Deep Staging: An Interpretable Deep Learning Framework for Disease Staging

Liuyi Yao* — *University at Buffalo*, Buffalo, USA, liuyiyao@buffalo.edu
Zijun Yao — *IBM Research*, Yorktown Heights, USA, zijun.yao@ibm.com
Jianying Hu — *IBM Research*, Yorktown Heights, USA, jyhu@us.ibm.com
Jing Gao — *Purdue University*, West Lafayette, USA, jinggao@purdue.edu
Zhaonan Sun — *IBM Research*, Yorktown Heights, USA, zsun@us.ibm.com

***Abstract*—Disease staging aims to measure the development of disease that uses clinical criteria to qualitatively classify the course of illness. Staging is a critical task in many clinical scenarios, for example, it could be used to guide the therapy and care, predict the clinical outcomes, optimize the utilization of resources, etc. Recently, data-driven disease staging using massive observational data has attracted significant attentions in literature. However, it is a technically challenging task not only because it is an unsupervised job without professional guidance as traditional ways do, but also it is crucial to generate clinically meaningful explanations in addition to stage prediction. In this work, we propose an interpretable deep learning framework, named Deep Staging, for data-driven explainable disease staging. The proposed approach could not only predict the disease stages based on observational medical data, but also generate clinically relevant characterizations of the disease stage outputs. Experiments on a real-world healthcare dataset demonstrate the effectiveness of the proposed framework.**

***Index Terms*—Disease Staging, Interpretation, Deep Learning**

I. Introduction

Disease Staging is a classification system that uses diagnostic, etiology and pathophysiology findings for measuring the progression of a disease [12]. A proper staging analysis is significantly helpful for a board range of healthcare scenarios, such as designing clinical trials, evaluating the diagnostic efficiency of physicians, assessing the quality of care, and understanding the allocation of healthcare resources [14].

Traditionally, the staging of disease is developed by domain experts based on professional knowledge and experiences gained from clinical practices, and the developed stages are often separated based on the values of one or a few gold standard biomarkers. For example, the Global Initiative for Chronic Obstructive Lung Disease (GOLD) developed a 4-stage system for COPD based on Forced vital capacity (FVC) and Forced expiratory volume (FEV-1) values [24]. A staging system of Chronic Kidney Disease (CKD) was developed by the National Kidney Foundation, which separates the disease into five stages based on Estimated Glomerular Filtration Rate (eGFR) value. The resulting staging systems are guaranteed to produce clinically meaningful disease stages, and are straight forward for clinical interpretation. However, such staging practice usually requires long-term efforts to develop, and is based on the availability of widely acknowledged biomarkers. For diseases without well-studied biomarkers, it is difficult to develop a staging system. In addition, the staging efforts are often disease-specific, and experiences are not easily generalized from one disease to another.

Recent years have witnessed a rapid explosion of electronic health records (EHR) including diagnosis, prescription, lab test, and clinical notes. Such a wealth of clinical data has motivated the development of data-driven methods for automatic disease staging. Although not many data-driven methods directly targeted at the disease staging problem, several works have focused on a highly related topics. The Hidden Markov Models (HMMs) and its extensions have been intensively studied to learn discrete states for multiple diseases [31], [28], [26]. However, one of the major drawbacks of HMM is that the Markov property assumes the future state only depends on the current state instead of the rich past history, which limits its generalization in medical usage. Recently, several deep learning methods have been developed to leverage historical information when modeling disease progression, such as the Recurrent Neural Network (RNN), DoctorAI [5], RETAIN [6]. Deep learning models are, in general, more capable of handling the complex medical history. However, due to the "black box" issue, it is difficult to generate clinically meaningful interpretations directly from the models.

In this article, we propose an interpretable deep learning framework, named *Deep Staging*, towards the problem of data-driven disease staging. The goal is to develop a generalized staging methodology that can identify the multiple stages of diseases and produce clinically meaningful interpretations in a self-supervised way. Specifically, Deep Staging employs the memory network [16] to analyze the vital long-term historical information and generates the personalized stage transition matrix based on the patient's current and historical health conditions. Moreover, Deep staging learns the representation of the patient's disease stages to interpret individual progression of illness. Experiment on the real-world EHR dataset shows that Deep staging is advantageous to both distinguish meaningful disease stages and generate corresponding medical interpretations.

* Work done when Liuyi Yao interned at IBM Research.

II. Related Work

A. Disease Progression Modeling.

Disease progression modeling (DPM) is a critical topic for caring chronic diseases such as Alzheimer's Disease and Diabetes [22]. A quality staging of patients helps to achieve optimal outcomes through early diagnosis, intervention, as well as predicting personalized pathological trajectories [1]. Many efforts have been put to use machine learning techniques for modeling the progression of diseases using observational data. For example, [13] derived a tree model to infer tumor progression from a comparative genome hybridization dataset. [19] developed a probabilistic Gaussian process model to quantify the diagnostic uncertainty of severity on Alzheimer's Disease. [2] introduced a mixture model of trees to describe evolutionary processes characterized by permanent genetic changes. Due to the nature of modeling states, Hidden Markov model (HMM) [31], [28], [26] and related latent variable model [25] have attracted heavy interests for DPM. For example, [32] proposed a continuous-time Markov process based model to learn disease progression of chronic diseases in an unsupervised manner. More work includes state space based models [1] and group lasso based model [34].

B. Deep Learning on EHR Data.

The ubiquitous electronic health record (EHR) system has created an unprecedented value for medical analytics with large-scale, fine-grained, and standardized contexts of patients. These data enabled deep learning techniques to learn more sophisticated patterns and have shown impressive performance on a variety of critical tasks. For diagnosis prediction, Doctor AI [5] developed a two-layer recurrent neural network (RNN) to predict the time and diagnosis of next visit using current history. Dipole [20] presented a bidirectional RNN with attention mechanism aiming to learn the temporal patterns with both the past and future visits. BRITS [3] proposed a bidirectional RNN with the capability of accommodating missing value in EHR time-series by imputing values as training variables for improving prediction performance. For medical representation, Med2Vec [6] learns the distributed embeddings of diagnosis codes and visits using the co-occurrence of codes in the same visit and the adjacency of visits in EHR history. GRAM [7], MiME [9] introduced the hierarchical medical ontologies to learn graph-based attention mechanism for medical concept embedding. More recent work focus on incorporating interpretabilty into healthcare modeling for delivering transparent decision making. For example, RETAIN [8] makes interpretable outcome prediction by adopting two RNNs to learn the attentions of historical visit and in-visit diagnosis respectively for aggregating visit embeddings as input for prediction layer. GCT [10] proposed a Transformer [30] based graph neural network to learn the underlying structure of in-visit medical concepts by self-attention mechanism regularized by prior knowledge.

III. Methodology

A. Problem Definition.

The studied problem is to generate the disease stage at different timestamps to indicate the disease severity, given the current clinical observations. Thus the problem is defined as: **Input:** The clinical observation of i-th patient at t-th timestamp X_t^i, where $X_t^i \in \mathcal{X}$ and $\mathcal{X}$ is the records of all patients. **Output:** The disease stage probability vector P_t^i, where the j-th element of P_t^i is the probability of i-th patient at j-th stage at t-th timestamp.

For notation clarity, we omit the patient indicator in the following sections, i.e., the superscription i. Additionally, in the following, we use the terminology "disease stage", "stage" and "state" exchangeably.

B. Framework Overview.

The framework of the proposed Deep Staging is shown in figure 1. It contains three components: The memory controller, the state transition and the outcome prediction. The memory controller stores the long-term information. The state transition component calculates the state probability based on the current clinical observations and the retrieved information from the memory controller. The outcome prediction component aims to obtain a meaningful state representation. Specifically, when a new observation X_t comes, the hidden state h_t is generated through the memory controller and retrieves and updates the memory storage. Next, based on the combination of the hidden state h_t and the retrieved information, the transition matrix Q_t is obtained. The state probability P_t is calculated by multiplying the transition matrix Q_t and the state probability in the last timestamp P_{t-1}. To better understand the learned state, we further obtain the current observation representation based on the hidden state and the state probability. Then, the following tasks are applied: reconstruct the current observation X_t; predict the observation of the next visit X_{t+1}, and predict the time gap (Δ_t) between the current and the next visit. The following sections describe each component in detail.

C. Memory Controller.

Long-term information is important for the disease staging estimation, since, for chronic diseases, the long-term historical information affects the patient's current condition. To fully utilize the long-term information, the memory controller should refresh the stored memory when new observations come and retrieve useful information when estimating the disease stage. To this end, we adopt the differentiable neural computer(DNC)[16] as the memory network. The DNC memory network contains the following two parts: (1) controller network; (2) the interaction with the external memory, including memory reading and memory writing. In the following, we will briefly describe the above two parts.

We first denote the external memory at timestamp t as $\mathbf{M}_t$ with $\mathbf{M}_t \in \mathbb{R}^{N_{\text{slot}} \times N_M}$, where N_{slot} is the number of memory slots and N_M is the size of each slot.

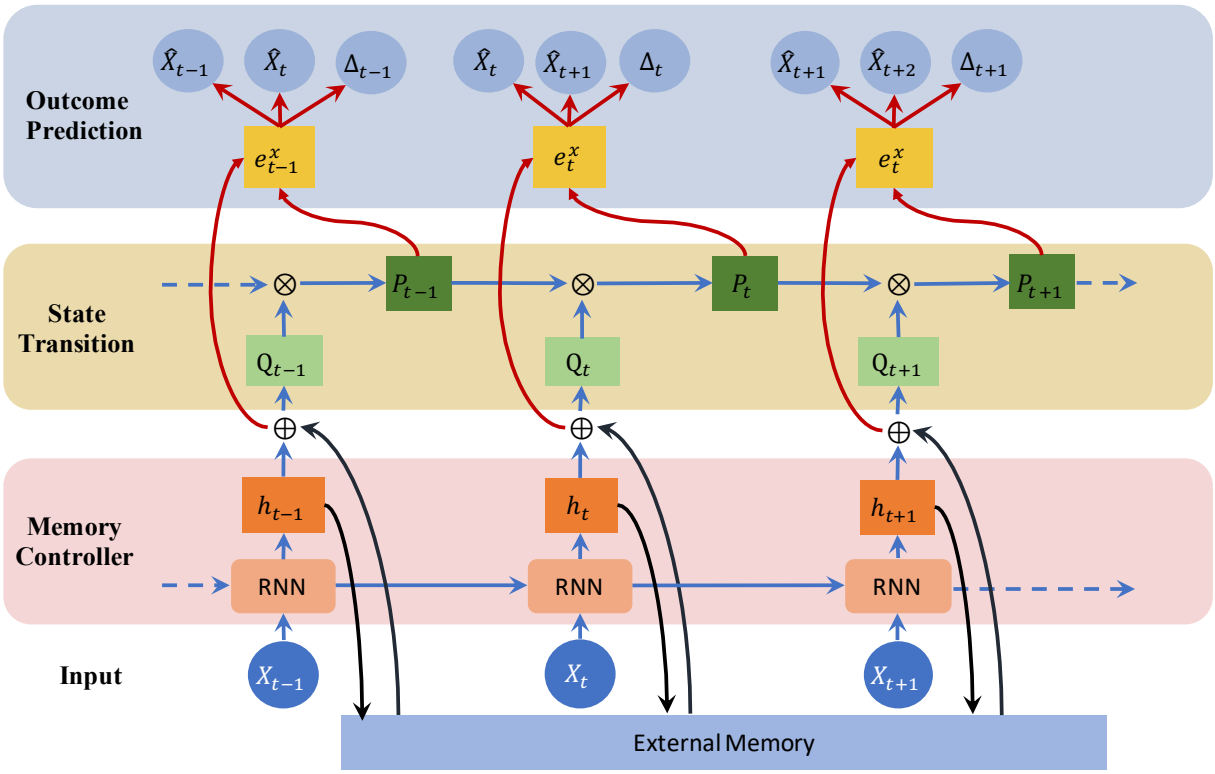


Fig. 1: Framework of Deep Staging Framework.

1) Controller Network:: The controller network receives the current observation and the hidden state of the previous timestamp. It then updates the hidden state of the network and generates the inferface vector which are the parameters used in the interaction with the external network. Specifically, the GRU [4] cell is adopted to update the hidden state:

$$\mathbf{h}_t = f_{\text{GRU}}(\mathbf{h}_{t-1}, X_t), \tag{1}$$

where f_{GRU} is the GRU cell, $\mathbf{h}_{t-1}$ is the hidden state in the previous timestamp, and X_t is the current observation.

After obtaining the hidden state of the current timestamp, the controller then generate the parameters responsible for the interaction with the external memory:

$$\{\mathbf{k}_t^l\}_{l=1}^R, \{\beta_t^{r,l}\}_{l=1}^R, \{\beta_t^{w,l}\}_{l=1}^R, \mathbf{r}_t, \mathbf{v}_t, \beta_t^{\text{write}} = f_{con}(\mathbf{h}_t), \tag{2}$$

where f_{con} is the linear function, $\{\mathbf{k}_t^l\}_{l=1}^R$ is the R reading keys, and $k_t^l \in \mathbb{R}^{N_M}$, $\beta_t^{r,l} \in \mathbb{R}$ and $\beta_t^{w,l} \in \mathbb{R}$ are the reading strength and writing strength separately, $\mathbf{r}_t \in \mathbb{R}^{N_M}$ and $\mathbf{v}_t \in \mathbb{R}^{N_M}$ are the erase vector and the write vector for memory updating, β_{write} is other memory updating parameters.

2) Memory Reading.: Memory Reading aims to retrieve the information from the external memory to support the downstream task. The memory reading is defined as:

$$\mathbf{\Phi}_t^l = \mathbf{w}_t^l \mathbf{M}_t, \tag{3}$$

where $l = 1, 2, \cdots, R$; $\mathbf{w}_l^m$ is the reading weight based on the similarity between the reading key and each memory slot, with its s-th element defined as:

$$\mathbf{w}_t^{l,s} = \frac{\exp\left(f_{\text{oneplus}}(\beta_t^{r,l}) D(\mathbf{k}_t^l, \mathbf{M}_t^s)\right)}{\sum_{j=1}^{N_{\text{slot}}} \exp\left(f_{\text{oneplus}}(\beta_t^{r,l}) D(\mathbf{k}_t^l, \mathbf{M}_t^j)\right)}, \tag{4}$$

where $s = 1, 2, \cdots, N_{\text{slot}}$; $\mathbf{M}_t^s \in \mathbb{R}^{N_M}$ is the s-th memory slot in the external memory $\mathbf{M}$; $f_{\text{oneplus}}(\cdot)$ is the one plus function, which is defined as: $f_{\text{oneplus}}(x) = 1 + \log(1 + e^x)$. The one plus function ensures the range of the reading strength is within $[1, \infty)$, and $D(\cdot, \cdot)$ is the cosine similarity function.

3) Memory Writing.: The external memory is updated by erasing the non-related information and adding the important information from the current observation. Mathematically, the external memory is updates as:

$$\mathbf{M_t} = (\mathbf{E} - \mathbf{w}_t^w \mathbf{r}_t) \odot \mathbf{M_{t-1}} + \mathbf{w}_t^w \mathbf{v}_t, \tag{5}$$

where E is the identity matrix, $\mathbf{w}_t$ is the writing weight calculated by the β_t^{write} (more details about $\mathbf{w}_t$ calculation, see [16]), $\mathbf{r}_t$ and $\mathbf{v}_t$ are the erasing and writing vector obtained from Eqn. (2).

D. State Transition.

The previous memory controller component extracts useful information from the current observations and long-term memory. With that extracted information, the state transition component outputs the state probability vector to indicate

the disease severity at the current timestamp. The state transition component mimics the classical HMM model, which multiplies the state probability at the previous timestamp with the transition matrix. The difference between those two models is that, in our proposed model, different patients at different timestamps have their transition matrix, which provides personalized disease staging modeling. In the following subsections, the transition matrix and state probability vector will be presented in detail.

1) Transition Matrix Calculation.: For personalized disease staging purposes, the transition matrix is updated based on the information provided by the memory controller, when a new timestamp comes. We assume there are total N_s disease stages. Therefore, the transition matrix $Q_t \in \mathbb{R}^{N_s \times N_s}$ and its element of i-th row and j-th column denotes the transition probability of j-th state to the i-th state. The transition matrix is calculated as:

$$Q_t^{ft} = \text{softmax}_{N_s}(f_{pos}(f_{ft}(A_t \times H_t))), \tag{6}$$

where

- Q_t^{ft} is the flattened transition matrix and $Q_t^{ft} \in \mathbb{R}^{N_s^2}$; The element of i-th row and j-th column in Q_t is the $((i-1) \times N_s + j)$-th element in Q_t^{ft}.
- H_t is the concatenated matrix of the memory reading vectors and the hidden state, and

$$H_t = [\mathbf{\Phi}_t^1, \cdots, \mathbf{\Phi}_t^R, \mathbf{h}_t] \in \mathbb{R}^{(R+1) \times d_h}, \tag{7}$$

  where $\mathbf{\Phi}_t^l$ is the vector retrieved by l-th reader from the external memory which is obtained from Eqn. (3); $\mathbf{h}_t$ is the hidden state of the memory controller; R is the number of the memory readers; d_h is the dimension of the hidden state, which is the same as the memory slot size, i.e., $d_h = N_M$.
- A_t is the structured self-attention weight, and $A_t \in \mathbb{R}^{N_a \times (R+1)}$. We adopt N_a attentions in order to capture different important parts of H_t [18]. Formally, A_t is defined as: $A_t = \text{softmax}(W_{A_2}\text{tanh}(W_{A_1}H_t^{'}))$, where $W_{A_1} \in \mathbb{R}^{N_{wa} \times d_h}$ and $W_{A_2} \in \mathbb{R}^{N_a \times N_{wa}}$ are the attention parameters.
- $f_{ft}(\cdot)$ is the flatten function which flat the matrix $A_t \times H_t \in \mathbb{R}^{N_a \times d_h}$ into $\mathbb{R}^{(N_a d_h)}$.
- $\text{softmax}_{N_s}(\cdot)$ denotes the softmax function applied to every N_s elements.
- $f_{W_q}(\cdot)$ denotes the fully-connected neural network with W_q as its parameter.

The final transition matrix Q_t is calculated by reshaping the Q_t^{ft} into the $N_s \times N_s$ matrix.

Attention Regularization. To reduce the redundancy of the attention, as suggested in [18], the following regularization is applied on the attention weight A_t":

$$\mathcal{L}_{att} = \sum_t ||A_t^{'}A_t - \mathbf{E}||_F^2, \tag{8}$$

where $\mathbf{E}$ denotes the identity matrix and $||\cdot||_F^2$ denotes the Frobenius norm of a matrix.

2) State Probability Calculation.: After obtaining the transition matrix, the state probability vector is calculated the same as the HMM model:

$$P_t = Q_t \times P_{t-1}, \tag{9}$$

where $P_{t-1} \in \mathbb{R}^{N_s}$ is the state probability at timestamp $t-1$, and Q_t is the transition matrix defined in Eqn. (6).

E. Outcome Prediction.

To make the learned state meaningful, the state probability vector, along with the hidden state, should support the downstream tasks. We design the following three tasks: (1) reconstruct the current observation; (2) predict the observation at the next timestamp; (3) predict the time gap between the current and the next observation.

To better support the downstream tasks, we first learn the representation of the patient's condition at the current timestamp:

$$e_t^x = f_{\text{rep}}\left([W_H H_t, P_t]; W_{\text{rep}}\right), \tag{10}$$

where H_t is the concatenated matrix of memory reading and the hidden state defined in Eqn. (7); $W_H \in \mathbb{R}^{1 \times (R+1)}$; $[W_H H_t, P_t] \in \mathbb{R}^{d_h + N_s}$ is the concatenation of the input two vectors; $f_{\text{rep}}(\cdot)$ denotes the neural network for representation learning with W_{rep} as its parameter.

With the representation of the current condition, we further adopt the neural network to do the reconstruction and the prediction:

$$\begin{aligned} \hat{X}_t &= f_{\text{rec}}(e_t^x; W_{\text{rec}}), \\ \hat{X}_{t+1} &= f_{\text{pred}}(e_t^x; W_{\text{pred}}), \\ \hat{\Delta}_t &= f_{\text{tp}}(e_t^x; W_{\text{tp}}), \end{aligned} \tag{11}$$

where $f_{\text{rec}}(\cdot)$, $f_{\text{rec}}(\cdot)$, f_{tp} are the neural works for current observation reconstruction, next observation and time gap prediction with W_{rec}, W_{pred}, and W_{tp} as their parameters, respectively.

F. Objective Function.

The overall objective function of the proposed method is:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{att}\mathcal{L}_{att} + \lambda_W ||\mathbf{W}||_2, \tag{12}$$

where

- $\mathcal{L}_{\text{task}}$ is the loss in the three tasks:

$$\begin{aligned} \mathcal{L}_{\text{task}} = \sum_{X_t \in \mathcal{X}} \Big[&\ell_{ce}(\hat{X}_t - X_t) + \ell_{ce}(\hat{X}_{t+1} - X_{t+1}) \\ &+ (\hat{\Delta}_t - \Delta_t)^2 \Big], \end{aligned} \tag{13}$$

  where $\mathcal{X}$ denotes the whole observation; ℓ_{ce} is the cross-entropy loss; X_t and X_{t+1} are ground-truth observation at the current and next timestamp; Δ_t is the ground-truth time gap between the current and the next observation.
- $\mathcal{L}_{att}$ is the attention regularization defined in Eqn. (8).
- $||\mathbf{W}||_2$ is the ℓ_2 regularization on all the neural network parameters excluding the intercept term.
- λ_{att} and λ_W are the hyper-parameters.

G. Implementation and Joint Optimization.

All the neural networks adopted in Deep Staging are standard feed-forward neural networks with Dropout [27] and ELU activation function [11]. The Adam [17] is adopted to optimize Eqn. (12).

IV. Experiments

In this section, we experiment on the CKD dataset to validate the following: (1) The proposed deep staging method has comparable performance with the-state-of-the-art predictive models in the prediction task. (2) The learned disease stages are meaningful by qualitative analysis.

A. Experiment Setup.

1) Dataset: The cohort used in this study came from a real-world longitudinal EHR database of over 300,000 patients over the course of 4 years. The data were collected from patients' billing information. During a patient's encounter with the EHR system, a set of International Classification of Diseases–Version 9 (ICD-9) codes were recorded to indicate the medical conditions the patient had at that time point. Other information, such as lab test results and medications, were also recorded. We identified a cohort of 3890 CKD patients based on the criterion that a patient was diagnosed with CKD (i.e., the patients with ICD-9 diagnosis code $585.XX$) at least once during his observation period. For each identified patient, his observations were segmented into non-overlapping 90-day windows using the first CKD diagnosis date as the index date. Records before and after the index date are all included. To ensure sufficient longitudinal information, we only include patients with at least one year record. At last, the cohort includes 3854 patients' records and a total of $52,313$ aggregated visits. The ICD-9 diagnosis codes of per visit are categorized according to the class listed in [33]. Table I lists more statistics about the CKD dataset.

# of patients	3854
# of visits	52,313
Avg. # of visits per patient	13.57
Max # of visits per patient	37
# of unique diagnosis codes	316
Avg. # of medical codes per visit	4.62
Max # of medical codes per visit	36
Max time gap	841
Min time gap	1
Median time gap	32

TABLE I: Statistics of CKD Dataset.

2) Baselines: The following state-of-the-art methods are adopted to compare the performance on the prediction task:
(1) RNN: RNN with GRU cell [15] is adopted as the comparison method. We feed the observation at the current timestamp into the GRU cell and predict the observation at the next timestamp using the hidden state produced by GRU cell.
(2) Doctor AI [5]: Doctor AI is a predictive RNN model with GRU cell in healthcare. It takes the concatenation of the multi-hot observation vector and the duration since the last event as the input, and predicts both the codes at the next visit and the time gap between the current and the next visit. The Doctor AI is initialized using the Skip=gram embeddings [21].
(3) Med2Vec [6]: Med2Vec efficiently learns the medical code representation following the idea of Skip-gram [21]. It predicts the diagnosis code appearance in the next timestamp based on the representation of the current observation.
(4) RETAIN [8]: RETAIN is an interpretable RNN model with a reverse-time attention mechanism, a visit-level and variable-level attention model. The target label's prediction can be explained by the importance weight of variables generated by the two-level attention mechanism. We change the final prediction layer of RETAIN to the softmax layer so that it can predict the appearance of multiple diagnosis codes.

3) Evaluation Metric: As suggested by [20], [8], we adopt the visit-level recall@k as the evaluation metric, which is defined as:

$$\text{Recall@}k = \frac{\text{\# of correct codes ranked in the top } k}{\text{\# of codes in this visit}}. \quad (14)$$

The higher recall@k is, the better the performance. In this experiment, we set k as 30, 20, 10.

4) Reproducibility: We randomly divide the dataset into training, validation and testing set with a split ratio of $0.6/0.1/0.3$. The validation set is used to select the best parameters. We implement the Deep Staging with Pytorch 1.3.1 [23]. The state probability vector is initialized as that in the beginning, all patients are in the stage 1. The batch size is set as 50, the learning rate is 0.001, the number of iteration is 100, and λ_W is set as 0.001. The external memory is set as 4 memory slot with each slot has size 20 (i.e., $N_{\text{slot}} = 4$, $N_M = 20$). The number of state N_s is set as 4. Table II summarizes the searching space of other hyper-parameters.

Hyper-parameter	Range
λ_{att}	$[0, 0.05, 0.1]$
R	$[2, 3, 4]$
N_a	$[1, 2, 3]$
# of hidden layers in the neural network	$[2, 3]$
dimension of each hidden layer	$[20, 50]$
dimension of e_t^x	$[30, 50, 100]$

TABLE II: Hyper-parameter Searching Space

All baseline methods share the same hyper-parameters search space with Deep Staging. The drop-out strategies with a drop-out rate of 0.5 are applied to all approaches.

B. Results Analysis.

Table III reports the mean and standard deviation of the recall@k over all the visits in the test set. The table shows that our proposed deep staging model has comparable results with RNN, Doctor AI, and RETAIN models on the prediction task. Although Med2Vec has the highest recall@k value, it lacks the interpretability. The results show that the proposed framework has great potential because it has good predictive performance, compared with the existing prediction model,

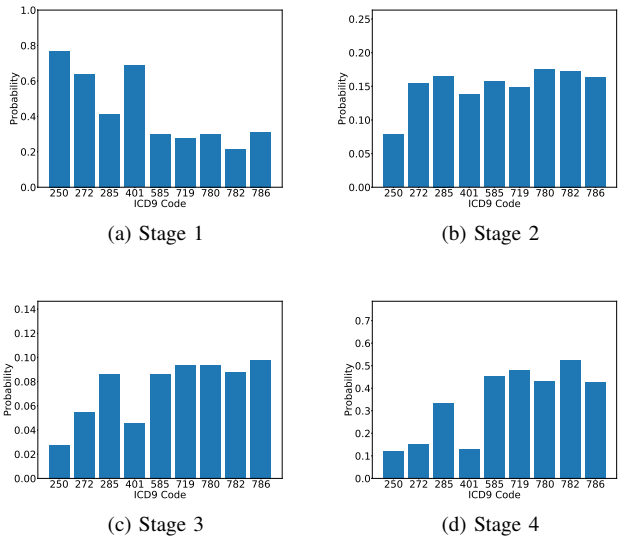
and reveals progression pathways to better understand the diseases.

Methods	Recall@30	Recall@20	Recall@10
Deep Staging	0.609 ± 0.314	0.524 ± 0.326	0.385 ± 0.319
RNN	0.602 ± 0.315	0.505 ± 0.330	0.378 ± 0.317
Doctor AI	0.599 ± 0.325	0.505 ± 0.335	0.374 ± 0.320
Med2Vec	0.707 ± 0.294	0.633 ± 0.316	0.518 ± 0.331
RETAIN	0.640 ± 0.313	0.555 ± 0.327	0.434 ± 0.329

TABLE III: Performance on CKD Dataset.

C. State Interpretation.

To uncover the inner meaning of the learned disease stages, we show the probabilities of a group of ICD-9 codes in the learned stages in Figure 2. From the figure, it is observed that the distributions of some critical ICD-9 codes differ in the four stages, indicating that the framework is capable of separating different typical status along the progression pathway. The probabilities also provide a clinical description of each learned stage in terms of the most common diagnosis in the corresponding stage.



(a) Stage 1 (b) Stage 2

(c) Stage 3 (d) Stage 4

Fig. 2: ICD-9 Codes First Appearing Probability.

To further explore the capability of separating different stages, we visualize the corresponding patient condition representation (i.e., e_t^x) by t-SNE [29]. For each stage, 100 representations are randomly selected, and decomposed. Figure 3 shows the results. The figure shows that within the same stage, the representations are clustered, and between the different stages, there is a clear difference in patients' condition representations. Those observations validate the ability to separate different stages.

D. Case Study.

To better demonstrate that the proposed deep staging framework can reflect the disease severeness, we provide a case study to show how the stage probability changes when critical diagnosis code appears. We randomly select one patient within the group where each patent has 5 visits due to the space limitation. Table IV shows the clinical event and the corresponding stage probability at each visit. At first, the patient is in stage 1. When the codes, 250, which is diabetes-related code 401, which is hypertension-related code, appear, the probability of stage 2 increases. In visit 5, the probability of stage 3 increases with critical codes' appearance: Chronic renal failure 585. An interesting observation is that at visit 3 and visit 4, the patient has the same clinical events, and their corresponding stage 2 probability is very similar. The above observations illustrate that the disease stage probability changes are consistent with the critical code appearances, which validates the Deep Staging model's ability to reveal the disease severeness.

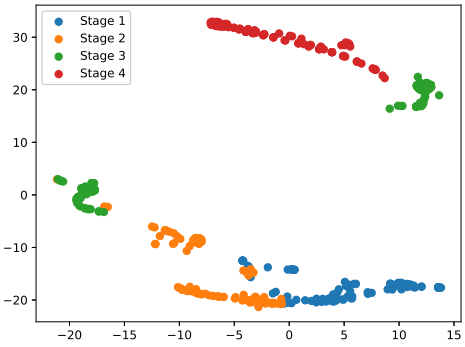


Fig. 3: Visualization of Patients' Representation (e_t^x) at Different Stages.

V. Conclusions

In healthcare, disease staging modeling is a vital and challenging task. Many existing disease staging modeling works either adopt the Hidden Markov Model or employ deep learning techniques, such as recurrent neural networks (RNNs). However, HMM-based models require the restrict Markov property, and RNN based approaches either cannot fully utilize the historical information or are unable to generate meaningful clinical interpretations. In this paper, we propose a novel framework, named Deep Staging, to overcome the challenges of disease staging modeling. By employing the memory network, Deep Staging can utilize the important historical information stored in the external memory. The personalized transition matrix learning provides the flexibility for disease staging modeling. Experimental results on the CKD dataset demonstrate that the proposed Deep Staging model is capable of separating different typical disease status with meaningful clinical descriptions.

VI. Appendix

Table V shows the meaning of the ICD9 codes mentioned in Figure 2.

Visit	Clinical Event	Stage 1	Stage 2	Stage 3	Stage 4
1	367: Disorders of refraction and accommodation	**0.785**	0.215	0	0
2	250: Diabetes mellitus; 272: Disorders of lipoid metabolism; 401: Essential hypertension; V76: Special screening for malignant neoplasms;	**0.616**	0.328	0.057	0
3	808: Fracture of pelvis; 401: See above; 820: Fracture of neck of femur; 599: Other disorders of urethra and urinary tractk;	**0.483**	0.374	0.130	0.013
4	401: See above; 808: See above; 599: See above; 808: See above;	0.378	**0.380**	0.200	0.042
5	401: See above; 808: See above; 820: See above; 780: General symptoms; 410: Acute myocardial infarction; E888: Accidental falls; 276: Disorders of fluid, electrolyte, and acid-base balance; 427: Cardiac dysrhythmias; 585: Chronic renal failure.	0.297	**0.361**	0.255	0.087

TABLE IV: The Clinical Events and The Stage Probability in Each Visit.

ICD9 Code	Meaning
250	Diabetes mellitus
272	Disorders of lipoid metabolism
285	Other and unspecified anemias
401	Essential hypertension
585	Chronic renal failure
719	Other and unspecified disorders of joint
780	General symptoms
782	Symptoms involving skin and other integumentary tissue
786	Symptoms involving respiratory system and other chest symptoms

TABLE V: ICD9 Code Meaning in Figure 2.

REFERENCES

[1] A. M. Alaa and M. van der Schaar. Attentive state-space modeling of disease progression. In *Advances in Neural Information Processing Systems*, pages 11338–11348, 2019.

[2] N. Beerenwinkel, J. Rahnenführer, M. Däumer, D. Hoffmann, R. Kaiser, J. Selbig, and T. Lengauer. Learning multiple evolutionary pathways from cross-sectional data. *Journal of computational biology*, 12(6):584–598, 2005.

[3] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li. Brits: Bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems*, pages 6775–6785, 2018.

[4] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[5] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*, pages 301–318, 2016.

[6] E. Choi, M. T. Bahadori, E. Searles, C. Coffey, M. Thompson, J. Bost, J. Tejedor-Sojo, and J. Sun. Multi-layer representation learning for medical concepts. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1495–1504, 2016.

[7] E. Choi, M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 787–795, 2017.

[8] E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 3504–3512, 2016.

[9] E. Choi, C. Xiao, W. Stewart, and J. Sun. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. In *Advances in Neural Information Processing Systems*, pages 4547–4557, 2018.

[10] E. Choi, Z. Xu, Y. Li, M. Dusenberry, G. Flores, E. Xue, and A. Dai. Learning the graphical structure of electronic health records with graph convolutional transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 606–613, 2020.

[11] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[12] J. E. Conklin, J. V. Lieberman, C. A. Barnes, and D. Z. Louis. "disease staging: Implications for hospital reimbursement and management". *Health Care Financing Review*, Suppl(Suppl):13–22, 1984.

[13] R. Desper, F. Jiang, O.-P. Kallioniemi, H. Moch, C. H. Papadimitriou, and A. A. Schäffer. Inferring tree models for oncogenesis from comparative genome hybridization data. *Journal of computational biology*, 6(1):37–51, 1999.

[14] J. S. Gonnella, M. C. Hornbrook, and D. Z. Louis. "staging of disease: A case-mix measurement". *JAMA*, 251(5):637–644, 02 1984.

[15] A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[16] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations, ICLR*, 2017.

[19] M. Lorenzi, M. Filippone, G. B. Frisoni, D. C. Alexander, S. Ourselin, A. D. N. Initiative, et al. Probabilistic disease progression modeling to characterize diagnostic uncertainty: application to staging and prediction in alzheimer's disease. *NeuroImage*, 190:56–68, 2019.

[20] F. Ma, R. Chitta, J. Zhou, Q. You, T. Sun, and J. 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.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[22] D. Mould. Models for disease progression: new approaches and uses. *Clinical Pharmacology & Therapeutics*, 92(1):125–131, 2012.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019.

[24] A. R. Patel, A. R. Patel, S. Singh, S. Singh, and I. Khawaja. Global initiative for chronic obstructive lung disease: the changes made. *Cureus*, 11(6), 2019.

[25] P. Schulam and S. Saria. A framework for individualizing predictions of disease trajectories by exploiting multi-resolution structure. In *Advances in Neural Information Processing Systems*, pages 748–756, 2015.

[26] K. A. Severson, L. M. Chahine, L. Smolensky, K. Ng, J. Hu, and S. Ghosh. Personalized input-output hidden markov models for disease progression modeling. volume 126 of *Proceedings of Machine Learning Research*, pages 309–330. PMLR, 07–08 Aug 2020.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[28] Z. Sun, S. Ghosh, Y. Li, Y. C. ang Amrita Mohan, C. Sampaio, and J. Hu. A probabilistic disease progression modeling approach and its application to integrated huntington's disease observational data. *JAMIA Open*, 2019.

[29] L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[31] X. Wang, D. Sontag, and F. Wang. Unsupervised learning of disease progression models. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 85–94, 2014.

[32] X. Wang, D. Sontag, and F. Wang. Unsupervised learning of disease progression models. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 85–94, 2014.

[33] Wikipedia contributors. List of icd-9 codes — Wikipedia, the free encyclopedia, 2019. [Online; accessed 11-October-2020].

[34] J. Zhou, J. Liu, V. A. Narayan, and J. Ye. Modeling disease progression via fused sparse group lasso. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1095–1103, 2012.