Stable Learning for AI Health - Preliminary

I have been interested in *causal representation learning* for health and business problems.

Authors	Title	Submission
Liu, Z., **Luo, Y.,** Zheng, D., Liu, Q., Zhong, R., Chang, D., Kong, D., Chen, Z.	Deep Stable Multi-Interest Learning for Sequential Recommendation	KDD 22’
Luo, Y., Liu, Z., Liu, Q.	Deep Stable Representation Learning on Electronic Health Records	KDD 22’

Microsoft

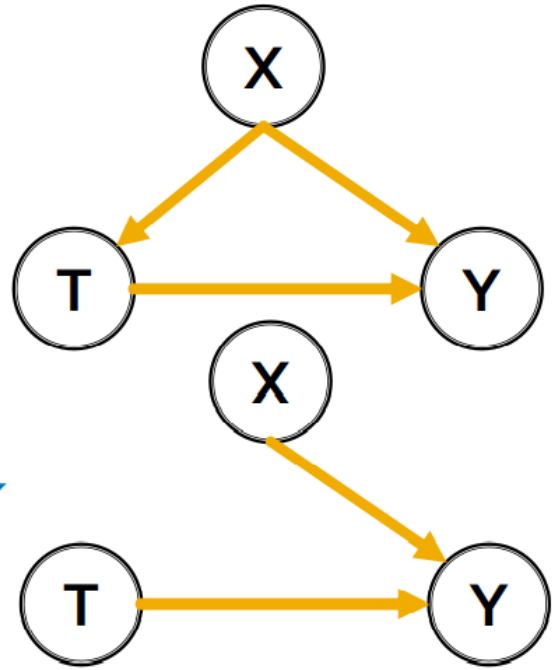
UNIVERSITY of WASHINGTON

Some Backgrounds on Causal Inference

Example: Treatment, Outcome and Confound

Goal: Estimate effect of a treatment T on an outcome Y
But, confound X influences both T and Y
To estimate $T \rightarrow Y$, break the dependence $X \rightarrow T$ (that is, $T \perp\!\!\!\perp X$)

Randomized experiments actively assign treatment T independent of any confound X
Thus, by construction: $T \perp\!\!\!\perp X$



Some Backgrounds on Causal Inference

Many challenges make causal inference hard to adapt to real-world applications.

1. Conditional Independence Assumption (CIA) and Ignorability

   $T \perp X$ for both observed and unobserved covariates, $(Y_0, Y_1) \perp T \mid X = x$

2. Estimation of Propensity Score for Sample Weighting (to mimic RTC)

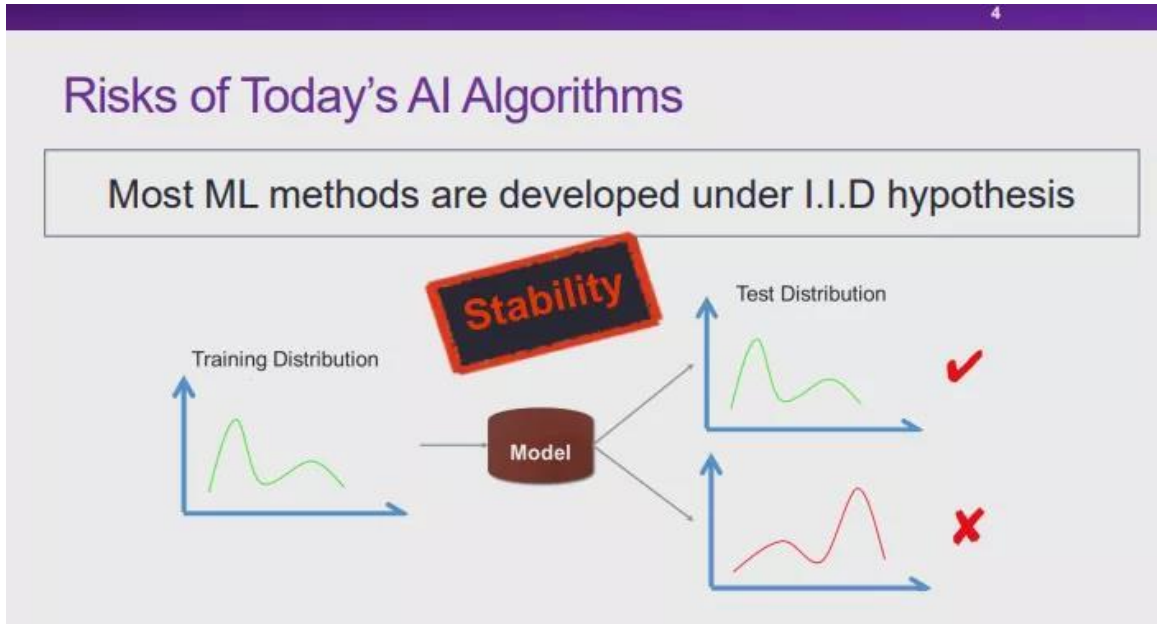
$$P(y|do(T=t)) = \sum_x P(y|t,x)P(x) = \sum_x \frac{P(y|t,x)P(t|x)P(x)}{P(t|x)} = \sum_x \frac{P(y,t,x)}{P(t|x)}$$

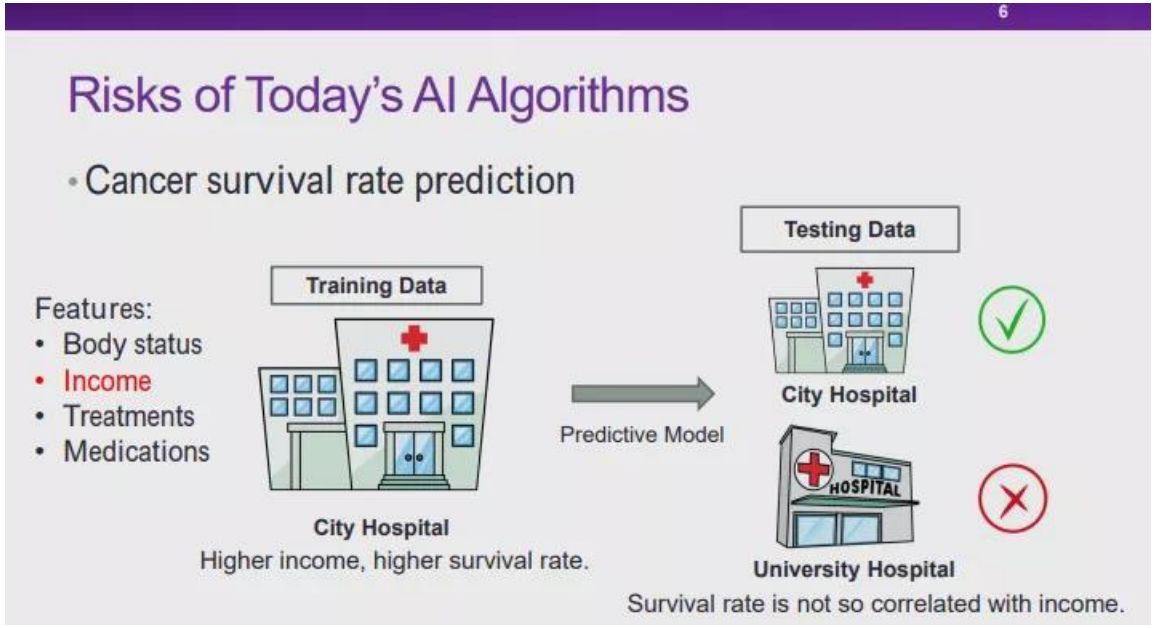
Propensity estimation accuracy: What if there are numerous Ts with limited data?

Some Backgrounds on OOD

Training and test distributions are not I.I.D.(independent and identically distributed)

Most OOD methods (domain adaption, causal transfer learning, etc.) need test distribution.





Stable Learning as Causation + OOD

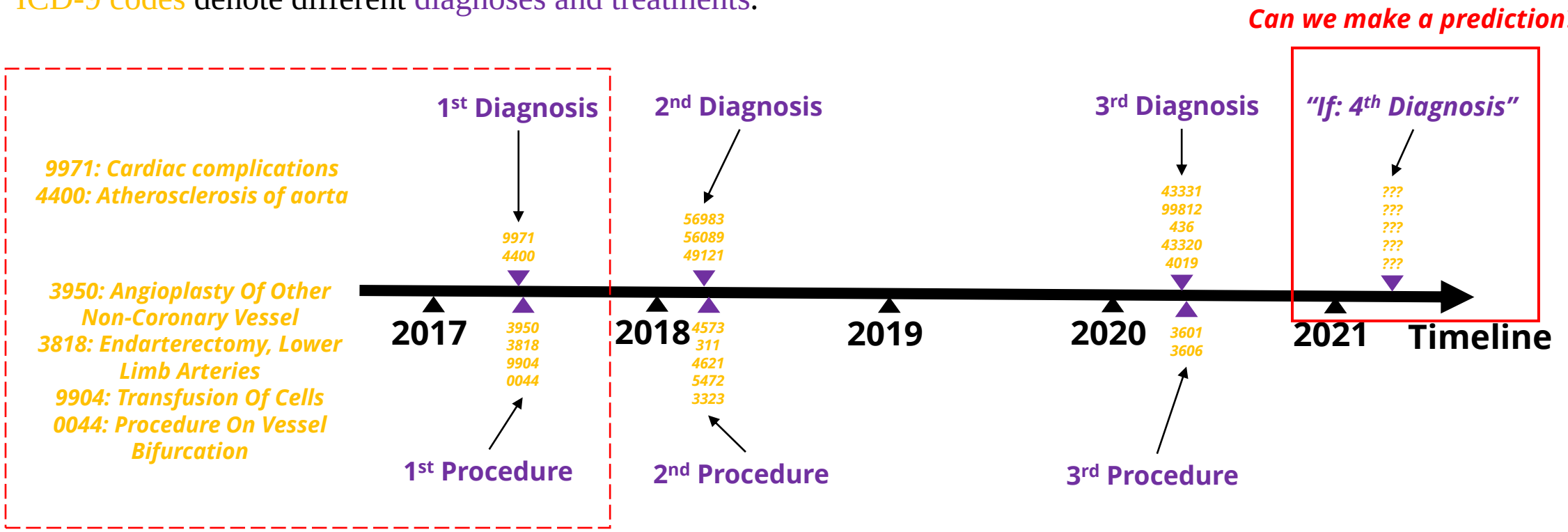
Stable Learning aims at *removing spurious relations* and *make stable prediction across distributions*.

Authors	Title	Conference
Kun Kuang, et al.	Stable Prediction across Unknown Environments	KDD 18’
Zheyan Shen, Peng Cui, Tong Zhang, Kun Kuang.	Stable Learning via Sample Reweighting	AAAI 20’
Zheyean Shen, et al.	Stable Learning via Differentiated Variable Decorrelation	KDD 20’
Xingxuan Zhang, et al.	Deep Stable Learning for Out-Of-Distribution Generalization	CVPR 21’

We adapt this technique (previously on linear models) to deep learning scheme for applications.

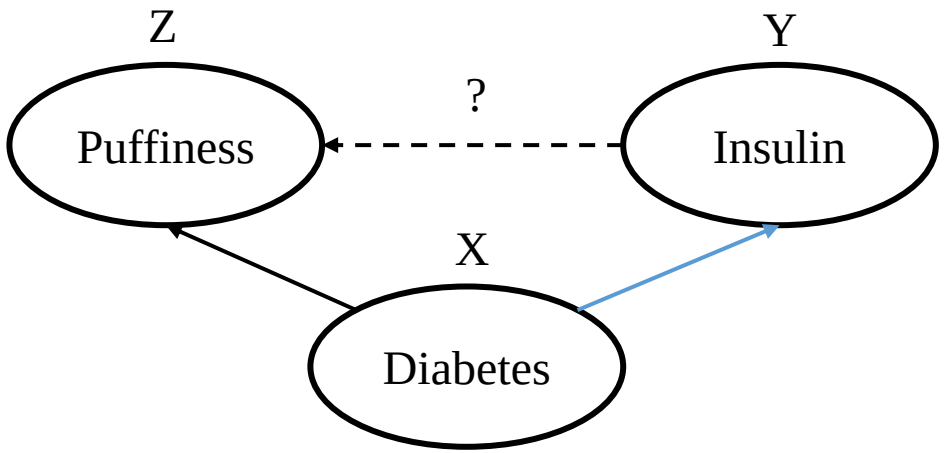
Diagnoses Prediction based on EHR

Sequential Electronic Health Records (EHR) with diagnosis and procedure information.
ICD-9 codes denote different diagnoses and treatments.



Causal Diagram on EHR

In this case, Y → Z would be spurious.



Patients with diabetes can take insulin (X → Y).
Diabetes may cause complication puffiness (X → Z).
Insulin does not cause puffiness (Z ⊥ Y).
Statistical models may learn "Insulin → Puffiness".

Here, X, Y, Z only denote variables.

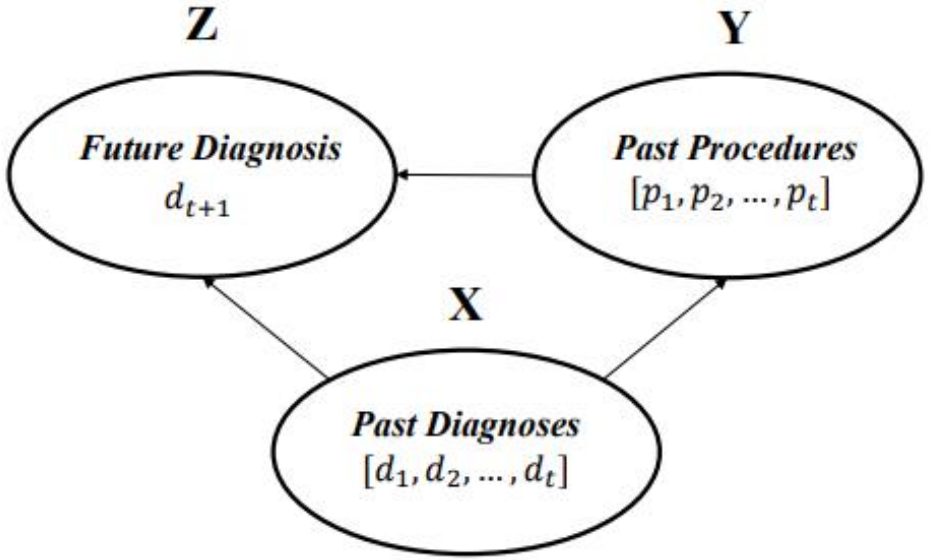
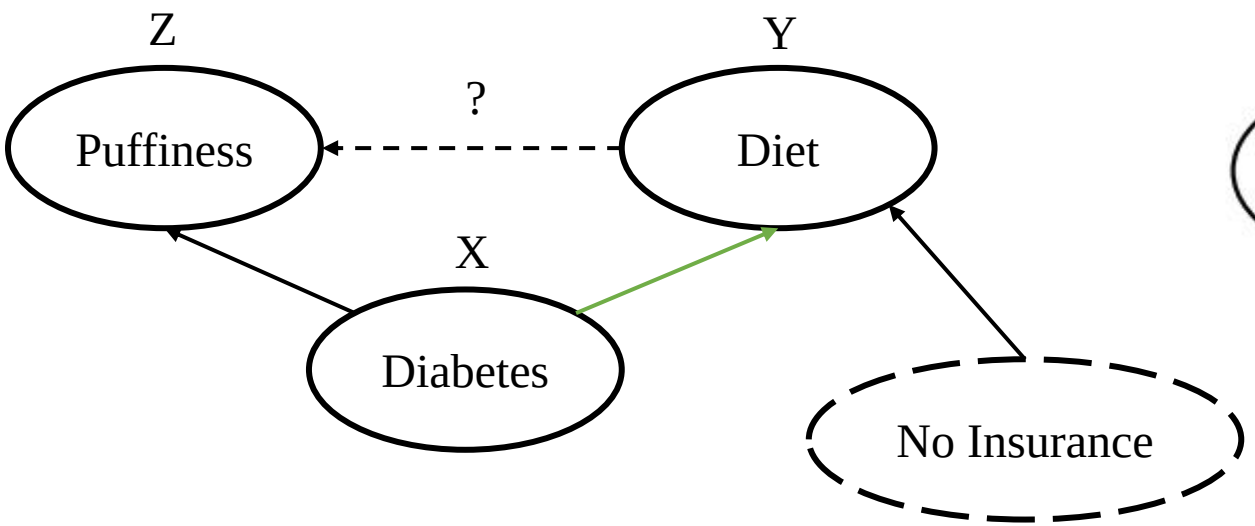


Figure 1: The causal diagram of diagnosis prediction in EHR.

Causal Diagram on EHR

Spurious Y → Z cannot generalize.



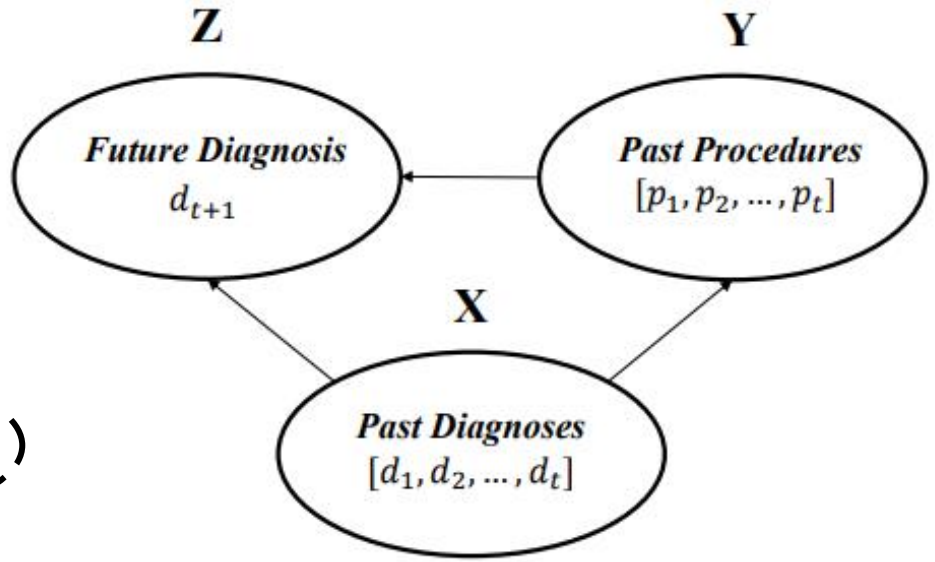
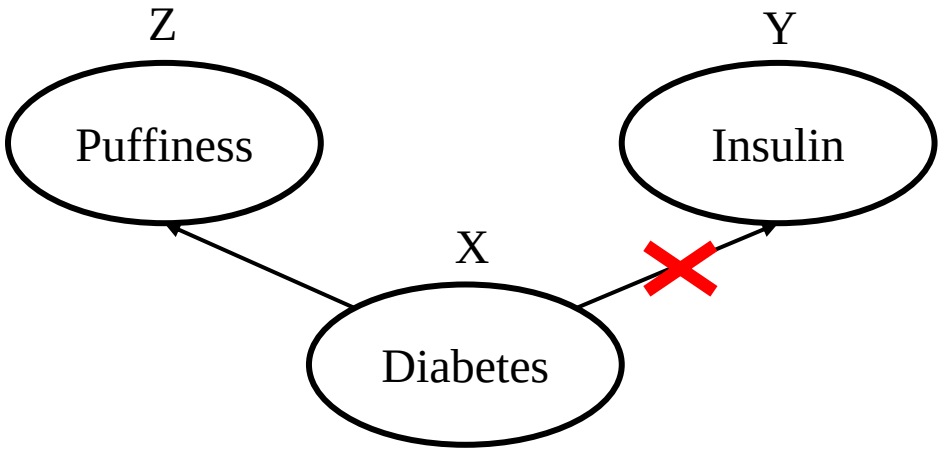


Figure 1: The causal diagram of diagnosis prediction in EHR.

X → Z is causal and stable in various environments.
When X → Y change (distribution shift),
statistical models may not generalize well.

Stable Learning on EHR

Spurious Y → Z will be removed.



Causal inference requires the **conditional covariance** between Treatment and Covariates to be zero. Here, X and Y shouldn't be correlated to remove spurious relation.

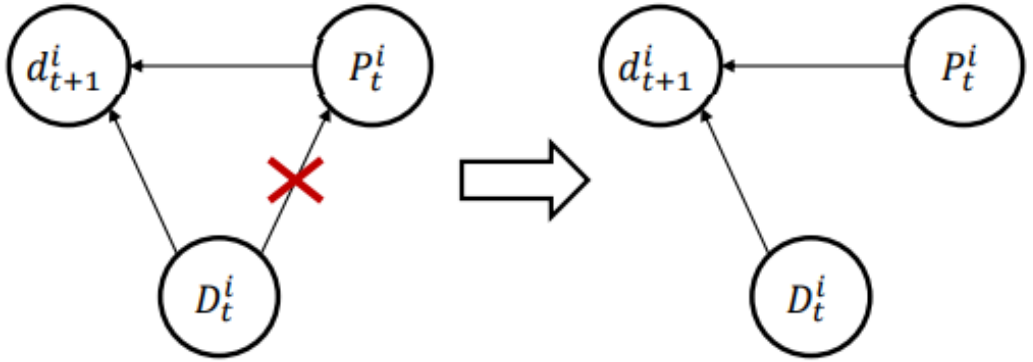


Figure 2: Decorrelation between past diagnoses D^i_t and past procedures P^i_t, for more accurate and stable prediction of future diagnosis d^i_{t+1}.

Stable Learning on EHR

Stable Learning aims at *removing spurious relations* and *make stable prediction across distributions*.

Q: How to measure **conditional covariances** between diagnoses and procedures?

A: Hilbert Schmidt Independence Criterion (HSIC) [1,2].

Q: How to minimize **conditional covariances** during/before model training?

A: Sample weighting (just like inverse propensity weighting).

[1] Daniel Greenfeld and Uri Shalit. 2020. Robust learning with the hilbert-schmidt independence criterion. In ICML. 3759–3768.

[2] Arthur Gretton, Kenji Fukumizu, Choon Hui Teo, Le Song, Bernhard Schölkopf, Alexander J Smola, et al. 2007. A kernel statistical test of independence.. In NeurIPS. 585–592.

What is HSIC?

- We are familiar with mutual information

$$I(X,Y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy \qquad (1)$$

- If $I(X,Y) = 0$, $p(x,y) \equiv p(x)p(y)$, X and Y are independent.

- HSIC is an independence testing statistics as the Hilbert-Schmidt norm of the cross-covariance operator between the distributions in Reproducing Kernel Hilbert Space (RKHS).
- Covariance: the joint variability of two random variables.

The computation of HSIC

> The calculation of cross-covariance matrix.

$$C[f,g] = \iint p(x,y)f(x)g(y)dxdy - \iint p(x)p(y)f(x)g(y)dxdy \tag{2}$$
$$= \mathbb{E}_{(x,y)\sim p(x,y)}[f(x)g(y)] - \mathbb{E}_{x\sim p(x)}[f(x)]\mathbb{E}_{y\sim p(y)}[g(y)]$$

> Here, f and g make up vector spaces F and G, respectively.
> C, the cross-covariance operator, maps F → G.
> Hint: whether X can transform to Y via any operators.

The computation of HSIC

> Sampling enough f and g from F and G to approximate

$$L_H = \sum_{f,g} \left(C[f,g]\right)^2 \tag{3}$$

$$\begin{aligned}\left(C[f,g]\right)^2 =& \left(\mathbb{E}_{(x,y)\sim p(x,y)}[f(x)g(y)]\right)^2 + \left(\mathbb{E}_{x\sim p(x)}[f(x)]\right)^2 \left(\mathbb{E}_{y\sim p(y)}[g(y)]\right)^2 \\ & - 2\left(\mathbb{E}_{(x,y)\sim p(x,y)}[f(x)g(y)]\right)\left(\mathbb{E}_{x\sim p(x)}[f(x)]\right)\left(\mathbb{E}_{y\sim p(y)}[g(y)]\right)\end{aligned} \tag{4}$$

> The value of expectation is equal across different samples

$$\mathbb{E}_{x\sim p(x)}[f(x)] = \mathbb{E}_{x'\sim p(x')}[f(x')]$$

The computation of HSIC

> Therefore, we can convert the square into two samples.

$$\begin{aligned}\left(\mathbb{E}_{x\sim p(x)}[f(x)]\right)^2 =& \left(\mathbb{E}_{x_1\sim p(x)}[f(x_1)]\right)\left(\mathbb{E}_{x_2\sim p(x)}[f(x_2)]\right) \\ =& \mathbb{E}_{x1\sim p(x),x2\sim p(x)}[f(x_1)f(x_2)]\end{aligned} \tag{5}$$

$$\begin{aligned}\left(C[f,g]\right)^2 =& \mathbb{E}_{(x_1,y_1)\sim p(x,y),(x_2,y_2)\sim p(x,y)}[f(x_1)f(x_2)g(y_1)g(y_2)] \\ &+ \mathbb{E}_{x1\sim p(x),x2\sim p(x),y1\sim p(y),y2\sim p(y)}[f(x_1)f(x_2)g(y_1)g(y_2)] \\ &- 2\mathbb{E}_{(x1,y1)\sim p(x,y),x2\sim p(x),y2\sim p(y)}[f(x_1)f(x_2)g(y_1)g(y_2)]\end{aligned} \tag{6}$$

> Now, we find it is hard to transverse F and G for estimation.

The computation of HSIC

> **No worries! The kernel trick can reduce calculation time!**

$$K(x_1, x_2) = \sum_i \alpha_i \psi_i(x_1) \psi_i(x_2) \tag{9}$$

$$\begin{aligned} L_H =& \mathbb{E}_{(x1,y1)\sim p(x,y),(x2,y2)\sim p(x,y)} \left[\sum_{i,j} \alpha_i \beta_j \psi_i(x_1) \psi_i(x_2) \phi_j(y_1) \phi_j(y_2) \right] \\ &+ \mathbb{E}_{x1\sim p(x),x2\sim p(x),y1\sim p(y),y2\sim p(y)} \left[\sum_{i,j} \alpha_i \beta_j \psi_i(x_1) \psi_i(x_2) \phi_j(y_1) \phi_j(y_2) \right] \\ &- 2\mathbb{E}_{(x1,y1)\sim p(x,y),x2\sim p(x),y2\sim p(y)} \left[\sum_{i,j} \alpha_i \beta_j \psi_i(x_1) \psi_i(x_2) \phi_j(y_1) \phi_j(y_2) \right] \end{aligned} \tag{12}$$

The computation of HSIC

- In essence, we have the HSIC:

$$\begin{aligned} HSIC(X,Y) =& \mathbb{E}_{(x_1,y_1)\sim p(x,y),(x_2,y_2)\sim p(x,y)} \left[K_X(x_1,x_2)K_Y(y_1,y_2)\right] \\ &+ \mathbb{E}_{x_1\sim p(x),x_2\sim p(x),y_1\sim p(y),y_2\sim p(y)} \left[K_X(x_1,x_2)K_Y(y_1,y_2)\right] \\ &- 2\mathbb{E}_{(x_1,y_1)\sim p(x,y),x_2\sim p(x),y_2\sim p(y)} \left[K_X(x_1,x_2)K_Y(y_1,y_2)\right] \end{aligned} \tag{13}$$

- With K being any kernel functions, we will have

$$HSIC(X,Y) = 0 \Leftrightarrow p(x,y) \equiv p(x)p(y) \tag{18}$$

- The time complexity is dependent on the size of X and Y.

Stable Learning on EHR

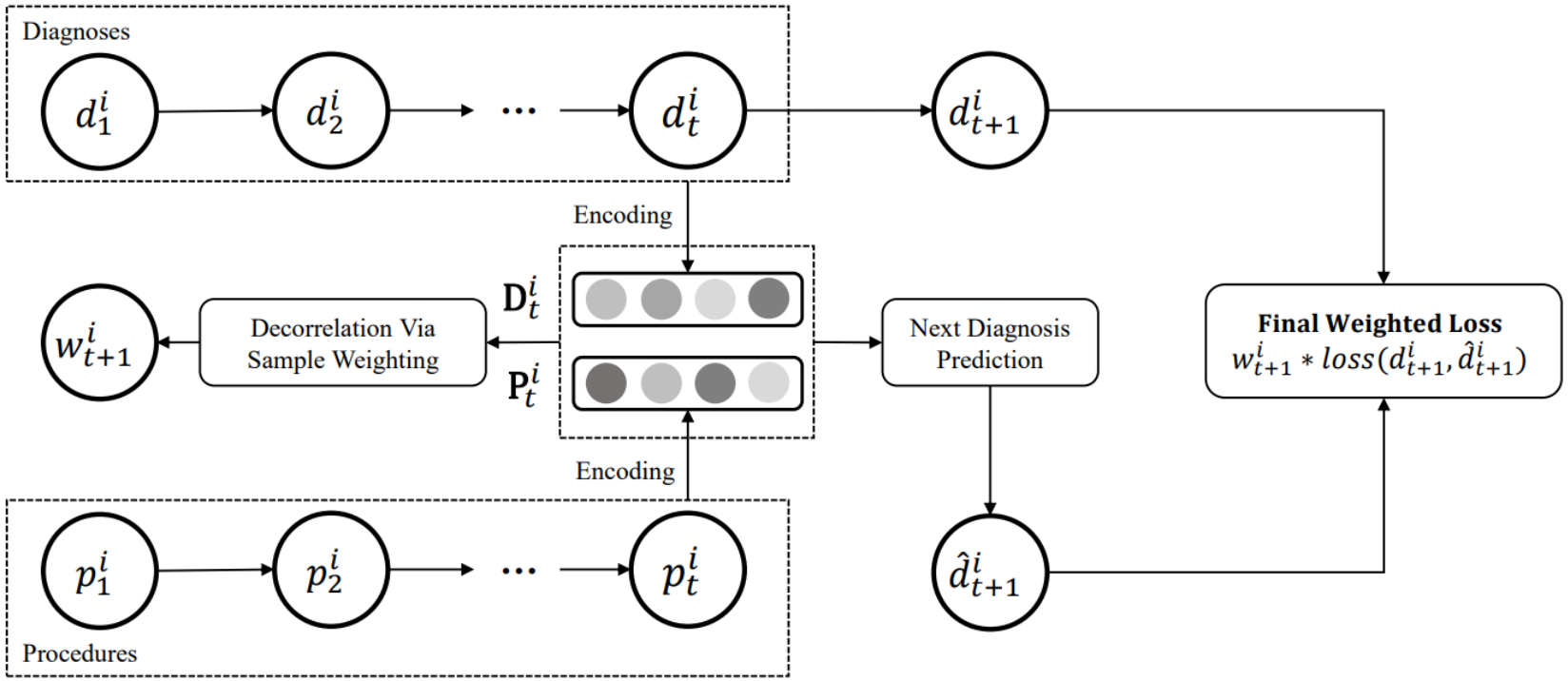


Figure 3: The schematic diagram of decorrelation between past diagnoses D_t^i and past procedures P_t^i via sample weighting.

Stable Learning on EHR

3.1 Problem Formulation

In the EHR data, we have a set of patients $V = \{v_1, v_2, ..., v_{|V|}\}$, and patient v_i has t^i visits. Diagnoses and procedures are both represented in International Classification of Diseases, Ninth Revision (ICD-9)[1] medical codes, where we have M unique diagnosis medical codes and N unique procedure medical codes. For each patient v_i after t visits, there exists a historical diagnosis sequence $D_t^i = [d_1^i, d_2^i, ..., d_t^i]$ and a historical procedure sequence $P_t^i = [p_1^i, p_2^i, ..., p_t^i]$. Each diagnosis and procedure are M-dimensional multi-hot vector and N-dimensional multi-hot vector respectively, which means that $d_j^i \in \{0,1\}^M$ and $p_j^i \in \{0,1\}^N$, where $1 \le j \le t$. In this work, we would like to predict future diagnoses, i.e., predicting what diseases a patient will have in the future, based on historical EHR. Specifically, in this work, given D_t^i and P_t^i, we need to predict future diagnosis d_{t+1}^i.

- First, we map diagnoses and procedures to embedded latent space (for deep learning).

$$\mathbf{D}_t^i = Encoder\left(D_t^i\right), \tag{1}$$

$$\mathbf{P}_t^i = Encoder\left(P_t^i\right), \tag{2}$$

For diagnosis prediction, we can learn a deep learning model $f(\cdot)$ that satisfies

$$\mathbf{d}_{t+1}^i = f\left(\mathbf{D}_t^i, \mathbf{P}_t^i\right). \tag{3}$$

Stable Learning on EHR

The squared Hilbert-Schmidt norm of the cross-covariance operator Σ_{DP} can be approximated by the unbiased calculation in the embedding space as

$$HSIC(\mathbf{D}, \mathbf{P}) = \frac{1}{|V| \cdot (t^i - 1)} \sum_{i=1}^{|V|} \sum_{t=1}^{t^i - 1} HSIC_{local}(\mathbf{D}_t^i, \mathbf{P}_t^i). \quad (12)$$

Specifically, if r denotes the hidden dimensionality, we can consider calculating the HSIC of each $\mathbf{D}_t^i \in \mathbb{R}^r$ and $\mathbf{P}_t^i \in \mathbb{R}^r$ by

$$HSIC_{local}(\mathbf{D}_t^i, \mathbf{P}_t^i) = \frac{1}{(n-1)^2} Tr(K_d J K_p J), \quad (13)$$

where Tr is the trace of a matrix, $J = 1/n$, K_D and K_P are any kernel matrices. We can consider RBF kernel to calculate

$$K_d(x_1, x_2) = \exp\left(\frac{\|x_1 - x_2\|_2^2}{\sigma^2}\right), \quad (14)$$

where $x_1, x_2 \in \mathbf{D}_t^i \in \mathbb{R}^r$ represent the values in different dimensions of the latent representation. Similatly, there is $K_p(x_1, x_2)$ where $x_1, x_2 \in \mathbf{P}_t^i \in \mathbb{R}^r$ represent different dimensions of the latent

- This means, we decorrelate each time's diagnosis and procedure for each patient.

Inspired by feature decorrelation techniques [47, 48], we propose to minimize HSIC by sample weighting to mitigate the dependency between diagnoses and procedures in the embedded space. We use w_t^i to denote the weight for patient i at the t-th visit. We denote the weighted samples as $\mathbf{WD}$ and $\mathbf{WP}$, where the weighted samples of patient i at the t-th visit, i.e., $\mathbf{WD}_t^i$ and $\mathbf{WP}_t^i$, are

$$\mathbf{WD}_t^i = w_t^i \, \mathbf{D}_t^i, \quad (15)$$

$$\mathbf{WP}_t^i = w_t^i \, \mathbf{P}_t^i \, . \quad (16)$$

To minimize the correlation between diagnoses and procedures, we propose to optimize w with HSIC as follows

$$w^* = \underset{w}{\operatorname{argmin}} \, HSIC(\mathbf{WD}, \mathbf{WP}). \quad (17)$$

Microsoft

UNIVERSITY of WASHINGTON

Stable Learning on EHR

Overall, we iteratively optimize the weighted loss and the HSIC by

$$Enc_{n+1}, Prd_{n+1} = \underset{Enc,Prd}{\operatorname{argmin}} \sum_{i=1}^{|V|} \sum_{t=1}^{t^i-1} w_t^i(n)\mathbf{L}_t^i, \tag{18}$$

where

$$\mathbf{L}_t^i = L(Prd(Enc(D_t^i), Enc(P_t^i)), d_{t+1}^i), \tag{19}$$

and

$$w(n+1) = \underset{w}{\operatorname{argmin}}\, \epsilon \cdot HSIC(wEnc_{n+1}(D), wEnc_{n+1}(P)). \tag{20}$$

Here, L denotes the cross-entropy loss function. Enc represents the encoder that maps diagnoses and procedures into the embedding space. Prd represents the final prediction layer that maps the latent representation into the one-hot probability vector. The architectures of Enc and Prd depend on the base model our method is used upon. Enc_n, Prd_n and $w(n)$ indicates encoder, final prediction layer and sample weights at the n-th iteration, and $w(0)$ is initially set as ones. ϵ is a coefficient that balances the learning rates for updating the neural network and sample weights.

Algorithm 1 The training process of CHE.

Require: Set of patients $V = \{v_1, v_2, ..., v_{|V|}\}$, each v_i's diagnosis sequence $D_{t^i}^i = \left[d_1^i, d_2^i, ..., d_{t^i}^i\right]$ and procedure sequence $P_{t^i}^i = \left[p_1^i, p_2^i, ..., p_{t^i}^i\right]$, maximum epoch $Epoch$, and any BaseModel.

Ensure: Model parameters $Enc()$ and $Prd()$ in the BaseModel.

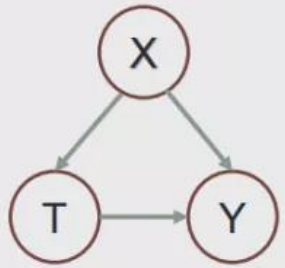
Initialize epoch indicator $n \leftarrow 0$.
Initialize best epoch indicator $n_{best} \leftarrow 0$.
Initialize model parameters $Enc_0()$ and $Prd_0()$ randomly.
Initialize weights $w_t^i(0) \leftarrow 1$, where $1 \le i \le |V|$ and $1 \le t \le t^i - 1$.
while early-stopping not reached and $n < Epoch$. **do**
    Update model parameters $Enc_{n+1}()$ and $Prd_{n+1}()$ according to Eq. (18), while keeping $w(n)$ fixed.
    Update weights $w(n+1)$ according to Eq. (20), while keeping $Enc_{n+1}()$ and $Prd_{n+1}()$ fixed.
    $n \leftarrow n + 1$.
    Update $n_{best} \leftarrow n$, if better result achieved on the validation set.
end while
return $Enc_{n_{best}}()$ and $Prd_{n_{best}}()$.

Microsoft

UNIVERSITY of WASHINGTON

Recap on Stable Learning

Revisit Directly Balancing for causal inference



Typical Causal Framework

Directly Confounder Balancing

Given a feature T

↓

Assign different weights to samples so that the samples with T and the samples without T have similar distributions in X

↓

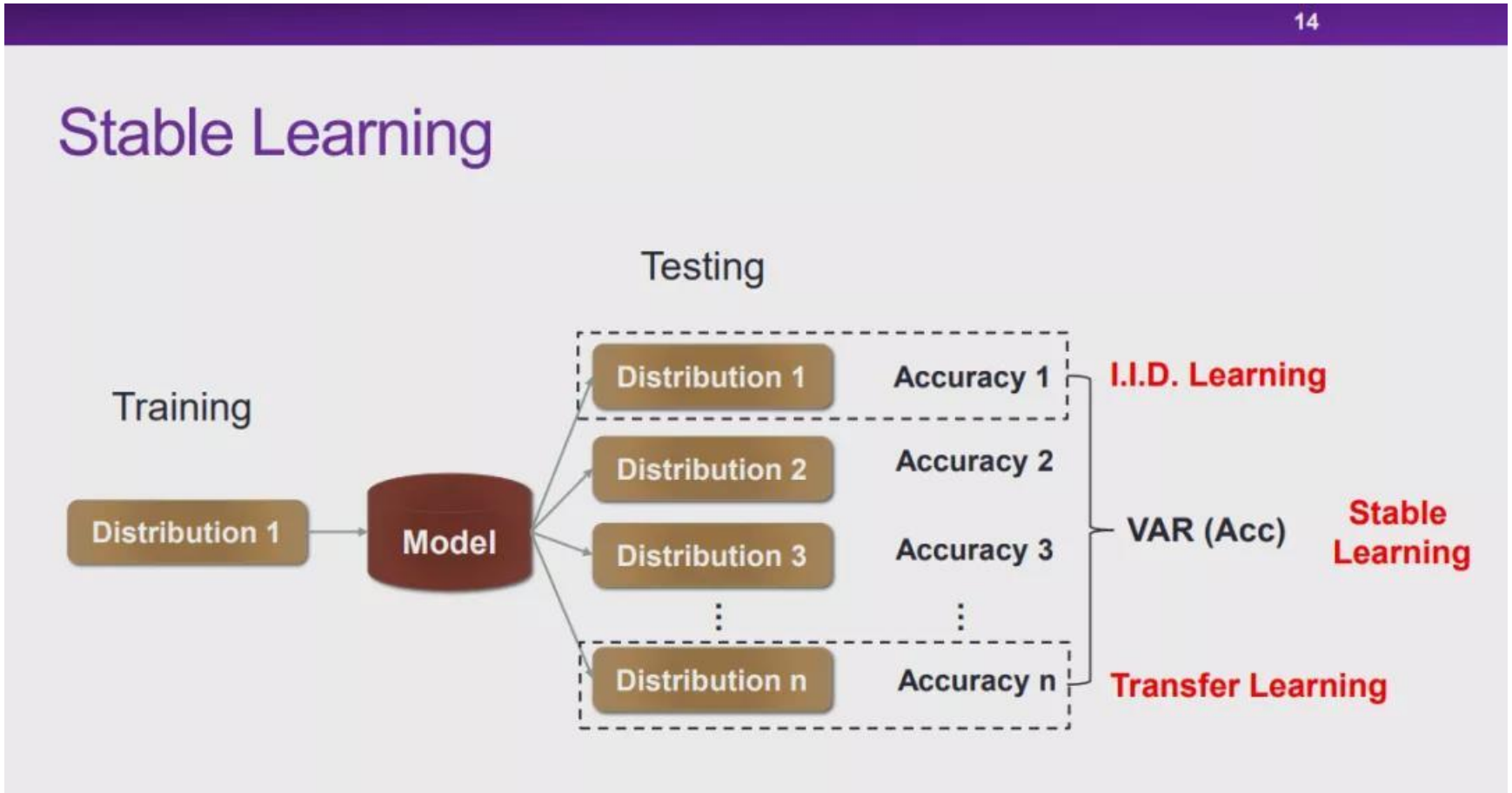
Calculate the difference of Y distribution in treated and controlled groups. (correlation between T and Y)

Sample reweighting can make a variable independent of other variables.

Microsoft

UNIVERSITY of WASHINGTON

Recap on Stable Learning



Experimental Datasets & Baselines on EHR

We evaluate our proposed sequential counterfactual learning method on two real-world datasets: MIMIC-III and MIMIC-IV.

- **MIMIC-III Dataset** We use diagnoses and procedures data from the Medical Information Mart for Intensive Care (MIMIC-III) database[2] [27], which contains patients who stayed in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012. Patients who had less than three admission records are excluded. After this preprocessing, the average number of visits for the 1970 selected patients is 3.69, the average number of codes in a visit is 13.23, the total number of unique ICD-9 codes in diagnoses is 3320, and the total number of unique ICD-9 codes in procedures is 988.
- **MIMIC-IV Dataset** We use diagnoses and procedures data from the Medical Information Mart for Intensive Care (MIMIC-IV) database[3] [27], which contains patients admitted to an ICU or the emergency department between 2008 and 2019. Patients who had less than three admission records are excluded. After this preprocessing, the average number of visits for the 10023 selected patients is 4.64, the average number of codes in a visit is 14.12, the total number of unique ICD-9 codes in diagnoses is 6274, and the total number of unique ICD-9 codes in procedures is 1973.

- **LSTM:** [24] A recurrent neural network with long-short term gating mechanism.
- **RETAIN:** [11] A two-level neural model based on reverse time attention for healthcare.
- **Dipole:** [35] An attention-based bidirectional recurrent neural network for healthcare.
- **Concare:** [37] A self-attention model that uses cross-head decorrelation to capture health context for healthcare.
- **StageNet:** [18] A deep learning model with stage-aware LSTM and convolutional modules for health risk prediction.

We denote above models as **BaseModels**, and we incorporate them with the **CHE** method as **CHE+BaseModels**.

While the historical EHR in the original dataset is naturally regarded as positive samples, we randomly generate negative samples that do not exist in the dataset and estimate their propensity scores via PW learning. Because the combination of various ICD-9 codes in a sequence is a large space as discussed in Section 3.2, we generate as many negative samples as possible (ten times larger than the original dataset) to make sure that the propensity estimation is as accurate as possible. We also incorporate PW with the above BaseModels, and name them as **PW+BaseModels**. For each PW+BaseModel, the propensity scores are calculated via training the corresponding BaseModel with permutation.

Experimental Setting on EHR

We conduct two diagnosis prediction experiments. In the first experiment, we aim at evaluating the performance of our proposed method when training data and test data are divided randomly by patients to approximately simulate I.I.D. distributions. Following prior works [11, 35], We randomly divide the dataset into the training, validation and testing set in a 0.75:0.1:0.15 ratio. In the second experiment, considering the insurance type, such as Medicare, Medicaid and Private, may affect procedures for similar diagnoses, we evaluate the performance when training data and test data are divided by the type of insurances to simulate the scenario of out-of-distribution generalization. Here, we divide all the Medicare data into the training and validation set in a 0.7:0.3 ratio and use the Private/Other (MIMIC-III/MIMIC-IV) data as the test set.

5.4 Evaluation Metrics

We adopt the top$k$ accuracy and normalized discounted cumulative gain (NDCG) to evaluate the diagnosis prediction performance. We use the same accuracy@k metric used in prior works [12, 35, 57], which is defined as the correct medical ICD-9 codes ranked in top$k$ divided by $\min(k, |y_t|)$, where $|y_t|$ is the number of ICD-9 codes in the $(t+1)$-th visit. NDCG@k further considers the normalization of gains and the ranking of correct medical codes, where codes with higher relevance will affect the final score more than those with lower relevance. In our experiments, we use $k \in [10, 20]$.

Experimental Results on EHR

Table 1: With random data division, performances of BaseModels, PW+BaseModels and CHE+BaseModels. Best performances are indicated by bold fonts. The improvement indicates the relative increase of CHE+BaseModel over BaseModel. $*$ denotes significant improvement of CHE+BaseModel, measured by t-test with p-value<0.01, over BaseModel and PW+BaseModel.

Approach	MIMIC-III				MIMIC-IV			
	NDCG@10	NDCG@20	ACC@10	ACC@20	NDCG@10	NDCG@20	ACC@10	ACC@20
LSTM	0.2648	0.2712	0.1779	0.2597	0.3469	0.3386	0.2167	0.3084
PW+LSTM	0.2669	0.2724	0.1791	0.2605	0.3488	0.3394	0.2177	0.3040
CHE+LSTM	**0.2756***	**0.2809***	**0.1853***	**0.2690***	**0.3589***	**0.3496***	**0.2246***	**0.3186***
Improv %	4.079%	3.577%	4.160%	3.581%	3.459%	3.249%	3.646%	3.307%
RETAIN	0.3409	0.3413	0.2305	0.3261	0.4095	0.3946	0.2568	0.3533
PW+RETAIN	0.3436	0.3449	0.2316	0.3241	0.4120	0.3981	0.2580	0.3527
CHE+RETAIN	**0.3545***	**0.3579***	**0.2353***	**0.3354***	**0.4231***	**0.4085***	**0.2630***	**0.3614***
Improv %	3.989%	4.864%	2.082%	2.852%	3.321%	3.523%	2.414%	2.293%
Dipole	0.3071	0.3104	0.2075	0.2959	0.3801	0.3710	0.2379	0.3352
PW+Dipole	0.3072	0.3110	0.2077	0.2965	0.3860	0.3754	0.2388	0.3376
CHE+Dipole	**0.3308***	**0.3342***	**0.2189***	**0.3120***	**0.4054***	**0.3932***	**0.2523***	**0.3529***
Improv %	7.717%	7.668%	5.494%	5.441%	6.656%	5.984%	6.053%	5.280%
Concare	0.2963	0.2979	0.1949	0.2793	0.3748	0.3615	0.2346	0.3226
PW+Concare	0.2972	0.2980	0.1952	0.2798	0.3720	0.3602	0.2335	0.3234
CHE+Concare	**0.3068***	**0.3121***	**0.2076***	**0.2935***	**0.3876***	**0.3760***	**0.2444***	**0.3371***
Improv %	3.544%	4.767%	6.516%	5.084%	3.415%	4.011%	4.177%	4.495%
Stagenet	0.3364	0.3379	0.2284	0.3222	0.3979	0.3853	0.2513	0.3471
PW+Stagenet	0.3343	0.3362	0.2267	0.3210	0.3960	0.3861	0.2529	0.3476
CHE+Staegnet	**0.3432***	**0.3467***	**0.2315***	**0.3295***	**0.4064***	**0.3976***	**0.2559***	**0.3541***
Improv %	2.021%	2.604%	1.357%	2.266%	2.136%	3.192%	1.830%	2.017%

NDCG@k relatively increases by 4.15%, and ACC@k relatively increases by 3.70%.

Experimental Results on EHR

Table 2: Under out-of-distribution data, performances of BaseModels, PW+BaseModels and CHE+BaseModels. Best performances are indicated by bold fonts. The improvement indicates the relative increase of CHE+BaseModel over BaseModel. $*$ denotes significant improvement of CHE+BaseModel, measured by t-test with p-value<0.01, over BaseModel and PW+BaseModel.

Approach	MIMIC-III				MIMIC-IV			
	NDCG@10	NDCG@20	ACC@10	ACC@20	NDCG@10	NDCG@20	ACC@10	ACC@20
LSTM	0.2082	0.2112	0.1420	0.1971	0.4395	0.4149	0.2519	0.3571
PW+LSTM	0.2075	0.2102	0.1413	0.2012	0.4468	0.4221	0.2525	0.3598
CHE+LSTM	**0.2182***	**0.2184***	**0.1492***	**0.2064***	**0.4649***	**0.4390***	**0.2692***	**0.3792***
Improv %	4.948%	3.409%	5.070%	4.718%	5.779%	5.809%	6.868%	6.189%
RETAIN	0.2385	0.2447	0.1615	0.2364	0.5195	0.4859	0.3019	0.4173
PW+RETAIN	0.2396	0.2443	0.1614	0.2362	0.5251	0.4888	0.3042	0.4195
CHE+RETAIN	**0.2503***	**0.2589***	**0.1687***	**0.2492***	**0.5413***	**0.5061***	**0.3126***	**0.4329***
Improv %	4.948%	5.803%	4.458%	5.415%	4.196%	4.157%	3.544%	3.738%
Dipole	0.2287	0.2367	0.1482	0.2264	0.4741	0.4469	0.2770	0.3873
PW+Dipole	0.2402	0.2481	0.1567	0.2354	0.4727	0.4433	0.2796	0.3865
CHE+Dipole	**0.2729***	**0.2772***	**0.1782***	**0.2631***	**0.5176***	**0.4905***	**0.3067***	**0.4280***
Improv %	19.33%	17.11%	20.24%	16.21%	9.175%	9.756%	10.72%	10.51%
Concare	0.2139	0.2229	0.1398	0.2139	0.4910	0.4616	0.2857	0.3974
PW+Concare	0.2122	0.2216	0.1414	0.2146	0.4947	0.4628	0.2878	0.4002
CHE+Concare	**0.2199***	**0.2310***	**0.1521***	**0.2264***	**0.5270***	**0.4944***	**0.3087***	**0.4267***
Improv %	2.805%	3.634%	8.798%	5.844%	7.332%	7.106%	8.050%	7.373%
Stagenet	0.2149	0.2224	0.1456	0.2171	0.5722	0.5423	0.3418	0.4754
PW+Stagenet	0.2145	0.2230	0.1451	0.2153	0.5830	0.5522	0.3499	0.4874
CHE+Staegnet	**0.2199***	**0.2310***	**0.1521***	**0.2264***	**0.6861***	**0.6567***	**0.4269***	**0.5879***
Improv %	2.327%	3.867%	4.464%	4.284%	19.91%	21.10%	24.90%	23.66%

NDCG@k relatively increases by 8.20% and ACC@k relatively increases by 9.20%.

Experimental Results on EHR

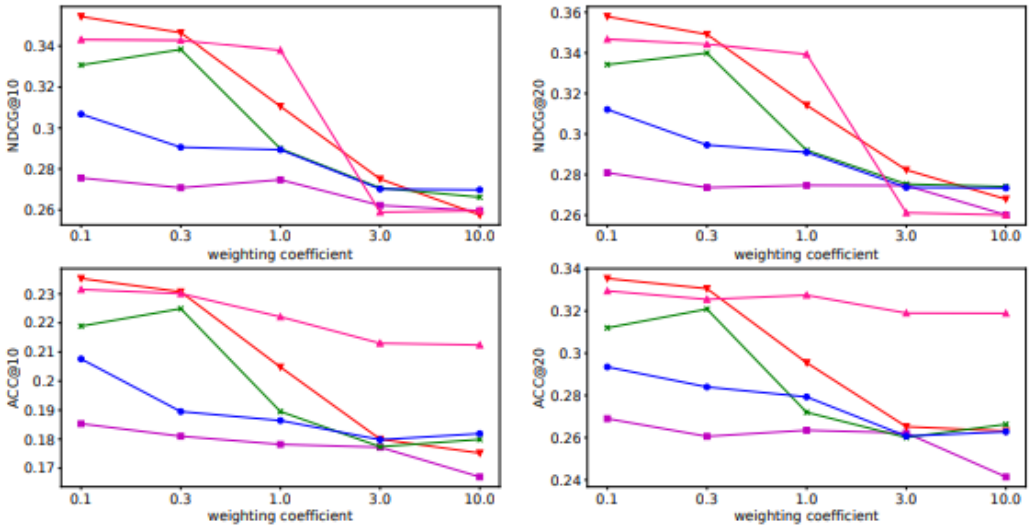


Figure 4: Performances of CHE+BaseModels with different weighting coefficients on the MIMIC-III dataset with random data division.

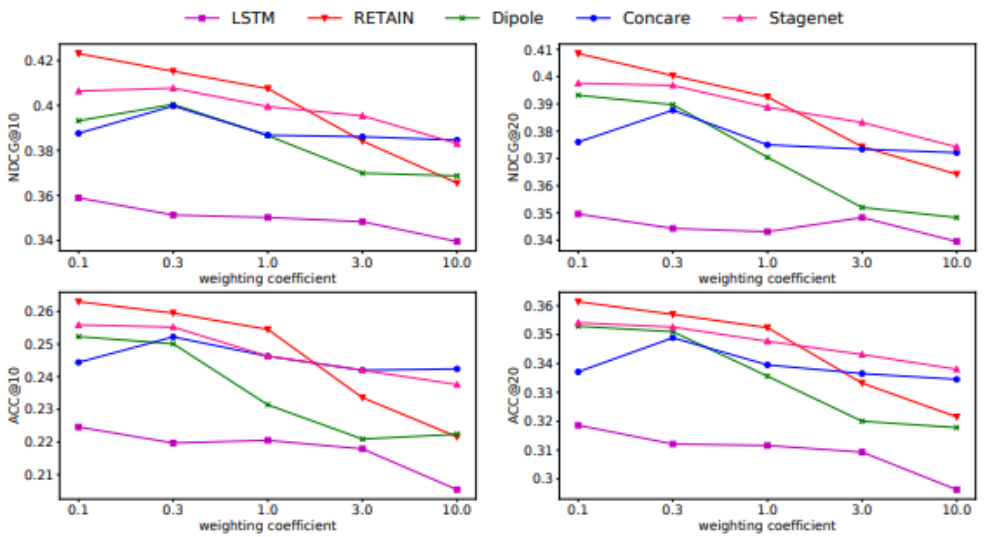


Figure 5: Performances of CHE+BaseModels with different weighting coefficients on the MIMIC-IV dataset with random data division.

Microsoft

UNIVERSITY of WASHINGTON

Experimental Results on EHR

Table 3: Feature interpretations of a patient from MIMIC-III.

Feature	CHE+Dipole			Dipole		
	1^{st}	2^{nd}	3^{rd}	1^{st}	2^{nd}	3^{rd}
Diagnosis	0.6841	1.710	1.201	0.8997	0.1366	1.714
Procedure	0.3413	0.1777	0.358	0.5469	0.1733	1.248

ture diagnosis. We know that the cause of diabetic retinopathy is diabetes. For patients with background diabetic retinopathy (ICD-9 code 36021), an ideal model should rely on related diseases such as diabetes to make prediction. In Table 3, we show the EHR of a patient in the MIMIC-III dataset and the feature interpretations, contribution of the features to the prediction, in CHE+Dipole and Dipole. Specifically, we apply gradient backpropagation for calculating feature interpretations [30, 32, 46, 49].

In this example, the diagnosis sequence is {4280, 5856}, {99592, 4280, 25060, 3572, V5861, V1251, 99662, 40391, 03811, 25050, 36201, 5856}, {03811, 5856, 99681, 42832}, {41401, 4280, 25050, 36201, 99591, 5856, 25060, 3572}. The future diagnosis to be predicted is background diabetic retinopathy (36021). The first visit contains two diseases that appear frequently among people, i.e. congestive heart failure (4280) and end stage renal disease (5856). The second visit contains some highly related features, such as diabetic retinopathy (25050), diabetes with neurological manifestations (25060), and polyneuropathy in diabetes (3572). In the third visit, the complications of transplanted kidney (99681) might be related. Compared with Dipole, CHE+Dipole pays more attention to causal features, i.e., the second visit with many highly related diagnoses. Moreover, the contributions of diagnosis and procedure are less correlated.

DESMIL - Stable Learning for RecSys

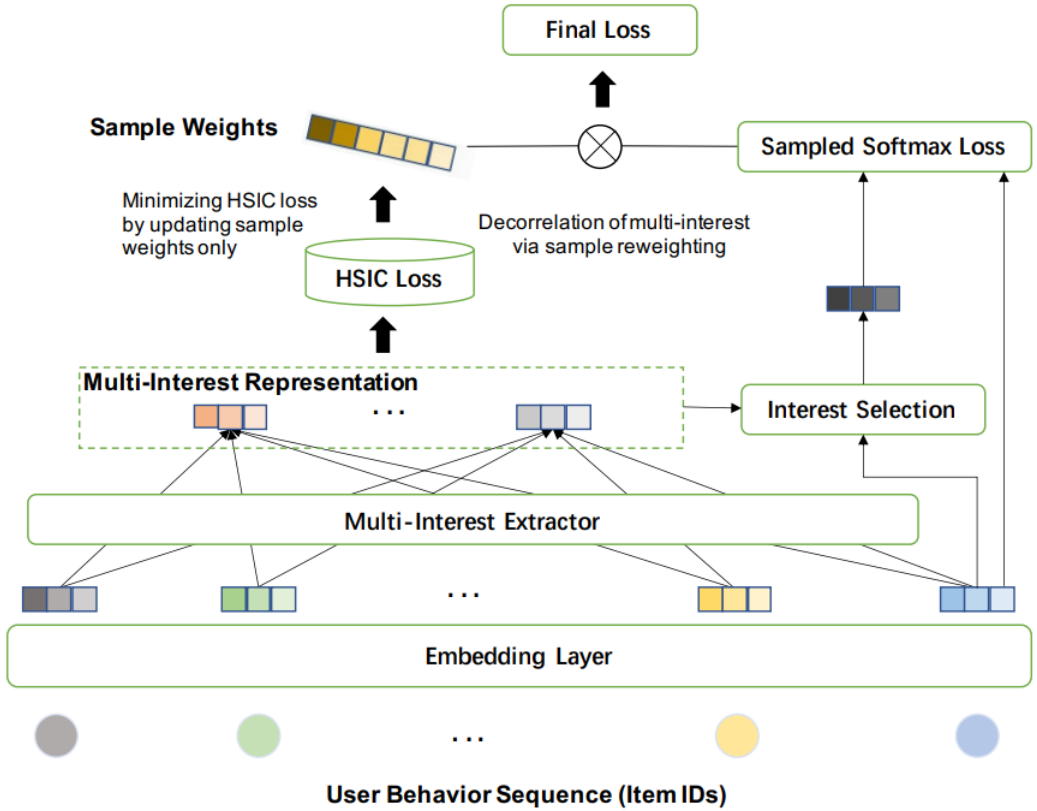


Figure 2: The overview of the proposed DESMIL. The input sequence is first embedded into dense representation to extract latent multi-interests. A HSIC loss is calculated based on multi-interest representations and optimized via sample weighting. The sample weights are then multiplied to the Softmax loss for final model optimization.

Microsoft

UNIVERSITY of WASHINGTON

DESMIL - Stable Learning for RecSys

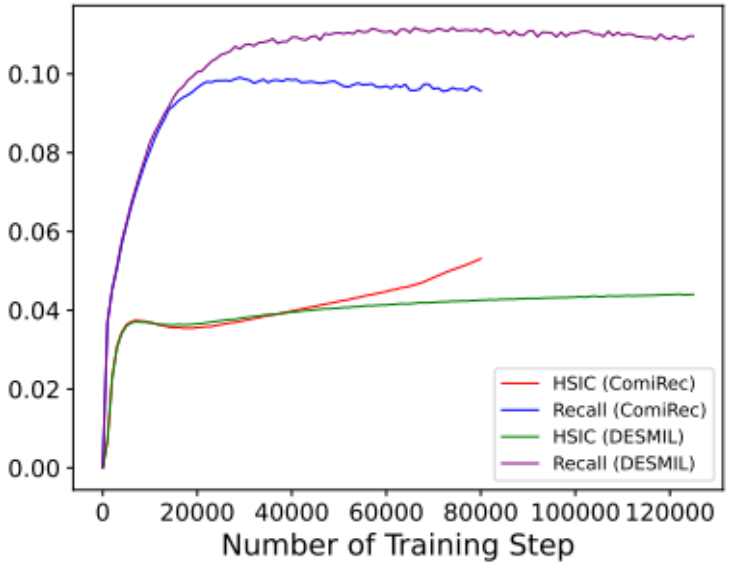


Figure 6: The curves of HSIC/Recall@50 on the validation set when training ComiRec and DESMIL on the Book dataset. With the use of early stopping, the training of them terminates at different step, which results in the different length of curves.

Microsoft

UNIVERSITY of WASHINGTON

Addendum - Stable Learning for RecSys

Table 1: Results on Public and industrial Datasets. Best performances are indicated by bold fonts and the strongest baselines are underlined. The improvement (Improv.) indicates the relative increase of our model over baselines on metrics.

Datasets	Metric	POP	GRU4Rec	Y-DNN	SASRec	MIND	ComiRec	CauseRec	DESMIL	Improv.
Book	Recall@20	1.37	3.47	4.40	4.76	5.10	5.92	5.75	**7.52**	27.03%
	Recall@50	2.40	6.50	7.31	7.78	7.64	9.35	9.36	**11.06**	18.16%
	NDCG@20	2.26	3.55	4.59	4.84	5.09	4.17	4.66	**5.46**	7.27%
	NDCG@50	3.94	4.42	5.54	5.74	5.97	5.47	6.28	**7.24**	15.28%
	HR@20	3.02	7.84	9.89	8.82	10.59	11.70	12.45	**14.86**	19.36%
	HR@50	5.23	12.38	14.94	13.79	15.56	18.04	20.23	**21.53**	6.43%
Movies and TV	Recall@20	3.59	13.20	12.38	14.43	14.87	15.46	15.30	**15.76**	1.94%
	Recall@50	6.62	17.66	17.31	18.27	19.55	18.87	19.24	**20.90**	6.91%
	NDCG@20	5.30	15.07	12.64	14.49	15.80	14.73	15.10	15.31	\
	NDCG@50	9.66	16.21	14.11	16.72	17.23	16.17	16.83	**17.36**	0.75%
	HR@20	6.51	22.67	21.32	23.25	25.34	25.87	25.94	**26.42**	1.85%
	HR@50	11.73	29.54	29.46	30.43	32.93	33.68	33.90	**34.80**	2.65%
CDs and Vinyl	Recall@20	0.993	4.39	5.24	6.92	7.55	7.96	7.77	**8.75**	9.92%
	Recall@50	1.89	6.07	7.72	8.52	10.32	11.23	11.12	**12.09**	7.66%
	NDCG@20	1.58	4.81	5.42	6.44	7.93	6.84	7.51	7.79	\
	NDCG@50	3.12	5.42	6.36	7.10	8.88	8.01	8.57	**8.86**	\
	HR@20	2.11	8.47	10.40	12.86	14.28	14.35	14.49	**15.73**	8.56%
	HR@50	4.08	11.79	15.46	16.29	19.38	20.26	20.66	**21.89**	5.95%
Industrial Dataset	Recall@20	1.01	6.31	6.28	6.81	6.96	7.23	7.02	**8.41**	16.32%
	Recall@50	1.75	9.88	10.76	11.02	11.29	11.51	11.76	**12.87**	9.44%
	NDCG@20	1.92	9.15	7.25	8.27	8.91	8.58	8.60	**9.19**	3.14%
	NDCG@50	3.28	10.60	9.20	9.78	10.75	10.35	10.54	**11.28**	4.93%
	HR@20	2.55	17.04	16.73	17.05	17.52	17.89	18.33	**20.50**	11.84%
	HR@50	4.38	24.97	23.94	25.11	26.21	26.56	26.77	**29.45**	10.01%

Addendum - Stable Learning for RecSys

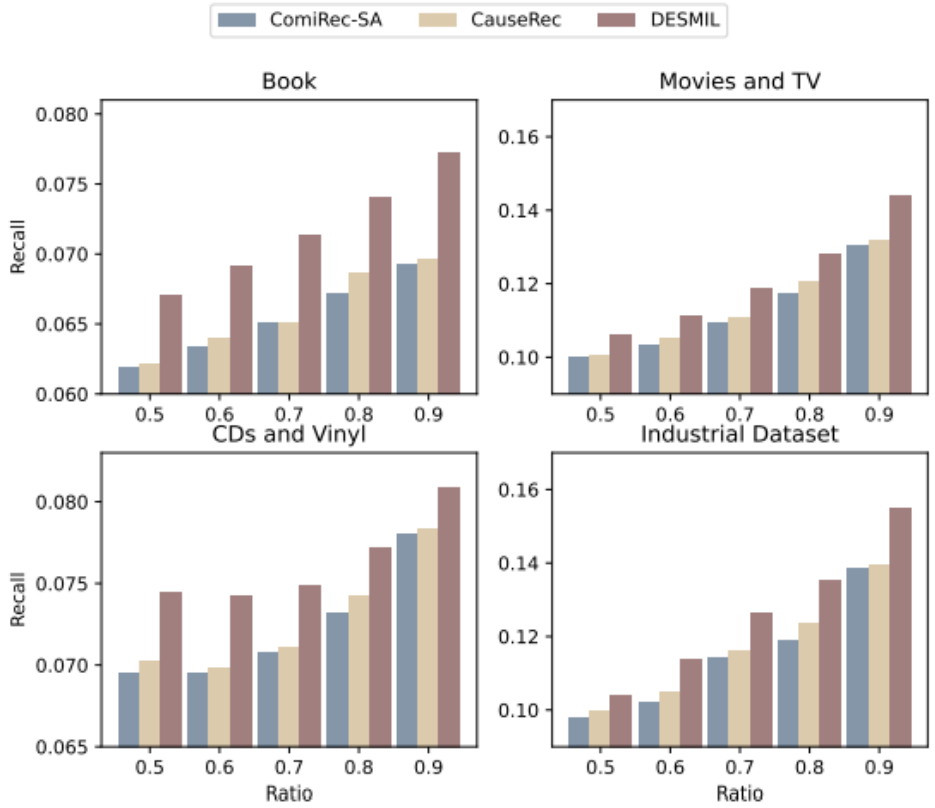


Figure 4: Comparison result of Recall@50 on four datasets with different ratio of simulated covariate shift.