Erasmus University Rotterdam
Erasmus School of Economics

Master Thesis Econometrics & Management Science
Business Analytics and Quantitative Marketing

Leveraging GANs to Alleviate the Unbalanced Data Set Problem and Improve Conversion Prediction

Author:
Damian Gorczynski *(466892)*

Supervisor:
dr. Kathrin Gruber
Second assessor:
dr. Andreas Alfons

January 17, 2022

Acknowledgements

I would like to express gratitude towards dr. Kathrin Gruber for her invaluable guidance in this research project. Your feedback pushed me to learn beyond my master's degree and bring my work to a higher level.

Abstract

The focus of this research paper is to answer the research question *To what extent can Generative Adversarial Networks alleviate the unbalanced dataset problem and improve the Attribution modelling of the online digital campaign data?*. This is accomplished by exploring how artificial data generation and anomaly detection methods can be used to alleviate the unbalanced dataset problem. Hence, two models are developed. The first model is a GAN supported Attribution Recurrent Neural Network (ARNN) model motivated by Ren et al. (2018) and Goodfellow et al. (2014), which aims to improve the ARNN model performance by modelling the conversion sequence distribution and then generating artificial convergent sequences to balance the dataset on which a standard ARNN model is then trained. The second model is an Adjusted Adversarially Learned Anomaly Detection model motivated by Zenati et al. (2018), which treats conversions as anomalies and tries to distinguish between the non-convergent sequences and their anomalous counterparts. The research finds that Generative Adversarial Networks can successfully generate artificial data that can improve model training and performance. On the other hand, the results suggest that the Generative Adversarial Network architecture is not able to improve the Attribution modelling of the online digital campaign data.

Table of Contents

1 Introduction

Online shopping has been on the rise for over a decade. Recently, it has been accelerated by the coronavirus pandemic which forced many physical stores to close due to mandatory country-level lockdowns. This led many shoppers to turn to online stores, where purchasing items was still possible with the option of affordable home delivery thereby fuelling the growth of the eCommerce business across the globe. Market analysts argue that the eCommerce industry will benefit the most from the pandemic. It is projected that the eCommerce industry will increase to 4.5 trillion USD in 2021, with current penetration rates of 15% to rise to 25% by 2025 (Mohsin, 2020). Furthermore, (Mohsin, 2020) outlines that 73% of online sales will be made via mobile devices by the end of 2021 with young consumers being the driving force behind the transition. This explains why social media networks such as Facebook and Instagram have such a strong effect on this trend.

With this immense growth in the sector, new opportunities arise for companies to use the terabytes of data generated by online shoppers to improve their business results. Online shopping makes it feasible for companies to track their customers more closely and use this data to improve their online marketing strategies to increase their conversion rates. In online marketing literature, conversion rate is defined as the percentage of customers who purchase a product after being targeted by online ad campaigns, where each interaction with the user is referred to as a touch-point or impression. Ren et al. (2018) noticed that there is a literature gap in modelling sequential patterns of individual internet users leading up to the final conversions. Despite the fact that many models assign the final touch-point full credit for leading the customer to their final purchase, the customer trip through different touch-points before the conversion is assumed to not be arbitrary. Hence, Ren et al. (2018) developed a Neural Network architecture to account for the sequential user interaction with the ad content, where experimental results show performance improvements over basic models.

Nonetheless, the methodology developed by Ren et al. (2018) may still not be completely appropriate for the problem at hand. With high traffic across the internet, it is common to observe conversion rates of 99:1, if not lower. And with the continuous rise of internet users, it is only expected that these conversion rates become more extreme. This is highly problematic because such low conversion rates introduce under-sampling problems that deteriorate Neural Network-based model performance. Hence, it is pivotal to account for this issue. One of the solutions is to treat conversion events as outliers and deploy outlier detection algorithms to detect the conversions. This approach has the potential to overcome the unbalanced class problem because outlier detection

algorithms are made to work with this kind of data and can be supervised, unsupervised and semi-supervised, thereby facilitating many possibilities of training the models without large fully labelled datasets. Applying outlier detection algorithms to model conversions is a novel approach. If successful, this research may spark an interest in the development of more tailored models to optimise online marketing strategies to achieve higher conversion rates and decrease marketing costs.

The issue discussed above gives rise to the research question of this paper:

To what extent can Generative Adversarial Networks alleviate the unbalanced dataset problem and improve the attribution modelling of the online digital campaign data?

In particular, two separate approaches are tested in this research paper. The first approach uses a Generative Adversarial Network to generate artificial conversion data that is then used to support training the Attribution model. The second approach implements an Attribution model within the Generative Adversarial Network for Anomaly Detection. Hence, the following sub-questions are taken into account:

1. Can Generative Adversarial Networks generate artificial conversion data to improve Attribution model performance?

2. Can Generative Adversarial Networks for Anomaly Detection be jointly implemented with an Attribution model to improve the Attribution model performance?

3. Can Neural Network based models competently model attribution and thereby efficiently allocate budgets?

This research finds that Generative Adversarial Networks can be successfully used to generate artificial data that can then supplement model training and alleviate the unbalanced dataset problem. Implementing Generative Adversarial Networks for Anomaly Detection fails to improve model performance, which may be due to sub-optimal model architecture or inefficient model training. Lastly, the Generative Adversarial Network architecture is not able to improve the attribution modelling of the online digital campaign data.

This research paper continues as follows. Section 2 gives an overview of existing Attribution models, starting from the most basic ones and elaborating to more advanced models. Section 3 describes the Criteo dataset and outlines the data preprocessing steps. Section 4 explains each method used in this research along with the evaluation metrics used for model comparison. Subsequently, Section 5 discusses the results and Section 6 concludes the research paper.

2 Literature

This section begins with an outline of existing literature on Attribution models and continues with a discussion of former research that motivates the use of Generative Adversarial Networks in this research paper.

2.1 Literature on existing Attribution model research

Many existing works propose different solutions to the Attribution problem. These solutions begin with traditional heuristic approaches and extend to advanced models that attempt to capture complicated user patterns over time. These approaches are discussed below, motivating the use of the Attribution model proposed by Ren et al. (2018).

Traditional Attribution models implement heuristic approaches of assigning conversion credit to different touch points (Wang et al., 2017). Some examples are first-touch, last-touch and linear-touch. The first-touch heuristic assigns all the credit to the first user interaction with ad content of a company, whereas the last-touch heuristic assigns it to the last user interaction. The linear-touch heuristic splits the credit uniformly across all user interactions with the ad content leading up to a conversion. The advantage of these methods is their simple and easy implementation at scale. However, heuristic models only implement insights of the convergent sequences, thereby ignoring the information from the larger population of non-convergent sequences.

Sequential pattern models go a step further from heuristic approaches. Probabilistic models (Shao and Li, 2011) and models based on Survival theory (Zhang et al., 2014) such as Additive Hazard models (Ji and Wang, 2017) were developed to capture the sequential nature of the user interaction data. These methods assume that user conversion is always positively influenced by each interaction with the ad content. However, this may not be the case in real life, where some ad content may discourage users from the final purchase, thus lowering the likelihood of their conversion.

Another issue seen across Attribution model research is the failure to distinguish between different types of user behaviors (Ren et al., 2018). For example, researchers treat clicks and conversions as the same event and ignore the difference between click and non-click impressions. This poses a problem because their models ignore relevant information behind user interaction with the ad content.

Ren et al. (2018) address the above issues through their Dual-Attention Recurrent Neural Network (DARNN) model. The model incorporates an Encoder for impression-level behavior modelling,

a Decoder for sequential prediction of click probability and a Dual-Attention model to combine the two to estimate the final conversion probability. As a result, the DARNN model utilises sequence-to-sequence prediction for user clicks and models post-view and post-click attribution patterns to estimate the final user conversion.

Model	**AUC**	**Log-loss**
Additive Hazard	0.6791	0.5067
Additional Multi-touch Attribution	0.8465	0.3897
ARNN	0.9793	0.1850
DARNN	0.9799	0.1591

Table 2.1: Criteo dataset results (Ren et al., 2018)

The DARNN model proves to be highly effective in the experimental set-up from Ren et al. (2018), shown in Table 2.1 on the Criteo dataset. However, Attribution problem datasets are known for being highly unbalanced, posing a problem to many statistical models, DARNN being one of them. Ren et al. (2018) deal with this problem by under-sampling from the non-convergent population to obtain a 1:20 ratio of convergent to non-convergent sequences.

While the DARNN model proves to be highly effective, only the encoder part of the model with a single attention layer based on impression-level patterns will be used, i.e. ARNN. This is due to the fact that the DARNN model trains nine deep-neural networks, while the ARNN model only trains three, posing a significantly higher computational requirement to estimate the model parameters. Given the performance results from Ren et al. (2018) displayed in Table 2.1, the ARNN model outperforms all models aside from the DARNN model, where its AUC performance is marginally lower and Log-loss is about 16% higher. Hence, the ARNN model is more ideal for the discussed research objective, because it requires significantly less computational resources while maintaining an outstanding performance.

2.2 Literature foundation for research paper contribution

The Generative Adversarial Networks (GANs) were first developed by Goodfellow et al. (2014). This complex neural network architecture has applications in speech, image and video generation, to name a few. GANs consist of two neural network architectures - Generator G and Discriminator D. The Generator G generates artificial data as close to the real samples, while the Discriminator

D tries to distinguish the artificial samples from the real samples. By training the two models against each other, the Generator G is able to reach a point where it generates artificial data that is indistinguishable from real samples, while the Discriminator D is not able to distinguish between real and artificial samples. As a result, the Generator G is able to model a complex high-dimensional distribution of real-world data. This capability is used to generate artificial convergent data to balance the dataset and improve Attribution model performance.

An alternative to using artificial data is to treat the convergent sequences as anomalies and attempt to deploy anomaly detection methods to detect convergent sequences. Zenati et al. (2018) propose an Adversarially Learned Anomaly Detection (ALAD) model based on Bi-directional GANs (BiGANs) (Donahue et al., 2017), that adversarially learns features for the anomaly detection task. The ALAD framework significantly improves anomaly detection performance from BiGANs, by ensuring for data-space and latent-space cycle-consistencies, and by stabilising the training. Zenati et al. (2018) validate this performance improvement on a range of image and tabular datasets. This research uses an adaptation of the ALAD framework to capture this capability while optimising for model training.

To assess the GAN induced performance improvement of the Attribution model, two-stage evaluation framework from Ren et al. (2018) is implemented on the test data. The first stage consists of comparing the theoretical performance via Area Under Curve (AUC), Log-loss and additional performance metrics described in Section 4.7. The second stage consists of comparing the budget allocation performance, which will be evaluated against the profit, conversion rate (CVR) and cost per conversion action (CPA). CVR is the ratio of converted sequences which reflects the ad placement effectiveness out of all the sequences that were funded. CPA is the ad placement cost averaged over all funded sequences which reflects the cost efficiency of ad campaigns.

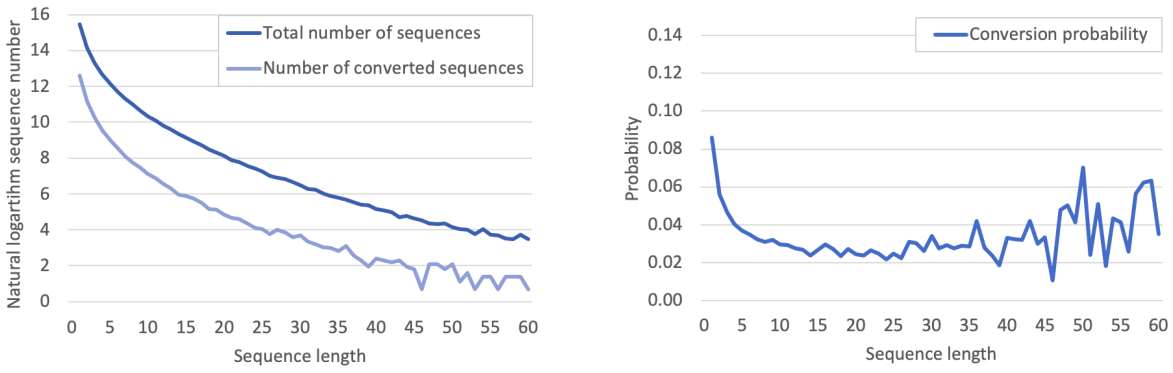
In summary, the contribution of this work is two-fold. First, a standard GAN model is deployed to generate artificial convergent sequence data, which is then used to support Attribution model training. Second, an Attribution model is implemented within the adapted ALAD framework with the purpose of treating conversion events as anomalies. Both of these approaches attempt to alleviate the unbalanced dataset problem that is a characteristic of the online digital campaign datasets.

3 Data

This section begins with an overview of the Criteo live traffic dataset used for attribution modelling, where variable description and dataset properties are given. Next, the data preprocessing methodology is explained.

3.1 Criteo dataset

The Criteo AI Lab is a pioneering company in computational advertising. They published a sub-sampled and anonymised dataset of live traffic data for Attribution modelling (Diemert Eustache, Meynet Julien et al., 2017). This dataset spans 30 days with 16.5 million impressions from 6.1 million users, 8.1 million impression sequences and 438 thousand conversions over 675 campaigns. Each impression corresponds to a banner that has been displayed to a user. Each banner has detailed information about the context, cost, if it was clicked and if it led to a conversion.



(a) Number of total and converted sequences w.r.t. user interaction sequence length

(b) Conversion probability w.r.t. user interaction sequence length

Figure 3.1: Conversion statistics w.r.t. sequence length

Figure 3.1 illustrates two important features of the Criteo dataset. First, sub-figure (a) shows that the longer sequence lengths have less samples, signifying that estimating conversion probabilities becomes harder with longer sequences. Second, sub-figure (b) shows that conversion probability decreases towards sequence lengths 15 to 25 and then increases in the Criteo dataset. These two observations may suggest that increased user exposure would lead to higher conversion probability, however, this assumption will not be utilised as this is not the focus of this research paper.

The Criteo live traffic dataset contains 22 variables. Each impression sequence can be identified by a unique user ID, conversion ID and campaign. Each sequence has a binary variable indicating

whether it concluded in a conversion. Each impression has information on campaign ID along with 9 categorical variables capturing contextual features associated to the display and the cost paid for the display. Additionally, the timestamp of the impression is given, which starts from 0 for the first impression, along with time since last click variable that is self-explanatory.

3.2 Data preprocessing

A user interaction sequence with ad content is uniquely identified by user ID, conversion ID and campaign. In this manner, each user interaction sequence is identified and sorted by timestamp in an ascending order, where first observation in a sequence is the most recent one. Next, sequences of minimal length of 20 and maximum length of 31 are extracted such that complex time-series relationships can be estimated. The sequence length is cut off at 31 due to longer sequences having less than 1000 samples for estimation. This results in 30,528 sequences, where 780 converge, i.e. the sample contains 2.56% conversions. Note that to replicate results from Ren et al. (2018), a separate sample is formed containing sequences of minimal length 5 and maximal length 20, where only a quarter of convergent sequences are used and non-convergent sequences are randomly under-sampled to obtain a 1:20 ratio of convergent to non-convergent sequences. All other steps remain the same. To estimate conversions, 12 features are used: impression timestamp, time since last click, campaign ID and nine contextual features. To format the data for model use, padding is used to ensure that each sequence has 31 time-steps.

The train and test sets are obtained by randomly stratifying on the conversion indicator, where 80% of the dataset is dedicated to the train set and the remaining 20% is dedicated to the test set. Thus, both sets contain only 2.56% of convergent sequences. In the case where a Generative Adversarial Network model is implemented, either the convergent or non-convergent sequences are removed from the train set to form an appropriate train set for the corresponding model objective. All models are evaluated on the same test set, where performance metrics are weighted to account for class imbalance.

To improve model training, batches are stratified on the conversion indicator within the train set, such that each batch contains roughly 2.56% of convergent sequences. As a result, it is expected that the gradient step taken within each batch is more optimal compared to when the conversion rates within different batches are inconsistent. When the train set is balanced by oversampling convergent sequences, each stratified batch is supplemented by randomly selected convergent sequences until the desired number of convergent sequences is reached.

4 Methodology

This section begins with an explanation of the Attribution model and the baseline used to evaluate model improvements from using artificial data generated by GANs. Subsequently, GAN, ALAD and streamlined adaptation of the ALAD model are explained. Next, two frameworks are discussed of how GANs are trained to generate artificial data, as well as how the adapted ALAD model is combined with the Attribution model. Finally, the model evaluation methodology is presented outlining the baseline used to evaluate model improvement via GANs, and outlining the theoretical and practical metrics deployed to compare the models.

4.1 Attribution model

The Attribution model implemented in this research paper is derived from the DARNN model in Ren et al. (2018). The Encoder for impression-level behavior modelling and the single Attention model of impression behavior is retained to estimate conversion probability. Let user u_i generate a behavior sequence $\{u_i, \{x_j\}_{j=1}^{k}, y_i\}$ of length k, where x_j is the impression feature vector and y_i indicates whether the user converts. The impression features x_j contain the impression timestamp, time since last click, campaign ID and 9 contextual features. The model has three main steps: capturing sequential user interaction with ad content, computing impression-to-conversion attention and estimating the final conversion probability. Figure 4.1 illustrates the Attribution model schema.

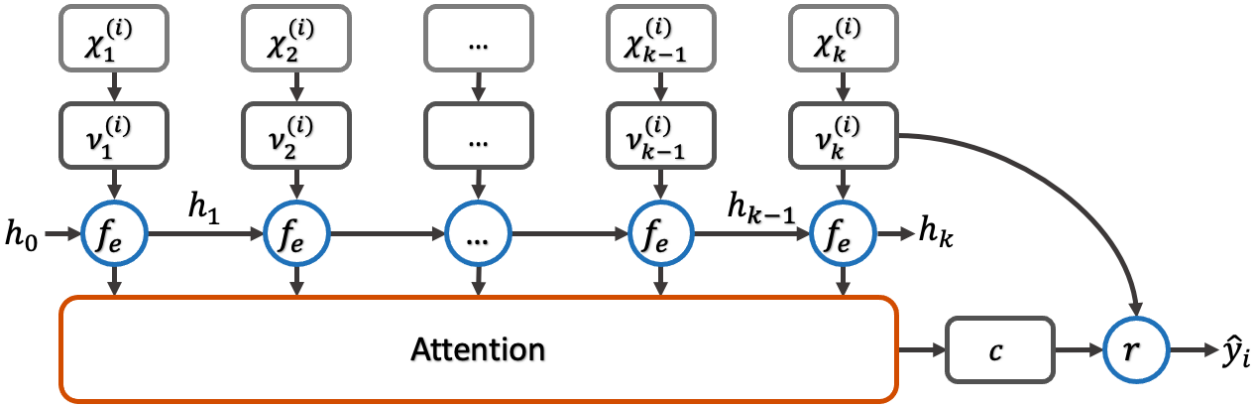


Figure 4.1: Attribution model ARNN schema

To capture sequential user interaction with ad content, a Recurrent Neural Network (RNN) is USED. First, each feature vector $x_j^{(i)}$ for user i is transformed into a dense representation vector $v_j^{(i)}$ to decrease the sparsity of original feature space. This has been implemented and shown to be ben-

eficial in many works, e.g. Ren et al. (2018) and He and Chua (2017). Next, the embedded features are fed into a standard LSTM network (Hochreiter and Schmidhuber, 1997) $f_e(v_j^{(i)}, h_{j-1}) = h_j$, where h_j is the hidden vector at time-step j.

To compute the impression-to-conversion attention, a unified energy-based function E is used - $e_j = E(h_j, v_{20})$. The Energy function E is an deep neural network with tanh activation function and it is implemented in the following manner

$$c = A(h_1, h_2, ..., h_{20}) = \sum_{j=1}^{20} a_j h_j \tag{4.1}$$

where

$$a_j = \frac{exp(e_j)}{\sum_{k=1}^{20} exp(e_k)} \tag{4.2}$$

In this manner, the learned parameter a_j is used to obtain the impression-to-conversion attention.

To estimate the final conversion probability $\hat{y}_i$, a deep neural network r with Sigmoid activation function is used. The neural network r takes as inputs the encoded feature vector $v_k^{(i)}$ and the impression-to-conversion attention c, i.e. $\hat{y}_i = p(y_i = 1|v^{(i)}) = r(v_k^i, c)$.

The loss function for the attribution problem is the binary cross entropy loss function, namely

$$L = \sum_{i=1}^{n} -y_i log(\hat{y}_i) - (1 - y_i) log(1 - \hat{y}_i) \tag{4.3}$$

4.2 Generative Adversarial Network

Generative Adversarial Networks (GANs) were first developed by Goodfellow et al. (2014) and they consist of two neural network architectures - Generator G and Discriminator D. The two network architectures are trained on unlabelled data $\{x^{(i)}\}_{i=1}^N$, where we denote the corresponding data space as $p_{\mathcal{X}}$.

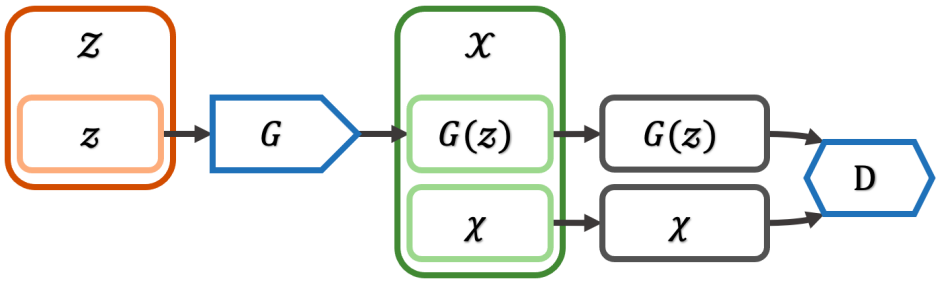


Figure 4.2: GAN schema

The goal of the Generator G is to master creating artificial data points identical to the realistic samples in a given data context. Formally, the Generator G learns to map random variables z drawn from the latent distribution $p_{\mathcal{Z}}$ - this can be the Gaussian distribution - to the input data space $p_{\mathcal{X}}$, where the mapping is denoted by $G(z)$. This can also be formulated as modelling the distribution $p_{\mathcal{X}}(x|z)$ using $p_G(x|z)$.

The goal of the Discriminator D is to distinguish between real and artificial samples in a given data context. Formally, the Discriminator D learns to distinguish between the training samples x, i.e. "real" data, and the artificial samples $G(z)$ produced by the Generator G from the random variable z, i.e. "fake" data.

Training of the two models is centered around the idea of iteratively improving the Generator G at fooling the Discriminator D and improving the Discriminator D at distinguishing between the "real" and "fake" samples. This is also referred to as the two-player mini-max game. Mathematical formulation of this objective function is $\min_G \max_D V(G, D)$, where $V(G, D)$ is defined below. The two deep neural networks G and D are optimised iteratively during the training procedure.

$$V(G, D) = \mathbb{E}_{x \sim p_{\mathcal{X}}}[log(D(x))] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[log(1 - D(G(z)))] \tag{4.4}$$

The optimal generator G^* will match the learned distribution $p_{G^*}(x|z)$ to the true distribution $p_{\mathcal{X}}(x|z)$, where the optimal discriminator $D^* = \frac{p_{\mathcal{X}}(x)}{p_{\mathcal{X}}(x)+p_G(x)}$. The corresponding proofs of the optimal solutions for G^* and D^* can be found in Goodfellow et al. (2014).

4.3 Adversarially Learned Anomaly Detection

The Adversarially Learned Anomaly Detection (ALAD) was proposed by Zenati et al. (2018). The ALAD architecture elaborates on the Bidirectional Generative Adversarial Networks (BiGAN) developed by Donahue et al. (2017), which are an extension of the standard GAN architecture discussed in Section 4.2.

BiGANs extend the ordinary GANs by including an Encoder E that maps the data x to the random variable z that belongs to the latent distribution $p_{\mathcal{Z}}$ and train the Discriminator D_{xz} to jointly discriminate between the "real" tuples $(x, E(x))$ and "fake" tuples $(G(z), z)$. Thereby, BiGANs try to match the distribution $p_E(x, z) = p_{\mathcal{X}}(x)p_E(z|x)$ to $p_G(x, z) = p_{\mathcal{Z}}(z)p_G(x|z)$. In this manner, the Encoder E tries to learn to invert the Generator G. This is accomplished through the optimisation of the objective function $\min_{G,E} \max_{D_{xz}} V(G, E, D_{xz})$, where $V(G, E, D_{xz})$ is defined below.

$$V(G, E, D_{xz}) = \mathbb{E}_{x \sim p_{\mathcal{X}}}[log(D_{xz}(x, E(z)))] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[log(1 - D_{xz}(G(z), z))] \tag{4.5}$$

As shown in Donahue et al. (2017), the optimal discriminator D^*_{xz} is $D^*_{xz} = \frac{p_E(x,z)}{p_E(x,z)+p_G(x,z)}$, which is similar to the optimal solution in Section 4.2. Zenati et al. (2018) argue that in practice, the joint distributions $p_E(x, z)$ and $p_G(x, z)$ will not be the identical due to the fact that training does not always converge to the solutions of the two-player mini-max game.

To solve this problem, they utilise the ALICE framework from Li et al. (2017), which approximates the conditional entropy $H^\pi(x|z) = -\mathbb{E}_{\pi(x,z)}[log(\pi(x|z))]$ using an additional Discriminator D_{xx}. Li et al. (2017) show that D_{xx} will enforce cycle-consistency in the model, i.e. $G(E(x)) \approx x$. The new objective function $\min_{G,E} \max_{D_{xz}} V_{ALICE}(G, E, D_{xx}, D_{xz})$ is defined as

$$V_{ALICE}(G, E, D_{xx}, D_{xz}) = V(G, E, D_{xz}) + V(G, E, D_{xx}) \tag{4.6}$$

where

$$V(G, E, D_{xx}) = E_{x \sim p_{\mathcal{X}}}[log(D_{xx}(x, x))] + E_{x \sim p_{\mathcal{X}}}[log(1 - D_{xx}(x, G(E(x))))] \tag{4.7}$$

To stabilize the training of the baseline ALICE model, Zenati et al. (2018) further regularize latent space conditional $H^\pi(z|x) = -\mathbb{E}_{\pi(x,z)}[log(\pi(z|x))]$ using an additional Discriminator D_{zz}. The new objective function $\min_{G,E} \max_{D_{zz}} V(G, E, D_{zz})$ is defined as

$$V(G, E, D_{zz}) = \mathbb{E}_{z \sim p_{\mathcal{Z}}}[log(D_{zz}(z, z))] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[log(1 - D_{zz}(z, E(G(z))))] \tag{4.8}$$

Combining the three two-player mini-max games, the ALAD architecture solves the objective function $\min_{G,E} \max_{D_{xx},D_{xz},D_{zz}} V(G, E, D_{xx}, D_{xz}, D_{zz})$, where

$$\min_{G,E} \max_{D_{xx},D_{xz},D_{zz}} V(G, E, D_{xx}, D_{xz}, D_{zz}) = \min_{G,E} \max_{D_{xz},D_{xx},D_{zz}} V(G, E, D_{xx}) + V(G, E, D_{xz}) + V(G, E, D_{zz}) \tag{4.9}$$

The schematic representation of this architecture is displayed in Figure 4.3. In blue are the Discriminators D_{xx}, D_{xz} and D_{zz}, the Generator G and the Encoder E. In grey are the tuples that are fed into each of the Discriminators. In darker shade of orange is the latent distribution space $p_{\mathcal{Z}}$ and in darker shade of green is the feature space $p_{\mathcal{X}}$.

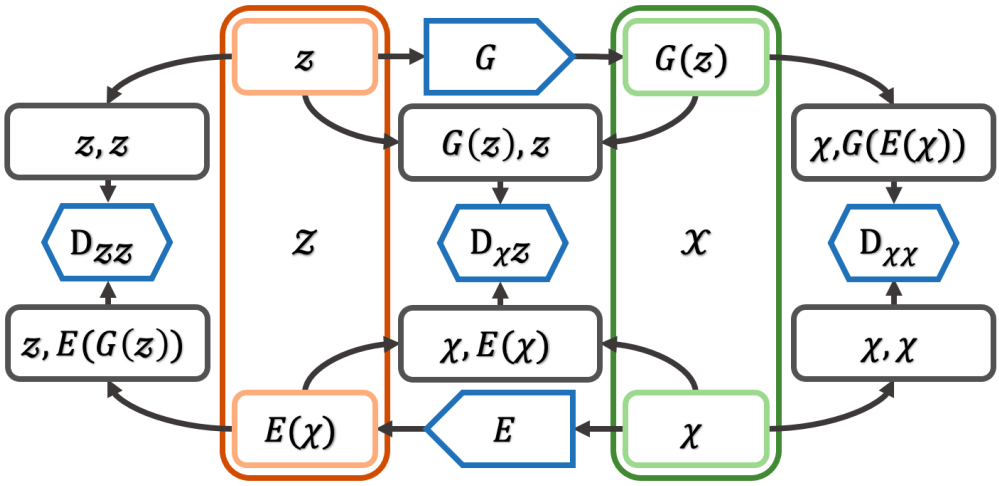


Figure 4.3: ALAD schema

The final component of the ALAD framework is the Anomaly Score $A(x)$ (Zenati et al., 2019). This score is a linear combination of a reconstruction loss $L_G(x)$ and a discriminator-based loss $L_D(x)$. Their relative weights are tuned through the hyperparameter α.

$$A(x) = \alpha L_G(x) + (1 - \alpha) L_D(x) \tag{4.10}$$

where $L_G(x) = ||x - G(E(x))||_1$ is the distance between the original samples and their reconstructions and $L_D(x)$ can be defined in two ways. First, $L_D(x) = ||f_D(x, E(x)) - f_D(G(E(x)), E(x))||_1$, where $f_D(\cdot)$ is the layer preceding to the final discriminator D_{xz} output. Second, $L_D(x) = \sigma(D_{xz}(x, E(x)), 1)$, which is the cross-entropy loss that captures the confidence of the discriminator D_{xz} that the sample comes from the "real" distribution $p_{\mathcal{X}}$. Zenati et al. (2018) showed that the first approach of defining $L_D(x)$ performs better.

4.4 Adjusted Adversarially Learned Anomaly Detection

The Adjusted Adversarially Learned Anomaly Detection (AALAD) is a simplified adaptation of the ALAD model from Zenati et al. (2018). The simplification is introduced to optimise the training process of the model by streamlining the architecture.

The ALAD objective function is defined as $\min_{G,E} \max_{D_{xx}, D_{xz}, D_{zz}} V(G, E, D_{xx}, D_{xz}, D_{zz})$, where each component is a neural network. The AALAD architecture removes the two neural networks D_{xx} and D_{zz}, and replaces them with a Mean Squared Error loss objective, where $E(G(z))$ and $G(E(x))$ are evaluated against z and x, respectively, i.e. $\mathbb{E}[(E(G(z)) - z)]^2 + \mathbb{E}[(G(E(x)) - x)]^2$. Hence, the

objective function is

$$\min_{G,E} \max_{D_{xz}} V(G, E, D_{xz}) + V(G, E) \tag{4.11}$$

where

$$V(G, E, D_{xz}) = \mathbb{E}_{x \sim p_{\mathcal{X}}}[log(D_{xz}(x, E(z)))] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[log(1 - D_{xz}(G(z), z))] \tag{4.12}$$

and

$$V(G, E) = \mathbb{E}[(E(G(z)) - z)]^2 + \mathbb{E}[(G(E(x)) - x)]^2 \tag{4.13}$$

The schematic representation of this architecture is displayed in Figure 4.4. In blue is the Discriminator D_{xz}, the Generator G and the Encoder E. In grey are the tuples that are fed into each of the Discriminators, where on each side, the $E(G(z))$ and $G(E(x))$ outputs are mapped to z and x, respectively. In darker shade of orange is the latent distribution space $p_{\mathcal{Z}}$ and in darker shade of green is the feature space $p_{\mathcal{X}}$.

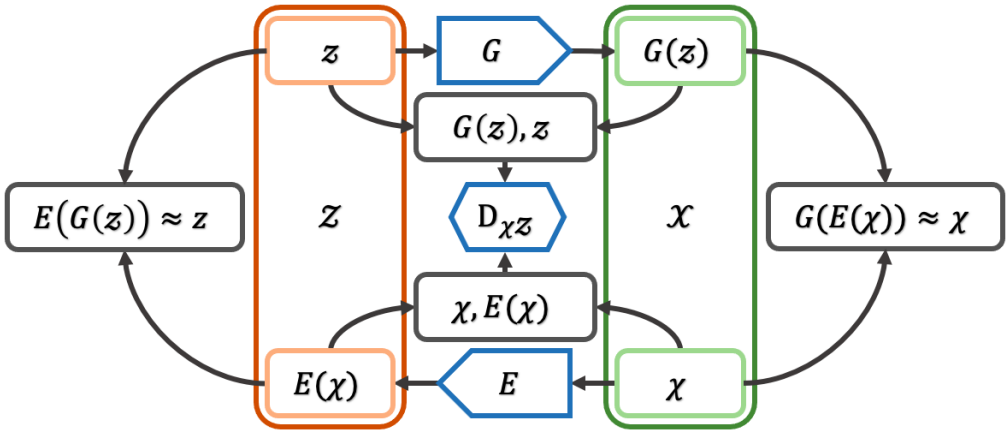


Figure 4.4: AALAD schema

The AALAD framework employs the Anomaly Score $A(x)$, which remains the same as that in (Zenati et al., 2019). This score is a linear combination of a reconstruction loss $L_G(x)$ and a discriminator-based loss $L_D(x)$, where their relative weights are tuned through the hyperparameter α.

$$A(x) = \alpha L_G(x) + (1 - \alpha) L_D(x) \tag{4.14}$$

4.5 Generative Adversarial Network supported Attribution

GAN supported Attribution model is a three-step method combining the Attribution model from Section 4.1 and the GAN framework from Section 4.2 to alleviate the unbalanced dataset problem. First, the Attribution model from Section 4.1 is used in place of the Discriminator D in the GAN framework presented in Section 4.2, where the Generator G is trained to model $p_{\mathcal{X}}(x|z)$ using $p_G(x|z)$ with the input distribution space $p_{\mathcal{X}}$ representing the distribution of convergent sequences. Second, the trained Generator G is used to produce a given number of additional convergent training sequences to balance the attribution dataset. Third, a new Attribution model is trained using the original training data combined with the artificial data generated by the trained Generator G. In this manner, the final Attribution model is trained in a more balanced data setting.

4.6 Attribution via Adjusted Adversarially Learned Anomaly Detection

Attribution via AALAD is a model combining the the Attribution model from Section 4.1 and the AALAD framework from Section 4.3. The joint model treats the conversion events as outliers, thereby attempting to alleviate the unbalanced data setting and improve the conversion events prediction. The model has the objective function of the AALAD architecture, namely, $\min_{G,E} \max_{D_{xx},D_{xz},D_{zz}} V(G, E, D_{xx}, D_{xz}, D_{zz})$, where D_{xz} takes form of the Attribution model. In this data setting, the non-convergent sequences form the input space $p_{\mathcal{X}}$, i.e. the "real" data, while the convergent sequences are regarded as the "fake" data that will are generated by the generator G. In this manner, the conversion events are determined using the anomaly score $A(x)$ (Zenati et al., 2019). This score is a linear combination of a reconstruction loss $L_G(x)$ and a discriminator-based loss $L_D(x)$, where their relative weights are tuned through the hyperparameter α, as discussed in Section 4.3. Since the model does not use real conversion sequences to train, it loses its capability of computing the impression-to-conversion attention used for budget allocation.

4.7 Model evaluation

The model evaluation framework is two-fold - theoretical and practical. To evaluate theoretical performance of the models, Log-loss, AUC, Recall, Precision, Accuracy, F-measure metrics are computed. To evaluate the practical performance of the models, the budget-allocation framework from Ren et al. (2018) is used.

Log-loss measures the confidence of the classification of a binary outcome via the function

$L = y \cdot log(P(y = 1|X)) + (1 - y) \cdot log(1 - P(y = 1|X))$. The farther the prediction is from its true value, the higher the penalty.

Recall, Precision and F-measure are metrics derived from the confusion matrix depicted in Figure 4.5, where higher metric score means better model performance. Recall is defined as $\frac{TP}{TP+FN}$ and conveys how many observations from the Positive class were classified correctly, otherwise known as the True Positive Rate. Precision is defined as $\frac{TP}{TP+FP}$ and conveys how many classes were classified as Positive and are in fact Positive. F-measure is defined as $\frac{2*Recall*Precision}{Recall+Precision}$ and attempts to capture Recall and Precision in one value. Accuracy is defined as $\frac{TP+TN}{TP+FP+TN+FN}$ and conveys the proportion of correctly classified observations.

		True values	
		Positive	Negative
True values	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Figure 4.5: Confusion matrix

Receiver Operator Characteristic (ROC) curve is used to evaluate binary classification models. It is a probability curve depicted in Figure 4.6, that plots True Positive Rate (*Recall*) against the False Positive Rate.

(a) $AUC = 0$ (b) $AUC = 0.5$ (c) $AUC = 1$

Figure 4.6: ROC curves with corresponding AUC metrics

The Area Under the Curve (AUC) summarises the ROC curve. It is the measure of the ability of a classifier to distinguish between classes. If $AUC = 1$, the classifier is able to distinguish between all classes correctly. If $AUC = 0$, the classifier miss-classifies all points, i.e. classified all Positive class points as Negative and vice versa. If $AUC = 0.5$, the classifier is not able to distinguish between Positive and Negative class points, i.e. the classifier predicts randomly or classifies all

points to one class.

To illustrate the practical performance of the models, two-stage budget allocation framework is deployed from Ren et al. (2018). It is important to note that the Criteo dataset is not fit to test budgeting frameworks that would be implemented in a business setting, which is why the proposed methodology is used. This framework consists of computing the budget allocation based on Return on Investment (ROI) from the training set and then evaluating the corresponding budget allocation on the test set. In particular, the budget B must be allocated across K campaigns $c_1, c_2, ..., c_K$.

$$ROI_{c_k} = \frac{\sum_{\forall y_i=1} Attr(c_k|y_i)y_i}{Money\ spent\ on\ campaign\ c_k} \tag{4.15}$$

with

$$Attr(c_k|y_i) = \sum_{j=1}^{20} a_j \cdot I(c_j = c_k) \tag{4.16}$$

where ROI_{c_k} is the total credit attributed to campaign c_k, $I(\cdot)$ is an indicator function and y_i is the conversion indicator. Next, the total budget B is allocated across each campaign such that each campaign receives budget $b_1, b_2, ..., b_K$, computed as

$$b_k = \frac{ROI_{c_k}}{\sum_{j=1}^{K} ROI_{c_j}} \cdot B \tag{4.17}$$

To test the budget allocation, each user interaction sequence in the test set is aggregated to create tuples $\{s_i, t_i, o_i, y_i, c_i\}_{i=1}^{n}$, where n is the total number of tuples, s_i is the sequence identifier, t_i is the serving time, o_i is the total cost of the sequence, y_i is the conversion indicator and c_i indicates the campaigns that the sequence is on. Then, Algorithm 1 is ran over the test set to test the budget allocation for each model.

Algorithm 1 Budget allocation evaluation

Input: Shuffled sequence tuples $\{s_i, t_i, o_i, y_i, c_i\}$ and budget allocations $b_1, b_2, ..., b_K$.

Output: Conversions Y and cost C.

$Y = 0$, $C = 0$
for $i = 1$ to n **do**
  if $b_{c_i} \geq c_i$ **then**
    $Y = Y + y_i$
    $C = C + c_i$,
    $b_{c_i} = b_{c_i} - c_i$
  else
    $C = C + b_{c_i}$
    $b_{c_i} = 0$
  end if
end for
return Y, C

5 Results

This section begins with a description of the final models and their training framework. Next, Attribution model results are given for the set-up that is described in Ren et al. (2018) to verify that the Attribution model in this research paper has similar performance. Finally, theoretical and practical results are discussed to answer the research question and its sub-questions from Section 1.

5.1 Final models

The Attribution model is trained on the full training dataset using the BCE loss. To improve model training, batches are stratified on sequence conversion indicator such that the fraction of conversions is consistent across batches. The GAN model used to generate artificial data is trained on a subset of the training dataset containing only convergent sequences. To solve for training issues that are common across GAN literature, Wasserstein loss with Gradient Penalty (Arjovsky et al., 2017) is used. The AALAD model is trained on the training dataset with non-convergent sequences only. The Generator and Discriminator networks are optimised using Wasserstein loss with Gradient Penalty (Arjovsky et al., 2017), while the Encoder is optimised using MSE loss.

To show the benefit of using artificial data generated by the GAN model, an additional Attribution model is trained by sampling convergent sequences with replacement in the training dataset.

This is done to match the ratio of convergent sequences in the model training phase with that in the GAN case. To be precise, the GAN and the baseline Attribution models will append convergent sequences to make them account for 10%, 30% and 50% of the training dataset. Finally, to make the metrics comparable and to account for the unbalanced dataset, Recall, Precision, F-measure and AUC are computed weighted by the respective class size. Due to the fact that Recall equals Accuracy when weighted, it will not be reported.

5.2 Replicating Ren et al. (2018) results

In this section, the replication results are presented and compared to those in Ren et al. (2018) to verify that the ARNN model in this research paper has similar performance. The difference in the dataset in Ren et al. (2018) is the ratio of 1:20 of convergent to non-convergent sequences. The theoretical and budget allocation capabilities are compared below.

Log-loss	**AUC**	**Precision**	**Accuracy**	**F-measure**
0.078	0.930	0.966	0.906	0.926

Table 5.1: Theoretical results on ARNN model

Table 5.1 shows the ARNN model results from this paper. The Log-loss is considerably smaller in this paper compared to the Log-loss of 0.185 in Ren et al. (2018). The AUC is inferior to that in Ren et al. (2018), namely, 0.930 vs 0.979. Hence, the ARNN model gives predictions that are generally closer to 0 or 1 with slightly worse ability to distinguish between the two classes correctly. Remaining metrics cannot be compared since Ren et al. (2018) do not report them.

Figure 5.1 clearly displays that with increased budget, the CPA, conversion number and CVR increase. This contrasts with results from Ren et al. (2018), where the CVR decreases as the budget increases. The CVR is below 0.048, meaning that the ARNN model is not able to allocated budget to reach higher conversion rate than by funding all the campaigns in the test set. On the other hand, the CPA is lower than if all the campaigns were funded.

To conclude, the ARNN model in this paper has close performance to that in the paper Ren et al. (2018), from a theoretical standpoint but under-performs in the budget allocation task.

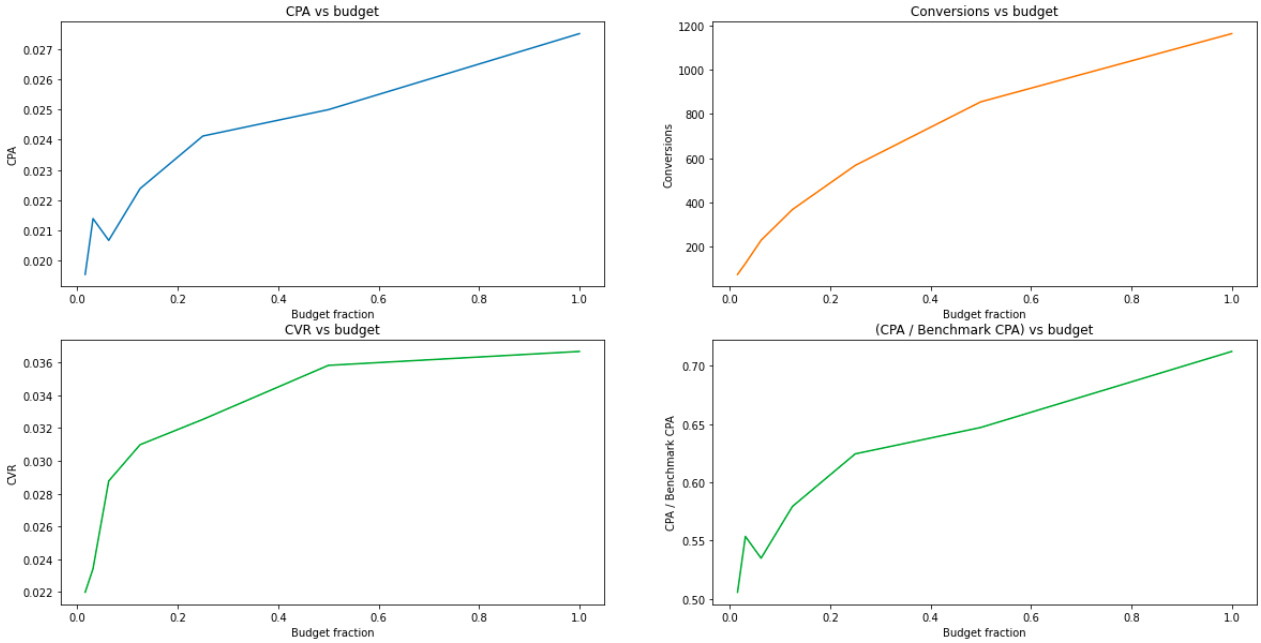


Figure 5.1: Budget Allocation Results for ARNN model

5.3 Theoretical results

This section begins with a discussion of the baseline ARNN and the GAN supported ARNN models. It then continues onto a comparison of the top-line models.

The results for the baseline ARNN and the GAN supported ARNN models are mixed. Focusing on the Log-loss and AUC metrics, the baseline ARNN model decreases in performance as the fraction of conversions increases. The opposite is true for the GAN supported ARNN model. The picture for Precision, Accuracy and F-measure is more complex. For the baseline ARNN model, the three measures are highest when the fraction of conversions is at 30%. In the case of the GAN supported ARNN model, Precision and F-measure are the highest when fraction of conversions is at 10% while Accuracy peaks at 30% and 50% of conversions. The top models in each case are the baseline ARNN with 10% of conversions and the GAN supported ARNN with 50% of conversions because these achieve the lowest Log-loss and highest AUC in their respective groups.

Model	Log-loss	AUC	Precision	Accuracy	F-measure
Baseline ARNN (10%)	0.091	0.895	0.979	0.959	0.967
Baseline ARNN (30%)	0.138	0.877	0.980	0.967	0.972
Baseline ARNN (50%)	0.195	0.857	0.976	0.953	0.962
GAN ARNN (10%)	0.062	0.908	0.917	0.977	0.940
GAN ARNN (30%)	0.062	0.932	0.905	0.979	0.933
GAN ARNN (50%)	0.058	0.937	0.914	0.979	0.939

Table 5.2: Theoretical results on ARNN baseline and GAN supported ARNN models

The ARNN model in Table 5.3 achieves a relatively high AUC score and a low Log-loss, in comparison with the replication case in Section 5.2. The Log-loss and Precision measures are superior while the AUC, Accuracy and F-measure are inferior. The baseline ARNN generally under-performs in Log-loss and AUC performance metrics. The GAN supported ARNN model manages to outperform all models in Log-loss and Precision, achieving the highest AUC in comparison to the rest of the models in the table, but not reaching the performance level of the model in Ren et al. (2018). The AALAD model substantially under-performs across all metrics.

Model	Log-loss	AUC	Precision	Accuracy	F-measure
ARNN	0.059	0.919	0.978	0.909	0.936
Baseline ARNN (10%)	0.091	0.895	0.979	0.959	0.967
GAN ARNN (50%)	0.058	0.937	0.979	0.914	0.939
AALAD	7.764	0.600	0.960	0.513	0.655

Table 5.3: Theoretical results on top-line models

In summary, the GAN supported ARNN model outperforms all other models in terms of Log-loss, AUC and precision. This model was capable of outperforming the standard ARNN model by all measures, and most importantly Log-loss and AUC. The increase of AUC by 0.018 is impressive due to the fact that the ARNN model AUC was already very high at 0.919. Both Log-loss and AUC are also superior to those in the replication case in Section 5.2. Thus, this approach shows a high potential in improving model performance by expanding the representation of the minority population in an unbalanced dataset.

5.4 Practical results

In this section, the budget allocation results are presented. These are summarised in four plots displaying CPA, number of achieved conversions, CVR and CPA divided by the benchmark CPA (total cost / all campaigns) w.r.t. budget as a fraction of the total budget (1, 1/2, 1/4, 1/8, 1/16, 1/32 and 1/64). Furthermore, the budget allocation results are discussed only for the top-line models from Table 5.3, with the AALAD model being left out. The remaining plots can be found in Appendix A. Note that the entire budget being allocated does not mean that all of it is used. Instead, it means that the budget is allocated across channels that may not use all of it, thereby leaving the budget unemployed.

The ARNN model budget allocation has a CPA that peaks at a quarter of the budget. Noticeably, the CPA sharply falls to about 0.32 when half the budget is allocated. The number of conversions and the CVR steadily increase with the budget, however, the conversion rate only reaches 0.018, when budget allocations are depleted. Thus, the conversion rate and the CPA are about 30% and 35% smaller than if all the campaigns in the test set are funded, respectively.

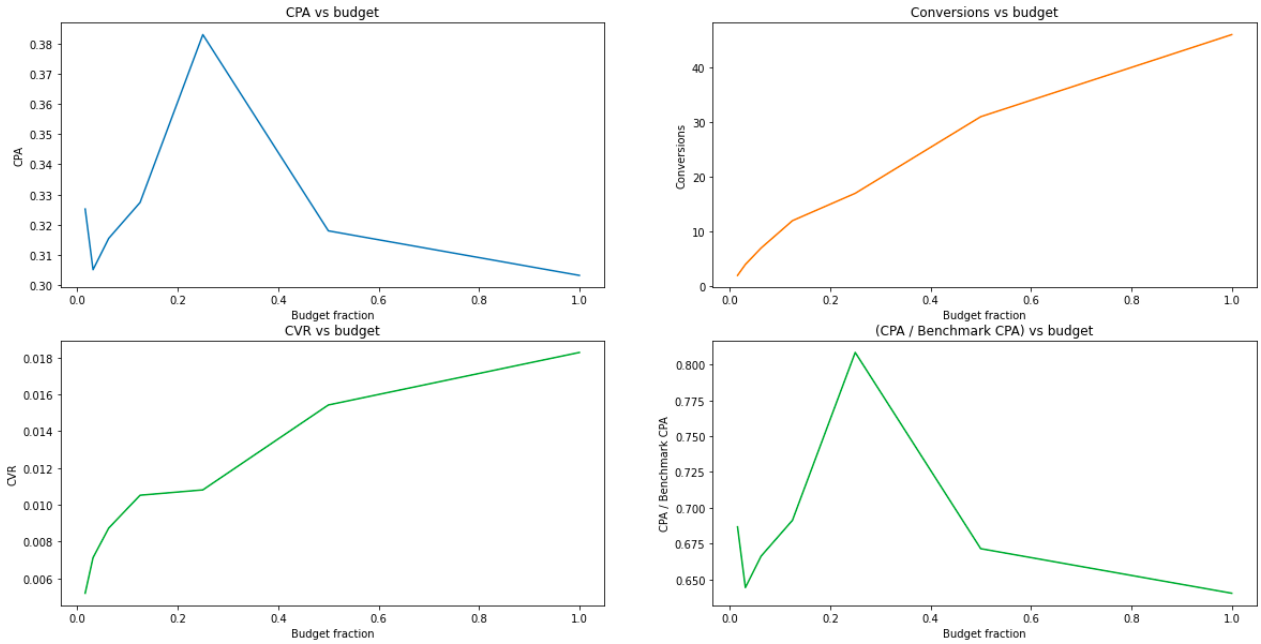


Figure 5.2: Budget Allocation Results for ARNN model

The ARNN baseline budget allocation displays very similar patterns to that of the ARNN model. The CPA peaks at a quarter of the budget with the conversions almost linearly falling to a minimum as more budget is used. The CVR increases with the budget at a similar pace to that

of the ARNN model. Nonetheless, the conversion rate only peaks at around 0.018, which puts the best performance at an equal level to that of the ARNN model.

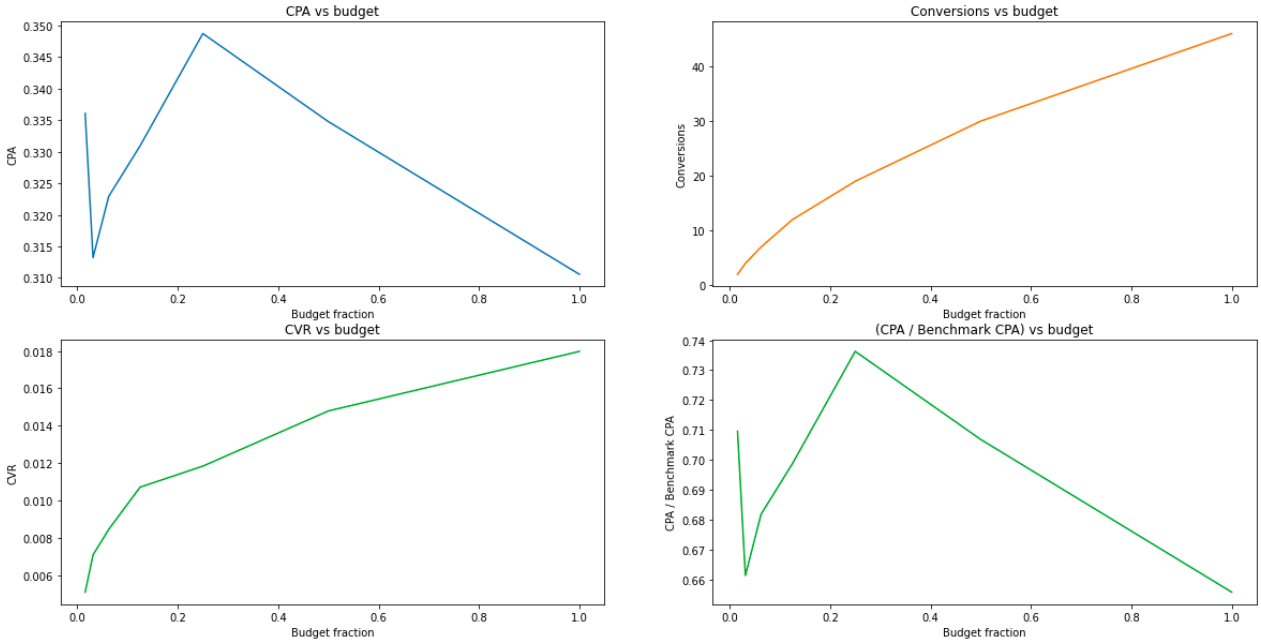


Figure 5.3: Budget Allocation Results for ARNN baseline with 10% conversions

The GAN supported ARNN model budget allocation produces a CPA that largely varies in the budgets smaller than a quarter of the full budget. The CVR increases with higher budget and reaches similar levels to that of the ARNN and baseline ARNN models. Hence, the budget allocation is still incapable of outperforming the test set CVR of 0.0256.

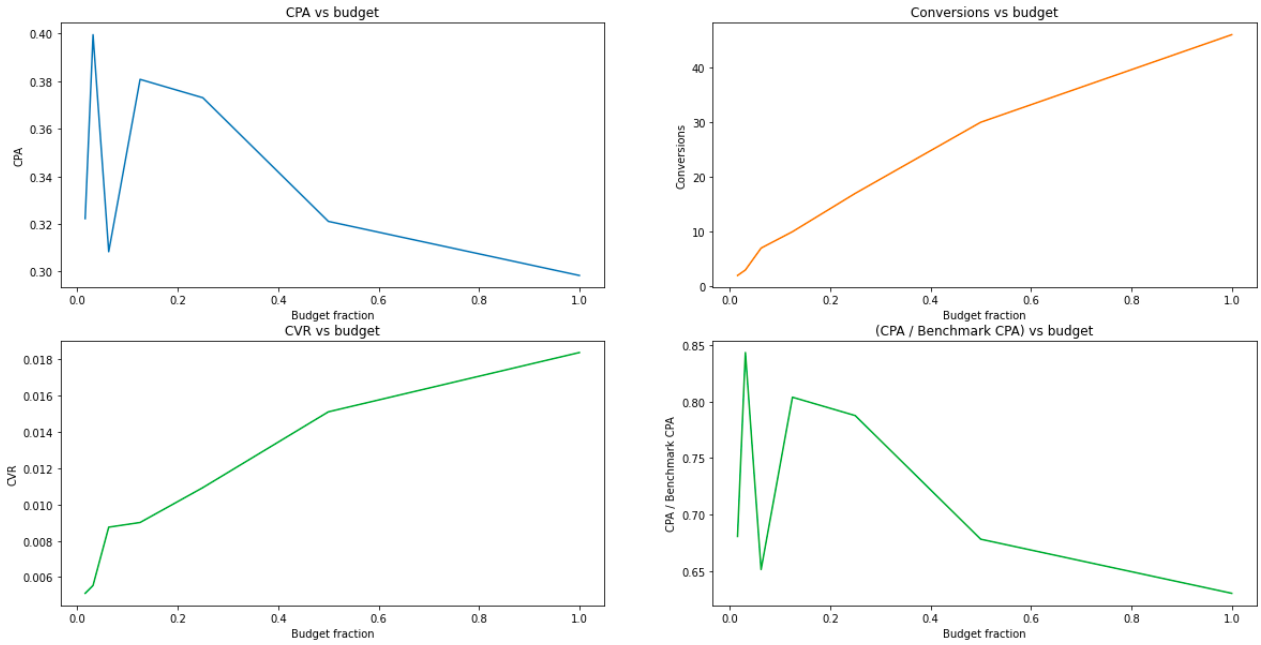


Figure 5.4: Budget Allocation Results for GAN supported ARNN model with 50% conversions

Ultimately, the budget allocations are very similar across the three models. In every case, the best CVR is achieved with all of the budget being allocated across channels. This also drives down the CPA which is at about 35% lower than the baseline but with a roughly 30% lower CVR.

6 Conclusion

This research paper aims to answer the research question *To what extent can Generative Adversarial Networks alleviate the unbalanced dataset problem and improve the Attribution modelling of the online digital campaign data?* supplemented by three research sub-question focusing on the artificial data generation and anomaly detection to solve for the unbalanced dataset problem, and budget allocation to take on the Attribution modelling challenge. To tackle all of the above, two models were developed. The first model is the GAN supported ARNN model motivated by Ren et al. (2018) and Goodfellow et al. (2014), which aims to improve the ARNN model performance by modelling the conversion sequence distribution and then generating artificial conversion sequences to balance the dataset on which a standard ARNN model is then trained. The second model is the AALAD model motivated by Zenati et al. (2018), which treats conversions as anomalies and tries to distinguish between the non-convergent sequences and their anomalous counterparts - convergent sequences.

The GAN supported ARNN model was able to outperform the ARNN model trained on the original training set, the baseline ARNN and the AALAD models. It was able to do so with superior Log-loss and AUC versus all other models. Furthermore, it produced more accurate estimates than the ARNN model, while the top accuracy was obtained by the baseline ARNN model. These results show that GANs can be successfully used to generate artificial data to improve the theoretical performance of the Attribution model. Beyond that, this result suggests that this approach can prove to be successful in other applications outperforming under-sampling and over-sampling approaches.

The AALAD model was implemented to test whether treating the Attribution challenge as an anomaly detection problem would be successful. The results show that this approach has not worked at all, under-performing all other models by a significant margin in Log-loss, AUC, Precision, Accuracy and F-measure. This may have happened due to several reasons. One of the causes may be the structure of the Generator, which was designed to approximately invert the ARNN model used as the Discriminator. Further, the AALAD model may not have been able to converge to an optimal solution. Last but not least, the Encoder may have not been able to properly invert the Generator. While all of the above may have been the cause, this research suggests that GANs for Anomaly Detection cannot improve the Attribution model performance.

With regards to the practical performance, the ARNN, baseline ARNN and GAN supported ARNN models all performed very similarly. The best CPA and CVR were achieved when the entire budget was allocated towards marketing campaigns, yet the CVR still remained lower than the CVR of the test set. This is also seen in the paper by Ren et al. (2018). However, as mentioned in the Methodology, the Criteo dataset is not fit to test budgeting frameworks that would be implemented in real life, which is why the approach from Ren et al. (2018) is used. Hence, it makes sense why the approach is not successful.

To conclude, the results of this research suggest that the Generative Adversarial Networks can be successfully used to generate artificial data that can supplement model training and alleviate the unbalanced dataset problem. However, the results also indicate that the Generative Adversarial Network architecture is not able to improve Attribution modelling of the digital campaign data.

One of the key areas to explore in future research is comparing Generative Adversarial Networks to other data augmentation techniques. For example, Synthetic Minority Oversampling Technique (SMOTE) is a data augmentation method invented by Chawla et al. (2002) that works by interpolating new data points based on neighboring minority class samples. Chawla et al. (2002) show that this approach improves the AUC measure across all experimental datasets in their research

paper. On the other hand, Blagus and Lusa (2013) find that SMOTE is mostly beneficial in low-dimensional data settings. Thus, it would be insightful to learn where the Generative Adversarial Networks take lead and where other data augmentation techniques triumph.

Another key area that should be explored is related to the fact that Neural Network-based methods require a thorough tuning process to arrive at a model specification that is close to optimal. The most evident example for this is that the ARNN model was able to perform close to the results of Ren et al. (2018) but with a noticeable difference. And while the GAN supported ARNN model outperformed all other candidates, the AALAD model far under-performed all the models used in this research paper and was not able to adequately fit the non-convergent dataset to classify the convergent sequences as anomalous. This shortcoming took place due to computational and time constraints. Thus, dedicating more computing power and time to model specifications can yield superior models and results, where most importantly the AALAD model could prove to be effective. Alternatively, an attempt to develop or adapt a different anomaly detection technique could prove to be successful at detecting convergent sequences.

References

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan.

Blagus, R. and Lusa, L. (2013). Smote for high-dimensional class-imbalanced data. *BMC bioinformatics*, 14:106.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Diemert Eustache, Meynet Julien, Galland, P., and Lefortier, D. (2017). Attribution modeling increases efficiency of bidding in display advertising. In *Proceedings of the AdKDD and TargetAd Workshop, KDD, Halifax, NS, Canada, August, 14, 2017*, page To appear. ACM.

Donahue, J., Krähenbühl, P., and Darrell, T. (2017). Adversarial feature learning.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks.

He, X. and Chua, T.-S. (2017). Neural factorization machines for sparse predictive analytics.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Ji, W. and Wang, X. (2017). Additional multi-touch attribution for online advertising. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 1360–366. AAAI Press.

Li, C., Liu, H., Chen, C., Pu, Y., Chen, L., Henao, R., and Carin, L. (2017). Alice: Towards understanding adversarial learning for joint distribution matching. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Mohsin, M. (2020). 10 ecommerce trends that you need to know in 2021 [infographic].

Ren, K., Fang, Y., Zhang, W., Liu, S., Li, J., Zhang, Y., Yu, Y., and Wang, J. (2018). Learning multi-touch conversion attribution with dual-attention mechanisms for online advertising. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.*

Shao, X. and Li, L. (2011). Data-driven multi-touch attribution models. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, page 258–264, New York, NY, USA. Association for Computing Machinery.

Wang, J., Zhang, W., and Yuan, S. (2017). Display advertising with real-time bidding (rtb) and behavioural targeting.

Zenati, H., Foo, C. S., Lecouat, B., Manek, G., and Chandrasekhar, V. R. (2019). Efficient gan-based anomaly detection.

Zenati, H., Romain, M., Foo, C. S., Lecouat, B., and Chandrasekhar, V. R. (2018). Adversarially learned anomaly detection.

Zhang, Y., Wei, Y., and Ren, J. (2014). Multi-touch attribution in online advertising with survival theory. In *Proceedings of the 2014 IEEE International Conference on Data Mining*, ICDM '14, page 687–696, USA. IEEE Computer Society.

Appendix A Budget allocation results

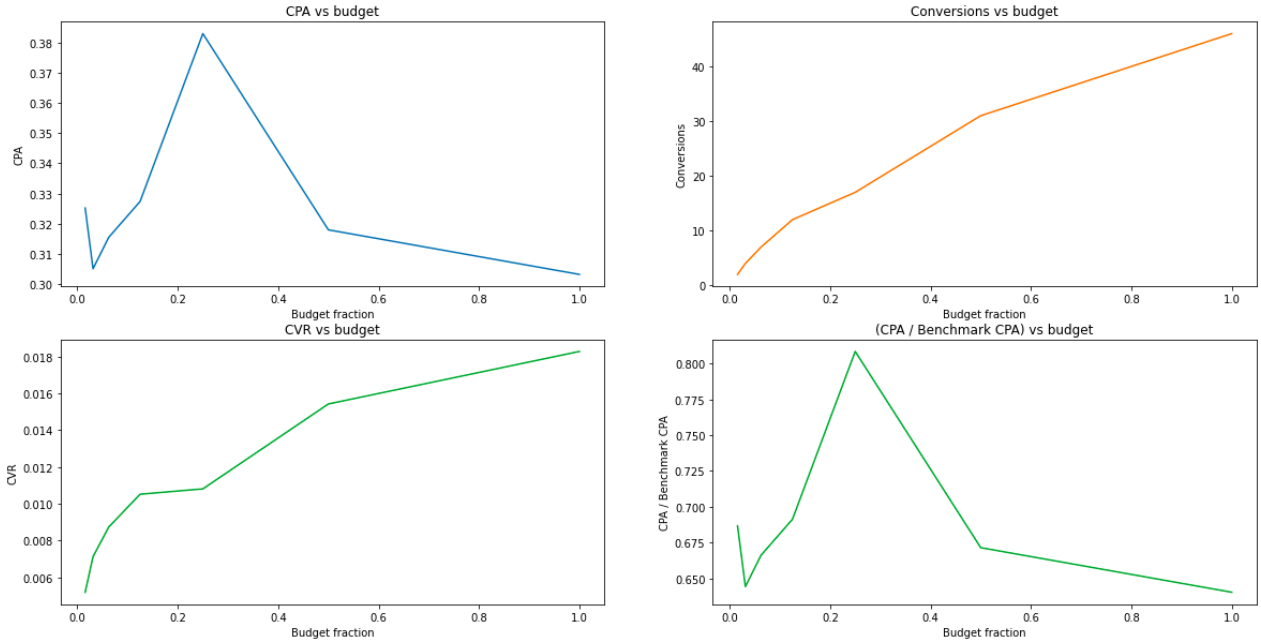


Figure A.1: Budget Allocation Results for ARNN model

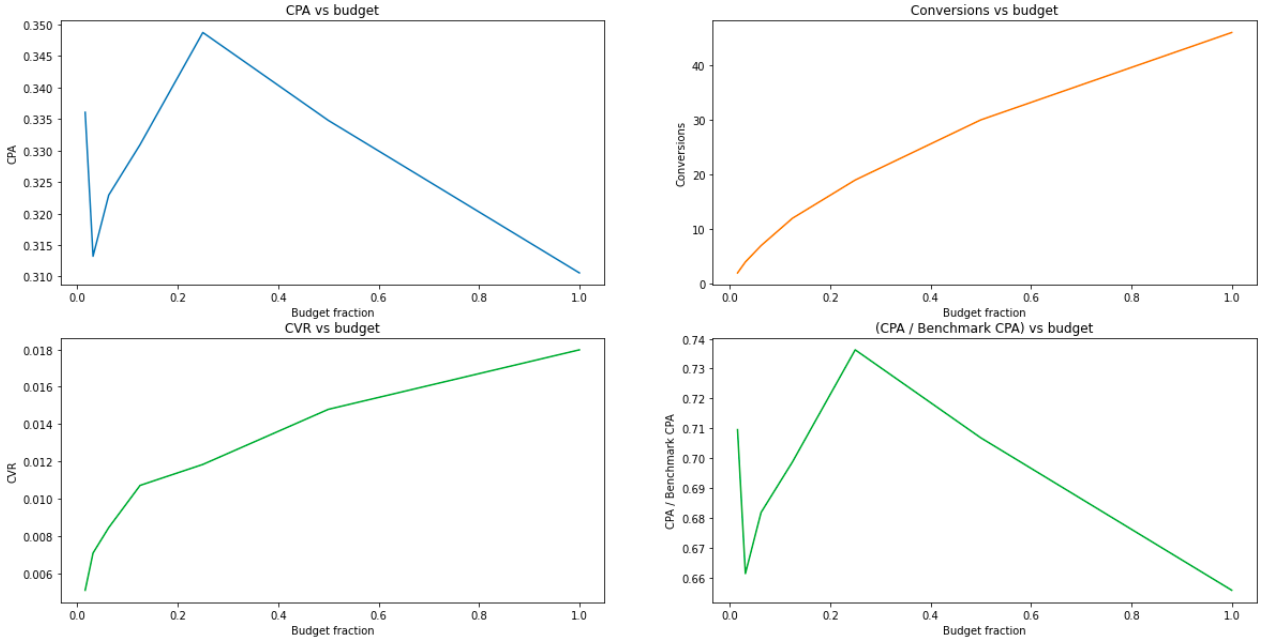


Figure A.2: Budget Allocation Results for ARNN baseline with 10% conversions

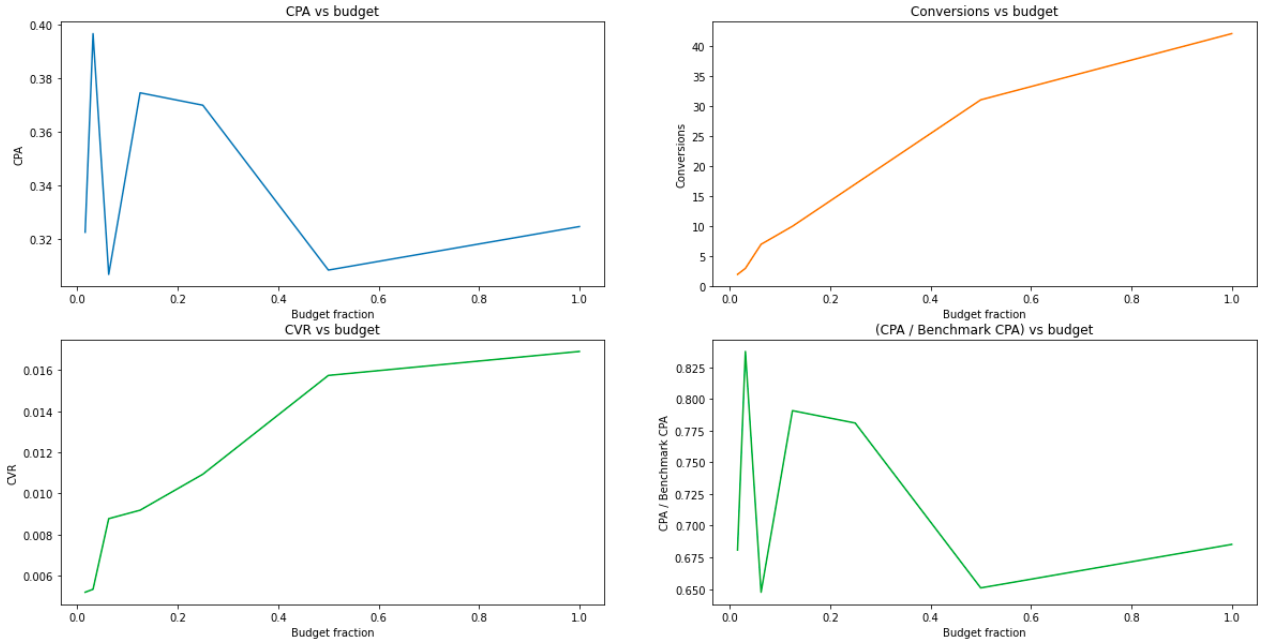


Figure A.3: Budget Allocation Results for ARNN baseline with 30% conversions

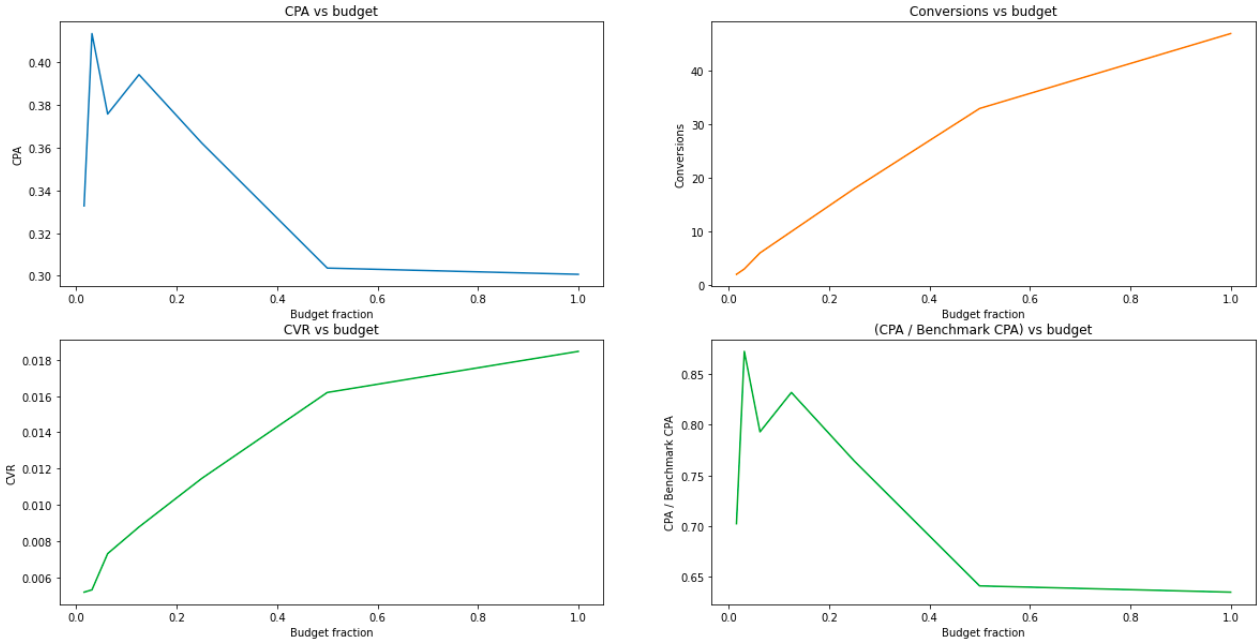


Figure A.4: Budget Allocation Results for ARNN baseline with 50% conversions

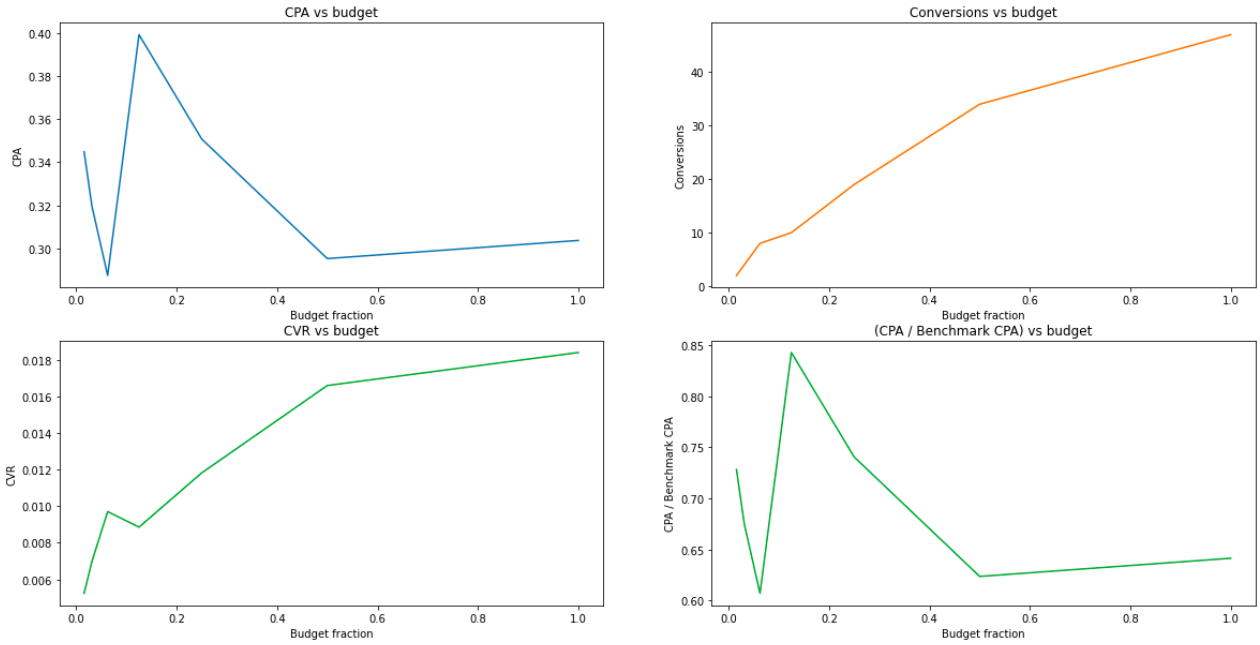


Figure A.5: Budget Allocation Results for GAN supported ARNN model with 10% conversions

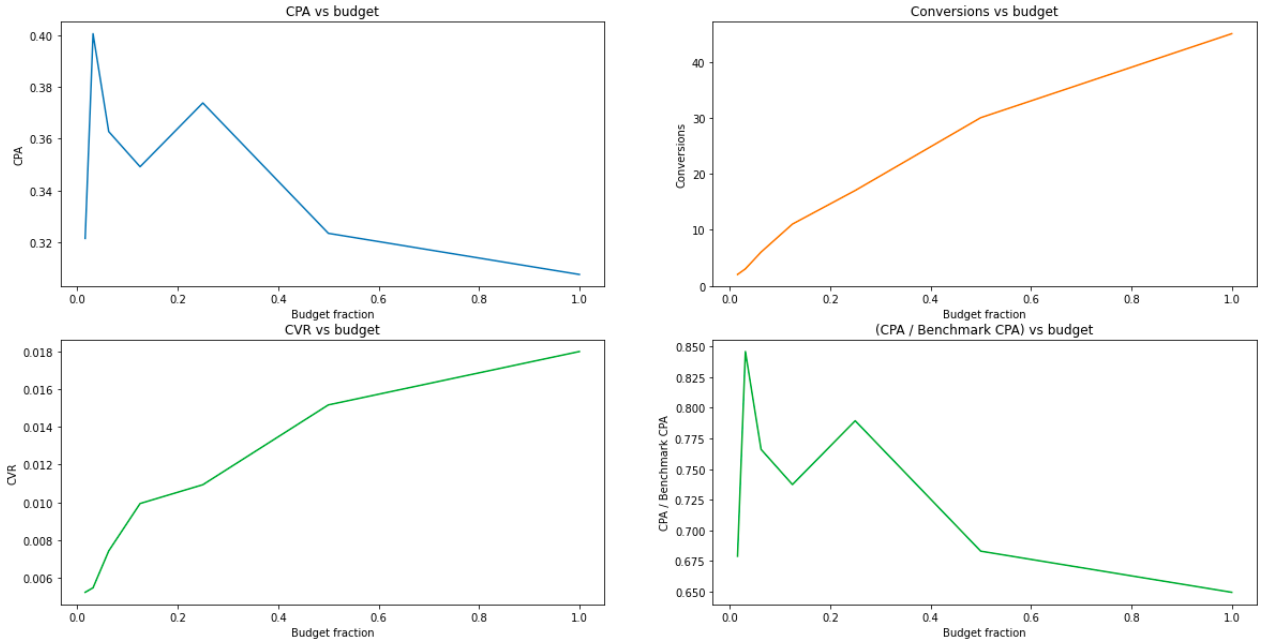


Figure A.6: Budget Allocation Results for GAN supported ARNN model with 30% conversions

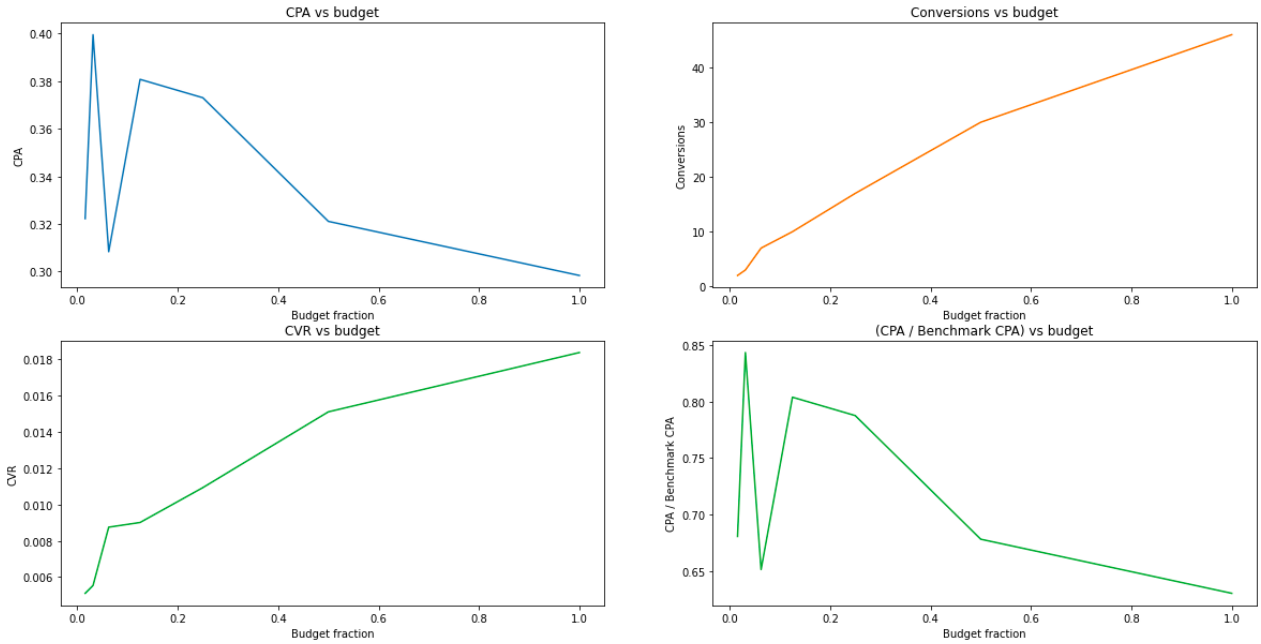


Figure A.7: Budget Allocation Results for GAN supported ARNN model with 50% conversions

Appendix B Code overview

All Python scripts below are in the Jupyter notebook format. The first three scripts are dedicated to replicating the results from Ren et al. (2018). The next five scripts are dedicated to the contribution of this research.

data_sampling_replication.ipynb Samples the dataset to only include sequences of minimal length 3 and maximal length 20 to decrease computational demand for any preprocessing steps. The output is a csv file.

data_preprocessing_replication.ipynb Prepossesses the dataset according to the steps outlined in Section 3.2 to replicate results found in Ren et al. (2018). The outputs are two csv files, one containing the train set data and the other containing the test set data.

attribution_model_replication.ipynb Implements the ARNN model to replicate results found in Ren et al. (2018). The outputs are a txt file containing theoretical performance results, a csv file containing budget results, and a PyTorch file with the trained ARNN model.

data_sampling.ipynb Samples the dataset to only include sequences of minimal length 20 and maximal length 31 to decrease computational demand for any preprocessing steps. The output is a csv file.

data_preprocessing.ipynb Prepossesses the dataset according to the steps outlined in Section 3.2. The output are two csv files, one containing the train set data and the other containing the test set data.

attribution_model.ipynb Implements the ARNN model. The outputs are a txt file containing theoretical performance results, a csv file containing budget results, and a PyTorch file with the trained ARNN model.

attribution_model_with_replication.ipynb Implements the Baseline ARNN model. The outputs are a txt file containing theoretical performance results, a csv file containing budget results, and a PyTorch file with the trained ARNN model.

gan_supported_attribution_model.ipynb Implements the GAN ARNN model. The outputs are a txt file containing theoretical performance results, a csv file containing budget results, and three PyTorch files of the trained ARNN, Generator and Critic models.

attribution_via_gan_anomaly.ipynb Implements the AALAD model. The outputs are a txt file containing theoretical performance results, a csv file containing budget results, and three PyTorch files of the trained Generator, Encoder and D_{xz} models.