Grokking-like effects in counterfactual inference

Spyridon Samothrakis, Ana Matran-Fernandez, Umar Abdullahi, Michael Fairbank, Maria Fasli

***Abstract*—We show that a typical neural network, which ignores any covariate/feature re-balancing, can be as effective as any explicit counterfactual method. We adopt the architecture of TARNet—a simple neural network with two heads (one for treatment, one for control) which is trained with a relatively high batch size. Combined with ensemble methods, this produces competitive results in four counterfactual inference benchmarks: IHDP, NEWS, JOBS, and TWINS. Our results indicate that relatively simple methods might be good enough for counterfactual prediction, with quality constraints coming from hyperparameter tuning. Our analysis indicates that the reason behind the observed phenomenon might be "grokking", a recently developed theory.**

I. Introduction

Counterfactual inference from observational data is a fundamental problem in data analysis [1, 2, 3]. Observational data are often readily available in large quantities [4], as opposed to data collected through thoroughly-designed randomised-trial experiments, which are time consuming and/or impossible to perform. For instance, consider a study on the effects of drink-driving, where getting people to drink and drive (to form a treatment group) might be seen as unethical. Collecting observational data on drink-driving is, however, a much simpler endeavour, as the data could, for example, come from existing historic records. It is thus, extremely important to have easy-to-use algorithms for the observational setting. Beyond this clear-cut use-case, it has also been postulated that counterfactual inference is fundamental to the way we do data science and to the out-of-distribution generalisational abilities of modern classifiers [5]. If our models somehow manage to learn the invariances of the world, they should be able to generalise much better in the standard supervised-learning setup, as distributional changes will not impact the quality of our estimators.

Modern work in counterfactual inference using neural networks builds upon a series of methods that attempt to perform feature (in the standard machine learning terminology) rebalancing [6]. This creates a host of mechanisms (some of them inspired by traditional counterfactual inference, some new) that aim at tackling the problem.

In this paper, we follow a very simple approach to perform counterfactual inference, inspired by traditional neural network training regimes. We trained a simple neural network to learn from the data using a single hidden layer of tanh neurons. We also combined multiple of these networks in classic ensembles (namely, bagging and boosting) and measured their ability to learn counterfactual effects directly. Lastly, we explored the influence of adding propensity scores as an additional input feature to the neural network for each of these variants and performed hyperparemeter optimisation. We show that even without any explicit feature rebalancing, neural networks (when using the right hyperparameters) can still competitive with respect to more advanced methods specific to counterfactual inference. We hypothesise that the phenomenon observed is similar to recently observed instances of rapid improvement after excessive training, called "grokking" [7].

The rest of this paper is organised as follows: we discuss the necessary background in Section II. We present our own method, which we call Simple TARNet (STARNet), in Section III. This is followed by a presentation of the results in Section IV. Section V closes the paper with a brief discussion of promising avenues that remain open for future work.

Correspondence: Spyridon Samothrakis, Institute for Analytics and Data Science, University of Essex, UK; email: `ssamot@essex.ac.uk`

II. Background

We start by introducing the potential outcomes framework, which can be broken down into the following components: a set of allowed observations $\mathcal{X}$, a binary set of treatments $t \in \{0, 1\}$, and a set of possible outcomes $\mathcal{Y}$. As an example, an observation, $x \in \mathcal{X}$, could include the attributes describing a person; the treatment values could represent {`smoker`, `non-smoker`}; and the outcome could represent the additional life expectancy in years, such as $\mathcal{Y} = [5, 10]$. We call $Y_t(x)$ the potential outcome of treatment t on the individual described by x [8, 9].

The fundamental problem of counterfactual inference is that it is not possible to simultaneously observe the outcomes for both the treatment and the control for each example: if one is a smoker (known as the factual outcome), there is no way of knowing what would have happened if they had never smoked (referred to as the counterfactual outcome). We could only postulate this through a series of assumptions, namely ignorability, stable unit treatment value, and common support [1, 8]. These properties effectively declare that the *backdoor criterion* [1] is true in the case we are examining. When these assumptions hold, one can define the Individualised Treatment Effect (ITE) as:

$$\text{ITE}(x) = Y_1(x) - Y_0(x) \qquad (1)$$

Due to the inability to simultaneously know both $Y_0(x)$ and $Y_1(x)$ (unless they are obtained from simulations), it is not usually possible to calculate the ITE directly from the data. Most often, a regressor is used to try and approximate the function $\hat{y}_t(x) \approx Y_t(x)$, for all x and t. The regressor is then used to "fill in" the missing elements. Depending on how one goes into finding this approximate estimation of counterfactual values, we have T-learners (for two separate regressors), S-learners (for a single regressor with some special properties)

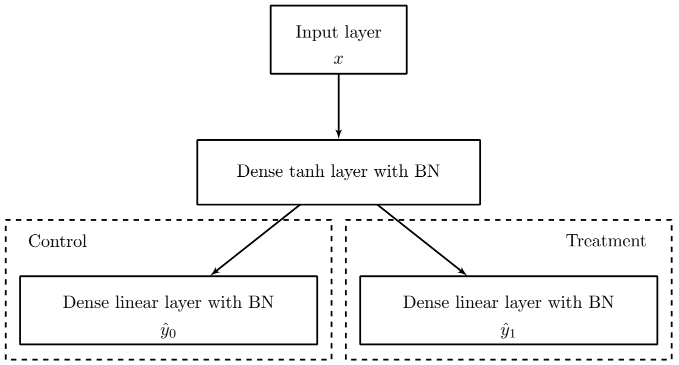


Fig. 1: STARNet architecture. Note that this is simply a TARNet, with the addition of batch normalisation (BN) in all layers, i.e., add-ons to make convergence easier.

and X-estimators (which perform some kind of meta-learning on T-learners) [10].

Another important measure is the Average Treatment Effect (ATE), which captures the average effect of an intervention on the entire population. This quantity is defined as:

$$\text{ATE}(\mathcal{X}) = \mathbb{E}_{x \sim p(x)} \left[\text{ITE}(x) \right] \tag{2}$$

where $p(x)$ is the distribution of x values in $\mathcal{X}$—this can be calculated empirically by averaging the individual treatment effects. Notable and popular methods for estimating the effects of interventions from observational data include propensity score methods [11] and domain adaptation [6]:

1) Propensity score methods: The *propensity score* is the probability that an individual gets assigned to the treatment group, given their observable feature set. Propensity score methods are quite often seen as central to counterfactual inference [11, 12]. One such method, from which we draw inspiration in this work, is termed covariate adjustment [13]. This simply means that the propensity score is calculated and added as an extra input feature to the estimator. Vansteelandt and Daniel [14] proved that such a linear regressor will behave identically to an ordinary linear regressor trained using a propensity-weighted cost function.

2) Domain Adaptation: It was recently understood that the problem of counterfactual inference could be seen through the lens of domain adaptation [6]. Domain adaptation methods align the control and treatment distributions in the feature space, so as to stop the estimator from being able to make predictions based on features other than the treatment. This can lead to an improved estimation of $\hat{y}_t(x)$. In a sense, this is very close to using a propensity-weighted cost function. Literature in this topic introduced the Treatment-Agnostic Representation Network (TARNet) neural architecture [15], which is highly reminiscent of relevant reinforcement learning efforts, with each action/treatment approximated by its own linear layer. We discuss this further in Section III, as part of our own modifications.

III. Methodology

A. STARNet

The core substrate for our experiments is a simple neural network with a single hidden layer of tanh units in an S-learner setup, which we present in Fig. 1. The hidden layer is coupled with a batch-normalisation layer. The network has two heads: one for the control outcome and one for the treatment outcome. We call our counterfactual neural network "Simple TARNet architecture" **(STARNet)**.

Let $\mathcal{D}$ denote a given dataset. For each individual $i \in \mathcal{D}$, there is a set of observations describing the individual, x_i, a parameter, $t_i \in \{0, 1\}$, detailing whether the treatment was given to the individual, and the observed outcome, $Y_{t_i}(x_i)$. This list of data for each individual may have been obtained by passive observation of an existing database, i.e., not obtained under the idealised controlled experimental conditions one would normally require for counterfactual analysis.

The system was trained by minimising a Mean-Squared Error (MSE) loss function with respect to the weight vector w of the neural network:

$$\begin{aligned}\mathcal{L}(\mathcal{D}, w) = \frac{1}{|\mathcal{C}| + |\mathcal{T}|} \Bigg(&\sum_{i \in \mathcal{C}} \left(Y_0(x_i) - \hat{y}_0(x_i)\right)^2 \\ &+ \sum_{i \in \mathcal{T}} \left(Y_1(x_i) - \hat{y}_1(x_i)\right)^2 \Bigg),\end{aligned} \tag{3}$$

where $\mathcal{C}$ is the subset of individuals who are in the control group (i.e., those who received no treatment), $\mathcal{T}$ is the subset of individuals in the treatment group, and $\mathcal{C} \cup \mathcal{T} = \mathcal{D}$. $\hat{y}_0(x_i)$ denotes the left-head output of the network (corresponding to its estimate for the effect of no treatment) and $\hat{y}_1(x_i)$ denotes the right-head output of the network (corresponding to the estimate for the effect of the treatment). $Y_0(x_i)$ and $Y_1(x_i)$ represent the corresponding ground-truth effects of non-treatment or treatment on individual i respectively. During training, (3) is intended for use with balanced mini-batches, such that an equal number of individuals from $\mathcal{C}$ and from $\mathcal{T}$ are selected. We note that this has an implicit effect of inserting a coefficient $|\mathcal{C}|/|\mathcal{T}|$ in front of the second summation in (3).

To test this, we also implement a modified version of STARNet that includes propensity scores calculated for each dataset. This modified architecture is shown in Fig. 2, where $\hat{P}(t = 0|x)$ and $\hat{P}(t = 1|x)$ represent the neural network's estimates of the probability that a particular individual is in the treatment/control group respectively, based only on the input features x. These two probabilities are fed as an extra input to the final double-headed layer, as shown in Fig. 2, to potentially enhance the overall network's ability to estimate $\hat{y}_0$ and $\hat{y}_1$. The loss function used for propensity score calculation is the simple cross entropy,

$$\begin{aligned}\mathcal{L}_{prop}(D, w) = -\frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \Bigg(&(1 - t_i) \log\Big(\hat{P}(t_i = 0|x_i)\Big) \\ &+ t_i \log\Big(\hat{P}(t_i = 1|x_i)\Big)\Bigg),\end{aligned} \tag{4}$$

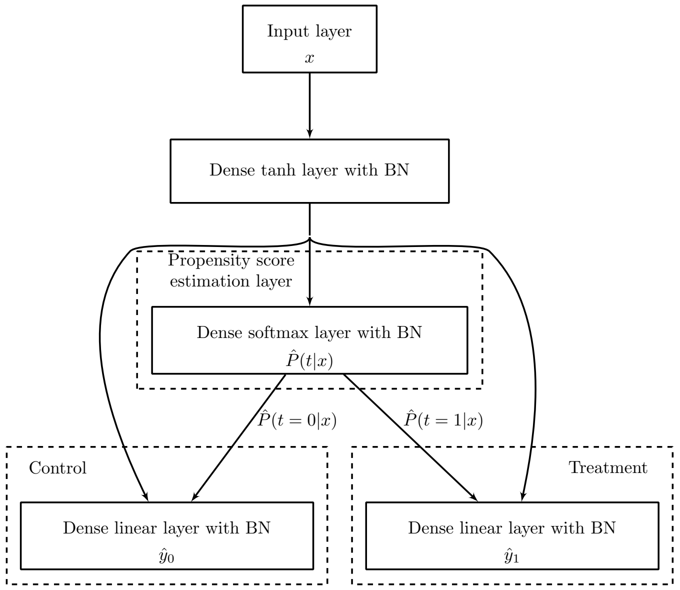


Fig. 2: STARNet architecture with propensity scores.

which is added to (3), where $t_i \in \{0, 1\}$ is the true label for whether the individual has been treated or not.

B. Training details

We trained the network using mini-batches of size 1024 sampled with replacement from the training set. Each mini-batch was chosen to include an equal number of treatment and control samples, namely 512 samples with $t_i = 0$, and 512 samples with $t_i = 1$. If there were less than 512 samples in either group, we used all samples—this did somewhat remove the balancing effect, but not severely enough to impact performance. We pre-processed all features (and outputs, in the regression benchmarks) from the datasets to be in the range $[-1, 1]$ and used one-hot encoding for categorical features.

Each layer of STARNet used L_2 regularisation with a coefficient of 0.001 and varied the number of neurons in the hidden layer, $N = 2^0, 2^1, 2^2, \ldots, 2^6$, in our experiments. We also tested the case where there is no hidden layer, in which case our setup becomes two separate linear estimators (i.e., a T-learner)—one for treatment and one for control. We refer to this case as "Linear" in Section IV.

C. Benchmarks

We evaluate the performance of STARNet and its variants on the four most widely used benchmarks in counterfactual inference, namely: Infant Health Development Program (IHDP), NEWS, JOBS, and TWINS. In this section, we give a short overview of each of them.

1) ***IHDP dataset****:* The Infant Health Development Program dataset was collected to investigate the effect of high-quality home visits and childcare on the future cognitive test score of a child [16]. We used the semi-simulated dataset from [17]. The dataset contains 25 features, including measurements about the child and the mother, and behavioural information about the pregnancy. The sizes of the treatment and control groups are 139 and 608 respectively. The outcome is the cognitive test score for the child. More details of the IHDP dataset can be found in [6, 15, 17].

2) ***JOBS dataset****:* The JOBS benchmark [18] is a combination of Lalonde's experimental data from the National Supported Work Program (NSWP) [19] and an observational study from the Panel Study of Income Dynamics (PSID) [20]. The experimental study from the NSWP is comprised of 297 participants who received training (the treatment group) and 425 control subjects that did not. The PSID observational comparison group contains an additional 2490 control individuals. The outcome in this dataset is the employment status (employed or unemployed) of each individual. We used the 8-feature set of [20], which contains information such as education, previous salary, and age.

3) ***NEWS dataset****:* The NEWS benchmark [6] is a simulated dataset in which the outcome to predict is the opinion of a customer exposed to news items for different topics either on a mobile (control group) or a desktop device (treatment group).

4) ***TWINS dataset****:* This benchmark is based on the register of twin births in the US between 1989–1991 [21, 22]. Of interest are twin pairs of same sex weighing less than 2 kg each. In this setting, the treatment variable represents whether a given twin is the heavier amongst the twin pair, and the outcome is the mortality of each twin within their first year of life, which is known from the birth register. This can be seen as having access to the outcomes of both treatment and control.

D. Performance Metrics

In order to measure the performance of STARNet, and compare it with other competitive efforts, in this section we present the metrics commonly used in counterfactual inference problems.

*1) Average Treatment Effect (*ATE*):* We define the error in the average treatment effect, ATE_ε, as

$$\begin{aligned}\text{ATE}_\varepsilon = \frac{1}{|\mathcal{D}|}\Bigg| \sum_{i\in\mathcal{D}} (\hat{y}_1(x_i) - \hat{y}_0(x_i)) \\ - \sum_{i\in\mathcal{D}} (Y_1(x_i) - Y_0(x_i)) \Bigg|\end{aligned} \tag{5}$$

Note that usually only one of the Y_t outcomes is available (the factual, which can be either treatment or control), making the calculation of ATE_ε impossible—unless, as in most benchmark scenarios, the datasets are simulated.

*2) Precision in Estimating Heterogeneous Treatment Effect (*PEHE*):* PEHE measures the quality of the counterfactual predictions of a model and its ability to reproduce the ground truth [17]. We measure the error in the Precision in Estimating Heterogeneous Treatment Effect (PEHE_ε) as:

$$\mathrm{PEHE}_\varepsilon = \Bigg(\frac{1}{|\mathcal{D}|}\sum_{i\in\mathcal{D}}\big([\hat{y}_1(x_i)-\hat{y}_0(x_i)] - [Y_1(x_i)-Y_0(x_i)]\big)^2\Bigg)^{1/2} \tag{6}$$

*3) Average Treatment on Treated (*ATT*):* For the JOBS dataset, since the treated individuals are part of the randomised control subset of the dataset (denoted as $\mathcal{D}_r$), while $\mathcal{C}$ is the control group, the true Average Treatment on Treated (ATT) can be computed. The ATT is defined by:

$$\mathrm{ATT} = \frac{1}{|\mathcal{T}|}\sum_{i\in T} Y_1(x_i) - \frac{1}{|\mathcal{C}\cap\mathcal{D}_r|}\sum_{i\in C\cap\mathcal{D}_r} Y_0(x_i) \tag{7}$$

while the error term, ATT_ε, is defined as:

$$\mathrm{ATT}_\varepsilon = \left|\mathrm{ATT} - \frac{1}{|\mathcal{T}|}\sum_{i\in T}(\hat{y}_1(x_i)-\hat{y}_0(x_i))\right| \tag{8}$$

4) Policy Risk ($\mathrm{P}_{\mathrm{risk}}$*):* The absence of (all) ground truth for the JOBS dataset prevents us from using ATE_ε and $\mathrm{PEHE}_\varepsilon$ for evaluation. Instead, we use the policy risk, $\mathrm{P}_{\mathrm{risk}}$, proposed by [15]. This quantity measures the average loss in value when a treatment is carried out according to the policy implied by an ITE estimator. Let

$$\Omega_y(x) = \begin{cases} 1 & \text{if } y_1(x)-y_0(x) > \lambda \\ 0 & \text{otherwise,} \end{cases}$$

where λ is a real parameter which can be tuned, but 0 is a sensible default. The policy risk is defined by:

$$\mathrm{P}_{\mathrm{risk}}(\Omega_y) = 1 - \Big(\mathbb{E}[Y_1|\Omega_y(x)=1] * p(\Omega_y(x)=1) + \mathbb{E}[Y_0|\Omega_y(x)=0] * p(\Omega_y(x)=0)\Big) \tag{9}$$

We calculate the policy risk from the randomised control subset of the data for $\lambda = 0$.

5) Counterfactual AUC: In binary outcome settings, provided one has access to the counterfactual truths, we can measure the Area under the Receiving Operating Curve (AUC) [23]. This is to be measured on the counterfactual samples of the test set (i.e., for any user we have data for treatment, we would measure the control case and vice versa).

IV. Results

A. Benchmarks

We performed our experiments using the setup described in Section III-B for each of the datasets, varying only the size of the hidden layer of the STARNet architecture. The networks were trained for 20,000 iterations in all cases. The validation sets were merged with the training set, and were not used to stop training (or anything else; see following subsections on validation information and selecting hyperparameters). We also created a bagging STARNet and a boosting STARNet (with the help of the bagging and AdaBoost implementations in scikit-learn [24] respectively) using ensembles of 10 (for IHDP and NEWS) or 50 (for JOBS for and TWINS) STARNet estimators[1].

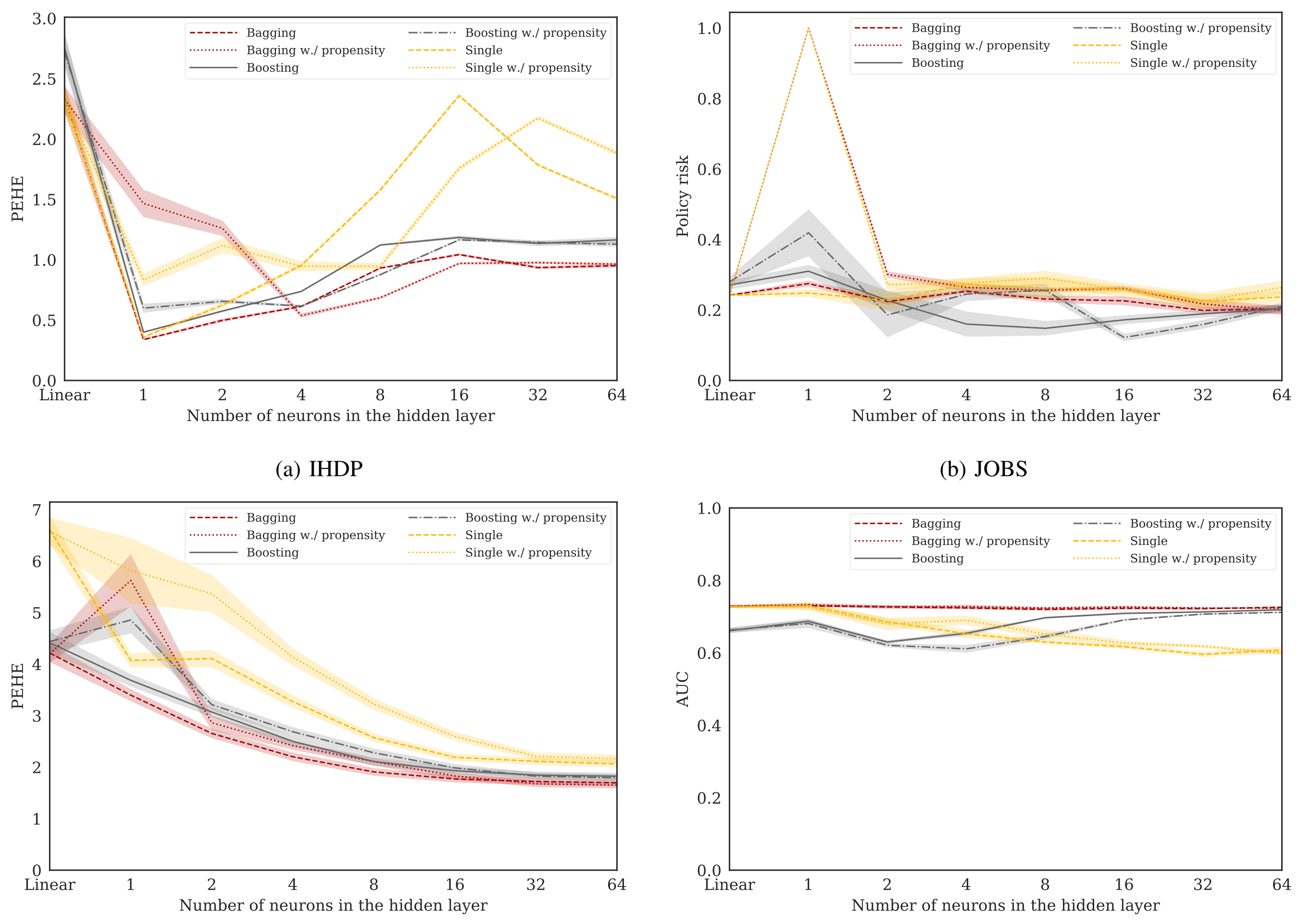
Fig. 3 shows the results of each of the STARNet variants on the four benchmarks, as a function of the size of the hidden layer. In Table I, we compare STARNet to other competitive methods (with the corresponding reference from which the results were extracted). For each of the STARNet variants we specify N, the number of neurons in the hidden layer for which we obtained the lowest ATE_ε, $\mathrm{PEHE}_\varepsilon$ or ATT, or the highest AUC. For each dataset, we have highlighted the best-performing algorithm.

In the IHDP dataset, we obtained the best performance for bagging STARNet with $N = 1$, both in terms of $\mathrm{PEHE}_\varepsilon$ (see Fig. 3a) and ATE_ε (reported in Table I). Increasing the size of the hidden layer decreases performance (i.e., higher $\mathrm{PEHE}_\varepsilon$) for the ensemble variants of STARNet. Our method easily outperforms the previous competitive results, provided the right number of neurons is set. In Fig. 3c, we can observe the performance of STARNet on the NEWS benchmark. In this case, we observe the opposite pattern than for IHDP: increasing N improves the performance of our models. Our best result in terms of ATE_ε is obtained for boosting STARNet with $N = 64$, but it should be noted that all the ensemble variants are capable of beating the competitive results previously reported, and the single-estimator methods (STARNet with and without propensity scores) remain on a par with them (see Table I). Our best result in terms of predicting individual outcomes, measured by $\mathrm{PEHE}_\varepsilon$, is obtained for bagging STARNet with propensity scores and $N = 64$, with $\mathrm{PEHE}_\varepsilon = 1.65 \pm 0.06$ (almost 1/4 of the $\mathrm{PEHE}_\varepsilon$ for linear STARNet), also beating the previous competitive method, BNN-2-2 [6], which stands at $\mathrm{PEHE}_\varepsilon = 2.0 \pm 0.1$.

If we now compare our results with our selection of methods, we beat a previous top performer (CEVAE [22]) in terms of ATT with bagging STARNet with propensity scores for $N = 16$ (see Table I). In terms of the quality of individual predictions however, the best result is obtained for boosting STARNet with propensity scores and $N = 64$, with $\mathrm{P}_{\mathrm{risk}} = 0.12 \pm 0.01$.

Following the pattern of the JOBS benchmark, the results for the TWINS dataset are reported in terms of the AUC in Fig. 3d (note that, in this case, higher is better, as opposed to the previous metrics). For the ensemble variants of STARNet, performance is stable or increasing for $N > 8$. In contrast, for the single estimator STARNet, performance drops with increasing N. Our best result in terms of the AUC is $\mathrm{AUC} = 0.73 \pm 0.00$, obtained for the linear case, bagging STARNet with $N = 1$ (both of them with and without propensity scores), and boosting STARNet with propensity scores, but it remains below the the strong baseline from [22], as highlighted in Table I. If we look at the average performance over the population, measured by ATE_ε, the best performers

[1]These numbers were chosen purely on the basis of training time constraints.



(a) IHDP (b) JOBS (c) NEWS (d) TWINS

Fig. 3: Results of STARNet for all datasets, for different hidden layer sizes. (a) Error in the PEHE for predicting the cognitive test score for the children in the IHDP dataset (lower is better); (b) Policy risk for predicting the employment status of the participants of the JOBS dataset (lower is better); (c) Error in the PEHE for predicting readers' experience in the NEWS dataset (lower is better); (d) Area under the curve for predicting the mortality of the unobserved twin in the TWINS dataset (higher is better). Shaded areas represent the standard deviation around the mean.

are the two versions of boosting STARNet, which outperform the results from [22].

Finally, if we compare the versions of STARNet that use propensity scores vs. the "standard" STARNet versions, it is clear from the subplots in Fig. 3 that adding propensity scores does not always result in significant performance improvements, and one needs to discover when this addition is beneficial. Overall, smaller networks tend to perform better. Without any special "counterfactual" alignment, STARNet is able to uncover the counterfactual structure, provided the correct hyperparameters (in this case, N) are found. We explore hyperparameter exploration in Sections IV-C–IV-D.

B. Grokking

Recent literature has demonstrated counter-intuitive observations can occur during neural-network training [7]: networks seem to forgo the traditional dynamics when it comes to overfitting. The "traditional" neural network wisdom was that in order to prevent overfitting, one must train on their data for as long as the validation dataset performance is not dropping. Modern research points towards a direction where, following an initial short-lived drop, validation performance picks up again after more training [28]. In the extreme, following a long period of overfitting, the network suddenly improves its performance rapidly; this phenomenon is known as "grokking" [7]. In Fig. 4 we show the Out-Of-Bag (OOB) R^2 estimates on the training set and smoothed PEHE_ε and ATE_ε on the test set as STARNet was training on the IHDP benchmark, starting at training iteration 8,000. The OOB score approximates leave-one-out cross-validation [29] and, combined with bagging/boosting, can be a very efficient way of dealing with small datasets.

Even though R^2 (on both the training and the test sets) has

Method	IHDP		JOBS		NEWS		TWINS	
	ATE$_\varepsilon$	PEHE$_\varepsilon$	ATT	P$_{\text{risk}}$	ATE$_\varepsilon$	PEHE$_\varepsilon$	ATE$_\varepsilon$	AUC
OLS	–	–	–	–	0.2 ± 0.0	3.3 ± 0.2	–	–
D ROBUST [6]	0.2 ± 0.0	5.7 ± 0.3	–	–	0.2 ± 0.0	3.3 ± 0.2	–	–
BART	–	2.3 ± 0.1	0.08 ± 0.03	0.25 ± 0.0	0.2 ± 0.0	3.2 ± 0.2	–	–
	[25]		[15]		[6]			
R Forest	–	6.6 ± 0.3	0.09 ± 0.04	0.28 ± 0.0	–	17.39 ± 1.24	–	–
	[25]		[15]		[26]			
C Forest	–	3.8 ± 0.2	0.07 ± 0.03	0.20 ± 0.02	–	17.59 ± 1.63	–	–
	[25]		[15]		[26]			
BNN-2-2 [6]	0.3 ± 0.0	1.6 ± 0.1	–	0.24 ± 0.02	0.3 ± 0.0	2.0 ± 0.1	–	–
TARNet [15]	0.28 ± 0.01	0.95 ± 0.0	–	0.21 ± 0.01	–	17.17 ± 1.25 [26]	–	–
CFR-WASS [15]	0.27± 0.01	0.76 ± 0.0	–	0.21 ± 0.01	–	16.93 ± 1.12 [26]	–	–
SITE [27]	–	0.66 ± 0.11	–	0.22 ± 0.01	–	–	–	–
GANITE [25]	–	2.4 ± 0.4	–	0.14 ± 0.01	–	18.2 ± 1.66 [26]	–	–
CEVAE [22]	–	–	0.03 ± 0.01	0.26 ± 0.0	–	–	0.03 ± 0.00	**0.82** ± **0.0**
Linear	0.29 ± 0.02	2.33 ± 0.11	0.07 ± 0.00	0.24 ± 0.00	0.24 ± 0.03	6.62 ± 0.27	0.04 ± 0.00	0.73 ± 0.00
STARNet	0.13 ± 0.01 $(N = 1)$	0.36 ± 0.01 $(N = 1)$	0.07 ± 0.02 $(N = 1)$	0.23 ± 0.02 $(N = 32)$	0.20 ± 0.03 $(N = 16)$	2.06 ± 0.06 $(N = 64)$	0.04 ± 0.01 $(N = 4)$	0.73 ± 0.01 $(N = 1)$
Bagging STARNet	**0.10** ± **0.00** $(N = 1)$	**0.34** ± **0.01** $(N = 2)$	0.03 ± 0.01 $(N = 16)$	0.20 ± 0.01 $(N = 32)$	0.16 ± 0.02 $(N = 64)$	1.70 ± 0.06 $(N = 64)$	0.03 ± 0.00 $(N = 2)$	0.73 ± 0.00 $(N = 1)$
Boosting STARNet	0.12 ± 0.00 $(N = 1)$	0.40 ± 0.01 $(N = 1)$	0.06 ± 0.00 $(N = 2)$	0.15 ± 0.02 $(N = 8)$	**0.15** ± **0.01** $(N = 64)$	1.82 ± 0.06 $(N = 64)$	**0.01** ± **0.00** $(N = 8)$	0.72 ± 0.00 $(N = 64)$
Linear with propensity	0.29 ± 0.02	2.33 ± 0.11	0.06 ± 0.00	0.24 ± 0.00	0.29 ± 0.03	6.57 ± 0.27	0.04 ± 0.00	0.73 ± 0.00
STARNet with propensity	0.18 ± 0.01 $(N = 1)$	0.83 ± 0.05 $(N = 1)$	0.08 ± 0.02 $(N = 16)$	0.23 ± 0.02 $(N = 32)$	0.21 ± 0.03 $(N = 16)$	2.16 ± 0.07 $(N = 64)$	0.03 ± 0.01 $(N = 2)$	0.73 ± 0.01 $(N = 1)$
Bagging STARNet with propensity	0.14 ± 0.01 $(N = 4)$	0.54 ± 0.02 $(N = 4)$	**0.02** ± **0.00** $(N = 16)$	0.20 ± 0.01 $(N = 64)$	0.16 ± 0.02 $(N = 32)$	**1.65** ± **0.06** $(N = 64)$	0.02 ± 0.00 $(N = 2)$	0.73 ± 0.00 $(N = 1)$
Boosting STARNet with propensity	0.14 ± 0.01 $(N = 1)$	0.60 ± 0.04 $(N = 1)$	0.07 ± 0.00 $(N = 2)$	**0.12** ± **0.01** $(N = 64)$	0.16 ± 0.02 $(N = 16)$	1.79 ± 0.07 $(N = 64)$	**0.01** ± **0.00** $(N = 32)$	0.71 ± 0.00 $(N = 64)$
Bagging STARNet with naive validation	0.12 ± 0.00	0.50 ± 0.01	0.07 ± 0.00	0.22 ± 0.01	0.16 ± 0.02	1.70 ± 0.06	0.03 ± 0.00	0.73 ± 0.00
	$(N = 2)$		$(N = 2)$		$(N = 64)$		$(N = 2)$	
Boosting STARNet with naive validation	0.17 ± 0.01	1.17 ± 0.02	0.06 ± 0.00	0.23 ± 0.03	**0.15** ± **0.01**	1.82 ± 0.06	0.05 ± 0.00	0.72 ± 0.00
	$(N = 64)$		$(N = 2)$		$(N = 64)$		$(N = 64)$	
Bagging STARNet with propensity and naive validation	0.14 ± 0.01	0.54 ± 0.02	0.09 ± 0.00	0.30 ± 0.01	0.16 ± 0.02	**1.65** ± **0.06**	0.02 ± 0.00	0.73 ± 0.00
	$(N = 4)$		$(N = 2)$		$(N = 64)$		$(N = 2)$	
Boosting STARNet with propensity and naive validation	0.14 ± 0.00	0.62 ± 0.01	0.09 ± 0.01	0.19 ± 0.06	0.18 ± 0.02	1.79 ± 0.07	0.05 ± 0.00	0.71 ± 0.00
	$(N = 4)$		$(N = 2)$		$(N = 64)$		$(N = 64)$	

TABLE I: State-of-the-art results for each dataset, and results from STARNet and its variants. All values are reported as mean ± standard deviation. Values for methods other than our own are extracted from the referenced sources.

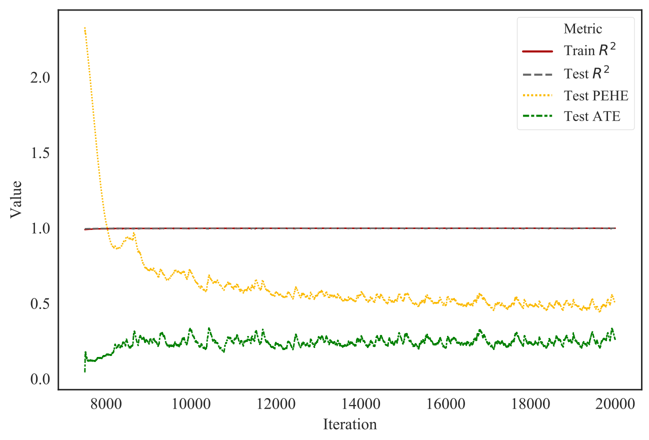


Fig. 4: Progress of ATE$_\varepsilon$, PEHE$_\varepsilon$, and R^2 on the test set for an IHDP realisation as a function of the number of iterations.

almost reached its maximum within the first 8,000 iterations, the error in the PEHE keeps decreasing, with an initial rapid drop and constant improvement afterwards. This marked drop in PEHE$_\varepsilon$ seems similar to a "smooth" version of grokking. The network, following extensive training, reaches a stage where counterfactual performance is achieved.

C. Validation Errors

We have not done hyperparameter tuning in our case, but one way to choose hyperparameters would be to use OOB R^2 estimates for each realisation and evaluate the test set on the model with the highest OOB score. Unfortunately, the number of ensemble networks required is high, which, combined with the very high number of epochs, would result in prohibitively long training times. We recorded the validation scores on the data that the network was not training on for a specific ensemble, which we call "naive validation", as a proxy.

Fig. 5 shows that the validation score changes markedly with the number of neurons, and that all errors as a function of N for each dataset, are closely correlated with their corresponding test scores (see Fig. 3). We are able to reach competitive results when using OOB scores to select the best model (see the bottom part of Table I for the rows that have

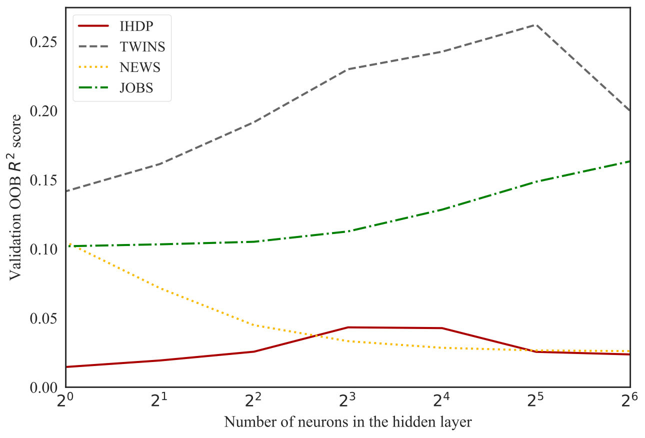


Fig. 5: Average validation OOB R^2 scores on each dataset using bagging STARNet (without propensity scores), as a function of the number of neurons in the hidden layer. We term this method of hyperparameter tuning "naive validation".

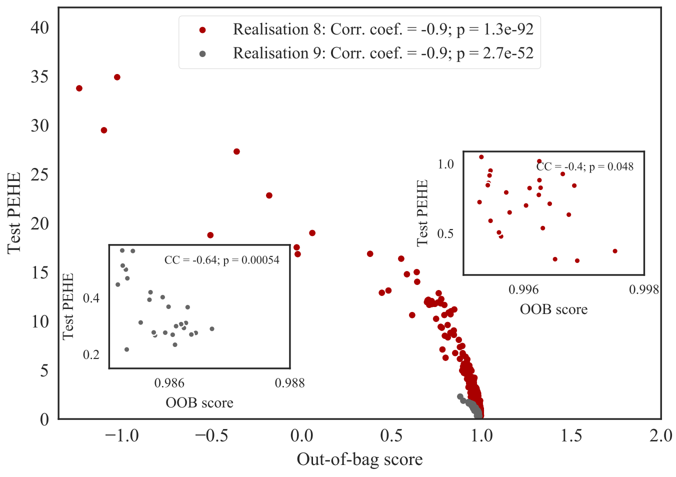


Fig. 6: Error in the PEHE vs. OOB R^2 score for realisations 8 and 9 of the IHDP benchmark, represented in different colours. The insets zoom into the top 10% of results separately for each realisation. Pearson correlation coefficients are also reported for each realisation and for the top 10% of results (in the insets).

"naive validation" on them), but they are clearly worse than the best possible scores.

D. Hyperparameter exploration

So far we have seen (in Sections IV-B and IV-C) that we need to train for as long as we can on datasets that will not overfit, and this seems to uncover the counterfactual effects. However, there is a large number of hyperparameters that we have ignored so far to tune for, and we expect to see gains if we do. To confirm this, we tested for different numbers of hidden layers $(3, 2, 1)$ of $(2, 4, 8, 16, 32, 64)$ neurons of type $(\tanh, \mathrm{elu})$ and L_2 regularisation strengths of $(0.1, 0.01, 0.001)$ as well as adding propensity scores $(True, False)$. We trained 100 neural networks in a bagging setup in order to obtain accurate OOB estimates.

We show the aggregate results in Fig. 6 for two different realisations of the IHDP benchmark, where we have also included the Pearson correlation coefficient r between OOB scores and error in PEHE on the test set for two different realisations. Although there is a very strong (and very significant) correlation when all hyperparameters are taken into account ($r = -0.9$), this is not strictly the case for the top 10% models sorted by their OOB scores (shown in the insets, with $r = -0.4$ and $r = -0.64$ for the two realisations). This indicates that we need strong regressors (i.e., we need to be able to predict irrespective of establishing any counterfactual relationship). In the top 10% architectures, this is not the case anymore, and reversals do happen. That is, (very) high R^2 values do not always correlate with similarly ranked PEHEε scores on the test set. Overall, it seems that using a smooth activation function (tanh) combined with a small network size tends to produce better results, provided the OOB R^2 score is high enough.

V. Discussion and Conclusion

We have shown that simple neural networks can do counterfactual inference, if the right hyperparameter regime is discovered. The first thing to note is that we have not done an extensive study on hyperparameter tuning — we have just showcased that models of high quality exist and uncovered some links between OOB scores and counterfactual performance. There are methods for discovering hyperparameters in counterfactual inference (e.g. as in [30]), and using them to find the correct number of layers/neurons remains work for the future. Considering that in a real-world setting there will be no access to the counterfactual outcome (hence making it hard to compare different hyperparameters), based on our experimental setup and results we postulate that the best model is that which contains as few neurons as possible in the hidden layer, trained with as many iterations as possible, in a bagging setup that allows one to more or less tune the number of neurons via OOB score validation. Note, however, that this is just a hint, and further work is needed in this area to confirm our initial results. In the future, we will focus on extensive hyperparameter tuning. We suspect that once a proper tuning framework is developed, the counterfactual inference problem might collapse into a straightforward representation learning problem, albeit one where "grokking" comes into play.

VI. Acknowledgements

Spyridon Samothrakis acknowledges the support of the ESRC Research Centre on Micro-Social Change (MiSoC) (ES/S012486/1) funded by the UK Economics and Social Research Council. Maria Fasli acknowledges the support of the Business and Local Government Data Research Centre BLG DRC (ES/S007156/1) funded by the Economic and Social Research Council (ESRC).

References

[1] J. Pearl, Causality. Cambridge university press, 2009.

[2] J. Peters, D. Janzing, and B. Schölkopf, Elements of causal inference: foundations and learning algorithms. MIT press, 2017.

[3] M. A. Hernan and J. M. Robins, "Causal inference," 2020.

[4] M. A. Hernán, J. Hsu, and B. Healy, "A second chance to get causal inference right: A classification of data science tasks," Chance, vol. 32, no. 1, pp. 42–49, 2019.

[5] B. Schölkopf, "Causality for machine learning," arXiv preprint arXiv:1911.10500, 2019.

[6] F. D. Johansson, U. Shalit, and D. Sontag, "Learning representations for counterfactual inference," arXiv preprint arXiv:1605.03661, 2016.

[7] A. Power, Y. Burda, H. Edwards, I. Babuschkin, and V. Misra, "Grokking: Generalization beyond overfitting on small algorithmic datasets," arXiv preprint arXiv:2201.02177, 2022.

[8] D. B. Rubin, "Estimating causal effects of treatments in randomized and nonrandomized studies." Journal of educational Psychology, vol. 66, no. 5, p. 688, 1974.

[9] ——, "Causal inference using potential outcomes: Design, modeling, decisions," Journal of the American Statistical Association, vol. 100, no. 469, pp. 322–331, 2005.

[10] S. R. Künzel, J. S. Sekhon, P. J. Bickel, and B. Yu, "Metalearners for estimating heterogeneous treatment effects using machine learning," Proceedings of the National Academy of Sciences, vol. 116, no. 10, pp. 4156–4165, 2019.

[11] M. J. Funk, D. Westreich, C. Wiesen, T. Stürmer, M. A. Brookhart, and M. Davidian, "Doubly robust estimation of causal effects," American journal of epidemiology, vol. 173, no. 7, pp. 761–767, 2011.

[12] P. C. Austin, "The relative ability of different propensity score methods to balance measured covariates between treated and untreated subjects in observational studies," Medical Decision Making, vol. 29, no. 6, pp. 661–677, 2009.

[13] M. C. Elze, J. Gregson, U. Baber, E. Williamson, S. Sartori, R. Mehran, M. Nichols, G. W. Stone, and S. J. Pocock, "Comparison of propensity score methods and covariate adjustment: evaluation in 4 cardiovascular studies," Journal of the American College of Cardiology, vol. 69, no. 3, pp. 345–357, 2017.

[14] S. Vansteelandt and R. M. Daniel, "On regression adjustment for the propensity score," Statistics in medicine, vol. 33, no. 23, pp. 4053–4072, 2014.

[15] U. Shalit, F. Johansson, and D. Sontag, "Estimating individual treatment effect: generalization bounds and algorithms," arXiv preprint arXiv:1606.03976, 2016.

[16] J. Brooks-Gunn, F.-r. Liaw, and P. K. Klebanov, "Effects of early intervention on cognitive function of low birth weight preterm infants," The Journal of pediatrics, vol. 120, no. 3, pp. 350–359, 1992.

[17] J. L. Hill, "Bayesian nonparametric modeling for causal inference," Journal of Computational and Graphical Statistics, vol. 20, no. 1, pp. 217–240, 2011.

[18] J. A. Smith and P. E. Todd, "Does matching overcome lalonde's critique of nonexperimental estimators?" Journal of econometrics, vol. 125, no. 1-2, pp. 305–353, 2005.

[19] R. J. LaLonde, "Evaluating the econometric evaluations of training programs with experimental data," The American economic review, pp. 604–620, 1986.

[20] R. H. Dehejia and S. Wahba, "Propensity score-matching methods for nonexperimental causal studies," Review of Economics and statistics, vol. 84, no. 1, pp. 151–161, 2002.

[21] D. Almond, K. Y. Chay, and D. S. Lee, "The costs of low birth weight," The Quarterly Journal of Economics, vol. 120, no. 3, pp. 1031–1083, 2005.

[22] C. Louizos, U. Shalit, J. M. Mooij, D. Sontag, R. Zemel, and M. Welling, "Causal effect inference with deep latent-variable models," in Advances in Neural Information Processing Systems, 2017, pp. 6446–6456.

[23] T. Fawcett, "An introduction to roc analysis," Pattern recognition letters, vol. 27, no. 8, pp. 861–874, 2006.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., "Scikit-learn: Machine learning in python," Journal of machine learning research, vol. 12, no. Oct, pp. 2825–2830, 2011.

[25] J. Yoon, J. Jordon, and M. van der Schaar, "Ganite: Estimation of individualized treatment effects using generative adversarial nets," 2018.

[26] P. Schwab, L. Linhardt, and W. Karlen, "Perfect match: A simple method for learning representations for counterfactual inference with neural networks," arXiv preprint arXiv:1810.00656, 2018.

[27] L. Yao, S. Li, Y. Li, M. Huai, J. Gao, and A. Zhang, "Representation learning for treatment effect estimation from observational data," in Advances in Neural Information Processing Systems, 2018, pp. 2633–2643.

[28] Z. Liu, O. Kitouni, N. Nolte, E. J. Michaud, M. Tegmark, and M. Williams, "Towards understanding grokking: An effective theory of representation learning," 2022. [Online]. Available: https://arxiv.org/abs/2205.10343

[29] J. Friedman, T. Hastie, and R. Tibshirani, The elements of statistical learning. Springer series in statistics New York, 2001, vol. 1, no. 10.

[30] Y. Saito and S. Yasui, "Counterfactual cross-validation: Effective causal model selection from observational data," arXiv preprint arXiv:1909.05299, 2019.