Multiple Treatment Effect Estimation using Deep Generative Model with Task Embedding

Shiv Kumar Saini
Adobe Research
Bangalore, Karnataka
shsaini@adobe.com

Sunny Dhamnani
Adobe Research
Bangalore, Karnataka
dhamnani@adobe.com

Aakash Srinivasan*
IIT Madras
Chennai, Tamil Nadu
s.aakash3431@gmail.com

Akil Arif Ibrahim*
IIT Roorkee
Roorkee, Uttarakhand
akilarif1997@gmail.com

Prithviraj Chavan*
IIT Kanpur
Kanpur, Uttar Pradesh
prithvi@cse.iitk.ac.in

ABSTRACT

Causal inference using observational data on multiple treatments is an important problem in a wide variety of fields. However, the existing literature tends to focus only on causal inference in case of binary or multinoulli treatments. These models are either incompatible with multiple treatments, or extending them to multiple treatments is computationally expensive. We use a previous formulation of causal inference using variational autoencoder (VAE) and propose a novel architecture to estimate the causal effect of any subset of the treatments. The higher order effects of multiple treatments are captured through a task embedding. The task embedding allows the model to scale to multiple treatments. The model is applied on real digital marketing dataset to evaluate the next best set of marketing actions. For evaluation, the model is compared against competitive baseline models on two semi-synthetic datasets created using the covariates from the real dataset. The performance is measured along four evaluation metrics considered in the causal inference literature and one proposed by us. The proposed evaluation metric measures the loss in the expected outcome when a particular model is used for decision making as compared to the ground truth. The proposed model outperforms the baselines along all five evaluation metrics. It outperforms the best baseline by over 30% along these evaluation metrics. The proposed approach is also shown to be robust when a subset of the confounders is not observed. The results on real data show the importance of the flexible modeling approach provided by the proposed model.

CCS CONCEPTS

• **Mathematics of computing** → **Causal networks**; *Bayesian computation*; *Variational methods*; • **Information systems** → **Online advertising**; • **Applied computing** → *Marketing*.

*Work done when the authors were affiliated with Adobe Research.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313744

KEYWORDS

Causal Inference; Individual Treatment Effect; Deep Generative Model; Variational autoencoder; Task Embedding; Counter-factual Analysis; What-if Analysis; Next Best Action; Multi-channel data

ACM Reference Format:
Shiv Kumar Saini, Sunny Dhamnani, Aakash Srinivasan, Akil Arif Ibrahim, and Prithviraj Chavan. 2019. Multiple Treatment Effect Estimation using Deep Generative Model with Task Embedding. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313744

1 INTRODUCTION

This paper provides a scalable approach for estimation of individual treatment effect using observational data in the setting where any subset of treatments from a set of treatments can be applied. The main motivation for this work is to provide a scalable approach for running counterfactual experiments on the effect of marketing actions on likelihood of conversion of a customer. The goal is to use the model for personalized online marketing actions. This problem is modeled as a causal treatment effect estimation problem where a subset of treatments can be applied. This setting is relevant for problems in a wide array of domains such as Economics [5], Medicine, Public Policy [14].

The prescribed approach for estimating causal effect is to conduct a randomized control trial. Unfortunately, running a randomized control trial is expensive and often not possible. Easy availability of observational data makes causal inference using observed data an attractive alternative to costly experiments. However, it is often the case that treatments in observational data were not assigned randomly. For example, a customer is targeted with an advertisement based on propensity of the customer to click the ad. Formally, the assignment of a treatment to a customer is not independent of the outcome. This can lead to misleading estimates of treatment effect [12]. In this particular example, the effect of the targeted advertisement will be overestimated when outcomes of "treated" and "untreated" customers are compared without any adjustment for the biased assignment. In addition to the biases due to assignment, there is likely a selection bias where an individual is self-selecting to be part of a treatment group [2, 16].

There is a large literature in various fields on causal inference using observational data. The Rubin Causal Model [29] and conditioning on observables [18], are two popular approaches to handle the bias due to assignment or selection on observables. There are recent papers that extend these models by using deep learning models (see [19, 31]). These approaches do not apply when the assignment or selection is based on unobserved variables. Selection on unobserved can be due to a user self-selecting treatment(s) on the basis of her wants and needs. These unobserved variable are called latent confounders. Under the assumption that there are observable proxies which can be used to recover these latent confounders, the causal effect is identifiable. Recent papers have used deep generative models to recover the causal effect in a setting where latent variables determine assignment [25, 34]. These models have shown promising result. Our approach follows this line of literature.

The main focus of this literature is the setting where the treatment is a binary [25] or a multinoulli [34] variable. The case where treatment can be any subset out of a set of treatments is less studied. The approach to deal with multiple treatments seems to be either to treat each subset as a separate treatment [24] or derive multiple treatment effect as a function of the treatment effects of the individual treatments in a subset. As an example for the latter, in a linear regression setting the effect of two treatments can be modeled as a sum of the individual effect of the two treatments plus an extra coefficient for the joint effect. Both of these approaches have drawbacks. Treating each subset as a separate treatment do not use the information across treatments. Besides, it is not a scalable approach if the model requires separate models for each treatment (see [19, 25, 31, 34] for such models). Another issue is that all subsets of treatments might not be observed. For example, in the real dataset used in this paper there are five treatments, which implies $2^5 = 32$ total subsets of the treatments. Only 26 out of 32 subsets are observed. Out of these 26 subsets, 8 are observed less than 10 times, 10 are observed between 10 to 100 times, and only 8 are observed more than 100 times. This makes it difficult for a large number of existing models to be applicable for the task at hand. The approach to impose a particular structure on the treatment effect of a subset has the obvious drawback that the functional form can be wrong.

Our approach overcomes both of these drawbacks. Our main contribution is that we use a task embedding [15] to scale the model presented in [25] for multiple treatments. We explicitly model the latent variables and the structure of the causal problem using a variational autoencoder [20]. The effect of multiple treatments is incorporated by using a task embedding to model the interdependence between the treatments. A task embedding allows a flexible representation of a task by multiplying a vector of zeros and ones, representing which treatments are applied, and a weight matrix which is learned. The resulting vector is passed on to the following layers of the network. The dimensions of the weight matrix are equal to the number of treatments times a constant which is a hyperparameter of the model. If two treatments have similar effect, then the columns corresponding to the two treatments will take similar values. If two treatment have similar values only for a subset of the observed covariates, this can also be represented by the task embedding. We call this model Task Embedding based Causal Effect Variational Autoencoder (TECE-VAE). The model is robust even when some of the subsets of treatments are not observed and uses information across observed subsets of treatments. The simple tweak of the architecture by incorporating a task-embedding provides a powerful tool to deal with multiple treatments.

The proposed approach, TECE-VAE, shows improved performance over a wide variety of competitive baselines. The model is compared against deep causal models as well as traditional models that have shown to work well under different settings. The baselines also include models that have done well in causal effect estimation competitions. The proposed approach is compared with the baselines on a semi-synthetic dataset with three treatments. The choice of the number of treatments in the semi-synthetic data is due to the difficulty in scaling some of the baselines to more than three treatments. Note that the proposed approach scales well to multiple treatment. Hence, the results are conservative from the perspective of the proposed model.

The result on semi-synthetic dataset shows that TECE-VAE improves upon the best baselines by over 30% on a variety of metric, including one proposed by us. Another set of experiments tests the robustness of the model when some of the covariates representing hidden confounders are not observed. The proposed approach continue to work well on the reduced set of covariates when 20%, 40%, 60% of the covariates are removed from the dataset. We also apply the model on a real world dataset to solve a real causal inference problem. The results indicate that assignment bias is a problem and explicitly modeling the latent variables is helpful.

Next, the literature related to this work is discussed.

2 RELATED WORK

In our setting, the assignment is based on observable as well as on latent variables. We make an assumption that there are proxies observed for the latent variables.

The closest paper to our work is [25]. This paper uses a Variational Autoencoder (VAE) [21] to model and estimate the joint distribution of the observed proxies, treatments, unobserved latent variables, and the outcome. Following Rubin's potential outcomes framework, this model uses a private path for each treatment. The assignment model decides which of these paths is taken. Due to a large number of potential outcomes where all combinations are not observed, this approach is not directly applicable in our setting.

There are other papers that have used deep models for causal inference. Johansson *et al.* [19] introduced Balancing Linear Regression (BLR) and Balancing Neural Network (BNN), which is used for counterfactual inference in the case of binary treatments. They formulate the problem of counterfactual inference as a domain adaptation problem. Shalit *et al.* [31] introduced Counterfactual Regression (CFR) and Treatment-Agnostic Representation Network (TARNet) for estimating individual treatment effect (ITE) from observational data in the case of binary treatments. The causal inference models in these papers use an architecture with private paths for each treatment. The reason given for choosing private paths is to preserve the effect of a treatment amidst the noise due to non-treatment covariates. This might happen because the number of non-treatment features is very large compared to the treatment features. Our model, TECE-VAE, does not use treatment specific private paths. Instead, we use a task embedding to represent each

combination of tasks. Even though the semi-synthetic data and the real world dataset have a large number of non-treatment feature, the estimated ITE in not noisy.

There are other recent papers that have used deep generative models for causal inference. Yoon *et al.* [34] introduced Generative Adversarial Nets for inference of Individualized Treatment Effects (GANITE), an approach using Generative Adversarial Network (GAN) [10] to estimate the individualized treatment effects in case of multiple treatments. We were not able to obtain the code for GANITE, hence excluded this model from the baselines.

In addition to the deep model, other models which are considered for comparison come from the set of models that have performed well in a causal inference competition [6]. These models are Random Forest (RF) [3], Gradient Boosting Machine (GBM) [8], and Bayesian Additive Regression Trees (BART) [4]. These are tree-based supervised learning approaches that work well in causal inference tasks [6]. The third one, BART, is a computationally intensive approach that took about 12 hours for one run on our semi-synthetic dataset. These approaches extend easily to multiple treatment. While the first two are not proposed as causal models, these can recover causal effect by conditioning on observable covariates.

In summary, the existing literature, with naive extensions for a subset of treatments, often introduces scalability issues since the number of possible subset of treatments increases exponentially with number of available treatments. Moreover these approaches might fail to account for the interdependence among the treatments. We address this gap in the literature by proposing a scalable model that addresses the problem of estimating individual treatment effect when any subset of the available treatments can be applied.

The next section describes the proposed approach in detail.

3 METHOD

The causal structure in Figure 1 depicts latent confounders ($\boldsymbol{z}$) that determine who gets what possible combination of treatments ($\boldsymbol{t}$). The latent confounders also influence the outcome (y). The proxy variables, denoted by $\boldsymbol{x}$, are noisy representations of latent confounders ($\boldsymbol{z}$) that are observed. In the context of online marketing, the casual structure can be described as follows: $\boldsymbol{x}$ represents observed online behavior of customers and $\boldsymbol{t}$ denotes the treatments which a marketer can apply (for example sending an e-mail, offering discount etc.) to improve the likelihood of conversion. In such a scenario, the unobserved intent or purpose $\boldsymbol{z}$ guiding such online behavior $\boldsymbol{x}$, not only influences the conversion rate y, but also marketer's choice of actions $\boldsymbol{t}$. Hence, $\boldsymbol{z}$ confounds both y and $\boldsymbol{t}$.

It has been established that the proxy variables ($\boldsymbol{x}$) should not be treated as usual confounders ($\boldsymbol{z}$) because it induces bias [9, 11, 13, 22, 27]. Since $\boldsymbol{z}$ is not observed, we estimate a representation of $\boldsymbol{z}$ and use it to control for confounding effects of the latent confounders to obtain an unbiased estimate for the causal effect of $\boldsymbol{t}$ on y.

We use variational autoencoder (VAE) for causal effect estimation. The formulation is influenced by Louizos *et al.* [25]. As noted earlier, we use task embedding to represent different subsets of treatments instead of using private paths. VAE allows recovery of the distribution of the latent confounders under the assumption that the proxies are observed for the latent confounders. Recent

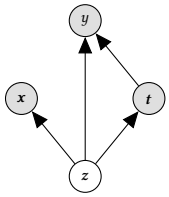

Figure 1: Causal graph where $\boldsymbol{x}$ represents observed proxies for confounders $\boldsymbol{z}$; $\boldsymbol{t}$ are treatments and y is outcome.

work on multiple treatments [33] show that weaker assumptions are sufficient to recover the causal effect of treatment when multiple treatments are applied. Under the assumption that there is no single cause confounder, the distribution of the latent confounders can be recovered from the assignment of the treatments in the observed data [33]. Note that we still need proxies for *all* latent confounders in case of *binary treatment*. Wang *et al.* [33] implies that the proxies for the latent confounders are not needed to identify the causal effect in case of *multiple treatments*. Inspired by this result, we also test the model when only a subset of the proxies are observed.

We can recover individual treatment effect under the causal model in Figure 1 if we can find the joint probability distribution $p(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{t}, y)$ [25, 26]. Using the causal structure, this joint probability distribution can be reduced to the product of the probability distributions $p(\boldsymbol{t}|\boldsymbol{z})$, $p(\boldsymbol{x}|\boldsymbol{z})$, $p(y|\boldsymbol{t}, \boldsymbol{z})$, and $p(\boldsymbol{z})$. These conditional probabilities are estimated using TECE-VAE.

First, let us introduce notations for describing the model. Let the total number of treatments be k. A subset of the treatments is denoted by a k-dimensional vector $\boldsymbol{t}$. Each element in the vector is a binary variable indicating whether a particular treatment is applied. For example, for $k = 5$, if the first and the third treatments are applied then $\boldsymbol{t} = [1, 0, 1, 0, 0]$. Let the dimension of the vector $\boldsymbol{x}$ be m. The proxy variables, $\boldsymbol{x}$, can be continuous, categorical, or binary. We denote these three types of variables by $\boldsymbol{x_c}$, $\boldsymbol{x_f}$ and $\boldsymbol{x_b}$, respectively. Similarly, the number of variables for each type is denoted by m_c, m_f and m_b, respectively. Finally, let $\boldsymbol{z}$ be a d length vector denoting unobserved latent confounders. The length of the latent confounders, d, is a hyperparameter. Experiments show that the results are not sensitive to the choice of d. This is a comforting result because the dimension of $\boldsymbol{z}$ is not known and sensitivity to the choice of d can make the application of the model to the real data less useful.

The proposed variation of VAE, like other autoencoder models, has a decoder and an encoder component. The causal VAE part is a specific architecture where the VAE mimics the dependencies shown in Figure 1. The design choices made in the architecture are either standard practices followed in VAE literature or are explicitly identified to be the contributions of this study. The details of the encoder and the decoder are presented next.

3.1 Encoder

In this part of the VAE, we recover the distribution of the latent counfounders $\boldsymbol{z}$. The posterior of $\boldsymbol{z}$ depends on $\boldsymbol{x}$, y and $\boldsymbol{t}$. Different parts of the network compute different probability distributions. Let g_i be the output of the layer as labeled in Figure 2. The treatment vector $\boldsymbol{t}$ is modeled as a vector of k Bernoulli random variables

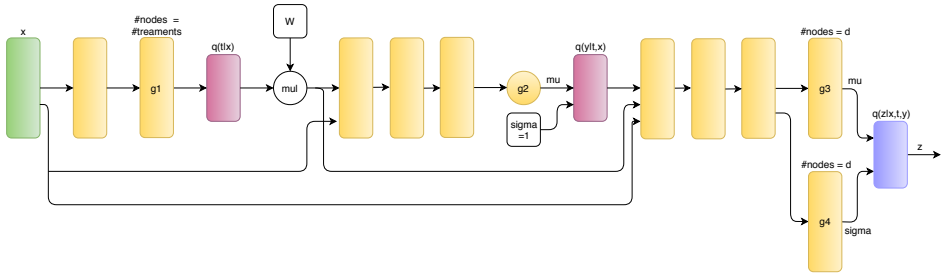

Figure 2: Encoder TECE-VAE, the encoder network takes input x to generate distribution for z. The green box refers to the input layer, input x is fed into yellow boxes that contain the network parameters to be trained. Purple boxes denote sampling step for various distributions. Note that weight matrix W is shown by an unshaded box.

which are independent given $\boldsymbol{x}$. Using the input $\boldsymbol{x}$, we obtain parameters for the distribution, $q(\boldsymbol{t}|\boldsymbol{x}) = \prod_{i=1}^{k} Bern(q_{\boldsymbol{t},i})$, where $q_{\boldsymbol{t},i}$ is the output of the layer labeled as g_1 in Figure 2.

The treatment vector, $\tilde{\boldsymbol{t}}$, is obtained by sampling from distribution $q(\boldsymbol{t}|\boldsymbol{x})$. Subsequently, we obtain a representation of the treatment vector $\tilde{\boldsymbol{t}}$ using the task embedding. The representation, denoted by $\boldsymbol{\tau}$, is obtained by projecting the k-dimensional treatment vector onto a k'-dimensional space ($\boldsymbol{\tau} \in \mathcal{R}^{k'}$) using an embedding matrix $\mathbf{W} \in \mathcal{R}^{k' \times k}$, $\boldsymbol{\tau} = \mathbf{W}\tilde{\boldsymbol{t}}$. The embedding weights, $\mathbf{W}$, are learned. Note that the embedding matrix $\mathbf{W}$ is shared between the encoder and decoder, in the next subsection it is shown how the decoder makes use of the embedding matrix. The dimension k' of the task embedding vector is a hyperparameter of the model. Such embeddings have been used in learning a curriculum of tasks in the domain of reinforcement learning [15]. By appropriately adjusting the dimensions of the embedding matrix $\mathbf{W}$, our model scales well for a large number of treatments. Furthermore, the use of task embedding accounts for the relationship between the treatments.

After finding $\boldsymbol{\tau}$, we concatenate it with $\boldsymbol{x}$ and feed it to the network to obtain parameters for $q(y|\boldsymbol{t},\boldsymbol{x})$ through g_2. For continuous y, we model $q(y|\boldsymbol{t},\boldsymbol{x})$ as $\mathcal{N}(\mu_{y_q}, 1)$. In case of binary outcomes, it is modeled as a Bernoulli distribution. We sample $\tilde{y}$ from the obtained distribution.

$$q(y|\boldsymbol{t},\boldsymbol{x}) = \mathcal{N}(\mu_{y_q}, 1), \quad \mu_{y_q} = g_2, \quad \boldsymbol{\tau} = \mathbf{W}\tilde{\boldsymbol{t}}$$

Finally, to compute posterior of $\boldsymbol{z}$ we estimate the mean and the variance using $\tilde{y}$, $\boldsymbol{x}$ and $\boldsymbol{\tau}$. The outputs i.e. $\mu_{\boldsymbol{z},i}$ and $\sigma_{\boldsymbol{z},i}$ are the outputs of the layers labeled g_3 and g_4 respectively.

$$q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{t},y) = \prod_{i=1}^{d} \mathcal{N}(\mu_{\boldsymbol{z},i}, \sigma_{\boldsymbol{z},i}), \quad \mu_{\boldsymbol{z},i}, \sigma_{\boldsymbol{z},i} = g_3, g_4 \tag{1}$$

The required joint probability distribution is the product of the probability distributions $p(\boldsymbol{x}|\boldsymbol{z})$, $p(\boldsymbol{t}|\boldsymbol{z})$, $p(y|\boldsymbol{t},\boldsymbol{z})$, and $p(\boldsymbol{z})$. Of these four terms, the distribution of latent confounders, $\boldsymbol{z}$, is output of the encoder part of the network. In the decoder network, we estimate the other three conditional probabilities.

3.2 Decoder

The latent confounder vector $\boldsymbol{z}$ is assumed to be vector of independent, univariate normal variables. The random variables that makeup the vectors $\boldsymbol{x_c}$, $\boldsymbol{x_f}$, and $\boldsymbol{x_b}$ are assumed to be independent of each other given $\boldsymbol{z}$ [25]. Every proxy variable in $\boldsymbol{x_c}$ is taken to be a univariate normal given $\boldsymbol{z}$. Similarly, every proxy variable in $\boldsymbol{x_b}$ is modeled as a Bernoulli(p) and every proxy variable in $\boldsymbol{x_f}$ is

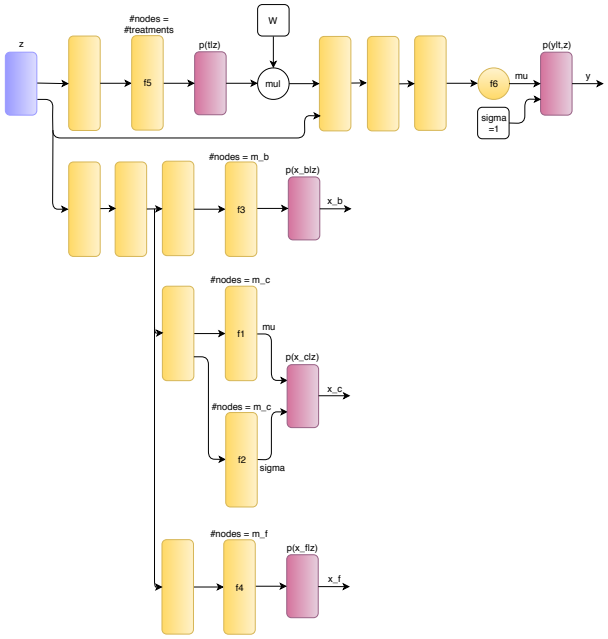

Figure 3: Decoder for TECE-VAE, the decoder network takes input z to reconstruct x, t and y. The input layer is depicted in blue, input z is fed into yellow boxes that contain the network parameters to be trained. Purple boxes denote sampling step for various distributions. Note that weight matrix W is shown by an unshaded box, this is the same matrix as used in the encoder network.

modeled as a categorical random variable that is independent of others conditional on $\boldsymbol{z}$.

Figure 3 shows the architecture of the decoder network. Different parts of decoder network predict different conditional distributions. Let f_i be the output of the layer as labeled in Figure 3. The decoder network works as follows: $\boldsymbol{z}$ is input to the network and the output of the layers labeled f_1 and f_2 are the parameters of the distribution $p(\boldsymbol{x_c}|\boldsymbol{z})$. Similarly, f_3 and f_4 output parameters for the binary and the categorical variables, respectively.

$$p(\boldsymbol{x_c}|\boldsymbol{z}) = \prod_{i=1}^{m_c} \mathcal{N}(\mu_{\boldsymbol{x},i}, \sigma_{\boldsymbol{x},i}), \qquad \mu_{\boldsymbol{x},i}, \sigma_{\boldsymbol{x},i} = f_1, f_2 \tag{2a}$$

$$p(\boldsymbol{x_b}|\boldsymbol{z}) = \prod_{i=1}^{m_b} Bern(p_{\boldsymbol{x},i}), \qquad p_{\boldsymbol{x},i} = f_3 \tag{2b}$$

$$p(\boldsymbol{x_f}|\boldsymbol{z}) = \prod_{i=1}^{m_f} Cat(\boldsymbol{q_{x,i}}), \qquad \boldsymbol{q_{x,i}} = f_4 \tag{2c}$$

Similar to encoder network the treatment vector $\boldsymbol{t}$ is represented by k Bernoulli random variables which are independent given $\boldsymbol{z}$. The conditional probability, $p(\boldsymbol{t}|\boldsymbol{z})$, is given by:

$$p(\boldsymbol{t}|\boldsymbol{z}) = \prod_{i=1}^{k} Bern(p_{\boldsymbol{t},i}), \qquad p_{\boldsymbol{t},i} = f_5 \tag{3}$$

Now, $\boldsymbol{x_c}$, $\boldsymbol{x_f}$ and $\boldsymbol{x_b}$ are sampled from corresponding distributions to obtain $\tilde{\boldsymbol{x}}$. Similarly $\tilde{\boldsymbol{t}}$ is sampled from $p(\boldsymbol{t}|\boldsymbol{z})$. For continuous y, we represent $p(y|\boldsymbol{t},\boldsymbol{z})$ as $\mathcal{N}(\mu_{y_p}, 1)$ and in case of binary outcomes we instead approximate with a Bernoulli distribution. In order to obtain $p(y|\boldsymbol{x},\boldsymbol{t},\boldsymbol{z})$ we transform the sampled $\tilde{\boldsymbol{t}}$ to $\boldsymbol{\tau}$ using the same embedding matrix $\mathbf{W}$. Next, $\boldsymbol{x}$, $\boldsymbol{z}$ and $\boldsymbol{\tau}$ are concatenated into a single vector which is then used to obtain the parameters for $p(y|\boldsymbol{t},\boldsymbol{z})$ as output of the layer f_6.

$$p(y|\boldsymbol{t},\boldsymbol{z}) = \mathcal{N}(\mu_{y_p}, 1), \quad \mu_{y_p} = f_6 \tag{4}$$

With equations 1, 2, 3 and 4, we can compute the joint probability distribution. We use variational inference to train the encoder-decoder. Note that during the training, we sample from $q(\boldsymbol{t}|\boldsymbol{x})$ and $p(\boldsymbol{t}|\boldsymbol{z})$. However during testing, if we want to find the effect of a particular combination of treatment, $\boldsymbol{t'}$, then we do not sample from these distributions. We instead directly pass $\boldsymbol{t'}$, which will be transformed into a state embedding. Also, observe that the embedding weights, $\mathbf{W}$, are the same for both the encoder and decoder. We have used variational inference algorithm available in the Python library Edward [32] for the training. The loss function is same as equation 10 in [25]. L2 regularization is added to the loss function, we then minimize this loss to estimate the model. The next section describes the datasets used for experimentation and associated proxy variables $\boldsymbol{x}$, outcome y, associated treatment vector $\boldsymbol{t}$ and confounders $\boldsymbol{z}$ for these datasets.

4 DATASETS

The proposed approach has been applied to three datasets. The first one is a user-conversion dataset for a product company that sells software subscriptions. This dataset contains outcomes for the marketing actions observed in the data. However, the evaluation of a causal model requires outcome for all potential marketing actions - observed or unobserved - for each customer. To evaluate the model, two semi-synthetic datasets are created using the covariates of the user-conversion dataset. The three datasets are described next.

User-Conversion Dataset: This dataset is constructed from the event level logs of the interaction events between a user and the organization's website as well as marketing actions. These interaction events come from four different data sources - web-analytics data, display ad impression data, email interaction data, and product usage data. We merge these data sources. Web-analytics data is stored in the form of a clickstream that records the online activities of an user on the company's website. Roughly, there is one row of data for each URL visited on the company's website. These visits include pages with information on the product features, product help, download trial versions or checkout. Each row contains information about the user's device, geography, source of visit, URL, time-stamp, product purchased etc. The display ad data provides information on ad impression on company's own website as well as other websites. This dataset stores which ads were shown to the user and whether the user clicked on the ad and many other ad-related properties. The email interaction dataset consists of email related information that was sent by the organization to the user. This dataset includes information such as whether a user opened the email, clicked on a link in the email, unsubscribed to emails from the company, description of the email, etc. Finally, product usage data contains information on a user's interactions with company's applications. Each row of the data stores information on the events such as application launch, application download etc. Each dataset described above uses the same identifier for the user, this identifier is used for merging the datasets.

The problem of a marketer working for the company is to decide the best subset of the treatments that can be applied to each user over the next three weeks. The set of treatments consists of five marketing actions which a marketer can potentially apply to each customer with varying degree of control. These are: 1) show paid search ads, 2) start a social media campaign, 3) send emails, 4) show ads on company's own website, and 5) show ads on other websites. One of the approaches to find the best set of actions is to estimate the `ITE` for each subset of the treatments. The action corresponding to the highest value of the `ITE` will be the optimal action for the user. The user-conversion information comes from the purchase events stored in web-analytics data. A user is said to be converted if he/she decides to makes a purchase in form of subscription to the company's product in the considered time-period.

In a supervised learning problem, the empirical design is such that the covariates are observed as of a time point. The outcome of interest or the target variable for the supervised model is observed for a period of time after the time point at which the covariates are observed. In our setting, where a marketer wants to ask counterfactual questions, with the goal of taking an action, the period of covariate observation is followed by a period during which applied treatments as well as the outcome are observed. The covariate observation period is three weeks in the user-conversion data. The applied marketing actions and the outcome event during the following three weeks are observed. Note that the marketing actions during the first three weeks will be part of the observed covariates, $\boldsymbol{x}$. The marketing actions applied after the purchase during the latter three week periods are ignored.

Once we have the stitched data and different periods for observing the user information, treatment applied $\boldsymbol{t}$ and conversion outcome y in place, we process the stitched data to create $\boldsymbol{x}$, $\boldsymbol{t}$ and y. The features $\boldsymbol{x}$ include Recency-Frequency-Monetary variables [7] as well as few columns are included as is. The recency features mainly accounted for how recently had the user interacted with the organization. The examples include how recently had the customer used a mobile application, interacted with an online-advertisement, opened a marketing email etc. The frequency features account for how often do these interactions occur. For instance, how frequently does a user launch an application, click on online-ads, check emails etc.

The set of five marketing actions are the treatments, $\boldsymbol{t}$. The treatments are assigned by the marketer to a user or self-selected by a user. For instance a marketer can send promotional emails to attract the customers for subscription or a customer can decide to search where a marketer can show a search ad. The outcome

variable y denotes if the user made a purchase. The purchase event in the data is subscription of the product.

In summary each row of the data contains user features $\boldsymbol{x}$, the treatment they were subjected to $\boldsymbol{t}$ and the outcome y if they subscribed to company's products/services. The dataset comprises of close to 1.2 million users. Due to the low percentage of the subscribers in the data, the negative class was down-sampled to create a balanced distribution of the subscribers and non-subscribers for training the models. The total number of features are 144. Out of these, 116 are continuous, 16 are Bernoulli, and 12 are categorical.

Semi-Synthetic Data: The original user-conversion dataset, as described above, contains the outcome corresponding to the observed subset of the treatments. However, for evaluating the models the outcome for every subset of the treatments is needed. This requires a dataset which contains outcome y for every subset of the treatments for each user. While the real-world dataset has five treatments, the synthetic data is limited to only three treatments. The number of treatments are limited to only three because we found it extremely hard to train many of the baseline models for five treatments.

Some of the baselines such as `BNN` [19], `BLR` [19], `CFR WASS` [31], and `TARNet` [31] train separate models for each subset of the treatments. In similar vein, `CEVAE` [25] employs separate paths for each subset of treatments. This implies that the number of parameters to be trained increases exponentially with the number of treatment. The problem is exacerbated by the fact that the number of observations in the data for each subset of the treatments is highly skewed. In the original user-conversion dataset, not all treatment subsets occur with the same frequency, some are frequently occurring treatments while some show up very rarely. This unbalanced distribution in the original dataset is shown in Figure 4. It shows a power distribution with no observation for 6 subset and a very few observations for another 10 or so subsets of marketing actions. This implies that for the treatment subset with fewer observations, the models will not be trained well, and the results will be poor.

Due to difficulty in estimating some of these baselines when the number of treatments are high, we restrict the total number of treatments to three. A dataset with fewer treatments is a conservative approach because it favors the baselines which are meant for a binary treatment.

Two semi-synthetic datasets are created to test how the different models perform in a setting that mimics the power law distribution of the observed subsets of the treatment vs when each subset appears uniformly. The frequency of the treatments in the uniform scenario is not completely equal but the differences are not stark. With these two settings, we not only test for the setting observed in a real world dataset like user-conversion data but also for the scenarios where the distribution for treatments is relatively uniform.

Three treatments imply a total of eight possible subsets of treatments. For evaluation of the causal effect, we need the `ITE` for each subset of the treatments for each individual user. Note that, to test the models in our setting (i.e. causal structure in Figure 1), we need observed proxies $\boldsymbol{x}$, latent confounders $\boldsymbol{z}$, observed treatments $\boldsymbol{t}$ assigned based on $\boldsymbol{z}$, and observed outcome y (for every treatment subset) determined by $\boldsymbol{z}$ and $\boldsymbol{t}$.

For generating the semi-synthetic datasets, we take the observed features, $\boldsymbol{x}$, from the user-conversion data. These are the observed proxies for the two semi-synthetic datasets. The semi-synthetic datasets are created using the following three steps:

(1) *Latent Confounder Creation:* Create latent counfouders $\boldsymbol{z}$ from the observed proxies $\boldsymbol{x}$.
(2) *Treatment Assignment Function:* Create the observed treatments as a function of the latent confounders for each user.
(3) *Outcome Function:* Create the observed outcome for each user and treatment subset as a function of the latent confounders and each treatment subset. Only the outcome corresponding to the observed treatment in step 2 is used for estimation. Other outcomes act as the ground truth for evaluation of the model.

Latent Confounder Creation: The observed proxies are then fed to an autoencoder [30] to model the hidden confounders. The middle layer of the autoencoder is used as the latent confounders. The size of the middle layer of the autoencoder is chosen to be 20. The latent values for each observation are used to generate the treatment to be assigned to each individual and the corresponding effect for each treatment. Note that the data generation approach used follows the causal structure in Figure 1, however, the data generating process is very different from the proposed model. Even the chosen dimensions of $\boldsymbol{z}$ for estimation and data generation are different. Next, the data generating process for obtaining the observed treatments and outcome is described in detail.

Treatment Assignment Function: As explained above, an autoencoder is used to obtain the latent confounders for each user. The Let Z be a matrix with n rows and d columns that contains the values of the latent confounders for each observation. Here n is the total number of observations and d is the total number of latent confounder variables. We set d to be 20 for the synthetic dataset. Let $\boldsymbol{t}_i$ be a row vector of length equal to the number of treatments. It specifies which subset of treatments was applied to i^{th} individual. Let k be the number of treatments. As discussed above, it is set to three. The outcome, $\boldsymbol{y}_i$, is a row vector that contains the `ITE` for each subset of the treatments for the i^{th} individual. The size of this vector is equal to the total number of possible treatment subsets, which is $2^3 = 8$ in this case.

We take a matrix θ, with d rows and the number of columns equal to the number of treatments. Each row $\theta_\mathbf{i}$ of θ is sampled from a standard multivariate normal distribution (mean is zero and covariance matrix is an identity matrix), i.e. $\theta_\mathbf{i} \sim \mathcal{N}(\mathbf{0}_{kX1}, \mathcal{I}_{kXk})$ where $i = 1, \ldots, d$.

We define a matrix μ, which is the difference between the matrix product of Z and θ and its mean taken across all the observations for balanced dataset. We subtracted each row of $Z \times \theta$ by the column-wise mean of $Z \times \theta$, which is a vector is size k. Let $mean_{all\ observations}(Z \times \theta)$ be a $n \times k$ matrix, with n duplicated rows containing column wise mean. For generating data with power law, -3 is added to all the elements of μ.

$$\mu = Z \times \theta - mean_{all\ observations}(Z \times \theta)$$

We took a covariance matrix Σ with k rows and k columns. The values that we chose are as follows:

$$\Sigma = \begin{bmatrix} 1 & 0.6 & 0.4 \\ 0.6 & 1 & -0.3 \\ 0.4 & -0.3 & 1 \end{bmatrix}$$

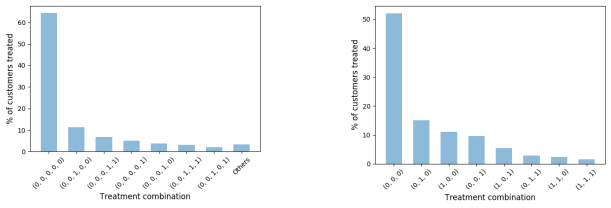

Figure 4: Distribution of treatments for user-conversion data (left) and unbalanced semi-synthetic data (right). Note that user-conversion dataset has 32 possible treatment subsets, we show top-7 in the graph while remaining 25 are aggregated to form *others* category. Unbalanced semi-synthetic dataset has 8 possible treatment subsets.

We define a row vector ϵ_i sampled from a multivariate normal distribution with mean μ_i (which is the i^{th} row of the matrix μ) and covariance matrix Σ.

An element of $\boldsymbol{t'(i)}$ is equal to 1 if the corresponding element in ϵ_i is greater than 0, otherwise 0. Hence $\boldsymbol{t'(i)}$ is a k-dimensional vector representing the treatments assigned to the i^{th} individual.

Outcome Function: We take a column vector $\boldsymbol{\beta}$ with length (d), where d denotes number of the latent confounder variables. Each element of $\boldsymbol{\beta}$ is sampled from a uniform distribution on the real interval [-10,10]. Let Z_i denote a column vector containing the i^{th} row of Z. We take a vector $\boldsymbol{\gamma}$ of size k. Each element of $\boldsymbol{\gamma}$ is sampled from a uniform distribution on the real interval [0,4]. Let y_{ij} denote the outcome of the i^{th} individual on applying j^{th} treatment vector $\boldsymbol{t}_j$.

The simulated outcome variable $y_{i,j}$ is given by the following equation:

$$y_{i,j} = {Z_i}^T \times \boldsymbol{\beta} + {\boldsymbol{t}_j}^T \times \boldsymbol{\gamma} + \xi_{i,j},$$

where ξ_{ij} is sampled from a normal distribution with mean as zero and variance as 10% of the absolute value of the sum of ${Z_i}^T \times \boldsymbol{\beta}$ and ${\boldsymbol{t}_j}^T \times \boldsymbol{\gamma}$. Hence for every individual i, the proxies($\boldsymbol{x}_i$), the treatment vector $\boldsymbol{t'(i)}$ and corresponding outcome $\boldsymbol{y}_i(\boldsymbol{t'(i)})$ forms the part of the training data. Figure 4 on the right shows the obtained treatment distribution for unbalanced semi-synthetic dataset.

The datasets described in this section are used to compare the proposed model with the baselines. In the next section we describe the details of the baselines and evaluation metrics.

5 EXPERIMENTS

In this section we describe experiments that compare the proposed approach to the existing treatment effect estimation techniques. Some of the baselines considered for the experiments extend naturally to multiple treatments, while others require some modification in the code. This section describes the changes made, the choice of hyperparameters and the evaluation metrics used for the experimentation.

5.1 Baselines

The baselines that naturally extend to multiple treatments are: k-nearest neighbor (k-NN), Random Forest (RF) [3], Gradient Boosting Machine (GBM) [8], and Bayesian Additive Regression Trees (BART) [4]. To implement k-NN, RF, and GBM, the python library scikit-learn [28] was used. To implement BART [4], the R package BayesTree[1] was used.

Other baseline models were created for binary treatment. These models are: Balancing Linear Regression (BLR) [19], Balancing Neural Network (BNN) [19], Counterfactual Regression with Wasserstein distance (CFR WASS) [31], Treatment-Agnostic Representation Network (TARNet) [31], and Causal Effect VAE (CEVAE) [25]. The available implementations for these models are not directly applicable to the multiple treatments setting. All of these baseline models except the last one were extended to multiple treatments setting, as done in [34]. All possible subsets of treatment can be taken as separate treatments and encoded as one-hot vectors. There are 2^k distinct treatment subsets so the size of one-hot vector is also 2^k. The models addressing the binary treatment-effect problem can be extended to multiple treatments by considering the problem as 2^k binary treatment-effect problems. And so the number of models required will be 2^k. The models that are extended to multiple treatment are BNN [19], BLR [19], CFR WASS [31], and TARNet [31]. We extended CEVAE [25] to handle multiple treatments. The number of distinct paths required is equal to the total number of treatment subsets 2^k, one path for each treatment subset.

For BNN [19], BLR [19], CFR WASS [31], and TARNet [31], the publicly available code[2] was used. For CEVAE [25], the publicly available code from github[3] was used. For the implementation of our model, Tensorflow [1] and Edward [32] were used. We used 63-27-10 % as train-validation-test split. We used the architecture as shown in figure 2 and 3. Unless specified we use 200 neurons per layer with *tanh* activation function. The size of $\boldsymbol{z}$ for CEVAE and TECE-VAE is 25. This was chosen based on experiments that showed that a large range of values for the dimension for $\boldsymbol{z}$ gives similar results. Figure 5 plots various evaluation metrics for different choices of the dimension of $\boldsymbol{z}$. The evaluation metrics do not change as the dimension of $\boldsymbol{z}$ is varied from 10 to 40.

These baselines as well as the proposed approach are compared using five causal inference evaluation metrics described next.

5.2 Evaluation Metrics

In the existing literature on binary treatments, the evaluation metrics used were RMSE of the estimated Individual Treatment Effect (ITE) [19], and absolute error in estimated Average Treatment Effect (ATE) [19, 25, 31]. These papers also report Precision in Estimation of Heterogeneous Effect (PEHE) [19, 25, 31] following [17]. We have used the evaluation metrics used in past as well as a few more that capture different aspects of a causal model used for choosing best subset of actions for each user. The evaluation metrics are described next.

The first two metrics - *RMSE* and *Absolute Error* - measure the difference between the ground truth and the estimated individual treatment effect (ITE) of the treatments. These metrics are used to evaluate the causal models in the multiple treatment effect literature [34]. In case of semi-synthetic data, the outcomes for all subsets

[1]The R package can be found at https://cran.r-project.org/package=BayesTree.
[2]Code available at https://github.com/clinicalml/cfrnet.
[3]Code available at https://github.com/AMLab-Amsterdam/CEVAE.

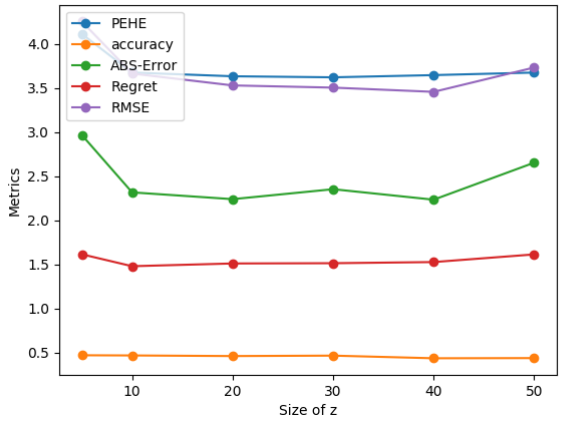

Figure 5: Values of various evaluation metrics for different dimensions of z, averaged over first three replications for unbalanced semi-synthetic data.

of the treatments are available in the data, and hence we calculate RMSE of the estimated outcomes for all subsets of the treatments. RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{|T|}\sum_{j\in T}(y_{i,j} - \hat{y}_{i,j})^2}$$

where, T is the set of all possible combinations of treatments, n is the number of observations, $y_{i,j}$ is the true outcome when j^{th} subset in T is applied on i^{th} observation, and $\hat{y}_{i,j}$ is the predicted outcome when j^{th} subset in T is applied on i^{th} observation. We also calculate absolute error in the same way as RMSE.

$$Absolute\ Error = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|T|}\sum_{j\in T}|y_{i,j} - \hat{y}_{i,j}|$$

The third metric, *Average PEHE Error*, measures the average of the RMSE between the actual and estimated difference between ITE for each treatment with no treatment ITE. This metric is used in binary treatment literature. We have extended it to multiple treatments setting. The subset with no treatments being applied is chosen as control and PEHE is calculated for each subset of all possible combinations of treatments with respect to control. Then, the error in PEHE is calculated for each treatment subset.

$$Error\ in\ PEHE_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((y_{i,j} - y_{i,0}) - (\hat{y}_{i,j} - \hat{y}_{i,0}))^2}, j \in (T - T0)$$

where, $T0$ is the subset with no treatments being applied (control) Average PEHE error is the mean of error in PEHE for all the subsets of all possible combinations of treatments.

$$Average\ PEHE\ Error = \frac{1}{|T|}\sum_{j\in(T-T0)} Error\ in\ PEHE_j$$

These three metrics measure the accuracy of the estimated causal effect with respect to the actual values. However, these do not capture the loss suffered due to the estimation error. To measure

Table 1: Out-of-sample metrics for the various models in the case of Unbalanced data. Bold indicates the model with the best performance for each metric.

Model Name	*RMSE*	*Absolute Error*	*Ave PEHE Error*	*Regret*	*Micro − Ave F − score*
RF	9.12 ± 0.84	5.35 ± 0.49	8.23 ± 0.94	3.40 ± 0.23	0.19 ± 0.03
k-NN	8.40 ± 0.49	5.09 ± 0.27	5.05 ± 0.30	5.71 ± 0.43	0.06 ± 0.01
GBM	6.71 ± 0.57	4.12 ± 0.35	6.29 ± 0.61	2.21 ± 0.18	0.40 ± 0.06
BLR*	15.74 ± 0.82	10.58 ± 0.65	5.41 ± 0.35	3.96 ± 0.46	0.11 ± 0.04
BNN-2-2*	15.43 ± 0.77	10.58 ± 0.59	5.51 ± 0.30	2.82 ± 0.38	0.22 ± 0.07
TARNet*	4.86 ± 0.22	3.38 ± 0.21	5.27 ± 0.32	2.62 ± 0.25	0.31 ± 0.05
CFR WASS*	8.10 ± 0.13	5.22 ± 0.11	6.15 ± 0.15	4.89 ± 0.14	0.08 ± 0.02
CEVAE*	22.79 ± 2.39	16.24 ± 1.78	16.49 ± 2.32	4.13 ± 0.39	0.11 ± 0.03
TECE-VAE	**4.01 ± 0.19**	**2.44 ± 0.12**	**3.95 ± 0.20**	**1.60 ± 0.18**	**0.51 ± 0.06**

* Original code modified for multiple treatments.

the loss with respect to the actual ITE, we propose a metric which is similar to the regret metric used in the Bandit literature [23]. Unsurprisingly, we call this metric *Regret*. It measures the loss due to using a particular causal inference model with respect to the ground truth. This is something that a marketer targeting customer for improving bottom line will deeply care about.

$$Regret = \frac{1}{n}\sum_{i=1}^{n}(y_{i,t_i} - y_{i,\hat{t}_i})$$

where, t_i is the true best action for i^{th} observation, and $\hat{t}_i$ is the predicted best action for i^{th} observation. In addition to these four metrics, we also compare micro-averaged F-score which compares the best action implied by an estimated model with the ground truth best action for each customer using the usual supervised learning metric - the F-Score.

Next section presents the results and discussions for these experiments.

6 RESULTS AND OBSERVATIONS

Table 1 compares the proposed approach with the baselines. This table present result on the semi-synthetic dataset that captures the skewness in the number of observation for each subset of treatments. There are 10 datasets created by changing the seed for the random draws. The evaluation metrics for each model are calculated on each of these 10 datasets. Table 1 presents mean and standard deviation of the evaluation metrics over the 10 runs. Note that we did not include the results for BART model because it takes about 12 hours for each run on the semi-synthetic data. The results for BART are included in Table 2 where the evaluation metrics are based on only one replication of the semi-synthetic dataset.

As shown in Table 1, the proposed approach TECE-VAE outperforms the baselines along each metric for the test data. The results for the training data are similar. Hence, the comparisons for the training data are not shown. The results are available on request. In the test data, TECE-VAE outperforms the closest baseline in terms of RMSE and Absolute Error, TARnet, by 21% and 39% respectively. This is a large improvement over the baselines. The implication is that, if one is interested in the *ITE* for each treatment, TECE-VAE provides a significant improvement over the baselines. In terms of *PEHE* and *Regret*, TECE-VAE outperforms the closest baselines by 33% and 38%, respectively. As noted earlier, *Regret* measures the loss due to applying the optimal subset of treatment for each

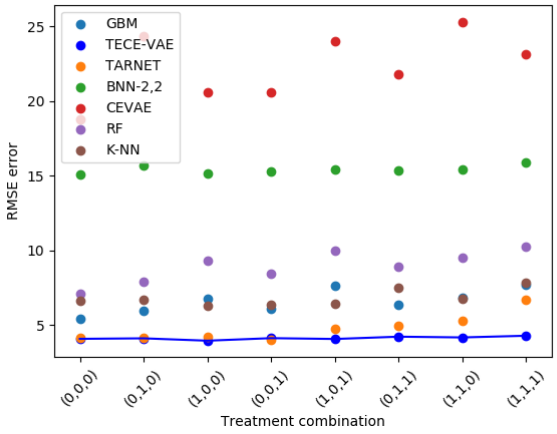

Figure 6: RMSE for each treatment subset averaged over 10 replications of unbalanced semi synthetic data. Treatments are sorted in descending order by the number of observations in the data.

customer using a particular causal inference model instead of the ground truth optimal treatments. This result shows that by using TECE-VAE the outcome, as measured by *Regret*, improves by at least 38%. The last metric is Micro F-Score, which measures accuracy of best ITE given by a causal model with respect to the ground truth optimal ITE. TECE-VAE outperforms the baselines along this metric as well.

Since, the number of observations for each subset of the treatments is different, a natural question is how the performance of the model varies by the number of times a particular subset of the treatments is observed. Figure 6 plots the *RMSE* metric for each treatment subset. The x-axis in the figure is sorted in decreasing order by the number of observation in the data. For example, the treatment subset (0,0,0) corresponds to no treatment and it is observed in the data for 39891 out of 69431 users; whereas the treatment subset (1,1,1) is observed only 9 times. As expected, the difference in the performance of TECE-VAE and other models increases as the number of observation for a treatment subset decrease. This result reinforces the assertion that the baseline models are not equipped to handle multiple treatments. Another observation from Figure 6 is that the performance of TECE-VAE as compared to TARnet starts to diverge around treatment subset (1,0,1). This subset of the treatments is observed 7244 times in the data, which is a large number. This implies that a large number of observations is needed to get comparable performance between the best baseline and TECE-VAE. As noted earlier, the main reason the model performs well is due to representation of treatments using a task embedding.

One question remains is how does TECE-VAE performs in a dataset where the distribution of the observed treatments is not skewed. To answer this question, we compared the models on a semi-synthetic dataset with uniform distribution of treatment subsets. Note that this dataset should be more favorable to the baseline techniques because many of these models estimate separate parameters for each treatment. The analysis on this dataset, presented in

Table 2: Out-of-sample metrics for the various models in the case of Balanced data. Bold indicates the model with the best performance for each metric.

Model Name	*RMSE*	*Absolute Error*	*Average PEHE Error*	*Regret*	*Micro − Averaged F − score*
k-NN	5.069	2.791	3.608	2.899	0.174
RF	4.516	2.430	3.449	2.147	0.261
GBM	3.781	2.072	3.434	1.617	0.440
BART	3.531	1.905	3.397	**1.533**	**0.450**
BLR*	15.305	11.207	3.440	1.985	0.193
BNN-2-2*	**3.142**	**1.848**	3.525	1.805	0.338
CFR WASS*	3.228	1.973	3.509	2.306	0.173
TARNet*	3.217	1.964	3.508	2.209	0.295
CEVAE*	4.598	2.429	4.337	1.938	0.226
TECE-VAE	**3.190**	**1.886**	**3.189**	1.656	0.298

* Original code modified for multiple treatments.

Table 2, shows that the proposed approach is among the top two models along all metrics. At the same time, the large difference between TECE-VAE and the baselines disappears. One interesting observation is that GBM, which is a supervised learning model and doesn't explicitly model the latent variables, works well when the observed treatment subsets are balanced. There are three important observations. First, if the number of treatments are balanced and the variance of the estimate is not needed, one can use GBM and expect reasonable results. Second, since GBM is available out of the box and easy to train, it should be an important baseline for any causal effect estimation model. Third, the comparison between TECE-VAE and CEVAE, where former uses a task embedding and latter uses private paths, reveal the benefits of representing treatments using a task embedding. Note that the number of treatments as well as number of observations per treatment subset are chosen to favor the baselines such as CEVAE. Even in this setting, TECE-VAE outperforms CEVAE along all evaluation metrics.

The next set of experiments are conducted to test the robustness of the proposed approach. The experiments so far assume that the complete set of proxies for the latent confounders is observed. However, in practice proxies for all latent confounders might not be observed. This call for testing the robustness of the proposed approach when a fraction of proxies are not observed. For this set of experiments, the semi-synthetic data generation process as well as the train-validation-test split is same as above. During model estimation, a fraction of proxies, $\boldsymbol{x}$, are not used for estimation. The comparisons are done for following four scenarios - all proxies observed, 20%, 40%, 60% proxies not observed. The proposed approach is compared against two baselines - TARnet and GBM. These two baseline models have been the second best models along two metrics - *RMSE*, which measures the accuracy of the estimate of ITE, and *Regret*, which measures the loss due to using an estimated decision rule as compared to the ground truth decision rule.

Figure 7 contains the results of these experiments. Each point in Figure 7 is an average of 10 runs where a different, randomly selected, subset of the observed proxies is removed. The repetitions ensure that the evaluation metrics do not suffer from the sampling bias. The comparison of the *RMSE* of the ITE for the three models shows similar trend with TECE-VAE being the best model. The comparison along the *Regret* metric shows interesting

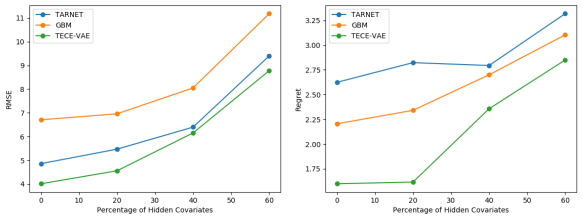

Figure 7: RMSE and Regret by 0%, 20%, 40%, and 60% covariates hidden, averaged over 10 replications for unbalanced semi-synthetic data.

results. TECE-VAE performance stays similar to 0% hidden proxies when 20% proxies are removed. The other two models show deterioration in performance. These results show that the proposed model is robust when a subset of proxies are not observed. Even the baselines are robust, albeit to a lesser degree. These results are in line with work recent which shows that in the case of multiple treatment the causal effect can be identified even when the latent confounders or proxies are not observed [33]. The intuition is that the assignment of multiple treatment provides information about the underlying latent confounders. A flexible model is able to utilize the observed assignment to recover the otherwise unobservable confounders and control for them to recover the causal effect. A deep generative model such as VAE provides a probabilistic factorization of the observed space in the form on the distribution of z. Wang *et al.* [33] show that such a representation identifies the unobserved confounders. The causal effect of multiple treatments is identified by controlling for this estimated representation of the input space.

User-Conversion Data: TECE-VAE is applied to the user- conversion dataset described in Section 4. The five treatments in this dataset are Search Ad, Social Ad, email sent, display ad in own domain, display ad outside own domain. The business question is to find the optimal subset of the treatments for each customer. Here, optimality is defined as the largest increase in the probability of achieving the desired outcome, which is a software subscription in this case. One might argue that some of the treatments cannot be chosen by a marketer. For example, a user searches and clicks on a search ad based on his or her own volition. It is not obvious how a marketer can influence this behavior. This is a valid point. However, the models are not used to directly target but inform targeting strategies. The counterfactual questions are hypotheticals asked to inform a decision. For example, TECE-VAE can be used to identify customers who can be influenced by a search ad. The profile data and search keywords of these customers can be used to find the profile of similar customer which can be used for targeting. A marketer can increase targeting of these keywords. Similarly, a customer matching the profile of someone influenced by a display ad can be sent display ad as and when the customer visits a website.

The results from the model show a few interesting findings. First, the prediction error as measured by RMSE of the predicted outcome is lower for TECE-VAE (0.397) than GBM (0.428) and RF (0.417), two popular supervised learning models. The RMSE of the observed outcome is not a suitable for evaluating a causal inference model. However, a causal inference model that fits the observed data is always desirable. One possible reason for the better performance might be due to explicit modeling of the latent state in TECE-VAE and use of a task embedding.

Second, the raw estimates of the treatment, as measured by the difference between the average conversion percentage for the consumer with treatment applied and for the consumer without any treatment, overestimates the effect of the treatment as compared with the average treatment effect measured by the model. For example, the raw estimate and the average treatment effect (ATE) for the treatment "Search Ad" are 46.5% and 21.8%, respectively; for "Social Ad" 37.8% and 13.8%, respectively; for "display ad outside own domain" 44.4% and 22.2%, respectively; for "display ad in own domain" 49.4% and 34.0%, respectively; and for "email sent" 22% and -3.42%, respectively. These difference in the two estimates show the extent of assignment and self-selection bias present in data. The effect of "email sent" is lowest among the treatments in both estimates. The ATE estimated using the model is a small but negative value! This neither means that this effect is negative for all individuals nor that sending email with other treatments in not optimal.

Third, the marketing actions that are actually applied are rarely optimal. The optimal marketing action suggested by TECE-VAE is actually observed only for 0.034% of the users. This might be due to the fact that some marketing action such as sending an email are easier than making someone actually search for an ad. However, the model shows that some marketing actions, such as sending and email or targeting with a social media campaign, might have a negative impact.

Fourth, in the test data only 15 out of a total of 32 subsets are optimal for at least one customer. Out of these only 8 subsets of the treatments are optimal for more than 50 users. None of these 15 optimal subsets are single treatment subsets. The subset that is optimal for the largest number of users is to apply all treatments but a social media campaign. Another important observation is that the effect of multiple treatments is not linear. For example, effect of showing an ad on own domain to everyone is a 11% points increase is likelihood of conversion. Whereas for a search ad to everyone increases the probability of subscription by 8% points. When the two treatment are applied jointly to everyone the increase in the probability is 23% points.

In summary, the results on real data show that the raw estimates from the data are highly biased, the use of model can lead to a large increase in the outcome metric, and the effect of multiple marketing actions is non-linear. These observations underline the need for a flexible model to to estimate ITE.

7 CONCLUSION

This paper provides a scalable approach for estimation of individual treatment effect using observational data in a setting where a subset from a set of treatments can be applied. We use task embedding to scale the model presented in [25] for multiple treatments. TECE-VAE is robust to unobserved treatments and uses information across treatments. We show the improvement in a variety of of experiments. TECE-VAE is compared against multiple baseline models that have shown to work well under different setting. The baselines also include models that have done well in causal effect estimation

competitions. TECE-VAE is applied on a real world dataset to solve a real causal inference problem.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Elias Bareinboim and Judea Pearl. 2012. Controlling Selection Bias in Causal Inference. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Neil D. Lawrence and Mark Girolami (Eds.), Vol. 22. PMLR, La Palma, Canary Islands, 100–108. http://proceedings.mlr.press/v22/bareinboim12.html

[3] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[4] Hugh A Chipman, Edward I George, Robert E McCulloch, et al. 2010. BART: Bayesian additive regression trees. *The Annals of Applied Statistics* 4, 1 (2010), 266–298.

[5] William B Dodds, Kent B Monroe, and Dhruv Grewal. 1991. Effects of price, brand, and store information on buyers' product evaluations. *Journal of marketing research* (1991), 307–319. https://www.jstor.org/stable/3172866?seq=1#page_scan_tab_contents

[6] Vincent Dorie, Jennifer Hill, Uri Shalit, Marc Scott, and Dan Cervone. 2017. Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *arXiv preprint arXiv:1707.02641* (2017).

[7] Peter S Fader, Bruce GS Hardie, and Ka Lok Lee. 2005. RFM and CLV: Using iso-value curves for customer base analysis. *Journal of marketing research* 42, 4 (2005), 415–430.

[8] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[9] Wayne A Fuller. 2009. *Measurement error models*. Vol. 305. John Wiley & Sons.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[11] Sander Greenland and Timothy Lash. 2008. Bias analysis. In *Modern epidemiology* (3 ed.), Kenneth J Rothman, Sander Greenland, Timothy L Lash, et al. (Eds.). Lippincott Williams & Wilkins, 345–380.

[12] Sander Greenland, James M Robins, and Judea Pearl. 1999. Confounding and collapsibility in causal inference. *Statistical science* (1999), 29–46.

[13] Zvi Griliches and Jerry A Hausman. 1986. Errors in variables in panel data. *Journal of econometrics* 31, 1 (1986), 93–118.

[14] Nabanita Datta Gupta and Marianne Simonsen. 2010. Non-cognitive child outcomes and universal high quality child care. *Journal of Public Economics* 94, 1-2 (2010), 30–43. https://www.sciencedirect.com/science/article/abs/pii/S0047272709001169

[15] Matthew John Hausknecht. 1995. Cooperation and Communication in Multiagent Deep Reinforcement Learning. , 114–117 pages.

[16] Miguel A. Hernán, Sonia Hernández-Díaz, and James M. Robins. 2004. A Structural Approach to Selection Bias. *Epidemiology* 15, 5 (2004), 615–625. http://www.jstor.org/stable/20485961

[17] Jennifer L Hill. 2011. Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics* 20, 1 (2011), 217–240.

[18] Guido W Imbens and Donald B Rubin. 2015. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press.

[19] Fredrik Johansson, Uri Shalit, and David Sontag. 2016. Learning representations for counterfactual inference. In *International Conference on Machine Learning*. 3020–3029.

[20] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[21] Diederik P Kingma and Max Welling. 2014. Stochastic gradient VB and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*.

[22] Manabu Kuroki and Judea Pearl. 2011. Measurement Bias and Effect Restoration in Causal In-ference. (2011).

[23] T.L Lai and Herbert Robbins. 1985. Asymptotically Efficient Adaptive Allocation Rules. *Adv. Appl. Math.* 6, 1 (March 1985), 4–22. https://doi.org/10.1016/0196-8858(85)90002-8

[24] Michael J Lopez, Roee Gutman, et al. 2017. Estimation of causal effects with multiple treatments: a review and new ideas. *Statist. Sci.* 32, 3 (2017), 432–454.

[25] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. 2017. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*. 6446–6456.

[26] Judea Pearl. 2009. *Causality*. Cambridge university press.

[27] Judea Pearl. 2012. On measurement bias in causal inference. *arXiv preprint arXiv:1203.3504* (2012).

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

[29] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* 66, 5 (1974), 688.

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1. MIT Press, Cambridge, MA, USA, Chapter Learning Internal Representations by Error Propagation, 318–362. http://dl.acm.org/citation.cfm?id=104279.104293

[31] Uri Shalit, Fredrik D. Johansson, and David A Sontag. 2017. Estimating individual treatment effect: generalization bounds and algorithms. In *ICML*.

[32] Dustin Tran, Alp Kucukelbir, Adji B Dieng, Maja Rudolph, Dawen Liang, and David M Blei. 2016. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787* (2016).

[33] Yixin Wang and David M Blei. 2018. The Blessings of Multiple Causes. *arXiv preprint arXiv:1805.06826* (2018).

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