RNN-Based Counterfactual Prediction

Jason Poulos

University of California, Berkeley

Abstract

This paper proposes an alternative to the synthetic control method (SCM) for estimating the effect of a policy intervention on an outcome over time. Recurrent neural networks (RNNs) are used to predict counterfactual time-series of treated units using only the outcomes of control units as model inputs. The proposed method does not rely on pre-intervention covariates to construct the synthetic control and is consequently less susceptible to p-hacking. RNNs are also capable of handling multiple treated units and can learn nonconvex combinations of control units. In placebo tests, RNNs outperform the SCM in predicting the post-intervention time-series of control units, while yielding a comparable proportion of false positives. The RNN-based approach contributes to a new generation of data-driven machine learning techniques such as matrix completion and the Lasso for generating counterfactual predictions.

Keywords: Counterfactual Prediction; Recurrent Neural Networks; Randomization Inference; Synthetic Controls; Time-Series Cross-Section Data

PhD Candidate, Department of Political Science, 210 Barrows Hall #1950, Berkeley, CA 94720-1950. *Email:* poulos@berkeley.edu. *Telephone:* +1-510-642-6323. I acknowledge support of the National Science Foundation Graduate Research Fellowship (DGE 1106400). This work used the computer resources of Stampede2 at the Texas Advanced Computing Center (TACC) under an Extreme Science and Engineering Discovery Environment (XSEDE) startup allocation (TG-SES180010). The Titan Xp GPU used for this research was donated by the NVIDIA Corporation.

1 Introduction

An important problem in the social sciences is estimating the effect of a binary intervention on an outcome over time. When interventions take place at an aggregate level (e.g., a state), researchers make causal inferences by comparing the post-intervention ("post-period") outcomes of affected ("treated") units against the outcomes of unaffected ("control") units. A common approach to the problem is the synthetic control method (SCM) (Abadie et al., 2010), which predicts the counterfactual outcomes of treated units by finding a convex combination of control units that match the treated units in term of lagged outcomes. Correlations across units are assumed to remain constant stable time.

The SCM has several limitations. First, the convexity restriction of the synthetic control estimator precludes dynamic, nonlinear interactions between multiple control units. Intuitively, one can expect that the treated unit may exhibit nonlinear or negative correlations with the control units. Ferman and Pinto (2016) demonstrate that the convexity restriction implies that the SCM estimator may be biased even if selection into treatment is only correlated with time-invariant unobserved covariates. Second, Ferman and Pinto (2018) demonstrate that the SCM is generally biased if treatment assignment is correlated with unobserved confounders, even when the number of pre-intervention periods grows (Ferman and Pinto, 2018). Moreover, the authors show that while the SCM minimizes imbalance in pre-period outcomes, the likelihood of finding exact balancing weights vanishes as the number of time periods increase, which results in bias.

While the strength of the SCM lies in its simplicity in setup and implementation, several problems arise from the lack of guidance on how to specify the SCM estimator. The specification of the estimator can produce very different results: Ferman et al. (2018) show, for example, how cherry-picking between common SCM specifications can facilitate p-hacking.Kaul et al. (2015) show that the common practice of including lagged versions of the outcome variable as model inputs can render all other covariates irrelevant. Klößner et al. (2017) demonstrates that the common practice of using cross-validation to select importance weights can

yield multiple values and consequently different results.

This paper proposes an alternative to the SCM that is capable of automatically selecting appropriate control units at each time-step, allows for nonconvex combinations of control units, and does not rely on pre-intervention covariates. The method uses recurrent neural networks (RNNs) to predict a counterfactual time-series of treated units using only control unit outcomes as model inputs. RNNs are a class of neural networks that take advantage of the sequential nature of time-series data by sharing model parameters across multiple time-steps (El Hihi and Bengio, 1995). Non-parametric models such as RNNs are useful for prediction problems because we do not have to assume a functional form on the data. In addition, RNNs can learn the most useful nonconvex combination of control unit outcomes at each time-step for generating counterfactual predictions. Relaxing the convexity restriction is useful when the data-generating process underlying the outcome of interest depends nonlinearly on the history of its inputs. RNNs have been shown to outperform various linear models on time-series prediction tasks (Cinar et al., 2017).

The proposed method builds on a new literature that uses machine learning methods for data-driven synthetic controls, such as matrix completion (Athey et al., 2017; Poulos, 2019), or two-stage estimators that reduces data dimensionality via L1-regularized regression (Doudchenko and Imbens, 2016; Carvalho et al., 2018) or matrix factorization (Amjad et al., 2018) prior to regressing the outcomes on the reduced data. These methods are data-driven in the sense that they are capable of finding an appropriate subset of control units for comparison in the absence of domain knowledge or pre-intervention covariates.

RNNs are end-to-end trainable and very flexible to a given sequential prediction problem. For example, they are capable of sharing learned parameters across time-steps and multiple treated units. while the SCM can be generalized to handle multiple treated units (e.g., Dube and Zipperer, 2015; Xu, 2017), the generalized the SCM is not capable of sharing model weights when predicting the outcomes of multiple treated units. Regularization methods such as dropout can easily be incorporated into RNN architectures to prevent overfitting

during the training process, which is problematic when the networks learn an overreliance on a few model inputs. Moreover, an attention mechanism can be included in the model in order to discern the contribution of each model input to the predicted counterfactual.

In the section immediately below, I describe the approach of using RNNs for counterfactual time-series prediction; Section 4 details the procedure for evaluating the models in terms of predictive accuracy and statistical significance; Section 3.4 presents the results of the placebo tests and discusses when the proposed method is expected to outperform the SCM; Section 6 concludes by discussing the contributions of the paper and offering potential avenues for future research.

2 Counterfactual prediction

The proposed method estimates the causal effect of a discrete intervention in observational panel data; i.e., settings in which treatment is not randomly assigned and there exists both pre- and post-period observations of the outcome of interest. Let $\boldsymbol{Y}$ denote a $\mathrm{N} \times \mathrm{T}$ matrix of outcomes for each unit $i = 1, \ldots, \mathrm{N}$ at time $t = 1, \ldots, \mathrm{T}$. $\boldsymbol{Y}$ is incomplete because we observe each element Y_{it} for only the control units and the treated units prior to time of initial treatment exposure, $\mathrm{T}_0 < \mathrm{T}$. Let $\mathcal{O}$ denote the set of (it) values that are observed and $\mathcal{M}$ the set of (it) missing values. Let the values of the $\mathrm{N} \times \mathrm{T}$ complete matrix $\boldsymbol{W}$ be $W_{it} = 1$ if $(it) \in \mathcal{M}$ and $W_{it} = 0$ if $(it) \in \mathcal{O}$. Note that the process that generates W_{it} is referred to the treatment assignment mechanism in the causal inference literature (Imbens and Rubin, 2015) and the missing data mechanism in missing data analysis (Little and Rubin, 2014). The pattern of missing data is assumed to follow from the simultaneous treatment adoption setting, where treated units are exposed to treatment at time T_0 and every subsequent period.

This setup is motivated by the Neyman (1923) potential outcomes framework, where for each it value there exists a pair of potential outcomes, $Y_{it}(1)$ and $Y_{it}(0)$, which represents

the response to treated and control regimes, respectively. The observed outcomes are

$$Y_{it} = \begin{cases} Y_{it}(0) & \text{if } W_{it} = 0 \text{ or } t < \mathrm{T}_0 \\ Y_{it}(1) & \text{if } W_{it} = 1 \text{ and } t \geq \mathrm{T}_0. \end{cases} \tag{1}$$

The problem of counterfactual prediction is that we cannot directly observe the missing potential outcomes and instead wish to impute the missing values in $\boldsymbol{Y}(0)$ for treated units with $W_{it} = 1$. The potential outcomes framework explicitly assumes unconfoundedness. In an observational setting, this assumption requires $(\boldsymbol{Y}(0), \boldsymbol{Y}(1)) \perp\!\!\!\perp \boldsymbol{W} | \boldsymbol{Y}(\mathcal{O})$, where $\boldsymbol{Y}(\mathcal{O})$ is the observed data.

The potential outcomes framework also implicitly assumes treatment is well-defined to ensure that each unit has the same number of potential outcomes (Imbens and Rubin, 2015). It also excludes interference between units, which would undermine the framework by creating more than two potential outcomes per unit, depending on the treatment status of other units (Rubin, 1990).

2.1 Relationship to matrix completion and covariate shift

The proposed approach is similar to the method of matrix completion via nuclear norm minimization (MC-NNM) proposed by Athey et al. (2017) to predict counterfactual outcomes. Matrix completion methods attempt to impute missing entries in a low-rank matrix by solving a convex optimization problem via NNM, even when relatively few values are observed in $\boldsymbol{Y}$ (Candès and Recht, 2009; Candes and Plan, 2010). The estimator recovers a $\mathrm{N} \times \mathrm{T}$ low-rank matrix by minimizing the sum of squared errors via nuclear norm regularized least squares. The estimator reconstructs the matrix by iteratively replacing missing values with those recovered from a singular value decomposition (Mazumder et al., 2010).

Athey et al. (2017) note two drawbacks of MC-NNM. First, the errors may be autocorrelated because the estimator does not account for time-series dependencies in the observed data. The estimator estimate patterns row- and column-wise, but treat the data as perfectly synchronized (Yoon et al., 2018). In contrast, the RNN-based approach described in Section 3 is exploits the temporal component of the data and therefore does not have the problem of autocorrelated errors.

Second, the MC-NNM estimator penalizes the errors for each observed value equally without regard to the fact that the probability of missingness (i.e, the propensity score), increases with t. Athey et al. (2017) suggest weighting the loss function by the propensity score, which is similar to the importance weighting scheme proposed by Cortes et al. (2008) to address the problem of covariate shift, which is a special case of domain adaptation (Huang et al., 2007; Bickel et al., 2009; Cortes et al., 2010).[1]

The covariate shift problem occurs when training and test data are drawn from different distributions. For notational ease, define the training set input-output pair as

$$\left(\boldsymbol{X}^{\text{train}}, \boldsymbol{Y}^{\text{train}}\right) = \left(\boldsymbol{Y}(\boldsymbol{W})^{\{t<\text{T}_0\}}, \boldsymbol{Y}(\boldsymbol{W})^{\{t\geq\text{T}_0\}}\right)$$

for units with $\boldsymbol{W} = 1$ and the test set pair $\left(\boldsymbol{X}^{\text{test}}, \boldsymbol{Y}^{\text{test}}\right)$ for units with $\boldsymbol{W} = 1$. In the proposed approach, the model weights learned on the training set is fit on $\boldsymbol{X}^{\text{test}}$ to predict $\boldsymbol{Y}^{\text{test}}$. The approach therefore assumes similarity between the distributions of $\boldsymbol{X}^{\text{train}}$ and $\boldsymbol{X}^{\text{test}}$. An extension of the RNN-based approach would consider weighting the training loss by the propensity score to reduce any discrepancy between these two distributions.

[1]Schnabel et al. (2016) first connected the matrix completion problem with causal inference in observational settings in the context of recommender systems under confounding. Johansson et al. (2016) formulates the general problem of counterfactual inference as a covariate shift problem. The problem of covariate shift is also related to recent work in transfer learning (Ben-David et al., 2007; Ganin et al., 2015).

2.2 Nonparametric regression

In its most basic form, counterfactual prediction can be represented as a nonparametric regression,

$$\hat{\boldsymbol{Y}}^{\text{train}} = \hat{f}_0\left(\boldsymbol{X}^{\text{train}}\right) + \epsilon^{(t)}, \tag{2}$$

where the noise variables $\epsilon^{(t)}$ are assumed to be i.i.d. standard normal and independent of the observed data. The nonlinear function $\hat{f}_0$ is estimated by minimizing the mean squared error,

$$\text{MSE} = \text{E}\left[\left(\boldsymbol{Y}^{\text{train}} - \hat{\boldsymbol{Y}}^{\text{train}}\right)^2\right]. \tag{3}$$

At test time, the estimated function is used to predict $\hat{\boldsymbol{Y}}^{\text{test}} = \hat{f}_0\left(\boldsymbol{X}^{\text{test}}\right)$. The estimated causal effect of the intervention is then

$$\hat{\boldsymbol{\phi}} = \boldsymbol{Y}^{\text{test}} - \hat{\boldsymbol{Y}}^{\text{test}}. \tag{4}$$

This treatment effect is calculated at every post-period time-step and is thus useful for understanding the temporal evolution of the causal effect.

3 RNNs for counteractual prediction

RNNs (Graves, 2012; Goodfellow et al., 2016) consist of an input $\boldsymbol{X} = \left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n_x)}\right)$, an output $\boldsymbol{Y} = \left(\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(n_y)}\right)$, and a hidden state $\boldsymbol{h}^{(t)}$. In the encoder-decoder network architecture described below, n_x and n_y can vary in length; in the plain vanilla RNN it is assumed $n_x = n_y = T$.

At each t, RNNs input $\boldsymbol{x}^{(t)}$ and pass it to the $\boldsymbol{h}^{(t)}$, which is updated with a function $g^{(t)}$

using the entire history of the input, which is unfolded backwards in time:

$$\begin{aligned} \boldsymbol{h}^{(t)} &= g^{(t)}\left(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t-1)}, \ldots, \boldsymbol{x}^{(1)}\right) \\ &= f_1\left(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}; \theta\right), \end{aligned} \tag{5}$$

where the nonlinear function $f_1(\cdot)$ and parameter θ is shared for all t. Parameter sharing is particularly useful in the current application because it allows for better generalization when the dimension of the training data is relatively small. The updated hidden state (5) is used to generate a sequence of values $\boldsymbol{o}^{(t)}$ in the form of log probabilities corresponding to the output. The loss function computes $\hat{\boldsymbol{y}}^{(t)} = \text{linear}(\boldsymbol{o}^{(t)})$ and calculates the loss. The total loss for the input-output pair is the sum of the losses over all t.

The RNNs are trained to estimate the conditional distribution of $\boldsymbol{y}^{(t)}$ given the past inputs and also the previous output. This is accomplished by offsetting the input-output pairs by one time-step so that the networks receive $\boldsymbol{y}^{(1)}$ as input at $t+1$ to be conditioned on for predicting subsequent outputs. This popular training procedure is known as teacher forcing because it forces the networks to stay close to the ground-truth output $\boldsymbol{y}^{(t)}$ (Lamb et al., 2016). Specifically, the RNNs are trained to maximize the log-likelihood

$$\log \Pr\left(\boldsymbol{y}^{(t)} | \boldsymbol{x}^{(1)} \ldots \boldsymbol{x}^{(t)}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(t-1)}\right). \tag{6}$$

3.1 Encoder-decoder networks

Encoder-decoder networks are the standard for neural machine translation (Cho et al., 2014; Bahdanau et al., 2014; Vinyals et al., 2014) and are also widely used for predictive tasks, including speech recognition (Chorowski et al., 2015) and time-series forecasting (Zhu and Laptev, 2017).

Encoder-decoder networks are trained to maximize (6), where n_x and n_y can differ. The encoder RNN reads in $\boldsymbol{x}^{(t)}$ sequentially and the hidden state of the network updates

according to (5). The hidden state of the encoder is a context vector $\boldsymbol{c}$ that summarizes the input sequence, which is copied over to the decoder RNN. The decoder generates a variable-length output sequence by predicting $\boldsymbol{y}^{(t)}$ given the encoder hidden state and the previous element of the output sequence. Thus, the hidden state of the decoder is updated recursively by

$$\boldsymbol{h}^{(t)} = f_1\left(\boldsymbol{h}^{(t-1)}, \boldsymbol{y}^{(t-1)}, \boldsymbol{c}; \theta\right), \tag{7}$$

and the conditional probability of the next element of the sequence is

$$\Pr(\boldsymbol{y}^{(t)}|\boldsymbol{y}^{(t)}, \ldots, \boldsymbol{y}^{(t-1)}, \boldsymbol{c}) = f_1\left(\boldsymbol{h}^{(t)}, \boldsymbol{y}^{(t-1)}, \boldsymbol{c};\, \theta\right). \tag{8}$$

Effectively, the decoder learns to generate outputs $\boldsymbol{y}^{(t)}$ given the previous outputs, conditioned on the input sequence.

3.2 Recurrent variational autoencoder

While the encoder-decoder architecture is effective for many sequential prediction tasks, the model does not learn a vector representation of the entire input. The variational autoencoder (VAE) (Kingma and Welling, 2013) is a generative model that learns a latent variable model for $\boldsymbol{x}^{(t)}$ such that new sequences $\boldsymbol{x'}^{(t)}$ can be generated by sampling from the latent space q. Similar to encoder-decoder networks, the VAE has an encoder that learns a latent representation of the input sequence and a decoder that maps the representation back to the inputs. The VAE architecture differs from encoder-decoder networks in that the VAE doesn't have a final dense layer that compares the decoder outputs to $\boldsymbol{x'}^{(t)}$ (i.e., it is a "self-supervised" technique). The other difference is that the VAE maps the inputs to a distribution over latent variables.

The recurrent VAE (RVAE) proposed by several researchers (Fabius and van Amersfoort, 2014; Chung et al., 2015; Bowman et al., 2015) for sequence generation consists of an encoder

RNN that maps $\boldsymbol{x}^{(t)}$ to a distribution over parameters of q. The model then randomly samples $\boldsymbol{z}$ from the latent distribution, $q(\boldsymbol{z}|\boldsymbol{x}^{(t)}) = q(\boldsymbol{z}; f_2(\boldsymbol{x}^{(t)}; \theta))$, where $f_2(\cdot)$ takes the form of a log-normal distribution in the empirical applications. Finally, a decoder RNN takes the form of a conditional probability model $\Pr(\boldsymbol{x}^{(t)}|\boldsymbol{z})$. The parameters of the model are learned by maximizing the loss function, which takes the difference between the log-likelihood between the decoder outputs $\boldsymbol{x}'^{(t)}$ and $\boldsymbol{x}^{(t)}$ and the relative entropy between $q(\boldsymbol{z}|\boldsymbol{x}^{(t)})$ and the model prior $\Pr(\boldsymbol{z})$. The latter component of the loss function acts as regularizer by forcing the learned latent distribution to be similar to the model prior.

3.3 Implementation details

The networks are implemented with the `Keras` neural network library (Chollet et al., 2015) in Python on top of a TensorFlow backend. The implementation details are as follows.[2]

Encoder-decoder networks The encoder takes the form of a two-layer Long Short-Term Memory (LSTM) network (Schmidhuber and Hochreiter, 1997), each with 128 hidden units, and the decoder is a single-layer Gated Recurrent Unit (GRU) (Chung et al., 2014) also with 128 hidden units. Each recurrent layer uses a linear activation function with weights initialized using Xavier initialization (Glorot and Bengio, 2010). An attention mechanism in the form of a softmax mask is included before the first hidden layer in order to generate a normalized distribution of the importance of each time-step regarding an input.[3]

RNN weights are learned with stochastic gradient descent on the MSE using `Adam` stochastic optimization (Kingma and Ba, 2014). As a regularization strategy, I apply dropout to the inputs and L2 regularization losses to the network weights. RNNs are trained in batches of size four for 10,000 epochs, which takes about 20 minutes to run on a 12GB NVIDIA

[2]Figures SM-1 and SM-2 illustrates the architectures of encoder-decoder networks and the VAE, respectively.

[3]In contrast to attention mechanisms used in neural machine translation to assist networks in learning the correct alignment between image pixels and target characters (Cho et al., 2014; Poulos and Valle, 2017), the attention mechanism used in this work is not expected to help the model perform better, but to help understand which time-steps and inputs contribute to the prediction.

Titan Xp GPU.

RVAE The encoder takes the form of a single-layer LSTM with 32 hidden units and the decoder is a two-layer LSTM with the number of hidden units equal to 32 and the number of predictors, respectively. The latent space $\boldsymbol{z}$ is implemented as a densely-connected layer with a dimension of 200 units.

3.4 Placebo tests

In this section, I evaluate the accuracy of the RNN-based approach on the following three datasets common to the synthetic control literature, with the actual treated unit removed from each dataset: Abadie and Gardeazabal's (2003) study of the economic impact of terrorism in the Basque Country during the late 1960s (N = 16, T = 43); Abadie et al.'s (2010) study of the effects of a large-scale tobacco control program implemented in California in 1988 (N = 38, T = 31); and Abadie et al.'s (2015) study of the economic impact of the 1990 German reunification on West Germany (N = 16, T = 44). For each trial run, I randomly select half of the control units to be treated and predict their counterfactual outcomes for periods following a randomly selected T_0. I compare the predicted values to the observed values by calculating the root-mean squared error (RMSE).

I benchmark the encoder-decoder networks and RVAE against the following estimators:

(a) DID Regression of $\mathbf{Y}$ on $\mathbf{W}$ and unit and time fixed effects (Athey et al., 2017)

(b) HR-EN Horizontal regression with elastic net regularization, with regularization term λ and shape parameter α selected by cross-validation (Athey et al., 2017)

(c) LSTM Baseline RNN in the form of a single unidirectional LSTM with output space dimensionality equivalent to the number of predictors

(d) MC-NNM Matrix completion via nuclear norm minimization, with the regularization term on the nuclear norm selected by cross-validation (Athey et al., 2017)

(e) SC-ADH Synthetic control method approached via exponentiated gradient descent (Abadie et al., 2010)

(f) VT-EN Vertical regression with elastic net regularization, with λ and α selected by cross-validation (Athey et al., 2017).

Figure 1 reports the average prediction error of the estimators in a simultaneous treatment adoption setting, with the estimates jittered horizontally to reduce overlap. Error bars represent 95% prediction intervals calculated using the standard deviation of the prediction distribution for 20 trial runs.

Across all estimators, the average RMSE decreases and prediction intervals narrow as T_0/T approaches unity because the estimators have more information to generate counterfactual predictions. The MC-NNM estimator generally outperforms all other estimators in terms of average RMSE across different ratios T_0/T. The strong performance of the MC-NNM estimator can be attributed to the fact that it is capable of using additional information in the form of pre-period observations of the treated units, whereas the regression-based estimators rely only on the pre-period observations of control units to predict counterfactuals.

(A) Basque Country terrorism data, $N_t = 8$ (B) California smoking ban data, $N_t = 19$

(C) West German reunification data, $N_t = 8$

Figure 1: Placebo tests under simultaneous treatment adoption: ⊖, DID; △, HR-EN; +, MC-NNM; ×, PCA; ◇, SC-ADH; ▽, SVD; □, VT-EN.

4 Hypothesis testing

In the placebo tests, the counterfactual outcome is known and the estimators can be evaluated in terms of the RMSE between the predicted and actual post-intervention outcomes of placebo treated units. RMSE measures the accuracy of the counterfactual predictions, and consequently the accuracy of the estimated treatment effect. However, this metric does not tell us anything about the statistical significance of estimated treatment effects.

Abadie et al. (2010) propose a randomization inference approach for calculating the exact distribution of placebo effects under the sharp null hypothesis of no effect. Cavallo et al.

(2013) extends the placebo-based testing approach to the case of multiple (placebo) treated units by constructing a distribution of *average* placebo effects under the null hypothesis. Firpo and Possebom (2018) derive the conditions under which the randomization inference approach is valid from a finite sample perspective.[4] Randomization p-values are obtained following these steps:

1. Estimate the observed test static μ^* by calculating the MSE for all J control units, which results in a matrix of dimension $(\mathrm{T}-\mathrm{T}_0)\times \mathrm{J}$. Taking the row-wise mean results in a $\mathrm{T}-\mathrm{T}_0$-length array of observed average placebo treated effects.

2. Calculate every possible average placebo effect μ by randomly sampling without replacement which $\mathrm{J}-1$ control units are assumed to be treated. There are $\mathcal{Q}=\sum_{\mathrm{g}=1}^{\mathrm{J}-1}\binom{\mathrm{J}}{\mathrm{g}}$ possible average placebo effects. The result is a matrix of dimension $(\mathrm{T}-\mathrm{T}_0)\times\mathcal{Q}$.[5]

3. Take a column-wise sum of the number of μ that are greater than or equal to μ^*.

Each element of the $(\mathrm{T}-\mathrm{T}_0)\times J$ matrix of counts obtained from the last step is divided by $\mathcal{Q}$ to estimate an array of exact two-sided p values, $\hat{p}$. Assuming that treatment has a constant additive effect Δ, I construct an interval estimate for Δ by inverting the randomization test. Let δ_Δ be the test statistic calculated by subtracting all possible μ by Δ. I derive a two-sided randomization confidence interval by collecting all values of δ_Δ that yield $\hat{p}$ values greater than or equal to a significance level α. I find the endpoints of the confidence interval by randomly sampling 1,000 values of Δ.

[4]Hahn and Shi (2017) analyze the approach from a repeated sampling perspective.

[5]Note that $\mathcal{Q}$ can be computationally burdensome when there are many control units. I set $\mathcal{Q}=10,000$ in which $\mathrm{J}>16$.

5 Application: Homestead acts and state capacity

5.1 Data and assumptions

In this section, I estimate the causal impacts of homestead acts on state government education spending. I create a state-level measure of state government education spending from the records of 48 state governments during the period of 1783 to 1932 (Sylla et al., 1993) and the records of 16 state governments during the period of 1933 to 1937 (Sylla et al., 1995a,b). Comparable measures for 48 states are drawn from U.S. Census special reports for the years 1902, 1913, 1932, 1942, 1962, 1972, and 1982 (Haines, 2010).

The data pre-processing steps are as follows. The measure is inflation-adjusted according to the U.S. Consumer Price Index (Williamson, 2017) and scaled by the total free population in the decennial census (Haines, 2010). Missing values are imputed separately in the pre- and -post-periods by carrying the last observation forward and remaining missing values are imputed by carrying the next observation backward. The raw outcomes data are log-transformed to alleviate exponential effects. Lastly, I remove states with no variance in the pre-period outcomes, resulting in a complete N $\times$ T matrix of size 33×156.

In this application, public land states are the treated units and state land states — i.e., states that were not crafted from the public domain and were therefore not directly affected by homestead policies — serve as control units. This group includes states of the original 13 colonies, Maine, Tennessee, Texas, Vermont, and West Virginia. The RNN-based approach assumes the distribution of $\boldsymbol{X}^{\text{train}}$ and $\boldsymbol{X}^{\text{test}}$ are similar. The assumption may be violated in the application because state land states had already been well-developed by the time of the passage of the homestead laws. Fig. SM-3 (left panel) shows that the pre-period distributions of education spending for control and treated units are visually similar; however, a two-sided t-test provides evidence to reject the null of equivalence for the difference-in-means between the two distributions ($p < 0.01$). While the no interference assumption cannot directly be tested, it is likely that state land states were indirectly affected by the out-migration of

homesteaders from frontier states.

Aggregating to the state level approximately 1.46 million individual land patent records authorized under the HSA,[6] I determine that the earliest homestead entries occurred in 1869 in about half of the frontier states, about seven years following the enactment of the Homestead Act. Using this information, I set $T_0 = 87$, which leaves $T - T_0 = 69$ post-period observations. The RNN-based approach assumes that treatment adoption is simultaneous across states; however, the date of initial treatment exposure varied as new frontier land opened between the period of 1869 to 1902.

5.2 Estimates

I train a encoder-decoder network to predict the counterfactual time-series of public land states, using only their previous history to generate predictions. Similar to the placebo tests on the SCM datasets, I evaluate the models two ways: first, I monitor the loss over 2,000 training epochs and save the model weights with the lowest error on a validation set consisting of the final 10% of the time-series.[7] Second, I calculate MSE (Eq. **??**) on $J = 18$ state land states that serve as placebo treated units. Encoder-decoder networks outperform the baseline LSTM in terms of minimizing the MSE in placebo tests (Tables SM-1 and SM-2).

Figure 18 plots the observed and counterfactual time-series for each outcome and region. Counterfactual predictions of state government finances in the absence of homestead acts generally tracks the observed time-series until the turn of the century, at which the counterfactual flattens and diverges from the increasing observed time-series. This delay can potentially be explained by the facts that homesteaders were required to make improvements on land for five years before filing a grant and homestead entries did not substantially accumulate until after the 1889 cash-entry restriction (Figures SM-**??** and SM-**??**).

Taking the difference between the observed and predicted time-series (Eq. 4) yields time-

[6]Land patent records provide information on the initial transfer of land titles from the federal government and are made accessible online by the U.S. General Land Office (`https://glorecords.blm.gov`).

[7]The models use both dropout and L2 regularization to control for overfitting on the training set. Figures SM-**??** and SM-16 record the training history of each model.

specific estimates of treatment effects. Figure 19 plots the temporal evolution of treatment effect estimates over the entire post-period and 95% randomization confidence intervals that are constructed by inverting the randomization test described in the previous section.[8] Fifty years after its passage, the estimated impact of the HSA on western state government education spending and revenue is 0.005 log points [-0.16, 0.19], and 0.61 log points [-0.19, 1.53], respectively. The confidence intervals surrounding these time-specific estimates contain zero, which implies that the estimated impacts are not significantly more extreme than the placebo treated effects estimated at the same time-step. The confidence interval on the estimated impact of the HSA on western state government expenditure in 1912, an increase of 0.17 log points [0.004, 0.3], does not contains zero, which implies that the estimated impact is significantly more extreme than the placebo treated effects.

An important characteristic of the estimates plotted in Figure 19 is the progressive widening of confidence intervals. Intuitively, counterfactual predictions become more uncertain as we move farther into the future. In the present application, confidence intervals may become implausibly wide because the post-period extends well into the twentieth century. In order to compare with the DD estimates described in the section below, I average over the entire post-period and find no evidence that the HSA impacted western state government education spending, 0.07 [-0.32, 0.46], expenditure, 0.16 [-0.21, 0.57], or revenue, 0.05 [-0.33, 0.46]. Estimates on the impact of the SHA on state capacity in the South are in the same direction and similar magnitudes.

The attention mechanism embedded within the networks provides clues as to which predictors the networks favor at each time-step during the training process. Figure 3 plots the attention mechanism applied to the test data, where each cell of the heatmap represents the mean attention weight regarding each time-step and predictor across all test samples. The attention distribution suggests that the networks favor heavily winner margins from the 1975 election and also winner margins from the 1988 and 2003 elections when predicting the

[8]Figure SM-20 plots the time-specific estimates of randomization p-values inferred from the exact distribution of average placebo effects under the null hypothesis.

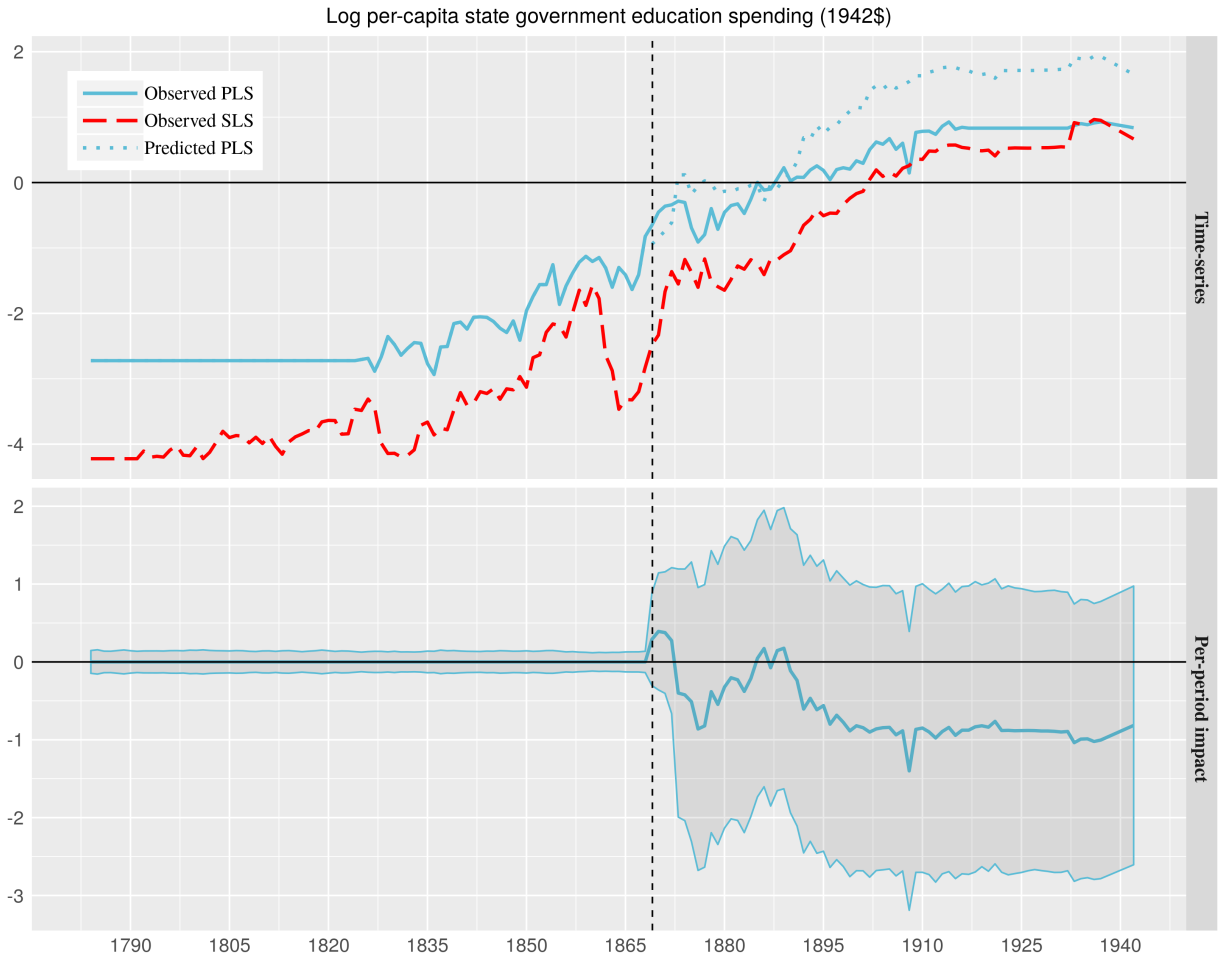

Figure 2: MC-NNM estimates of treatment exposure on state government education spending, 1809 to 1982.

post-period. Attention is generally distributed evenly across predictors, which suggests that the networks rely on many different non-treated cities to construct the counterfactual.

Figure 3: Attention mechanism as a function of predictors (i.e., winner margins by city) and time-steps for encoder-decoder networks. Attention is the normalized (softmax) distribution of the importance of each time-step regarding a predictor.

6 Conclusion

This paper makes a methodological contribution in proposing a novel alternative to the SCM for estimating the effect of a policy intervention on an outcome over time in settings where appropriate control units are unavailable. The SCM is growing in popularity in the social sciences despite its limitations — the most obvious being that the choice of specification can lead to different results, and thus facilitate p-hacking. By inputting only control unit outcomes and not relying on pre-period covariates, the proposed method offers a more principled approach than the SCM.

In placebo tests, RNN-based models outperform the SCM in terms of predictive accuracy while yielding a comparable proportion of false positives. RNNs have advantages over the SCM in that they are structured for sequential data and can learn nonconvex combinations of predictors, which is beneficial when the data-generating process underlying the outcome of interest depends nonlinearly on the history of its inputs. RNNs are also capable of handling multiple treated units and can learn nonconvex combinations of control units, which is useful because the model can share parameters across treated units, and thus generate more precise predictions in settings in which treated units share similar data-generating processes.

The RNN-based approach joins a new generation of data-driven machine learning techniques for generating counterfactual predictions. Machine learning techniques in general have an advantage over the SCM in that they automatically choose appropriate predictors without relying on pretreatment covariates; this capability limits "researcher degrees of free-

dom" that arises from choices on how to specify the model. RNNs have an advantage over alternative machine learning algorithms because they are specifically structured to exploit the sequential nature of time-series data by sharing model parameters across time-steps.

Future research might investigate through simulations how the interaction between RNN complexity (as determined by the number of hidden layers or nodes) and data dimensionality impacts predictive accuracy. Simulations will also allow us to assess the exact impact of data dimensionality, the proportion of treated units, convexity versus non-convexity in the modeled relationship, and the length of the pre-period on the choice between RNNs and the SCM. Future work might formalize a set of assumptions necessary for the approach to be valid and provide further proof of consistency of the estimator under these assumptions.

References

Abadie, A., Diamond, A. and Hainmueller, J. (2010) Synthetic control methods for comparative case studies: Estimating the effect of california's tobacco control program. *Journal of the American Statistical Association*, **105**, 493–505.

— (2015) Comparative politics and the synthetic control method. *American Journal of Political Science*, **59**, 495–510.

Abadie, A. and Gardeazabal, J. (2003) The economic costs of conflict: A case study of the basque country. *The American Economic Review*, **93**, 113–132.

Amjad, M., Shah, D. and Shen, D. (2018) Robust synthetic control. *The Journal of Machine Learning Research*, **19**, 802–852.

Athey, S., Bayati, M., Doudchenko, N., Imbens, G. and Khosravi, K. (2017) Matrix Completion Methods for Causal Panel Data Models. *ArXiv e-prints*.

Bahdanau, D., Cho, K. and Bengio, Y. (2014) Neural Machine Translation by Jointly Learning to Align and Translate. *ArXiv e-prints*.

Ben-David, S., Blitzer, J., Crammer, K. and Pereira, F. (2007) Analysis of representations for domain adaptation. In *Advances in neural information processing systems*, 137–144.

Bickel, S., Brückner, M. and Scheffer, T. (2009) Discriminative learning under covariate shift. *Journal of Machine Learning Research*, **10**, 2137–2155.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R. and Bengio, S. (2015) Generating sentences from a continuous space. *arXiv:1511.06349*.

Candes, E. J. and Plan, Y. (2010) Matrix completion with noise. *Proceedings of the IEEE*, **98**, 925–936.

Candès, E. J. and Recht, B. (2009) Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, **9**, 717.

Carvalho, C., Masini, R. and Medeiros, M. C. (2018) Arco: an artificial counterfactual approach for high-dimensional panel time-series data. *Journal of Econometrics*, **207**, 352–380.

Cavallo, E., Galiani, S., Noy, I. and Pantano, J. (2013) Catastrophic natural disasters and economic growth. *Review of Economics and Statistics*, **95**, 1549–1561.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014) Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *ArXiv e-prints*.

Chollet, F. et al. (2015) Keras. `https://keras.io`.

Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K. and Bengio, Y. (2015) Attention-based models for speech recognition. In *Advances in Neural Information Processing Systems*, 577–585.

Chung, J., Gulcehre, C., Cho, K. and Bengio, Y. (2014) Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *ArXiv e-prints*.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C. and Bengio, Y. (2015) A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems*, 2980–2988.

Cinar, Y. G., Mirisaee, H., Goswami, P., Gaussier, E., Aït-Bachir, A. and Strijov, V. (2017) Position-based content attention for time series forecasting with sequence-to-sequence rnns. In *International Conference on Neural Information Processing*, 533–544. Springer.

Cortes, C., Mansour, Y. and Mohri, M. (2010) Learning bounds for importance weighting. In *Advances in Neural Information Processing Systems*, 442–450.

Cortes, C., Mohri, M., Riley, M. and Rostamizadeh, A. (2008) Sample selection bias correction theory. In *International Conference on Algorithmic Learning Theory*, 38–53. Springer.

Doudchenko, N. and Imbens, G. W. (2016) Balancing, Regression, Difference-In-Differences and Synthetic Control Methods: A Synthesis. *ArXiv e-prints*.

Dube, A. and Zipperer, B. (2015) Pooling multiple case studies using synthetic controls: An application to minimum wage policies.

El Hihi, S. and Bengio, Y. (1995) Hierarchical recurrent neural networks for long-term dependencies. In *Neural Information Processing Systems*, vol. 400, 409.

Fabius, O. and van Amersfoort, J. R. (2014) Variational recurrent auto-encoders. *arXiv:1412.6581*.

Ferman, B. and Pinto, C. (2016) Revisiting the synthetic control estimator. Available at `https://mpra.ub.uni-muenchen.de/81941/`.

— (2018) Synthetic controls with imperfect pretreatment fit. Available at: `https://sites.google.com/site/brunoferman/research`.

Ferman, B., Pinto, C. and Possebom, V. (2018) Cherry picking with synthetic controls. Available at: `https://mpra.ub.uni-muenchen.de/85138/1/MPRA_paper_85138.pdf`.

Firpo, S. and Possebom, V. (2018) Synthetic control method: Inference, sensitivity analysis and confidence sets. *Journal of Causal Inference*, **6**.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M. and Lempitsky, V. (2015) Domain-Adversarial Training of Neural Networks. *arXiv e-prints*, arXiv:1505.07818.

Glorot, X. and Bengio, Y. (2010) Understanding the difficulty of training deep feedforward neural networks. In *Artificial Intelligence and Statistics*, vol. 9, 249–256.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*. Cambridge, MA: MIT press.

Graves, A. (2012) Neural networks. In *Supervised Sequence Labelling with Recurrent Neural Networks*, 15–35. Springer.

Hahn, J. and Shi, R. (2017) Synthetic control and inference. *Econometrics*, **5**, 52.

Haines, M. R. (2010) Historical, Demographic, Economic, and Social Data: The United States, 1790-2002. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor], 2010-05-21. `doi.org/10.3886/ICPSR02896.v3`.

Huang, J., Gretton, A., Borgwardt, K. M., Schölkopf, B. and Smola, A. J. (2007) Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, 601–608.

Imbens, G. W. and Rubin, D. B. (2015) *Causal Inference in Statistics, Social, and Biomedical Sciences*. Cambridge: Cambridge University Press.

Johansson, F., Shalit, U. and Sontag, D. (2016) Learning representations for counterfactual inference. In *International Conference on Machine Learning*, 3020–3029.

Kaul, A., Klößner, S., Pfeifer, G. and Schieler, M. (2015) Synthetic control methods: Never use all pre-intervention outcomes together with covariates. Available at: `http://www.oekonometrie.uni-saarland.de/papers/SCM_Predictors.pdf`.

Kingma, D. P. and Ba, J. (2014) Adam: A Method for Stochastic Optimization. *ArXiv e-prints*.

Kingma, D. P. and Welling, M. (2013) Auto-encoding variational bayes. *arXiv:1312.6114*.

Klößner, S., Kaul, A., Pfeifer, G. and Schieler, M. (2017) Comparative politics and the synthetic control method revisited: A note on Abadie et al. (2015). *Swiss Journal of Economics and Statistics*.

Lamb, A. M., Goyal, A. G. A. P., Zhang, Y., Zhang, S., Courville, A. C. and Bengio, Y. (2016) Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, 4601–4609.

Little, R. J. and Rubin, D. B. (2014) *Statistical Analysis with Missing Data*. Hoboken, NJ: John Wiley & Sons.

Mazumder, R., Hastie, T. and Tibshirani, R. (2010) Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, **11**, 2287–2322.

Neyman, J. (1923) On the application of probability theory to agricultural experiments. *Annals of Agricultural Sciences*, **51**. Reprinted in Splawa-Neyman et al. (1990).

Poulos, J. (2019) State-Building through Public Land Disposal? An Application of Matrix Completion for Counterfactual Prediction. *arXiv e-prints*, arXiv:1903.08028.

Poulos, J. and Valle, R. (2017) Attention networks for image-to-text. *arXiv e-prints*, arXiv:1712.04046.

Rubin, D. B. (1990) Comment: Neyman (1923) and causal inference in experiments and observational studies. *Statistical Science*, **5**, 472–480.

Schmidhuber, J. and Hochreiter, S. (1997) Long short-term memory. *Neural Computation*, **9**, 1735–1780.

Schnabel, T., Swaminathan, A., Singh, A., Chandak, N. and Joachims, T. (2016) Recommendations as treatments: Debiasing learning and evaluation. *arXiv:1602.05352*.

Splawa-Neyman, J., Dabrowska, D., Speed, T. et al. (1990) On the application of probability theory to agricultural experiments. essay on principles. section 9. *Statistical Science*, **5**, 465–472.

Sylla, R. E., Legler, J. B. and Wallis, J. (1993) Sources and Uses of Funds in State and Local Governments, 1790-1915: [United States]. Ann Arbor, MI: Inter-university Consortium for Political and Social Research, 2017-05-21. `doi.org/10.3886/ICPSR06304.v1`.

— (1995a) State and Local Government [United States]: Sources and Uses of Funds, Census Statistics, Twentieth Century [Through 1982]. Ann Arbor, MI: Inter-university Consortium for Political and Social Research, 2017-05-21. `doi.org/10.3886/ICPSR06304.v1`.

— (1995b) State and Local Government [United States]: Sources and Uses of Funds, State Financial Statistics, 1933-1937. Ann Arbor, MI: Inter-university Consortium for Political and Social Research, 2017-05-21. http://doi.org/10.3886/ICPSR06306.v1.

Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I. and Hinton, G. (2014) Grammar as a Foreign Language. *ArXiv e-prints*.

Williamson, S. H. (2017) Seven ways to compute the relative value of a us dollar amount, 1774 to present. `MeasuringWorth.com`. [Online; accessed 01-October-2017].

Xu, Y. (2017) Generalized synthetic control method: Causal inference with interactive fixed effects models. *Political Analysis*, **25**, 57–76.

Yoon, J., Zame, W. R. and van der Schaar, M. (2018) Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Transactions on Biomedical Engineering*.

Zhu, L. and Laptev, N. (2017) Deep and Confident Prediction for Time Series at Uber. *ArXiv e-prints*.