Evaluating Uses of Deep Learning Methods for Causal Inference

ALBERT WHATA¹, CHARLES CHIMEDZA²

¹School of Natural and Applied Sciences, Sol Plaatje University, Kimberley 8301, South Africa (e-mail: albert.whata@spu.ac.za)
²School of Statistics and Actuarial Science, University of the Witwatersrand, Johannesburg 2050, South Africa (e-mail: Charles.Chimedza@wits.ac.za)

Corresponding author: Albert Whata (e-mail: albert.whata@spu.ac.za).

ABSTRACT Logistic regression is a popular method that is used for estimating causal effects in observational studies using propensity scores. We examine the use of deep learning models such as the deep neural network (DNN), PropensityNet (PN), convolutional neural network (CNN), and convolutional neural network-long short-term memory network (CNN-LSTM)) to estimate propensity scores and evaluate causal inference. Deep learning models, unlike logistic regression, do not depend on assumptions regarding (i) how variables are selected, (ii) specification of the correct functional form, (iii) statistical distributions of the variables, and (iv) interactions are specified. If these assumptions are not met when using logistic regression, one may obtain biased estimates of treatment effects due to not achieving covariate balance. We conducted studies using simulated data with different sample sizes (N = 500, N = 1000, N = 2000), 15 covariates, a continuous outcome and a binary exposure. These data were used in seven scenarios that were different in the degree of nonlinearity and non-additivity associations between the exposure and covariates. The estimation of propensity scores was considered as a classification task, and performance metrics that included the classification accuracy, the receiver operating characteristic curve area under the curve (AUCROC), the covariate balance, the standard error, the absolute bias, and the 95% confidence interval coverage were evaluated for each model. Overall, CNN and CNN-LSTM achieved good results for covariate balance, classification accuracy, AUCROC, and Cohen’s Kappa. Logistic regression provided substantially better bias reduction, but it had subpar performance based on classification accuracy, AUCROC, Cohen’s Kappa, and 95% confidence interval coverage. The results suggest that deep learning methods, especially CNN, may be useful for estimating propensity scores that are used to estimate causal effects.

INDEX TERMS Causal inference; deep learning; deep neural networks; logistic regression; propensity scores; covariate balance

NOMENCLATURE

| Term          | Definition                                      |
|---------------|-------------------------------------------------|
| Acc           | Accuracy                                        |
| ASAMD         | Average standardised absolute mean difference   |
| AUC ROC       | Area under curve receiver operating characteristic |
| ATE           | Average treatment effect                        |
| CNN-LSTM      | Convolutional neural network-long short-term memory |
| CNN           | Convolutional neural network                     |
| DNN           | Deep neural network                             |
| LR            | Logistic regression                             |
| MCC           | Mathew’s correlation coefficient                |
| NIR           | No Information Rate                             |
| PN            | PropensityNET                                   |

I. INTRODUCTION Typically, observational studies focus on estimating the causal effect that a treatment has on the outcome. The ‘gold’ standard to infer the causal effect of treatment is through randomised clinical trials (RCT) [1]. This is because with RCTs, a researcher is able to randomise the treatment assignment mechanism, ensuring that the treatment and control groups are comparable [1]. By comparing the average values of the outcome between the control and treatment groups, a researcher can obtain unbiased estimates of the treatment effect [1, 2]. Due to ethical considerations, random assignment may not always be done. Reference [3] states that assigning individuals randomly to the control condition or the treatment condition may be unethical. For example, individuals assigned to the control group may not benefit from an important resource (e.g., receiving antiretroviral drugs that save lives) compared to those in the treatment group who receive the important resource. Due to the imbalance present in the confounders, which affects both the treatment and the
outcome, evaluating causal inference in observational studies becomes challenging. According to [4] then it becomes necessary to use techniques such as weighting or matching to achieve balance in the observed covariates between control and the treated and the groups.

The propensity scores introduced by [1] provide a useful way to assign individuals to different treatment conditions when random assignment fails due to the presence of ethical constraints. The propensity score summarises a multidimensional set of confounders into a one-dimensional summary. This is a key property of propensity scores, which means that if there is balance in the propensity score distribution between the control and treatment groups, then the observed confounders are balanced in expectation between the control and treatment groups. This simplifies the process of adjusting the multivariate set of observed characteristics by simply adjusting the one-dimensional propensity scores [2]. Generally, propensity scores are estimated using logistic regression (LR). Using LR to estimate propensity scores and achieve covariate balance usually involves an iterative process of adding interaction terms and non-linear transformations of the explanatory variables until an acceptable covariate balance is obtained [2]. However, this process of achieving covariate balance is not only time consuming, but it does not guarantee that covariate balance may be reached or improved [2].

Although propensity score methods have been widely used, there are still some notable and specific problems that need to be addressed. First, the use of deep learning methods to estimate propensity scores remains open. Secondly, the use of the propensity scores obtained from deep learning models to assess covariate balance is still very open. With the advent of ‘Big data’ and increased computing capabilities, new possibilities have opened up, where more deep learning methods are needed to estimate propensity scores and assess covariate balance. For example, [2, 5, 6] have proposed machine learning algorithms for estimating propensity scores and assessing causal inference.

This paper is structured as follows. In Section II, we give a theoretical background for estimating the average effects of treatment using propensity scores. In Section III, we outline the research methods, i.e., the data generation mechanisms using Monte Carlo simulations, a description of the proposed algorithms, a description of the experiments to be carried out, and the performance evaluation metrics. In Section IV, we present the results and discussion. Finally, Section V concludes the paper by summarising the findings and making recommendations for further work.

II. THEORETICAL BACKGROUND

We estimate average treatment effects using propensity scores by considering a set-up where there are $N$ units indexed by $i = 1, \ldots, N$ and $W_i \in \{0, 1\}$, a binary indicator for treatments where $W_i = 0$ indicates that unit $i$ received the control treatment and $W_i = 1$ indicates that unit $i$ received the treatment. Furthermore, if we let $X_i$ be an $L$-component vector of features, covariates or pretreatment variables that are known to not be affected by treatment, we can formally define the propensity score as $p(x) = Pr(W_i | X_i = x)$ [7]. Thus, the propensity score is the conditional probability of assignment to a certain treatment given a vector of observed covariates, features, or pretreatment variables [6]. Propensity scoring is a statistical technique that is very useful in evaluating treatment effects, especially when using quasi-experimental or observational data [8]. However, two vital assumptions related to causality must be considered before using propensity scores, and these are the assumption of Ignorable Treatment Assignment [4] and the endogeneity assumption [9].

The counterfactual approach depends on the assumption ("unconfoundedness") stated below [10]:

$$W_i \perp (Y_i(0), Y_i(1)) | X_i$$  \hspace{1cm} (1)

The assumption indicates that the outcomes $(Y_i(1), Y_i(0))$ are independent of $W_i$ given the covariates $X_i$ [3]. This assumption is also stated as the Ignorable Treatment Assignment Assumption [4]. Confounding bias is usually controlled by using propensity scores. Propensity scores achieve this by estimating the probability of $W_i$ given the covariates $X_i$. The Rubin Causal Model or the potential outcomes framework [11] depends on this assumption. This assumption holds in a randomised experiment without the need to condition on covariates. However, [7] state that the assumption can be justified in observational studies if the researcher can observe all variables that affect the assignment of the unit to a treatment. Reference [12] states that it is important to ensure that the propensity score is strictly between 0 and 1. This requirement is known as the positivity assumption. Estimates of treatment effects may be biased when the positivity assumption [13] is not met.

Reference [2] states that if Equation 1 holds conditional on the set of covariates $(X_i)$, then it should also hold conditional on the propensity score ($p(x)$). This means that if the distribution of the propensity scores is balanced between the control and treatment groups, then the distribution of the observed covariates will also be balanced in expectation between the groups. Reference [14] refers to this as the balancing property of the propensity score. Subsequently, the one-dimensional propensity score can be used instead of the multivariate set of observed variables, $X_i$’s, to achieve covariate balance. This paper focuses on the inverse propensity of treatment weighting to adjust for confounding. There are basically two methods that can be considered to implement covariate balance, namely: propensity score matching (PSM) [15] and inverse probability of treatment weighting (IPTW) [16, 17]. PSM involves finding units with the same propensity scores in the control and treated groups, then forming a matched data set of the original data [3]. If the assumptions of
unconfoundedness and positivity hold [18], then the average effect of treatment is estimated by comparing the control and treatment groups in the matched data set. In addition, IPTW employs propensity scores to weight the observations to achieve covariate balance. When calculating ATE, units in the treatment group are assigned a weight equal to $1/p(x)$. On the other hand, a weight of $1/(1-p(x))$ is assigned to the units in the control group [2]. Attaching these weights to the treatment and control groups ensures that the covariate distributions of the treatment and control groups are comparable. Therefore, estimates of unbiased treatment effects under unconfoundedness [4] are obtained from weighted differences in the average outcomes of the treated and control observations. The IPTW will be used to evaluate the covariate balance because according to [2], the IPTW gives a lower bias compared to PSM.

A. PROBLEM STATEMENT

Propensity scores are generally estimated using logistic regression. Estimation of propensity scores using logistic regression requires assumptions regarding (i) how variables are selected, (ii) specification of the correct functional form, (iii) statistical distributions of the variables, and (iv) interactions are specified [5]. If the assumptions are not met, one may obtain biased estimates of treatment effects due to the failure to achieve covariate balance. Propensity scores are primarily used to achieve a covariate balance between the treatment group and the control group to obtain valid and unbiased estimates of the treatment effect. According to [14], propensity scores are used to adjust for observed confounding through matching, subclassification, weighting, regression or their combinations. Main effects LR propensity score models have generally been found to provide acceptable covariate balance [5]. However, as models become more complex with interactions and non-linear terms, LR propensity score models have produced large biases when estimating average treatment effects [5]. Machine learning algorithms have generally been employed to perform classification and prediction. Classification tasks use an input data set $\mathcal{D}$ of size $N$ and the corresponding target classes: $\mathcal{D} = \{(x(i), y(i))\}_{i=1}^{N}$, where $X \subset \mathbb{R}$ represents the feature space and $Y = \{1, ..., K\}$ is the label space [19]. We define the classification problem as one of mathematical optimisation. The loss or objective function can be expressed as the cross-entropy between $Y$ and $\hat{Y}$. The cross-entropy loss function is almost the only choice for classification tasks in practice. This loss function, $L(Y, \hat{Y}) \geq 0$, estimates the extent to which the true value of a model $\hat{Y}$ differs from the predicted value $Y$. Ideally, we are given $\mathcal{D}$, where each $(x(i), y(i)) \in (X \times Y)$ and a classifier is therefore a function that maps the input feature space to the label space $f: X \rightarrow R^K$. Reference [2] states that machine learning algorithms are typically designed to minimise misclassification rates and not to estimate class-membership probabilities. However, the classification task can be used to estimate the probabilities of class-membership. For example, [5, 6, 20, 21] used machine learning algorithms to estimate class membership probabilities and found that machine learning algorithms work rather well in this regard and can be used successfully to estimate propensity scores using class membership probabilities. Estimating class membership probabilities using deep neural networks has not been widely used. Therefore, the literature on the use of deep learning methods to estimate propensity scores using class-membership probabilities is still limited. It is vital to investigate whether deep neural networks can reduce or eliminate the reliance on LR assumptions that include functional form, variable selection, distribution of variables, and specification of interactions. In addition, statistical machine learning techniques, such as deep neural networks, are needed to estimate propensity scores and evaluate whether these deep neural networks perform better than LR in the estimation of propensity scores and bias reduction when estimating average treatment effects.

B. RELATED WORK

Researchers have made several efforts to use the power of machine learning techniques for causal inference problems [6]. References [2, 22] have compared machine learning algorithms to model propensity scores. Reference [23] reported on the full potential of machine learning to estimate the average effects of treatment with propensity score methods and found that machine learning methods can be helpful in high-dimensional data sets (that is, a large number of covariates and observations). Reference [24] proposed matching methods based on the random forest to obtain covariate balance between the control and treatment groups for arge observational study data. The authors noted that their approach provided better estimates of the effect of treatment. Reference [6] concluded in their paper that although the assumptions of logistic regression are well understood, those assumptions are often ignored. They noted that boosting (meta-classifiers) [25] and, to a lesser extent, decision trees (particularly CART) [5], appear to be the most important in propensity score analysis, but extensive simulation studies are needed to establish their utility in practice. Reference [26] constructed a normalized empirical probability density distribution (NEPDF) matrix and trained a convolutional neural network (CNN) on the NEPDF matrix for causality predictions. The authors demonstrated that the use of the NEPDF matrix allowed CNN to work very well on image classification problems for the task of causal inference. By using experiments on simulated and real data, their method generally worked well on a diverse set of input data types.

Reference [27] proposed an approach to adapt neural networks to process incomplete data, and found that neural networks give results comparable to methods that require complete data in training. Reference [28] states that there has been limited adoption of deep learning algorithms in the social sciences due to the lack of sufficient data. Reference [29] estimated propensity scores through simulation studies using a deep neural network referred to in their research as
PropensityNet instead of traditional logistic regression and verified the superior performance of their proposed PropensityNet over logistic regression in estimating propensity scores. This paper extends the work of [29] by developing a deep neural network (DNN) that aims to improve PropensityNet performance. The aim is to determine whether deep learning methods; DNN, PropensityNet, convolutional neural network (CNN), and convolutional neural network-long short-term memory (CNN-LSTM) can be used to estimate propensity scores. Furthermore, the article seeks to assess whether deep learning methods are better at reducing bias in estimated average treatment effects compared to logistic regression. Specifically, the paper makes the following contributions.

(a) Estimate propensity scores and assess covariate balance for logistic regression (LR), deep neural network (DNN), PropensityNet (PN), convolutional neural network (CNN), and the convolutional neural network-long short-term memory (CNN-LSTM) algorithm.

(b) Compare the performance of deep learning methods and logistic regression in estimating the average treatment effects using simulation techniques.

(c) Assess the performance of the deep learning methods when they are applied to a real-world data set.

III. RESEARCH METHOD

A. DATA GENERATION USING MONTE-CARLO SIMULATIONS

To assess the performance of deep learning models and logistic regression, we performed a series of Monte Carlo simulation experiments that follow the structure of [30] adapted from [2, 5, 20, 22]. Samples of sizes N = 500, N = 1000 and N = 2000, a binary treatment variable. The general LR model is expressed as follows:

\[ Pr[Y_i|W_i, X_i] = \frac{1 + exp(-\alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_4 + \alpha_5 X_5 + \alpha_6 X_6 + \alpha_7 X_7 + \alpha_8 X_9 + \alpha_9 X_{10})}{1 + exp(-\gamma_1 W_i)} \]

Random numbers between 0 and 1 were generated from a uniform distribution using the R software, setting \( Y_i = 1 \) if the randomly generated number was less than \( Pr[Y_i|W_i, X_i] \) and 0 otherwise. The binary outcome variable for each scenario was used in training the deep learning models (DNN, PN, CNN, and CNN-LSTM) and consequently predicted propensity scores for each of these models. Logistic regression was used to generate a continuous outcome variable \( Y_i \) (for each scenario A-G), as a function of \( W_i \) and \( X_i \) and setting the effect of treatment \( W_i \) to be constant with coefficient \( \gamma_i = -0.4 \) as proposed by [5, 20].

\[ Y_i = -\alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_4 + \alpha_5 X_8 + \alpha_6 X_9 + \alpha_7 X_{10} + \gamma_1 W_i \]

Weighted linear regressions of \( Y_i \) as a function of \( W_i \) and \( X_i \) were performed for Scenarios A-G using \( 1/p(x) \) and \( 1/(1-p(x)) \) as weights to estimate the effect of treatment for each scenario. We use the same parameter values \( \alpha_i \) through \( \alpha_7 \) as were used by [5, 20] in Equations 2 and 3.

B. LOGISTIC REGRESSION

Logistic regression (LR) is a common and useful statistical technique that is used to estimate propensity scores [3, 32]. Reference [6] report that other techniques include discriminant analysis [33], general boosted models [20], classification trees [34], and neural networks [35] just to mention a few. They point out that several propensity score analyses use LR to estimate the scores. In its basic form, LR is a statistical model that uses a logistic function to model a binary dependent variable. The general LR model is expressed as follows:

\[ \ln \left( \frac{Pr(W_i = 1|X_i)}{1 - Pr(W_i = 1|X_i)} \right) = x_i^T \beta \]

where \( x_i \) is a vector of the continuous and dummy variables described in Section III-A, and \( \beta \) is the vector of parameters. According to [7], the propensity score is stated as \( p(x) = \)
$Pr(W_i = 1|X_i)$. In this paper, LR is used to estimate the propensity score, $p(x)$.

Logistic regression is mathematically constrained to produce probabilities between $[0, 1]$ and therefore is attractive for probability prediction [37]. Logistic regression can be easily implemented in a wide variety of statistical software such as R, SPSS, STATA, and SAS. There are several shortcomings that can result from estimating propensity scores using logistic regression. Reference [24] states that logistic regression is prone to misspecification errors that result in imprecise estimates of the propensity score. Missing data presents problems when estimating propensity scores using logistic regression, and these missing data must be dealt with beforehand. Covariates with a large proportion of observations with missing data become unusable when logistic regression is implemented. According to [6], the performance of logistic regression is below par compared to other methods that estimate propensity scores, such as tree ensembles or other machine learning algorithms.

### C. USING DEEP LEARNING METHODS FOR CLASSIFICATION

1) The Cross-Entropy Loss Function

The cross-entropy loss function, $L(Y, f(X))$, is almost the only choice for classification tasks in practice. As discussed in Section II-A, a classifier is represented by the mapping $f: X \rightarrow R^K$.

For multiclassification $L(Y, f(X))$ is defined as:

$$L(Y, f(X)) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} y_{ij} \log(f_{j\theta}|x_i) \quad (5)$$

where $\theta$ is the set of parameters of the classifier, $y_{ij}$ corresponds to the $j$th element of the one-hot encoded label of the $x_i$, $y_i \in \{0, 1\}^K$ such that $1^T y_i = 1 \forall i$ and $f_j$ denotes the $j$th element of $f$. Note that $\sum_{j=1}^{K} f_j(x_{ij}; \theta) = 1$, and $f_j(x_{ij}; \theta) \geq 0, \forall j, i, \theta$, are the outputs obtained using softmax [38].

Equation 5 gives the average differences between the predicted outputs (probabilities) and the target probabilities. Reference [39] indicates that the above can be described as a classification model $\mathcal{N}$ with an architecture $A$, and a vector of parameters $\Theta$ that express the output as a function of the input as follows:

$$\hat{Y} = \mathcal{N}(X; A, \Theta) \quad (6)$$

$A$ will represent the design choices of a deep learning model and $\Theta$ are the parameters that will be tuned during training. Design choices $A$ usually include (i) how the layers are organised, (ii) the activation function, which could be a rectified linear unit $ReLU$ [40], hyperbolic $tanh$, logistic function, (iii) the number of layers of the deep learning model and (iv) the nodes of a layer, $n_l$, $l = 1, 2, 3, ..., L$. For example, for a deep neural network (DNN), the parameter $\Theta$ will include the weights $W_l$ and the biases $b_l$ of the layers, $l = 1, 2, 3, ..., L$.

### D. DEEP NEURAL NETWORK (DNN)

A DNN model is created based on algorithms called artificial neural networks (ANNs) that are structured as stacks of layers on top of each other. It can employ supervised and unsupervised learning [39]. DNN models use weights that are contained in hidden layers. These weights are adjusted during training as they take in and process input. The purpose of adjusting the weights is to find patterns that give better predictions. A DNN self-learns, and the researcher is not required to specify in advance any patterns to consider. Deep learning methods are based on a branch of machine learning called representation learning (feature learning) [41]. These methods perform automatic feature selection compared to machine learning algorithms that require feature selection by the researcher before they are used.

#### TABLE 1. Architecture of DNN

| Type of Layer     | Output shape | Number of Parameters |
|-------------------|--------------|----------------------|
| Dense             | (None, 64)   | 768                  |
| Dropout           | (None, 64)   | 0                    |
| Dense             | (None, 32)   | 2080                 |
| Dropout           | (None, 32)   | 0                    |
| Dense             | (None, 16)   | 528                  |
| Dropout           | (None, 16)   | 0                    |
| Batch Normalization | (None, 16) | 64                   |
| Dense             | (None, 1)   | 17                   |

The DNN architecture shown in Table 1 consists of four dense hidden layers, that is, fully connected layers. Reference [42] states that deep learning models are built by using compatible layers that enable useful data-transformations. This means that every layer in a deep learning model will only accept input tensors of a certain shape and will return output tensors of a certain shape. In addition, [42] indicate that there is no need to worry about compatibility of interconnected layers because they are built to match the shape of the incoming layer. The output shape of a layer refers to the dimensions of a tensor that is returned by that layer. For example, Table 1 shows that the first hidden layer of DNN will return a tensor of dimension (None, 64). This first hidden layer has an output shape equal to (None, 64) with 64 neurons/units. The second layer automatically infers as its input shape the output shape from the first layer. A None dimension is used to allow for a variable batch size, that is, it is a dynamic dimension of a batch (mini batch) that allows one to set any batch size to the deep neural network. None is the first dimension that does not need to be fixed at this stage, unless in very specific cases (for instance, when working with stateful=True LSTM layers). The batch size is then defined in the fit or predict phase of the models.

The dropout layers after the dense layers are used as a powerful regularisation technique [43] to prevent the models...
from overfitting. The architecture uses batch normalisation to stabilise the learning process and significantly reduce the number of training epochs for DNN.

The DNN learning process with the architecture shown in Table 1 involves two important steps: the first step is the forward propagation phase of the training data, which takes in the raw data from the input layer. The second step involves the back-propagation of the error signal [44]. Neurons in the hidden layers are used to process the data which is passed on to the output layers to generate the output data. These output data are passed to the next layer by non-linear functions [29]. These non-linear functions are referred to as activation functions. Examples of activation functions include the logistic function, the hyperbolic tangent \( \tanh \), and the rectified linear unit \( \text{ReLU} \). Their main purpose is to convert an input signal from an input node in a DNN to an output signal. In this work, we use the \( \text{ReLU} \) activation function, as it substantially reduces the computational cost of training and guarantees faster computation and convergence [45]. \( \text{ReLU} \) offers better performance and generalization in deep learning compared to the \( \text{sigmoid} \) and \( \tanh \) activation functions. For a detailed discussion and comparison of the different activation functions, see [45]. Reference [44] provides a detailed procedure for how these two steps work.

For multiclass models, DNN uses a \textit{softmax layer} [39] as its last layer to retain the probabilities of each class. The \textit{softmax layer} that is used for classification into \( K \) classes is defined as follows [39, 46]:

\[
f(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad \forall j \in \{1, \ldots, K\}
\]

The softmax function, \( f(x_i) \), produces an output between 0 and 1, with the sum of the probabilities equal to 1. Since we are performing binary classification, we will use a \textit{sigmoid} function as the last layer to give probabilities between 0 and 1. These probabilities are a measure of the propensity score for each new/test unit.

**E. PROPENSITYNET**

PropensityNet (PN) is a deep neural network that can be used to estimate propensity scores that was proposed by [29]. PN is similar to DNN and it consists of five dense layers, that is, fully connected, as shown in Table 2. PropensityNet uses \textit{Adam} [47] as an optimiser and the binary cross-entropy as an error metric, as it is solves a binary classification problem. A \textit{sigmoid} function is used as the last layer to give probabilities between 0 and 1. These probabilities are a measure of the propensity score for each new/test unit. Keras with Tensorflow backend in R [48] was used to build PropensityNet.

**F. CONVOLUTIONAL NEURAL NETWORKS (CNN)**

Table 3 shows the architecture of the CNN model used in this paper.

CNN is a deep learning algorithm that can produce cutting-edge binary or multiclassification results. In contrast to machine learning algorithms that require a user to choose the best features, a CNN model can automatically extract important features from the input data set [49]. The CNN model consists of a nonlinearity layer (convolutional layer) followed by a max-pooling layer, and a fully connected layer [50]. As shown in Table 3, CNN’s architecture shows that the output shape of the first hidden layer is a \( 15 \times 64 \) matrix. This means that there are 64 filters, and each filter will contain 15 weights. The output of the first layer of CNN is fed into the second layer which again contains 64 filters. Therefore, using the same logic as used in the first layer, the output matrix of the second layer will be a matrix of size \( 15 \times 64 \). The architecture in Table 3 uses a max pooling layer. This layer prevents overfitting by reducing the complexity of the output. Table 3 also shows that there is another 1D CNN layer with a matrix of size \( 15 \times 32 \) that is used to learn higher-level features. A dropout layer is employed to prevent overfitting and increase accuracy on the test data. The architecture includes a flatten layer that is used to convert the data into a 1-dimensional array that is then fed to the last layer. The last layer is a dense (fully connected layer) that will output probabilities that estimate the propensity scores as described in Section III-H.

CNNs have produced excellent results when they were used for natural language processing (NLP) [51], computer vision [52], spam detection [53], text classification [54], topic categorisation [55], and image classification [56]. In this

---

**TABLE 2. Architecture of PN**

| Type of Layer | Output shape  | Number of Parameters |
|---------------|---------------|----------------------|
| Dense         | (None,32)     | 512                  |
| Dropout       | (None,32)     | 0                    |
| Dense         | (None,32)     | 1056                 |
| Dropout       | (None,32)     | 0                    |
| Dense         | (None,32)     | 1056                 |
| Dropout       | (None,32)     | 0                    |
| Dense         | (None,32)     | 1056                 |
| Dense         | (None,1)      | 33                   |

**TABLE 3. Architecture of CNN**

| Type of Layer | Output shape  | Number of Parameters |
|---------------|---------------|----------------------|
| Conv1D        | (None, 15, 64)| 128                  |
| Maxpooling1D  | (None, 15, 64)| 0                    |
| Conv1D        | (None, 15, 64)| 4160                 |
| Maxpooling1D  | (None, 15, 64)| 0                    |
| Conv1D        | (None, 15, 32)| 2080                 |
| Maxpooling1D  | (None, 15, 32)| 0                    |
| Dropout       | (None, 15, 32)| 0                    |
| Flatten       | (None, 480)   | 0                    |
| Dense         | (None, 128)   | 61568                |
| Dropout       | (None, 128)   | 0                    |
| Dense         | (None, 1)     | 129                  |
paper, we will use the CNN to perform binary classification on non-image or non-sequential or non-text data. The sigmoid function is used as the last layer to give probabilities (propensity scores) between 0 and 1.

G. CONVOLUTIONAL NEURAL NETWORKS-LONG SHORT-TERM MEMORY NETWORK (CNN-LSTM)

Typically, long short-term memory networks (LSTMs) [57] have been used to learn long-term dependencies through recurrently connected memory blocks (subnets). They are an example of recurrent neural networks (RNNs) [57]. RNNs are described in detail in [58]. In this paper, we use a hybrid model that combines the CNN and the LSTM models to process non-temporal/sequential data. Therefore, we investigate whether the hybrid model can be modified to estimate class-membership probabilities (propensity scores). To do this, we will use a sigmoid function as the last layer to give probabilities (propensity scores) between 0 and 1. A detailed description of CNN-LSTM and other hybrid models can be found in [59]. The architecture of the hybrid CNN-LSTM model is shown in Table 4.

| Type of Layer | Output shape | Number of Parameters |
|---------------|--------------|----------------------|
| Conv1D        | (None, 15, 64) | 128                  |
| Maxpooling1D  | (None, 15, 64) | 0                    |
| Conv1D        | (None, 15, 64) | 4160                 |
| Maxpooling1D  | (None, 15, 64) | 0                    |
| Conv1D        | (None, 15, 32) | 2080                 |
| Maxpooling1D  | (None, 15, 32) | 0                    |
| Dropout       | (None, 15, 32) | 0                    |
| LSTM          | (None, 15, 32) | 8320                 |
| Flatten       | (None, 480)   | 0                    |
| Dense         | (None, 128)   | 61568                |
| Dropout       | (None, 128)   | 0                    |
| Dense         | (None, 1)     | 129                  |

H. EXPERIMENTS

We performed experiments to fit deep learning methods and evaluated their performance in performing the classification task, and in estimating propensity scores. We provide some comparisons of LR, DNN, PN, CNN, and CNN-LSTM. We focus on analysing the performance of the aforementioned deep learning models in estimating propensity scores and not explaining the technical details of the methods. To estimate propensity scores using deep learning models, we used the covariates \( X_{ij} \) and the outcome variable \( Y_{ij} \) for all units as input and \( W_i \) as output.

The architecture of DNN used in this study is shown in Table 1. The proposed DNN is a variation of PropensityNet. DNN will have four hidden layers. The activation functions used are ReLU for DNN, PN, CNN, and CNN-LSTM. Reference [29] used Adadelta [47] as an optimiser for PropensityNet. However, for DNN, CNN, and CNN-LSTM will use the efficient Adam [45, 60] optimiser. Since we are performing binary classification, the binary cross-entropy is employed as the loss function. The dropout technique is used in all deep learning models to prevent overfitting [43]. Each deep learning model uses a sigmoid function as the last layer to give output probabilities between 0 and 1. Thus, the output is made up of probabilities, \( p(x) = Pr(W_i|X_i = x) \in [0, 1] \) for each unit of the test data set and these probabilities are the measure(s) of the propensity scores.

Keras [48] with the TensorFlow back end in R is used to build the deep learning models described above. Using R [61], we build logistic regression models using \( W_i \) as the dependent variable and \( X_i \) as the regressors to estimate propensity scores. We train the deep learning models using \( (X_i, Y_i) \) as the inputs for each unit. We have used simulation-based research for model performance evaluation because the true treatment effects are usually unknown in the real world when working with observational data [5].

I. EVALUATION METHODOLOGY

The performance of the models used in this paper was evaluated using the scenarios A−G, that were generated using a series of Monte Carlo simulation experiments that follow the structure of [30]. These scenarios A−G, represent different levels of linearity and additivity (including quadratic and interaction terms ) in the true propensity score models as described in Section III. 1000 data sets for each of the sample sizes equal to \( N = 500, N = 1000 \) and \( N = 2000 \) were used in each of the seven scenarios (A−G). The generated datasets are clearly imbalanced. Therefore, we used SMOTE (Synthetic Minority Oversampling Technique) [62] to closely match the minority class with the majority class.

The following evaluation metrics are used to assess whether our models estimate propensity scores that result in valid estimates of treatment effects [5, 20]. Therefore, these metrics are used to assess how LR, and the deep learning methods are performing in the binary classification task to obtain propensity scores as well as in estimating the treatment effects.

(i) Absolute Bias: Absolute bias is used to evaluate how the average treatment effect obtained from 1000 simulations of the different sample sizes agrees with the true value −0.4 for each scenario,

(ii) Standard error (s.e): This is calculated as the average standard error of the treatment effects resulting from the 1000 simulations for each scenario and different sample size. The smaller the average standard error, the less the spread and the more likely that an estimated treatment effect sample mean is close to the true value -0.4 of the treatment effect,

(iii) Average standardised absolute mean difference (ASAMD): We used Cobalt [63] to calculate the average standardised absolute mean difference between the treatment and control groups after incorporating propensity score weights. The ASMD is the average
of the absolute values of the standardised difference in means across all covariates for different scenarios and sample sizes. The average value of 1000-simulations is referred to as the mean ASAMD [5]. Lower ASAMD values suggest that the treatment and control groups are comparable for a given set of covariates.

(iv) **Accuracy (Acc):** Classification accuracy is used as a metric for evaluating the classification performance of our models. The classification accuracy simply gives the proportion of predictions that our model got right. For example, the higher the classification, the better the model is at classifying 0s as 0s and 1s as 1s. We note that classification accuracy may not be a good performance metric when one is working with a rare outcome $Y_i$, where there is a significant disparity between the number of 0s and 1s. Our outcome variable is a "rare" binary outcome $Y_i$ with $p(Y_i) \approx 0.02$ of the minority class and it is highly imbalanced. As a result, other performance metrics such as Cohen’s Kappa, AUC-ROC, and No Information Rate (NIR) are also used to evaluate the class-imbalanced data considered in this study instead of classification accuracy.

(v) **Cohen’s Kappa ($\kappa$):** is calculated using the formula, $\kappa = \frac{Pr(a) - Pr(e)}{1 - Pr(e)}$, where $Pr(a)$ represents the observed actual agreement, and $Pr(e)$ represents the chance agreement. In binary classification, accuracy is a common performance metric. However, it can be misleading in the case of imbalanced data [64]. For an imbalanced data set, the classification task may be influenced by the majority class. Therefore, instead of using the classification accuracy metric, Cohen’s Kappa will be used as one of the metrics to evaluate the agreement between the actual classes and the classes predicted by the DNN, PN, CNN, and CNN-LSTM and LR models. Cohen’s Kappa takes values between 0 and 1 with a value of 1 implying perfect agreement and values less than 1 imply less perfect agreement between actual and predicted classes [65].

(vi) **No Information Rate (NIR) and P-Value [Acc > NIR]:** The “no-information rate (NIR)” is the highest proportion of observed classes. This means that given a ‘rare’ binary outcome $Y_i$ with $p(Y_i) \approx 0.02$, the majority class has a probability approximately equal to 98%. A model whose classification accuracy is say 90% and the NIR is 98% tells us that if we just pick the majority class, we will be correct 98% of the time. A hypothesis test is also computed to evaluate whether the overall accuracy rate is greater than the rate of the majority class.

(vii) **AUC-ROC** is used as a performance measure to classify the binary class variable $Y_i$. The AUC-ROC is a probability curve that represents the degree or measure of separability. This means that it gives us a measure of how a model is capable of distinguishing between classes. For example, the higher the AUC, the better the model is at predicting 0s and 1s correctly. The AUC-ROC is a function of sensitivity and specificity.

(viii) **95% CI coverage:** This is the proportion of the 1000 data sets that contained the true treatment effect with 95% confidence.

**IV. RESULTS AND DISCUSSION**

1) **Parameter Settings**

We adapted the parameters settings from [59] and adjusted them to give the following optimal parameter settings (Tables 5 and 6) that were used to train CNN and CNN-LSTM models.

**TABLE 5. Optimum parameter settings for CNN and CNN-LSTM training**

| Parameters                        | CNN  | CNN-LSTM |
|-----------------------------------|------|----------|
| Droup-out                         | 0.1  | 0.1      |
| Learning-rate                     | 0.001| 0.001    |
| Batch-size                        | 64   | 64       |
| Number of LSTM cells              | -    | 32       |
| Number of CNN Kernels             | 64   | 64       |
| Number of training epochs         | 100  | 100      |
| Number of iterations for a scenario | 1000 | 1000    |

**TABLE 6. Optimum parameter settings for training DNN and PN**

| Parameters                        | DNN  | PNN |
|-----------------------------------|------|-----|
| Droup-out                         | 0.2  | 0.2 |
| Learning-rate                     | 0.001| 0.001|
| Batch-size                        | 128  | 128 |
| Number of hidden layers           | 3    | 5   |
| Batch Normalization               | 64   | -   |
| Trainable Parameters              | 3457 | 3713|
| Number of epochs                  | 100  | 100 |
| Number of iterations              | 1000 | 1000|

Using the parameter settings described in Tables 5 and 6, the results obtained by using the dataset of size $N = 1000$ generated from the Monte Carlo simulations, are presented first. Subsequently, we present performance comparisons of the different models using different sample sizes. This study was conducted to evaluate the performance of LR, DNN, CNN, CNN-LSTM and PN in estimating propensity scores, assessing covariate balance, and consequently, average treatment effects. Several critical discoveries about the performance of the different methods used in this paper in estimating propensity scores and average treatment effects were made. The aim was to determine whether propensity scores can be estimated using deep learning methods. The study extended the work by [2, 5, 20, 29] incorporating nonparametric statistical tests such as Cohen’s Kappa and the No Information Rate. Hypotheses tests were also performed to evaluate whether the overall accuracy rate was greater than the majority class (NIR) rate for each model. The performance of these propensity score methods was tested under different sample size conditions.
### A. SIMULATIONS OF $N = 1000$ SAMPLE SIZES

**TABLE 7.** Performance metrics for logistic regression (LR); deep neural network (DNN); PropensityNet (PN), convolutional neural network-long-short term model (CNN-LSTM), and convolutional neural network (CNN) for samples of size $N = 1000$.

| Scenario          | Metric     | LR    | DNN   | PN    | CNN   | CNN-LSTM | Average |
|-------------------|------------|-------|-------|-------|-------|----------|---------|
|                   | ASAMD      | 0.051 | 0.402 | 0.251 | 0.021 | 0.021    | 0.075   |
|                   | Absolute Bias | 0.005 | 0.025 | 0.010 | 0.013 | 0.013    | 0.038   |
|                   | Standard Error | 0.027 | 0.045 | 0.019 | 0.015 | 0.015    | 0.027   |
|                   | Accuracy (%) | 64.40 | 99.09 | 73.87 | 99.63 | 92.30    | 33.66   |
|                   | AUCROC (%)  | 27.91 | 99.76 | 50.18 | 99.34 | 86.32    | 33.66   |
|                   | Cohen' Kappa | 0.046 | 0.819 | 0.001 | 0.988 | 0.735    | 0.022   |
|                   | 95% Coverage | 90.60 | 96.10 | 100.00| 99.00 | 99.00    | 82.19   |
|                   | p-value [Acc > NIR] | 1.000 | 0.010 | 0.000 | 0.000 | 0.000    | 1.000   |

VOLUME xxxxx, 2021
ASAMD is an excellent measure to assess the covariate balance because it can effectively predict the bias of the average effect of treatment [66]. Reference [2] suggested as a "rule of thumb" a more stringent criterion for obtaining ASAMD values lower than 0.1 to achieve covariate balance. Using Table 7, DNN, and PropensityNet did not achieve covariate balance because their average ASAMD of each of these models was greater than 0.1 in all scenarios. On the other hand, LR, CNN and CNN-LSTM achieved covariate balance, as their respective mean ASAMD values were all less than 0.1.

Propensity score weighting is an important preprocessing technique used to achieve covariate balance. Achieving covariate balance justifies ignorability assumption on the observed covariates, which in turn allows for a valid causal inference to be made. Although logistic regression achieved covariate balance (Table 7), in all scenarios A-G, its absolute biases were consistently higher than those of DNN, PropensityNet, CNN, and CNN-LSTM. This important finding may suggest that achieving covariate balance may not be enough to lower absolute bias, or it may be that the ASAMD does not adequately measure covariate balance. This finding is supported by [2, 5, 67].

Table 7 shows that the absolute bias for the logistic regression for scenario A was generally acceptable and low (0.005, 95% CI =90.6%). Scenario A represents additivity and linearity (only main effects). However, as the scenarios became more nonadditive and non-linear (scenarios B-G), the performance of logistic regression was poor. For example, with moderate non-additivity and nonlinearity (scenario G), LR produced an average absolute bias of 0.043 and a 95% CI coverage of 51.1% (scenario G). These results show that the LR propensity score model gave higher estimates of the true causal effect of the treatment as the data became more non-additive and nonlinear. Furthermore, Table 7 shows that deep learning models; DNN, PN, CNN, and CNN-LSTM had low absolute biases averaging (in all scenarios) 0.009, 0.012, 0.018, and 0.015, respectively, and high 95% CI coverage. The DNN displayed the lowest bias as non-additivity, and nonlinearity in the data was increased. Therefore, DNN performed better in reducing absolute bias compared to LR, PN, CNN, and CNN-LSTM.

Standard errors for PN, CNN, and CNN-LSTM were comparable with averages of 0.019, 0.015, and 0.015 across all scenarios, respectively. On the other hand, the standard errors for LR and DNN were higher than those of PN, CNN, and CNN-LSTM as shown in Table 7. The low standard errors for PN, CNN, and CNN-LSTM were coupled with very good 95 percent CI coverage, which averaged 100.00%, 97.86%, and 95.00%, respectively for PN, CNN, and CNN-LSTM, across all scenarios.

Table 7 shows that on average, DNN, CNN, and CNN-LSTM gave higher classification accuracy values of 99.71%, 99.93%, and 97.03%, respectively, in all scenarios compared to LR (33.66%) and PN (75.72%). This means that DNN, CNN, CNN-LSTM were able to accurately classify the rare binary outcome variable $Y_i$, compared to PN and LR. The accuracy performance metric can be a useful measure if we have the same number of samples per class, but if we have an imbalanced set of samples, it is not useful at all. Therefore, we considered other measures such as the AUC-ROC, which gives the performance of a model, while addressing the issue of class imbalance to evaluate the performance of our models.

Based on the AUC-ROC, CNN performed better than the other models in all scenarios. The average AUC-ROC in the different scenarios was 27.9% for LR, 99.89% for CNN, 97.6% for DNN, 93.97% for CNN-LSTM, and 54.6% for PN, as shown in Table 7. The AUC-ROC results revealed that CNN returned a better classification compared to other algorithms. The AUC-ROC for LR and PN was poor and unacceptable. The AUC-ROC gives a better measure of accuracy because it is a function of sensitivity and specificity. The AUC-ROC curve is insensitive to differences in class proportions.

In addition to evaluating the models using classification accuracy and AUC-ROC, we also considered Cohen’s Kappa. The average Cohen’s Kappa values for LR, DNN, PN, CNN, and CNN-LSTM were 0.022, 0.906, 0.003, 0.998, and 0.881, respectively, for all scenarios (Table 7). Cohen’s Kappa statistic values for DNN, CNN, CNN-LSTM indicates that there was substantial agreement between the actual classes and the predicted classes [65]. Thus, DNN, CNN, CNN-LSTM can handle the imbalanced class problem very well. This means that DNN, CNN, CNN-LSTM are doing a good job of predicting propensity scores and also classifying 0s and 1s. Cohen’s Kappa statistic values for LR and PN did not offer any agreement between actual classes and predicted classes. These models are not capable of distinguishing between classes, that is, they are not good at predicting 0s and 1s correctly.

Table 7 shows that on average the $p$-value [Acc $> \text{NIR}$ ] is $0.000 < 0.05$, $0.009 < 0.005$, and $0.032 < 0.05$ for CNN, DNN, and CNN-LSTM, respectively. This means that the average classification accuracy for CNN, DNN, and CNN-LSTM is significantly greater than the No Information Rate. Thus, CNN, DNN, and CNN-LSTM are useful models to predict propensity scores that are used to calculate the average effects of treatment. The $p$-values [Acc $> \text{NIR}$ ] for LR and PN were not significant as they were all greater than 0.05. With the no-information rate in mind, we now see that the accuracy of the LR model is poor. We note that a good model is one where the no information rate is significantly less than the classification accuracy. It is important to check whether the accuracy is significantly greater than the no information rate (NIR) to determine whether the model is actually doing anything useful for the particular outcome it claims to predict.
B. PERFORMANCE COMPARISONS FOR SAMPLES OF SIZE $N = 500$, $N = 1000$, AND $N = 2000$.

In this section we investigate the discriminative performances of LR, DNN, PN, CNN, and CNN-LSTM models. These models were evaluated on the same training datasets derived from the datasets of sizes $N = 500$, $N = 1000$, and $N = 2000$.

1) Performance Comparison of the Classification Accuracy

According to Fig. 1, we can see that based on the classification accuracy, the deep learning models CNN, DNN, and CNN-LSTM outperform LR across the different sample sizes. These results demonstrate that these models are effective for estimating propensity scores using class-membership probabilities compared to LR.

2) Performance Comparison of the AUCROC

As shown in Fig. 2, we find that the AUCROC of CNN and DNN are consistently greater than 0.9 for the different sample sizes. The comparison results in Fig. 2, for the different models empirically demonstrate that CNN and DNN are very useful models for classifying imbalanced datasets. However, the combination of CNN and LSTM does not appear to improve the AUROC. This may be due to the fact that LSTMs, an instance of recurrent neural networks, are more suitable for temporal data such as time series [68] or sequential data [59].

3) Performance comparison of the ASAMD

Fig. 2 shows the ASAMD values of LR, DNN, PN, CNN, and the hybrid model CNN-LSTM. We see that LR, CNN, and CNN-LSTM produced ASAMD values less than 0.1 compared to DNN and PN that produced ASAMD values greater than 0.1. Therefore, LR, CNN, and CNN-LSTM models achieved covariate balance, and they are highly useful for estimating propensity scores and achieving covariate balance. This is the first time that the CNN, and a hybrid CNN-LSTM model have been used to estimate propensity scores, and at the same time assess covariate balance of the CNN and CNN-LSTM models.

4) Performance comparison of Absolute Bias

The results in Fig. 4 clearly show that LR had higher absolute bias across all the sample sizes, compared to CNN that consistently produced lower absolute bias. Furthermore, DNN produced the least bias for samples of size $N = 500$ and 1000. These results show that CNN, DNN, and CNN-LSTM are useful models for reducing absolute bias of the estimated causal effects of the treatment. This means that these deep learning methods can be used as alternatives to LR for estimating propensity scores and consequently, average causal effect estimates of the treatment.
5) Performance comparison of Cohen's Kappa

The Cohen's Kappa results shown in Fig. 5 for CNN, DNN, and CNN-LSTM clearly outperform those of LR and PN. This means that CNN, DNN, and CNN-LSTM offer better agreement between the actual classes and the predicted classes compared to LR and PN. Therefore, CNN, DNN, and CNN-LSTM have greater predictive capability compared to LR and PN.

6) CPU elapsed time

Table 9 shows the CPU elapsed time in seconds for LR, DNN, PN, CNN, and CNN-LSTM, respectively. These times represent the average time it takes for each model to execute 100 epochs. Furthermore, Table 9 shows that CNN and CNN-LSTM have more than 18 times the number of trainable parameters than those of DNN and yet their model training time is comparable to that of DNN. These results show that CNN takes much less time to execute 100 epochs than DNN and PN. Moreover, the results in Table 9 show that the additional time, and the additional trainable parameters of CNN-LSTM do not offer better performance compared to CNN. Therefore, we can conclude that CNN has outperformed all the models based on the performance metrics and the model training times.

C. CASE STUDY

We apply the proposed models to a quasi-real-world data set [69]. This publicly available dataset is used a case study to assess how well our proposed models perform when they are used to estimate propensity scores using a real-world dataset. This data set comes from the Atlantic Causal Inference Conference (ACIC) Data Challenge 2019 (https://www.mcgill.ca/epi-biostat-occh/seminars-events/atlantic-causal-inference-conference-2019). The data set has a known estimate of the population ATE. The challenge was to calculate an estimate of the population ATE. Some of the covariates used in the ACIC 2019 data challenge were derived from simulations and publicly available data sets. The link to the R code for the data generation processes is available in [69]. Various challenges of estimating the average treatment effects were incorporated into the processes to generate the binary treatment assignment, as well as the binary or continuous outcomes. These challenges include violations of the positivity assumption, different proportions of true confounders among the observed covariates, heterogeneity of the treatment effect, and nonlinearity of the response surface. The data sets consist of 3200 low-dimensional data sets and 3200 high-dimensional data sets. We randomly selected a low-dimensional data set of size \( N = 5735 \) with 26 covariates and the true \( \text{ATE} = 2.5274 \).

The chosen data set has six binary variables, one categorical variable with four levels, 14 continuous variables, and five integer variables. The data set is unbalanced with a less frequent binary outcome \( Y_i \) with \( P(Y_i) \approx 0.12 \) and a binary treatment variable \( A_i \) with \( P(A_i) \approx 0.46 \).

The results in (Table 10) show that the average treatment effect estimates obtained from the CNN had the least biased causal effect estimates compared to the other models.

Furthermore, the ASAMD results for DNN and PN were greater than 0.1, indicating that these models did not achieve more than 18 times the number of trainable parameters than those of PN, and yet its model training time is comparable to that of PN. These results show that CNN takes much less time to execute 100 epochs than DNN and PN. Moreover, the results in Table 9 show that the additional time, and the additional trainable parameters of CNN-LSTM do not offer better performance compared to CNN. Therefore, we can conclude that CNN has outperformed all the models based on the performance metrics and the model training times.
covariates balance. CNN gave much better classification accuracy (99.97%). The AUC-ROC for CNN was between 90%-100%, which is considered excellent [70]. The AUC-ROC for LR and PN were between 30%-60% suggesting that these models were poor at classifying 0s and 1s. Table 10 shows that Cohen’s Kappa for CNN is above 0.90, indicating excellent agreement between actual and predicted classes. There was no agreement between the actual and predicted classes for PN because it had a very low Cohen Kappa value. Furthermore, the classification accuracy values for CNN, DNN, and CNN-LSTM were significantly higher than the No Information Rate (NIR) as the p-value [Acc > NIR] for each of these models was significant (> 0.05). This means that CNN, DNN, and CNN-LSTM are useful models to estimate propensity scores because they all produced high classification accuracy, Cohen’s Kappa, and AUC-ROC values. Furthermore, they produced low absolute bias values when applied to the complex ACIC real-world dataset compared to LR. The ACIC 2019 data set used in this paper was complex because challenges such as nonlinearity of the response surface, treatment effect heterogeneity, varying proportion of true confounders among the observed covariates, and near violations of the positivity assumption were incorporated into the data generation process of the dichotomous treatment assignment, and the binary or continuous outcomes.

V. CONCLUSION
In conclusion, our simulation results show that using deep learning models (CNN, DNN, and CNN-LSTM) offers a number of advantages over logistic regression in the estimation of the propensity score. In this paper, we have considered the estimation of propensity scores as the classification task described in Sections III and III-C. We have modelled not only a class label for a data item, but also a probability of class membership (propensity scores). In performing the classification task, the functional forms for LR and deep learning models differ in that LR is a parametric method, whereas deep learning models are semi-parametric or non-parametric. This distinction is important because the contribution of parameters in LR (coefficients and intercept) can be interpreted, whereas this is not always the case with the parameters of a deep learning models (weights). Due to nonlinearity in the hidden neurons of deep learning models, the output of deep learning models is a non-linear function of the inputs. This means that in performing the classification task, deep learning models are more flexible compared to LR.

LR has low model complexity and loses its flexibility because of the need to perform feature selection. On the other hand, deep learning algorithms learn the features from the data instead of handcrafted feature extraction. Therefore, compared to LR, deep learning models are more flexible. However, deep learning models are more susceptible to overfitting with improper configuration. Proper configuration of some deep learning models includes restricting the network size, decreasing the number of variables and hidden neurons, pruning the network after training, and regularization. This paper has presents an in-depth comparison between LR and deep learning algorithms. The research used several performance metrics to compare LR and deep learning models. Generally, deep learning models outperformed LR. The difference in performance comes from the nonlinearity of the solutions developed by LR and the deep learning models. Furthermore, LR applies nonlinearity to each individual variable followed by a linear multivariate transformation, whereas deep learning models apply a non-linear transformation in a truly multidimensional space. This means that deep learning models can estimate many more parameters as well as many more permutations of the parameters than LR.

The fact that deep learning models outperformed LR shows that deep learning models achieve a good ratio of data points to parameters, thereby achieving more reliable estimates of the propensity scores. The reliability of the propensity scores derived from the deep learning models supported by the good performance metrics results that these models produced. For example, the results show that CNN, DNN, and CNN-LSTM can significantly improve the 95% CI coverage and also significantly reduce bias over a range of sample sizes from N = 500 to N = 2000, and scenarios A-G, compared to logistic regression. CNN, DNN, and CNN-LSTM also have excellent predictive performance in modelling rare binary outcomes. These deep learning models proved to be useful in predicting propensity scores, as they produced excellent values for classification accuracy, AUC-ROC, Cohen’s Kappa and significant p-values [Acc > NIR]. PN with five fully connected layers performed poorly compared to DNN with four hidden layers. Furthermore, we have shown that deep learning models can successfully be employed as suitable options for estimating propensity scores. The advantage of using deep learning models compared to logistic regression is that they do not depend on assumptions regarding (i) how variables are selected, (ii) specification of the correct functional form, (iii) statistical distributions of the variables, and (iv) interactions are specified [18]. If assumptions are not met when using logistic regression, biased estimates of treatment effects may be obtained due
to not achieving covariate balance. The literature on using deep learning methods to estimate propensity scores using class-membership probabilities is still limited. This study has shown that with the correct configuration, deep learning methods can be employed to reduce or eliminate the reliance on logistic regression assumptions regarding variable selection, the functional form, distribution of variables, and specification of interactions. Furthermore, CNN and CNN-LSTM have shown that they can perform better than logistic regression in estimating the propensity score and achieving covariate balance. Furthermore, they can reduce bias when used to estimate the average effects of treatment. Thus, the deep learning models, CNN, DNN, and CNN-LSTM can be used in situations where the objective is to reduce absolute bias in the causal effect estimates.

We strongly recommend that further research should focus on applying CNN, DNN, and CNN-LSTM models in more real-life situations to estimate propensity scores. More research should be done using these models to compare the performance of propensity score matching and propensity score weighting when there are multiple treatment groups. The results of such research will further inform researchers and practitioners regarding the advantages and disadvantages of deep learning methods in situations where we have different propensity score analysis techniques (matching vs. weighting) and more than two treatment groups. To our knowledge, the application of propensity scores derived from deep learning methods to estimate average treatment effects has not been explored for studies with multiple treatment groups.

REFERENCES
[1] P. R. Rosenbaum and D. B. Rubin, “The central role of the propensity score in observational studies for causal effects,” Biometrika, vol. 70, no. 1, pp. 41–55, 1983.
[2] M. Cannas and B. Arpino, “A comparison of machine learning algorithms and covariate balance measures for propensity score matching and weighting,” Biometrical Journal, vol. 61, no. 4, pp. 1049–1072, 2019, doi: 10.1002/bimj.201800132.
[3] A. Olmos and P. Govindasamy, “Propensity scores: a practical introduction using r,” Journal of MultiDisciplinary Evaluation, vol. 11, no. 25, pp. 68–88, 2015. [Online]. Available: https://journals.sfu.ca/jmde/index.php/jmde_1/article/view/431
[4] P. C. Austin, “An introduction to propensity score methods for reducing the effects of confounding in observational studies,” Multivariate behavioral research, vol. 46, no. 3, pp. 399–424, 2011, doi: 10.1080/00273171.2011.568786.
[5] B. K. Lee, J. Lessler, and E. A. Stuart, “Improving propensity score weighting using machine learning,” Statistics in medicine, vol. 29, no. 3, pp. 337–346, 2010, doi: 10.1002/sim.3782.
[6] D. Westreich, J. Lessler, and M. J. Funk, “Propensity score estimation: machine learning and classification methods as alternatives to logistic regression,” Journal of clinical epidemiology, vol. 63, no. 8, p. 826, 2010, doi: 10.1016/j.jclinepi.2009.11.020.
[7] S. Athey and G. W. Imbens, “Machine learning methods for estimating heterogeneous causal effects,” stat, vol. 1050, no. 5, 2015. [Online]. Available: https://www.gsb.stanford.edu/faculty-research/working-papers/machine-learning-estimating-heretogeneous-casual-effects
[8] M. S. Ali, D. Prieto-Alhambra, L. Lopes, D. Ramos, N. Bispo, M. Y. Ichihara, J. M. Pescarini, E. Williamson, R. Fiaconne, M. L. Barreto et al., “Propensity score methods in health technology assessment: principles, extended applications, and recent advances,” Frontiers in Pharmacology, vol. 10, p. 973, 2019, doi: 10.3389/fphar.2019.00973.
[9] J. Antonakis, S. Bendahan, P. Jaccart, and R. Lalive, “Causality and endogeneity: problems and solutions,” The Oxford handbook of leadership and organizations, p. 93, 2014, doi: 10.1093/oxfordhb/9780199755615.013.007.
[10] S. Athey, G. W. Imbens, and S. Wager, “Approximate residual balancing: De-biased inference of average treatment effects in high dimensions,” arXiv preprint arXiv:1604.07125, 2016, doi: 10.1111/rssb.12268.
[11] P. W. Holland, “Statistics and causal inference,” Journal of the American Statistical Association, vol. 81, no. 396, pp. 945–960, 1986, doi: 10.1002/j.2330-8516.1985.tb00125.x.
[12] J. Kang, W. Chan, M.-O. Kim, and P. M. Steiner, “Practice of causal inference with the propensity of being zero or one: assessing the effect of arbitrary cutoffs of propensity scores,” Communications for statistical applications and methods, vol. 23, no. 1, p. 1, 2016, doi: 10.5351/CSAM.2016.23.1.001.
[13] M. L. Petersen, K. E. Porter, S. Gruber, Y. Wang, and M. J. Van Der Laan, “Diagnosing and responding to violations in the positivity assumption,” Statistical methods in medical research, vol. 21, no. 1, pp. 31–54, 2012, doi: 10.1177/0962280210386207.
[14] K. Imai and M. Ratkovic, “Covariate balancing propensity score,” Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 76, no. 1, pp. 243–263, 2014, doi: 10.1111/rssb.12027.
[15] S. J. Beal and K. A. Kupzyk, “An introduction to propensity scores: what, when, and how,” The Journal of Early Adolescence, vol. 34, no. 1, pp. 66–92, 2014, doi: 10.1177/0736978X13481422.
[16] F. Li, K. L. Morgan, and A. M. Zaslavsky, “Balancing covariates via propensity score weighting,” Journal of the American Statistical Association, vol. 113, no. 521, pp. 390–400, 2018, doi: 10.1080/01621459.2016.1260466.
[17] P. C. Austin and E. A. Stuart, “Moving towards best practice when using inverse probability of treatment weighting (iptw) using the propensity score to esti-
mate causal treatment effects in observational studies,” Statistics in medicine, vol. 34, no. 28, pp. 3661–3679, 2015, doi: 10.1002/sim.6607.

[18] E. A. Stuart, “Matching methods for causal inference: A review and a look forward,” Statistical science: a review journal of the Institute of Mathematical Statistics, vol. 25, no. 1, p. 1, 2010, doi: 10.1214/09-STS313.

[19] Y. Zhong and M. Zhao, “Research on deep learning in apple leaf disease recognition,” Computers and Electronics in Agriculture, vol. 168, p. 105146, 2020, doi: 10.1016/j.compag.2019.105146.

[20] S. Setoguchi, S. Schneeweiss, M. A. Brookhart, R. J. Glynn, and E. F. Cook, “Evaluating uses of data mining techniques in propensity score estimation: a simulation study,” Pharmacoepidemiology and drug safety, vol. 17, no. 6, pp. 546–555, 2008, doi: 10.1002/pds.1555.

[21] R. Wyss, A. R. Ellis, M. A. Brookhart, C. J. Girman, M. Jonsson Funk, R. LoCasale, and T. Stürmer, “The role of prediction modeling in propensity score estimation: an evaluation of logistic regression, bcart, and the covariate-balancing propensity score,” American journal of epidemiology, vol. 180, no. 6, pp. 645–655, 2014, doi: 10.1093/aje/kwu181.

[22] B. B. P. de Vries, M. van Smeden, and R. H. Groenwold, “Propensity score estimation using classification and regression trees in the presence of missing covariate data,” Epidemiologic Methods, vol. 7, no. 1, 2018, doi: 10.1515/em-2017-0020.

[23] K. Brown, P. Merrigan, and J. Royer, “Estimating average treatment effects with propensity scores estimated with four machine learning procedures: Simulation results in high dimensional settings and with time to event outcomes,” Available at SSRN 3272396, 2018, doi: 10.2139/ssrn.3272396.

[24] P. Zhao, X. Su, T. Ge, and J. Fan, “Propensity score and proximity matching using random forest,” Contemporary clinical trials, vol. 47, pp. 85–92, 2016, doi: 10.1016/j.cct.2015.12.012.

[25] Y. Zhu, D. L. Coffman, and D. Ghosh, “A boosting algorithm for estimating generalized propensity scores with continuous treatments,” Journal of causal inference, vol. 3, no. 1, pp. 25–40, 2015, doi: 10.1515/jci-2014-0022.

[26] Y. Yuan, X. Ding, and Z. Bar-Joseph, “Causal inference using deep neural networks,” arXiv preprint arXiv:2011.12508, 2020. [Online]. Available: https://arxiv.org/abs/2011.12508.

[27] M. Śmieja, Ł. Struski, J. Tabor, B. Zieliński, and P. Szurek, “Processing of missing data by neural networks,” in Advances in Neural Information Processing Systems, 2018, pp. 2719–2729. [Online]. Available: https://papers.nips.cc/paper/2018/file/411e1bf081d1674ca6091fc59a266f-Paper.pdf

[28] M. H. Farrell, T. Liang, and S. Misra, “Deep neural networks for estimation and inference,” arXiv preprint arXiv:1809.09953, 2018.

[29] V. Ramachandra, “Deep learning for causal inference,” arXiv preprint arXiv:1803.00149, 2018.

[30] C. M. Setodji, D. F. McCaffrey, L. F. Burgette, D. Almirall, and B. A. Griffin, “The right tool for the job: choosing between covariate balancing and generalized boosted model propensity scores,” Epidemiology (Cambridge, Mass.), vol. 28, no. 6, p. 802, 2017, doi: 10.1097/EDE.0000000000000734.

[31] S. Gharibzadeh, M. A. Mansournia, A. Rahimiforoushani, A. Alizadeh, A. Amouzegar, K. Mehrbani-Zeinabadi, and K. Mohammad, “Comparing different propensity score estimation methods for estimating the marginal causal effect through standardization to propensity scores,” Communications in Statistics-Simulation and Computation, vol. 47, no. 4, pp. 964–976, 2018, doi: 10.1080/03610918.2017.1300267.

[32] M.-J. Woo, J. P. Reiter, and A. F. Karr, “Estimation of propensity scores using generalized additive models,” Statistics in medicine, vol. 27, no. 19, pp. 3805–3816, 2008, doi: 10.1002/sim.3278.

[33] 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, doi: 10.1162/003465302760198218.

[34] A. Linden, S. D. Uysal, A. Ryan, and J. L. Adams, “Estimating causal effects for multivalued treatments: a comparison of approaches,” Statistics in Medicine, vol. 35, no. 4, pp. 534–552, 2016, doi: 10.1002/sim.6768.

[35] B. Macukow, “Neural networks–state of art, brief history, basic models and architecture,” in IFIP international conference on computer information systems and industrial management. Springer, 2016, pp. 3–14.

[36] A. J. Dobson and A. G. Barnett, An introduction to generalized linear models. CRC press, 2018.

[37] C. J. Muller and R. F. MacLehose, “Estimating predicted probabilities from logistic regression: different methods correspond to different target populations,” International journal of epidemiology, vol. 43, no. 3, pp. 962–970, 2014, doi: 10.1093/ije/dyu029.

[38] Z. Zhang and M. Sabuncu, “Generalized cross entropy loss for training deep neural networks with noisy labels,” in Advances in neural information processing systems, 2018, pp. 8778–8788. [Online]. Available: https://arxiv.org/abs/1805.07836

[39] A. Sivaram, L. Das, and V. Venkatasubramanian, “Hidden representations in deep neural networks: Part 1. classification problems,” Computers & Chemical Engineering, vol. 134, p. 106669, 2020, doi: 10.1016/j.compchemeng.2019.106669.

[40] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis, “Adaptive activation functions accelerate convergence in deep and physics-informed neural networks,” Journal of Computational Physics, vol. 404, p. 109136, 2020. [Online]. Available: https://arxiv.org/abs/1906.
[41] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” IEEE transactions on pattern analysis and machine intelligence, vol. 35, no. 8, pp. 1798–1828, 2013, doi: 10.1109/TPAMI.2013.50.

[42] F. Chollet, Deep learning with Python. Simon and Schuster, 2021.

[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.

[44] Y. Hu, S. Luo, L. Han, L. Pan, and T. Zhang, “Deep supervised learning with mixture of neural networks,” Artificial Intelligence in Medicine, vol. 102, p. 101764, 2020, doi: 10.1016/j.artmed.2019.101764.

[45] C. Nwankpa, W. Jomah, A. Gachagan, and S. Marshall, “Activation functions: Comparison of trends in practice and research for deep learning,” arXiv preprint arXiv:1811.03378, 2018. [Online]. Available: https://arxiv.org/abs/1811.03378

[46] J. Friedman, T. Hastie, and R. Tibshirani, The elements of statistical learning. Springer series in statistics New York, 2001, vol. 1.

[47] M. D. Zeiler, “Adadelta: an adaptive learning rate method,” arXiv preprint arXiv:1212.5701, 2012. [Online]. Available: https://arxiv.org/abs/1212.5701

[48] F. Chollet, J. Allaire et al., “R interface to keras,” 2017. [Online]. Available: https://github.com/rstudio/keras

[49] M. Jogin, M. Madhulika, G. Divya, R. Mehnaga, S. Apoorva et al., “Feature extraction using convolution neural networks (cnn) and deep learning,” in 2018 3rd IEEE international conference on recent trends in electronics, information & communication technology (RTEICT). IEEE, 2018, pp. 2319–2323.

[50] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” Insights into imaging, vol. 9, no. 4, pp. 611–629, 2018.

[51] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative study of cnn and rnm for natural language processing,” arXiv preprint arXiv:1702.01923, 2017.

[52] L. Hongtao and Z. Qinchuan, “Applications of deep convolutional neural network in computer vision,” Journal of Data Acquisition and Processing, vol. 31, no. 1, pp. 1–17, 2016.

[53] G. Jain, M. Sharma, and B. Agarwal, “Spam detection on social media using semantic convolutional neural network,” International Journal of Knowledge Discovery in Bioinformatics (IJKDB), vol. 8, no. 1, pp. 12–26, 2018.

[54] A. Jacobi, O. S. Shalom, and Y. Goldberg, “Understanding convolutional neural networks for text classification,” arXiv preprint arXiv:1809.08037, 2018.

[55] R. Johnson and T. Zhang, “Effective use of word order for text categorization with convolutional neural networks,” arXiv preprint arXiv:1412.1058, 2014.

[56] N. Jmour, S. Zayen, and A. Abdelkrim, “Convolutional neural networks for image classification,” in 2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET). IEEE, 2018, pp. 397–402.

[57] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[58] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[59] A. Whata and C. Chimedza, “Deep learning for sars cov-2 genome sequences,” IEEE Access, vol. 9, pp. 59597–59611, 2021.

[60] D. P. Kingma and J. B. Adam, “a method for stochastic optimization. arxiv preprint arXiv:1412.6980, 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[61] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2017. [Online]. Available: https://www.R-project.org/

[62] L. Lusa et al., “Evaluation of smote for high-dimensional class-imbalanced microarray data,” in 2012 11th international conference on machine learning and applications, vol. 2. IEEE, 2012, pp. 89–94.

[63] N. Greifer, “cobalt: Covariate balance tables and plots,” R package version, vol. 2, 2017. [Online]. Available: https://nrgreifer.github.io/cobalt/

[64] J. Akosa, “Predictive accuracy: A misleading performance measure for highly imbalanced data,” in Proceedings of the SAS Global Forum, 2017, pp. 2–5.

[65] J. R. Landis and G. G. Koch, “The measurement of observer agreement for categorical data,” biometrics, vol. 33, no. 1, pp. 159–174, 1977.

[66] E. A. Stuart, B. K. Lee, and F. P. Leacy, “Progression of disease: a method for modeling recurrent events,” Biostatistics, vol. 6, no. 1, pp. 21–37, 2005.

[67] S. Apoorva et al., “Feature extraction using convolution neural networks (cnn) and deep learning,” in 2018 3rd IEEE international conference on recent trends in electronics, information & communication technology (RTEICT). IEEE, 2018, pp. 2319–2323.

[68] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” Insights into imaging, vol. 9, no. 4, pp. 611–629, 2018.

[69] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative study of cnn and rnm for natural language processing,” arXiv preprint arXiv:1702.01923, 2017.

[70] L. Hongtao and Z. Qinchuan, “Applications of deep convolutional neural network in computer vision,” Journal of Data Acquisition and Processing, vol. 31, no. 1, pp. 1–17, 2016.

[71] G. Jain, M. Sharma, and B. Agarwal, “Spam detection on social media using semantic convolutional neural network,” International Journal of Knowledge Discovery in Bioinformatics (IJKDB), vol. 8, no. 1, pp. 12–26, 2018.

[72] A. Jacobi, O. S. Shalom, and Y. Goldberg, “Understanding convolutional neural networks for text classification,” arXiv preprint arXiv:1809.08037, 2018.

[73] R. Johnson and T. Zhang, “Effective use of word order for text categorization with convolutional neural networks,” arXiv preprint arXiv:1412.1058, 2014.
ALBERT WHATA is working towards a PhD in Statistics, and he currently works at Sol Plaatje University South Africa as a Statistics Lecturer. He holds a master’s degree in Statistics. His research interests are in the fields of causal inference, biostatistics, econometrics, and machine learning and deep learning with applications to statistics.

DR CHARLES CHIMEDZA currently works at the School of Statistics and Actuarial Science, at the University of the Witwatersrand South Africa as a Senior Lecturer and the Vice Head of the School of Statistics and Actuarial Studies. His research interests are in the fields of robust statistics, change point models, mixed models, and statistical computing.