THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

COLLEGE OF INFORMATION SCIENCES AND TECHNOLOGY

SeleHiTANet: Boosting Health Risk Prediction via Data Selection on Hierarchical Time-aware Attention Networks

Sirui Qi
SUMMER 2023

A thesis
submitted in partial fulfillment
of the requirements
for baccalaureate degrees
in Computer Science and Mathematics
with honors in Information Sciences and Technology

Reviewed and approved* by the following:

Fenglong Ma
Assistant Professor
Thesis Supervisor

Nicklaus A. Giacobe
Associate Teaching Professor
Honors Adviser

*Signatures are on file in the Schreyer Honors College.

Abstract

At present, due to today's high medical costs and a shortage of medical personnel, there is a great demand for sickness prediction in the healthcare field. Currently, transformers-based methods were early successful in sick prediction. Hierarchical Time-aware Attention Network(HiTANet) is a model used to predict the health situation of the patients base on the visiting information and visiting times at local and global stages, which simulate how doctors' making decision in risk prediction. We make a model called SeleHiTANet that improves the traditional model by automatically ruling out irrelevant visits and codes by effectively skimming the electronic health records (EHRs) data. In SeleHiTANet, we used parts of the model in MedSkim to be the ICD-9 code selection mechanism. MedSkim proved that making visits and codes decision in model input could improve the model's performance in predicting the diagnosis. This improvement can help healthcare models to focus on the most important information in the EHRs, improving the accuracy of diagnoses and treatment plans. By utilizing advanced algorithms, our new model will be able to quickly and efficiently scan through large amounts of EHRs data, saving time and reducing the risk of human error. We evaluate the performance of the SeleHiTANet model on EHRs and try to show whether it outperforms the original model in terms of risk prediction accuracy. SeleHiTANet model incorporates the ICD-9 code selection mechanism into HiTANet. In our experiments, we used Accuracy (Acc), Precision (Pre), Recall, F1, and the Area Under Curve (Auc) scores as the evaluation metrics. Experimental results show that the proposed SeleHiTANet model outperforms baselines and successfully improves the performance of the transformer-based models for the health risk prediction task.

Table of Contents

List of Figures

List of Tables

Acknowledgements

Thanks to the guidance of Prof. Fenglong Ma, who offered the direction and the idea for this research.

Chapter 1 Introduction

1.1 Background

The development of transformer-based models revolutionizes how we make healthcare predictions using patients' data. These models, like HiTANet [1], have provided promising early results in risk prediction tasks. They combined the historical data of the patient's health records with temporal information to predict the likelihood of patients falling sick in the future. Based on the transformer-based models, which excel at processing sequential data, we can analyze patients' historical health records and establish patterns over time. Unlike traditional models that might disregard the time order of health events, these models can effectively collect information on the temporal dependencies between different health incidents and use them to offer a better risk assessment on health condition prediction. Because of the ensuring data privacy and security, we cannot access the data like the patients' age for prediction and training. So transformer-based models [2] should only consider public data like the electronic health records (EHRs) [3] data with the ICD codes [4], which means we should pay attention to the inherent complexity and noisiness of EHRs [3] data. Also, when we design the model, because of the limitation of the types of data, we should increase the utilization of the data as much as possible because fewer types of input data would make the models hard to get better results. The future of healthcare prediction will likely continue to be built by the development of transformer-based models when we continue to refine these techniques and navigate these challenges.

1.2 Motivation

Commentary, the International Classification of Diseases (ICD) codes [4] play an important role in recording the patients' sick information. They include most possible diagnoses that doctors can write and most situations that people know. We can get that most of the doctors making the sickness prediction are based on the previous situation of the patients, which are included in the visiting data written by the ICD-9 codes [4], which means we can make the prediction based on the standardized medical classification codes. These standardized medical classification codes capture and record the health status and the diseases or health conditions of the patients. By integrating these codes with EHRs, researchers and healthcare providers are able to draw a more comprehensive picture of the patient's health history. As shown in Figure 1.1, a patient's EHR and the respective ICD [4] codes are represented.

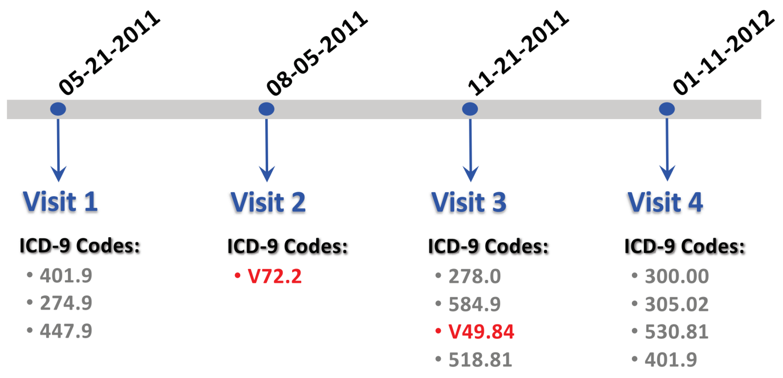


Figure 1.1: An example of a patient's icd-9 code data who will have heart failure in the future [5]. The EHR data is time-ordered that includes four visits. Each time of visit records some diagnosis codes that the doctor finds in one visit. Red codes mean useless codes and gray codes mean normal codes.

Besides, we should know that EHR data can include some noisy data, and not all ICD codes in a patient's record may be pertinent to the prediction task at hand. For instance, in Figure 1.1, the codes ”V49.84”, which signifies bed confinement status [4], and ”V72.2”, indicating a dental examination [4], may not be directly relevant when predicting something like heart failure. When doctors consult patients, they filter out irrelevant diagnoses to ensure they're not misled by ”noise” data. This ”noise” refers to diagnoses or information that do not contribute to understanding the patient's current condition or predicting future health risks [6]. On the other hand, our model also mirrors doctors' approach to conducting diagnoses, and we should consider this kind of doctors' actions in code selection.

As such, there is a growing understanding that effectively filtering out these ”noise” codes could potentially enhance the performance of predictive models. On one side, we can use medical knowledge to identify and remove such unrelated data of the patients to achieve this. Otherwise, selection algorithms or attention mechanisms can help to automatically pay attention to the most predictive elements within the EHR data and remove them, which can offer better and more efficient models by removing useless and interfering information. Figure 1 shows one of the possible situations of this kind of thing with an example where visit 3 contains the ”V49.84” code and visit 2 includes the ”V72.2” code. Based on the definition and medical knowledge, we can find out that these codes are useless for making the prediction. So omitting these codes, which are not associated with heart failure, can make these kinds of models' predictive focus on more relevant information.

However, it is worth noting that while filtering out such codes might improve the model's predictive performance for specific tasks, it is essential to consider the potential implications of removing such information. For example, while bed confinement status ”V49.84” or a dental examination ”V72.2” may not directly relate to heart failure based on medical knowledge, it may also indirectly influence the overall health situation of the patients. Therefore, the design of the selection mechanism should be careful when deciding which codes to include or exclude in the predictive models.

Specifically, the code selection mechanism is working for removing non-relevant data codes compared with the sick, which we want to predict from the input EHR data with Gumbel-Softmax [7], which code should be skipped. We choose the Gumble-Softmax [7] for making ICD-9 code selection for sickness prediction here because it offers a continuous and differentiable approximation of the discrete argmax operation, which can make it feasible to incorporate into gradient-based learning models. By adding the Gumbel-Softmax function into the models that process the EHRs for making the predictions, it can learn to select or not select certain codes based on their relevance to the prediction target sicks. The model effectively learns to assign more influence to the more relevant codes and no influence to the non-relative codes. This approach allows the model to focus on the most important parts of the data, potentially leading to improved predictive performance. The Gumble-softmax [7] function lets the model easily collect the distinctions and complexities between different diagnoses in sickness prediction using ICD-9 codes while maintaining robustness and flexibility.

Except for the code selection, we should also consider the time information of each visit. Currently, most studies focus on discerning the influence of time from longitudinal patient data and decay the patients' historical visiting information by time information [8, 1]. However, the time between visits and the sequence of events in healthcare can provide significant insights into disease progression and patient outcomes. Recognizing this, we use the transformer-based model HiTANet [1] to embed the time information and code information into hidden representations and then combine these to learn local attention weights.

Using HiTANet [1], we can model each patient's journey through their healthcare encounters, accounting for the temporal relationships between each visit and their respective ICD-9 codes [4]. This enables the model to grasp the patterns in the temporal progression of a patient's condition to increase disease risk prediction accuracy. Importantly, we aspire for our model to simulate the diagnostic process undertaken by doctors. Physicians, when diagnosing, consider not only treating each visit in isolation but also the patient's overall medical history. Our use of global and local attention weights mirrors this holistic approach. Furthermore, HiTANet provides an overall representation from the time-aware transformer, which encapsulates all the time-aware visit embedding that includes both visit and time information.

We leverage this overall representation in a global synthesis stage, integrating time information to derive global attention weights. We got this kind of attention weight based on the overall meaningful visiting data of the patients and the time information. These weights offer a comprehensive, longitudinal view of the patient's health status. By merging local and global attention weights, we create a more comprehensive and workable patient representation for disease risk prediction, similar to a doctor conducting a diagnosis. This dual-attention mechanism allows our model to accommodate the intricate interactions among the visits and between the ICD-9 codes [4] while combining local and global temporal dynamics, increasing its ability to predict disease risk accurately.

1.3 Our Contributions

Our models have the following technical contributions:

• We have adapted a time-aware Transformer model called HiTANet [1] combined with a code selection mechanism which is based on Gumbel-Softmax [7] with filter out redundant data. This process is executed in stages: initially, our model selects relevant information from the visiting data, followed by embedding the time information into this selected visit data. Subsequently, a local attention weight is learned for each relevant visit. Then we calculate a global attention weight with the patient's overall situation and time information. All of the procession in the model focuses only on the significant visiting data, thereby reducing computational load and improving efficiency.

• An integral part of our contribution is our model's performance. After being anxiously tested on the different datasets, including the Heart Failure, Amnesia, Kidney disease, Dementia, and COPD datasets, our model demonstrated substantial improvements over traditional baselines like SVM [9] and LSTM [10]. This kind of promotion of the result validates the effectiveness of the code selection mechanism in enhancing the Transformer-based model's prediction capabilities.

• Moreover, by employing the code selection mechanism in conjunction with a time-aware Transformer, our model can capture intricate interactions among different visits and ICD-9 codes [4] in a patient's medical history. The code selection mechanism helps to prioritize important ICD-9 codes [4], and the time-aware Transformer effectively incorporates the temporal information with the visiting data. This unique blend of techniques offers a more nuanced approach to disease risk prediction.

• Finally, our model goes beyond mere prediction and strives to simulate the physician's diagnostic process. It takes a holistic view of patient history, integrating local and global attention weights to represent the patient's health status comprehensively. This mimics how doctors consider a patient's entire health history, not just isolated visits when making a diagnosis. Also, when the doctor makes the diagnosis, they would not consider the useless situation in the previous doctor visit. In doing so, our model bridges the gap between artificial intelligence and human healthcare decision-making, potentially paving the way for future more advanced AI applications in healthcare.

1.4 Summary

In the backdrop of advancements in transformer-based models and their use in healthcare predictions, a new model has been proposed by us that uses the time-aware Transformer model HiTANet [1] and a code selection mechanism based on Gumbel-Softmax [7]. The model analyzes historical health records based on the ICD9 [4] codes, effectively capturing temporal dependencies between health incidents and embedding it with the visiting data to analyze for gaining insights into patient health history, which are working for predicting the patients' sicks.

However, the model's design is mindful of the inherent complexity and noisiness of Electronic Health Records (EHRs) data. The approach includes a selection mechanism that effectively filters out irrelevant ICD9 [4] codes which means useless visiting data in the patients' visiting history, improving the predictive performance of the model. The selection mechanism uses the Gumbel-Softmax function to determine the relevance of ICD codes for disease risk prediction, assigning more influence to more relevant codes.

In addition to code selection, the model considers the time information of each visit using HiTANet, creating a more comprehensive patient representation. It learns local attention weights for each relevant visit and calculates global weights considering patients' overall situation and time information. It simulates the doctors' consulting with the patients. Our model's blend of techniques allows it to capture intricate interactions among different visits and ICD-9 [4] codes, offering a better approach to disease risk prediction than traditional models.

The main contributions of this model are the adaptation of the time-aware Transformer model combined with a code selection mechanism and its performance improvement over traditional models, validated through testing on several datasets. Lastly, the model strives to simulate the physician's diagnostic process, taking a holistic view of patient history. This approach bridges the gap between artificial intelligence and human healthcare decision-making, indicating a promising future for AI applications in healthcare.

Chapter 2 Related Work

2.1 Electronic Health Records

Electronic Health Records (EHRs) [3] are digital versions of patients' records in clinician offices, hospitals, and other healthcare settings. They contain comprehensive patient medical history, including diagnoses, medical test results, laboratory results, clinical imaging, physician notes, etc.[11] In most cases, the data in electronic health records are not made public because most people still don't want their privacy released [12]. However, some non-sensitive parts of a patient's data will be made public with their permission to promote academia. This approach is to prevent the patient's situation from being tracked through EHRs, and it is also a method to protect patients while promoting the development of medical academia. EHRs are designed to be accessed by authorized healthcare providers and staff from various healthcare settings, providing an integrated, real-time, patient-centered record that aids in making informed care decisions. [3] This system can lead to improved patient care by enhancing efficiency, quality, safety, and coordination. [13]

2.2 Health Risk Prediction

This type of prediction is extremely important in healthcare because it can allow for early interventions and preventive measures, which can greatly improve a person's prognosis and potentially prevent the onset of serious illnesses. [14] On the other hand, the better result of the prediction would also offer people better thinks about the However, making incorrect sickness predictions can lead to disastrous consequences. In some respects, the human body is fragile. Incorrect predictions can lead to wrong decisions, such as taking the wrong medication or missing the optimal treatment time. These situations can cause significant, and even irreversible, damage to the human body. Sickness prediction is not a game; its margin for error is very slim. Moreover, we cannot predict the impact of wrong decisions due to the complexity of the human body. The doctors' ability is an important factor influencing us to determine whether the sicks are the right prediction. Moreover, not all trained doctors are capable of treating all diseases. Nowadays, the cost of training doctors is so high that there is a shortage of doctors in most places [15]. In underdeveloped and even some developed areas, the lack of doctors results in many people being unable to receive the necessary diagnosis.

In artificial intelligence and machine learning, health risk prediction usually involves training a model on a large dataset of health records to recognize patterns or correlations that indicate an increased risk of certain diseases. For instance, Prof Ma designed a diagnosis prediction framework incorporating diagnosis code descriptions via convolutional neural networks to learn meaningful code representations from patients' information. [16]. These complex models may consider various factors, including temporal trends and interactions between different health conditions. For example, a health risk prediction model could be used to determine the individuals with a high risk of developing heart disease based on various clinical records [17]. The model would be trained on a large dataset of patient records and then used to predict the risk for new patients based on their characteristics.

However, not all of the dataset currently offers the detail of each patient, like age, blood pressure, cholesterol levels, and smoking status [18]. Because of the lack of data and the limitation of the data type, it is important to make predictions based on the limited data with limited kinds. When designing the models making the sickness prediction, we should ensure that all data necessary for the training and testing are not hard to get.

In fact, with the rising cost of healthcare, the importance of sickness prediction is also correspondingly heightened. Accurate sickness prediction can simultaneously save patients' anticipated costs in various aspects, such as deferred treatment fees and additional laboratory fees, optimizing the allocation of medical resources. [19]. Also, preventing unnecessary medical tests, hospitalizations, and surgeries, can save both resources and time. In addition, accurate predictions could also allow for more personalized and effective treatment plans, improving overall patient care and outcomes [19].

2.3 ICD-9 Codes

The International Classification of Diseases, Ninth Revision (ICD-9) [4] is a system of coding created by the World Health Organization (WHO). It is a kind of official system which arrange the codes to different diagnoses and procedures [4]. It's used to classify different diseases and signs, symptoms, special findings, diagnoses, and things that can cause injury or diseases.

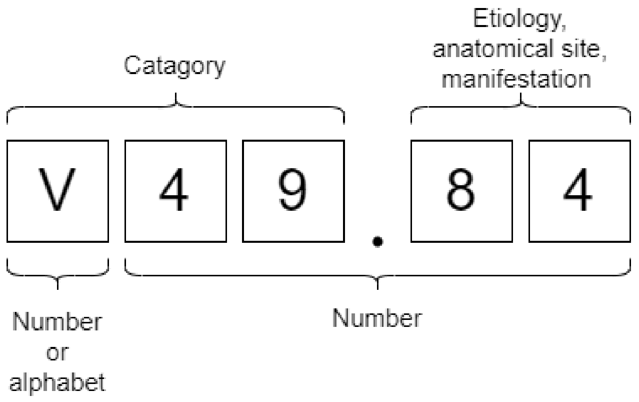


Figure 2.1: An example of the ICD-9 code structure.

The ICD-9 has a table with a number list for different diseases, a disease list in ABC order, and a system to sort surgeries, tests, and treatments [4]. Figure 2.1 shows the structure of the ICD-9 codes [4].

These ICD-9 codes [4] are used in healthcare settings, such as hospitals and doctors' offices, to represent patient diagnoses and procedures performed. They are important for accurate medical record keeping and billing, as they describe the exact nature of a patient's diagnosis or the procedures performed during a hospital stay. Each disease, disorder, or symptom has a specific ICD-9 code, allowing easy communication and data collection between healthcare providers, researchers, and insurance companies, regardless of language or specific naming conventions.

In data science and machine learning, the ICD-9 code [4] also holds a significant position. As a standard for quantifying diagnostic results, the ICD-9 [4] enables operability in data. The ICD-9 code [4] also provides a basic categorization for diseases; in Figure 2.1, we can see that the first three numbers and alphabets mean the categories. Based on the ICD-9 code [4], we can infer if there are fundamental relationships between different diagnoses. This principle allows for simple analysis and classification of data. The quantifiable nature of the ICD-9 code [4] makes it easy to store and process data. The existence of the ICD-9 code [4] means that doctors' diagnoses no longer require language model analysis, improving the accuracy of machine learning as a sickness prediction tool and reducing time consumption.

2.4 Code Selection Mechanism

The Code Selection Mechanism is a mechanism used to select specific data. Each disease will have more or less symptoms appearing beforehand. To standardize cases, we use codes [4] for representation. Here, the ’codes’ often refer to the International Classification of Diseases-9 (ICD-9) codes [4] used in electronic health records (EHRs). While EHRs contain a wealth of information, not all of it is necessarily pertinent to every prediction task. For example, a patient’s record may contain ICD codes related to a previous dental examination or bed confinement status. These codes may not be directly relevant when predicting heart failure risk. This is where the code selection mechanism comes into play.

Just as doctors ignore irrelevant diseases when diagnosing, we can also consider disregarding meaningless ICD-9 codes when designing our model [4]. The code selection mechanism works for removing and filters out the less relevant or ”noisy” ones based on the prediction of the target sicks, improving the model’s predictive performance [20, 21]. There are different techniques for code selection, including selection algorithms, attention mechanisms, and methods like the Gumbel-Softmax [7].

An instance of this is seen in the Medskim model [5], which uses Gumbel-Softmax [7] over a one-layer feedforward network approach that allows the model to automatically identify important encodings relevant to a particularly sick and focus on these encodings to improve predictive performance. Consequently, this leads to higher predictive accuracy, making the Medskim [5] model more effective and reliable for medical risk prediction tasks.

A good code selection mechanism should try to balance removing irrelevant information and potentially meaningful health indicators. Due to the lack of human knowledge in healthcare and medicine, we still don’t know all kinds of reasons which can cause each kind of sickness. While certain codes may not seem directly relevant, they could indirectly influence a patient’s overall health situation, and we may don’t know about it. Because of that, they may still carry some predictive value.

2.5 Attention Mechanism

In machine learning, attention is a technique that helps models focus on specific relevant input parts when processing data. It is a way to add weights to input data that help the model pay more attention to important parts and pay less attention to other parts. Attention-based models are mostly mainly to find the weight for the visit data and get the prediction of the patients by using a weighted sum operation.

The concept of attention in machine learning simulates human visual attention. This mechanism makes humans dynamically focus on a specific subset of information while processing more significant amounts. It is just like people watching the images and reading the articles, always focusing on the important part of what they want. Similarly, attention models in machine learning allow the algorithm to focus on important parts of the input data while reducing the focus on less relevant parts [22].

In a sequence-to-sequence task, for example, attention mechanisms can be used to weigh the importance of each input when predicting each output. This is particularly useful in tasks such as machine translation, where certain output words (in the translated sentence) are more closely related to certain input words (in the source sentence) than others [23]. In the computer vision field, the attention mechanism weights each pixel in an image in different ways based on the training images so that during testing, it can focus the analysis on important (high-weighted) areas, similar to how humans view images [24]. For instance, when analyzing images of handwritten digits, the attention mechanism allows the model to more easily ignore peripheral parts and concentrate its attention on the central, more critical parts [24].

In healthcare predictive modeling, attention mechanisms can help determine which visit data of the patients are most relevant to predicting the target sick and which are less [25, 26]. Then the model can use the attention it got for prediction. This helps to improve model performance by allowing it to focus on crucial information.

In our model, we are trying to simulate a doctor's actions. SeleHiTANet provides local attention, similar to when a doctor examines a patient's case, reviewing each instance of the illness and seeking similar cases based on experience. At the same time, it provides global attention akin to a doctor analyzing the patient's overall condition. Combining these two types of attention is similar to a doctor examining a patient by considering both the overall condition and each check-up situation to arrive at a final diagnosis.

2.6 Transformer

The Transformer is a neural network architecture with a self-attention mechanism [2]. It can consider the entire sequence of tokens at once and weigh their impact differently when predicting each output. The transformer allows the model to capture complex dependencies between words or tokens regardless of their distance in the sequence.

Traditionally, sequence processing relies on recurrence like Recurrent Neural Networks [27] that should be based on the previous and cannot run parallelly because RNN [27] needs to run with the previous status and cannot do the parallel calculations in GPU with multiple cores. Compared with RNN [27], the Transformer-based model allows parallel computation and captures global dependencies in the data, leading to more efficient learning and better performance on tasks like machine translation, text summarizing, and more [2]. Moreover, since transformer-based models do not have the problem of the input sequence, they do not suffer from the issue of gradient vanishing as RNNs [27] do, which can significantly reduce the loss of important information [2].

In its original form, the Transformer model build by an encoder and a decoder, each composed of a stack of identical layers containing self-attention and feed-forward neural network components [2]. Since its introduction, variations of the Transformer architecture have been developed for various tasks in healthcare and other domains like computer vision and NLP [28]. It mainly works on the NLP and is still less used in healthcare [29, 30, 31]. In healthcare, Transformer models can be used to process EHRs to predict disease risk, where the model can learn to attend to critical past medical events and pay less attention to irrelevant data.

2.7 Gumble-softmax

The Gumbel-Softmax is an important tool in machine learning for representing distributions encountered in unsupervised learning, language modeling, attention mechanisms, and reinforcement learning domains [7]. It also provides a means for a continuous and differentiable approximation of a sample derived from a discrete distribution. This aspect makes it particularly valuable in the sphere of deep learning, where differentiable functions are often the mainstay.

The foundation of the Gumbel-Softmax estimator [7] lies in the Gumbel-Max trick [32], a technique devised to generate samples from a discrete distribution. However, the Gumbel-Max trick [32] has limitations because of its non-differentiable nature. Because of that, the softmax function was used as a continuous, differentiable approximation to $\arg\max$ for solve this problem [7, 32]. It means the Gumbel-Softmax was built as a differentiable version that can offer a smoothed approximation of the Gumbel-Max trick. Compared to Gumbel-Max, Gumbel-Softmax can obtain effective estimations by sampling a number of points without losing gradient information.

The Gumbel-Max trick offer a good way [7] to get samples z from a categorical distribution with class probabilities π :

$$z = \text{one\_hot}(\arg\max_i [g_i + \log \pi_i])$$

where $g_1 \dots g_k$ are i.i.d samples drawn from $\text{Gumbel}(0,1)^1$. It [7] uses the softmax function as a continuous, differentiable approximation to $\arg\max$ and generates k-dimensional sample vectors $y \in \Delta^{k-1}$ where

$$y_i = \frac{\exp\left(\left(\log\left(\pi_i\right) + g_i\right)/\tau\right)}{\sum_{j=1}^{k} \exp\left(\left(\log\left(\pi_j\right) + g_j\right)/\tau\right)} \quad \text{for } i = 1, \dots, k.$$

The density of the Gumbel-Softmax distribution [7] is:

$$p_{\pi,\tau}\left(y_1, \dots, y_k\right) = \Gamma(k)\tau^{k-1}\left(\sum_{i=1}^{k} \pi_i / y_i^{\tau}\right)^{-k} \prod_{i=1}^{k}\left(\pi_i / y_i^{\tau+1}\right)$$

This characteristic of differentiability is essential when the goal is to backpropagate through samples drawn from a discrete distribution. It finds relevance in diverse applications, such as Reinforcement Learning, where it facilitates the selection of actions, or in Natural Language Processing, where it aids in determining the sequence of words to generate. Based on the experiment, Gumbel-Softmax can be used to efficiently train semi-supervised models [33, 7] without costly marginalization over unobserved categorical latent variables. It experimentally shows that Gumbel-Softmax outperforms all single-sample gradient estimators on both Bernoulli variables and categorical variables [7].

In a healthcare setting, particularly within machine learning models, the Gumbel-Softmax function proves instrumental in guiding the selection process. It can decide whether the data with some features, for instance, in our model, ICD-9 codes, should be included or excluded based on their relevance to the task of prediction. Exactly, we are using the Gumbel-Softmax estimator [7] as an important part of the code selection mechanism in the design of our model, which was used in MedSkim [5] model before.

Remarkably, it achieves this while still allowing for the propagation of gradients through the selection process. This feature is essential for training the model via methods that rely on gradient-based optimization. Overall, the Gumbel-Softmax estimator significantly contributes to building robust, efficient, and trustworthy predictive models in healthcare. It also helps the model do effective estimations on classifying decisions of the model's input data.

2.8 Time-aware Models

Time-aware models represent an innovative class of predictive tools that incorporate temporal information to enhance the accuracy and depth of their predictions. It allows us to define how models change over time, which makes them highly valuable in various fields, including finance and healthcare. Most existing models do not consider the influence of time on the data. Although they consider time, traditional Time-aware models operate directly using time decay functions [34, 8, 35]. However, in sickness prediction, the impact of time on the case is extremely important. Indeed, the same diagnosis made at different times could lead to vastly different final disease conditions. Therefore, when we designed SeleHiTANet, we also considered this issue, just as we did when designing HiTANet [1]. We added time embedding into the visit data so that the model would consider time information while contemplating the case.

In essence, time-aware models weave the temporal dimension into the models' structure, giving them a richer understanding of the data and more accurate prediction. Adding time into the mix doesn't just give us a better understanding of how things change and lets us model processes that depend on time. So that we can track trends and shifts that happen over different periods, making our predictions even better.

2.9 Summary

Electronic Health Records [3] mainly offer the data for our model's training and testing, which collect from the real world. Health risk prediction involves analyzing various data, like medical histories, lifestyle habits, genetic information, and environmental factors, to anticipate future health problems. These predictions help facilitate early interventions, optimize medical resource allocation, and enable personalized treatment plans, improving patient outcomes. However, inaccurate predictions can have harmful consequences like incorrect medication usage or missed treatment opportunities. Implementing machine learning and artificial intelligence allows for more sophisticated health risk predictions. Models are trained on large datasets of health records to recognize patterns indicating disease risks, although these models may be limited by data availability and complexity.

The ICD-9 codes [4], which the World Health Organization created, assign codes to diagnoses and procedures for standardization of communication among healthcare providers and data analysis. The code selection mechanism filters out less relevant codes to improve the predictive performance of machine learning models. Attention-based models in machine learning helps identify the most relevant patient visit data for prediction, and the Transformer model allows for efficient learning and better performance by capturing complex dependencies between data. Currently, transformer-based models become a part of the revolution in healthcare prediction. Based on the model HiTANet [1] and EHRs data, we can find out that it is early successes in risk prediction tasks, which use historical EHRs data combined with the time data to predict the future sick of patients [1]. In the healthcare setting, Gumbel-Softmax plays a crucial role in the selection process of models, allowing for the propagation of gradients through the selection process. Finally, time-aware models incorporate the significant impact of time on health predictions into the design, leading to more accurate outcomes.

Chapter 3 Methodology

3.1 Task Definition

Exactly, the task of the SeleHiTANet model is trying to find a better way to make the sick prediction compared with the original models. In EHR data, all patients' data are recorded as a list that includes the diagnosis with one or more ICD-9 codes each time. So we use math way, which can run with the model easily to define the tasks of prediction first [1]. All of the definitions of different kinds of input are these [1]:

Definition 1 (Diagnosis Codes). We use $C = \{c_1, c_2, \cdots, c_N\}$ to represent all ICD-9 [4] diagnosis codes in here, each c_i means that the patient have or not the i-th diagnosis in each historical visit. If it is 1, the patient has the i-th diagnosis at that historical visit; otherwise, it does not. And we make a variable c_* in each visit data representing the overall diagnosis [1]. It would be added to the end of visit data of each visiting [1]. For each c_*, all of them in each visit are initially set to 0 except $\mathbf{x}_{*t}$. N is the size of C which is the size of ICD-9 [4] set include all of the diagnosis codes.

Definition 2 (Binary EHR Data). We also make matrix which is $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T, \mathbf{x}_*]$ to represent all of the visit information of one patient [1]. In here, based on input of the data [1], we designed the t-th visit data of the patient which is $\mathbf{x}_t \in \{0, 1\}^{N+1}$. Each visit t-th is a binary vectorİf the patient have i-th diagnosis of the ICD-9 code which $c_i \in \{c_1, \cdots, c_N\}$, then $\mathbf{x}_{ti} = 1$, otherwise $\mathbf{x}_{ti} = 0$. We also make$\mathbf{x}_* \in \mathbb{R}^{N+1}$ become the symbol of the overall situation of the patients [1]. It is a one-hot vector [1], and exactly we should not care about this value initially because it would change to the same value in the future modeling based on the patients' data [1]. $\mathbf{x}_*$ only have the special code $c_* = 1$ and other codes are 0, which means $\mathbf{x}_{t*} = \mathbf{x}_{*t} = 0$. We set all the patients have the same initial $\mathbf{x}_*$ [1].

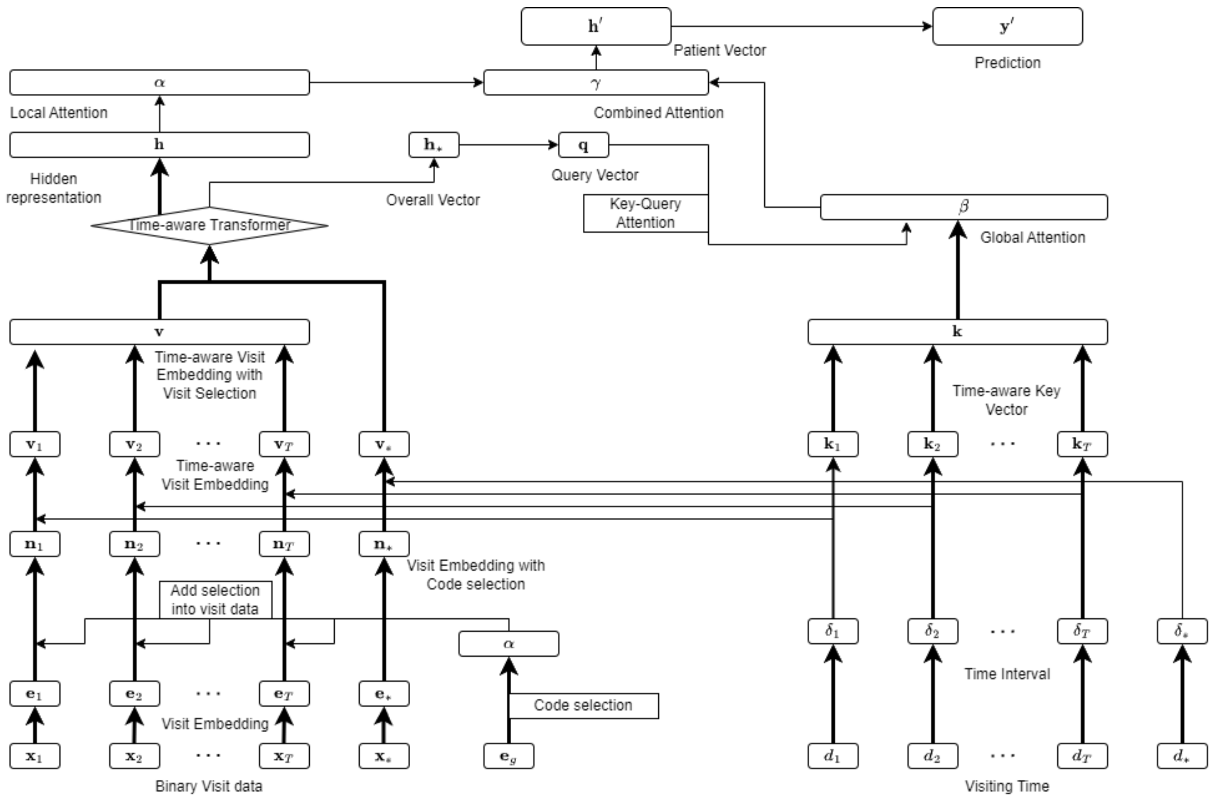
Definition 3 (Time Interval). We make $\mathbf{D} = [d_1, d_2, \cdots, d_T, d_*]$ become the visit time record of visit $\mathbf{X}$ [1]. Here, the t-th visit has the visit time record as d_t [1], meaning how long from a hypothetical initial time 0 to the t-th visit. For the overall visiting data of the patient, which is $\mathbf{x}_*$, we set $d_* = d_T$, which is the last time record of the patient [1]. For easier to use the model, we make $\delta_t = d_T - d_t$ which is the time interval between the last time record and the t-th visit for which d_T is predicting time [1]. In the model, we would transfer the input $\mathbf{d}$ into $\boldsymbol{\Delta} = [\delta_1, \delta_2, \cdots, \delta_T, \delta_*]$. We change $\mathbf{D}$ to $\boldsymbol{\Delta}$ because we want the values that record time to have the property that a larger time distance has less influence on the prediction and would not have a negative record. This kind of property simulates the doctors making the decision that pay more attention to the closer health record, not the further record.

Based on these inputs $\mathbf{X} \in \{0, 1\}^{T \times (N+1)}$ and $\boldsymbol{\Delta}$, we want to get the output that shows whether the patients have the sick we are predicting or not. [1].

Exactly this kind of task is not new for academia. However, most of the models have not arrived at the perfect result that humans need because of the complexity of the situations and the limitations of models so we try to design a new model that can perform better than old main models like HiTANet [1] in doing the risk prediction.

3.2 The Proposed Model

This section will present the SeleHiTANet model (Figure 3.1). This model has three major parts: Finding local attention value, finding global attention value, and making the prediction with a combination of both attention values. In finding local attention value, there are five parts: encoding visit data, doing code selection, visiting, embedding the time, time-aware Transformer, and finding local attention. In finding global attention value, there are three parts: getting a time-aware Key Vector, key-query attention mechanism, and finding global attention.



$\mathbf{h}'$
Patient Vector
$\mathbf{y}'$
Prediction
α
Local Attention
γ
Combined Attention
$\mathbf{h}$
Hidden representation
$\mathbf{h}_*$
$\mathbf{q}$
Query Vector
Overall Vector
Key-Query Attention
β
Global Attention
Time-aware Transformer
$\mathbf{v}$
Time-aware Visit Embedding with Visit Selection
$\mathbf{k}$
Time-aware Key Vector
$\mathbf{v}_1$ $\mathbf{v}_2$ $\mathbf{v}_T$ $\mathbf{v}_*$
Time-aware Visit Embedding
$\mathbf{k}_1$ $\mathbf{k}_2$ $\mathbf{k}_T$
$\mathbf{n}_1$ $\mathbf{n}_2$ $\mathbf{n}_T$ $\mathbf{n}_*$
Visit Embedding with Code selection
Add selection into visit data
α
δ_1 δ_2 δ_T δ_*
Time Interval
Code selection
$\mathbf{e}_1$ $\mathbf{e}_2$ $\mathbf{e}_T$ $\mathbf{e}_*$
Visit Embedding
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_T$ $\mathbf{x}_*$ $\mathbf{e}_g$
d_1 d_2 d_T d_*
Binary Visit data
Visiting Time

Figure 3.1: ICD-9 code with Code Selection for a patient

3.2.1 Visit Data Encoding

For each patient data, we have a binary visit vector x_t, which is sparse. The binary visit vector is sparse because the medical records only document the different diagnoses obtained at different points in time. In fact, a patient cannot have too many illnesses simultaneously. Hence each episode of illness is relatively infrequent within the ICD-9 framework. This leads to the binary visit vector being sparse. However, dense matrices are easier to incorporate and compute within models. So firstly, we are trying to encode the diagnosis codes matrix into a dense space which is easier to calculate $\mathbf{e}_t \in \mathbb{R}^m$as follows [1]:

$$\mathbf{e}_t = \mathbf{W}_e \mathbf{x}_t + \mathbf{b}_e, \tag{3.1}$$

where $\mathbf{W}_e \in \mathbb{R}^{m\times(N+1)}$ is the weight matrix and $\mathbf{b}_e \in \mathbb{R}^m$ is the bias vector. Also, we are doing the target embedding of g, which is $\mathbf{e}_g$. The visit embedding of each patient can be represented by matrix $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_T, \mathbf{e}_*]$ [1]. Each of $\mathbf{e}_t$ means the t-th visit embedding, and $\mathbf{e}_*$ means the visit embedding for the overall situation. Here we choose to use the Transformer structure because we can easily know how the visit information is embedded in the models and analysis the model easily [1]. The reason is that compared with the RNN model, the transformer would not lose much information. The benefits of employing a Transformer are two [1]. It combines each visit data with all other visit data in a self-attention method [1].

3.2.2 Code Selection

As we said, we choose Gumbel-Softmax [7] as the code selection mechanism. We try to apply this mechanism as it works in Medskim model [5] because it was proved in working on healthcare data before. Based on the visit embeddings which we got in visit data embedding $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_T, \mathbf{e}_*]$ and the target embedding $\mathbf{e}_g$ we got, we can getting the codes which we should keep in using Gumbel-Softmax [7] before doing the transformers [5]:

$$\begin{aligned} \mathbf{p}_s &= \mathrm{Softmax}\left(\mathbf{W}_p\left[\mathbf{e}_s, \mathbf{e}_g\right] + \mathbf{b}_p\right), \\ a'_m &= \mathrm{Binarize}\left(\frac{\exp\left(\left(\log\left(\mathbf{p}_s[0]\right) + g_0\right)/\tau\right)}{\sum_{j=0}^{1}\exp\left(\left(\log\left(\mathbf{p}_s[j]\right) + g_j\right)/\tau\right)}\right), \end{aligned} \tag{3.2}$$

where $\mathbf{p}_s \in \mathbb{R}^2$ is a vector with which have the probability that the code would be selected or not selected for the s-th code, τ is the softmax temperature, g_j is independent and identically distributed random samples which chosen from $\mathrm{Gumbel}(0, 1)$, and $[\cdot, \cdot]$ means the operation of concatenation [5]. Based on the calculating, $\mathbf{W}_p \in \mathbb{R}^{2\times(2\times m)}$ and $\mathbf{b}_p \in \mathbb{R}^2$ are parameters [5] [5]. After doing these, we multiply

$$\mathbf{n}_t = \sum_{s=1}^{N+1} a'_s * \mathbf{x}_{t,s} * \mathbf{e}_s, \tag{3.3}$$

where $\mathbf{n}_t \in \mathbb{R}^m$ is the visit embedding and $\mathbf{N} = [\mathbf{n}_1, \mathbf{n}_2, \cdots, \mathbf{n}_T, \mathbf{n}_*]$.

3.2.3 Time Embedding

After this, we should embed the time vector Δ into the visit space with the $\mathbf{N}$ which we got from the code selection mechanism [1].

$$\begin{aligned} \mathbf{f}_t &= \mathbf{1} - \tanh\left(\left(\mathbf{W}_f \frac{\delta_t}{180} + \mathbf{b}_f\right)^2\right), \\ \mathbf{r}_t &= \mathbf{W}_r \mathbf{f}_t + \mathbf{b}_r, \end{aligned} \tag{3.4}$$

where $\mathbf{W}_f \in \mathbb{R}^a, \mathbf{b}_f \in \mathbb{R}^a, \mathbf{W}_r \in \mathbb{R}^{m \times a}$, and $\mathbf{b}_r \in \mathbb{R}^m$ are all parameters and δ_t is the time interval vector that we got [1]. $\mathbf{r}_t \in \mathbb{R}^m$ is the time vector in visit space of $\mathbf{N}$ [1]. Based on the basic medical knowledge, in sick prediction tasks, the most recent visits are more significant for doing the task. As a result, visits occurring closer to the last one should be prioritized and activated. This approach enables different time distances to exert various influences on sick prediction. After mapping the time vector into the same space of $\mathbf{N}$, we perform an addition operation to embed the time information into the visit embedding [1],

$$\mathbf{v}_t = \mathbf{n}_t + \mathbf{r}_t, \tag{3.5}$$

which $\mathbf{v}_t \in \mathbb{R}^m$ is the input of the Time-aware Transformer [1].

3.2.4 Time-aware Transformer

After doing the time embedding, we have the input matrix $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_T, \mathbf{v}_*]$, with a Transformer which we use F to represent it [1]. We use this transformer to get the connection between each time of visiting data with the time information [1].

$$[\mathbf{h}_1, \mathbf{h}_2 \cdots, \mathbf{h}_T, \mathbf{h}_*] = F\left([\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_T, \mathbf{v}_*]\right), \tag{3.6}$$

where $\mathbf{h}_t \in \mathbb{R}^l$ is the t-th visiting hidden representation, which includes the combination between all of the visiting information with time embedding and their connections [1]. In the models training and testing, $l = m$. About the transformer F [1, 2], first, we add position embedding, which embedded the position information of the visits (for t-th visit, the position information is t) to the $\mathbf{V}$ including time information and visiting code information which we got previously [1]. After doing these, we use the scaled dot-product attention to build the connection between each visiting data [1]. Finally, we use a feed-forward network to increase the feature of each embedding position. Transformer contains a positional encoding procedure to embed the position information into the visiting embedding [1]. We try to find out the inner position encoding values that are

$$\begin{aligned} PE_{(t,2i)} &= \sin\left(t/10000^{2i/m}\right), \\ PE_{(t,2i+1)} &= \cos\left(t/10000^{2i/m}\right), \end{aligned} \tag{3.7}$$

where m is the dimension size of the hidden space, and i is the placement of the visiting data with position embedding PE [1]. The position embedding will be added to $\mathbf{v}_t$ for embedding positions into visit embeddings with time information [1]. So we find $\mathbf{p}_t$ there is

$$\mathbf{p}_t = \mathbf{v}_t + PE_t = \begin{bmatrix} \mathbf{v}_{(t,1)} + \cos\left(t/10000^{0/m}\right) \\ \mathbf{v}_{(t,2)} + \sin\left(t/10000^{2/m}\right) \\ \mathbf{v}_{(t,3)} + \cos\left(t/10000^{4/m}\right) \\ \vdots \end{bmatrix}, \tag{3.8}$$

which $\mathbf{p}_t \in \mathbb{R}^m$. Based on the results, we use three fully connected layers to generate three additional representations as $\mathbf{q}'_t, \mathbf{k}'_t, \mathbf{v}'_t \in \mathbb{R}^m$ [1]. They are represented as query vectors, keys vectors, and values vectors. For these values, we have

$$\begin{aligned} \mathbf{q}'_t &= \mathbf{W}_q\mathbf{p}_t + \mathbf{b}_q, \\ \mathbf{k}'_t &= \mathbf{W}_k\mathbf{p}_t + \mathbf{b}_k, \\ \mathbf{v}'_t &= \mathbf{W}_v\mathbf{p}_t + \mathbf{b}_v, \end{aligned} \tag{3.9}$$

which have $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{m\times m}$ and $\mathbf{b}_q, \mathbf{b}_k, \mathbf{b}_v \in \mathbb{R}^m$. We set the number of attention groups as 4. Because of that, in combing them, when

$$\mathbf{q}'_t = \begin{bmatrix} \mathbf{q}'_{t1} \\ \mathbf{q}'_{t2} \\ \mathbf{q}'_{t3} \\ \mathbf{q}'_{t4} \end{bmatrix}, \quad \mathbf{k}'_t = \begin{bmatrix} \mathbf{k}'_{t1} \\ \mathbf{k}'_{t2} \\ \mathbf{k}'_{t3} \\ \mathbf{q}'_{t4} \end{bmatrix}, \mathbf{v}'_t = \begin{bmatrix} \mathbf{v}'_{t1} \\ \mathbf{v}'_{t2} \\ \mathbf{v}'_{t3} \\ \mathbf{v}'_{t4} \end{bmatrix}, \tag{3.10}$$

with $\mathbf{q}'_{ti}, \mathbf{k}'_{ti}, \mathbf{v}'_{ti} \in \mathbb{R}^{\frac{m}{4}}$, we can further build these vectors into three matrix because we are using multi-head self-attention mechanism to building the connection between each visiting of the patient data [1]. These three two-dimension matrix are $\mathbf{Q}'_t, \mathbf{K}'_t, \mathbf{V}'_t$ which is

$$\begin{aligned}\mathbf{Q}'_t &= \begin{bmatrix}\mathbf{q}'_{t1} & \mathbf{q}'_{t2} & \mathbf{q}'_{t3} & \mathbf{q}'_{t4}\end{bmatrix}, \\ \mathbf{K}'_t &= \begin{bmatrix}\mathbf{k}'_{t1} & \mathbf{k}'_{t2} & \mathbf{k}'_{t3} & \mathbf{q}'_{t4}\end{bmatrix}, \\ \mathbf{V}'_t &= \begin{bmatrix}\mathbf{v}'_{t1} & \mathbf{v}'_{t2} & \mathbf{v}'_{t3} & \mathbf{v}'_{t4}\end{bmatrix},\end{aligned} \tag{3.11}$$

that have $\mathbf{Q}'_t, \mathbf{K}'_t, \mathbf{V}'_t \in \mathbb{R}^{4\times\frac{m}{4}}$ and the final attention fusion is:

$$\mathbf{X}'_t = \text{Attention}\left(\mathbf{Q}'_t, \mathbf{K}'_t, \mathbf{V}'_t\right) = \text{Softmax}\left(\frac{\mathbf{Q}'_t \mathbf{K}'^{T}_t}{\sqrt{d_k}}\right)\mathbf{V}'_t, \tag{3.12}$$

where d_k is the dimension of attention embedding and $\mathbf{X}'_t \in \mathbb{R}^{4\times\frac{m}{4}}$ [1]. We used this formula because we decided to set the number of heads in the attention mechanism to become 4 [1]. Based on the test result, we set it to 64. For getting the final attention fusion $\mathbf{X}'_t$, we designed the matrix $\mathbf{Q}'_t, \mathbf{K}'_t, \mathbf{V}'_t$ as the query matrix, key matrix, and value matrix [1]. After doing these, we make $\mathbf{X}'_t \in \mathbb{R}^{4\times\frac{m}{4}}$ that $\mathbf{X}'_t = \begin{bmatrix}\mathbf{x}'_{t1} & \mathbf{x}'_{t2} & \mathbf{x}'_{t3} & \mathbf{x}'_{t4}\end{bmatrix}$ from two-dimension to one dimension which is $\mathbf{x}'_t \in \mathbb{R}^m$ then it have $\mathbf{x}'_t = \begin{bmatrix}\mathbf{x}'_{t1} \\ \mathbf{x}'_{t2} \\ \mathbf{x}'_{t3} \\ \mathbf{x}'_{t4}\end{bmatrix}$. Finally. we build a feed-forward layer which is working like that

$$\mathbf{h}_t = \text{FFN}\left(\mathbf{x}'_t\right) = \max\left(\mathbf{0}, \mathbf{x}'_t\mathbf{W}_1 + \mathbf{b}_1\right)\mathbf{W}_2 + \mathbf{b}_2. \tag{3.13}$$

when $\mathbf{W}_1 \in \mathbb{R}^{b\times m}, \mathbf{W}_2 \in \mathbb{R}^{m\times b}, \mathbf{b}_1 \in \mathbb{R}^b, \mathbf{b}_2 \in \mathbb{R}^m$ [1]. We set the dimension size of the feed-forward space, which is used in the middle of the feed-forward network, is b [1].

3.2.5 Local Attention Weights

Exactly, just like doctors visiting patients, they would not only focus on current visits but also review historical EHRs with the records highly related to the target disease because historical EHRs are important parts of making the basic prediction. For making the simulation on the model that looks like real doctors, we try to get an attention score η_t for each time of visit except overall visit $\mathbf{h}_*$ [1],

$$\eta_t = \mathbf{W}_\eta^\top \mathbf{h}_t + b_\eta \tag{3.14}$$

where $\mathbf{W}_\eta \in \mathbb{R}^l$ and $b_\eta \in \mathbb{R}$ are parameters here [1]. After we got the attention vector $\boldsymbol{\eta} = [\eta_1, \eta_2, \cdots, \eta_T] \in \mathbb{R}^N$ based on the equation 3.14, we applied a softmax layer on the attention vector for generating the local attention weights [1], which is,

$$\boldsymbol{\alpha} = \mathrm{Softmax}(\boldsymbol{\eta}) = [\alpha_1, \alpha_2, \cdots, \alpha_T] \tag{3.15}$$

3.2.6 Time-aware Key Vector

In the previous part, we get the local attention weight for each visiting data just like a doctor checking a patient based on each visit situation [1]. Now we simulate doctors how to find the patients' overall situation for making the final judgment with the overall diagnosis $\mathbf{h}_*$ [1]. To simulate it, we are using a new kind of key-query attention mechanism that embeds the time information [1]. Firstly, we build a query vector $q \in \mathbb{R}^s$ for the overall diagnosis $\mathbf{h}_*$ which is:

$$\mathbf{q} = \mathrm{ReLU}\left(\mathbf{W}_q \mathbf{h}_* + \mathbf{b}_q\right) \tag{3.16}$$

where $\mathbf{W}_q \in \mathbb{R}^{s \times l}$ and $\mathbf{b}_q \in \mathbb{R}^s$ are parameters [1]. We only build the query vector by the overall diagnosis because we only consider the patients' overall situation in this part. We set the activation function as ReLU for only saving the positive values because the negative values are mostly close to useless and positive values are more valuable [1]. For keeping the time information of the sick which we want to predict (just like doctors focus on some special time record when they are checking the situation of the patients), we still need to embed time information δ_t into a latent space as follows [1]:

$$\begin{aligned} \mathbf{o}_t &= \mathbf{1} - \tanh\left(\left(\mathbf{W}_o \frac{\delta_t}{180} + \mathbf{b}_o\right)^2\right), \\ \mathbf{k}_t &= \tanh\left(\mathbf{W}_k \mathbf{o}_t + \mathbf{b}_k\right), \end{aligned} \tag{3.17}$$

where $\mathbf{W}_o \in \mathbb{R}^n, \mathbf{b}_o \in \mathbb{R}^n, \mathbf{W}_k \in \mathbb{R}^{s \times n}$, and $\mathbf{b}_k \in \mathbb{R}^s$ are parameters, and $\mathbf{k}_t \in \mathbb{R}^s$ is key vector which embed the time information into the overall situation of the patients [1]. This formula is similar to the previous formula in time embedding but in a different target. This formula attempts to capture the significance of time information only based on the property of the sicks that we want to predict. Here we only use the time information, which positively affects prediction, so we add the ReLU activation function for saving time information.

3.2.7 Key-query Attention Mechanism and Global Attention Weights

Continuously, we combine Time-aware key vectors $\mathbf{k}$ and query vector $\mathbf{q}$ together to find the significance of each time interval in the risk prediction process and get the attention scores based on the key-query attention mechanism in Transformer [1, 2]. Based on the attention mechanism, we have the attention weight that [1]:

$$\phi_t = \frac{\mathbf{q}^\top \mathbf{k}_t}{\sqrt{s}}. \tag{3.18}$$

After getting attention weight, just like HiTANet [1], we use softmax to normalize the attention weights and find out the global attention weights:

$$\boldsymbol{\beta} = \mathrm{Softmax}(\boldsymbol{\phi}) = [\beta_1, \beta_2, \cdots, \beta_T]\,. \tag{3.19}$$

3.2.8 Combined Attention Weights

In 3.2.5 and 3.2.7, we got two attention vectors for two different kinds of view in diagnosis: the local attention vector $\boldsymbol{\alpha}$ and the global attention vector $\boldsymbol{\beta}$. Local attention vector simulates doctor checking each time of visit data with doctors' diagnosis procedure, and global attention vector simulates doctor checking overall visit data with retrospectively analyzes. Because they are considered predictions from different perspectives, we should combine these two attention weights like a doctor making the final diagnosis decision that considers all situations. Because of that, we should try to embed the overall representation $\mathbf{h}_*$ into a space to show how much the model should pay attention to local attention or global attention and then normalize it with a softmax layer like this [1]:

$$\mathbf{z} = \mathrm{Softmax}\left(\mathbf{W}_z \mathbf{h}_* + \mathbf{b}_z\right) = [z_\alpha, z_\beta]\,, \tag{3.20}$$

where $\mathbf{W}_z \in \mathbb{R}^{2\times l}$ and $\mathbf{b}_z \in \mathbb{R}^2$ are parameters [1]. After getting $\mathbf{z}$, which shows how much the model should pay attention to local or global attention, we use this to get the overall attention weight [1]. We got the rate of overall attention weights at each visiting time of the patients as follows [1]:

$$\gamma_t' = \alpha_t * z_\alpha + \beta_t * z_\beta \tag{3.21}$$

Finally, we try to find out attention score γ_t based on the average overall prediction weight for each visit as follows [1]:

$$\gamma_t = \frac{\gamma_t'}{\sum_{j=1}^{T} \gamma_j' + 1 * 10^{-5}} \tag{3.22}$$

We add 1e-5 in the equation to avoid zero situations for the $\sum_{j=1}^{T} \gamma_j'$.

3.2.9 Prediction

Based on the combined attention scores γ_t, which we got in 3.2.8, and the hidden representation $\mathbf{h}$, which records all of the information, we can finally get the patient vector that combined the overall attention with the hidden representations [1]:

$$\mathbf{h}' = \sum_{t=1}^{T} \gamma_t' \mathbf{h}_t \tag{3.23}$$

We use a simple linear layer with a softmax layer to make a prediction [1]:

$$\mathbf{y}' = \operatorname{Softmax}\left(\mathbf{W}_u \mathbf{h}' + \mathbf{b}_u\right), \tag{3.24}$$

where $\mathbf{W}_u \in \mathbb{R}^{2\times l}$ and $\mathbf{b}_u \in \mathbb{R}^2$ are parameters [1]. In the formula, $\boldsymbol{\theta}$ represent all parameters like each kind of $\mathbf{W}, \mathbf{b}$ [1]. For showing the result of the model, we use $\mathbf{y}$ as the ground truth of the model [1]. Like traditional models [1], the cross-entropy between the ground truth and the prediction probabilities$\mathbf{y}'$ is used to calculate the loss for this model. Because of that, our task in the model is trying to minimize the average of cross-entropy loss [1]:

$$\mathcal{L}(\theta) = -\frac{1}{|\mathcal{P}|} \sum_{p=1}^{|\mathcal{P}|} \left(\mathbf{y}_p^\top \log\left(\mathbf{y}_p'\right) + (\mathbf{1} - \mathbf{y}_p)^\top \log\left(1 - \mathbf{y}_p'\right)\right), \tag{3.25}$$

where $|\mathcal{P}|$ is the total number of patient data.

3.3 Model Training

We first separated the set into input, validation, and test datasets for the training algorithm. Next, we begin to train the model. In training, we randomly initialize the parameter θ of the SeleHiTANet model first. Then we do the training in several epochs. For each epoch, we try to cycle all inputs in the input dataset and send them into the model processing with the parameter θ. Each cycle would update θ according to the gradient of loss $\mathcal{L}$. After doing these, we would calculate the average loss on the validation data set with inputs we got in the cycle and find the θ_{best} which has the least average loss $\mathcal{L}_v^{\min}$ on the validation set with the input based on the parameter that we have. Then we cycle some epochs on the training and finally return the best parameter θ_{best} for testing the SeleHiTANet model.

3.4 Algorithm of SeleHiTANet

The algorithm is modified based on the HiTANet model's algorithms [1]

Algorithm 1 Training process of the SeleHiTANet model

```
Input: Training set $\mathcal{D}_t$, and validation set $\mathcal{D}_v$
Output: Trained model parameter $\boldsymbol{\theta}_{\text{best}}$
  Initialize the parameter $\boldsymbol{\theta}$ of SeleHiTANet randomly (Initialize would not influence result);
  for $epoch = 1$ to $EPOCH$ do
    Initialize the sample order of training set $\mathcal{D}_t$ randomly(Initialize would not influence result)
    for $(X, \Delta, y) \in \mathcal{D}_t$ do
      Obtain dense embeddings $\mathbf{e}$ based on Eq. (3.1)
      Calculate selecting actions $\mathbf{a}'$ based on Eq. (3.2)
      Calculate selected dense embeddings $\mathbf{n}$ based on Eq. (3.3)
      Obtain time embeddings $\mathbf{r}$ based on Eq. (3.4)
      Calculate hybrid input $\mathbf{v}$ based on Eq. (3.5)
      Encode $\mathbf{v}$ to obtain $\mathbf{h}$ with the transformer $F$ based on Eq. (3.6)
      Calculate local attention $\alpha$ based on Eq. (3.14) and Eq. (3.15)
      Calculate query vector $\mathbf{q}$ with $\mathbf{h}_*$based on Eq. (3.16)
      Obtain time-aware key vector $\mathbf{o}$ based on Eq. (3.17)
      Calculate global attention weight  with $\phi$ based on Eq. (3.18)
      Calculate local attention $\beta$ based on Eq. (3.19)
      Calculate embedded overall representation $\mathbf{z}$ based on Eq. (3.20)
      Calculate the overall attention weights $\gamma'$ based on Eq. (3.21)
      Calculate the final attention score $\gamma'$ based on Eq. (3.22)
      Calculate the representations of patient vectors $\mathbf{h}'$ based on Eq. (3.23)
      Calculate the prediction $\mathbf{y}'$ based on Eq. (3.24)
      Calculate the prediction loss $\mathcal{L}$ based on Eq. (3.25)
      Add new parameters $\boldsymbol{\theta}$ based on the gradient of $\mathcal{L}$
    end for
    Calculate the average validation loss $\mathcal{L}_v$ on validation set $\mathcal{D}_v$
    if $\mathcal{L}_v < \mathcal{L}_v^{\min}$ then
      $\boldsymbol{\theta}_{best} = \boldsymbol{\theta}$
      $\mathcal{L}_v = \mathcal{L}_v^{\min}$
    end if
  end for
```

Chapter 4 Experiments

4.1 Datasets

Our model utilizes a meticulously preprocessed, private dataset of chronic obstructive pulmonary disease (COPD), Heart Failure, Kidney Disease, Amnesia, and Dementia cohorts, extracted from an authentic EHR database. We've provided a detailed dataset overview in Table 4.1. These cohorts were selected by the need to replicate real-world patient conditions and demographics.

We make the sick prediction task become a binary classification problem. The dataset encompasses various types of visiting data with different situations to simulate real-world medical diagnostic processes closely. When selecting data for positive cases, we employed a specific method - we identified the initial disease diagnosis date and included EHR data from the six months preceding that date.

On the other hand, for each patient classified in the negative control group, we removed the visits from the final year and used the remaining visits as the input data. This approach helped us to balance the data and provide a fair comparative ground for our predictive model. Also,we restricted the maximum length of each patient's record in the dataset to 50 visits.

Dataset	COPD	Heart Failure	Kidney Disease	Amnesia	Dementias
Case (Positive)	7314	3080	2810	2982	2385
Control (Negative)	21942	9240	8430	8946	7155
Avg visits per patient	30.39	38.74	39.09	39.00	41.05
Avg codes per visit	3.50	4.24	4.40	4.70	4.71
Unique ICD-9 codes	10053	8692	8802	9032	7813

Table 4.1: Dataset Details

4.2 Baselines

Our research focused on contrasting the SeleHiTANet model with an array of baseline models that included SVM [9], LSTM [10], Dipole [36], T-LSTM [8], and HiTANet [1]. Notably, the true baseline for our study was HiTANet, given its direct relevancy to our work. The remaining models were included for comparative purposes, their data being gathered from the published HiTANet paper. While this comparison might give an impression of informality, it's worth noting that HiTANet has already demonstrated superior performance over these other models in prior studies. If the SeleHiTANet performs better than HiTANet, the SeleHiTANet model is better than all other baseline models.

To show completeness and offer more perspectives of the field, we ensured that a representative baseline was included for each methodological category. This includes Classical Methods, Plain RNNs, Attention, Time, and Transformer-Based Models. This comprehensive comparison aimed to give an all-encompassing view of how SeleHiTANet stands against different approaches.

Our evaluation involved several metrics: Accuracy (Acc), Precision (Pre), Recall, F1 score, and the Area Under Curve (Auc) score [37]. For doing the evaluation, there are true positives(TP, positive sample with true prediction), false positives(FP, positive sample with false prediction), false negatives(FN, negative sample with false prediction), and true negatives(TN, negative sample with true prediction) in the dataset.

Accuracy is the most direct performance measure we mostly use in many areas. It is simply a ratio of correct predictions to the total,

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}, \tag{4.1}$$

The precision score is the ratio of correctly predicted positive to the total predicted, which was predicted to be positive, which means how many mistakes in the positive predictions. High precision scores depend on a low false positive rate,

$$Precision = \frac{TP}{TP + FP}, \tag{4.2}$$

Recall score is the ratio of correctly positive predictions to total positive predictions in the actual dataset, which means how many positive cases were found in predictions. It depends on how many right predictions the model can find out,

$$Recall = \frac{TP}{TP + FN}, \tag{4.3}$$

The F1 score is the harmonic average of Precision and Recall scores. It tries to find the balance between Precision and Recall scores. It considers how many right positives they make and how many right predictions the model can discover.

$$F1\ Score = \frac{2 * (Recall * Precision)}{Recall + Precision}, \tag{4.4}$$

The area under the Roc Curve score (Auc) measures the entire two-dimensional area underneath the entire receiver operating characteristic curve (Roc Curve) from (0,0) to (1,1) [38]. Roc Curve plots the true positive rate, which is the Recall score against the false positive rate, as the discrimination threshold of the classifier is varied [38]. It considers different types of metrics and finds the balance between them to get all of their benefits [38]. A model whose predictions are 100% correct has an Auc of 1; a model whose predictions are 100% wrong has an Auc of 0. It is typically computed using numerical methods on the Roc curve [38]. The random classifier would have a 0.5 score and plot a diagonal line from the bottom left to the upper right [37, 38].

Given the nature of our binary classification task, if we only consider accuracy, it might provide a skewed perspective, hence the decision to use a broader range of metrics. Among these, the Auc score was chosen as the key result indicator because it captures the trade-off between sensitivity and specificity, providing a holistic view of model performance.

4.3 Implementations

For the SeleHiTANet model, we implement it in PyTorch [39] and train on a Win 10 with 16GB memory and a GeForce RTX 3070 GPU. When doing the task, we set the values similar to the model HiTANet [1]. We are using the machine learning methods we implement by scikit-learn [40]. We also only consider the top 256 frequent diagnosis codes to represent the EHR records that we have [1]. Also, we make the batch size 50 for all the methods [1]. And we set the dimension $l = m$ of the final hidden state for prediction to 256 [1]. If there is no hierarchical structure, we will set the layer of RNN or Transformer to 1 [1].

Dropout [41] method is a kind of way that the model gives up some of the inputs in the layers. We utilize Dropout methods [41] to mitigate overfitting, and we set the dropout rate to 0.5. The effectiveness of dropout lies in its ability to randomly deactivate certain features with a given probability during model training. This enhances the model's generalization capability, preventing it from relying too much on specific local features.

We trained the data by Adam optimizer [42]. The proposed SeleHiTANet's learning rate is set to 1e-4, $m = 256$, $a = 64$, $s = 64$, and $n = 64$ which also similar to HiTANet [1]. Similarity [1], for the hyper-parameters of the Transformer, we set the dimension size of attention embedding to become $d_k = 64$ with the multi-head number which is 4, and the size of the middle feed-forward network $b = 1024$.

For all methods, we randomly split each dataset into three parts: training, validation, and testing. They are arranged with a ratio of 0.75:0.10:0.15 randomly. We try to find the best model on the validation dataset and report its performance on the test dataset. We run the training and testing five times randomly and report both average results and standard deviation between results for testing performance. Running the complete model five times can reduce the randomness of the test results. The same model will not operate the same when training, validating, and testing on the same dataset. Its parameters and test results may not necessarily be the same. Therefore, we ran it five times to get closer to the real result.

4.4 Results

4.4.1 COPD

In Table 4.2, the SeleHiTANet model has only a Pre score lower than SVM [9] model in the COPD dataset and has the same Acc score compared with HiTANet [1].

	Acc	Pre	Recall	F1	Auc (Main)
SVM	0.804	**0.713**	0.319	0.441	0.639
LSTM	0.807	0.680	0.461	0.548	0.693
Dipole	0.821	0.687	0.477	0.562	0.704
T-LSTM	0.818	0.687	0.525	0.595	0.722
HiTANet	**0.840**	0.707	0.583	0.637	0.752
SeleHiTANet	**0.840**	0.706	**0.585**	**0.639**	**0.755**
(std)	0.001	0.014	0.035	0.015	0.013

Table 4.2: Average Performance on COPD Prediction

4.4.2 Heart Failure

In Table 4.3, the SeleHiTANet model has Acc score lower than T-LSTM [8] model and a Pre score lower than SVM [9] model on the Heart Failure database.

	Acc	Pre	Recall	F1	Auc (Main)
SVM	0.784	**0.757**	0.327	0.457	0.644
LSTM	0.812	0.640	0.510	0.561	0.708
Dipole	0.794	0.713	0.445	0.542	0.687
T-LSTM	**0.831**	0.695	0.527	0.598	0.727
HiTANet	0.823	0.724	0.587	0.647	0.750
SeleHiTANet	0.823	0.712	**0.614**	**0.658**	**0.759**
(std)	0.004	0.016	0.044	0.019	0.015

Table 4.3: Average Performance on Heart Failure Prediction

4.4.3 Kidney Disease

Based on the result in Table 4.4, the SeleHiTANet model has only a Pre score lower than SVM [9] model in the Kidney disease dataset.

	Acc	Pre	Recall	F1	Auc (Main)
SVM	0.840	**0.777**	0.545	0.641	0.745
LSTM	0.823	0.680	0.572	0.616	0.739
Dipole	0.843	0.771	0.571	0.656	0.755
T-LSTM	0.832	0.728	0.524	0.608	0.729
HiTANet	0.851	0.743	0.668	0.702	0.792
SeleHiTANet	**0.856**	0.748	**0.682**	**0.713**	**0.800**
(std)	0.004	0.020	0.024	0.006	0.007

Table 4.4: Average Performance on Kidney Disease Prediction

4.4.4 Amnesia

According to Table 4.5, the SeleHiTANet model has Pre score lower than T-LSTM [8] only in the Amnesia dataset.

	Acc	Pre	Recall	F1	Auc (Main)
SVM	0.835	0.694	0.558	0.619	0.740
LSTM	0.811	0.706	0.455	0.548	0.694
Dipole	0.827	0.695	0.522	0.589	0.723
T-LSTM	0.826	**0.765**	0.420	0.542	0.689
HiTANet	0.848	0.727	0.597	0.654	0.762
SeleHiTANet	**0.854**	0.734	**0.616**	**0.669**	**0.773**
(std)	0.003	0.019	0.027	0.012	0.010

Table 4.5: Average Performance on Amnesia Prediction

4.4.5 Dementia

Table 4.6 shows that the SeleHiTANet model has the best performance compared with all of the baseline models in all different metrics working on the Dementia database.

	Acc	Pre	Recall	F1	Auc (Main)
SVM	0.783	**0.757**	0.153	0.255	0.569
LSTM	0.793	0.607	0.552	0.573	0.713
Dipole	0.803	0.644	0.422	0.507	0.673
T-LSTM	0.798	0.643	0.450	0.521	0.682
HiTANet	0.810	0.622	0.553	0.584	0.723
SeleHiTANet	**0.812**	0.624	**0.564**	**0.591**	**0.727**
(std)	0.007	0.025	0.029	0.010	0.008

Table 4.6: Average Performance on Dementia Prediction

4.5 Analysis

For analysis of the data, we try to show the situation of the training loss, and test loss in each epoch in different datasets(Figures 1, 3, 5, 7,and 9). Also, we want to know the changing of the AUC scores in training the model (Figures 2, 4, 6, 8, and 10). All of these figures are shown in Appendix. Simultaneously, the figure also displays the standard deviations when each model is trained to 20%, 40%, 60%, 80%, and 100%, to evaluate whether the model has been trained to its limit. In fact, due to the same initialization, the standard deviation will increase from small to large at the beginning of the training. However, as the model gets closer and closer to the target, the standard deviation will gradually decrease until it is approximately zero.

4.5.1 Score Analysis

In Tables 4.2, 4.3, 4.4, 4.5, and 4.6, we got the average performance of the SeleHiTANet model and other baseline models on COPD, Heart Failure, Kidney disease, Amnesia, and Dementia prediction with std which represents the values of standard deviation for these scores in different datasets. Compared with HiTANet [1] and most of the other baselines, SeleHiTANet has better results in most of the scores and always better in the Auc scores, which are our mainly considering scores in all of the datasets.

Unsurprisingly, SeleHiTANet shows a significant improvement over the HiTANet [1] model, as it introduces a Code Selection Mechanism. This mechanism equips SeleHiTANet with the capability to meticulously pinpoint and prioritize pertinent medical codes, thus augmenting its predictive prowess. This enhancement is pivotal in making the model more proficient in navigating complex Electronic Health Records (EHR) data. It also enables the model to efficiently filter out unnecessary information, thereby producing more accurate predictions.

Code Selection Mechanism, which comes from Medskim model [5] and is applied in SeleHiTANet, enhances the model's adaptability, allowing it to perform more effectively in real-world applications where the data is unstructured and noisy. This proves beneficial in healthcare scenarios where identifying the most significant indicators from a large volume of data is crucial for precise risk prediction.

Moreover, the innovative design of SeleHiTANet underscores the importance of model interpretability. The model offers a more transparent view of its decision-making process by selecting and prioritizing specific medical codes. This transparency fosters trust in the model's predictions, making it an invaluable tool for healthcare professionals. Consequently, the enhanced model, with its superior performance and interpretability, represents a significant contribution to the field of predictive healthcare modeling.

For both the Acc and Pre scores, the reason of SeleHiTANet is not the best one, but it is still better than HiTANet [1] may be that the transformer-based method has limited in doing the prediction underlying Acc score. According to Table 4.1, the data distribution is not balanced, just like the actual patient situation. In any database, there is significantly more negative data than positive data. For the accuracy (Acc) score, the data imbalance can lead to an inability to accurately measure whether the model is truly better during testing. For instance, when negative data makes up 99% and positive data only accounts for 1% of the total, if the model predicts everything as negative, it would still have an accuracy score of 0.99, even though the model has not produced any effective results. Precision (Pre) only considers the proportion of samples that are positive among those predicted as positive. However, if the model only predicts certain positive samples as positive, the precision score will become very high, but the actual effect of the model is quite limited. Therefore, in our datasets, the Accuracy(Acc) score and precision score cannot fully demonstrate the quality of the model. Similarly, since SeleHiTANet is an improvement upon HiTANet [1], the negative impacts on the prediction that exist in HiTANet [1] itself would remain in SeleHiTANet.

However, compared to other models, while SeleHiTANet does make progress, the score improvement is not very substantial. I speculate that SeleHiTANet also incorporates a local-based attention mechanism in other parts. This mechanism also attempts to mimic the situation when doctors review medical records. It assigns different attention to the medical record based on the record itself and the target disease to be predicted. The local-based attention mechanism will pay more attention to highly correlated diseases and vice versa. It will ignore the data. The Code Selection mechanism operates under similar circumstances and does not consider irrelevant medical records. The only difference is that the local-based attention mechanism is relatively not absolute. The visit data ignored by the Code Selection mechanism won't influence the subsequent operations.

4.5.2 Model Analysis

According to Appendix, we can see the situation of the model processing. At the beginning of training, we can observe a near-linear decrease in both training and testing loss as the number of epochs increases, regardless of the dataset. All training losses start to decline from a similar numerical point. After reaching a certain number of epochs, the decrease in test loss gradually diminishes, and it may even increase, while the training loss continues to drop linearly. This scenario is a manifestation of overfitting. Our model is designed to prevent this from occurring. We select the best hyper-parameters, not the ones from the last iteration, for testing.

Simultaneously, in Figures 1, 3, 5, 7, and 9, we observe that with each epoch (analyzed in the images from each lowest starting point), the AUC score of the training data progressively increases. The training between approximately 20% and 60% especially results in a swift elevation of the AUC score. Later on, the increase in the AUC score of the training data slows down. Similarly, the standard deviation is also influenced by the AUC score. As the AUC score increases more rapidly, the standard deviation enlarges. Both the AUC score and standard deviation decline later. Notably, for Dementias, the AUC score's growth rate is particularly fast between 20% and 50% of training (relative to other times).

In reality, we notice a significant deceleration in the progression of the training loss, the testing loss, and the AUC (Area Under the Curve) score of the training data around the 17-18th epoch. This suggests a diminishing return on further training past this point - as the model has largely converged, and the room for substantial improvement becomes less likely with each subsequent epoch. This is an important consideration in model training as it signifies a point of diminishing returns on the computational resources invested, where further training may not lead to substantial gains in model performance.

What this means for SeleHiTANet and similar models is that around 20 epochs seem to be an optimal stopping point for training. This empirical finding can be invaluable for saving computational and temporal resources by averting unnecessary training epochs. In practical terms, once the model reaches this point, it is likely that it has already learned most of the discernible patterns in the data and can provide reliable predictions.

This optimal point may be changed based on the complexity and size of the dataset. Nevertheless, it provides a useful guideline and further highlights the importance of monitoring model performance metrics throughout the training process to identify the optimal stopping point.

Chapter 5 Conclusion and Future Work

5.1 Conclusion

The introduction of SeleHiTANet makes some advances in the field of healthcare risk prediction. Drawing upon the robust attributes of the transformer-based HiTANet [1] model and incorporating the ICD-9 [4] code selection mechanism from MedSkim [5], SeleHiTANet improved the process with selecting through EHRs data. It effectively and automatically filters out irrelevant visits and codes that let healthcare models focus on the most important information and ignore useless information, thereby refining the accuracy of diagnoses and treatment planning.

SeleHiTANet is evident in practical applications, where it showcases superior performance in predicting heart failure, COPD, kidney disease, amnesia, and dementia. Even though the SeleHiTANet's performance in terms of Accuracy (Acc) and Precision (Pre) scores isn't the highest and carries forward some of its predecessor's limitations while introducing meaningful enhancements, it outdoes not only the original HiTANet model but also other baseline models across three distinct metrics, including the crucial auc score. This impressive performance underscores the potential of the code selection mechanism in bolstering the efficiency of transformer-based models in health risk prediction.

The success of SeleHiTANet promises a future where healthcare decision-making becomes more precise, data-driven, and reliable. By accurately filtering critical information, it enhances the potential of predictive modeling, thereby enabling the development of more efficient treatment strategies and personalized patient care. Ultimately, this innovation illuminates a path towards optimized healthcare delivery, leading to improved patient outcomes and streamlined healthcare operations.

Furthermore, SeleHiTANet also underscores the importance of simulating physician practices. This methodology allows the model to effectively learn from the wealth of experience of medical professionals and make accurate predictions accordingly. It shows that one of the vital factors in developing sophisticated predictive models in healthcare is to consider and mimic how physicians analyze patient data and make decisions. This entails understanding disease correlations, paying attention to relevant patient history, and making prudent selections among various medical codes, all while bearing the ultimate goal of accurate disease prediction.

5.2 Future Work

In the future, there is a great deal of room for improvement in this model. In fact, this model only considers the patient's EHR and visit time. In addition to these two factors, many other factors can affect the occurrence of diseases, such as age, climate, job, income, etc. These factors have not been taken into consideration by SeleHiTANet. In future designs, for example, from the perspective of age, we can infer the patient's age from external data or analysis of EHR and embed this information into the model. Similarly, in the future, we can provide the model with specific embeddings to fit different application environments and enhance prediction accuracy. Even when the data is being used, we can predict certain data based on the data situation in order to enhance the model's attention to that type of data. For example, when predicting geriatric diseases, we can use body condition data to predict age and apply it to predicting illnesses in the elderly. We can even predict congenital diseases or diseases of unborn children through parents' illnesses and genetic sequencing. In fact, we can try to combine the parents' EHRs to get a new EHR for their kids to use in the model.

Not only for the model, we can also improve data analysis. In fact, ICD-10 [43] and ICD-11 [44] are already in use. We can try to apply this model to EHRs using ICD-10 and ICD-11 to expand the scope of SeleHiTANet. Looking ahead, the primary objectives for advancing our model are broadening its applicability and improving its predictive accuracy. Firstly, we aim to extend the model's reach beyond its current capabilities to encompass a wider range of medical conditions and patient demographics. Secondly, we strive to enhance the prediction accuracy of our model. The intention is to refine the underlying algorithms, optimize the selection and weighting mechanisms, and continually adapt to new, real-world data for continuous learning. At the same time, we can also change the result of disease prediction into probability to measure the warning to patients.

Moreover, we aspire to incorporate a more complicated handling of imbalanced datasets, which is similar to the doctor, and implement advanced techniques for data normalization and stratification. In various scenarios, the proportion of patient data can vary. For instance, the incidence rate of diseases isn't identical across different departments within a hospital. The Emergency Department may see more acute conditions, while the Geriatric Department might encounter more chronic illnesses. Consequently, our models need to adjust accordingly to these differences. They should be flexible and adaptive, taking into account the unique characteristics of patient populations in different settings. This will enable the model to maintain high predictive accuracy even in uneven data distribution scenarios. Through these targeted improvements, we envision the evolution of the model into a robust, reliable, and universally applicable tool in healthcare.

Appendix

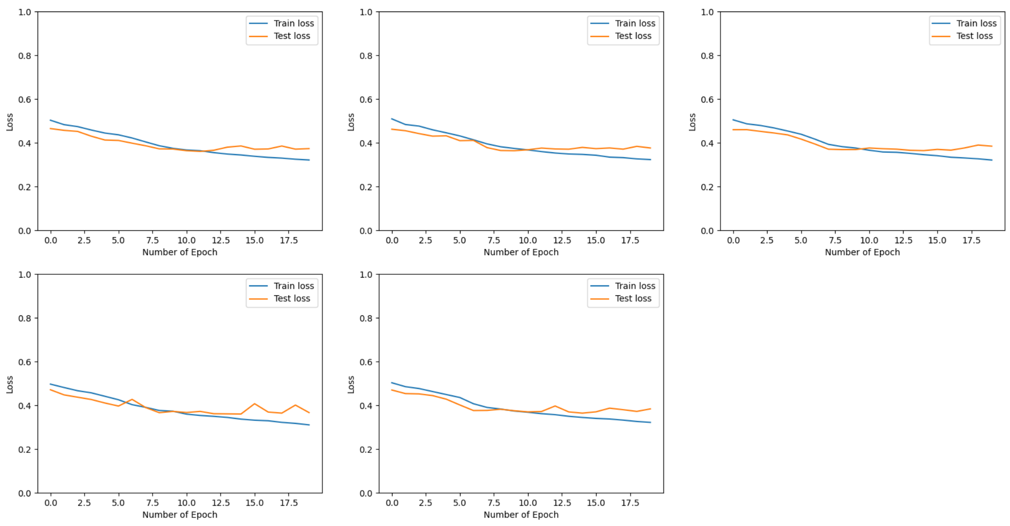


Figure 1: Number of Epoch vs the Train loss and Test loss in COPD dataset

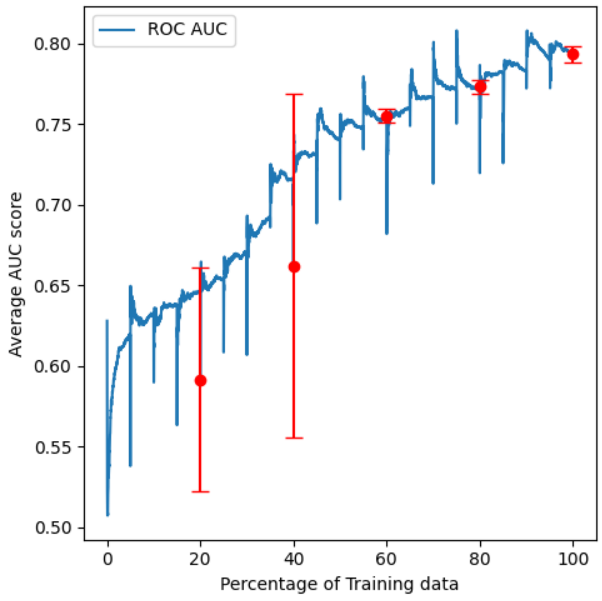


Figure 2: Percentage of Training data training vs the average training AUC score in COPD dataset

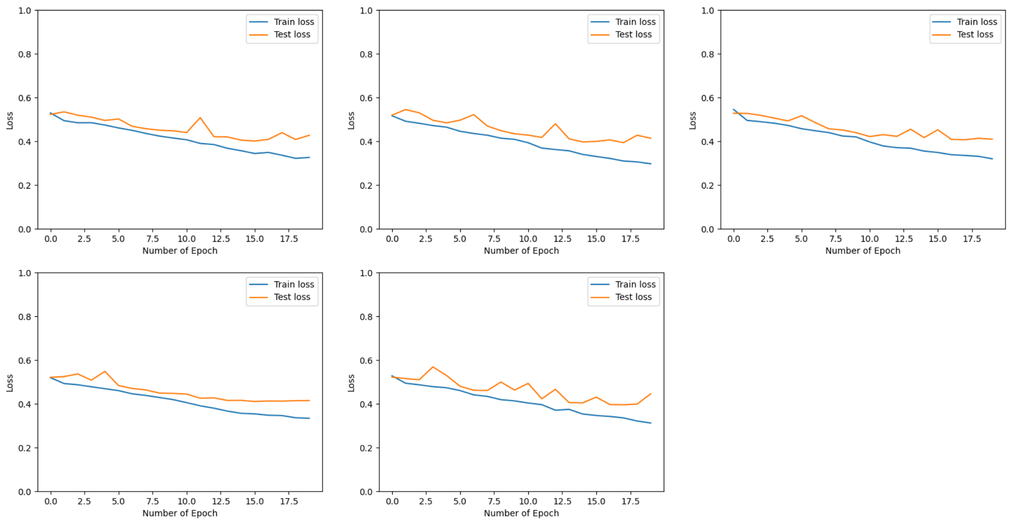


Figure 3: Number of Epoch vs the Train loss and Test loss in Heart Failure dataset

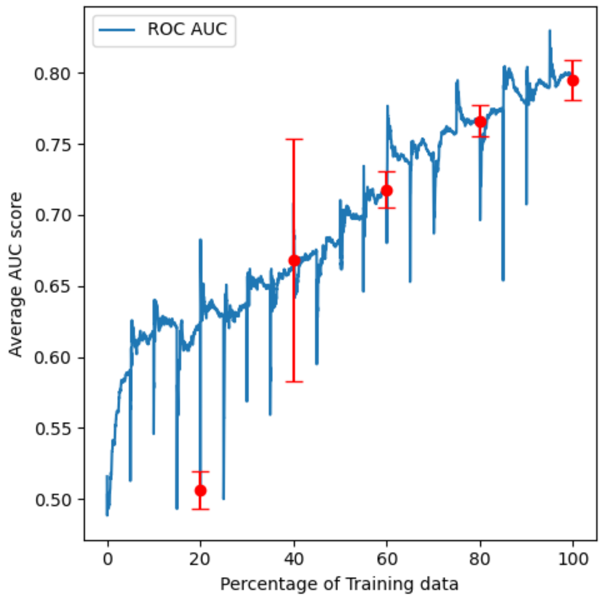


Figure 4: Percentage of Training data training vs the average training AUC score in Heart Failure dataset

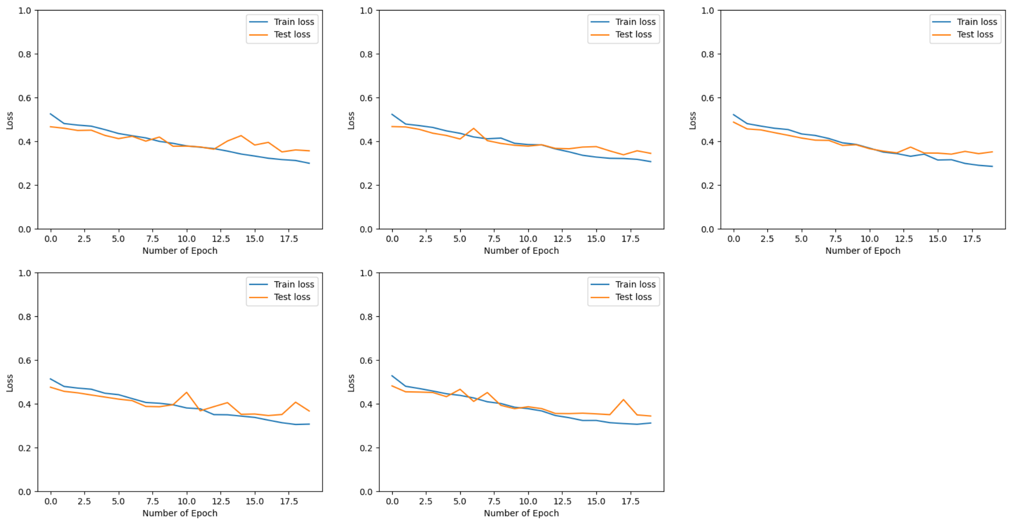


Figure 5: Number of Epoch vs the Train loss and Test loss in Kidney Disease dataset

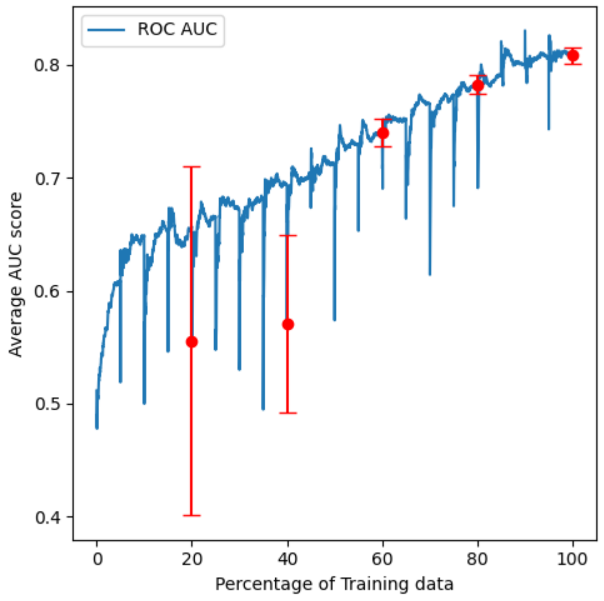


Figure 6: Percentage of Training data training vs the average training AUC score in Kidney Disease dataset

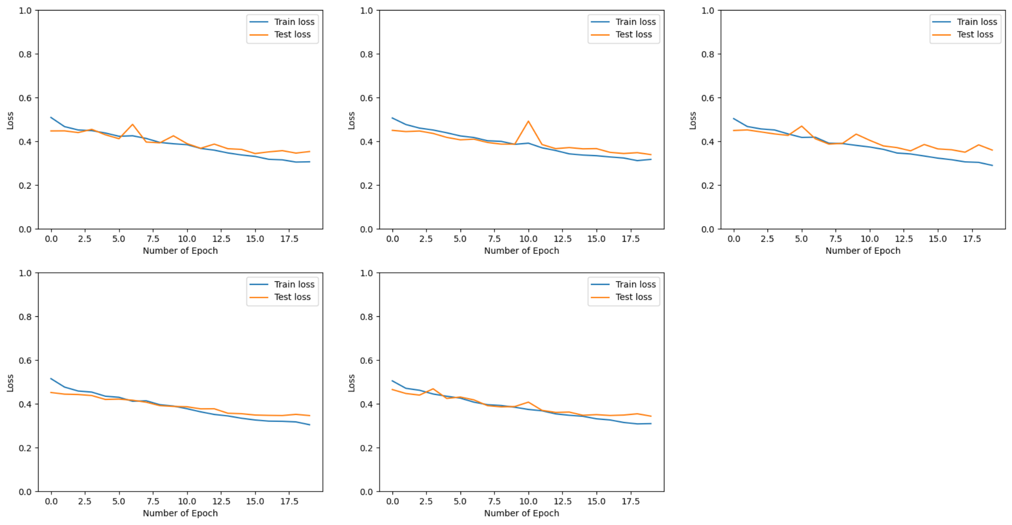


Figure 7: Number of Epoch vs the Train loss and Test loss in Amnesia dataset

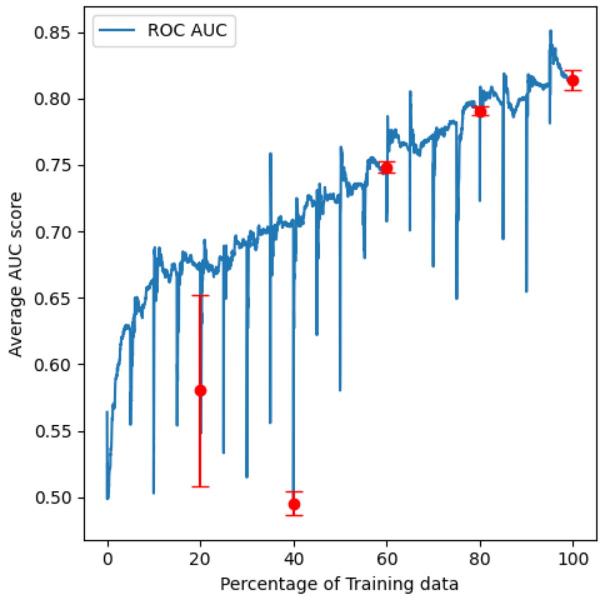


Figure 8: Percentage of Training data training vs the average training AUC score in Amnesia dataset

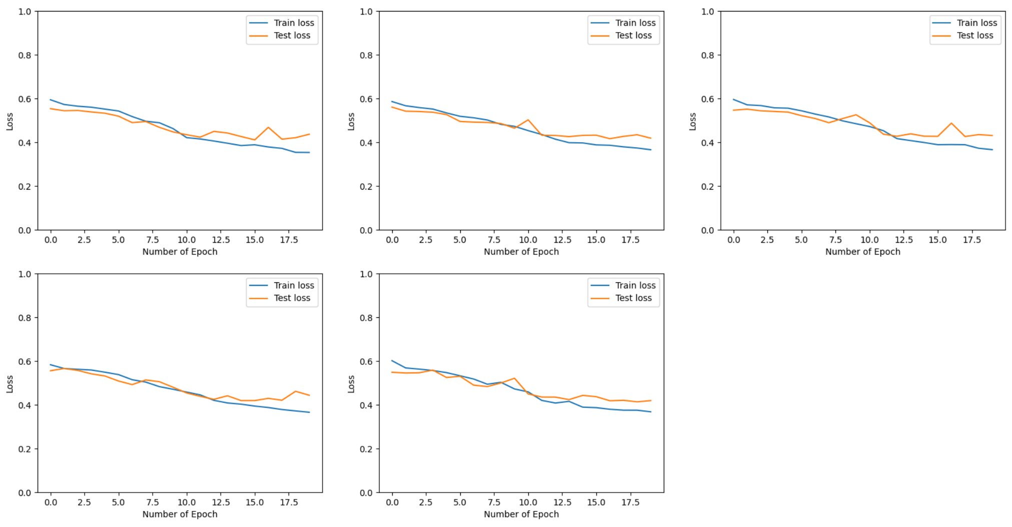


Figure 9: Number of Epoch vs the Train loss and Test loss in Dementia dataset

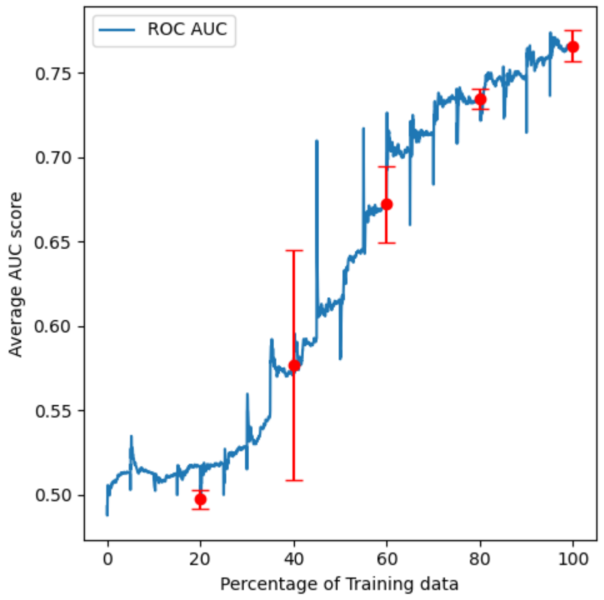


Figure 10: Percentage of Training data training vs the average training AUC score in Dementia dataset

Bibliography

[1] Junyu Luo, Muchao Ye, Cao Xiao, and Fenglong Ma. Hitanet: Hierarchical time-aware attention networks for risk prediction on electronic health records. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 647–656, 2020.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[3] Nir Menachemi and Taleah H Collum. Benefits and drawbacks of electronic health record systems. *Risk management and healthcare policy*, pages 47–55, 2011.

[4] National Center for Health Statistics and Centers for Medicare and Medicaid Services. International classification of diseases, ninth revision, clinical modification (icd-9-cm), 1979. Available from: https://www.cdc.gov/nchs/icd/icd9cm.htm.

[5] Suhan Cui, Junyu Luo, Muchao Ye, Jiaqi Wang, Ting Wang, and Fenglong Ma. Medskim: Denoised health risk prediction via skimming medical claims data. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 81–90. IEEE, 2022.

[6] Davood Karimi, Haoran Dou, Simon K Warfield, and Ali Gholipour. Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical image analysis*, 65:101759, 2020.

[7] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[8] Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. Patient subtyping via

time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 65–74, 2017.

[9] Lipo Wang. *Support vector machines: theory and applications*, volume 177. Springer Science & Business Media, 2005.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] Sarah A Pendergrass and Dana C Crawford. Using electronic health records to generate phenotypes for research. *Current protocols in human genetics*, 100(1):e80, 2019.

[12] Chrysanthi Papoutsi, Julie E Reed, Cicely Marston, Ruth Lewis, Azeem Majeed, and Derek Bell. Patient and public views about the security and privacy of electronic health records (ehrs) in the uk: results from a mixed methods study. *BMC medical informatics and decision making*, 15:1–15, 2015.

[13] Elham Mahmoudi, Neil Kamdar, Noa Kim, Gabriella Gonzales, Karandeep Singh, and Akbar K Waljee. Use of electronic medical records in development and validation of risk prediction models of hospital readmission: systematic review. *bmj*, 369, 2020.

[14] Benjamin A Goldstein, Ann Marie Navar, and Michael J Pencina. Risk prediction with electronic health records: the importance of model validation and clinical context. *JAMA cardiology*, 1(9):976–977, 2016.

[15] Richard M Scheffler and Daniel R Arnold. Projecting shortages and surpluses of doctors and nurses in the oecd: what looms ahead. *Health Economics, Policy and Law*, 14(2):274–290, 2019.

[16] Fenglong Ma, Yaqing Wang, Houping Xiao, Ye Yuan, Radha Chitta, Jing Zhou, and Jing Gao. A general framework for diagnosis prediction via incorporating medical code descriptions. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1070–1075. IEEE, 2018.

[17] Senthilkumar Mohan, Chandrasegar Thirumalai, and Gautam Srivastava. Effective heart disease prediction using hybrid machine learning techniques. *IEEE access*, 7:81542–81554, 2019.

[18] Thomas C Rindfleisch. Privacy, information technology, and health care. *Communications of the ACM*, 40(8):92–100, 1997.

[19] Stefano A Bini. Artificial intelligence, machine learning, deep learning, and cognitive computing: what do these terms mean and how will they impact health care? *The Journal of arthroplasty*, 33(8):2358–2361, 2018.

[20] Arkaitz Artetxe, Andoni Beristain, and Manuel Grana. Predictive models for hospital readmission risk: A systematic review of methods. *Computer methods and programs in biomedicine*, 164:49–64, 2018.

[21] Manal Alghamdi, Mouaz Al-Mallah, Steven Keteyian, Clinton Brawner, Jonathan Ehrman, and Sherif Sakr. Predicting diabetes mellitus using smote and ensemble machine learning approach: The henry ford exercise testing (fit) project. *PloS one*, 12(7):e0179805, 2017.

[22] Yao Zhou, Haonan Wang, Jingrui He, and Haixun Wang. From intrinsic to counterfactual: On the explainability of contextualized recommender systems. *arXiv preprint arXiv:2110.14844*, 2021.

[23] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. *Advances in neural information processing systems*, 28, 2015.

[24] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R Martin, Ming-Ming Cheng, and Shi-Min Hu. Attention mechanisms in computer vision: A survey. *Computational visual media*, 8(3):331–368, 2022.

[25] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in neural information processing systems*, 29, 2016.

[26] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine learning for healthcare conference*, pages 301–318. PMLR, 2016.

[27] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning internal representations by error propagation, 1985.

[28] Anthony Gillioz, Jacky Casas, Elena Mugellini, and Omar Abou Khaled. Overview of the transformer-based models for nlp tasks. In *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, pages 179–183. IEEE, 2020.

[29] Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

[30] Junyuan Shang, Tengfei Ma, Cao Xiao, and Jimeng Sun. Pre-training of graph augmented transformers for medication recommendation. *arXiv preprint arXiv:1906.00346*, 2019.

[31] Yikuan Li, Shishir Rao, José Roberto Ayala Solares, Abdelaali Hassaine, Rema Ramakrishnan, Dexter Canoy, Yajie Zhu, Kazem Rahimi, and Gholamreza Salimi-Khorshidi. Behrt: transformer for electronic health records. *Scientific reports*, 10(1):7155, 2020.

[32] Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1948.

[33] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, 27, 2014.

[34] Tian Bai, Shanshan Zhang, Brian L Egleston, and Slobodan Vucetic. Interpretable representation learning for healthcare via capturing disease progression through time. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 43–51, 2018.

[35] Bum Chul Kwon, Min-Je Choi, Joanne Taery Kim, Edward Choi, Young Bin Kim, Soonwook Kwon, Jimeng Sun, and Jaegul Choo. Retainvis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE transactions on visualization and computer graphics*, 25(1):299–309, 2018.

[36] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1903–1911, 2017.

[37] Saharon Rosset. Model selection via the auc. In *Proceedings of the twenty-first international conference on Machine learning*, page 89, 2004.

[38] Caren Marzban. The roc curve and the area under it as performance measures. *Weather and Forecasting*, 19(6):1106–1114, 2004.

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[40] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[42] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[43] World Health Organization. *The ICD-10 classification of mental and behavioural disorders: diagnostic criteria for research*, volume 2. World Health Organization, 1993.

[44] World Health Organization et al. Icd-11 for mortality and morbidity statistics (2018). 2018.

Academic Vita

Sirui Qi

EDUCATION

Bachelor of Science, Double major in Computer Science and Math 08/2019 – 08/2023
Schreyer Honors College at Pennsylvania State University State College, United State

PROFESSIONAL EXPERIENCE

Software Engineer Intern 06/2017 – 07/2017
Kamfu Technology Co. Ltd Foshan, China
● Maintenance and upgrade the software that offers civic service to the residents of the city
● Understand how the software offers civic service to the residents of the city

Learning Assistant and Grader 01/2021 – 05/2023
College of Engineering at Pennsylvania State University State College, United States
● Discrete Mathematics for Computer Science (CMPSC 360) in 2021 spring, Data Structures and Algorithms (CMPSC 465) in 2021 fall, Introduction to the Theory of Computation (CMPSC 464) from 2022 spring to current
● Grade the students' homework, quizzes, and exam in this course and offer study support to the students during Office Hours and ordinary times.

Research Assistant, Data Engineering 05/2023 – present
Smeal College of Business at Pennsylvania State University State College, United States
● Collect the market data from websites by web scraping technology.
● Offering the data support for the research in department of the marketing at Penn State.

SKILLS

C/C++, JAVA, Pascal, Python, JavaScript, SQL, Pytorch, Tensorflow, Jupyter etc. (Programming and tech. skills)

PROJECTS

Dashboard Tweaking 08/2022 – 12/2022
● Design the new dashboard for capstone students' alignment
● Upgrade the interface which becomes more user-friendly.
● Connect the dashboard with the database in real-time

Implement different models for POS tagging 02/2023 – 03/2023
● Make Hidden Markov model, logistic regression model, and multilayer perceptron model for doing the Part-of-speech(POS) tagging work.
● Personally response for doing the Hidden Markov model for POS tagging and tuning the parameters.

Creating the summarization of TED talks with text generation 04/2023 – 05/2023
●I created a model for summarizing TED talks based on the dataset with over 2000 TED talk transcripts.
●It is challenging to summarize due to the lengthy, speech-like structure of the talks. I opted for a seq2seq model with GRU layers composed of an encoder and decoder.
●The model indicates a level of functionality that can successfully summarize TED talks.

AWARDS

Top 500 in 81st William Lowell Putnam Mathematical Competition 02/2020
Due to the COVID-19 pandemic, the competition hold in an unofficial mode.

Top 50 in the ICPC East Central NA Regional Contest 11/2019
ACM/ICPC
Team name: Penn We