Architecture Learning of Deep Neural Networks for Counterfactual Inference

Author One, Author Two, Author Three
Authors Details
Authors Email

Abstract

We propose a novel approach for automatically inferring appropriate architectures of deep neural networks for the task of counterfactual inference over observational data. The individualised causal effect of an intervention or treatment is modelled in terms of a multi-task learning problem using a deep neural network which consists of a number of layers that are shared among the factual and counterfactual outcomes and a number of outcome-specific layers. Our approach enables automatically selecting an appropriate architecture (i.e. number of shared and outcome-specific layers) by exploiting inferred characteristics of the dataset such as the propensity score, the extend of shared complexity between the two outcomes and the outcome-specific complexities. This way, we achieve an efficient method of model-selection while avoiding computationally expensive hyper-parameter searches over the space of possible architectures. We conduct experiments on a synthetic dataset allowing us to parametrize and fully control the characteristics of the data before applying our approach to a real-world observational study for which we infer the characteristics in order to derive an appropriate architecture. As shown in the experiments, our method outperforms the state-of-the-art.

Introduction

The technological advancements of recent years have resulted in an increasing availability of data in various fields such as healthcare, education, and economics. This data can be used to make predictions concerning unseen data points on the basis of statistical models. When dealing with observational studies, we are often particularly interested in the task of predicting the individualised treatment effect that certain intervention has on a given subject or context. In the case of electronic health records, for instance, a dataset typically consists of a set of patients each with individual features, a treatment assignment indicator (i.e. whether or not they received the treatment), and an observed outcome which we call the *factual outcome*. The quantity we are interested in is the *counterfactual outcome* (i.e. the outcome had the patient received a different treatment assignment) because it allows us to compute the individualised treatment effect helping us make informed decision during treatment planning.

Classical works have focused on estimating average treatment effects through variants of propensity score matching (Rubin, 2005; Austin, 2011; Abadie and Imbens, 2016; Rosenbaum and Rubin, 1983; Rubin, 1973). More recent works tackled the problem of estimating individualised treatment effects using representation learning (Johansson et al., 2016; Shalit et al., 2017), Bayesian inference (Hill, 2011), and standard supervised learning (Wager and Athey, 2015).

Recent works have shown that the problem can be effectively framed in terms of a multi-task learning problem using deep neural networks (Alaa et al., 2017). The network has a set of layers that are shared among both the factual and the counterfactual outcomes and a number of outcome-specific layers. However, the questions of how to select an appropriate architecture (i.e. the number of shared layers, and the number of outcome-specific layers) may drastically influence the expressiveness and computational complexity of the model and remains an open challenge. While there are various general approaches for model selection and architecture learning in neural networks, they do not make use of the specific nature of causal inference.

In this paper, we propose a novel approach for automatically learning appropriate architectures of deep neural networks for the task of counterfactual inference over observational data. This is achieved by exploiting inferred characteristics of the dataset such as the propensity score, and the "shared complexity" among the different outcomes and the difference between the outcome surfaces. For instance, if one of the outcomes follows a much more complex function than the other, this should be reflected in a potentially asymmetric architecture which utilises a higher number of outcome-specific layers for the more complex outcome.

Problem Formulation

We represent each subject i in our population with a d-dimensional feature vector $X_i \in \mathcal{X}$, and two *potential outcomes* $Y_i^{(1)}, Y_i^{(0)} \in \mathbb{R}$ which are drawn from a distribution $(Y_i^{(1)}, Y_i^{(0)}) \mid X_i = x \sim \mathbb{P}(\cdot \mid X_i = x)$. This way, the *in-*

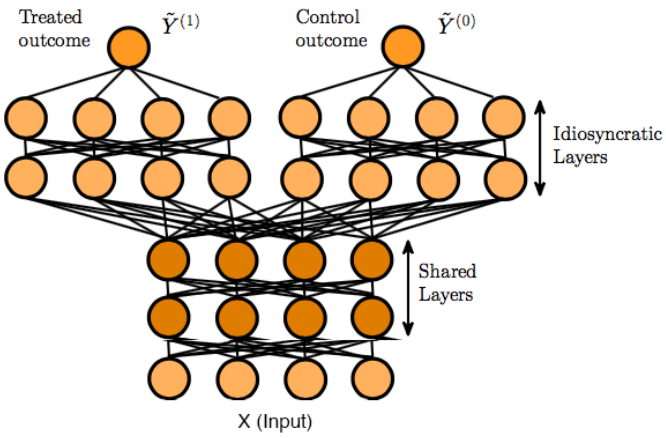

Figure 1: DRAFT: Architecture of a Deep Counterfactual Network (DCN). Objective is to learn appropriate values for number of different layers (here $L_s = L_{i,0} = L_{i,1} = 2$).

dividualised treatment effect for subject i can be expressed as

$$T(x) = \mathbb{E}[Y_i^{(1)} - Y_i^{(0)} \mid X_i = x]. \quad (1)$$

Given this definition, the objective is to approximate the function $T(x)$ using an observational dataset $\mathcal{D}$ consisting of n independent samples. Each sample is comprised of a tuple $\langle X_i, W_i, Y_i^{(W_i)} \rangle$, where X_i represents the subject's features, $W_i \in \{0, 1\}$ the treatment assignment indicator, and $Y_i^{(W_i)}$ and $Y_i^{(1-W_i)}$ the respective *factual* and *counterfactual* outcome. The treatment assignment is a random variable depending on the subjects' features, i.e. $W_i \not\perp X_i$. The assignment reflects a domain-specific policy which can be captured in terms of the probability $p(x) = \mathbb{P}(W_i = 1 \mid X_i = x)$ called the *propensity score*.

We are following the approach of (Alaa et al., 2017) and are using a *deep counterfactual network* (DCN) to infer $T(x)$ from $\mathcal{D}$. The DCN treats the problem as a multi-task learning problem using a deep neural network with an architecture illustrated in figure 1. The network uses a number L_s of shared layers, a number $L_{i,0}$ of idiosyncratic (outcome-specific) layers for the *treated outcome*, and a number $L_{i,1}$ for the *control outcome*.

We are interested in learning an appropriate architecture of the DCN, i.e. coming up with suitable values for L_s, $L_{i,0}$, and $L_{i,1}$.

Architecture Learning

We propose a novel approach of automatically learning a suitable architecture for the DCN by exploiting relevant characteristics of the dataset which are specific to the problem of causal inference. These characteristics such as the propensity score, the shared complexity of the response surfaces, and the individual complexity of each outcome function, can be inferred from the data to inspire a suitable architecture. This way, we achieve an efficient method of model-selection while avoiding computationally expensive hyper-parameter searches over the space of possible architectures.

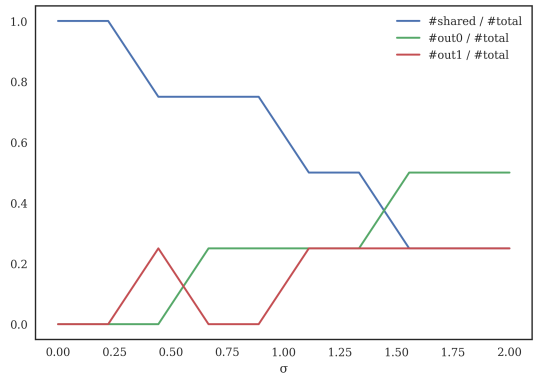

Figure 2: DRAFT: Influence of similarity parameter σ on the learnt optimal architecture with $L_{\text{total}} = 4$ fixed.

The approach is based on the observation that certain architectures are more suitable than others for the task of counterfactual inference. In particular, it can be shown empirically (see appendix) that the best-performing architectures consistently follow specific empiric ratios between the number of shared and total layers, and the respective idiosyncratic and total layers. The optimal ratios are not static, however, but depend on specific characteristics of the dataset. Once these characteristics are known, we can directly compute the desired number of layers and obtain a suitable architecture without the need to perform a hyper-parameter search.

In the following, we will discuss each characteristic in detail, including its intuition and formalisation. We describe how the characteristic can be inferred from the data and how it can be used to inform the architecture of the DCN.

Relevant Characteristics

For the problem of counterfactual inference, we have identified the following main characteristics of the data.

(a) Selection Bias Each dataset can be considered a partition of subjects into treated and untreated (control) subjects depending on their treatment assignment indicator W_i as described in the previous section. Governed by the underlying treatment policy the dataset might be heavily imbalanced and skewed towards a specific treatment. Intuitively, this potential imbalance is relevant for our model because we might profit from a corresponding asymmetric architecture, i.e. a model where the number of outcome-specific layers differ from each other.

Formally, the selection bias $\mathbf{B}$ can be quantified in terms of the average propensity score

$$\mathbf{B} = \frac{1}{n} \cdot \sum_{i=0}^{n-1} \mathbb{P}(W_i = 1 \mid X_i).$$

The individual propensity scores can be estimated from the dataset using a neural network for which we treat the prediction of the treatment assignment as a binary classification problem in supervised learning.

(b) Similarity of Outcome Response Surfaces The different outcome functions of the treated and untreated subjects can be conceptualised in terms of a shared part governed by features and their correlations that are mostly the same for both outcomes and an outcome-specific part that is inherently different.

Intuitively, if the outcome functions are potentially complex but rather similar to each other, this should be reflected in a high number of shared layers in the network in contrast to a relatively low number of outcome-specific layers.

Formally, we can model the different outcomes as functions

$$\begin{aligned} f_1(X_i) &= \underbrace{g(X_i, \lambda^{(1)})}_{\text{shared}} + \beta_1 \cdot \underbrace{exp(h_1(X_i, \mu^{(1)}))}_{\text{outcome-specific}} \\ f_0(X_i) &= \overbrace{g(X_i, \lambda^{(0)})}^{\text{shared}} + \beta_0 \cdot \underbrace{exp(h_0(X_i, \mu^{(0)}))}_{\text{outcome-specific}} \end{aligned} \tag{1}$$

where g represents a common function and h_0, h_1 are outcome-specific (e.g. polynomial functions of different degrees). The relative weight of the shared part in comparison to the outcome-specific part is captured by $\beta_0, \beta_1 \in \mathbb{R}$ (see below). In this model, the vectors $\lambda^{(1)}, \mu^{(1)}$ represent the coefficients of a specific response surface for the factual outcome whereas we model their counterfactual counterparts as

$$\lambda_i^{(0)} \sim \mathcal{N}(\lambda_i^{(1)}, \sigma) \qquad \mu_i^{(0)} \sim \mathcal{N}(\mu_i^{(1)}, \sigma). \tag{2}$$

This way, we can use the parameter $\sigma \in \mathbb{R}^0$ to formalise a measure of similarity $\mathbf{S} = \sigma$ between the two outcome surfaces. A low sigma corresponds to a high similarity which should be reflected in a large amount of shared layers, whereas an increasing sigma should result in a smaller proportion of shared layers.

There are multiple ways to estimate $\mathbf{S}$ for a new dataset. In our approach, we train two separate feed-forward neural networks – one exclusively for the treated subjects and one exclusively for the untreated subjects. After training, we compare the coefficients according to equation (2) and get an estimate for $\mathbf{S}$.

(c) Individual Complexity of each Response Surface In addition to the complexity that is shared across both response surfaces, the outcomes normally possess an individual part that is outcome-specific and follows a completely different type of function. For instance, one of the outcomes might be linear whereas the other outcome might be a polynomial function of a higher degree. Intuitively, the outcome with the more complex function should have a higher number of outcome-specific layers in our model in order to capture the more complex correlations between the features, leading to an overall asymmetric architecture.

We use the same model for the outcome functions as defined equations (1).

where g represents a common function and h_0, h_1 represent polynomial functions of different degrees (e.g. linear vs. quadratic). This way, we can use the parameters $\beta_0, \beta_1 \in \mathbb{R}^0$ to model the relative importance of the individual outcome-specific complexity in relation to the shared one.

Draft: How can we learn this from the data in a non-synthetic scenario?
Draft: How does it inform the architecture (number of layers) concretely?

Algorithm

Algorithm 1 Architecture Learning

procedure ARCHITECTURE LEARNING FOR DCN
Input: Dataset $\mathcal{D}$, L_{total}

 # Estimate Characteristics
$\tilde{p} \leftarrow$ *Learned via propensity network*
$\tilde{\sigma} \leftarrow$ *Learned via separate network*
$\tilde{\beta_0}, \tilde{\beta_1} \leftarrow$ *Learned via separate network*

 # Derive Architecture
$\tilde{p} \leftarrow$ *Learned via propensity network*
$\tilde{\sigma} \leftarrow$ *Learned via separate network*
$\tilde{\beta_0}, \tilde{\beta_1} \leftarrow$ *Learned via separate network*
Draft. See updated version.
Return: $L_s, L_{i,0}, L_{i,1}$

Experiments

The experiments are conducted on two different datasets. Firstly, we use a synthetic model which allows us to parametrise and fully control the characteristics mentioned in the previous section. This gives us the power to investigate how the different characteristics influence the performance of the learnt architecture. Secondly, we run the experiments on the *UNOS* dataset (consisting of information regarding patients who underwent an organ transplantation) to show how our approach generalises to a real-world dataset for which we do not have access to the characteristics directly but have to infer them from the data in order to learn a suitable architecture. In both cases, we compare the performance of a DCN whose architecture was learnt by our approach to a generic DCN and a number of other baseline approaches and architectures.

As we are dealing with counterfactual inference, we generally do not have access to the ground truth counterfactual outcomes, making it difficult to evaluate the performance of our predictions on real-world data. As a consequence, we adopt a semi-synthetic experimental setup (Hill, 2011;

Johansson et al., 2016) for which we use the original covariates and treatment assignments but simulate the outcomes according to a specific response surface described in detail in the next section.

Since it is our objective to learn an appropriate architecture by exploiting characteristics of the data, the number of (shared and outcome-specific) layers varies across the different datasets. However, throughout both networks we are using a fully-connected architecture with 200 hidden units in all layers (ReLU activation) and we evaluate the performance in terms of the mean squared error (MSE) of the estimated treatment effect. The datasets are split into a training set (80%) and test set (20%) and evaluated exclusively on the test set, averaging over 100 experiments for which new outcomes are drawn each time.

Synthetic Model

Data Generation For the synthetic model, we draw $n = 1000$ samples in the form of a tuple $\langle X_i, W_i, Y_i \rangle$ for each subject i. Each $X_i = (x_{i,0}, x_{i,1}, \ldots, x_{i,24})$ consists of $d = 25$ covariates for each subject which are independently drawn from a uniform distribution, i.e.

$$x_{i,0}, x_{i,1}, \ldots, x_{i,24} \overset{iid}{\sim} \mathcal{U}(0,1)$$

keeping each covariate strictly non-negative. For the treatment assignment indicator $W_i \in \{0,1\}$, we define

$$\tilde{p}(X_i) = \frac{1}{1 + exp(-\alpha \sum_{x_j \in X_i} x_j)}$$

$$W_i \sim Bernoulli(\tilde{p}(X_i))$$

where $\tilde{p}(X_i)$ represents the subject's propensity score and the parameter $\alpha \in \mathbb{R}^0$ gives us a way to control the imbalance between the treated and untreated subjects. For $\alpha = 0$, we get $\tilde{p}(X_i) = 0.5$ corresponding to a maximum balance between number of treated and untreated subjects whereas an increasing alpha shifts the distribution towards a higher proportion of treated subjects. We compute both outcomes $Y^{(0)}, Y^{(1)} \in \mathbb{R}$ as

$$Y_i^{(0)} = f_0(X_i) + \mathcal{N}(0,1)$$

$$Y_i^{(1)} = f_1(X_i) + \mathcal{N}(0,1)$$

which are governed by their corresponding outcome functions f_0 and f_1 defined as

$$f_1(X_i) = \sum_{x_j \in X_i} \lambda_j^{(1)} x_j + \beta^{(1)} \cdot exp(\sum_{x_j \in X_i} \mu_j^{(1)} x_j^2)$$

$$f_0(X_i) = \sum_{x_j \in X_i} \lambda_j^{(0)} x_j + \beta^{(0)} \cdot exp(\sum_{x_j \in X_i} \mu_j^{(0)} x_j).$$

Each outcome function consists of a linear part governed by the coefficients in the vectors $\lambda^{(0)}$ and $\lambda^{(1)}$ respectively, and an outcome-specific polynomial part inside the exponential function governed by the coefficients in the vectors $\mu^{(0)}$ and $\mu^{(1)}$. In the case of f_1 this is a quadratic function whereas for f_0 we are using a linear function. The parameters $\beta^{(0)}, \beta^{(1)} \in \mathbb{R}^0$ let us control the weight of the outcome-specific polynomial part in comparison to the common linear part.

Table 1: (Draft) Performance on Synthetic Dataset

Algorithm	**MSE**
k-NN	5.30 ± 0.30
Causal Forest	3.86 ± 0.20
BART	3.50 ± 0.20
BNN	2.45 ± 0.10
NN-4	2.88 ± 0.10
DCN	2.58 ± 0.06
DCN-LA	2.31 ± 0.05

The vectors $\lambda^{(1)}, \mu^{(1)}$ for the treated outcome function f_1 are drawn based on the data generation process designated as the "Response Surface B" setting in (Hill, 2012). For the outcome function f_0 of untreated subjects, we draw the vectors $\lambda^{(0)}, \mu^{(0)}$ from a normal distribution centred around their corresponding treated counterpart, i.e.

$$\lambda_i^{(0)} \sim \mathcal{N}(\lambda_i^{(1)}, \sigma) \qquad \mu_i^{(0)} \sim \mathcal{N}(\mu_i^{(1)}, \sigma),$$

with a standard deviation σ. This way, we can use the parameter $\sigma \in \mathbb{R}^0$ to control the similarity between the two outcome surfaces. Finally, we can set $Y_i = W_i \cdot Y_i^{(1)} + (1 - W_i) \cdot Y_i^{(0)}$ representing our *factual outcome* for subject i. The other (i.e. counterfactual) outcome is not used in the training set but needed later for evaluation purposes.

In summary, we receive a synthetic model which is parametrised by α, $\beta^{(0)}$, $\beta^{(1)}$, and σ each corresponding to a different characteristic we are interested in: The parameter α defines the skewness of the treatment assignment (i.e. the portion of treated vs. untreated subjects), $\beta^{(0)}$ and $\beta^{(1)}$ define the emphasis of the outcome-specific parts of the equation in relation to their common linear part, and σ determines the overall similarity between the two outcome surfaces.

Draft: Results and Discussion Table 1 shows the results of our evaluation of the learnt architecture (DCN-LA) in comparison with other existing methods.

Details of the experiment (total number of layers, epochs)

Description of the other methods

Discussion of impact of parameters

Graph: Impact of sigma as shown in figure 2

Graph: Impact of β_0 and β_1

Discussion of results

UNOS Dataset

In order to show how our approach generalises in a real-world scenario, we are running the experiments on the *UNOS* dataset.

Draft: About the dataset

Source and meaning of UNOS

Description of dataset (number of samples, covariates, etc.)

Relevant statistics

Treatment assignment and outcome

Draft: Results and Discussion

Description of how the characteristics are inferred from the dataset

Description of how the architecture is learnt using the characteristics

Discuss results table

Show graph and influence of different parameters

Table 2: (Draft) Performance on UNOS Dataset

Algorithm	MSE
k-NN	5.30 ± 0.30
Causal Forest	3.86 ± 0.20
BART	3.50 ± 0.20
BNN	2.45 ± 0.10
NN-4	2.88 ± 0.10
DCN	2.58 ± 0.06
DCN-LA	2.31 ± 0.05

Conclusions and Future Research

Counterfactual inference over observational data is of great importance in various areas such as healthcare, education, and economics. Deep neural networks are highly suitable for the task and represent the state-of-the art as they are able to capture complex relations in the outcome surfaces. However, it remains and open challenge of how to select an appropriate architecture for models. This is true in particular in the case of *deep counterfactual networks* which treat the problem as a multi-task learning problem with different numbers of shared and outcome-specific layers.

Our approach addresses this issue and provides an effective way to automatically learn a suitable architecture by inferring relevant characteristics from the data and incorporating them into the model selection. As shown in the experiments, our approach outperforms the state-of-the-art.

References

Abadie, A. and Imbens, G. W. (2016). Matching on the estimated propensity score. *Econometrica*, 84(2):781–807.

Alaa, A. M., Weisz, M., and van der Schaar, M. (2017). Deep Counterfactual Networks with Propensity-Dropout. *ArXiv e-prints*.

Austin, P. C. (2011). An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate Behavioral Research*, 46(3):399–424. PMID: 21818162.

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

Johansson, F., Shalit, U., and Sontag, D. (2016). *Learning representations for counterfactual inference*, volume 6, pages 4407–4418. International Machine Learning Society (IMLS).

Rosenbaum, P. R. and Rubin, D. B. (1983). The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55.

Rubin, D. B. (1973). Matching to remove bias in observational studies. *Biometrics*, 29(1):159–183.

Rubin, D. B. (2005). Causal inference using potential outcomes. *Journal of the American Statistical Association*, 100(469):322–331.

Shalit, U., Johansson, F. D., and Sontag, D. (2017). Estimating individual treatment effect: generalization bounds and algorithms. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3076–3085, International Convention Centre, Sydney, Australia. PMLR.

Wager, S. and Athey, S. (2015). Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *ArXiv e-prints*.