reason, in this project we explore the novel task of predicting ICD codes throughout a patient stay using the clinical notes available, without relying on the discharge summary.

The task of real-time ICD code prediction presents additional challenges compared to the standard post-discharge coding task. The complete EHR sequence's length exceeds that of the discharge summary, which poses a challenge even for hierarchical transformer models. Valuable information for ICD code prediction is likely to be scattered throughout the sequence, necessitating the exploration of models or algorithms capable of effectively handling extensive text. Furthermore, these notes are typically less concise and summarized, providing a weaker signal for ICD-9 code

prediction. This project's objective is to investigate the feasibility of utilizing these notes for real-time ICD-9 code prediction and to explore algorithms and methods that enhance the model's capacity to handle long text and make early predictions.

The potential applications of real-time ICD-9 coding are very diverse. Firstly, such a system goes beyond the post-discharge diagnostic practices and it can be used for prognosis, identifying early disease risks and potential procedures to be realized throughout a patient's hospital stay. Real-time ICD-9 coding can be used to guide clinicians in diagnostic practices and aid hospitals in optimizing resource allocation, for instance, by enabling them to proactively plan for specialists or operation theater requirements depending on forecasted procedures. Another potential application of early prediction of ICD-9 codes is for billing planning and estimating total cost before a patient is discharged. Furthermore, such a system could reduce the workload on medical staff by eliminating the need to generate a highly detailed discharge summary to make ICD-9 code predictions, which is very time-consuming. Alternatively, the development of an interpretable temporal model could assist clinicians in the creation of the discharge summary by identifying crucial documents within the lengthy EHR sequence that are important for the final ICD-9 code predictions.

1.2 Aims

The objective of this project is to bridge an existing gap in ICD-9 coding research by developing a model capable of making accurate predictions throughout a patient's hospital stay, without relying solely on the discharge summary. This will require the exploration of the complete EHR sequence, which contains a variety of notes aside from the discharge summary, and is an untapped source of information which could be valuable for real-time disease prediction. The project aims to define a novel task concerning temporal ICD-9 code prediction, and establishing a baseline in the literature with a standardized evaluation framework. The ulterior goal is to open avenues for real-time disease monitoring, procedure prediction and improvement of hospital resource planning.

More specifically, the project will address the following research questions:

- **RQ1**: How to design a model for ICD-coding that can make predictions at any point in time? This includes defining a new prediction task on the MIMIC-III dataset, as well as a model architecture that can make predictions at any point during the hospital stay by looking only at past notes. The model predictions are expected to get better as more information becomes available.
- **RQ2**: What auxiliary or pre-training tasks can improve the model training? For instance, exploiting unsupervised pre-training on a document-level, or defining auxiliary losses for predicting future document embeddings.
- **RQ3**: Which methods can be used to overcome the sequence length limitation of current models? The complete EHR sequence is far longer than the discharge

summary, which calls for a methodology to make predictions based on a very extended amount of text.

1.3 Contributions

The project explores the research questions detailed in Section 1.2, and as a result, makes the following contributions:

- The establishment of a novel task of temporal ICD-9 code prediction, and a corresponding evaluation framework to allow for model comparison.
- The exploration of multiple dataset configurations of the complete Electronic Health Record sequence for the novel temporal ICD-9 code prediction task.
- The development of a temporal model that can make ICD-9 code predictions at any point throughout a patient's hospital stay using the notes available up that point.
- The specification of novel auxiliary tasks oriented towards the enhancement of temporal ICD-9 code predictions.
- The proposal of an algorithm to overcome sequence length limitations of current hierarchical transformer models.
- The exploration of model interpretability and relevance of different document types to ICD-9 code predictions.

1.4 Ethical considerations

After careful consideration of the ethics checklist [3], we have determined that no ethical conflicts are applicable to this project. While clinical data is inherently sensitive, it is important to note that the MIMIC-III dataset has undergone a rigorous de-identification process, following the guidelines outlined by the Health Insurance Portability and Accountability Act (HIPAA). This de-identification process ensures that the dataset can be used for research purposes on an international scale [4].

While no conflicts were identified, automating ICD coding raises several ethical considerations. On one hand, ICD coding plays a crucial role in making clinical and billing decisions. Manual coding approaches are known to be error-prone and time-consuming [5]. Therefore, the endeavor to train machine learning systems for automatic ICD coding holds great potential for streamlining medical coding processes, alleviating the burden on hospital staff, and simplifying a task that can be highly error-prone when performed manually, mainly due to the length Electronic Health Records (EHRs) and the specificity of ICD codes [6].

However, it is crucial to acknowledge that machine learning systems for ICD coding are trained in a supervised manner using data collected from hospitals. This data is, therefore, subjected to biases derived from manual coding practices. Consequently, automated approaches run the risk of replicating these coding errors, which may include miscoding due to misunderstandings of abbreviations and synonyms, or overbilling due to unbundling errors, where more codes are entered than necessary. Furthermore, automated systems may also suffer from distribution shifts, potentially affecting their portability across various EHR systems in different hospitals [5]. To address these concerns, it is important to study the interpretability of automatic coding models and develop tools that enable human coders to understand and supervise the decisions made by ICD coding models.

Chapter 2

Background

2.1 Clinical Diagnosis Prediction

This section provides an introduction to the clinical diagnosis prediction problem. First, the ICD coding task is presented, which refers to the standardized classification of diagnosess and procedures based on a sequence of free-text reports from the hospital stay, and the challenges it poses. Second, it provides an overview of MIMIC-III, which is the dataset used throughout this project.

2.1.1 ICD-code prediction problem

ICD-9 codes are part of the International Classification of Diseases, which is a system developed by the World Health Organization (WHO) to classify diagnoses and procedures. These codes are used worldwide to understand the extent, causes, and consequences of diseases and mortality. In hospitals, they are used for health record-keeping, collecting disease statistics, and recording causes of death on death certificates. These codes also support payment and billing systems, hospital service planning, and health services research. They provide a standardized method for collecting diagnoses, allowing for large-scale data collection and analysis [1].

Many current works focus on predicting ICD codes from Electronic Health Records (EHR). EHRs contain clinical data describing the symptoms and treatments of patients during a hospital stay [7]. Traditionally, this task is carried out manually by the hospital's Medical Record Department, and it is very time-consuming and prone to errors [8]. It also requires extensive knowledge of medicine, coding rules and clinical terms. Therefore, research efforts have focused on developing a computational approach that can automatically assign ICD-9 codes to unstructured EHR [2].

Automating ICD code prediction is challenging due to several factors. First, the label space contains over 15,000 codes, making it a very high-dimensional problem. Second, EHR are an unstructured source of information consisting mostly of free-text notes. The clinical notes within EHR are often lengthy, and are filled with irrelevant information for ICD-9 code prediction. Moreover, clinical language is highly special-

ized, containing many non-standard abbreviations and a very distinct vocabulary to the general domain [2]. Furthermore, each physician has a particular style in writing and describing the symptoms and treatments, even for the same disease, as there are many ways to describe it [8].

2.1.2 MIMIC dataset

The Medical Information Mart for Intensive Care (MIMIC) is a large database containing multi-modal information related to patient stays at the Beth Israel Deaconess Medical Center in Boston, Massachusetts. This information ranges from free text notes, such as reports by nurses and healthcare providers regarding patient status, radiology and electrocardiogram reports, to vital sign information, and laboratory analysis results. Moreover, it contains details regarding the admission date, the length of stay, and the assigned ICD-9 code at the end of the stay [4]. The specific categories included in the dataset are depicted in Table 2.1.

Type	**Description**
Billing	coded data for billing and administrative purposes (includes ICD codes)
Descriptive	demographic, admission and discharge times
Dictionary	definitions for concepts, such as ICD codes
Interventions	procedures
Laboratory	blood and urine analysis, microbiology reports
Medications	administered medication
Physiologic	vital signals
Notes	free-text reports by nurses and doctors
Reports	free-text reports of electrocardiogram and imaging studies

Table 2.1: Description of types of information included in the MIMIC-III dataset [4]

In this project the focus will be on how to use free-text notes and reports to predict the final ICD-9 code. The ICD-9 codes will be used as labels to train our transformer models for classification. Moreover, since the focus is also on modeling the diagnosis temporally, we will make use of the timestamps in all the clinical notes, and introduce a new task on temporal prediction.

2.2 Transformers

This section provides background on transformer models. Firstly, it describes the components of the vanilla transformer architecture. Secondly, it describes the various usage modes of transformers depending on the parts of the architecture that are employed. Additionally, it provides an overview of the most commonly used pre-training tasks for encoder architectures, which will be the method employed throughout this project. Finally, it describes the sequence-length limitations of transformers that arise from the self-attention complexity and the solutions explored by current research.

2.2.1 Architecture

Transformers have revolutionized the field of Natural Language Processing and their success has even propagated to other domains such as computer vision or speech recognition [9]. This architecture arose to solve the machine translation (MT) or sequence-to-sequence problem, where the goal is to translate a sequence from a source domain to another target domain using an encoder-decoder fashion. The encoder produces a representation of the source sequence, and the decoder uses this representation to generate the target sequence one word at a time. Additionally, this architecture has achieved state-of-the-art results in many other tasks [9], becoming the go-to-model for most NLP problems.

The architecture of a vanilla transformer for sequence-to-sequence modeling is depicted in Fig. 2.1. It consists of an encoder and decoder, each of which is a stack of several identical blocks. The encoder block employs multi-head self-attention, position-wise feed-forward network and residual connections to ease the training of a deeper model. The decoder block follows a similar structure but it applies masked self-attention, as well as cross-attention to attend to the input sequence as well [9]. The elements of this architecture are described in detail in the following subsections.

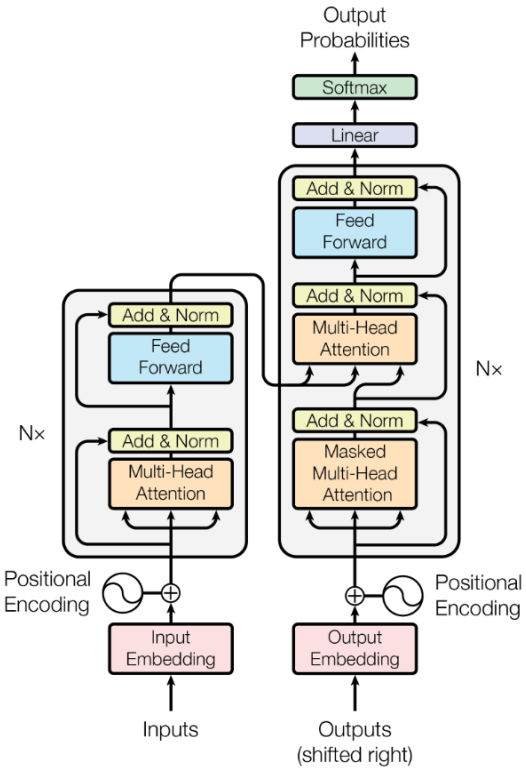


Figure 2.1: Transformer architecture for sequence-to-sequence tasks. Figure extracted from Vaswani et al. [10]

Attention Modules

The attention mechanism employed in the original work [10] is the so-called *Scaled Dot-Product Attention*. This module involves a set of queries, keys, and values. The queries represent the element that requires encoding or decoding, whereas the keys and values represent the references and contents that the model is searching for. Specifically, the keys are employed to compute the similarity or relevance to the query, while the values are used to obtain the final representation.

Formally, the attention module is defined by the following equations [10] :

$$\text{Attention}(Q, K, V) = \text{SoftMax}(\frac{QK^T}{\sqrt{D_k}})V = AV \tag{2.1}$$

Here, $Q \in \mathbb{R}^{N \times D_k}$ denotes the query matrix, $K \in \mathbb{R}^{M \times D_k}$ represents the key matrix, $V \in \mathbb{R}^{M \times D_v}$ is the value matrix, and N, M and D_m denote the target sequence length, source sequence length and model internal dimensionality, respectively. The matrix $A \in \mathbb{R}^{NxM}$ is the attention weight matrix, which is obtained through the dot product between the key and query embeddings, followed by a (scaled) softmax layer. The underlying idea is that elements in the source sequence that are relevant to encode the current element in the target sequence will have a high dot-product similarity score. The final output is obtained by multiplying with the values matrix, resulting in a set of embeddings for each element in the target sequence $Attention(Q, K, V) \in \mathbb{R}^{N \times D_v}$, which have been obtained attending to all elements in the source sequence.

In practice, the transformer employs multi-head attention. In this approach, the queries, keys and values are projected to a lower dimensional space, i.e., $Q_i = QW_i^Q \in \mathbb{R}^{D_k/H}$ and $V_i = VW_i^V \in \mathbb{R}^{D_v/H}$, where H are the number of heads. Subsequently, single-head attention is applied, and the outputs are then concatenated to go back to the original dimensionality of D_k and D_v. Finally, a linear layer is applied to the concatenated embeddings to combine information across attention heads. The equation for multi-head attention is the following[10]:

$$\text{MultiHeadAttention}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_H)W^o \tag{2.2}$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.3}$$

The purpose of employing multiple attention heads is to allow the model to attend to different elements according to various characteristics. For instance, in the context of language translation, some attention heads might focus on identifying the specific word to be translated in the source sequence, whereas others might attend to adjectives or adverbs that influence the final decoded word.

Layer Norm and Residual Connections

In practice, the transformer incorporates many layers of encoder and decoder blocks. To facilitate the training of such deep networks, layer normalization and residual

connections are employed. Layer normalization is similar to batch normalization but it is applied to the hidden units within a particular layer, rather than across the samples of a mini-batch. More specifically, layer normalization attempts to "fix" the mean and variance of the inputs to each layer by normalizing across the hidden dimension. This is expected to regularize the model through covariate shift reduction [11]. On the other hand, residual connections establish connections between each layer and its preceding layer. These connections facilitate the addition of layers to the model in a more regularized manner, as the upper layers undertake the task of refining the output generated by the lower layers [12].

Position-wise Feed Forward Network

Finally, after the layer norm and the residual connection is applied, the output is run through a position-wise feed forward network. This is similar to a standard fully connected network but it is applied to each position in the sequence "separately and identically" [10]. The idea is to apply a non-linear transformation in an element-wise manner to improve generalization capabilities.

2.2.2 Usage methods

The transformer architecture is verstatile and can be employed in three distinct modes [9].

Encoder

The encoder part of the transformer architecture can be used to obtain a sequence representation, which can be used for many different downstream tasks such as sequence classification or question answering. The encoder processes all the tokens of the sequence in parallel, and generates an embedding for each token by attending to all tokens in the sequence. Encoders employ self-attention, which means that the queries, keys and values all come from the same input sequence. In some cases, an additional [CLS] token is added to the beginning of the sequence, and its embedding can be used as a representation of the entire sequence [9].

Decoder

The decoder part of the transformer architecture can be employed as a language model to generate or complete a sequence. In this case, the transformer employs masked self-attention. The masking process implies that for each element in the sequence, the resulting embedding is generated by attending only to previous elements of the same sequence. The final representation can be used to predict the probabilities corresponding to the following word, enabling the completion or generation of text [9].

Encoder-Decoder

The complete encoder-decoder architecture of transformers can be used to solve sequence-to-sequence tasks. Initially, the input sequence is encoded using self-attention. Then, the target sequence is generated through a decoder which can attend to previously decoded tokens using masked self-attention, as well as all the tokens in the input sequence using cross-attention. In cross-attention, queries sourced from the target sequence, while the keys and values are derived from the source sequence [9]. This usage mode is suitable for applications related to sequence-to-sequence problems, such as machine translation, or even in cross-modal domains for image captioning [13].

2.2.3 Unsupervised pre-training

Throughout this project, transformer models will be used to encode text sequences for medical diagnosis prediction. These models are transformer encoders, which are trained to generate token and sentence-level representations that can be applied to many downstream tasks, such as question answering or sentence classification. Using a pre-trained language model such as BERT or RoBERTa presents a notable advantage: these models have undergone extensive unsupervised pre-training, enabling them to develop a deep understanding of language and achieve state-of-the-art performance in numerous downstream tasks through fine-tuning [14].

Unsupervised learning refers to the methodology of training machine learning models using unlabeled data. In the domain of Natural Language Processing, it is common to define unsupervised tasks based on the inherent structure of text. For instance, language models can be trained through next-word prediction. Similarly, transformer encoders such as BERT are pre-trained using two unsupervised tasks: Masked Language Modeling and Next Sentence Prediction [14].

Masked Language Modeling (MLM) consists in predicting masked tokens of a sentence. This is achieved by selectively masking out a portion of tokens in the sentence and using the embeddings generated by the transformer encoder to predict those masked tokens. However, a potential drawback of this approach is that the model might only learn good representations for the [MASK] token. To mitigate this issue, the original word is only masked 80% of the time, while 10% of the time the original word is kept, and the remaining 10% of the time it is replaced by a random word. On the other hand, Next Sentence Prediction (NSP) involves predicting whether two sentences sequentially follow each other in the text. The idea is that many downstream tasks, such as question answering, involve the need to understand the relationship between sentences. However, some frameworks, such as RoBERTa [15] do not incorporate this pre-training task.

2.2.4 Adapting to long text sequences

Standard transformer models encounter limitations regarding input sequence length due to the quadratic complexity of self-attention. The current research landscape is actively exploring preprocessing methods and adapted architectures to handle extended texts for various tasks, including text generation, summarization, and, particularly pertinent to this project, text classification [16].

Pre-processing approaches

Standard transformer models can be adapted for long text classification through the application of appropriate pre-processing techniques. These pre-processing approaches can be categorized into three main groups:

- **Truncation**. This basic approach involves limiting the text to the maximum input length of transformers, typically 512 tokens for models like BERT or RoBERTa. However, this method often discards crucial information dispersed throughout the text [16].
- **Chunking**. Chunking involves dividing the long text into smaller snippets and processing each chunk with a pre-trained language model to capture local semantics. Afterward, a representation for each chunk, such as the [CLS]-token embedding, can be extracted, and document-wise representations can be derived by aggregating individual chunk representations using methods like max-pooling or mean-pooling. While effective for local context, this approach sacrifices the ability to detect long-range dependencies [16].
- **Selection**. Another approach focuses on developing techniques to identify relevant or "salient" information within lengthy documents [16]. This approach operates on the assumption that crucial data for document classification is condensed within a shorter text segment, and the goal is to learn to identify this vital information for predictions [17].

Adapted architectures

Research efforts have also been directed towards adapting transformer architectures to handle longer text inputs, falling into three main categories [16]:

- **Efficient transformers**. These models aim to reduce the self-attention complexity to allow the processing of a longer sequence by employing various attention patterns. Some of the most popular models include the LongFormer [18], which uses a local attention pattern with dilations to increase receptive field, or BigBird [19], which employs a graph-based attention mechanism and reduces the complexity with sparsification algorithms. Moreover, some works also propose to learn attention patterns, such as the Route Transformer [20], which applies a K-means algorithm to learn token clusters that should be connected through attention.

- **Recurrent transformers**. The Transformer-XL model [21] introduces a recurrent mechanism to the standard transformer architecture. In this approach, the lengthy document is divided into snippets, and each of these snippets undergoes recurrent processing by the transformer. More precisely, after processing each chunk, the resulting hidden states are cached and then concatenated with the subsequent chunk before being fed into transformer. This mechanism enables the inclusion of previous context, allowing the model to maintain attention to the entire sequence in a recurrent fashion [21].

- **Hierarchical transformers**. Finally, some architectures involve chunking and the inclusion of a second hierarchical transformer model. First, individual text snippets are processed using a pre-trained language model. Subsequently, representations of these chunks are extracted, usually in the form of the [CLS]-token embedding. Then, a second transformer is employed to operate on these chunk representations and capture long-range dependencies within the text [22]. This approach has proven successful in ICD-9 code prediction, as detailed in Section 2.3.4.

2.3 Models For ICD-Code Prediction

This section provides an overview of the state-of-the-art models for ICD-9 code prediction. Firstly, the concept of multi-label attention is introduced, which is a pivotal architectural choice common among all the models in the literature. Secondly, the convolutional and recurrent architectures are described. Then, the first attempts to apply transformer-based models are explained, along with the limitations that were encountered to surpass the performance of previous recurrent approaches. Finally, the last subsection describes the document-level methods, which are a solution to adapt transformer models for long sequences using a hierarchical architecture.

2.3.1 Label-wise attention network

Prior to diving into the state-of-the-art models for ICD-9 code prediction, it is important to explain a pivotal architectural feature shared among all these models: the Label-Wise Attention Network (LWAN). It was first introduced by the CAML model [2], which is a convolutional approach, and it was subsequently adopted by later RNN and transformer-based models.

Label-wise attention is used in multi-label sequence classification models to facilitate the generation of different document representations for each label while maintaining a common model backbone. In binary sequence classification, aggregation methods like max-pooling or mean-pooling are usually sufficient to obtain a single document representation to predict the binary label probabilities. However, these methods are not effective for multi-label classification because they cannot accurately predict multiple labels. In multi-label classification, each label needs its own

document representation. Label-Wise Attention allows us to create separate document embeddings for each label by focusing on the relevant tokens. For instance, in medical diagnosis, certain diseases are more strongly linked to specific clinical terms, so their token embeddings are given more weight.

The LWAN is appended right before the classification head to obtain label-wise document representations. Formally, the input to this network is a sequential document representation in the form of a matrix $H \in \mathbb{R}^{d_c \times N}$, where N is typically the number of elements in the sequence, and d_c is the dimensionality of the model. Then for each label l, a (learnable) embedding $u_l \in \mathbb{R}^{d_c}$ is defined, which acts as a query vector. Then the dot product between each label embedding and each token embedding is calculated and passed through a softmax to obtain the normalized attention weights:

$$\alpha_l = \text{SoftMax}(H^T u_l) \tag{2.4}$$

Afterwards, the document representation for label l can be obtained through the attention-weighted average of token embeddings:

$$v_l = \sum_{n=1}^{N} \alpha_{l,n} h_n \tag{2.5}$$

This can be interpreted as a form of cross-attention, where labels act as queries and the token embeddings act as keys and values. LWAN also has the benefit of providing an explainability framework for the assigned diagnosis, because it automatically highlights the portions of the text, i.e. the tokens, that can explain the medical diagnosis [2].

2.3.2 CNN and LSTM-based methods

Before the rise of transformer-based models, clinical diagnosis from unstructured text mainly relied on convolutional neural networks (CNNs) and long short-term memory networks (LSTMs). These models utilized pre-trained word embeddings, typically from word2vec, and were trained from scratch for the task of predicting medical diagnoses with a convolutional or recurrent architecture. Despite their relative simplicity, some of these models achieved high-performance benchmarks, as illustrated in Figure 2.2.

One early successful CNN-based model for ICD-9 code prediction was the Convolutional Attention for Multi-label Classification (CAML) model [2]. CAML employs pre-trained word2vec embeddings and combines neighboring word representations using convolutional filters (Fig. 2.3). It further utilizes label-wise attention (LWAN) to acquire label-specific document representations, as explained in Section 2.3.1. An extension of this model, CAML-DR, incorporates description regularization to enhance the classification of rare medical codes.

Another advancement came with the MultiResCNN model [23], which built upon CAML by employing multiple convolutional filters of varying sizes and introduced

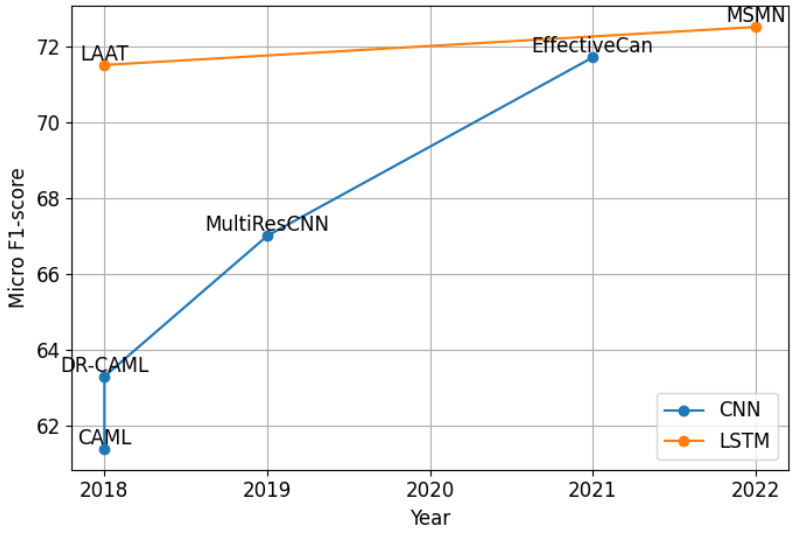


Figure 2.2: Performance of CNN and LSTM-based models for ICD-9 code prediction on top 50 labels over the years.

residual connections for deep network training. This approach allows the model to capture different text patterns of various lengths, similar to n-grams, and expanded its receptive field. Finally, the posterior work of the Effective Convolutional Attention Network (EffectiveCAN) model [24] introduced multi-layer attention, enabling label-specific multi-scale attention representations.

In contrast to the CNN-based approaches, the Label Attention LSTM (LAAT) model [25] proposed to employ a bidirectional LSTM to model the document sequence, followed by a label-wise attention network to obtain label-wise document representations for multi-label classification. Further enhancements were achieved with the Multi-Synonym Matching Network (MSMN) [26] . This approach expands code descriptions into multiple code synonyms and uses LSTM networks to obtain synonym representations. Then it computes attention between each synonym representation and different document representation heads. MSMN set a robust baseline against which transformer-based approaches are later compared, notably improving F1-score through multi-synonym attention.

2.3.3 Transformer-based methods

Transformers can be directly applied to the ICD-9 code task by truncating the input text to the appropriate number of tokens, typically 512 for BERT and RoBERTa-based models. Various studies [27] have explored different variants of these models, obtaining poor results, as depicted in Figure 2.4. Other investigations have attempted to leverage transformers specifically designed for long input texts, such as the Longformer [18], obtaining substantially better results (+10.8 F1-score), but still not surpassing state-of-the-art performance set by LSTM-models.

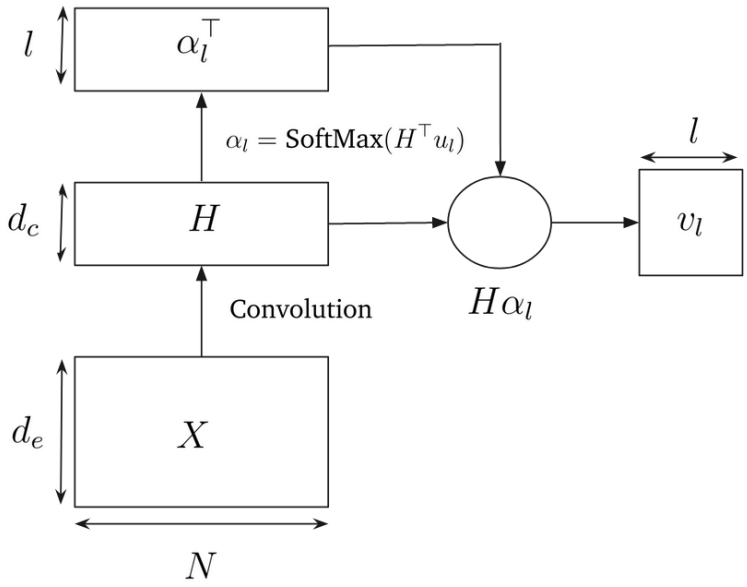


Figure 2.3: CAML convolutional architecture for ICD-9 code prediction employing label-wise attention (adapted from [2]).

More specifically, Ji et al [27] explored the use of different BERT-variants for ICD-9 coding. These variants differ in the domain on which they were pre-trained. General-domain models, such as the BERT-based model, are pre-trained on datasets that are not specialized in any field, such as BookCorpus or Wikipedia. On the other hand, mixed-domain models are pre-trained from scratch or with continual pre-training on datasets that are closely related to the clinical field, such as the biomedical domain. For instance, BlueBERT is trained from scratch using PubMed texts and MIMIC-III clinical notes, while BioBERT undergoes continual pre-training on PubMed abstracts and full-text articles. PubMed-BERT, on the other hand, is pre-trained from scratch using MedPub publications. Finally, in-domain models, such as ClinicalBERT, undergo continual pre-training on the MIMIC-III dataset, which is used for the downstream task of ICD-9 code prediction.

Figure 2.4 illustrates the performance of these BERT-type models on the ICD-9 coding task [27]. Within the standard transformer models, PubMedBERT exhibits the highest performance, followed by ClinicalBERT. Pre-training in the mixed-domain or in-domain context significantly enhances performance compared to using the BERT-base model (+5.7 Micro F1-score). Nevertheless, RoBERTa achieves a similar performance to PubMedBERT without any additional pre-training.

Furthermore, Dai et al [22] explored the use of long-document transformers [22]. In particular, the LongFormer [18] employs a local sliding window-based attention where each token attends to a window of neighbouring tokens, except for the [CLS]-token which retains global attention. The resulting model can therefore process se-

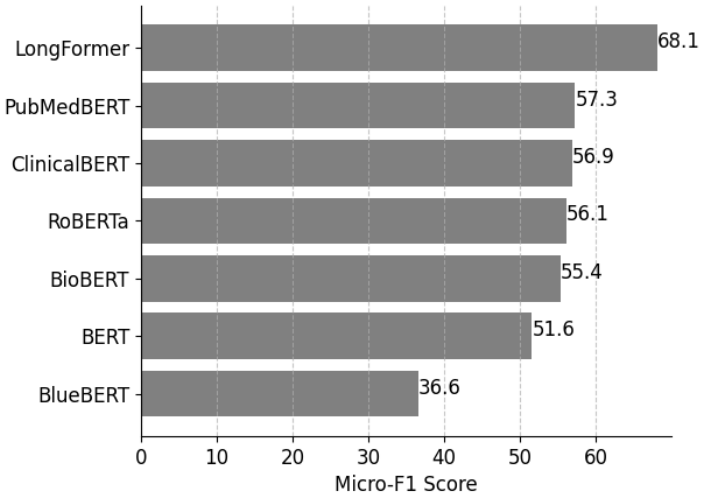


Figure 2.4: Micro-F1 score of transformer-based models on the ICD-9 coding task (top 50-labels of the MIMIC-III dataset). The graph shows the experiments done using BERT variants and the RoBERTa-base model, which process up to 512 tokens, as well as the LongFormer model, which processes up to 4018 tokens. The performance of BERT variants is extracted from the experiments run by Ji et al [27], and the performance of RoBERTa and LongFormer is extracted from the work by Dai et al [22]

quences of up to 4094 tokens. The performance of the LongFormer on the ICD-9 coding task without any further pre-training is significantly better (+10.8 F1-score) than PubMedBERT, as shown in Fig. 2.4, which highlights the importance of access to a larger portion of clinical notes from patients to make accurate diagnosis. This underscores the necessity for the development of document-level methods capable of processing the complete sequence of clinical reports, as elaborated upon in the subsequent section.

2.3.4 Document-level methods

Efforts to apply pre-trained transformers without further modifications to the ICD-9 coding problem were unsuccessful [27] [22], as explained in Section 2.3.3. The discharge summaries contain 3,594 tokens on average, while the combined set of notes contains an average of 21,916 tokens per patient stay [28]. Crucial information to predict patient diagnoses is likely to be dispersed throughout these notes, thus models that truncate the text at the 512-token limit risk overlooking a significant portion of relevant data.

To address this limitation, document-level methods have been proposed, which can handle longer sequences by splitting them into multiple documents. These document-level models have a hierarchical architecture, as depicted in Fig. 2.6. First, the document is split into chunks employing a pre-defined criteria. The first layer of the

model is a pre-trained transformer encoder which runs over each individual chunk. After this step, some models keep all the token embeddings for the subsequent layer, while others use the [CLS]-token embedding as a chunk representation. Afterwards a second model runs over all the chunk representations to extract global features. Finally a label-attention module is used to aggregate all chunk embeddings and obtain the probabilities for each of the 50-top labels of the MIMIC-III dataset. The document-level architecture is depicted in Fig. 2.6.

	Document Splitting		
	Chunk repre.	**Chunk length**	**Chunk Splitting**
PubMedBERT-hier [27]	[CLS]	512-token	Equal.
TrLDC [22]	[CLS]	128-token	Equal + Overlap.
PAAT [29]	All tokens	max. 4096-token	Structure-based
HiLAT [30]	All tokens	512-token	Equal
HTDC [28]	All tokens	512-token	Equal

Table 2.2: Comparison of Document-level models for ICD-9 code prediction in terms of the chunk length, splitting method and representation.

	First Layer	**Second Layer**	**Label-Attention**
PubMedBERT-hier [27]	PubMedBERT	Transformer	Standard
TrLDC [22]	RoBERTA (L*)	Transformer	Standard
PAAT [29]	Clinical Long-Former	Bi-LSTM	Partition
HiLAT [30]	ClinicalPlusXLNet	None	Token+Chunk-level
HTDC [28]	Roberta-PM	Transformer	Standard

Table 2.3: Comparison of Document-level models for ICD-9 code prediction in terms of the base pre-trained transformer, architecture of the second layer and type of label attention.

As seen in Figure 2.5, the key factor distinguishing methods that surpass the performance of LSTM-based models is the chunk representation choice. For this reason, in the following subsections these models are categorized into two main groups: 1) models that utilize the [CLS] token of each segment, and 2) models that consider all tokens within the aggregation process. A comparison of the various model architectures is summarized in Tables 2.3 and 2.2, and each model is explained in more detail in the following subsections.

[CLS]-token models

The first category of models adopts the embedding of the [CLS] token as a representation of each chunk, which is subsequently inputted into a second transformer layer.

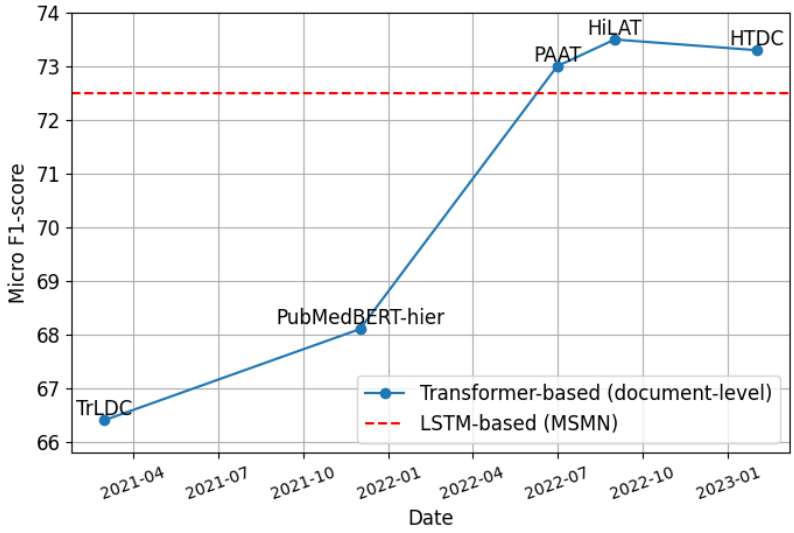


Figure 2.5: Performance of document-level methods (transformer-based) over the years on the ICD-9 code prediction task (50 top labels only). The figure also shows the LSTM-baseline set by MSMN [26]

TrLDC. The TrLDC (Transformer-based model for Long Document Classification) model [22] was the first to explore this hierarchical structure and achieved a micro-F1 score of 66.4 for the top 50 ICD-9 labels. The study explores different strategies for segmenting the document and concludes that overlapping segments of 128 tokens yield optimal results for the MIMIC-III dataset. This idea is that document segmentation may introduce context fragmentation, and retaining 25% of tokens from contiguous segments helps alleviate this issue. Additionally, the study finds that dividing contiguous segments into equal sizes is preferable to splitting it according to the document structure. Lastly, the final model employs a pre-trained transformer, namely RoBERTA, which is trained from scratch on biomedical articles and clinical notes [15].

PubMedBERT-hier. The PubMedBERT-hier model [27] also has a hierarchical structure using the [CLS]-token embedding similar to TrLDC [22]. The model performance is slightly higher (68.1 Micro-F1 score) than TrLDC. The main difference is that the documents are split into 512-token segments, and the base transformer is the PubMedBERT model [31], which is pre-trained from scratch on the biomedical domain using PubMed publications. The model also adds a Label-Wise Attention Network atop the second transformer to produce one final label-specific document embedding, which is finally used to obtain the label probabilities.

Neither the TrLDC [22] nor the PubMedBERT-hier [27] models are able to surpass the performance of CNN and LSTM-based approaches. This was quite surprising, given the recent success of transformers in a wide range of tasks, and that CNN and LSTM-based models are much smaller and do not employ any sort of large-scale pre-training. This prompted further research on the topic of applying transformers to

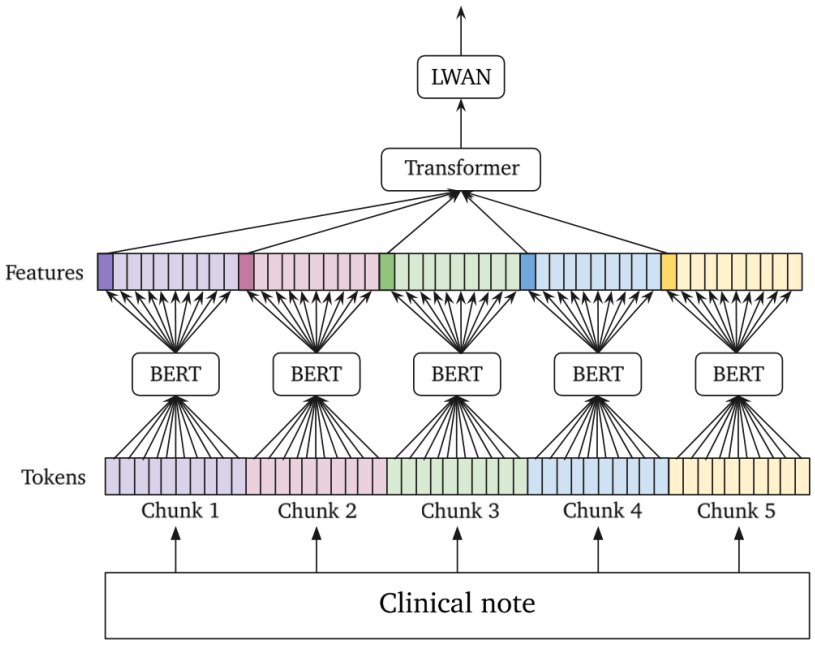


Figure 2.6: Example of a hierarchical model architecture. The figure is adapted from the PubMedBERT-hier model architecture [27].

the ICD-9 coding problem.

All-token models

Applying transformers to the ICD-9 coding task was initially unsuccessful. Models that simply truncated the input sequence were bound to miss a large portion of relevant data, and models that followed a hierarchical structure and used the [CLS]-token embedding as a chunk representation do not have enough capacity to accurately make predictions for all the labels. For this reason, subsequent models focused on adapting the architecture to allow for all tokens to be taken into account for the final document embedding.

PAAT. The PAAT (**PA**rtition **AT**tention) model [29] achieves a Micro-F1 score of 73.0 on the top 50 labels, surpassing the performance of the best LSTM-based model (MSMN [26]). The model uses the Clinical Long-Former as a first-layer transformer, which has a maximum input of length 4096-tokens. Afterwards, a bi-LSTM runs over the full sequence of token embeddings to obtain global token-embeddings which can attend to all tokens in the different segments. Finally a partition-based label attention network produces the output probabilities. The partition-based LWAN operates on individual chunks or partitions rather than the full token sequence. This ensures selection of the most relevant tokens within each chunk, resulting in a +0.8 F1-score improvement.

HiLAT. The HiLAT (**Hi**erarchical **L**abel-**AT**tention) model [30] achieves state-of-the-

art performance of 73.5 Micro-F1 score on the top 50-labels. The base pre-trained transformer is the ClinicalPlusXLNet, which is obtained by further pre-training XLNet [32] on the MIMIC-III dataset. After running the ClinicalPlus XLNet over individual segments of 512 tokens each, the chunk representation is obtained through a token-level attention mechanism. Afterwards, the chunks are combined through a simple chunk-level attention layer, without the need to add any other transformer or LSTM model on top.

Even though HiLAT achieves state-of-the-art performance, the training speed is four times slower than RoBERTa variants because ClinicalPlusXLNet needs to compute all possible sequence permutations to capture bidirectional context [30]. To speed up training, it is also possible to freeze the ClinicalPlusXLNet parameters and only learn the attention parameters. In this case, the performance decreases from 73.5 Micro-F1 to 73.2 Micro-F1 [29], which is still very close to state-of-the-art. It should be noted that the superior performance of HiLAT should be attributed to the development of ClinicalPlusXLNet, rather than an improved architecture for ICD-9 code prediction. This is corroborated by the fact that when used in combination with other BERT models, such as PubMedBert, it achieves a much lower performance (-9.7 Micro-F1).

HTDC. The HTDC (**H**ierarchical **T**ransformer for **D**ocument Classification) [28] also achieves close to state-of-the-art performance of 73.3 Micro-F1 score. The architecture of this model is similar to PubMedBERT-hier but for the choice of pre-trained base transformer, which is a RoBERTa model pre-trained on PubMed and MIMIC-III. The second layer is also a transformer, but it runs over the sequence of all tokens instead of just the [CLS] embeddings. The ablation studies show that this is a key design choice, since it increases the performance by 2.3 Micro F1-score.

Furthermore, the HTDC model is the first to take into account additional notes from the Electronic Health Record, without relying solely on the discharge summary. For this reason, this model will serve as a starting point for this project.

2.4 Research on the full EHR Sequence

This project aims to explore a unique task focused on predicting ICD-9 codes throughout a hospital stay, differing from the approaches discussed in Section 2.3. Unlike those models that solely utilize the discharge summary, our task involves the complete Electronic Health Record (EHR) sequence. As most existing ICD-9 code prediction models do not leverage the entire EHR sequence, we delve into other relevant works addressing diverse tasks that exploit the complete EHR sequence. This exploration serves as a source of inspiration and also sheds light on how to handle the challenges posed by the extended EHR data.

2.4.1 Readmission prediction

A foundational work for our evaluation framework is ClinicalBERT [33]. This work explores a novel task of predicting readmission probability throughout a patient's stay, which is similar to our proposed task of predicting ICD-9 codes at any temporal point during a hospital stay.

The ClinicalBERT model, a variant of BERT, undergoes pre-training on clinical notes from the MIMIC-III dataset using standard masked language modeling and next sentence prediction. However, the distinctive aspect lies in its fine-tuning process, which involves the task of predicting readmission probability at various temporal points of the hospital stay. This task involves forecasting readmission probabilities throughout a patient's stay based on the available notes up to that point, and it therefore employs the full EHR sequence as input data.

There are two relevant insights from the approach in ClinicalBERT to tackle the readmission probability task, that are applicable to our task of interest. First of all, the model adopts a segmentation strategy to make predictions based on an extended amount of text. Each segment is processed individually to calculate the readmission probability. Afterwards, the predictions are combined using a weighted average approach. Secondly, ClinicalBERT introduces a method of reporting performance metrics in a temporal manner. Since the model predicts readmission probabilities at every point in time, it needs to define a strategy to report on its predictive power. The strategy consists in defining specific timeframes— such as predicting readmissions between 24 to 48 hours after admission and 48 to 72 hours after admission— that are used to report aggregated metrics. Our project will adopt a similar temporal evaluation approach, adjusting the cut-offs to our dataset's characteristics.

2.4.2 Discharge summary generation

Also related to our work is the task of generating discharge summaries. Recent research [34] [35] [36] [37] has focused on automating the summarization of clinical notes, as this leads to a significant reduction in the workload for medical staff. The task of generating discharge summaries also requires engagement with the complete EHR sequence, and it represents a separate research direction from our primary focus, which is related to early ICD-9 code prediction.

In particular, K. Pal et al [34] explore the generation of discharge summaries based on previous clinical notes. The complete EHR sequence far exceeds the input length limitations of abstractive summarization models, such as BART, T5, and Longformer. For this reason, they investigate various configuration set-ups to shorten the length of the EHR sequence. Specifically, they define multiple dataset configurations, such as selecting only the nursing notes, combining the earliest and latest nursing notes, or extracting the first three lines from each section within the latest note. In this project, we will follow a similar approach and evaluate the model's performance using different dataset configurations tailored to the characteristics of our task.

Chapter 3

Experimental set-up

This chapter outlines the experimental set-up of the project, detailing steps such as dataset access, pre-processing, exploratory data analysis, and task specification with its evaluation framework. First, the process of obtaining access to the MIMIC-III dataset is explained. The preprocessing pipeline for creating the project's dataset is then presented, involving a compilation of free-text clinical note sequences labeled with their corresponding ICD-9 codes. The subsequent section dives into an exploratory analysis of the complete EHR sequence. This involves both a qualitative comparison of different note types within the EHR sequence and a quantitative breakdown of clinical note distribution by document category and temporal aspects of hospitalization. These insights lead to the proposal of a novel task focusing on the temporal prediction of ICD-9 codes during a patient's hospital stay. The task's specifics, including time cut-offs for aggregated metrics and an evaluation framework, are detailed in the final section.

3.1 Dataset Access

The MIMIC-III dataset [4] contains information regarding clinical care of patients in hospitals, therefore its access is restricted and users must ensure that the data is handled in a respectful manner. Researchers wishing to access this dataset are required to complete two steps:

1. Complete a course in Health Insurance Portability and Accountability Act (HIPAA) that instructs researchers on how to conduct studies with human participants protecting their rights and identity.

2. Sign a data usage agreement highlighting the requirement for appropriate data handling and prohibiting any task aiming to re-identify individuals.

Both steps were completed and dataset access was granted.

3.2 Dataset Preprocessing

The MIMIC-III dataset contains a collection of CSV files. The tables used for this project are:

1. PROCEDURES_ICD: table mapping each hospital admission identifier to each procedure ICD-9 code.
2. DIAGNOSES_ICD: table mapping each hospital admission identifier to each diagnosis ICD-9 code.
3. NOTEEVENTS: table containing free-text notes from hospital admissions mapped to its corresponding hospital admission id. Notes are also timestamped. The types of notes in this table are: nursing, radiology, physician, electrocardiogram, respiratory, echography, nutrition, general, rehab services, social work, case management, consult, pharmacy and discharge summary.

The preprocessing pipeline starts by reading the PROCEDURES_ICD and DIAGNOSES_ICD Tables. The codes in both tables are not in the right format, since they do not have any periods, so this might lead to issues if they are combined directly. For this reason, the periods need to be added back in the right place: for procedures, dots after the first 2 digits, and for diagnosis, after the first 3 digits. Then both tables are concatenated and the code column is renamed to an "absolute code" column.

The label space is vast, so only the top 50 labels are used for this project. The train, validation and test splits used for the top 50 labels are extracted from the CAML model [2]. The splits table is merged with the dataframe of absolute codes, obtaining a new table mapping each hospital admission identifier to its corresponding split and absolute code.

Finally, the NOTEEVENTS table is loaded, and merged with the previously obtained table of splits and absolute codes. The resulting table is filtered so that only the samples corresponding to the top 50 labels are retained. The NOTEEVENTS table contains one row per hospital admissions id and note. All the notes need to be merged into a single document, therefore the dataframe is aggregated on hospital admission identifier, concatenating all notes from the same stay into a list of notes. The labels are also aggregated into a multi-hot vector for multi-label classification. The resulting table that will be used to train and evaluate the model contains the following columns:

- HADM_ID: hospital admission identifier
- TEXT: list of free-text notes
- CATEGORY: list of note categories
- CHARTTIME: list of note timestamps
- SPLIT_50: split label (train / dev / test)
- LABEL: multi-hot vector of top-50 labels

3.3 Exploratory Analysis

This section presents an exploratory data analysis of the complete EHR sequence. This includes a motivating qualitative discussion comparing the discharge summary, which is commonly used by current ICD-9 models, with other types of notes present in the complete EHR sequence. Additionally, this section presents a quantitative analysis of the different document types in the EHR sequence, as well as the variation in the volume of notes generated throughout a patient's hospital stay. Overall, this exploratory analysis reveals the untapped potential of information present within the complete EHR sequence, which could be valuable to create real-time disease prediction systems. This is key to motivate the formulation of a novel task concerning the temporal prediction of ICD-9 codes throughout a hospital stay, described in the subsequent section.

3.3.1 Qualitative discussion

Current models for ICD-9 code prediction, such as the ones described in Section 2.3, can only be used after the patient has left the hospital and the corresponding discharge summary has been generated. This constraint hinders the real-time availability of patient diagnosis and procedure predictions. To address this limitation, this project aims to study the development of a model that can make temporal ICD-9 code predictions throughout a patient's hospital stay using the clinical notes available at that point in time, without explicitly looking at the discharge summary. Developing a model that can make live predictions of diagnosis and procedures would allow hospitals to keep a live statistic of diagnosis and procedures, and plan the resources optimally.

Throughout a patient's hospitalization, numerous clinical notes are generated. Among these, the discharge summary holds particular significance as it provides a concise compilation of the patient's medical history, hospital stay details, discharge medications, and diagnoses. As a result, the discharge summary directly relates to the ICD-9 code prediction task and serves as a valuable resource for model development. However, the process of generating this discharge summary is time-consuming and is typically performed only at the very end of the patient's stay. On the other hand, there are many other types of notes that are generated during the hospital stay and could be used to predict the diagnosis and procedure codes before the patient is discharged. However, these notes offer more extensive information but may lack the focused relevance found in discharge summaries.

A qualitative comparison between a radiology note and a discharge summary note is shown in Figures 3.1a and 3.1b. In the discharge summary, numerous ICD-9 code labels are directly listed under the "Discharge Diagnosis" section. On the other hand, the radiology note provides a less concise description of a specific procedure. For instance, the procedure known as PTCA (Percutaneous transluminal coronary angioplasty) can be inferred from the radiology note's description, although it is not

```
Admission Date:  [**2186-11-21**] Discharge Date:  [**2186-11-28**]

Service:  CARDIOTHORACIC

(...)  Major Surgical or Invasive Procedure:
[**2186-11-21**] - AVR 25mm St.  [**Male First Name (un) 923**] Porcine
Valve

History of Present Illness:
Splendid 80 year old gentleman who has severe aortic stenosis.  He sustained
and inferior wall MI on [**2-/2178**] followed by a PTCA. He recently had
a colonic tumor/polyp removed in [**Month (only) 216**] and developed chest
pain as well as dyspnea perioperatively.  A cardiac catheterization was
performed which revealed sever aortic stenosis and no flow limiting disease.
His ejection fraction was preserved.

Discharge Diagnosis:
Aortic Stenosis, Hyperlipidemia, PAF, HTN, COPD, IMI, PTCA [**2178**],
Postop confusion, Left TKR, Colonic polyp removal CRI, Depression, (...)
```

(a) Discharge summary extract from a random patient in the MIMIC-III dataset.

```
[**2186-11-21**] 11:38 AM
CHEST PORT. LINE PLACEMENT Clip [**Clip Number (Radiology) 62117**] Reason:
postop film
Admitting Diagnosis:  AORTIC STENOSIS/ AORTIC VALVE REPLACEMENT/SDA (...)
_____________________________________________________________________
FINAL REPORT
HISTORY: 80-year-old man status post AVR. Please assess position of tubes
and lines.

TECHNIQUE: A single AP supine porTable chest radiograph was obtained (...)
FINDINGS: There is an endotracheal tube in appropriate position with the
tip approximately 3.5 cm above the carina.  There is a right IJ Swan-Ganz
catheter with tip within the main pulmonary artery.  (...)  There has
been interval sternotomy and aortic valve replacement.  There is normal
postoperative appearance of the mediastinum with associated cardiomegaly.
The lungs are grossly clear.  There may be a small left pleural effusion.

IMPRESSION: (...)  Normal postoperative appearance of the heart and
mediastinum status post sternotomy and AVR.
```

(b) Radiology note extract from a random patient in the MIMIC-III dataset.

explicitly mentioned, unlike in the discharge summary.

3.3.2 EHR note distribution

Limited research has explored the complete sequence of clinical notes of a hospital stay, with most research being focused on the discharge summary. However, the complete EHR sequence is on average around 30,000 tokens long, and only 13 %

Documents	Length (# chunks)					
	mean	q1	q2	q3	min	max
Discharge Summary	6.8	4.0	6.0	8.0	31	1.0
EHR excl. Discharge Summary	49.0	10.0	19.0	47.0	2949.0	0.0
EHR (all notes)	55.8	16.0	27.0	55.0	2967.0	1.0

Table 3.1: Summary of statistics (average, 25th, 50th and 75th percentile, minimum and maximum) of the number of chunks (512-token segments) in the discharge summary compared to the complete EHR sequence of documents with and without excluding the discharge summary.

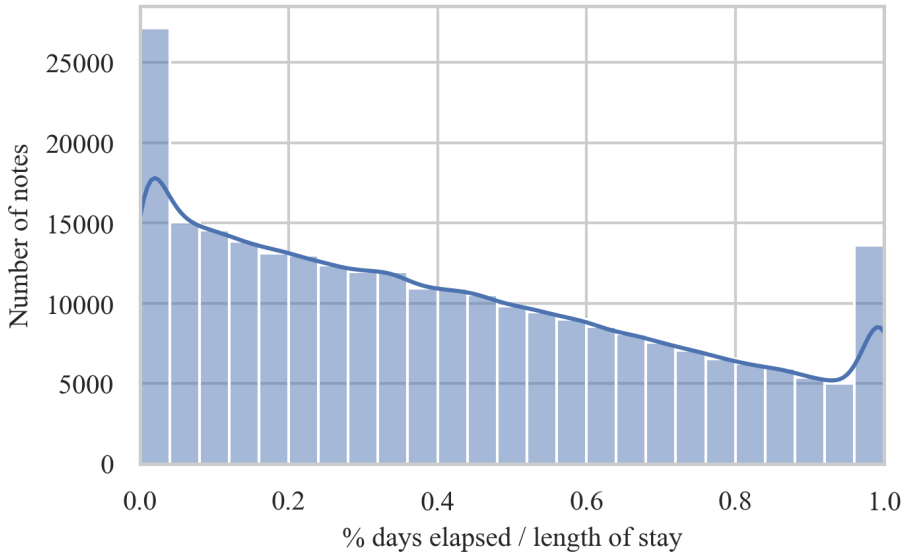


Figure 3.2: Histogram of the distribution of the number of clinical notes against time elapsed during the hospital stay.

of this corresponds to the discharge summary, as shown in Table 3.1. This indicates that there is a large portion of clinical notes that could be useful to predict ICD-9 codes at any point during the hospital stay.

Figure 3.2 provides insights into the volume of notes present at various percentages of days elapsed during the total length of stay. Notably, there is an initial peak of notes at the beginning of the stay, followed by a gradual decrease in note volume as the days elapsed increase. A second peak emerges once the stay is complete, aligning with the discharge summary. Consequently, a significant volume of notes is available before the patient's discharge, representing a valuable resource for making live predictions of ICD-9 codes.

Furthermore, Figure 3.3 presents box plots illustrating the length attributed to each type of note within the EHR sequence Specifically, it visualizes the distribution of the number of "chunks", which represent 512-token segments, for each document

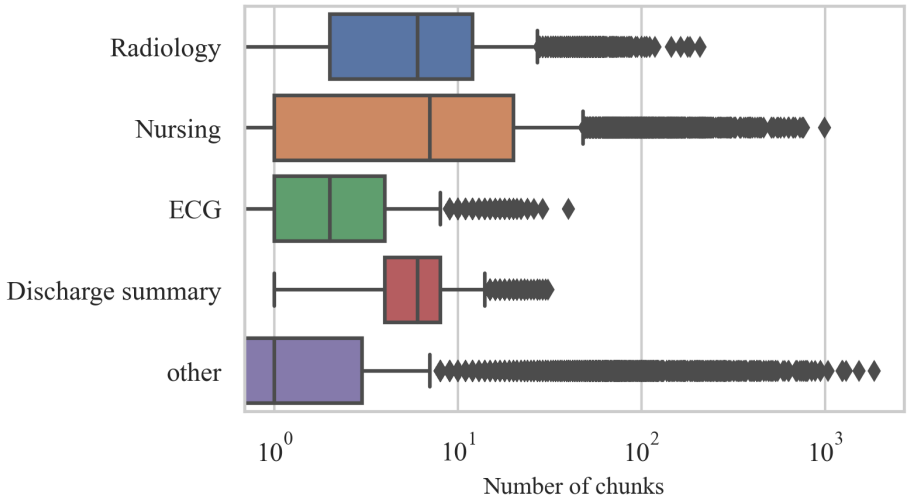


Figure 3.3: Box plot of the number of chunks (512-token segments) occupied by each document category in the clinical note sequence. The box spans the range between the first and third quartiles, including the median as a vertical line in the middle. Beyond the box, vertical lines indicate the maximum and minimum values, excluding outliers. Outliers are represented by dots located beyond this range.

category. The dataset comprises a total of 15 categories. Among these, the physician, respiratory, echo, nutrition, general, rehab services, social work, case management, consult, and pharmacy categories constitute the minority and are grouped under the ”others” category. The categories which occupy a wider span within the EHR sequence are radiology and nursing, followed by the discharge summary, ECG, and others.

3.4 Temporal ICD-9 Coding Task

This section presents the formulation of the novel temporal IC9-coding task and its corresponding evaluation framework.

3.4.1 Task specification

Based on our review of state-of-the-art models for ICD-9 code predictions and the analysis of temporal distribution in the MIMIC-III dataset, we have identified a gap in the current research concerning real-time ICD-9 code prediction. To address this, we propose a novel task of developing models capable of predicting ICD-9 codes at any point during a patient’s hospital stay without looking at the discharge summary, but using the available free-text notes instead.

In order to evaluate the performance of models on this task, there is the need to define a standardized framework. Our approach is inspired by the ClinicalBERT model, which evaluates the predictive power of readmission probabilities at differ-

ent cutoff times since admission, as described in Section 2.4.1. Similarly, we will assess the performance of the temporal ICD-9 coding task at specific periods of time elapsed since the patient was admitted at the hospital. The cut-off times are selected to be the 25%, 50%, and 75% percentiles of the total volume of notes present in our dataset, which are shown in Table 3.2. These correspond to a 2, 5 and 13 day cut-off, respectively. Additionally, we include a task of predicting the code using all notes except the discharge summary, given its late production and potential impact on medical staff workload.

	q1	**q2**	**q3**	**avg**	**min**	**max**
days elapsed	1.8	5.2	12.8	10.3	0	192
number of notes	112,594	225,160	337,726	-	-	

Table 3.2: Statistics on days elapsed since admissions for clinical notes in the MIMIC-III dataset. The 25th, 50th and 75th percentiles will be used as temporal evaluation points throughout this project.

To sum up, we propose the task to predict ICD-9 codes at any point during the hospital stay using the notes available (see Fig. 3.4). To evaluate the performance, we will compare the predictive power at different points throughout the EHR sequence:

- "2 days", "5 days", "13 days": predicting the ICD-9 codes using the notes generated in a 0—2 days, 0—5 days and 0—13 days window.
- "excl. DS": predicting the code using all notes except the discharge summary.
- "all": predicting the codes with all the notes, which corresponds to the standard post-discharge task.

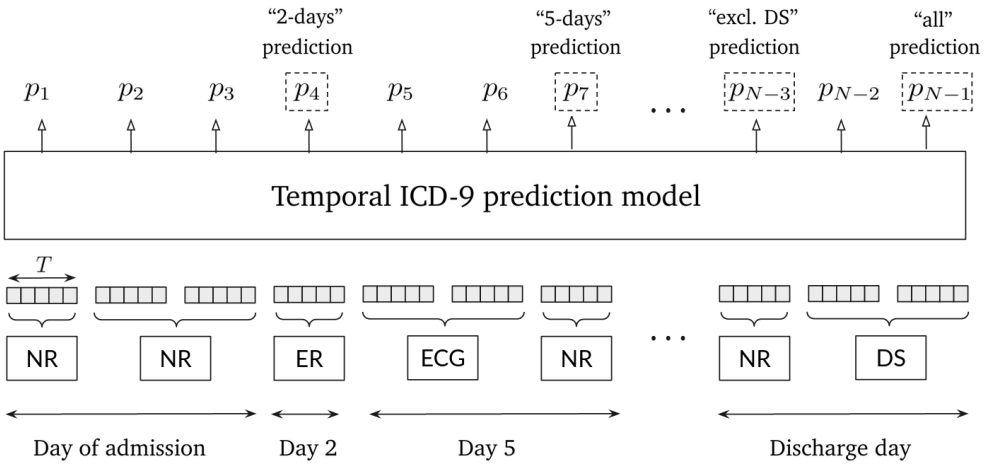


Figure 3.4: Temporal ICD-9 code prediction task illustration.

This task of temporal ICD-9 code prediction presents unique characteristics compared to the conventional ICD-9 code prediction relying on the discharge summary:

- **Adapting current models for temporal predictions**: Existing models process the entire note volume at once. To achieve temporal predictions, we will need to employ masking techniques, ensuring that the model only considers past information when making predictions at specific time points.

- **Length of clinical records**: The average discharge summary contains only 6.8 chunks of 512 tokens, while the other medical notes comprise an average of 49.0 chunks. The sheer volume of notes exceeds the capacity of current transformer models, necessitating a well-defined selection strategy and experimental set-up.

- **Low correlation with ICD-9 codes**: While the discharge summary directly relates to patient diagnosis and procedures, other notes, like radiology notes, may contain more detailed descriptions of medical procedures, images, or symptoms, which are only indirectly correlated with the ICD-9 taxonomy.

These challenges are tackled throughout this project with the design of a temporal model (Section 4.2), the implementation of a novel extended-length algorithm capable of processing the complete EHR sequence (Section 5.2.2), and the exploration of auxiliary tasks to improve the representations of pre-discharge documents (Section 5.1).

3.4.2 Evaluation framework

The model's evaluation will encompass various metrics, following a similar approach to the original ICD-9 code prediction work by Mullenbach et al. [2]. However, a key difference for our project is the need for evaluation at multiple temporal cut-off points. This section outlines how the evaluation samples for each time frame are constructed.

Firstly, the notes in the EHR sequence are divided into chunks of 512 tokens each, and these chunks corresponding to all notes are concatenated to form the text sequence:

$$s = [s_1, \ldots, s_N] \tag{3.1}$$

where N represents the maximum number of chunks allowed, limited to 16 for model development. s_i refers to the i-th chunk of the sequence. Each text sequence is associated with a time sequence d containing the timestamp of each note in days, a sequence of note categories c, and a set of L labels l:

$$d = [d_1, \ldots, d_N], \qquad c = [c_1, \ldots, c_N], \qquad l = [l_1, \ldots, l_L] \tag{3.2}$$

With this information, we can calculate the indices corresponding to each temporal cut-off by finding the last index below each time frame:

$$cutoff_t = \max_{d_i \leq t} i \tag{3.3}$$

where t is the temporal cut-off, i.e., $t \in \{2, 5, 13\}$ days, with the special cases of $t =$ "excl. DS" representing the last note before the discharge summary, and $t =$ "all" representing the full sequence. These temporal cut-offs are calculated as follows:

$$cutoff_{\text{excl. DS}} = \max_{c_i \neq \text{DS}} i \tag{3.4}$$

$$cutoff_{\text{all}} = N - 1 \tag{3.5}$$

Hence, for every validation sample (s, l), we obtain 5 samples corresponding to 5 different temporal views:

$$(s[: cutoff_t], l), \quad t \in \{2 \text{ day, } 5 \text{ day, } 13 \text{ day, excl. DS, all}\} \tag{3.6}$$

The aggregated evaluation metrics will be calculated on these five temporal views of the dataset.

3.4.3 Evaluation metrics

For evaluating the performance of our model, we will employ the micro-averaged and macro-averaged F1-score along with the area under the ROC curve (AUC-score) [2]. The F1-score is the harmonic mean of precision and recall metrics, whereas the AUC-score is the area under the curve formed by the true positive rate (recall) and false positive rate [38]. Formally:

$$\text{Micro-F1} = 2 \times \frac{\text{Precision}_{micro} \times \text{Recall}_{micro}}{\text{Precision}_{micro} + \text{Recall}_{micro}} \tag{3.7}$$

$$\text{Macro-F1} = 2 \times \frac{\text{Precision}_{macro} \times \text{Recall}_{macro}}{\text{Precision}_{macro} + \text{Recall}_{macro}} \tag{3.8}$$

$$\text{Micro-AUC} = \int_0^1 \text{TPR}_{micro} d\text{FPR}_{micro} \tag{3.9}$$

$$\text{Macro-AUC} = \int_0^1 \text{TPR}_{macro} d\text{FPR}_{macro} \tag{3.10}$$

These metrics can be derived from micro and macro-averaged precision, recall (true positive rate), and false positive rate. Micro-averaged metrics give the same weight to each sample by using the total counts of samples, whereas macro-averaged metrics assign equal weight to all classes by using the counts per class. Formally:

$$\text{Precision}_{micro} = \frac{TP}{TP + FP} \qquad \text{Precision}_{macro} = \sum_{i=1}^{L} \frac{TP_i}{TP_i + FP_i} \tag{3.11}$$

$$\text{Recall}_{micro} = \frac{TP}{TP + FN} \qquad \text{Recall}_{macro} = \sum_{i=1}^{L} \frac{TP_i}{TP_i + FN_i} \tag{3.12}$$

$$\text{FPR}_{micro} = \frac{FP}{FP + TN} \qquad \text{FPR}_{macro} = \sum_{i=1}^{L} \frac{FP_i}{FP_i + TN_i} \tag{3.13}$$

$$\text{TPR}_{micro} = \frac{TP}{TP + FN} \qquad \text{TPR}_{macro} = \sum_{i=1}^{L} \frac{TP_i}{TP_i + FN_i} \tag{3.14}$$

where L is the number of labels, which is limited to the top 50 ICD-9 codes, and the counts are defined as follows:

- TP: True Positives (correctly predicted positive instances).
- TN: True Negatives (correctly predicted negative instances)
- FP: False Positives (incorrectly predicted positive instances)
- FN: False Negatives (incorrectly predicted negative instances)

To guide the development experiments, the Micro-F1 score will be used.

3.5 Task Set-ups

As discussed in the Section 3.3.2, the complete sequence of clinical notes is, on average, 55.8 chunks long, each of which consists of 512 tokens. This length exceeds the capacity of current models, necessitating a selection strategy. Previous works [34] have addressed this challenge by defining various experiment set-ups with different note selection approaches for evaluation. For model development, a limit of 16 chunks of 512 tokens will be used, aligned with the GPU capabilities. The temporal ICD-9 coding task will be evaluated using three experiment set-ups.

3.5.1 "Latest" set-up

In this approach, the last chunks of the EHR sequence will be selected. This includes the discharge summary and the latest previous notes within the sequence limit. It aligns with current state-of-the-art models, such as the HTDC [28], that rely on the discharge summary for predictions. However, the chunk distribution statistics shown in Table 3.3 shows that this set-up has some limitations:

- The discharge summary dominates most of the sequence, with only 7 chunks on average being filled with in-hospital clinical notes.
- There is little variation of the notes used in different time intervals, ranging only between 5.0 for the 2-day window and 6.8 for the 13-day window. This limits study of the evolution of ICD-9 code prediction over time.
- The number of patients in the validation set for the 2-day, 5-day and 13-day window is considerably lower than the total (858 patients in the 2-day window versus 1573 in total). The reason is that many patients have long discharge summaries or EHR sequences, therefore selecting the last 16 chunks means that early notes from such time frames are excluded from evaluation. This leads to fewer patients in the validation set for early time ranges.

	2 days	5 days	13 days	excl. DS	all
# chunks / patient	$5.0_{\pm 3.0}$	$6.3_{\pm 3.0}$	$6.8_{\pm 2.9}$	$6.9_{\pm 2.9}$	$14.8_{\pm 2.5}$
# patients	858	1,145	1,401	1,520	1,573

Table 3.3: "Latest" set-up: select the latest 16 chunks of the sequence. This table displays the average and standard deviation of chunks per patient and the count of patients at different temporal cut-offs (development set).

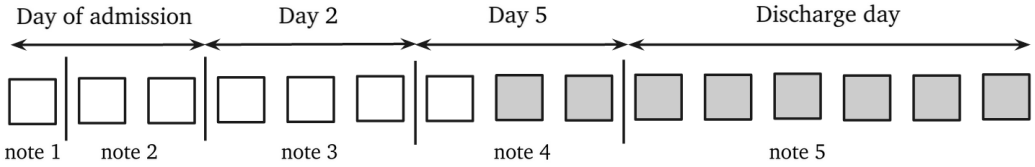


Figure 3.5: Depiction of the note selection strategy in the "Latest" set-up. The limit is set to 8 chunks for illustration purposes.

3.5.2 "Random" set-up

In order to ensure a fair comparison between predictions at different points in the temporal line, it is essential to evaluate the same set of patients at every temporal cutoff. To achieve this, the second set-up retains the first note, ensuring there is always some signal for early prediction. Additionally, the last note is included, and the rest of the sequence is filled by randomly sampling notes along the temporal line.

The benefits of this set-up are twofold:

- Firstly, it guarantees an equal number of patients for each time range in the validation set, except when considering all the notes, where 14 patients only possess the discharge summary and cannot be included in the early ICD-9 coding problem (Table 3.4).
- Secondly, it introduces more variation in the number of chunks across different time ranges, ranging from 4.4 chunks at the 2-day window to 6.6 chunks at the 13-day window. However, it is important to note that the discharge summary still occupies a significant portion of the sequence (as shown in Table 3.4).

	2 days	5 days	13 days	excl. DS	all
# chunks / patient	$4.4_{\pm 2.9}$	$5.8_{\pm 3.1}$	$6.6_{\pm 3.0}$	$6.7_{\pm 3.0}$	$14.9_{\pm 2.3}$
# patients	1,559	1,559	1,559	1,559	1,573

Table 3.4: "Random" set-up: always keep first and last note, and random sample chunks in between. This table displays the average and standard deviation of chunks per patient and the count of patients at different temporal cut-offs (development set).

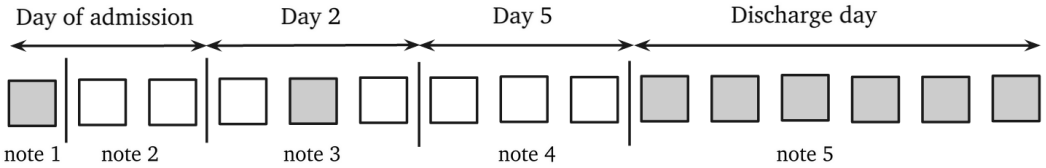


Figure 3.6: Depiction of the note selection strategy in the "Random" set-up. The limit is set to 8 chunks for illustration purposes.

3.5.3 "Limited DS" set-up

The third set-up aims to mitigate the issue of the discharge summary dominating most of the sequence. It is analogous to "Random" set-up, but with a key difference: the discharge summary is limited to only 4 chunks, allowing the model to focus more on earlier notes. This approach involves selecting the last 4 chunks of the last note, the first note, and then randomly sampling notes in between until the maximum length is reached.

As seen in Table 3.5, this set-up proves to be the most coherent for the experiments:

- Firstly, it introduces a larger variation in the number of chunks within each time cut-off, ranging from 5.7 chunks at the 2-day cut-off to 9.4 chunks at the 13-day cut-off. This will enable a more accurate study of how predictions change as more data (chunks of text) become available.
- Moreover, limiting the discharge summary allows for more chunks from earlier time ranges to be included.

	2 days	**5 days**	**13 days**	**excl. DS**	**all**
# chunks / patient	$5.7_{\pm 3.6}$	$7.9_{\pm 3.7}$	$9.4_{\pm 3.5}$	$9.9_{\pm 3.3}$	$13.8_{\pm 3.4}$
# patients	1,559	1,559	1,559	1,559	1,573

Table 3.5: "Limited DS" set-up: analogous to "Random" set-up but limiting the discharge summary to 4 chunks. This table displays the average and standard deviation of chunks per patient and the count of patients at different temporal cut-offs (development set).

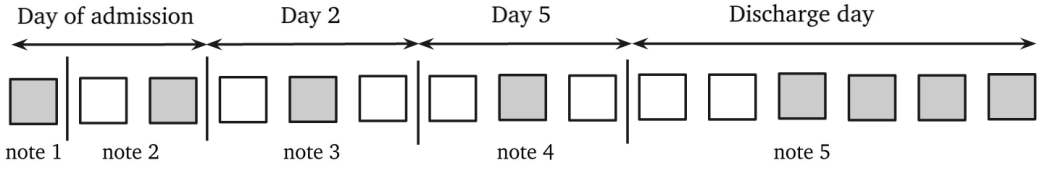


Figure 3.7: Depiction of the note selection strategy in "Limited DS" set-up. The limit is set to 8 chunks for illustration purposes.

Chapter 4

Model Design

This section presents the design of a temporal model architecture oriented towards solving the real-time ICD-9 coding task presented in the previous section. The first section describes the non-temporal model, which has an architecture based on a simplified version of the HTDC model [28], and serves as our starting point. Then, the second section describes the adaptations we propose to allow for predictions to be made at different points in the temporal line considering only past notes available.

4.1 Non-Temporal Model

This section presents a non-temporal model architecture for ICD-9 code prediction. This architecture is a simplified version of the HTDC model [28], described in section 2.3.4, and it provides a basis upon which to build our temporal model MMULA. Specifically, this section describes the process of splitting the sequence into chunks, running the chunks through a pre-trained transformer encoder, employing label-wise attention to obtain label-specific sequence embeddings and the loss used. This is summarized in the diagram displayed in Fig. 4.1. All the components are described in detail in the following subsections.

4.1.1 Chunk splitting

Firstly, the text sequence must be divided into segments, or chunks, of equal size, with each chunk containing $T = 512$ tokens. This size aligns with the maximum input length for RoBERTa pre-trained transformer encoders. Consequently, every sequence will yield a variable number of chunks based on its length. However, a constraint on the maximum number of segments N must be imposed to prevent memory overflow.

The selection of chunks relies on the chosen configuration, as discussed in Section 3.5. To recap, within the "Latest set-up," the last N chunks are used. Conversely, the "Random set-up" involves a random selection of segments while retaining those corresponding to the first and last notes. Lastly, in the "Limited DS" set-up, the

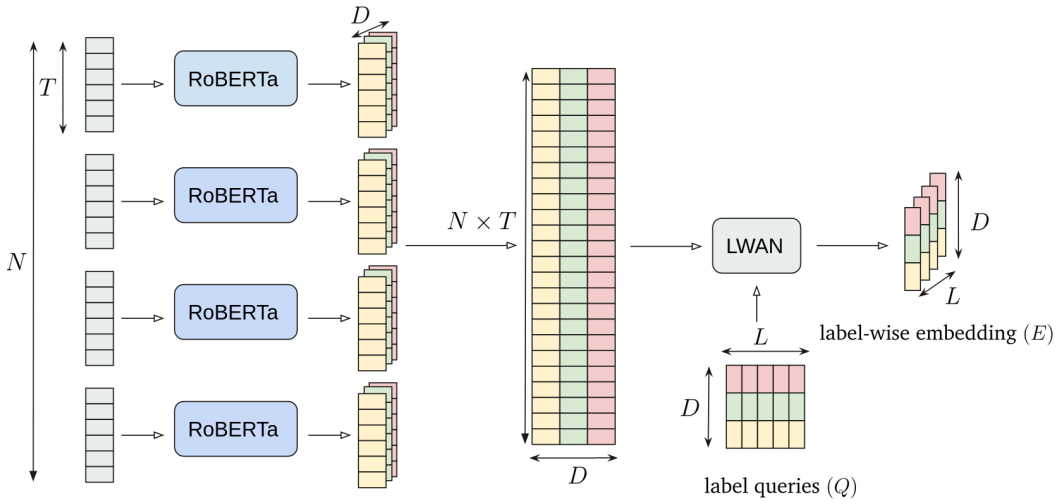


Figure 4.1: Architecture of the non-temporal model architecture (based on a simplified version of HTDC [28])

random chunk selection approach is also adopted, but the discharge summary—i.e. the last note—is confined to only 4 segments.

4.1.2 Pre-trained transformer encoder

The choice of the pre-trained transformer is essential for the ICD-9 code prediction task because it contains valuable information related to our specific domain and speeds up the training process. For our project, we use the same RoBERTa pre-trained model as in HTDC, which is the `RoBERTa-base-PM-M3-Voc` checkpoint. This model has been pre-trained on two domains: 1) PubMed and PMC, which cover biomedical publications (mixed-domain), and 2) MIMIC-III, a clinical dataset (in-domain). It uses a BPE vocabulary derived from PubMed [39].

The model processes each of the N chunks through the pre-trained transformer, resulting in a concatenation of all token embeddings, denoted as h, with a total of $N \times T$ embeddings, each having a dimensionality $D = 768$ (the RoBERTa dimension).

4.1.3 Label-wise attention

Next, the model employs a Label-Wise Attention Network (LWAN) to generate label-wise document embeddings by attending to the relevant tokens for each label. The LWAN implementation in HTDC can be described as follows:

The keys and values are the token encodings, represented as $h \in \mathbb{R}^{(N \times T) \times D}$. Additionally, label queries are defined as $q \in \mathbb{R}^{L \times D}$, where L is the number of labels. The attention weights a_w are computed using a softmax dot product between h and q:

$$a_w = \text{SoftMax}(q \cdot h^T) \in \mathbb{R}^{L \times (N \times T)} \tag{4.1}$$

Subsequently, the final label-wise document embeddings are obtained by applying these attention weights to the token embeddings:

$$d = a_w \cdot h \in \mathbb{R}^{L \times D} \tag{4.2}$$

These embeddings are then projected using linear weights $w \in \mathbb{R}^{L \times D}$, and a sigmoid function is applied to obtain the probabilities. For a specific label l, with its corresponding document embedding $d_l \in \mathbb{R}^{1 \times D}$ and weight $w_l \in \mathbb{R}^{1 \times D}$, the probability for that label is calculated as:

$$p_l = \text{sigmoid}(w_l \cdot d_l^T) \tag{4.3}$$

4.1.4 Binary cross-entropy loss

The loss applied is the standard binary cross-entropy loss, which is computed for each label independently and then averaged across the 50 labels. The multi-label binary cross-entropy loss is calculated as follows:

$$\mathcal{L} = -\frac{1}{L} \sum_{l=1}^{L} [y_l \cdot \log(p_l) + (1 - y_l) \cdot \log(1 - p_l)] \tag{4.4}$$

where:

- L is the number of labels (50 in this case),
- y_l is the ground truth binary label for label l,
- p_l is the predicted probability for label l (obtained from the sigmoid activation as mentioned earlier).

The loss is calculated independently for each label l, and then the average is taken over all labels to obtain the final binary cross-entropy loss.

4.2 Temporal Model: MMULA

This section describes our proposed model architecture designed to solve the temporal ICD-9 prediction task, which we refer to as the **M**asked **MU**ltihead **L**abel-Wise **A**ttention (**MMULA**) model. We propose several modifications to the previously defined non-temporal model. These adaptations include the choice document representation, the incorporation of a masked hierarchical transformer to merge information from different documents, and the use of masked multihead label-wise attention to generate label-wise embeddings and predictions at distinct temporal points throughout the sequence. This architecture is summarized in the diagram of Fig. 4.2, and all the components are described in the following subsections.

4.2.1 Chunk representation

First of all, the chunk representation used after the first-layer pre-trained transformer encoder is modified. Instead of keeping all the token embeddings from each segment, the chunk representation is selected to be the embedding of the [CLS]-token. This simplification reduces experiment time, allowing for more tests. Additionally, it clarifies the concept of document embedding, making it more practical than managing a sequence of token embeddings.

After processing each chunk with the pre-trained transformer, we obtain a sequence of N chunk embeddings of dimension D, which we denote as $e \in \mathbb{R}^{N \times D}$.

4.2.2 Masked hierarchical transformer

Our temporal model introduces a masked hierarchical transformer that operates on the embeddings generated by the pre-trained transformer. This transformer runs over the sequence of chunk embeddings $e \in \mathbb{R}^{N \times D}$ and generates another sequence of embeddings $h \in \mathbb{R}^{N \times D}$ which combines information over past documents.

A straightforward implementation of a hierarchical transformer encoder is not suitable for our temporal model, as it would introduce future information into earlier chunk embeddings. Consequently, we propose to employ a masked attention mechanism to prevent attention to future chunk embeddings. This is achieved with the use of an attention mask $a \in \mathbb{R}^{N \times N}$, defined as:

$$a[i,j] = \begin{cases} 0, & \text{if } j \leq i \\ -\infty, & \text{otherwise} \end{cases} \tag{4.5}$$

The masked transformer uses a variant of the attention formula described in Section 2.2.1 which involves the attention mask a:

$$\text{MaskedAttention}(e_q, e_k, e_v, a) = \text{SoftMax}(\frac{e_q e_k^T}{\sqrt{D_k}} + a)e_v \tag{4.6}$$

where the e_q, e_k and e_v are the projections of the embeddings $e \in \mathbb{R}^{N \times D}$ to the query, key and value space. Specifically, the mask causes scaled dot-product values associated with future indices to be set to $-\infty$ before the softmax operation. Consequently, attention weights for future elements are nullified [10].

After the application of the masked hierarchical transformer, the sequence embeddings $h \in \mathbb{R}^{N \times D}$ are obtained, where each representation $h_i \in \mathbb{R}^D$ for $i \in \{0, ..., N-1\}$ has been obtained by attending to all past chunk embeddings $\{e_i\}_{j=0}^{i}$. This allows information from various data chunks to be combined in an autoregressive manner, enabling the model to capture wider patterns and dependencies in the data.

4.2.3 Masked multi-head label-wise attention

We propose a novel module, which we call Masked Multi-head Label-wise Attention, to generate temporal label-wise embeddings and enable ICD-9 code predictions over time. This approach has two key differences from existing label-wise attention methods. Firstly, it utilizes masked attention to create temporal embeddings. Secondly, it employs a more general multihead attention mechanism involving transformations to query, key, and value spaces, followed by concatenation of multiple attention heads.

The starting point are the sequence embeddings $h \in \mathbb{R}^{N \times D}$ obtained through the masked hierarchical transformer. Then, the goal is to obtain N label-wise document embeddings, denoted as $d_t \in \mathbb{R}^L$ for $t \in \{1, \cdots N\}$ which will be later converted into N temporal label-wise probabilities $p_{t,l}$ for $t \in \{1, \cdots N\}$ and $l \in \{1, \cdots L\}$, one for each sequence position and label.

Specifically, the process involves applying masked multihead label-wise attention at various sequence positions $t \in \{1, \cdots, N\}$. The label embeddings $q \in \mathbb{R}^{L \times D}$ serve as queries, whereas keys and values are sequence embeddings $h \in \mathbb{R}^{N \times D}$. To prevent attention beyond position t, an attention mask $\bar{a}_t \in \mathbb{R}^{L \times N}$ is used, which is constant in the label dimension, and varies in the temporal dimension:

$$\bar{a}_t[:, i] = \begin{cases} 0, & \text{if } i \leq t \\ -\infty & \text{otherwise} \end{cases} \tag{4.7}$$

Then the label-wise document embedding $d_t \in \mathbb{R}^{L \times D}$ at position t is obtained by using masked multihead attention (as described in equation 2.2):

$$d_t = \text{MaskedMultiHeadAttention}(q, h, h, \text{mask} = a_t) \tag{4.8}$$
$$= \text{Concat}(\text{head}_1, \cdots, \text{head}_n)W^o \tag{4.9}$$

where head within masked multihead attention employs the masked attention concept from the previous section (equation 4.6):

$$\text{head}_i = \text{MaskedAttention}(qW_i^Q, hW_i^K, hW_i^V, a_t) \tag{4.10}$$

Finally, temporal probabilities result from projecting the embedding using linear weights $w \in \mathbb{R}^{L \times D}$ followed by a sigmoid activation. Formally, the probability at time t for label l is calculated using the the label weight $w_l \in \mathbb{R}^D$ and the label document embedding at position t, denoted as $d_{t,l} \in \mathbb{R}^D$:

$$p_{t,l} = \text{sigmoid}(w_l \cdot d_{t,l}) \tag{4.11}$$

The model's output is a probability matrix $p \in \mathbb{R}^{N \times L}$, containing probabilities for each label at each temporal point. The masking process within the transformer and label attention modules ensure that time t probability calculations consider only past documents, fulfilling the requirements of the temporal prediction task.

In the practical implementation, temporal label-wise probabilities for each chunk position $t \in \{1, \cdots N\}$ are computed in parallel by assigning the temporal dimension to the batch dimension. This parallelized approach results in the collective computation of the sequence of temporal label-wise document embeddings $d \in \mathbb{R}^{N \times L \times D}$ and probabilities $p \in \mathbb{R}^{N \times L}$ in a single iteration. Note how this differs from the outputs obtained using non-temporal models, where d has shape $L \times D$ and p has shape L.

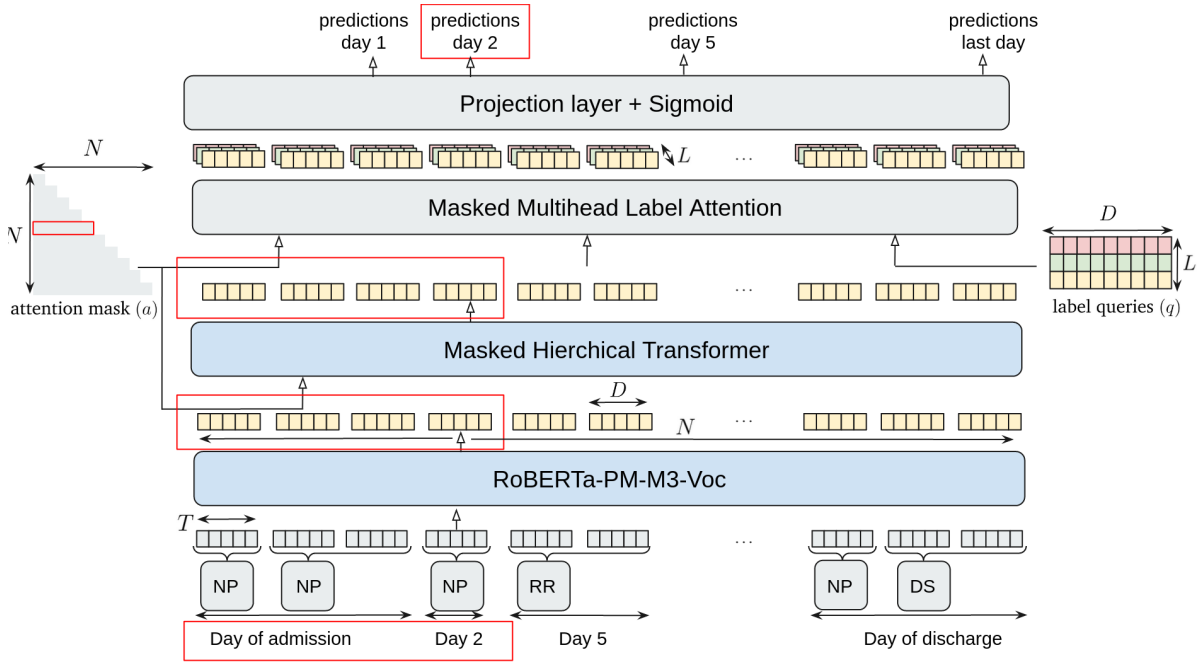


Figure 4.2: Architecture of our temporal model, named MMULA (Masked Multihead Label-wise Attention), which is designed to enable temporal predictions. The portions outlined by red rectangles exemplify the information flow for the 2-day prediction task.

Chapter 5

Enhancing early prediction

This section explores two distinct research avenues to enhance the performance of the previously developed model MMULA on the temporal ICD-9 coding task. The development experiments (Section 6.1.2) suggest that the discharge summary plays a vital role, and without it, early document embeddings struggle to predict ICD-9 codes throughout a patient's hospital stay (pre-discharge). For this reason, in Section 5.1 we will explore the use of multi-objective training to refine early document embeddings using domain-specific knowledge through two distinct auxiliary tasks. On the other hand, the experiments also reveal a strong input sequence length limitation which especially impacts early prediction. Consequently, in Section 5.2 we will propose an improved algorithm to handle a very lengthy sequence of text, such as the complete EHR sequence, eliminating the need for a note selection strategy.

5.1 Multi-Objective Training

The first line of research is to refine document embeddings with auxiliary objectives during training. The current model is trained in an end-to-end fashion, learning the document embeddings that are optimal for the end goal of ICD-9 code prediction. The idea behind multitask or multiobjective learning is to introduce a supervision signal on the intermediate representations, which are the document embeddings, employing domain-specific knowledge to improve the final task [40].The first task aims to optimize embeddings for next document category prediction, whereas the second task consists in the prediction of the last document embedding, which contains the discharge summary, to enhance early document embeddings.

5.1.1 Document Category prediction

The first auxiliary task is to optimize document embeddings for next document category prediction. This approach is based on the idea that the types of future notes that the patient will receive are related to the diagnoses and procedures. For instance, a patient with a heart condition is more likely to receive an electrocardiogram report (ECG) [41]. Consequently, enhancing document embeddings to predict future note

categories is expected to improve ICD-9 code prediction.

Specifically, we aim to optimize the embeddings obtained with the masked transformer, denoted as $h \in \mathbb{R}^{N \times D}$, to predict the category of the subsequent document. This corresponds to a multi-class classification problem with 15 classes: 14 document types plus one additional class for the "end-of-sequence" (EOS). To solve this task, we add a dedicate classification head which takes as input a document embedding $h_t \in \mathbb{R}^D$ and generates a vector probabilities for each class $p_t \in \mathbb{R}^{15}$. The overall architecture is depicted in Fig. 5.1.

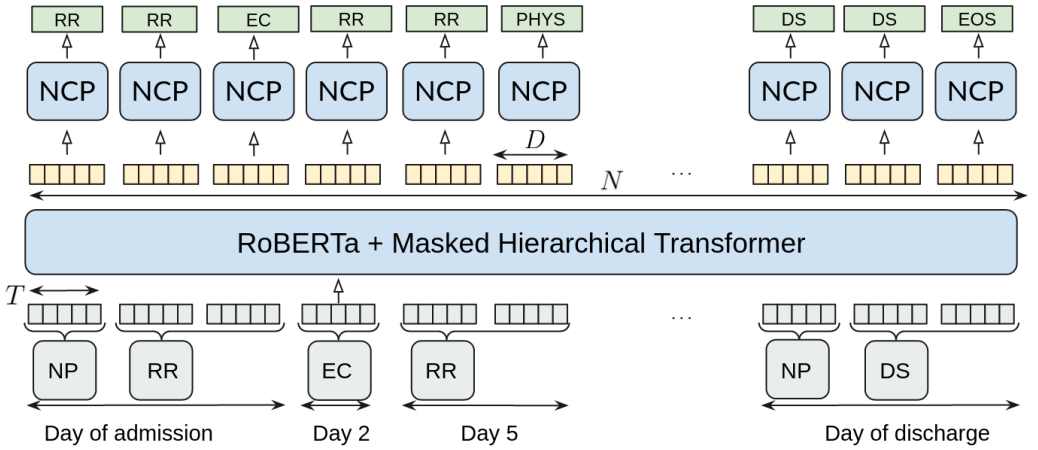


Figure 5.1: Next document category prediction (NCP) auxiliary task architecture.

The classifier (NCP) is a neural network with two layers and a ReLU activation. The initial linear layer projects the embeddings from size D to $D/2$, followed by the second linear layer, which further projects from $D/2$ to $C = 15$. The final layer applies the softmax function, which is well-suited for multiclass classification. Consequently, the next category probabilities $p_t \in \mathbb{R}^C$ for each document index t is are computed as:

$$p_t = \text{SoftMax}(\text{Linear}_2(\text{ReLU}(\text{Linear}_1(h_t)))) \tag{5.1}$$

The true labels for this task $y_{t,i}$, where t indicates the sequence index and i represents the category index, are established based on the extended category sequence $\hat{c}$:

$$\hat{c} = (c_1, \cdots, c_N, [\text{EOS}]) \tag{5.2}$$

$$y_{t,i} = \begin{cases} 1 & \text{if } \hat{c}_{t+1} = i \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

The loss applied is a cross-entropy loss, which is suited for multi-class classification, averaged across all sequence indexes:

$$\mathcal{L}_{NCP} = -\frac{1}{N} \sum_{t=1}^{N} \sum_{i=1}^{C} y_{t,i} \cdot log(p_{t,i}) \tag{5.4}$$

The final loss consists of the binary cross-entropy loss for ICD-9 code prediction, defined in Section 4.1.4, plus the weighted auxiliary loss:

$$\mathcal{L}_{total,1} = \mathcal{L} + w_{aux}\mathcal{L}_{NCP} \tag{5.5}$$

Here, w_{aux} is the weight assigned to the auxiliary loss term, which signifies the importance given to the auxiliary task during training.

5.1.2 Document Embedding prediction

The second auxiliary task addresses the challenge of not having access to the discharge summary through a patient's hospital stay. The objective here is to optimize early document embeddings for the generation of the last document embedding, which does have access to the summary. By accomplishing this, the auxiliary task aims to encourage early document embeddings to focus on information relevant to the summary and likely related to the final diagnoses and procedures, therefore enhancing the final ICD-9 code prediction task.

Specifically, we aim to optimize the embeddings obtained with the masked transformer, denoted as $h_t \in \mathbb{R}^D$ for t in $\{1, \cdots, N\}$ to predict the last embedding h_N. To solve this auxiliary task, we propose to attach a document embedding generator, which is a neural network, onto the hierarchical transformer. At each temporal position, this neural network generates an additional document embedding, which is subsequently optimized to closely resemble the last document embedding using a cosine loss. This architecture is depicted in Fig. 5.2.

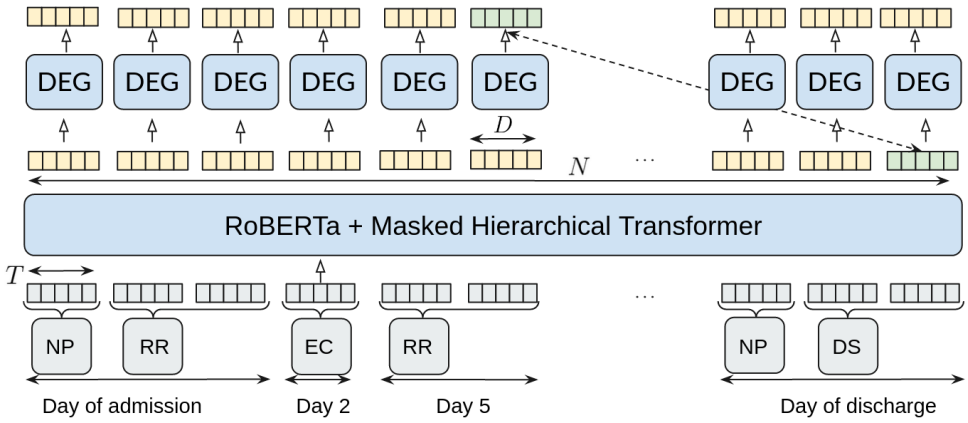


Figure 5.2: Last document embedding generation (DEG) auxiliary task architecture.

Mathematically, the document embedding generator follows an encoder-decoder structure. The document embedding for index t, represented as $h_t \in \mathbb{R}^d$, undergoes dimension reduction to $D/2$ through a linear layer, followed by a second linear layer for restoration to the original dimension. ReLU activation functions are applied. In summary, the predicted document embedding $\hat{h}_t$ for index t is calculated as:

$$\hat{h}_t = \text{Linear}_2(\text{ReLU}(\text{Linear}_1(h_t))) \tag{5.6}$$

The employed loss is the cosine embedding loss [42], commonly used for non-linear embedding learning [43]. In this context, the predicted embedding is denoted as $\hat{h}_t$, as previously explained, while the ground truth embedding is taken to be the last embedding h_N. The last embedding is detached from the computational graph for the loss calculation. Consequently, the auxiliary loss is computed as the average cosine embedding loss over all indices $t \in \{1, \cdots, N\}$:

$$\mathcal{L}_{DEG} = \frac{1}{N} \sum_{t=1}^{N} (1 - \cos(\hat{h}_t, h_N)) \tag{5.7}$$

Once more, the final loss is derived by adding a weighted auxiliary loss to the binary cross-entropy loss:

$$\mathcal{L}_{total,2} = \mathcal{L} + w_{aux}\mathcal{L}_{DEG} \tag{5.8}$$

5.2 Extended-Length Algorithm

The second research direction involves developing an algorithm to enable hierarchical transformers to process very lengthy text sequences for classification. This direction is inspired by the experiments described in Section 6.1.2. The results indicate that the subpar performance in early prediction tasks could be attributed to a limited allocation of text for early time cut-offs.

Conventionally, models for ICD-9 code classification employ the discharge summary, which typically spans 7 chunks, each containing up to 512 tokens. While this exceeds conventional transformer capacity, it remains feasible through hierarchical architectures or the use of sparse attention in a long transformer. However, when making early predictions without the discharge summary, the average sequence length is around 49 chunks (Table 3.1). Since there is no specific summary section, all documents need consideration for accurate predictions. This calls for an improved algorithm to handle such extended text.

With this motivation, in this section we propose a novel algorithm to enable models to handle text sequences of indefinite length for classification. It is based on random sampling during training and an iterative inference process. Even though this algorithm will be applied to our current model, MMULA, in theory it should be applicable to any model for text classification which uses a label-wise attention network and a transformer base model.

5.2.1 Complete EHR distribution set-up

The algorithm discussed in this section can be applied to the entire sequence of clinical notes. As a result, the evaluation methodology employed here is different from

the three previous set-ups, each of which employs different strategies to truncate the sequence in order to fit within the 16 chunk constraint, as described in Section 3.5. For this reason we introduce the "Complete EHR" set-up, which does not impose limitations on the number of chunks allowed.

	2 days	**5 days**	**13 days**	**excl. DS**	**all**
# chunks / patient	$17.9_{\pm 22.1}$	$27.6_{\pm 33.9}$	$35.8_{\pm 42.4}$	$40.4_{\pm 47.7}$	$48.4_{\pm 48.1}$
# patients	1,559	1,559	1,559	1,559	1,573

Table 5.1: "Complete EHR" set-up: select all the chunks comprising the complete EHR sequence. This table displays the average and standard deviation of chunks per patient and the count of patients at different temporal cut-offs (development set). It should be noted that a generous cap of 181.0 chunks is employed due to memory overflow issues.

The statistical characteristics of the "Complete EHR" set-up, are provided in Table 5.1. The results indicate a significant increase in the allocation of text chunks for earlier time cutoffs. A comparison with the distribution of the "Limited DS" set-up presented in Table 3.5 underscores this trend: the number of chunks for the 2-day cutoff increases from 5.7 to 17.9, for the 5-day cutoff it rises from 7.9 to 27.6, for the 13-day cutoff it expands from 9.4 to 35.8, and for the exclusion of the discharge summary it extends from 13.8 to 40.4. This highlights the considerable amount of text that was previously omitted using the standard model.

The patient numbers in the validation set remain consistent with set-ups 2 and 3. Specifically, there are 14 fewer patients in the validation set for early prediction tasks compared to the standard predictive task that encompasses all text. This is due to these patients only having a discharge summary.

5.2.2 Algorithm specification

The Extended-Length Algorithm (ELA) is a novel solution to overcome the sequence length limitation of current hierarchical transformer models. Upon analyzing this model architecture, a key bottleneck was identified: the first-layer transformer. Experimentally, it was found that the batch size can only grow up to 16 before straining our GPU resources. Yet, the label-wise attention network does not have input length restrictions. Consequently, we propose to do inference by running the model on the entire text in batches of 16 chunks, retaining only the last document embeddings before the label-wise attention step. Then all the document embeddings can be parsed through the label-wise attention network to generate a prediction based on the entire text.

However, this approach poses a challenge: during training, it is not feasible to propagate gradients across all document embeddings. Training solely on a portion of

the sequences seems insufficient for predicting with the full text. To address this, we propose to adopt a strategy akin to data augmentation, which involves random sequence sampling during training. This method introduces diverse augmentations of the sequence aggregating to only 16 chunks for each training iteration.

The resulting approach is summarized in the pseudo-code algorithms shown in 1 and 2, which we have named **E**xtended-**L**ength text classification **A**lgorithm (**ELA**). We call this an algorithm, rather than a model, because it relates to the training and inference processes, rather than the model architecture itself. It consists of two parts: the training loop, and the iteration loop.

Training loop (Algorithm 1). For each episode of training, the loop iterates over the training dataset $\mathcal{D}_{train}$, processing each data sample (s, y), where s represents the input sequence and y is the corresponding label. Within the loop, a random selection of indexes is made to create a subset of the input sequence, with the maximum number of chunks set as N. This sub-sequence, denoted as s_{rand}, is used for the model's forward pass, which returns both predictions p and the final document embeddings h. The binary cross-entropy loss ($\mathcal{L}$) between the predictions p and the true labels y is calculated, followed by a backward pass through the model and an optimization step.

Inference loop (Algorithm 2). For each input sequence s in the test set $\mathcal{D}_{test}$, the loop processes it in chunks of size N, where N is the maximum number of chunks. Within the inner loop, the input sequence is split into batches, and the model's forward pass is executed to obtain predictions p_{batch} and document embeddings h_{batch} for each batch. The embeddings for each batch are appended to the h_{list}, which holds the embeddings of all processed chunks. After processing all chunks of the input sequence, the embeddings are concatenated along the batch dimension to form the complete embedding sequence h. Finally, the label-wise attention network (model.lwan) is applied to h to obtain predictions p based on the complete sequence.

Algorithm 1 ELA Training loop

$\mathcal{D}_{train} \leftarrow$ training set (sequence-label pairs)
$N \leftarrow$ max. number of chunks
for each episode **do**
  for each (s, y) in $\mathcal{D}_{train}$ **do**
    $m \leftarrow min(N, len(s))$
    select m random indices $i_1, ...i_m$ from the sequence $[0, \cdots, len(s) - 1]$
    sort $i_1, \cdots, i_m$ in ascending order
    $s_{rand} \leftarrow [s[i_1], \cdots, s[i_m]]$
    $p, h \leftarrow$ model.forward(s_{rand})
    $\mathcal{L} \leftarrow BCE(y, p)$
    do backward pass and optimizer step
  end for
end for

Algorithm 2 ELA Inference loop

$\mathcal{D}_{test} \leftarrow$ test set (no labels)
$N \leftarrow$ max. number of chunks
for each s in $\mathcal{D}_{test}$ **do**
    $h_{list} \leftarrow$empty list
    for i in range$(0, len(s), N)$ **do**
        $s_{batch} \leftarrow s[i : i + N]$
        $p_{batch}, h_{batch} \leftarrow$ model.forward(s_{batch})
        append h_{batch} to h_{list}
    end for
    $h \leftarrow$ concatenate h_{list} along batch dim.
    $p \leftarrow$ model.lwan(h)
end for

Chapter 6

Evaluation

This chapter presents the evaluation of MMULA and the various enhancement ideas detailed in the previous chapter. Specifically, the first section describes the experiments performed on MMULA related to hyper-parameter tuning and temporal evaluation across different setup configurations. Then, the following section evaluates the effect of multi-objective training with the two auxiliary tasks previously specified, as well as the effect of training and inference using the extended-length algorithm (ELA). The subsequent section presents ablation studies of our model. Finally, the last section provides a qualitative evaluation of our model through an interpretability analysis.

6.1 Temporal evaluation of MMULA

In this section we present the experiments conducted on the MMULA model, which we have developed to tackle the temporal ICD-9 code prediction task. First, the hyperparameters introduced by the temporal architecture are tuned. Then, the model is evaluated using a temporal cut-off framework, as defined in Section 3.4.2, across different set-up configurations. Since this is a novel task, these results serve as a benchmark for temporal ICD-9 code prediction. Finally, the results are analyzed to understand the factors that contribute to limited early prediction power and guide model enhancement. In particular, this analysis guides the research directions explored in Chapter 5.

6.1.1 Model tuning

The hyperparameters introduced by the temporal architecture MMULA undergo tuning. These hyperparameters include the number of attention heads in the multi-head label-wise attention network, as well as the number of layers and attention heads within the hierarchical transformer. The learning rate is also tuned, which is a critical parameter in the training process.

The hyper-parameter tuning experiments are run with only 8 chunks to speed up the search. The evaluation metrics are reported on the standard task to predict with

the entire sequence, herein referred to as "all". The additional temporal prediction tasks at "2 days", "5 days", "13 days" and "excl. DS" will be evaluated in the subsequent sections. The hyperparameters are tuned sequentially, by exploring a range of values to determine the optimal, and then using this value for the experiments of the subsequent hyperparameter. The results are shown in Table 6.1. It was observed that having a single attention head is optimal, therefore simplifying the architecture to basic label-wise attention. Similarly, favoring a shallow hierarchical transformer with just one layer and one attention head proved effective. The optimal learning rate was identified as $5e-5$.

	Micro-F1	Macro-F1	Micro-AUC	Macro-AUC
# attn heads (LWAN) = 1	**63.53**	**58.27**	**90.81**	**88.17**
# attn heads (LWAN) = 2	63.48	58.37	90.72	87.97
# attn heads (LWAN) = 3	63.38	58.3	90.76	88.11
# layers (HT) = 1	**63.59**	**58.37**	**90.89**	**88.25**
# layers (HT) = 2	63.38	58.24	90.8	88.18
# layers (HT) = 3	56.02	46.01	87.98	86.09
# attn heads (HT) = 1	**63.41**	**58.37**	**90.69**	**87.92**
# attn heads (HT) = 2	63.34	57.55	90.83	88.05
# attn heads (HT) = 3	63.31	57.87	90.72	88.1
lr = 1e-5	61.6	53.53	91.02	88.18
lr = 5e-5	**63.29**	**57.87**	**90.78**	**88.12**
lr = 1e-4	63.12	57.5	91.05	88.24

Table 6.1: Results of hyper-parameter tuning (development set). The experiments are run with 8 chunks in the "Limited DS" set-up, and the metrics are calculated considering all chunks.

6.1.2 Set-up experiments

We assess our temporal model for ICD-9 code prediction, MMULA, using the temporal evaluation framework outlined in Section 3.4.2. This framework involves computing aggregated metrics to evaluate model performance using notes within three distinct time windows: 0-2 days, 0-5 days, and 0-13 days. We also include the task of predicting ICD-9 codes using all notes except for the discharge summary, or alternatively, using all notes to replicate the standard ICD-9 coding task.

Our GPU resources allow us to process a maximum of 16 chunks per sequence, which corresponds to a maximum input of 8,192 tokens. Given that the complete Electronic Health Record (EHR) sequence comprises an average of 56 chunks in length, we evaluate our model using three different sequence configurations. These configurations employ distinct strategies for chunk selection, referred to as the "Latest," "Random" and "Limited DS" setups (as detailed in Section 3.5). The results are

presented in Tables 6.2, 6.3, 6.4, and 6.5.

	2 days	5 days	13 days	excl. DS	all
latest	33.5 ± 0.6	37.6 ± 0.3	38.2 ± 0.1	37.9 ± 0.2	71.0 ± 0.3
random	29.2 ± 0.5	35.1 ± 0.4	37.7 ± 0.5	38.4 ± 0.4	**71.3** ± 0.2
limited DS	**39.8** ± 0.1	**45.2** ± 0.1	**49.1** ± 0.1	**50.2** ± 0.2	65.2 ± 0.1

Table 6.2: Comparison of Micro-F1 score (dev set) across set-ups.

	2 days	5 days	13 days	excl. DS	all
latest	27.9 ± 0.7	32.5 ± 0.4	33.4 ± 0.2	33.1 ± 0.2	66.1 ± 0.5
random	24.2 ± 0.6	30.1 ± 0.4	32.5 ± 0.5	33.3 ± 0.5	**66.7** ± 0.2
limited DS	**33.7** ± 0.3	**39.7** ± 0.3	**43.6** ± 0.2	**45.1** ± 0.3	60.2 ± 0.3

Table 6.3: Comparison of Macro-F1 score (dev set) across set-ups.

	2 days	5 days	13 days	excl. DS	all
latest	77.5 ± 0.3	79.4 ± 0.4	79.3 ± 0.3	78.8 ± 0.3	94.0 ± 0.0
random	72.1 ± 0.2	76.0 ± 0.2	78.5 ± 0.2	79.3 ± 0.2	**94.2** ± 0.1
limited DS	**76.8** ± 0.1	**80.7** ± 0.1	**84.1** ± 0.1	**85.0** ± 0.1	92.0 ± 0.1

Table 6.4: Comparison of Micro-AUC score (dev set) across set-ups.

	2 days	5 days	13 days	excl. DS	all
latest	73.7 ± 0.5	76.3 ± 0.2	76.4 ± 0.2	76.2 ± 0.2	92.0 ± 0.1
random	69.0 ± 0.2	72.8 ± 0.2	75.2 ± 0.1	76.1 ± 0.1	**92.2** ± 0.2
limited DS	**73.7** ± 0.2	**77.5** ± 0.1	**80.8** ± 0.1	**81.9** ± 0.1	89.6 ± 0.1

Table 6.5: Comparison of Macro-AUC score (dev set) across set-ups.

6.1.3 Analysis of temporal predictions

The analysis of the results obtained in the previous section shed light on the characteristics of the novel task of temporal ICD-9 code prediction. Several insights can be extracted by comparing the performance across different set-ups and temporal cut-offs:

1. Micro-F1 improvement over time: In set-ups 2 and 3, prediction accuracy improves as more time elapses and additional chunks become available. This validates the initial requirement of developing a model whose predictions gradually improved over time. The only exception is “Latest” set-up, where there

is a marginal decrease in Micro-F1 score between the 13 day cut-off and the exclusion of DS. This can be attributed to the fact that there is a minimal variation between the chunks selected for both cut-offs, as elaborated in Section 3.5.

2. Importance of the discharge summary: There is a significant drop in performance when the DS is excluded from the sequence in all set-up configurations. In the "Latest" set-up, it drops from 71.0 to 39.8, in the "Random" set-up set-up, it decreases from 70.6 to 39.5, and in the "Limited DS" set-up, the drop is from 65.1 to 50.3. This can be attributed to two main reasons.

    - Limited availability of chunks. In early time cut-offs, less number of chunks are considered due to the sequence length limitation of the model. This is further corroborated by the experiments done in "Limited DS" set-up as elaborated in the next point.
    - Weakened alignment between early clinical notes and ICD-9 codes, i.e., less direct relation with the diagnosis and procedure codes. This inspires the first line of research described in Section 5.1, which is regarding the enhancement of early document embeddings to resemble discharge summary embeddings, which could potentially improve early prediction accuracy.

3. Effect of set-up configuration: The "Limited DS" set-up has significantly better results in early prediction compared to the first two set-ups. This can be attributed a larger portion of the sequence corresponding to early time cut-offs, as the discharge summary is confined to only 4 chunks. This leads to an average improvement of 7.9 in Micro-F1 for the 2-day cut-off, 9.4 for the 5-day cut-off, 10.6 for the 13-day cut-off and 10.8 for the "excl. DS" case versus second set-up. This inspires the second line of research described in Section 5.2.2 concerned with developing a method to make predictions based on a very extended amount of text.

These insights provided motivation for Chaper 5, where several ideas to enhance early prediction are proposed.

6.2 Evaluation of enhancement proposals

In this section, we present the outcomes of the experiments conducted to assess the enhancement strategies outlined in Chapter 5. Throughout this section, the configuration labeled as "Limited DS" has been chosen, since it has proved to be the optimal set-up for temporal ICD-9 code prediction (Section 6.1.3). We evaluate and compare the proposed enhancements, specifically the auxiliary tasks for multi-objective training and the extended-length algorithm. Furthermore, we present a discussion of the obtained results and limitations.

6.2.1 Auxiliary task comparison

The various auxiliary tasks are assessed using the development set. The experiments were conducted with a 16-chunk limit, and the auxiliary weight was set empirically as $w_{aux} = 0.1$. The Micro-F1 score is employed as the metric for comparing different enhancements. The chosen configuration for evaluation was the "Limited DS" set-up, as it yields a robust signal for early prediction, as discussed in Section 6.1.3.

	2 days	**5 days**	**13 days**	**excl. DS**	**all**
MMULA	39.8 $_{\pm 0.1}$	45.2 $_{\pm 0.1}$	49.1 $_{\pm 0.1}$	50.2 $_{\pm 0.2}$	**65.2** $_{\pm 0.1}$
MMULA + aux 1	40.2 $_{\pm 0.5}$	45.4 $_{\pm 0.4}$	49.0 $_{\pm 0.4}$	50.1 $_{\pm 0.4}$	65.0 $_{\pm 0.1}$
MMULA + aux 2	**40.8** $_{\pm 0.3}$	**46.1** $_{\pm 0.5}$	**50.0** $_{\pm 0.4}$	**51.2** $_{\pm 0.4}$	65.0 $_{\pm 0.1}$

Table 6.6: Comparison Micro-F1 score on the development set for different auxiliary training objectives, namely the next document category prediction ("aux 1") and last document embedding prediction ("aux 2"). Experiments run with 16 chunks on "Limited DS" set-up. Results show the average and standard deviation across 3 replications.

The outcomes of the experiments are detailed in Table 6.6. The first auxiliary task, namely, next document category prediction, yields only minor improvements in Micro-F1 score for the 2 days and 5 days cut-off, and shows roughly the same performance for the rest of evaluation points. On the other hand, the second task, which is the last document embedding prediction, is able to substantially enhance performance, contributing to an an increase of +0.6, +0.7, +1.0, and +1.1 in Micro-F1 score for the 2 days, 5 days, 13 days, and discharge summary exclusion cut-offs, respectively.

The evaluation shows that multi-objective training is useful to improve temporal ICD-9 code prediction. The auxiliary task of document embedding generation achieves a superior performance than the next category prediction. This can be attributed to a stronger supervision signal on the intermediate representations. The first task is only indirectly linked to the final diagnosis prediction, whereas the second task optimizes the document embeddings to closely resemble the embedding inclusive of the discharge summary. This summary has proven to be a robust signal for ICD-9 code prediction. Furthermore, the findings suggest the value of approaches that propagate information from the future to the past. Given the autoregressive nature of the model, where document embeddings depend on past context, this auxiliary task introduces a way of conveying future information to the past during training without explicit attention to future documents.

It is also worth noting that the multi-objective approach has some limitations, as the performance for the standard ICD-9 code prediction task utilizing all notes is roughly similar. Specifically, the performance experiences a slight decrease from a Micro-F1 score of 65.2 to 65.0 with multi-objective training. This underscores the need for further investigation into the architecture, which might involve fine-tuning

the auxiliary weight or refining the auxiliary losses.

6.2.2 Extended-length algorithm (ELA) evaluation

The novel extended-length algorithm (ELA), described in Section 5.2.2, is evaluated using the development set. The experiments are conducted with the "Complete EHR" set-up, as described in Section 5.2.1. In the training phase, the number of chunks is constrained to 16, and random sampling is applied. In the inference phase, the full sequence can be utilized, with a generous limit of 181.0 chunks.

The 181.0 chunk limitation is derived from the distribution of text lengths within the dataset. Despite the average sequence length being 56 chunks, there are numerous outliers, with some sequences extending up to 2,967 chunks, equivalent to a tensor of 1,519,104 tokens. This large size can lead to memory overflow issues. To mitigate this, we choose to impose a cap of 181.0 chunks, which corresponds to the 95th percentile. This ensures that 95% of the samples are unaffected by length limitations, representing a significant improvement over the previous constraint of 16 chunks.

The results in Table 6.7 show the performance of our model MMULA with and without utilizing the extended-length algorithm for training and inference.

	2 days	**5 days**	**13 days**	**excl. DS**	**all**
MMULA	39.8 ± 0.1	45.2 ± 0.1	49.1 ± 0.1	50.2 ± 0.2	65.2 ± 0.1
MMULA + ELA	**46.2** ± 0.1	**50.9** ± 0.1	**53.6** ± 0.2	**54.3** ± 0.2	**71.1** ± 0.1

Table 6.7: Comparison of Micro-F1 score on the development set for the extended-length algorithm versus baseline. The MMULA experiments are run with 16 chunks on "Limited DS" set-up. The MMULA+ELA experiments are run with a limitation of 16 chunks during training (using random sampling), and a cap of 181.0 chunks during inference. Results show the average and standard deviation across 3 replications.

The findings of these experiments are presented in Table 6.7. The outcomes reveal a substantial enhancement in early prediction due to the implementation of the extended-length algorithm. The Micro-F1 score exhibits improvements of +6.4, +5.7, +4.5, and +4.1 for the 2-day, 5-day, 13-day, and exclusion of the discharge summary scenarios, respectively. This highlights the importance of considering the entire EHR sequence for accurate early predictions.

Furthermore, the performance of the standard ICD-9 code prediction task using all available notes also experiences an improvement, rising from a Micro-F1 score of 65.2 to 71.1. It is worth mentioning that this algorithm surpasses the performance of the MMULA model trained on the "Latest" set-up by a margin of 0.1 Micro-F1 (Table 6.2), which is the typical configuration for ICD-9 code models. However, its performance is slightly below the MMULA model trained with the "Random" set-up

by 0.2 Micro-F1 (Table 6.2). This suggests the need for further exploration into the random sampling technique utilized in the training loop of the extended-length algorithm, as there is room to close the 0.2 Micro-F1 gap and match the optimal set-up's performance.

6.3 Ablation studies

In this section we conduct ablation studies to discern the components that contribute the most significant performance enhancements.

6.3.1 Effect of document embedding generator

An ablation study is conducted on the second auxiliary task to examine the impact of a key architectural choice: the discharge embedding generator (DEG). As detailed in Section 5.1.2, the generation head involves a neural network with an encoder-decoder design that produces a document embedding at each time point. Ablation experiments are conducted by removing this component. The ablated model still incorporates the auxiliary objective, but it removes the dedicated generation head, therefore guiding early document embeddings to resemble the last document embedding, without any intermediate transformation.

The results of the ablation study are presented in Table 6.8. The findings reveal that the document embedding generator (DEG) contributes significantly to a Micro-F1 score improvement of +3.2, +2.8, +1.8, and +0.7, corresponding to different temporal cut-offs. This underscores the essential role of the document embedding prediction in architectural design. Our intuition suggests that due to the complexity of the task, having a dedicated neural network for this purpose is key for achieving a performance improvement.

	2 days	5 days	13 days	excl. DS	all
aux 2	**40.8** ± 0.3	**46.1** ± 0.5	**50.0** ± 0.4	**51.2** ± 0.4	**65.0** ± 0.1
aux 2 (ablate DEG)	37.6 ± 0.6	43.9 ± 0.6	48.2 ± 0.5	49.5 ± 0.6	65.0 ± 0.2

Table 6.8: Comparison of Micro-F1 score on the development set of the model MMULA trained with the second auxiliary task (last document embedding prediction) with and without ablating the document embedding generator component. Results show the average and standard deviation across 3 replications.

6.3.2 Effect of random sampling in ELA

Finally, we conduct an ablation study on the extended-length algorithm (ELA) to evaluate the necessity of the modified training loop within the algorithm. We carry

out this study by excluding random sampling from the training loop, thus employing only the last 16 chunks during training, akin to the "Latest" set-up or training with the discharge summary. Consequently, the resulting model is trained in a conventional manner, yet performs inference on the complete EHR sequence through the iterative process outlined in Section 5.2.2.

The results of the ablation experiments are presented in Table 6.9. The outcomes demonstrate that the ablated model performs considerably worse in early prediction, with a reduction in Micro-F1 scores of -8.4, -5.7, -4.5, and -4.1 for the 2-day, 5-day, 13-day, and exclusion of the discharge summary scenarios, respectively. Notably, there is also a decrease in performance for the standard task, with the Micro-F1 score declining from 71.1 to 70.2.

	2 days	**5 days**	**13 days**	**excl. DS**	**all**
ELA	**46.2** ± 0.1	**50.9** ± 0.1	**53.6** ± 0.2	**54.3** ± 0.2	**71.1** ± 0.1
ELA (ablate rand. sampl.)	37.8 ± 0.2	42.5 ± 0.1	45.4 ± 0.1	46.3 ± 0.1	70.2 ± 0.2

Table 6.9: Comparison of Micro-F1 score on the development set of the model MMULA with the extended-length algorithm with and without performing random sampling during training. Results show the average and standard deviation across 3 replications.

These results are surprising since, theoretically, the last 16 chunks of the sequence, which include the discharge summary, should offer the most robust training signal. However, training with diverse random sequence augmentations proves to be more effective. This could be attributed to two factors. First, the random sampling process is likely to yield sub-sequences with a substantial number of documents unrelated to the discharge summary. Consequently, the model becomes better optimized for early predictions during training, a factor reflected in the outcomes for various temporal cut-offs. Second, the ablated model experiences a greater discrepancy between training and inference distributions. In training, the last chunks are consistently employed, while during inference, the entire sequence is utilized.

6.4 Test set evaluation

In this section the final experiments on the held-out test set are presented. These experiments aim to evaluate our temporal model, MMULA, across the three distinct set-up configurations. Additionally, the impact of multi-objective training and the extended-length algorithm on the test set is assessed. We select the second auxiliary task, specifically the non-ablated generation of the last document embedding, and the extended-length algorithm, without ablating the random sampling strategy. These configurations were determined to be optimal based on our experiments on development set.

The outcomes of the experiments conducted on the test set are presented in Table 6.10. The results indicate that the incorporation of the second auxiliary task enhances the Micro-F1 score by +0.6, +0.5, +0.6, and +0.6 for the 2-day, 5-day, 13-day, and exclusion of the discharge summary scenarios, respectively, compared to the "limited DS" set-up. Notably, the most remarkable improvement is realized by the implementation of the extended-length algorithm (ELA), yielding a Micro-F1 score increase of +6.5, +5.9, +4.9, and +4.9 for the respective temporal cut-offs. However, it is worth noting that the Micro-F1 score for the standard prediction task is -0.3 lower than the performance achieved using the MMULA model with the "latest" and "random" set-ups. This trend was also evident in the development set.

setup	**2 days**	**5 days**	**13 days**	**excl. DS**	**all**
MMULA (latest)	32.6 $_{\pm 0.1}$	36.1 $_{\pm 0.1}$	37.3 $_{\pm 0.2}$	36.8 $_{\pm 0.2}$	70.6 $_{\pm 0.2}$
MMULA (random)	28.8 $_{\pm 0.1}$	33.7 $_{\pm 0.1}$	36.6 $_{\pm 0.2}$	37.3 $_{\pm 0.3}$	**70.6** $_{\pm 0.2}$
MMULA (limited DS)	39.5 $_{\pm 0.1}$	44.4 $_{\pm 0.2}$	48.0 $_{\pm 0.2}$	48.6 $_{\pm 0.1}$	64.3 $_{\pm 0.2}$
MMULA + aux 2	40.1 $_{\pm 0.7}$	44.9 $_{\pm 0.7}$	48.6 $_{\pm 0.7}$	49.2 $_{\pm 0.7}$	64.1 $_{\pm 0.1}$
MMULA + ELA	**46.0** $_{\pm 0.2}$	**50.3** $_{\pm 0.1}$	**52.9** $_{\pm 0.2}$	**53.5** $_{\pm 0.2}$	**70.3** $_{\pm 0.1}$

Table 6.10: Micro-F1 score evaluated on held-out test set. The set-up configuration utilized for the MMULA + aux 2 experiments is the "Limited DS". The results show the mean and standard deviation across three replications.

In conclusion, both multi-objective training with the last document embedding generation and the extended-length algorithm exhibit substantial enhancements for early ICD-9 code prediction. Furthermore, applying the extended-length algorithm to the MMULA model is suitable for the task of predicting ICD-9 codes throughout a patient's stay. This yields a model with optimal early prediction performance, making it suitable for real-time disease and procedure tracking within a hospital setting, while maintaining performance close to optimal levels for the standard post-discharge task.

6.5 Model Interpretability

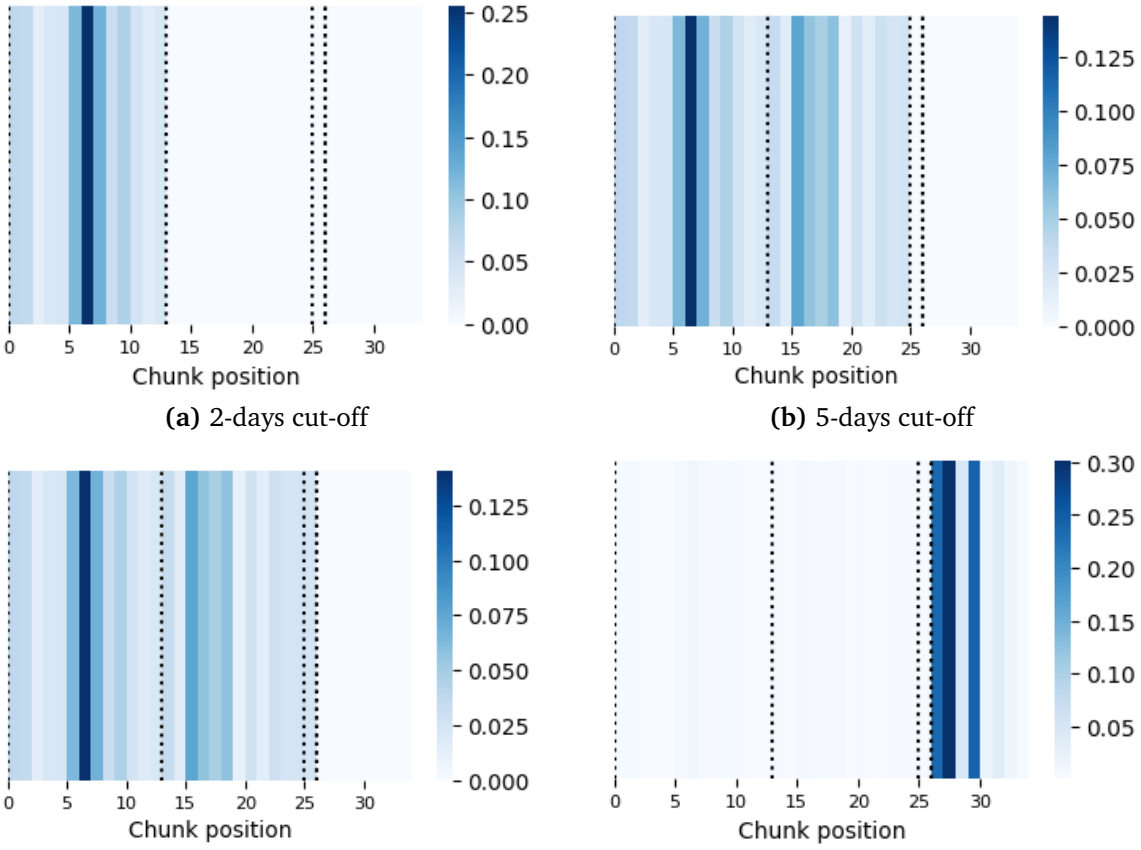
This section provides an overview of the model interpretability. Our model employs label attention, which provides an intrinsic explainability framework that can be used to understand which documents are more important for predicting ICD-9 codes at different points throughout the hospital stay. This is specially important in the clinical domain, where understanding the factors behind automated decisions is crucial. Moreover, it can also be used to aid the medical staff in discerning the documents that are relevant from the very lengthy clinical record of each patient.

Specifically, this section is organized in two parts. First, we show the attention patterns of the model when making predictions for individual patients at different points

in time. Secondly, we make a quantitative evaluation of the average attention weight across different note categories. This sheds light into the types of documents that are more useful to predict patient's diagnosis and procedures when the discharge summary is not available, and can be used to guide future research on the topic.

6.5.1 Explaining local predictions

In this section we present a visualization of the local model interpretability. Specifically, Fig. 6.1 shows the attention weights generated by the model MMULA when making predictions for a random patient and an associated label in the validation set at different temporal cut-offs. The attention weights correspond to the normalized scalar product between the chosen label embedding and each chunk embedding. A higher weight is associated with an increased relevance of the particular document to predicting a specific disease.



(a) 2-days cut-off

(b) 5-days cut-off

(c) 13-days cut-off

(d) Last day cut-off

Figure 6.1: Heat map depicting the attention weights versus chunk position for a random patient in the validation set. The various plots depict the attention pattern obtained when making predictions at different time cut-offs using the MMULA model in cooperation with the extended-length algorithm.

The visualization illustrates the masked attention mechanism employed within the

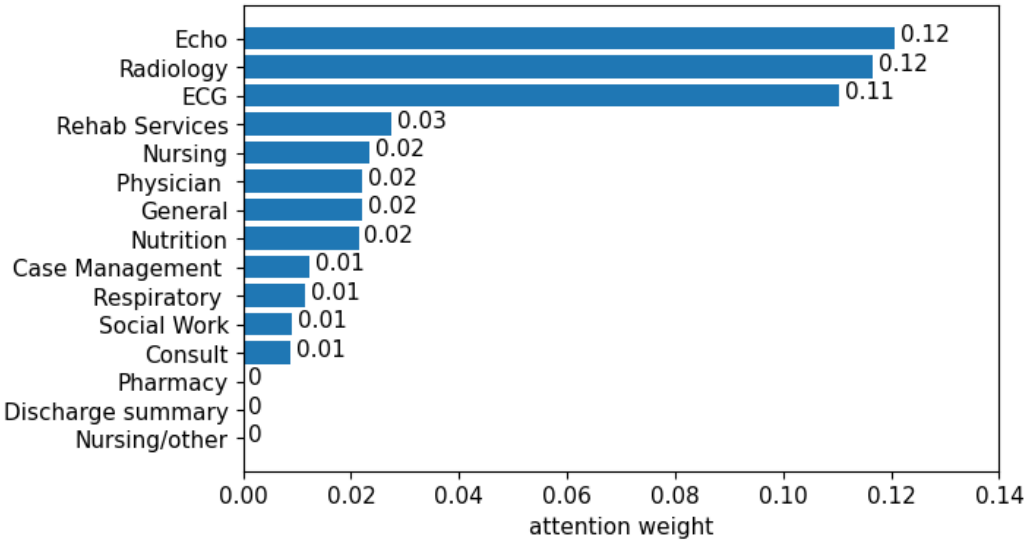
model, which operates in an "autoregressive" manner. The dashed lines represent the 2-days, 5-days and 13-days cut-offs. In Fig.6.1a it is evident that the model only focuses on the first 13 chunks when making a prediction at the 2-day cut-off. When increasing the temporal cut-off, posterior documents gain more relevance in the prediction. Finally, in the last day prediction the model is allowed to attend to the full sequence. The attention pattern of the last day scenario differs from the previous patterns, since the model focuses mostly on the discharge summary, while omitting the previous notes.

6.5.2 Attention across document categories

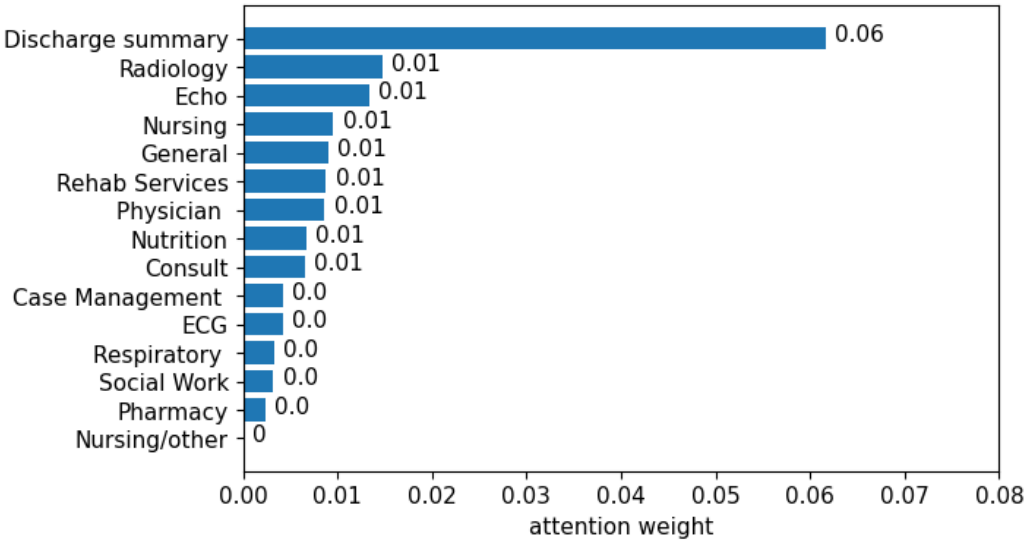
Finally, we analyze the importance of each note category for the temporal prediction of ICD-9 codes. Since our work is the first to explore the complete sequence of clinical notes to make ICD-9 code predictions throughout the hospital stay, it is important to interpret the learnings of the model. In particular, we are interested to see which notes are more useful to make early ICD-9 code predictions in the absence of the commonly used discharge summary. This could be used to guide future research on the topic, facilitating a more informed subsequence sampling process.

The results are shown in Fig. 6.2. When making predictions within the 2-day cut-off, the document types with higher average attention weight are echocardiography, radiology and electrocardiogram reports. On the other hand, when making predictions for the standard task of predicting the ICD-9 codes on the last day, the discharge summary has the highest average attention weight by a large margin from the rest of document types. This is followed by the radiology and echocardiography reports. However, as opposed to the results obtained within the 2-day cut-off, the electrocardiogram, or ECG, has a very low average attention weight for the last-day prediction scenario.

These results shed light on the types of reports that are valuable for making real-time prediction of ICD-9 codes. Within the 2-day cut-off, all the reports which have diagnostic characteristics receive the highest attention weights. Indeed, the echocardiography report is the description of a test which uses ultrasounds to identify abnormalities in the heart structure, and is used by cardiologists to diagnose heart diseases [44]. The radiology report is a record which details the result of imaging procedures such as X-rays and MRIs with the goal of diagnosing diseases [45]. Finally, the electrocardiogram test measures the electrical activity of the heart to identify cardiac conditions [41]. Therefore, all of these reports are highly technical and are specifically generated to assist physicians with diagnostic practices and, in the absence of the discharge summary, they are the most valuable document types to make early predictions of ICD-9 codes.



(a) 2-days cut-off



(b) Last day cut-off

Figure 6.2: Average attention weight per document type at different temporal cut-offs. The model used is MMULA in cooperation with the extended-length algorithm, to allow the consideration of the full EHR sequence.

Chapter 7

Conclusion and Future Works

7.1 Conclusion

The goal of this project was to address a gap in current ICD-9 coding research by developing a model capable of making accurate predictions throughout a patient's hospital stay, without relying solely on the discharge summary, and exploring several research avenues to enhance early prediction. The ultimate goal was to pave the way for real-time disease monitoring and procedure prediction, which could be used to improve hospital resource planning.

We have fulfilled the project's requirements and made several contributions to the field of ICD-9 coding. First, we have established a novel task on the MIMIC-III dataset concerning temporal ICD-9 code prediction, as well as a standardized evaluation framework tailored to the characteristics of our dataset to allow for model comparison. We have developed a temporal model architecture that allows for predictions to be made at any point during the hospital stay using the available notes, and we have evaluated the model employing various set-up configurations.

Additionally, we have shown that some auxiliary tasks can be used to enhance early ICD-9 code prediction. We have specified two distinct auxiliary tasks which introduce a supervision signal on the document representations employing domain-specific knowledge to improve the final task. Our experiments have shown that last document embedding generation is a particularly effective auxiliary task, and its integration was able to improve model performance significantly across early temporal cut-offs.

Moreover, we have explored another research avenue concerning the development a novel solution to the sequence length limitation problem faced by current hierarchical transformer models. The extended-length algorithm (ELA) enables to run inference based on a very extended text, while maintaining the same training time and memory requirements. The application of this algorithm allowed for predictions to be made considering the complete EHR sequence, and in consequence, the performance for early ICD-9 code prediction was substantially enhanced. After con-

ducting an ablation study the algorithm's random sampling process during training was found to be key in achieving superior performance, pointing toward potential avenues for further refinement. Moreover, this algorithm is problem-agnostic and has the potential to be applied to numerous real-world datasets which face similar length-related issues.

Finally, we have exploited the model's intrinsic interpretability framework to shed light into the learning process and identify the key documents that are valuable for early ICD-9 code prediction. Notably, the model was found to pay attention to reports with diagnostic nature when the discharge summary was missing. Moreover, this interpretability framework can be used to guide medical staff in the identification of key documents within the very lengthy clinical record sequence, guiding the diagnostic process and assisting with the time-consuming task of generating the final discharge summary document.

7.2 Limitations and future work

Throughout the project, we have identified several areas with the potential for further development to overcome current limitations.

Firstly, future research could focus on scaling up the model and corresponding experimentation. The project's primary focus was to establish a benchmark for a novel task and explore research directions aimed at enhancing early prediction and extending the sequence-length limitations of current models. Due to computational constraints, we were limited to running experiments using the [CLS]-token representation and a maximum length of 16 chunks. However, there is potential to scale up this work by retaining all token representations and increasing computational resources to allow for the allocation of 32 chunks or more. This expansion would facilitate a comparison with current state-of-the-art models.

Within the field of multi-objective training, we have identified a trade-off between enhancing early prediction and maintaining the performance of the standard ICD-9 code task. To address this, it may be valuable to explore additional loss functions for the auxiliary task of document embedding prediction. For instance, implementing a contrastive approach instead of the cosine embedding loss, or predicting a mean-pooled document embedding of the discharge summary instead of the last embedding, which could provide a less noisy training signal.

Furthermore, the extended-length algorithm has also demonstrated promising results in handling sequences of significant length. Ablation studies have revealed that the random sampling process within the training loop plays a key role in achieving superior performance. Further research into this algorithm is necessary to explore whether other sub-sampling methods could yield improved results, potentially surpassing the performance of the standard ICD-9 code last-day prediction task. For instance, one approach could involve calculating gradients of output scores with

respect to each document embedding and keeping only the embeddings with the largest gradient for the backward pass. Another approach could be to utilize the attention weights of the model to identify the most relevant documents to which the gradient needs to be propagated.

Lastly, the work undertaken in this project, particularly the extended-length algorithm, holds promise for application to numerous real-world datasets facing similar challenges posed by current transformer models. Datasets like the Arxiv Dataset, 20NewsGroup, or IMDB Review consist of lengthy text sequences paired with classification tasks and are currently being explored. Further research should delve into whether the newly developed extended-length algorithm can enhance performance on these datasets.

Bibliography

[1] World Health Organization, "International statistical classification of diseases and related health problems (ICD)." https://www.who.int/standards/classifications/classification-of-diseases. [Accessed 3rd August 2023].

[2] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein, "Explainable prediction of medical codes from clinical text," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018. doi:10.18653/v1/N18-1100.

[3] Imperial College London, "Legal, social, ethical and professional requirements." https://wiki.imperial.ac.uk/pages/viewpage.action?spaceKey=docteaching&title=Legal%2C+Social%2C+Ethical+and+Professional+Requirements. [Accessed 12th July 2023].

[4] A. E. W. Johnson, T. J. Pollard, L. Shen, L.-W. H. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "MIMIC-III, a freely accessible critical care database," *Scientific Data*, vol. 3, no. 160035, 2016. doi:10.1038/sdata.2016.35.

[5] A. Sonabend W, W. Cai, Y. Ahuja, A. Ananthakrishnan, Z. Xia, S. Yu, and C. Hong, "Automated ICD coding via unsupervised knowledge integration (UNITE)," *International journal of medical informatics, 139, 104135*, 2020. doi:10.1016/j.ijmedinf.2020.104135.

[6] B.-H. Kim, Z. Deng, P. S. Yu, and V. Ganapathi, "Can current explainability help provide references in clinical notes to support humans annotate medical codes?," *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis*, 2022. doi:10.48550/arXiv.2210.15882.

[7] P. B. Jensen, L. J. Jensen, and S. Brunak, "Mining electronic health records: towards better research applications and clinical care," *Nature reviews. Genetics*, vol. 13, pp. 395–405, Jun 01, 2012. doi:10.1038/nrg3208.

[8] M. Li, Z. Fei, M. Zeng, F.-X. Wu, Y. Li, Y. Pan, and J. Wang, "Automated ICD-9 Coding via A Deep Learning Approach," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 16, no. 4, pp. 1193–1202, 2019. doi:10.1109/TCBB.2018.2817488.

[9] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, vol. 3, pp. 111–132, 2022. doi:10.1016/j.aiopen.2022.10.001.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NIPS)*. doi:10.48550/arXiv.1706.03762.

[11] S. Takagi, Y. Yoshida, and M. Okada, "Impact of layer normalization on single-layer perceptron — statistical mechanical analysis," *Journal of the Physical Society of Japan*, vol. 88, p. 74003, Jul 15, 2019. doi:10.7566/JPSJ.88.074003.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016. doi:10.1109/CVPR.2016.90.

[13] C. S. Narendra Patwardhan, Stefano Marrone, "Transformers in the Real World: A Survey on NLP Applications ," 2023. doi:10.3390/info14040242.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *North American Association for Computational Linguistics (NAACL)*, 2019. doi:10.48550/arXiv.1810.04805.

[15] Z. Liu, W. Lin, Y. Shi, and J. Zhao, "A Robustly Optimized BERT Pre-training Approach with Post-training," *Chinese Computational Linguistics*, vol. 12869, pp. 471–484, 2021. doi:10.48550/arXiv.1907.11692.

[16] Z. Dong, T. Tang, L. Li, and W. X. Zhao, "A survey on long text modeling with transformers," 2023. ArXiv [PrePrint]. https://arxiv.org/abs/2302.14502.

[17] M. Ding, C. Zhou, H. Yang, and J. Tang, "CogLTX: Applying BERT to Long Texts," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[18] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," Apr 10, 2020. ArXiv [PrePrint]. https://arxiv.org/abs/2004.05150.

[19] M. Zaheer, G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed, "Big Bird: Transformers for Longer Sequences," *Neural Information Processing Systems (NeurIPS)*, Jul 28, 2020. doi:10.48550/arxiv.2007.14062.

[20] A. Roy, M. Saffar, A. Vaswani, and D. Grangier, "Efficient content-based sparse attention with routing transformers," *CoRR*, vol. abs/2003.05997, 2020. doi:10.48550/arXiv.2003.05997.

[21] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. doi:10.48550/arXiv.1901.02860.

[22] X. Dai, I. Chalkidis, S. Darkner, and D. Elliott, "Revisiting Transformer-based Models for Long Document Classification," *Findings of the Association for Computational Linguistics (EMNLP)*, Apr 2022. doi:10.48550/arxiv.2204.06683.

[23] F. Li and H. Yu, "ICD Coding from Clinical Text Using Multi-Filter Residual Convolutional Neural Network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, p. 8180, -04-03 2020.

[24] Y. Liu, H. Cheng, R. Klopfer, M. R. Gormley, and T. Schaaf, "Effective Convolutional Attention Network for Multi-label Clinical Document Classification," *Conference on Empirical Methods in Natural Language Processing*, pp. 5941–5953, 2021. doi:10.18653/v1/2021.emnlp-main.481.

[25] T. Vu, D. Nguyen, and A. Nguyen, "A label attention model for ICD coding from clinical text," *In Proceedings of IJCAI*, Jul 13, 2020. doi:10.24963/ijcai.2020/461.

[26] Z. Yuan, C. Tan, and S. Huang, "Code Synonyms Do Matter: Multiple Synonyms Matching Network for Automatic ICD Coding," *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, p. 808. doi:10.18653/v1/2022.acl-short.91.

[27] S. Ji, M. Hölttä, and P. Marttinen, "Does the magic of BERT apply to medical code assignment? A quantitative study," *Computers in biology and medicine*, vol. 139, p. 104998, Dec 01, 2021. doi:10.1016/j.compbiomed.2021.104998.

[28] C. B. L. Ng, D. Santos, and M. Rei, "Modelling Temporal Document Sequences for Clinical ICD Coding," *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Feb 24, 2023. doi:10.48550/arxiv.2302.12666.

[29] H. Y. D. K. S. Kim, "An Automatic ICD Coding Network Using Partition-Based Label Attention ," *SSRN Electronic Journal*, 2022. doi:10.2139/ssrn.4255745.

[30] L. Liu, O. Perez-Concha, A. Nguyen, V. Bennett, and L. Jorm, "Hierarchical label-wise attention transformer model for explainable icd coding," *Journal of biomedical informatics*, vol. 133, p. 104161, Sep 01, 2022.

[31] 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*, vol. 3, pp. 1–23, Jan 31, 2022. doi:10.1145/3458754.

[32] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: generalized autoregressive pretraining for language understanding," *Neural Information Processing Systems*, Jun 19, 2019. doi:10.48550/arXiv.1906.08237.

[33] K. Huang, J. Altosaar, and R. Ranganath, "ClinicalBERT: modeling clinical notes and predicting hospital readmission," Apr 10, 2019.

[34] L. M. C. E. Koyena Pal, Seyed Ali Bahrainian, "Neural summarization of electronic health records," 2023. ArXiv [PrePrint]. https://arxiv.org/pdf/2305.15222.pdf.

[35] J. Mullenbach, Y. Pruksachatkun, S. Adler, J. Seale, J. Swartz, G. G. Mckelvey, H. H. Dai, Y. Yang, and D. Sontag, "CLIP: A Dataset for Extracting Action Items for Physicians from Hospital Discharge Notes," *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021. doi:10.18653/v1/2021.acl-long.109.

[36] M. L. Yang Liu, "Text Summarization with Pretrained Encoders," *EMNLP*, Mar 25, 2019. doi:10.48550/arxiv.1903.10318.

[37] P. J. Liu, "Learning to Write Notes in Electronic Health Records," *Proceedings of Machine Learning Research*, vol. 85, Aug 08, 2018. doi:10.48550/arxiv.1808.02622.

[38] V. S. Vit Skvara, Tomas Pevny, "Is AUC the best measure for practical comparison of anomaly detectors?," 2023. ArXiv [PrePrint] https://arxiv.org/pdf/2305.04754.pdf.

[39] 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," *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, p. 146, -11-19 2020. doi:10.18653/v1/2020.clinicalnlp-1.17.

[40] S. Toshniwal, H. Tang, L. Lu, and K. Livescu, "Multitask learning with low-level auxiliary tasks for encoder-decoder based speech recognition," *Interspeech 2017*, -08-20 2017. doi:10.21437/interspeech.2017-1118.

[41] John Hopkins Medicine, "Electrocardiogram." "`https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/electrocardiogram`.

[42] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss," pp. 1360–1369, IEEE, Mar 2020. doi:10.1109/WACV45572.2020.9093286.

[43] Pytorch, "Cosine embedding loss." `https://pytorch.org/docs/stable/generated/torch.nn.CosineEmbeddingLoss.html`.

[44] P. N. Van, H. P. Huy, and L. T. Quoc, "Echocardiography segmentation using neural ode-based diffeomorphic registration field," *IEEE Transactions On Medical Imaging*, vol. XX, Jun 16, 2023. doi:10.48550/arxiv.2306.09687.

[45] M. Alarifi, T. Patrick, A. Jabour, M. Wu, and J. Luo, "Understanding patient needs and gaps in radiology reports through online discussion forum analysis," *Insights into imaging*, vol. 12, p. 50, Apr 19, 2021. doi:10.1186/s13244-020-00930-2.