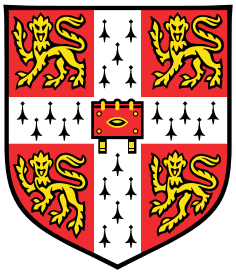
Matching Networks for Individual Organ Transplantation Allocation



Can Xu

Supervisor: Prof. Mihaela van der Schaar

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
MPhil in Machine Learning and Machine Intelligence

Wolfson College

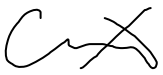
August 2020

Declaration

I, Can Xu of Wolfson College, being a candidate for the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

I also declare that this report contains less than 13000 words, excluding declarations, bibliography, photographs and diagrams, but including tables, footnotes, figure captions, appendices.

Can Xu
August 2020



Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Mihaela van der Schaar, for her continuous support of my MPhil research. Thanks for her patience, motivation, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis.

Besides my supervisor, I would like to thank the rest members who helped my MPhil project: Ioana Bica and Ahmed M. Alaa. Thanks for their comprehensive help and creative suggestions throughout all the time of the research.

My sincere thanks also go to Dr. Richard Turner, my course leader of MPhil in Machine Learning and Machine Intelligence, and all other instructors: Prof. Carl Rasmussen, Prof. Phil Woodland, Prof. Ioannis Kontoyiannis, Prof. Per Ola Kristensson, Dr. Ignas Budvytis, Dr. Miguel Hernandez Lobato, and Dr. Kate Knill, for their inspirational lectures and all other help to my academic life at Cambridge.

I thank all my classmates of MPhil MLMI class 2019-2020: Sasha Shisheya, Wenlong Chen, James Branigan, Tudor Paraschivescu, Wen Wu, Florian Langer, Woramanot Yomjinda, Conor Foy, Giannis Tsetis, Igor Adamski, Yuxin Chang, Andrew Lee, Andrius Ovsianas, Arduin Findeis, Rui Xia, Matthew Ashman, and Victor Bourgin, for their help in the entire academic year.

Last but not the least, I would like to thank my family: my parents Tingxiu Xiang and Yongzhu Xu, for giving birth to me in the first place and supporting me throughout my entire life, and my fiancee Yaqin Huang, for her accompaniment and supports.

Abstract

Organ allocation for organ transplantation is a vital task because good allocation policies promote patients' lives significantly. Existing allocation policies have limitations due to the ignorance of information about potential outcomes of allocating other donors to recipients. Methods for estimating individual treatment effects (ITE), which has been applied to various areas, are naturally applicable to overcome limitations that existing allocation policies have. By considering organ donors as treatments, methods used to estimate counterfactuals can be translated to estimate potential outcomes of organ transplantation. The advantage of counterfactual estimation is that it is possible to estimate all potential outcomes (counterfactuals) of transplantation using an organ from all different types of donors for each recipient. Hence, there is more information for each recipient, and a better allocation policy can be learned benefiting from additional information. However, estimating counterfactuals is a challenging task, especially in our case, since ground truth counterfactuals are never fully observed, and we have to deal with non-one-hot high-dimensional treatments as well as handling bias for more than two treatment types. In this paper, we propose a deep neural network to estimate potential outcomes of organ transplantation, called Self-Clustering Counterfactual Network (SCCN). Different from most of the existing works on ITE estimation, SCCN focuses on non-one-hot high-dimensional treatments, and the model can merge similar treatment types, which provides more interpretability of data. We show our proposed SCCN outperforms benchmark models in various aspects. Besides, we propose an allocation policy naturally driven from outputs of SCCN, called Matching First allocation policy. We show Matching First produces comparable results to state-of-art allocation policies. Compared to the real policy used by human experts, the policy extends the average survival time of recipients for 12.4% and reduces the death rate for 35.2%.

Table of contents

List of figures vii

List of tables ix

1 Introduction 1

2 Related Works 4

3 Problem Formulation 7

4 Self-Clustering Counterfactual Network (SCCN) 9

4.1 Clustering . . . 11

4.2 Optimization . . . 12

5 Experiments 14

5.1 Datasets . . . 14

5.2 Evaluation Metrics . . . 17

5.3 Baseline Models and Hyperparameter Setting . . . 19

5.4 Experiment Results . . . 21

5.4.1 Experiment 1: Performance on Data with Simple Structure . . . 21

5.4.2 Experiment 2: Ability of Merging Clusters . . . 22

5.4.3 Experiment 3: Representation Learning . . . 25

5.4.4 Experiment 4: Performance on Real-World Data . . . 26

5.4.5 Experiment 5: Alternate Training . . . 28

5.4.6 Experiment 6: Allocation Policy . . . 30

6 Conclusion 32

7 Discussion and Future Works 33

References 36

8 Appendix I: Pseudo-Code of Matching First Allocation Policy 39

9 Appendix II: Full Structure & Hyperparameter Setting of SCCN 40

10 Appendix III: Source Code 42

List of figures

4.1 The architecture of a multi-branched neural network for estimating counterfactuals. NN stands for neural network. **t** is a soft clustering of the type of donor **o**. Φ is representation of recipient **r**. **y** is a vector of all possible outcomes of **r** receiving organs from donors. y' is the observed factual. L_y is a factual loss. L_r is a loss regularizing representation of **r**. L_{cls} is the loss of cluster. 9

4.2 Causal graph of the problem setting. 10

5.1 Distributions of recipient populations for each type of donor in the GMixBiased dataset. For each type of donor, the corresponding distribution of the recipient population is different, which means the dataset is biased. 16

5.2 The network structure of a multi-branched neural network used to estimate counterfactuals. In this sample figure, the number of branches of the network is 3, which is consistent with the number of clusters. 20

5.3 Evolutions of validation loss of SCCN and baseline models as the training epoch is increasing. 21

5.4 Results of clustering of K-Means, EM, DEC, and jointly train DEC in SCCN. Since clustering algorithms do not have class labels, colors are randomly assigned to each cluster, and the same color in different plots does not represent the same class. Sample points are mapped to dimension 2 using PCA [23]. 23

5.5 Representations learned of SCCN trained on FPLTSD with various weights of representation loss. 26

5.6 Evolution of precision of estimated factual outcome of SCCN trained on FPLTSD with increasing weights of representation loss. Each point in the plots is an average precision of 20 trials. 26

5.7 Clusters of SCCN trained with and without alternate training. SCCN is trained on the GMixOverlap dataset. Colors are randomly assigned to each cluster, and the same color in different plots does not represent the same class. 29

7.1 GANITE based architecture. A GAN is built to generate a fully observed dataset, and another GAN is trained on the generated dataset to estimate counterfactuals. $\mathbf{z_G}$ and $\mathbf{z_I}$ are two random variables following a standard normal distribution. 34

9.1 The full structure of SCCN with the initial setting of the number of clusters being 3. On the right-hand side is the counterfactual estimation component, and on the left-hand side is the clustering component. 40

List of tables

2.1 Comparison of SCCN to existing methods on ITE estimation. 5

5.1 Ratios of pairings of recipients and donors with different blood types in the PLTSD dataset. Rows are for fixed donor blood types, and columns are for fixed recipient blood types. As shown, about 98% of donors are paired with recipients having the same blood type, which means the dataset is biased. . 17

5.2 Precisions and average variance of estimated counterfactual vectors of baseline models and SCCN evaluated on the GMixTiny dataset. The average precision of the top 50 trails out of 100 trails is taken. The variances of estimated counterfactual vectors are averaged over the top 10 trails out of 20 trails. 22

5.3 Precisions of baseline models and SCCN evaluated on the GMixOverlap dataset. The average precision of the top 50 trails out of 100 trails is taken. The number of clusters is set to be 5 and 8 to evaluate the performance of models under different settings. 24

5.4 Factual outcome precisions of SCCN and benchmarks trained on GMixBiased and FPLTSD datasets with and without representation loss. Since linear estimators do not have representation learning sub-component, inapplicable entries of the table are left blank. 25

5.5 Performance of SCCN and baseline models evaluated on PLTSD. Models with K-Means clusters have identical ϵ_{PEEF} and ϵ_{PEDF}, because K-Means produces hard cluster assignments. Representation loss is inapplicable for Models with linear estimators, and the corresponding entries are left blank. 27

5.6 Performance of SCCN with and without alternate training evaluated on the PLTSD dataset. 29

5.7 Performance of allocation policies evaluated using simulations run on the PLTSD dataset. Simulations of Matching First policy are run 30 times, and the top 10 and top 3 trails are reported. 31

9.1 Other hyperparameter settings of SCCN. 41

Introduction

Estimating counterfactuals is a primary task of estimating individual treatment effects (ITE), which proposes the question of what would be the outcomes if different actions have been applied. It is a fundamental problem that has been applied to various areas, such as estimating heterogeneous effects of drugs [3, 26], verifying causal factors of certain disease [11], and selection of treatments over time [5]. Organ allocation for organ transplantation is another problem that estimating ITE is naturally applicable. By considering organ donors as treatments, methods used to estimate counterfactuals can be translated to estimate potential outcomes of organ transplantation, and estimated outcomes can then be used to allocate organs.

The problem of our interests is that given a sequence of recipients waiting for organ donors, for a newly arrived organ donor, we want to allocate the organ to a recipient in the waiting sequence so that the overall benefit of the whole group of recipients is maximized. Recent research on organ allocation mainly focuses on improving the accuracy of predictions of transplantation outcomes [18, 21], or producing better matching of recipients and donors [34]. Most of the available works suggest allocation policy regarding scores/measurements evaluated based on predictions of the outcome of certain recipient-donor pairs. One of the main limitations of existing works is that allocations are only determined by transplantation outcomes of recipient-donor pairs but ignored lots of information about potential outcomes. Considering the case where there are recipients A and B, and two donors m and n arriving sequentially. Pair A-m, B-m, A-n, and B-n have predicted outcomes of surviving 100 days, 80 days, 150 days, and 10 days respectively. The optimal allocation policy is assigning donor m to recipient B and assigning donor n to recipient A. However, most of the existing allocation policies will allocate donor m to recipient A due to the arrival order of donors. When the allocator allocates donor m, it does not know about the potential outcomes of allocating other donors to these recipients.

To emphasis this kind of limitation, our work focuses on adapting counterfactual estimation methodology to solve the organ allocation problem. By considering donors as treatments, we are able to estimate the outcomes of assigning different types of donors to individual recipients. The advantage of counterfactual estimation methods is that, for each recipient, it is possible to estimate all potential outcomes (counterfactuals) of transplantation using organs from different types of donors. Hence, there would be more information for each recipient, and a better allocation policy can be learned benefiting from additional information.

However, estimating potential outcomes is a challenging task. The problem of ITE estimation differs from traditional supervised problems, as described by *Spirtes* [27]. A major challenge of estimating counterfactuals is that we only observe their reaction to one of the possible actions/treatments for each unit/patient. The entire vector of all possible outcomes is never observed for each unit/patient, since we only observe the factual outcome while all other counterfactual outcomes are not observed. The problem is typically solved using multi-task models [2, 26, 36]. Some recent works also manage the problem using generative adversarial networks (GAN) [33].

Another challenge is that data of ITE studies often suffer from selection bias. Estimating ITE is an important sub-field of causal inference. According to *Pearl* [22], Randomized Controlled Trial (RTC) is the only proper method for learning causal relationships, where treatment assignments are not depending on individual patients. However, because of financial and ethical issues, researches of ITE estimation usually use observational studies, in which treatment assignments depend on patient features and data contains selection biases. For instance, HER-2 targeted therapies are only applied to HER-2 positive breast cancer patients, so we never observe outcomes of applying HER-2 targeted therapies to other types of breast cancer patients. The biased data makes it difficult to estimate outcomes for certain subgroups of patients. To handle selection biases, previous studies [17] use the estimated probability of receiving treatment, called propensity scores, to reweight data. Some other recent works [9, 12, 32] use representation learning to eliminate selection biases.

Besides, the problem is even more challenging in our case. Different from most of existing ITE estimation studies [2, 4, 26, 32, 36], in which treatments are binary, we focus on non-one-hot high-dimensional treatments, as donor features are normally very high-dimensional and not necessarily one-hot. Therefore, in our case, we have to deal with

high-dimensional treatments as well as handle bias for multiple treatment types. In order to eliminate selection biases, many existing ITE studies on binary treatments use KL-divergence or integral probability metric (IPM) [19, 28] to balance distributions of patient groups. However, KL-divergence and IPM only take two distributions, so these studies cannot be naturally extended to high-dimensional treatment cases. Although some of the other studies [33] are able to be extended to one-hot high-dimensional treatment cases, there is rarely researches that actually focus on non-one-hot high-dimensional treatments.

In this paper, we propose a matching neural network, named Self-Clustering Counterfactual Network (SCCN), that attempts to estimate potential outcomes for each recipient if given different types of organ donors. We treat donors as high-dimensional treatments. Donors are clustered as different types, and counterfactuals are estimated regarding different types of donors. The clustering is a soft assignment so that we know the confidence level and can apply probability analysis in later steps. Both the counterfactual model and the clustering model are jointly trained so that the clustering is problem specific. Besides, we drive an allocation policy, called Matching First allocation policy. Allocations are based on potential outcomes estimated. We show that SCCN outperforms benchmark models in various aspects, and SCCN has the ability to merge similar donor clusters. We also show that the Matching First allocation policy produces comparable results to state-of-art allocation policies. Besides, compared to real policy determined by human experts, Matching First extends the average survival time of recipients for about 12.4% and improves the death rate for about 35.2%.

We summarize our contributions as follows:

1. We propose a novel allocation policy based on estimated potential outcomes, which overcomes some limitations of existing allocation policies.

2. We introduce a method of handling non-one-hot high-dimensional treatments in ITE estimation, as well as a method to learn balanced representation for more than two types of treatments, to reduce selection bias.

3. We propose a network with the ability to merge similar treatment types, which gives more freedom for hyperparameter tuning as well as more interpretability of data.

Related Works

Previous works in both ITE estimation and organ transplantation are closely related to our work. Because the problem of our consideration lays in between these two fields, we refer to works in both fields.

For traditional ITE estimation, a separate model is learned for each treatment. In this approach, selection bias is not taken into consideration. Learned models are biased towards distributions of corresponding treatments' populations. Another typical approach is considering treatment as a feature. In this approach, a single model is learned to estimate all counterfactuals or treatment effects, and distributions of different treatments' populations are adjusted to handle selection biases. For instance, *Wager & Athey* [30] use random forest, and *Johansson et al.* [12] and *Shalit et al.* [26] use deep neural nets to solve treatment effects estimation problem under this single model methodology. Besides, *Alaa & Schaar* [2] and *Alaa et al.* [3] use multi-task approaches, such as a multi-task Gaussian process, to estimate treatment effects. Our work is most similar to works of *Alaa et al.* [3] and *Shalit et al.* [26]. In all our work and these two works, deep neural networks with multiple branches are used to estimate potential outcomes. In our works, each branch of the deep neural network is used to perform one task: estimating the potential outcome of certain treatment. Shared layers are added before branching the network to learn representations [26] or capture "commonality" among learning tasks [3]. In *Shalit*'s work [26], selection bias is lessened by balancing distributions of learned representations of different treatments. IPM is minimized between representation distributions to force distributions to be similar. On the other hand, in *Alaa*'s work [3], selection bias is treated by a separate propensity network. Dropout layers are added in the network, with dropout probability calculated based on outputs of the propensity network. Under this setting, dropout probability is different for each training sample. Samples that lay in a region of poor treatment assignment overlap have higher dropout probability, so such samples have less impact on the model.

Methods	high-dimensional treatments	non-one-hot treatments	handling bias	non-fixed number of treatment types
[2]			√	
[3]			√	
[12]			√	
[26]			√	
[30]			√	
[33]	√			
SCCN	√	√	√	√

Table 2.1 Comparison of SCCN to existing methods on ITE estimation.

There are also other novel works on counterfactual estimation and selection bias elimination. In the work of *Yoon et al.* [33], counterfactual estimation and selection bias are resolved using GAN. In their proposed model GANITE, A GAN is built to generate unbiased datasets, while another GAN is trained using generated unbiased datasets to estimate potential outcomes. It is shown in the original paper that GANITE performs well, particularly when the dataset has high biases. Besides, it is proposed by *Alaa et al.* [2], regularizing posterior variance is a better way of ameliorating selection bias in some cases, instead of minimizing distributional distance. One main problem of minimizing the distributional distance to eliminate selection bias is that biased data are often highly informative and predictive. For instance, in the medical setting, the reason why data is biased is that doctors assign treatments based on predictive features.

Unlike many existing ITE estimation works, our work focuses on non-one-hot high-dimensional treatments, and we investigate balancing distributions of learned representations for more than two treatment types. Besides, in distinction from existing works, in which the number of treatment types is fixed, our proposed model has the ability to merge similar treatment types, which provides more interpretability of data.

On the other hand, our work is closely related to organ transplantation and organ allocation. Previous works on organ transplantation focus on developing a more accurate risk model for predicting survival after transplantation [18, 21]. In the work of *Nilsson et al.* [21], a deep neural net is proposed with classification and regression trees to predict transplantation outcomes and evaluate the impact of recipient-donor variables on survival. It is shown that the deep neural net model outperforms other existing scoring models [18, 21]. Instead of improving the accuracy of prediction of survival, other works focus on improving recipient-donor matching [34]. *Yoon et al.* [34] partition recipient-donor feature space into subspaces and use a separate prediction model for each subspace. In this architecture, each independent prediction model is trained to solve a more specific and

less general sub-problem. Therefore, models for subspaces of matched recipient-donor pairs are expected to be more robust than models trained to solve the general problem.

Problem Formulation

We aim to investigate methods designed for estimating counterfactuals on solving organ allocation problem. The principal goal is to estimate potential outcomes for each recipient, given different types of donors. Inherited from conventional setups of ITE estimation problem, in our case, we consider recipients as patients, donors as different types of treatments, and all potential outcomes of receiving organs from all types of donors as factual and counterfactuals.

Consider a population of recipients where each recipient is described by a feature vector $\mathbf{r} \in \mathscr{R}$, in which $\mathscr{R} \subseteq \mathbb{R}^{d_r}$ and d_r is the number of features for each recipient $\mathbf{r}$. Consider another population of donors where each donor is described by a feature vector $\mathbf{o} \in \mathscr{O}$, in which $\mathscr{O} \subseteq \mathbb{R}^{d_o}$ and d_o is the number of features for each donor. Let $\mathbf{y} \in \mathscr{Y} \subseteq \mathbb{R}^k$ denote a counterfactual vector of certain recipient. By considering $\mathbf{o}$ as treatment or intervention, the setting is naturally similar to the Neyman–Rubin causal model [24], described as follow. Assume for a patient $\mathbf{r}$, and a treatment $\mathbf{o}$, $y_1, y_2, ..., y_k$, being entries of $\mathbf{y}$, are all potential outcomes. For each patient, we only observe one of the potential outcomes, denoted as $y_f \in \{y_i\}_{i=1}^k$. In other words, for each patient, only certain entry of $\mathbf{y}$ is observed. Different from typical Neyman–Rubin causal model where treatments are binary, which means $\mathbf{o} \in \{0, 1\}$, we consider $\mathbf{o}$ as a high-dimensional and not necessarily one-hot vector (in other words, $d_o \geq 2$).

For dataset, we consider an observational dataset $\mathscr{D} = \{\{\mathbf{r}^{(n)}, \mathbf{o}^{(n)}, y_f^{(n)}\}\}_{n=1}^N$, consisting of N independent recipient-donor pairs with their corresponding observed transplantation outcomes $y_f^{(n)}$.

There are two important assumptions we made, which are essential assumptions that most of previous works [2, 26, 33] make.
Assumption 1: Overlap. For any recipients $\mathbf{r}$ and any donor $\mathbf{o}$, the probability of assigning

r to **o** is non-zero.

$$\forall \mathbf{r} \in \mathscr{R}, \forall \mathbf{o} \in \mathscr{O}, 0 < \mathbb{P}(\mathbf{o}|\mathbf{r}) < 1$$

With this assumption, it is reasonable to estimate all counterfactuals for any patients.
Assumption 2: Unconfoundedness. Given recipient **r**, potential outcomes **y** and donor assignments **o** are conditionally independent.

$$\mathbf{y} \perp \mathbf{o} \mid \mathbf{r}$$

With these assumptions, the goal is to find a mapping from the recipient space $\mathscr{R}$ to the counterfactual space $\mathscr{Y}$ using the observational dataset $\mathscr{D}$. In other words, we wish to have a estimation model learning a function $\mathbf{f} : \mathscr{R} \rightarrow \mathscr{Y}$. Under this setting, we desire to be able to estimate all outcomes $\mathbf{f}(\mathbf{r})$, given any recipient $\mathbf{r} \in \mathscr{R}$. The objective to be maximized is the likelihood of potential outcomes given recipients and donor features. Denoting vector of all other potential outcomes except y_f as $\mathbf{y_{cf}}$, the likelihood can be decomposed as:

$$\mathbb{P}(\mathbf{y}|\mathbf{r},\mathbf{o}) = \mathbb{P}(y_f|\mathbf{r},\mathbf{o})\mathbb{P}(\mathbf{y_{cf}}|\mathbf{r},\mathbf{o}) \tag{3.1}$$

We show how to optimize these two components in following sections.

Self-Clustering Counterfactual Network (SCCN)

We desire to train a model to predict all possible outcomes of a recipient receiving all types of organs. To achieve this, we propose Self-Clustering Counterfatual Network (SCCN). We cluster donors into different types, and counterfactuals are estimated regarding all types of donors. Both the clustering model and the counterfactual model are trained jointly so that the clustering is problem specific.

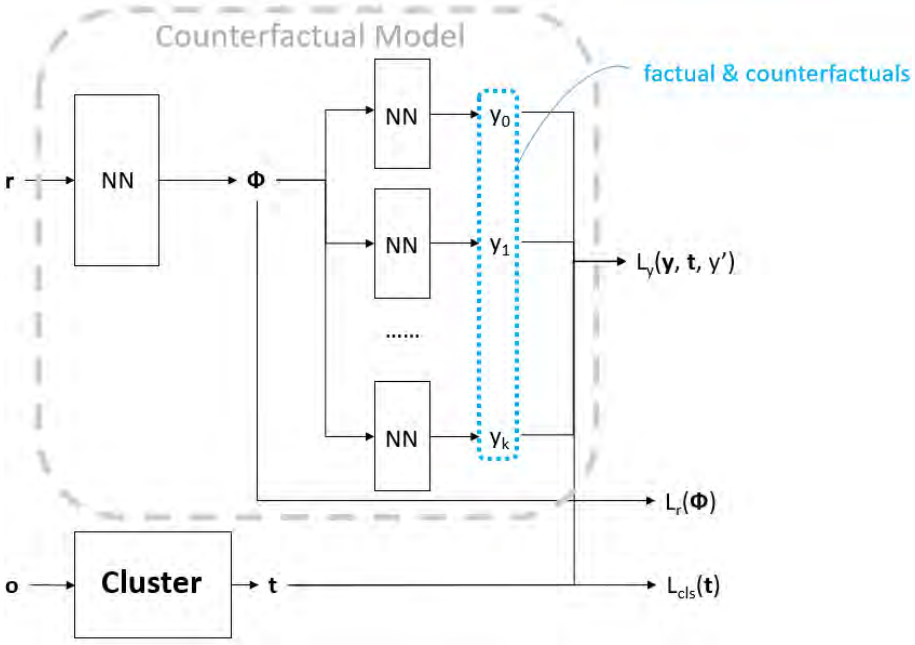
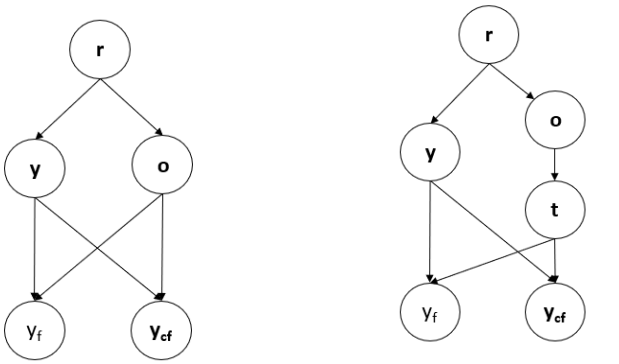


Fig. 4.1 The architecture of a multi-branched neural network for estimating counterfactuals. NN stands for neural network. **t** is a soft clustering of the type of donor **o**. Φ is representation of recipient **r**. **y** is a vector of all possible outcomes of **r** receiving organs from donors. y' is the observed factual. L_y is a factual loss. L_r is a loss regularizing representation of **r**. L_{cls} is the loss of cluster.



(a) original

(b) with latent variable **t**

Fig. 4.2 Causal graph of the problem setting.

Similar to typical counterfactual estimation methods [2, 26, 36], where counterfactuals are estimated using multi-task models, we use a deep multi-branched neural network. The network is estimating the function $\mathbf{f}(\Phi(\mathbf{r})) = \mathbf{y}$. Each branch of the network performs as solving one task: estimating one of the counterfactuals. Each branch aims to estimate the function $f_j(\Phi(\mathbf{r})) = y_j$. Formally,

$$\mathbf{f}(\Phi(\mathbf{r})) = \mathbf{y} = \{f_j(\Phi(\mathbf{r})) = y_j\}_{j=1}^{k}$$

The number of branches matches the number of clusters, which is a hyperparameter that can be set manually.

There are mainly two differences between our setting and typical counterfactual estimation setting: (a) our treatments are not binary; (b) our observed treatments are not hard assignments, but soft assignments produced by our clustering model. Because treatments in our setting are considered high-dimensional, many existing methods developed on binary treatment cases are not applicable. Other methods developed on high-dimensional one-hot treatment cases cannot be applied directly either, since treatments in our case are not necessarily one-hot. However, by clustering donors into different types, methods work for high-dimensional one-hot treatment cases could be applicable. Assume there is a latent variable **t**, representing the type of a donor. We can marginalize out **t** to compute

the likelihood of factual outcomes. Mathematically,

$$\mathbb{P}(y_f|\mathbf{r},\mathbf{o}) = \int \mathbb{P}(y_f, t|\mathbf{r},\mathbf{o})dt \tag{4.1}$$

$$= \int \mathbb{P}(y_f|\mathbf{r},t)\mathbb{P}(t|\mathbf{o})dt \tag{4.2}$$

in which $\mathbb{P}(t|\mathbf{o})$ is estimated by the clustering model using soft clustering $\mathbf{t}$, and parameters are eliminated because of conditional independence. The likelihood of observing factual data $\mathbb{P}(y_f|\mathbf{r},\mathbf{o})$ is one of the sub-components of the objective that we want to maximize, as shown in **Equation (3.1)**.

4.1 Clustering

The clustering model clusters donors into different types. Since there is no class label, only unsupervised clustering architectures are applicable. There are many unsupervised clustering architectures, such as K-means [15], Expectation Maximization (EM) [6], Deep Embedded Clustering (DEC) [31], etc. In this work, we adapt DEC architecture as the clustering method.

DEC involves training an Autoencoder at first to learn a representation of donors. According to *Xie et al.* [31], the encoder part of the Autoencoder is then used to learn a clustering using the following loss function:

$$t_{ij} = \frac{(1+||\mathbf{d}_i - \mu_j||^2)^{-\frac{1}{2}}}{\sum_j (1+||\mathbf{d}_i - \mu_j||^2)^{-\frac{1}{2}}}$$

$$p_{ij} = \frac{\frac{t_{ij}^2}{\sum_i t_{ij}}}{\sum_j \frac{t_{ij}^2}{\sum_i t_{ij}}}$$

$$L_{cls} = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{t_{ij}}$$

$\mathbf{d}_i$ is the learned representation of donor $\mathbf{o}^{(i)}$, which is produced by the encoder. μ_j is a clustering center, which is randomly initialized. t_{ij} represent the probability of donor $\mathbf{o}^{(i)}$ belongs to cluster j. μ_j-s and weights of the encoder are the parameters that need to be optimized during training. p_{ji}-s are the target distribution. According to the original paper, the choice of p_ij has the following properties: (1) strengthen predictions (i.e., improve cluster purity), (2) put more emphasis on data points assigned with high confidence, and

(3) normalize loss contribution of each centroid to prevent large clusters from distorting the hidden feature space.

The structure of the DEC model allows us to efficiently train the clusters and the counterfactual model jointly using error backpropagation. Clustering methods, such as K-means and EM, are hard to be trained jointly with our counterfactual neural network. Some other widely used unsupervised clustering methods using self-organizing map [13] involve competition learning, so that it is also challenging to apply those methods in this case.

4.2 Optimization

In this subsection, we describe the loss functions we use to optimize each part of SCCN. The empirical loss function of the counterfactual model consists of a factual loss, a representation loss, and the clustering loss described in the previous section.

The factual loss aims to maximize the likelihood of observing factual outcomes. In this work, the square error function is used as the factual loss function. This is, in fact, assuming Gaussian prior, since minimizing square is equivalent to maximizing likelihood with Gaussian prior [7]. As shown in **Equation (4.1) & (4.2)**, since we use soft assignment of clustering, the estimated factual outcome is an expectation of all estimated counterfactuals $\hat{\mathbf{y}}$ over treatment clustering distribution $\mathbf{t}$. Formally,

$$L_y(\hat{\mathbf{y}}, \mathbf{t}, y_f) = (\hat{\mathbf{y}} \cdot \mathbf{t} - y_f)^2$$

The factual loss only optimizes one sub-component in **Equation (3.1)**. The other component, representing the likelihood of counterfactual outcomes, also has to be optimized. Because counterfactual outcomes are never observed, it is impossible to compute counterfactual loss directly. Instead, distributional distance is minimized. As shown in works of *Johansson et al.* [12] and *Shalit et al.* [26], a linear combination of factual loss and distributional distance is an over bound of counterfactual loss. Therefore, a representation loss is used to regularize representations learned and also reduce selection bias. Similar to previous works [12, 26], we minimize distance among distributions of representations of recipients assigned different types of donors. However, since we have more than two types of donors, there are more than two distributions. Hence, traditional distance measurements (such as IPM, KL) are not applicable. To achieve optimizing representation network, we make two assumptions:

1. distributions of representations are Gaussian distributions;

2. distributions of representations have diagonal covariance matrices.

With these two assumptions, balancing distributions of representations is the same as regularizing means and variances of these distributions. Naively, this is minimizing the "variance" of all distributions, which means minimize the variance of means of distributions and variance of variances of representation distributions.

$$L_r(\Phi) = \sum_{c_j} (\mathbf{Mean}_{\mathbf{r}\in c_j}[\Phi(\mathbf{r})] - \mathbf{Mean}_{\mathbf{r}\in\mathscr{D}}[\Phi(\mathbf{r})])^2 + (\mathbf{Var}_{\mathbf{r}\in c_j}[\Phi(\mathbf{r})] - \mathbf{Var}_{\mathbf{r}\in\mathscr{D}}[\Phi(\mathbf{r})])^2$$

Besides, the distance between donors is often proportional to donors' similarity. Donors close to each other are more likely to be of the same type. Therefore, we also include the clustering loss described in the previous section in our objective function, to encourage donors that are close to each other to be clustered into the same type.

The empirical loss function of the counterfactual model $L = \frac{1}{N}\sum_i L_y + \alpha L_r + \beta L_{cls}$, in which L_y is a factual loss, L_r is a representation loss, L_{cls} is the clustering loss, and α and β are hyperparameters balancing losses of different parts of the model.

Experiments

In this part, we describe experiments conducted to evaluate our proposed models. Models are trained 100 times with different initializations, and datasets are randomly shuffled every time for cross-validation. The performance of each model is evaluated using the average performance of the top 50 trails out of 100 trails. For all experiments, datasets are divided 80%/10%/10% into training/validation/test sets.

5.1 Datasets

Due to the nature of the problem, it is difficult to comprehensively evaluate the performances of models using only real-world data. Since we never observe all counterfactuals in the real world, there is no ground truth target to evaluate estimated counterfactuals. Previous works [16, 26, 33] use both semi-synthetic datasets and randomized controlled trials (RCT) datasets. We use several synthetic and semi-synthetic datasets to evaluate our proposed counterfactual models from various aspects. We also use a real-world dataset to test the performance of the model under real circumstances.

GMixTiny: This dataset is a synthetic dataset generated using a Gaussian mixture model. The dataset contains 10000 pairs of recipients feature vectors and donor feature vectors. Recipient features are generated from a 128-dimensional Gaussian distribution with a randomly selected centre and random diagonal covariance matrix. Donor features are generated from a Gaussian mixture model consists of 3 64-dimensional Gaussian distributions. All Gaussian distributions have the same weights but different means and variances. All means and variances are randomly selected. As the number of Gaussian mixture components suggests, there are three types of donors, so a 3-dimensional target counterfactual vector is attached to each recipient-donor pair. All counterfactual vectors are generated from a Gaussian distribution with the fixed mean and fixed diagonal covariance matrix.

Formally,

$$\begin{aligned}
&\mathbf{r} \sim \mathcal{N}(\mathbf{m}_r, \mathbf{v}_r); \mathbf{m}_r, \mathbf{v}_r \in \mathbb{R}^{128} \\
&\mathbf{o} \sim \frac{1}{3}\mathcal{N}(\mathbf{m}_{o1}, \mathbf{v}_{o1}) + \frac{1}{3}\mathcal{N}(\mathbf{m}_{o2}, \mathbf{v}_{o2}) + \frac{1}{3}\mathcal{N}(\mathbf{m}_{o3}, \mathbf{v}_{o3}); \\
&\qquad\qquad \mathbf{m}_{o1}, \mathbf{m}_{o2}, \mathbf{m}_{o2}, \mathbf{m}_{v1}, \mathbf{m}_{v2}, \mathbf{m}_{v3} \in \mathbb{R}^{64} \\
&\mathbf{y} \sim \mathcal{N}(\begin{bmatrix} 100 \\ 500 \\ 2000 \end{bmatrix}, \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix})
\end{aligned}$$

The dataset is used to evaluate counterfactual models under an ideal situation. The choice of parameters of the counterfactual Gaussian distribution makes it easy to distinguish the outcomes of different donor classes. Recipients and donors are paired randomly, which means the dataset can be used as a dataset collected using RCT, and the dataset does not suffer from selection bias.

GMixOverlap: Similar to GMixTiny, this data set is a synthetic dataset generated using a Gaussian mixture model. The differences between this dataset and GMixTiny are: (i) donor feature vectors are generated from a Gaussian mixture model with five components with different weights; (ii) some entries of counterfactual vectors are sampled from identical Gaussian distributions.

$$\begin{aligned}
&\mathbf{r} \sim \mathcal{N}(\mathbf{m}_r, \mathbf{v}_r); \mathbf{m}_r, \mathbf{v}_r \in \mathbb{R}^{128} \\
&\mathbf{o} \sim \sum_{i=1}^{5} w_i \mathcal{N}(\mathbf{m}_{oi}, \mathbf{v}_{oi}); \mathbf{m}_{oi}, \mathbf{m}_{vi} \in \mathbb{R}^{64} \\
&\mathbf{y} \sim \mathcal{N}(\begin{bmatrix} 100 \\ 1000 \\ 100 \\ 1000 \\ 100 \end{bmatrix}, \begin{bmatrix} 10 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 10 \end{bmatrix})
\end{aligned}$$

The dataset is used to evaluate the performance of the clustering model trained jointly with the counterfactual model. As shown, although there are five classes of donors, some of them overlap and can be merged into the same class. There are only two distinct donor classes out of 5 classes. Weights of Gaussian mixture components are balanced regarding the two "hidden" classes. As the result described below, jointly trained clustering models have the ability to merge similar donor classes.

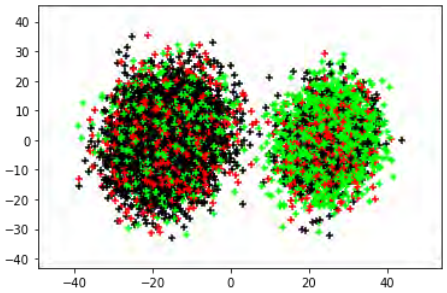


Fig. 5.1 Distributions of recipient populations for each type of donor in the GMixBiased dataset. For each type of donor, the corresponding distribution of the recipient population is different, which means the dataset is biased.

GMixBiased: Different from the above datasets, this dataset is a synthetic dataset with biases. This dataset is used to evaluate model performance when data is biased. The dataset is generated using a Gaussian mixture model. Donor features are generated from a Gaussian mixture model with three components of different means and covariance matrices. To incorporate biases, recipient features are generated from a Gaussian mixture model with two components. It means there are two types of recipients, and donors are not randomly paired to recipients. For different types of recipients, the probabilities of pairing each type of donor to the recipients are different. In this way, distributions of recipient populations for each type of donor are inequivalent, so the dataset is biased.

Paired Liver Transplant Standard Dataset (PLTSD): This is a real-world dataset, in which pairs of recipients and donors are extracted from the Liver Transplant Standard Dataset. The dataset contains 4460 recipient-donor pairs. Recipients are described by 55-dimensional feature vectors, and donor feature vectors have a dimensionality of 28. Because this is a real-world dataset, and full counterfactual vectors are never observed in the real world, the dataset contains only factual outcomes. Thus, the evaluation metrics are different from those of other fully observed datasets, described in later sections. Besides, the dataset is biased because of ethical and other issues. For instance, about 98% of donors are paired with recipients having the same blood type, which means transplantation outcomes of recipient-donor pairs with different blood types are rarely observed in the dataset.

Fully-Observed Paired Liver Transplant Standard Dataset (FPLTSD): This is a semi-synthetic dataset generated based on PLTSD. One limitation of PLTSD is that only factual outcome is observed, but entire vectors of counterfactuals are not observed in PLTSD. Because of the limitation, it is impossible to apply many evaluation metrics to evaluate models

Blood Type	O	A	B	AB
O	0.9863	0.0026	0.0106	0.0005
A	0.0010	0.9752	0	0.0238
B	0	0	0.9933	0.0067
AB	0	0.0138	0	0.9862

Table 5.1 Ratios of pairings of recipients and donors with different blood types in the PLTSD dataset. Rows are for fixed donor blood types, and columns are for fixed recipient blood types. As shown, about 98% of donors are paired with recipients having the same blood type, which means the dataset is biased.

trained on PLTSD. To overcome the disadvantage, we create this semi-synthetic dataset, in which recipients' and donors' features are identical to PLTSD, but potential outcomes are generated. In order to generate potential outcomes, the dimensionality of counterfactual vectors has to be set. We assume there are three types of donors, which means the dimensionality of counterfactual vectors is set to be 3. We also assume that there are two types of donors. Recipients and donors are then clustered using the Expectation-Maximization algorithm. For each combination of recipient type and donor type, the mean and variance of the potential outcome are estimated using observed data in PLTSD. For each recipient, the counterfactual vector is then generated from a Gaussian distribution with corresponding estimated means and variances. In this way, the expected potential outcome for each combination of recipient type and donor type is kept invariant to observed data in PLTSD.

5.2 Evaluation Metrics

To evaluate model performance, we use three precision metrics: precision in estimation of expected factual outcomes (PEEF), precision in estimation of deterministic factual outcomes (PEDF), and weighted precision in estimation of counterfactual outcomes (WPEC).

As mentioned, because of soft clustering, clustering output $\mathbf{t}$ is a probability distribution of donor $\mathbf{o}$ belonging to each donor type. The estimated factual outcome is the expectation of all counterfactual outcomes over donor type clustering distribution. Mathematically, $\hat{y_f} = \hat{\mathbf{y}} \cdot \mathbf{t}$. Therefore, PEEF is used to evaluate the precision of estimated factual outcomes.

$$\epsilon_{PEEF} = \frac{1}{N} \sum_{n=1}^{N} (\hat{\mathbf{y}}^{(n)} \cdot \mathbf{t}^{(n)} - y_f^{(n)})^2$$

Different from case to case, deterministic clustering may be preferred. In these cases, the estimated factual outcome is considered as the corresponding entry of the estimated counterfactual vector with the highest clustering probability, which means $\hat{y_f} = \hat{\mathbf{y}}[\arg\max_i \mathbf{t}[i]]$, where the operator [] is the indexing operator. Therefore, evaluating the precision of estimated deterministic factual outcome is necessary. We use PEDF as a precision measurement of estimated deterministic factual outcome.

$$\epsilon_{PEDF} = \frac{1}{N}\sum_{n=1}^{N}(\hat{\mathbf{y}}^{(n)}[\arg\max_i \mathbf{t}^{(n)}[i]] - y_f^{(n)})^2$$

Besides, we use weighted square Euclidean distance between estimated counterfactual vectors and ground truth counterfactual vectors to evaluate the precision of the entire estimated counterfactual outcome vectors. Since the jointly trained clustering model can merge similar donor types, the counterfactual model is expected to have reduced precision in estimating counterfactuals of eliminated donor types. It is unreasonable to include estimated outcomes of eliminated donor types into the evaluation. Therefore, distance measurements of estimation and ground truth are weighted by the proportion of donor types.

$$\begin{aligned}
\epsilon_{WPEC} &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{w}\cdot(\hat{\mathbf{y}}^{(n)} - \mathbf{y}^{(n)})^2 \\
\mathbf{w} &= \{w_m\}_{m=1}^{M} \\
w_m &= \frac{1}{N}\sum_{n=1}^{N}\mathbb{I}_m(\arg\max_i \mathbf{t}^{(n)}[i])
\end{aligned}$$

in which M is the number of clusters, which is a hyperparameter set manually, and $\mathbb{I}$ is the indicator function. One problem of ϵ_{WPEC} is that because of the unsupervised clusters, cluster labels of the learned clusters do not necessarily match donor type labels in datasets. In other words, donor type 1 in the dataset may be mapped to cluster 3 in learned clusters, which means columns of estimated potential outcome vectors may have to be shuffled to match ground truth vectors. Since cluster labels are randomly assigned, there is no fixed rule of how columns of counterfactual vectors should be shuffled. Therefore, we compute ϵ_{WPEC} of all possible shuffled counterfactual vectors and ground truth vectors, and the best ϵ_{WPEC} is taken.

In addition to the precision metrics described above, we also evaluate model performance from another aspect: accuracy of the type of donors with the highest potential outcome.

Since the type of donors with the highest potential outcome is an essential factor in determining allocation policy, the accuracy of the predicted highest potential outcome donor type is a vital measurement of model performance. The accuracy is evaluated as:

$$AoDT = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_{\arg\max_a \mathbf{y}^{(i)}[a]}(\arg\max_b \hat{\mathbf{y}}^{(i)}[b])$$

Same as ϵ_{WPEC}, since there is no class label when training clusters, cluster labels of the learned clusters do not necessarily match donor type labels in datasets. Therefore, $AoDT$ is computed for all possible shuffled $\hat{\mathbf{y}}$, and the best $AoDT$ is taken.

5.3 Baseline Models and Hyperparameter Setting

We use baseline models sharing the same fundamental structure of our proposed model, where there is a clustering component to classify donors as well as a counterfactual estimation component. To our knowledge, there is no existing architecture that has equivalent problem formulation as ours. Therefore, combinations of classic clustering and estimation architectures are used as baseline models. Typically, K-Means and EM clusters are used as the clustering component of the baseline models. Baseline clusters are pre-trained and fixed. The clusters are independent of the counterfactual estimation task, and they are not updated while training the counterfactual estimation component. As for the counterfactual estimation component, linear regression models and multi-branched neural nets (similar to our proposed counterfactual model) are used as baselines. We also use stand-alone DEC clusters plus branched neural nets as a benchmark, in comparison to SCCN with jointly trained DEC clusters.

The number of clusters (donor types) is one of the most critical hyperparameters. For synthetic datasets, the number of clusters is set to be the number of Gaussian mixture components used to generate donor features, since the number of Gaussian mixture components naturally represents the number of donor types under consideration. For dataset PLTSD, the number of clusters is manually set to be 5. A sufficiently large number of clusters does not have a significant impact on our proposed model because the model has the ability to merge clusters.

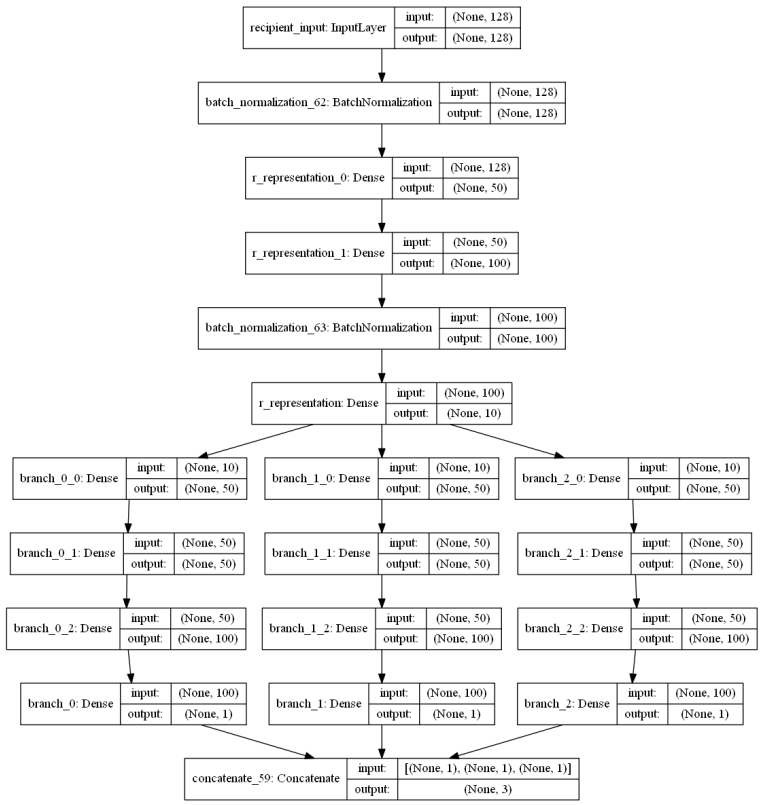
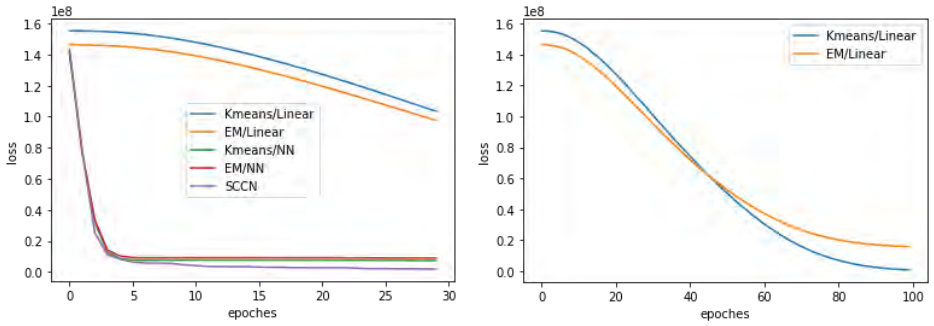


Fig. 5.2 The network structure of a multi-branched neural network used to estimate counterfactuals. In this sample figure, the number of branches of the network is 3, which is consistent with the number of clusters.



(a) (b)

Fig. 5.3 Evolutions of validation loss of SCCN and baseline models as the training epoch is increasing.

5.4 Experiment Results

5.4.1 Experiment 1: Performance on Data with Simple Structure

In the first experiment, we use the GMixTiny dataset to demonstrate how our proposed counterfactual model performs under the ideal situation. We use 8000 recipient-donor pairs for training, 1000 pairs for validation, and 1000 pairs for testing. The training-validation-testing process is repeated 100 times with different random initialization. Average precisions of the top 50 trials are taken to reduce the effects of local optimal.

In **Figure 5.3** we show evolution of model losses as training epoch is increasing. The figure demonstrates that SCCN and models with deep neural net counterfactual estimators require much less training iterations to converge. As shown in figure (a), after 30 epochs, models with linear counterfactual estimators are far from convergence, while other models approached convergence very quickly. After 100 epochs, baseline models with linear counterfactual estimators approaches convergence. Based on this observation, performances of linear counterfactual estimator models are evaluated after training 100 epochs, and other models are evaluated after training 30 epochs.

For each model, we use two precision metrics to evaluate their performance. Results are reported in **Table 5.2**. For baseline models with K-Means clusters, although K-Means assigns hard clustering to each input unit, the assignments are translated to one-hot vectors, so that the formula of computing ϵ_{PEEF} is consistent. As a result, SCCN outperforms all baseline models on the GMixTiny dataset, in terms of both factual outcome precision and counterfactual vector precision. The precision of factual outcomes improved about 32%

Model	ϵ_{PEEF}	ϵ_{WPEC}	Avg. Var. of counterfactuals
K-Means/Linear	8.9075	14.4785	414.53
EM/Linear	8.8190	13.5865	411.09
K-Means/Multi-Branched NN	5.7627	14.5612	347.22
EM/Multi-Branched NN	6.2182	14.4179	219.89
DEC/Multi-Branched NN	6.8472	14.6970	190.86
SCCN	**5.3838**	**12.5031**	**83.50**

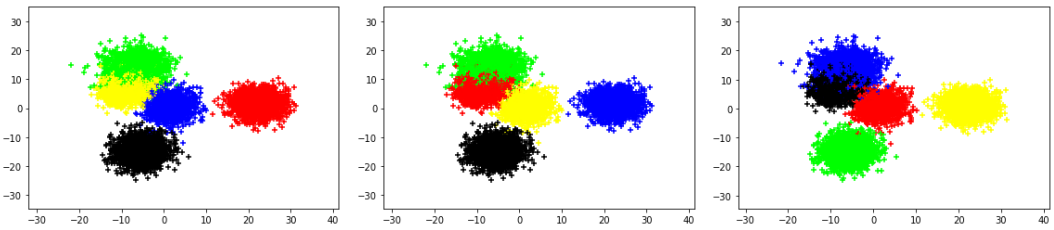
Table 5.2 Precisions and average variance of estimated counterfactual vectors of baseline models and SCCN evaluated on the GMixTiny dataset. The average precision of the top 50 trails out of 100 trails is taken. The variances of estimated counterfactual vectors are averaged over the top 10 trails out of 20 trails.

from the best benchmark model, and the precision of potential outcomes improved for about 66%.

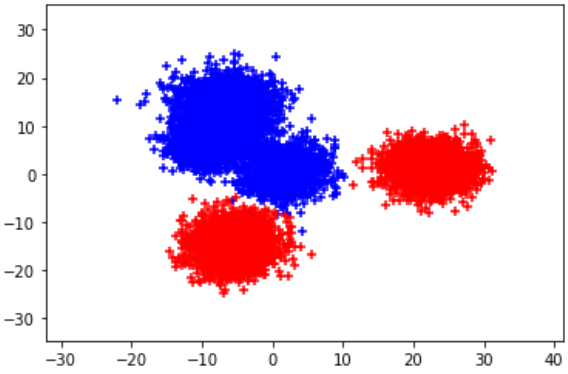
We also evaluated the average variance of estimated counterfactual vectors on the same dataset. In the GMixTiny dataset, there is only one type of recipients. Ground truth counterfactual vectors are generated from a Gaussian distribution with a fixed centre and variances of 100 for each dimension. A good estimation model should have estimated counterfactual vectors with variances matched the variances of the Gaussian distribution from where the data is generated. Variance is computed for each column of estimated counterfactual vectors (each type of donor), and the average is taken for variances of all columns. Variances reported in **Table 5.3** are averages of top 10 trails out of 20 trails. As a result, SCCN achieves the lowest average variance of 83.50, which significantly outperforms other benchmarks. The results suggest that SCCN actually understands and learns the structure of the dataset.

5.4.2 Experiment 2: Ability of Merging Clusters

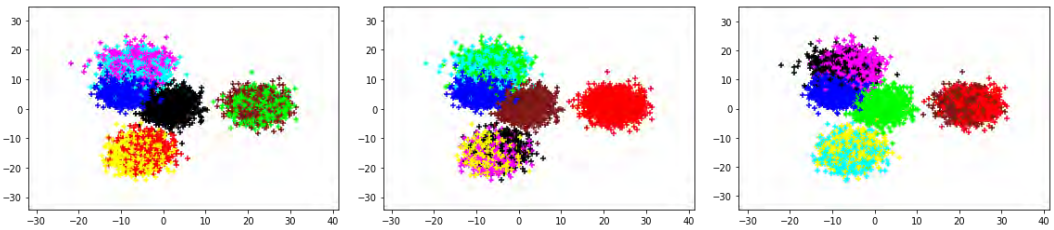
In the second experiment, we demonstrate that SCCN has the ability of merging similar clusters. Models are evaluated on the GMixOverlap dataset. As described in the previous section, donor features in the dataset are generated from 5 different Gaussian distributions, which means there are five generative classes for donors. However, donors from 3 out of 5 classes lead to similar outcomes, and donors from the other two classes lead to another type of outcome (details described in previous **section 6.1.1**). Therefore, there are, in fact, two significant donor classes. In this experiment, we show that SCCN can merge similar donor types, independent from the hyperparameter setting of the number



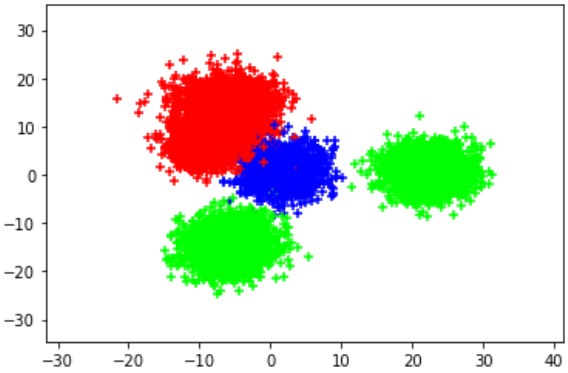
(a) K-Means (b) EM (c) DEC



(d) jointly trained DEC initialized with 5 clusters



(e) K-Means (f) EM (g) DEC



(h) jointly trained DEC initialized with 8 clusters

Fig. 5.4 Results of clustering of K-Means, EM, DEC, and jointly train DEC in SCCN. Since clustering algorithms do not have class labels, colors are randomly assigned to each cluster, and the same color in different plots does not represent the same class. Sample points are mapped to dimension 2 using PCA [23].

Model	5 clusters ϵ_{PEEF}	8 clusters ϵ_{PEEF}
K-Means/Linear	9.6768	9.7185
EM/Linear	11.0664	9.7039
K-Means/Multi-Branched NN	8.7396	8.7061
EM/Multi-Branched NN	8.7003	8.7165
DEC/Multi-Branched NN	8.7078	8.6890
SCCN	**8.6157**	**8.6489**

Table 5.3 Precisions of baseline models and SCCN evaluated on the GMixOverlap dataset. The average precision of the top 50 trails out of 100 trails is taken. The number of clusters is set to be 5 and 8 to evaluate the performance of models under different settings.

of donor types.

Models are trained with the number of clusters set to be 5 and 8. The dimensionality of estimated counterfactual vectors matches the number of clusters, which is varying in this case. It does not necessarily match the dimensionality of observed ground truth counterfactual vectors. Hence, it is inappropriate to evaluate the model by ϵ_{WPEC}. Models are evaluated by the precision of estimated factual outcome ϵ_{PEEF}.

As shown in **Table 5.3**, SCCN outperforms all other models, especially on the precision of estimated factual outcomes. One observation is that EM/Linear has significantly worse performance when there are 5 clusters. It is believed that the model is often trapped to local optimal. Besides, as shown in the table, having more clusters does not have a significant impact on model performance. Different clusters can have similar estimation functions so that having extra clusters does not significantly affect model performance.

As shown in **Figure 5.4**, jointly trained DEC clusters in SCCN merges similar clusters. Extra clusters are often eliminated during training so that the setting of the number of clusters does not have significant effects on SCCN. With different settings of the number of initial clusters, the jointly trained clusters tend to approach the same optimal clustering. Although some times extra clusters are not eliminated completely (in **Figure 5.4**, when initialed with 8 clusters, there are 3 clusters left, while the optimal number of clusters is 2), we can also see the intention of approaching the optimal merging. The ability to merge similar clusters does not only provide more robustness but also shows more interpretability of the data. In the real world, the number of types of donors is latent and unknown. The ability to merge similar types provides us insight into how many significant donor types there are, which is beneficial for further studies.

GMixBiased		
Model	ϵ_{PEEF}	ϵ_{PEEF} with rep loss
K-Means/Linear	9.6376	
EM/Linear	9.4781	
K-Means/Multi-Branched NN	8.8587	8.8561
DEC/Multi-Branched NN	8.7489	8.1911
SCCN	**8.1349**	**8.1024**
FPLTSD		
Model	ϵ_{PEEF}	ϵ_{PEEF} with rep loss
K-Means/Linear	14.2852	
EM/Linear	14.2537	
EM/Multi-Branched NN	13.1531	13.1380
SCCN	**13.0868**	**13.0776**

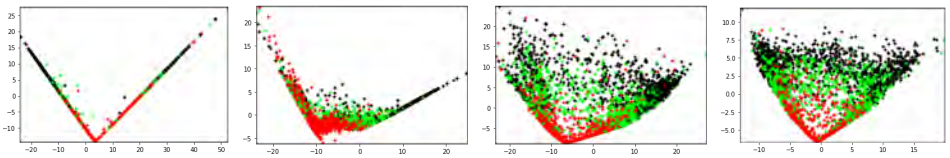
Table 5.4 Factual outcome precisions of SCCN and benchmarks trained on GMixBiased and FPLTSD datasets with and without representation loss. Since linear estimators do not have representation learning sub-component, inapplicable entries of the table are left blank.

5.4.3 Experiment 3: Representation Learning

In this experiment, we demonstrate how representation learning influences our proposed model. In previous experiments, selection bias is ignored, since datasets used in previous experiments contains randomly paired recipient-donor pairs, which does not suffer from selection bias. In this experiment, we evaluate model performance on GMixBiased and FPLTSD datasets. As described above, GMixBiased is a synthetic dataset incorporated selection bias, and FPLTSD is a semi-synthetic dataset based on real-world data, which naturally comes with biases. We first measure the performance of SCCN and benchmark models based on basic precision metrics ϵ_{PEEF}. Results are reported in **Table 5.4**.

As reported in **Table 5.4**, SCCN shows better precisions on GMixBiased datasets with and without representation loss. For our proposed SCCN, the precision of estimated factual outcome improved a little for about 3% with representation loss. As for the FPLTSD dataset, since the dataset is based on real-world data, the dataset has a more complex data structure, so models generally have worse performance than models trained on GMixBiased. However, it is still observable that SCCN outperforms other benchmark models.

We also test how representation loss affects representations learned as well as the per-



(a) $\alpha = 0.001$ (b) $\alpha = 0.01$ (c) $\alpha = 1$ (d) $\alpha = 10$

Fig. 5.5 Representations learned of SCCN trained on FPLTSD with various weights of representation loss.

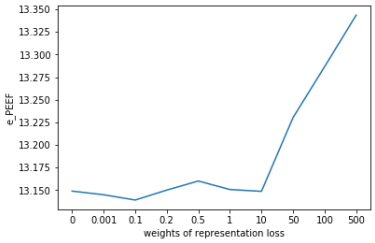


Fig. 5.6 Evolution of precision of estimated factual outcome of SCCN trained on FPLTSD with increasing weights of representation loss. Each point in the plots is an average precision of 20 trials.

formance of SCCN. As mentioned in **section 4.2**, the empirical objective function of our consideration is a linear combination of factual loss, clustering loss, and representation loss. The weight of representation loss α controls whether the model would put more efforts on minimizing representation loss or other losses. In other words, with a large α, the model would focus more on learning balanced representations rather than minimizing the error of estimated factual outcomes. As shown in **Figure 5.5**, as the weight of representation loss increases, SCCN learns more balanced representations.

Results in **Table 5.4** suggests that with a well-tuned weight of representation loss, representation learning promotes model performance. However, extremely balanced representations often lead to worse model performance. Biased observational data is usually highly informative and predictive since it often incorporates knowledge of human experts. Minimizing distributional distance results in losing that information. As shown in **Figure 5.6**, large α results in performance decline.

5.4.4 Experiment 4: Performance on Real-World Data

We evaluate SCCN and baseline models on the real-world dataset PLTSD. The dataset has a much complex data structure than synthetic datasets used in previous experiments, and the true number of donor types is latent. In this experiment, we set the number of

PLTSD		
Model	ϵ_{PEEF}	ϵ_{PEDF}
K-Means/Linear	14.3448	14.3448
EM/Linear	14.2813	14.2813
K-Means/Multi-Branched NN	13.1038	13.1038
EM/Multi-Branched NN	13.1453	13.1452
SCCN	13.0821	13.1192
Model with rep loss	ϵ_{PEEF}	ϵ_{PEDF}
K-Means/Multi-Branched NN	13.0877	13.0877
EM/Multi-Branched NN	13.1407	13.1407
SCCN	**12.9938**	**13.0612**
FPLTSD		
Model	AoDT	
K-Means/Linear	.6342	
EM/Linear	.6256	
K-Means/Multi-Branched NN	.6410	
EM/Multi-Branched NN	.6562	
DEC/Multi-Branched NN	.6430	
SCCN	**.6737**	

Table 5.5 Performance of SCCN and baseline models evaluated on PLTSD. Models with K-Means clusters have identical ϵ_{PEEF} and ϵ_{PEDF}, because K-Means produces hard cluster assignments. Representation loss is inapplicable for Models with linear estimators, and the corresponding entries are left blank.

clusters to be 5 for all models. Since PLTSD is a real-world dataset, entire ground truth counterfactual vectors are never observed. It is impossible to evaluate the precision of estimated counterfactual vectors. Instead, we use ϵ_{PEEF} and ϵ_{PEDF} to evaluate models' performance. Besides, unlike our generated synthetic datasets, where recipients and donors are paired simulating RCT, PLTSD contains biases. Therefore, we train models with and without representation loss to evaluate the effects of representation learning on model performance.

As shown in **Table 5.5**, SCCN outperforms baseline models on the precision of estimation of expected factual outcomes, especially with representation loss. For SCCN with representation loss, ϵ_{PEEF} improved about 9% from the best benchmark. SCCN also shows comparable results on the precision of estimation of deterministic factual outcomes. It is also observable that training with representation loss enhances model performance when data suffers from biases. Representation learning improved the performance of SCCN for

about 8.5%.

Additional to the precision of estimated outcomes, models' performances are also evaluated by the accuracy of predicting the best donor type (AoDT). Since the PLTSD dataset does not contain ground truth potential outcome vectors, it is impossible to calculate AoDT on PLTSD. Instead, we evaluate models on the FPLTSD dataset, in which recipient and donor features are identical to data in PLTSD, and counterfactual vectors are generated as described in **section 5.1**. Models are trained 50 times, and the average AoDT of the top 20 trails are reported in **Table 5.5**. As a result, SCCN achieves the highest AoDT of 67.37%, which improved for 2.6% from the best benchmark and improved for 51% from random guessing.

5.4.5 Experiment 5: Alternate Training

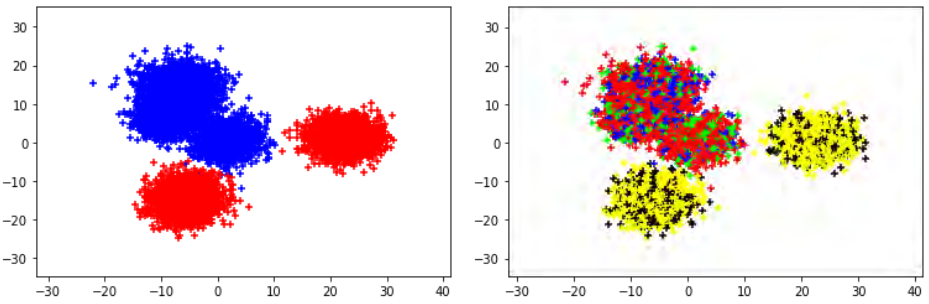
One limitation of SCCN is that SCCN tends to sacrifice clustering certainty to maximize the precision of estimation of expected factual outcomes. The limitation results in SCCN having worse performance on the estimation of deterministic factual outcomes (shown in **Table 5.5**). Although, in this case, expected factual outcomes are more considered, in other cases, deterministic factual outcomes could be crucial. To emphasize the precision of estimation of deterministic factual outcomes, we introduce alternate training, with which SCCN shows better ϵ_{PEDF}.

In the alternate training approach, we train one of the clustering component and counterfactual estimation component of SCCN, with the other component frozen. Besides, when training the counterfactual estimator, an entry of the estimated counterfactual vector corresponding to the assigned donor cluster is taken to compute factual loss, instead of using the expectation of a counterfactual vector as estimated factual to compute factual loss. With the edited factual loss, the estimator is encouraged to maximize the precision of estimation of deterministic factual outcomes, rather than expected factual outcomes. Formally, the alternate training approach involves the following steps:

1. Use K-Means to fit donor feature vectors as the initialization of clustering targets of the clustering component.
2. Train a clustering model.
3. Train the counterfactual estimator using the edited factual loss function, while the clusters are fixed.

Model	ϵ_{PEEF}	ϵ_{PEDF}
SCCN	**13.0821**	13.1192
SCCN alternate trained	13.1382	**13.0708**

Table 5.6 Performance of SCCN with and without alternate training evaluated on the PLTSD dataset.



(a) without alternate training (b) with alternate training

Fig. 5.7 Clusters of SCCN trained with and without alternate training. SCCN is trained on the GMixOverlap dataset. Colors are randomly assigned to each cluster, and the same color in different plots does not represent the same class.

4. Take the index of the entry of each estimated counterfactual vector that is the closest to the ground truth factual outcome as the new clustering target of the clustering component.

5. Go back to step 2.

As a result, alternate training improves SCCN's precision of estimation of deterministic factual outcomes, but ϵ_{PEEF} decreases as the cost. Besides, we also test how alternate training effects SCCN's ability to merge clusters. SCCN is trained on the GMixOverlap dataset with and without alternate training. Plots of clustering results is shown in **Figure 5.7**.

With alternate training, extra clusters are not eliminated but expanded to cover the whole subspace of the group of similar clusters. This is because, in step 3, clusters are fixed, so some branches of the branched neural network are forced to estimate the ground truth factual outcome. While without alternate training, these branches may not be updating significantly because their corresponding clusters may have been merged. Therefore, in step 4, some clustering targets may be set to be clusters that should have been merged without alternate training. Thus, clusters are not merged with alternate training.

5.4.6 Experiment 6: Allocation Policy

In this experiment, we evaluate an allocation policy naturally driven from our proposed SCCN model. One of the main disadvantages of existing allocation policies is that allocations usually only depend on predicted transplantation outcomes (factual outcomes) of certain recipient-donor pairs, ignoring potential outcomes (counterfactual outcomes) of other possible pairings. Conventional allocation policies, such as First Come First Serve (FCFS), Utility First, and Benefit First, are all based on estimated factual outcomes only. As mentioned, in many cases, these allocation policies result in sub-optimal results due to arriving orders of donors and ignorance of potential outcomes.

We propose an allocation policy naturally driven from outputs of SCCN, called **Matching First** allocation policy. In this policy, a newly arrived donor is allocated to the first recipient in the waiting sequence, for which the donor type with the best counterfactual matches the type of the arrived donor. If there is no such recipient in the sequence, the arrived donor is allocated by Benefit First policy. We compare Matching First policy to the three widely used allocation policies mentioned above as well as the real-world policy, where human experts determine allocations.

1. **Real Policy**: a new donor is allocated based on human expert knowledge. This is the policy by which the real-world dataset PLTSD is collected.
2. **FCFS**: a new donor is always allocated to the first recipient in the waiting sequence.
3. **Utility First**: a new donor is allocated to the recipient with the best predicted transplantation outcome (survival time after transplantation).
4. **Benefit First**: a new donor is allocated to the recipient with the highest benefit, where the benefit is defined as a positive difference between predicted survival time after transplantation and survival time if not undergoing transplantation.
5. **Matching First**: a new donor is allocated to the first recipient in the sequence for which the corresponding donor type of the best estimated potential outcomes matches the type of the new donor.

We use real-world dataset PLTSD to evaluate allocation policies. Different from previous experiments, in this experiment, recipients and donors are treated as sequential data. All recipients in PLTSD are treated as a sequence of waiting recipients, and donors are treated as a series of new donors. Recipients and donors are randomly shuffled so that recipients and donors are not paired as in original PLTSD. A regression neural network is

Allocation Policy	Death Rate	Avg. Survival	Avg. Benefit
Real Policy	0.2417	2457.78	500.37
FCFS	0.2417	2431.49	460.40
Utility First	0.3045	2255.11	388.71
Benefit First	0.1558	2770.57	633.68
Matching First top 10	0.1582	2755.69	632.28
Matching First top 3	0.1566	2762.69	654.34

Table 5.7 Performance of allocation policies evaluated using simulations run on the PLTSD dataset. Simulations of Matching First policy are run 30 times, and the top 10 and top 3 trails are reported.

trained to predict the transplantation outcome given a recipient-donor pair. The outputs of the regression neural net are used as ground truth outcomes during simulations. The performance of an allocation policy is evaluated by the ratio of death due to condition deteriorated on the waiting list (before transplantation), the average survival time of all recipients, and the average benefits of all recipients undergoing transplantation. Good allocation policies are expected to have a low ratio of death, but high average survival time as well as high average benefits. Simulation results are shown in **Table 5.7**.

It is observable that among previously existing policies, Benefit First significantly outperforms other policies. Among existing policies, Benefit First policy produced the lowest ratio of death, the highest average survival time, and the highest average benefits as well. Our proposed Matching First policy also achieves comparable results. In terms of death rate and average survival time, the differences between Matching First and Benefit First are less than 2%. Besides, Matching First improves the average benefits of recipients for 3.3%, compared to Benefit First policy. Compared with the real policy determined by human experts, Matching First reduced the death rate for 35.2%, and, on average, Matching First extends the survival time of recipients for about 12.4%.

Conclusion

In this paper, we proposed a novel method for solving the organ allocation problem for organ transplantation. Different from many existing works on organ transplantation, we adapted architectures for ITE estimation problems to investigate the limitations of existing works due to the ignorance of potential outcomes. Besides, we introduced a novel counterfactual estimation method to handle non-one-hot high-dimensional treatments. The proposed SCCN does not require binary or one-hot treatments in distinction from many previous works on ITE estimation. The model also has the ability to merge similar treatments, so that the number of treatments does not have to be fixed.

We have shown the proposed SCCN outperforms all benchmark models in various aspects, in terms of factual estimation, counterfactual estimation, the accuracy of predicting the best type of treatments, etc. The ability to merge clusters provides more freedom of hyperparameter tuning and gives more interpretation of the data. We also introduced a method of balancing representation learning for multiple (more than two) treatments and discussed how representation learning influences model performance. As for allocation policy, the proposed Matching First allocation policy achieved comparable and even slightly improved results compared to state-of-art policies. Matching First policy also achieved significant improvement over the real allocation policy used by human experts.

Discussion and Future Works

Our work introduces a novel solution to the problem of organ allocation that adapts ideas for solving ITE estimation problems. The problem of our consideration is vital in both machine learning and medical fields. The problem is related to human lives, and we are investigating a much more challenging problem that differs from traditional supervised or unsupervised learning problems. In this problem, ground truth outcomes are never fully observed, and it is challenging to handle high-dimensional treatments.

It also goes beyond the paper. For problems related to human lives, ethics always has to be taken into consideration. This work is for research purposes, and it can not be directly put into practice. One reason is that there is still a long way to go until introduced architectures can reach a practical level. Another more important reason is that there are many other factors that have to be taken into accounts, such as ethics and fairness, in the real world. However, this work can be investigated to produce helpful tools for doctors, as providing helps to human lives is always one of the essential purposes of researches.

Our work provides various inspirations about how to allocate resources, handle non-one-hot high-dimensional treatments, etc. There are many potentials of our proposed architecture. Further studies could focus on developing better counterfactual models, developing better allocation algorithms, or extending model abilities.

For counterfactual estimation:

1. Better representation loss could be investigated for the replacement of the current loss function. If data is sampled from a Gaussian mixture model, balancing distributions of samples that belong to different classes is contradicted to the data structure. Instead of balancing distributions, we can minimize posterior variance [3], and other architectures [32] are applicable.

2. The Jointly trained clustering model could be improved. The work of *Guo et al.* [8] is applicable. Besides, the clustering model could be improved to eliminate limitations that the current clustering model has (clusters tend to assign uncertain classes to minimize factual loss).

3. The architecture of GANITE [33] could also be adapted, rather than using multi-branched neural networks. Potentially, the counterfactual model could have structure as shown in **Figure 7.1**.

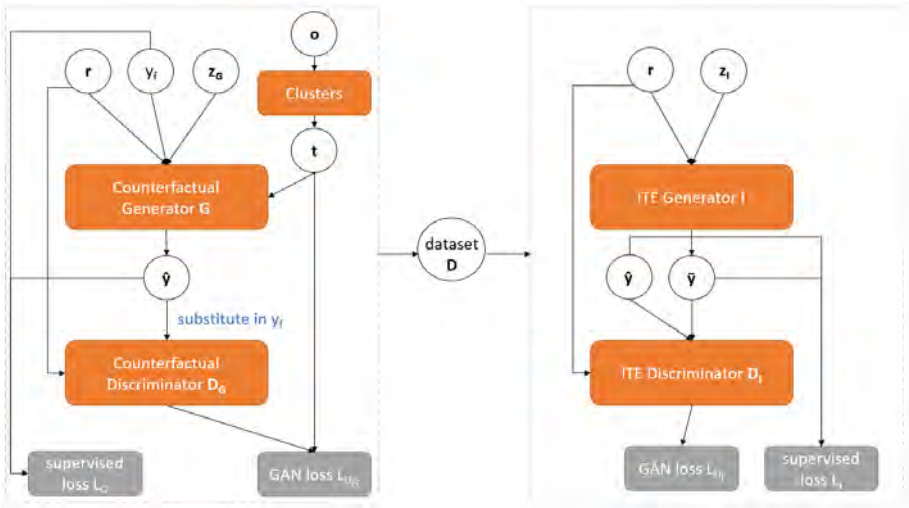


Fig. 7.1 GANITE based architecture. A GAN is built to generate a fully observed dataset, and another GAN is trained on the generated dataset to estimate counterfactuals. $\mathbf{z_G}$ and $\mathbf{z_I}$ are two random variables following a standard normal distribution.

For organ allocation:

1. With our proposed architecture, we could adapt more statistics to obtain a more comprehensive allocation policy. For instance, since we have estimations of potential outcomes and distributions of donors, for a newly arrived donor, we could calculate the expected improvement of transplantation outcome as:

$$EI(\mathbf{o},\mathbf{r}) = \mathbb{E}_{\mathbf{o}'}[\max\{outcome(\mathbf{o}',\mathbf{r}) - outcome(\mathbf{o},\mathbf{r}),0\}]$$

   By Monte Carlo approximation [20], we can approximate expected improvement as:

$$EI(\mathbf{o},\mathbf{r}) = \frac{1}{N}\max\sum_{i=1}^{N}\{outcome(\mathbf{o}'_i,\mathbf{r}) - outcome(\mathbf{o},\mathbf{r}),0\}$$

where $\mathbf{o'}_i$ are sampled from the distribution of donors. A high expected improvement suggests that there is a better organ for the recipient with high probability, which means maybe the organ should not be allocated to the recipient.

2. Many architectures designed for recommendation systems could be applied to the allocation problem. In recommendation systems, units are recalled and ranked. The allocation process is similar to the ranking process, where the ranking could be based on estimated counterfactuals and sequential information. Besides, some existing ranking algorithms are able to react to online transplantation outcomes.

3. The problem can be solved using sequence to sequence models. Instead of assigning a recipient to each arrived donor, models could be developed to assign a sequence of recipients to a sequence of donors. The donor sequence could be given, or a prediction model could be developed to predict future organ arrivals. For sequence to sequence model, many existing architectures are applicable, such as recurrent neural networks (RNN) [25], long short-term memory (LSTM) [10], and attention networks [29].

From a broader view, the problem we are considering is not only an organ allocation problem but can also be viewed as a resource allocation problem. Our work can be naturally translated to solve the resource allocation problem in other fields, such as cloud computing resource allocation and financial resource allocation. Traditional solutions to resource allocation are mainly dynamic programming and heuristic algorithms. More recent studies involving machine learning try to estimate the allocation outcome for each unit [35]. However, similar to organ allocation studies, none of the existing studies have tried to estimate all potential outcomes for each unit.

Additionally, this work could be extended to work on time-variant cases, where the recipient and donor features are changing over time. Innovative works [1, 5, 14] on time series can be applied to extend our proposed model's ability to work on time series data.

References

[1] Ahmed M. Alaa and Mihaela van der Schaar. “Attentive state-space modeling of disease progression”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 11338–11348.

[2] Ahmed M. Alaa and Mihaela van der Schaar. “Bayesian Inference of Individualized Treatment Effects using Multi-task Gaussian Processes”. In: *Advances in Neural Information Processing Systems 30*. 2017, pp. 3424–3432.

[3] Ahmed M. Alaa, Michael Weisz, and Mihaela van der Schaar. “Deep Counterfactual Networks with Propensity-Dropout”. In: *ICML Workshop on Principled Approaches to Deep Learning* (2017).

[4] Ahmed Alaa and Mihaela Schaar. “Limits of estimating heterogeneous treatment effects: Guidelines for practical algorithm design”. In: *International Conference on Machine Learning*. 2018, pp. 129–138.

[5] Ioana Bica et al. “Estimating Counterfactual Treatment Outcomes over Time Through Adversarially Balanced Representations”. In: *International Conference on Learning Representations* (2020).

[6] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.

[7] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[8] Xifeng Guo et al. “Improved deep embedded clustering with local structure preservation.” In: *IJCAI*. 2017, pp. 1753–1759.

[9] Negar Hassanpour and Russell Greiner. “Learning Disentangled Representations for CounterFactual Regression”. In: International Conference on Learning Representations. Sept. 25, 2019.

[10] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[11] M. Höfler. “Causal inference based on counterfactuals”. In: *BMC Medical Research Methodology* 5.1 (Sept. 13, 2005), p. 28.

[12] Fredrik Johansson, Uri Shalit, and David Sontag. “Learning representations for counterfactual inference”. In: *International conference on machine learning*. 2016, pp. 3020–3029.

[13] T. Kohonen. “The self-organizing map”. In: *Proceedings of the IEEE* 78.9 (Sept. 1990), pp. 1464–1480.

[14] Changhee Lee and Mihaela van der Schaar. “Temporal Phenotyping using Deep Predictive Clustering of Disease Progression”. In: *arXiv preprint arXiv:2006.08600* (2020).

[15] S. Lloyd. “Least squares quantization in PCM”. In: *IEEE Transactions on Information Theory* 28.2 (Mar. 1982), pp. 129–137.

[16] Christos Louizos et al. “Causal effect inference with deep latent-variable models”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 6446–6456.

[17] Jared K. Lunceford and Marie Davidian. “Stratification and weighting via the propensity score in estimation of causal treatment effects: a comparative study”. In: *Statistics in Medicine* 23.19 (2004), pp. 2937–2960.

[18] Dennis Medved et al. “Improving prediction of heart transplantation outcome using deep learning techniques”. In: *Scientific Reports* 8.1 (Feb. 26, 2018), p. 3613.

[19] Alfred Müller. “Integral probability metrics and their generating classes of functions”. In: *Advances in Applied Probability* (1997), pp. 429–443.

[20] Kevin P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[21] Johan Nilsson et al. “The International Heart Transplant Survival Algorithm (IHTSA): A New Model to Improve Organ Sharing and Survival”. In: *PLOS ONE* 10.3 (Mar. 11, 2015), e0118644.

[22] Judea Pearl. *Causality*. Cambridge university press, 2009.

[23] Karl Pearson. “LIII. On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[24] Donald B. Rubin. “Causal inference using potential outcomes: Design, modeling, decisions”. In: *Journal of the American Statistical Association* 100.469 (2005), pp. 322–331.

[25] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.

[26] Uri Shalit, Fredrik D. Johansson, and David Sontag. “Estimating individual treatment effect: generalization bounds and algorithms”. In: *International Conference on Machine Learning*. International Conference on Machine Learning. July 17, 2017, pp. 3076–3085.

[27] Peter Spirtes. “Introduction to causal inference.” In: *Journal of Machine Learning Research* 11.5 (2010).

[28] Bharath K. Sriperumbudur et al. “On the empirical estimation of integral probability metrics”. In: *Electronic Journal of Statistics* 6 (2012), pp. 1550–1599.

[29] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[30] Stefan Wager and Susan Athey. “Estimation and inference of heterogeneous treatment effects using random forests”. In: *Journal of the American Statistical Association* 113.523 (2018), pp. 1228–1242.

[31] Junyuan Xie, Ross Girshick, and Ali Farhadi. “Unsupervised deep embedding for clustering analysis”. In: *International conference on machine learning*. 2016, pp. 478–487.

[32] Liuyi Yao et al. “Representation Learning for Treatment Effect Estimation from Observational Data”. In: *Advances in Neural Information Processing Systems 31*. 2018, pp. 2633–2643.

[33] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. “GANITE: Estimation of Individualized Treatment Effects using Generative Adversarial Nets”. In: International Conference on Learning Representations. Feb. 15, 2018.

[34] Jinsung Yoon et al. “Personalized Donor-Recipient Matching for Organ Transplantation”. In: *Thirty-First AAAI Conference on Artificial Intelligence*. Feb. 12, 2017.

[35] Jixian Zhang et al. “Machine learning based resource allocation of cloud computing in auction”. In: *Comput. Mater. Continua* 56.1 (2018), pp. 123–135.

[36] Yao Zhang, Alexis Bellot, and Mihaela van der Schaar. “Learning Overlapping Representations for the Estimation of Individualized Treatment Effects”. In: *arXiv:2001.04754 [cs, stat]* (Feb. 17, 2020).

Appendix I: Pseudo-Code of Matching First Allocation Policy

Algorithm 1: Pseudo-code of Matching First

Input: list of waiting recipients "R", donor "d", trained counterfactual estimation model "CF", trained clustering model "CLS", list of estimated survival time of recipients without transplantation "S", threshold of maximum number of recipients to search "limit"

```
idx = 0;
Compute type of the given donor: dType = CLS.predictDonorType(d);
while idx < limit do
    r = R[idx];
    Estimated best donor type: bestDonorType = CF.predictBestDonorType(r);
    if dType == bestDonorType then
        Allocate d to r;
        return
    else

    end
    idx += 1;
end
Compute benefits of recipients: benefits = CF.predictFactual(R[:limit]) - S[:limit];
Allocate d to the recipient with the highest benefit
```

Appendix II: Full Structure & Hyperparameter Setting of SCCN

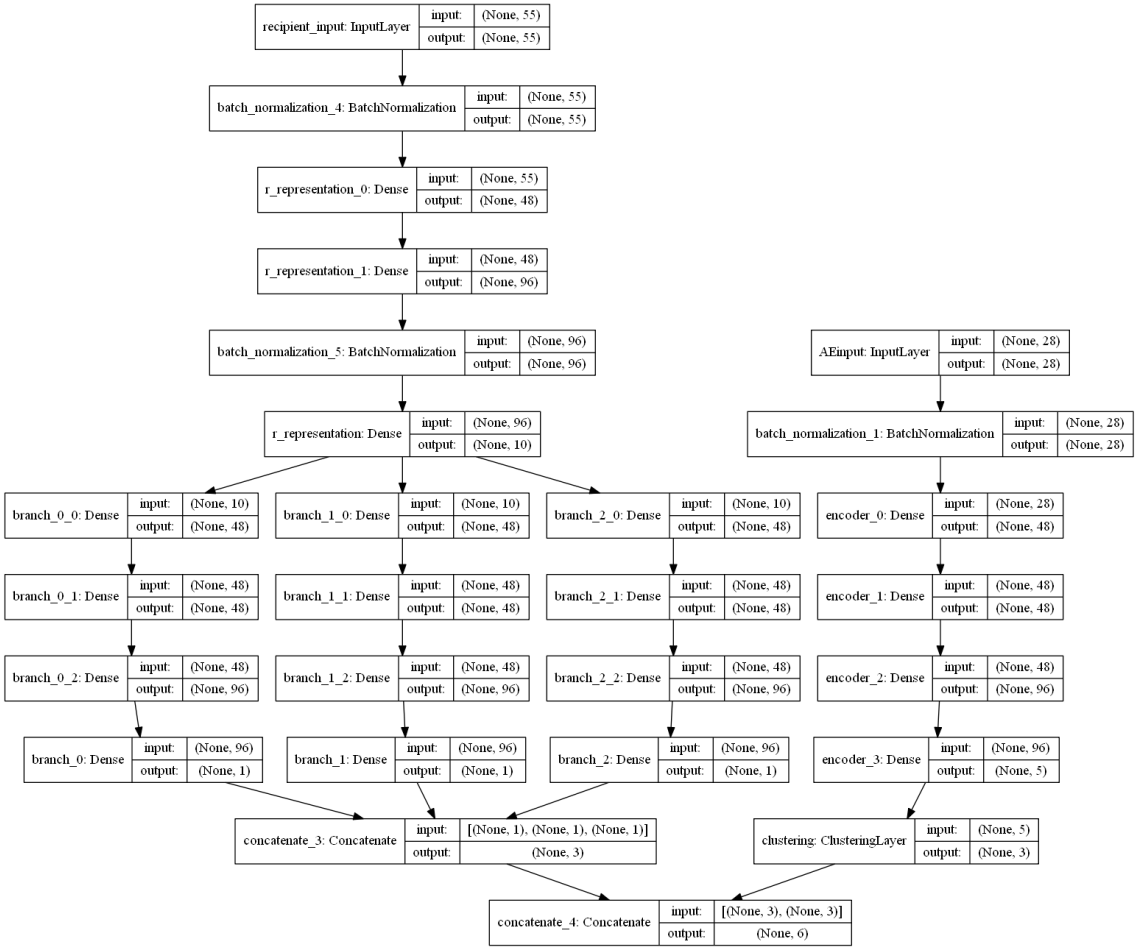


Fig. 9.1 The full structure of SCCN with the initial setting of the number of clusters being 3. On the right-hand side is the counterfactual estimation component, and on the left-hand side is the clustering component.

Hyperparameter	Settings of hyperparameters
Layer initialization	random normal initialization for weights and biases
Optimizer	Adam optimizer
Activation function	ReLU
Batch size	256
Hidden state dimension of representation network	48/96/10
Hidden state dimension of branched network	48/48/96/1
Hidden state dimension of clustering network	48/48/96/5/#clusters
α, β	0.2, 10

Table 9.1 Other hyperparameter settings of SCCN.

Appendix III: Source Code

See source code at:
https://github.com/CanXu960728/Matching-Networks-for-Organ-Allocation
which includes code, datasets, and detailed descriptions of datasets.