Erasmus University Rotterdam

Erasmus School of Economics

Master Thesis Econometrics and Management Science

Business Analytics and Quantitative Marketing

Causal Interpretable Credit Conversion allocation in Multi-Touch Attribution using Neural Networks

---

Author:

Angad Singh

Student Number:

532588as

Supervisor:

dr. Eoghan O'Neill

Second Assessor:

dr. Kathrin Gruber

Abstract

With the surge in online advertising, the abundance of advertisements has made the consumer journey more complex. Determining the impact of different channels on conversions has become a significant challenge. Multi-touch attribution (MTA) addresses this by assigning conversion credits to channels based on their influence on user conversions. Recurrent Neural Networks and Shapley Values have been commonly used to calculate these credits. However, Shapley Values are computationally expensive and may not capture causality effectively. This research paper aims to explore alternative methods, such as the Incremental Value Heuristic (IVH) and the Simplified Shapley Value method, to obtain causal conversion credits with clearer interpretations. These methods will be compared with the conversion credits generated by an attention mechanism in a neural network. Using the Criteo data set, consisting of more than 16 million impressions and 45 thousand conversions recorded over 30 days, the study investigates the feasibility of IVH and the Simplified Shapley Value method in capturing causal conversion credits. The findings illuminate the suitability of IVH and Simplified Shapley Values to obtain meaningful and interpretable conversion credits, offering advertisers and online ad exchanges valuable insights for effective campaign design and marketing budget optimisation. The main finding is that the model discussed in this paper provides more accurate and better calibrated conversions predictions.

March 2024

Contents

Acknowledgements

I want to express my gratitude to Dr. Eoghan O'Neill for being my thesis supervisor. Your feedback and insights have been very helpful. I also want to thank Yassine Ben Haddou and Rohit Khiara for being my support and motivation throughout the process of writing this paper. Lastly, I want to give credit to François Chollet, who is the author of Chollet (2021). This book has helped me grasp a firm understanding of deep learning and has provided me with the practical skills and tools that enabled me to build the deep learning models discussed in this paper.

1 Introduction

Online shopping and advertising are becoming increasingly important. According to Leong, Hew, Ooi, and Dwivedi (2020), the money spent on online advertising worldwide was approximately USD 240 billion in 2018 and is expected to grow at an annual rate of 37% to 40% from 2019 to 2025. With the rise of online advertising, the consumer journey has become progressively more complex. Each day, consumers are exposed to a multitude of advertisements through various online channels, for example, social networks, search engines, emails, and websites. Mapping the journey that a consumer takes through these various channels is of great importance to online advertisers when deciding their pricing strategy or guiding their budget allocation. However, it remains a challenge to identify which advertisements are most crucial in creating a conversion.

The issue of Multi-touch Attribution (MTA) pertains to the process of determining the value of each customer touchpoint leading up to a conversion. It is about identifying which advertisements and channels play a significant role in influencing a consumer's decision to make a purchase (Du, Zhong, Nair, Cui, & Shou, 2019; Yang, Dyer, & Wang, 2020; Kumar et al., 2020). A touchpoint in this context refers to any interaction a consumer has with a brand, product, or service. The allocation of the correct conversion credit to each touchpoint is important for numerous reasons. Firstly, it can be used as input for a campaign design. Second, conversion credits can be used to create an optimal allocation of the marketing budget. Lastly, it can provide information on why a campaign worked or did not work (Du et al., 2019).

Various conversion attribution methodologies have been used throughout the years. The default methodology in the industry used to be the "last touch attribution" (LTA) model, which is a heuristic-based model that assigns all conversion credits to the touchpoint which a consumer has last interacted with after a conversion (Dalessandro, Perlich, Stitelman, & Provost, 2012). Other heuristic-based methods include "linear touch attribution," where conversion credits are distributed equally across each touchpoint with which a consumer has interacted with, or "first-touch attribution," where all conversion credits are assigned to the first channel after conversion. However, these oversimplifications ignore the complexity and multifaceted nature of consumer behaviour, which can lead to inaccurate credit assignment.

After the heuristic-based models, data-driven models started to emerge. In this paper, data-driven methods are defined as approaches that leverage data to address the issue of MTA, excluding the use of Neural Networks. An example is the Logistic Regression (LR) proposed by Shao and Li

(2011). Other examples include the Simple Probabilistic (SP), Additive Hazard (AH), and Additional Multi-touch Attribution (AMTA) models outlined in Dalessandro et al. (2012), Y. Zhang, Wei, and Ren (2014), and Ji and Wang (2017), respectively. More recently, neural networks have emerged as a growing solution to address the issue of MTA (Ren et al., 2018; Arava, Dong, Yan, Pani, et al., 2018; Du et al., 2019; Kumar et al., 2020; Yao, Gong, Zhang, Chen, & Bi, 2022). These approaches, as shown in Yao et al. (2022), exhibit significant superiority over non-machine learning methods to predict user conversion. However, they rely on sequential modelling techniques that contribute to long training times. Additionally, these approaches often employ models with multiple layers, which pose challenges in interpreting the model due to its complexity. It is also unclear whether these models capture causal effects instead of mere correlation.

To address the problem of poor interpretability, many methods use Shapley Values (Shapley et al., 1953) to allocate conversion credits (Du et al., 2019, Yao et al., 2022). Shapley Values have been criticised for being computationally expensive (Verdinelli & Wasserman, 2023), but novel methods have been suggested to make the computation more feasible, such as the Simplified Shapley Value Method outlined in Zhao, Mahboobi, and Bagheri (2018). Shapley Values were introduced in cooperative game theory to assess the marginal contribution of each individual player in a game. It can be summarised as the expected value of the marginal contribution over all permutations of players. In MTA, Shapley Values can be used to obtain allocation credits by treating each marketing channel as a player in a game.

To address the issue of causality, Yao et al. (2022) conduct a simulation study to examine the performance of their method in confounding scenarios. The method aims to mitigate confounding bias from both static and dynamic perspectives. However, this paper uses the accuracy of the conversion prediction as a proxy to assess the effectiveness of the method in mitigating confounding bias, but it is not clear how the higher prediction accuracy is translated into causal conversion credits. The research paper does not compare the "true" allocation of conversion credits with the one generated by their method.

The aim of this research is to use a neural network to develop an attribution model that, under certain assumptions, is causally interpretable. To achieve this goal, the following research question is introduced:

How can a Neural Network be employed to derive causally interpretable conversion credits within the framework of Multi-Touch Attribution?

This study evaluates the effectiveness of the Dual-Attention Recurrent Neural Network (DARNN) model, as described in Ren et al. (2018). First, the model is modified to include multiple recurrent layers, with the aim of investigating whether this enhances the accuracy of conversion predictions. This is explained in more detail in Section 2.1. Second, Simplified Shapley Values are used in conjunction with the DARNN model to determine whether the model produces causal attribution scores. The DARNN model is chosen for evaluation for several reasons. First, it generates both conversion and click predictions, both of which can be used to assign attribution scores. Second, this method significantly exceeds the models that do not utilise a neural network mentioned above in terms of conversion prediction accuracy.

The dataset, referenced as Diemert Eustache, Galland, and Lefortier (2017), is provided by Criteo, a company specialising in online advertising research. This data set encompasses 30 days of live traffic data, including impressions shown to users and whether a displayed banner led to a click and/or conversion.

In summary, this paper unveils significant findings. The model discussed in this paper not only yields better accuracy and better calibrated conversion predictions relative to the DARNN model, but also offers valuable insights by ranking the most influential channels through the use of the attribution measures discussed above.

The structure of this paper is as follows. First, Section 2 offers a review of the related literature. Section 3 provides a complete description of the data set. The methodology used to derive conversion predictions, click predictions, and attribution scores is discussed in Section 4. The results obtained from the methods outlined in the methodology are presented in Section 5. Lastly, Section 6 summarises the results and findings and suggests directions for further research.

2 Related Work

The Related Work section is divided into three parts. First, in Section 2.1, the literature which uses recurrent neural networks in the context of MTA will be discussed. Then, in Section 2.2, various attribution measures that have been used in the literature to allocate conversion credits will be discussed.

2.1 Sequential Modelling using RNN in MTA

Recurrent Neural Networks (RNNs) have a long history, with one of the first use cases dating back to 1982 (Hopfield, 1982). Since then, RNNs have found successful applications in various fields. For example, RNNs have been used for tasks such as handwriting recognition (Graves et al., 2008), Machine Translation (Sutskever, Vinyals, & Le, 2014), and recognition of medical diagnoses (Lipton, Kale, Elkan, & Wetzel, 2015). A comprehensive and in-depth overview of RNNs and their diverse applications can be found in Lipton, Berkowitz, and Elkan (2015). The use of RNNs have also been extensively used within the MTA framework (Du et al., 2019; Kumar et al., 2020; Ren et al., 2018; Yao et al., 2022).

The rationale behind employing RNNs lies in their ability to capture dependencies between sequences, a characteristic that is expected to be present in the context of MTA. For example, consider a customer who is exposed to multiple ads or touchpoints over a period of time. Using an RNN, the model can incorporate the order and timing of these touchpoints and learn the dependencies between them. This enables the model to better understand the impact of each touchpoint on the final conversion and to attribute appropriate credit to each marketing channel or advertisement.

The first paper to leverage an RNN for sequential user modelling in the context of MTA is Ren et al. (2018). The DARNN model proposed in the paper uses a long-short-term memory (LSTM)-RNN[1], which is a type of RNN that can capture long-term dependencies.

Figure 1 shows the architecture of the DARNN model. The model uses an encoder to model impression-level user behaviour, followed by a decoder for click-level sequential prediction. Section 4.1.3 discusses in more detail the workings of an encoder-decoder architecture and explains the intuition behind the use of the structure in the context of MTA. One drawback of the DARNN model could be that it only uses one recurrent layer, potentially missing out on discovering intricate temporal relationships and capturing more nuanced patterns in the data.

[1]There is a discrepancy between the code provided by Ren et al. (2018) and the paper. Although the paper recommends using an LSTM-RNN, the code uses a GRU-RNN. To increase computational efficiency, a GRU-RNN is employed as described in further detail in Section 4.1.2.

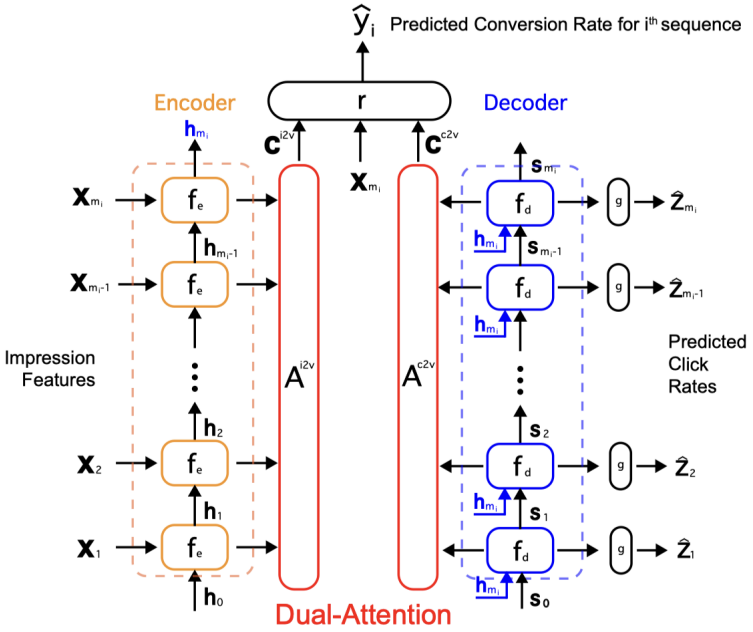


Figure 1: DARNN Architecture. Source: Ren et al. (2018)

Arava et al. (2018) propose the Deep Neural Net With Attention multi-touch attribution (DNAMTA) model, which uses a Time Decay Attention Layer to account for differences in time gaps between touchpoints. The paper also introduces a Fusion model (Fusion DNAMTA) built on top of the DNAMTA model with the Time Decay attention layer. The Fusion DNAMTA is equipped to handle user characteristics such as age, sex, and other static user information to mitigate confounding from a static perspective. Similarly to Ren et al. (2018), the model uses an RNN with a single layer. Furthermore, the model is not suitable for the Criteo data set, since it does not disclose information on the specific features that cover the user characteristics.

Du et al. (2019) utilise a bi-directional RNN to capture dependencies between sequences. A bi-directional RNN is a Neural Network that allows for backward recurrence. Their rationale for using a neural network with backward recurrence is that future ad impressions at time $t+1, \ldots, T$ can help predict conversion at time t. However, the sequence and features of future advertising impressions may not always be known at time t. For this reason, an RNN with only forward recurrence will be considered.

By stacking multiple recurrent layers, a Deep Recurrent Neural Network can be constructed, as shown in Figure 2. Deep RNNs offer the advantage of capturing complex relationships between

input and output at each time step.

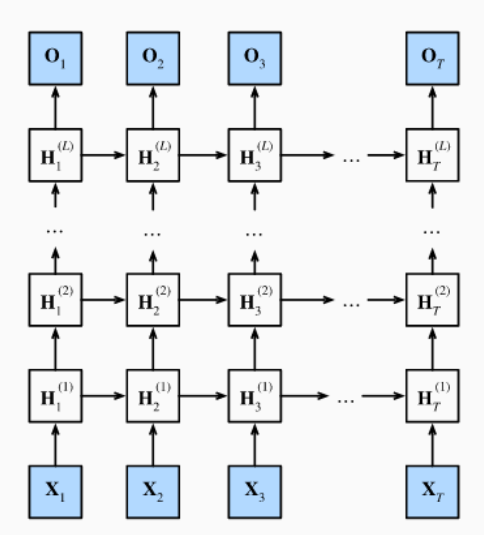


Figure 2: Graph of a Deep RNN. Source: A. Zhang et al. (2021)

Each layer of the network, denoted as $H_t^{(l)}$, where $l = 1, \ldots, L$ and $t = 1, \ldots, \tau$, represents a hidden state. In this paper, the term Deep RNN specifically refers to recurrent neural networks where there is a recurrence between all cells in every layer of the neural network, as shown in Figure 2. This architecture is commonly referred to as stacked RNN in the literature. Deep RNNs have demonstrated successful applications in various domains. For example, in neural machine translation tasks, Dabre and Fujita (2021) achieved superior performance by employing stacked RNNs compared to single-layer vanilla models. Furthermore, Graves, Mohamed, and Hinton (2013) uses a deep RNN for speech recognition and achieved state-of-the-art results in that field. Similarly, in character-level language modelling, Hermans and Schrauwen (2013) used deep RNN and achieved remarkable performance results, exceeding other types of recurrent neural networks.

In the domain of MTA, a deep RNN has been applied in the work by Yang et al. (2020). The research introduces the DeepMTA model, which uses stacked layers of phased LSTM (Neil, Pfeiffer, & Liu, 2016). However, the model does not employ an encoder-decoder structure to address the data-sparsity issue discussed earlier. Furthermore, the model does not incorporate an attention mechanism to highlight the significance of impressions and/or clicks.

To further investigate the application of Deep Recurrent Neural Networks within the MTA framework, this paper investigates whether the inclusion of multiple stacked recurrent layers in the DARNN model enhances the accuracy of conversion prediction. It is worth mentioning that several papers regarding MTA have employed multi-layered RNNs in their studies. However, in these papers

(Arava et al., 2018; Kumar et al., 2020; Yao et al., 2022), the deeper layers of the neural networks do not use recurrence.

2.2 Attribution Measures

Within the fields of statistics and machine learning, there has been a growing interest in the concept of explainability (Verdinelli & Wasserman, 2023). Understanding and interpreting the output of a prediction model in MTA is useful because it helps identify which marketing touchpoints are most influential, allowing advertisers and ad exchanges to focus resources on effective channels and improve decision-making. It also builds trust by providing transparency and data-driven insights to stakeholders.

Attribution measures are a significant aspect that aim to measure the importance of a channel within the underlying Data Generating Process (DGP). It involves quantifying the degree to which displaying a particular impression has influenced the probability of conversion for a specific user. Such measurements play a crucial role in the allocation of conversion credits.

The use of Shapely values to allocate conversion credits is fairly common in online advertising (Dalessandro et al., 2012; Du et al., 2019; Yao et al., 2022). The concept of Shapley values originates from cooperative game theory (Shapley et al., 1953), where it was initially used to assess the individual marginal contribution of players in a game. In the context of MTA, conversion credits are allocated using Shapley values by treating each marketing channel as a player within a game. This game represents different marketing strategies or campaign designs.

Another commonly used method for attribution of importance to channels in online advertising is the Incremental Value Heuristic (IVH) (Singal, Besbes, Desir, Goyal, & Iyengar, 2019). IVH can be defined as the change in the conversion probability of a customer when an ad is withheld from their path. Arava et al. (2018) have also used IVH, referring to it as incremental scores. IVH offers several benefits, including its tractability for calculation and clear interpretation. However, it has some drawbacks. Singal et al. (2019) argue that IVH can lead to incorrect allocation. The extent to which this could occur is further researched in this paper.

To further investigate the use of Simplified Shapley Values and IVH in MTA as conversion credit allocation measures, an investigation is conducted to determine whether Simplified Shapley Values and IVH can be used in conjunction with the DARNN model to acquire attribution credits that are interpretable. To examine the causal nature of the conversion credits obtained from the attribution measures, conversion journeys will be simulated. An assessment is made to investigate whether these

attribution measures provide causal estimates under certain assumptions. Furthermore, this paper also highlights the biases and their consequences for these measures if these assumptions are not valid.

3 Data

The data Diemert Eustache et al. (2017) used for this research are supplied by Criteo. Criteo is an online advertising research company. The authors have made this data set publicly available for the purpose of attribution modelling in real-time auction-based advertising (Eustache, Julien, Galland, & Lefortier, 2017).

The data set consists of live traffic data for 30 days. It comprises over 16 million impressions and 45 thousand conversions from approximately 700 campaigns. In this data set, impressions may correspond to click actions, and each touchpoint in the user action sequence is labelled to indicate whether a click has occurred. Additionally, if the sequence of touchpoints results in a conversion event, the corresponding conversion ID is provided. For each display, nine contextual features associated with the display are also provided. These are used to learn the click-and-conversion model. These nine contextual features are categorical. The meanings of these features are not disclosed by Criteo. The timestamp of each impression is also given. The data set also includes additional information, among others, such as the price Criteo paid for the display (cost) and the cost-per-order (cpo) to represent the expenses associated with each order when the conversion is attributed to Criteo. However, these variables will not be used for this research.

To make the data ready for use, pre-processing and cleaning must be performed. The same approach will be followed as described in Ren et al. (2018). Since conversion is a rare event, researchers perform downsampling. The sampling rules and sequence preparation rules are as follows: If a user has multiple conversion events, the action sequence is split according to the conversion time to ensure that each sequence has at most one conversion. Sequences with a minimum length of three and a maximum length of 20 are extracted with a sequence duration of 14 days. All sequences that lead to a conversion have been retained and uniformly sampled to be twenty times smaller than the number of nonconverted sequences. Figure 3 shows the distribution of the sequence lengths.

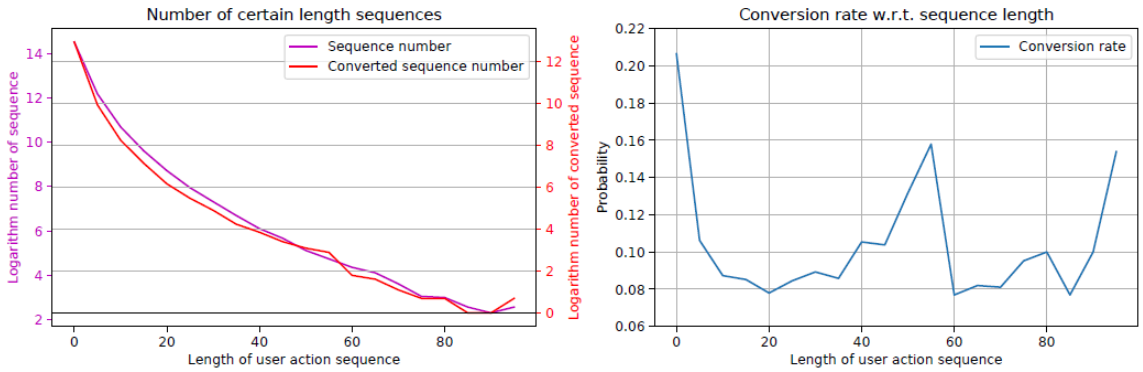


Figure 3: Sequence length distribution and Conversion distribution against the sequence length. Source: Ren et al. (2018)

The left plot in Figure 3 shows the distribution of the sequence lengths and the length of sequences which lead to a conversion on a logarithmic scale. From this plot it follows that longer sequences are much more rare than shorter ones. Also, the number of nonconverted sequence lengths and converted sequence lengths seem to be highly correlated. The plot on the right shows the density of the conversion rate with respect to the conversion length. From this plot it follows that a longer behaviour sequence does not necessarily mean that the probability of conversion is higher and that not all the touchpoints have an additive positive influence. Some touchpoints might have negative effects.

4 Methodology

The methodology consists of three sections. Section 4.1 explains the key components used for Sequence-to-Sequence modelling. Section 4.2 delves into the model used to estimate conversion and clicks. Section 4.3 outlines the training procedure and evaluation metrics. Lastly, Section 4.4 discusses the attribution measures employed to obtain conversion credits.

4.1 Key Components for Sequence-to-Sequence Modeling

4.1.1 Embedding Layer

Frequently, categorical variables are represented as *one-hot* or *dummy* vectors. This means that each category within the variable is transformed into a binary vector where the position representing that category is marked as 1 and all other positions are marked as 0 (Johannemann, Hadad, Athey, &

Wager, 2019). The problem with this approach is that the binary vector can become extremely large. In the context of natural language processing, the size of a vector would be the size of the number of unique words (vocabulary size). Another disadvantage of using this approach is that each word is equally distant from each other, similar words. In the context of text classification or sentiment analysis, it could be beneficial for a model to understand that words like 'good' and 'great' are often used in similar contexts.

Embedding layers circumvent these issues by transforming categorical data, e.g. user IDs, and words, into a dense continuous vector of a fixed size. An embedding layer reduces the dimensionality of the input data by representing each word as a much smaller continuous vector. This makes the computational problem much more manageable. Furthermore, the continuous vectors produced by an embedding layer can capture semantic relationships between words or items that could improve the accuracy of the prediction.

Within the context of MTA, an embedding could thus capture the relationships between the categorical features related to the touchpoints, such as the channel. An important point to note is that the embedding layer does not start with any understanding of the relationships between the items it is embedding. Instead, it learns these relationships from the training data during the neural network training process. The weights in the embedding layer are learnt through backpropagation, just as the weights in any other layer of the network.

4.1.2 Gated Recurrent Unit (GRU)

When RNNs were first introduced, the challenges associated with learning long-term dependencies, mainly due to issues such as vanishing and exploding gradients, became evident. To address these issues, Hochreiter and Schmidhuber (1997) introduced the Long Short-Term Memory (LSTM) RNN. The term LSTM comes from a unique idea. In regular recurrent neural networks, there is long-term memory through slowly changing weights that encode general knowledge. There is also short-term memory through temporary activations passing between nodes. The LSTM model adds an intermediate storage called a memory cell, made up of simpler nodes in a specific pattern. The gated recurrent unit (GRU), introduced by Cho, Van Merriënboer, Bahdanau, and Bengio (2014), presents a simplified version of the LSTM memory cell. It typically delivers similar performance, but with the added benefit of faster computation, as noted by Chung, Gulcehre, Cho, and Bengio (2014).

A GRU cell has two main components: a Reset Gate and an Update Gate. Figure 4 shows the

architecture of a GRU cell.

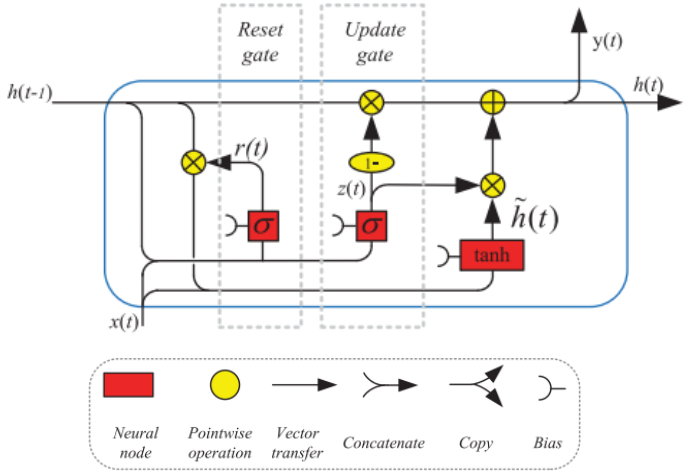


Figure 4: Architecture of a GRU cell. Source: Yu et al. (2019)

Based on Figure 4, the GRU cell can be mathematically expressed as follows

$$\mathbf{r}_t = \sigma\left(W_{rh}\mathbf{h}_{t-1} + W_{rx}\mathbf{x}_t + \mathbf{b}_r\right), \tag{1}$$

$$\mathbf{z}_t = \sigma\left(W_{zh}\mathbf{h}_{t-1} + W_{zx}\mathbf{x}_t + \mathbf{b}_z\right), \tag{2}$$

$$\tilde{\mathbf{h}}_t = \tanh\left(W_{\tilde{h}h}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + W_{\tilde{h}x}\mathbf{x}_t + \mathbf{b}_{\tilde{h}}\right), \tag{3}$$

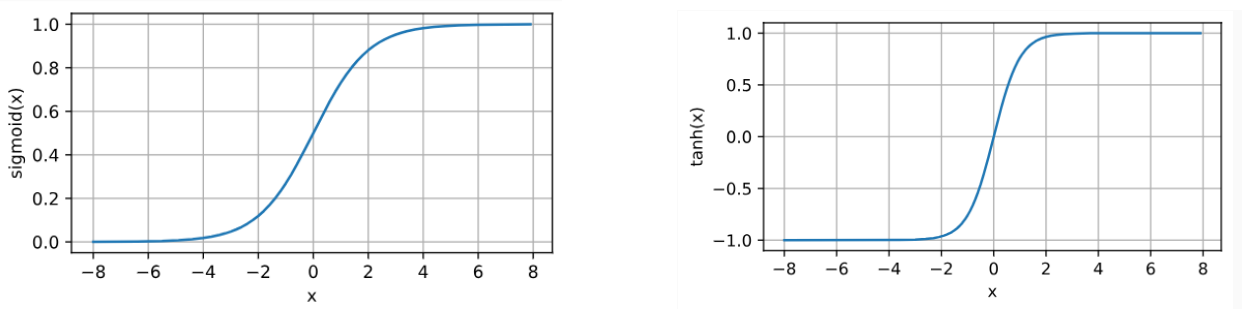
$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t, \tag{4}$$

where $W_{\cdot\cdot}$ are weight matrices and $\mathbf{b}_{\cdot}$ are bias vectors. The operator $\odot$ denotes the Hadamard product, which is the element-wise product of two matrices. Lastly, $\mathbf{x}_t$ and $\mathbf{h}_t$ denote the data and the hidden state, respectively, at time t.

Equations (1) and (2) show the reset gate and the update gate, respectively. $\mathbf{z}_t$ incorporates the effect of the update gate, by determining how much the new hidden state $\mathbf{h}_t$ matches with the old hidden state $\mathbf{h}_{t-1}$. This becomes more clear in equation (4). When $\mathbf{z}_t$ is close to 0, the new hidden state will be close to $\mathbf{h}_{t-1}$. On the contrary, whenever $\mathbf{z}_t$ is close to 1, the new hidden state approaches the candidate hidden state $\tilde{\mathbf{h}}_t$. The candidate hidden state, $\tilde{\mathbf{h}}_t$ shown in equation (3), is constructed by determining how much of the old hidden state is "*reset*". It does this by taking the Hadamard product between $\mathbf{r}_t$ and $\mathbf{h}_{t-1}$. If $\mathbf{r}_t$ is 1, the full previous hidden state, $\mathbf{h}_{t-1}$ is used in the candidate hidden state. In contrast, if $\mathbf{r}_t$ is 0 only the input at time t ($\mathbf{x}_t$) will be used in the hidden candidate state at time t.

A sigmoid activation function (σ) is used to ensure r_t and z_t are always between 0 and 1. The

sigmoid activation function essentially maps all real numbers to $(0, 1)$ shown in Figure 5a. The hyperbolic tangent activation function (tanh) is also used in equation (3) (to calculate the hidden candidate state). It maps all the real numbers to $(-1, 1)$ which can be seen in Figure 5b. This means that the hidden state (h_t) and the candidate hidden state $(\tilde{h}_t)$ are always in between $(-1, 1)$ at every time step.



(a) Sigmoid activation function: $f(x) = \frac{1}{1+e^{-x}}$

(b) Tanh activation function: $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$

Figure 5: Activation functions

4.1.3 Encoder-Decoder structure

In sequence-to-sequence modelling, an encoder-decoder structure is used to handle input and output sequences that are unaligned and/or are of varying length. An example where encoders-decoders are commonly used is machine translation tasks. When translating a sentence (which can be regarded as a sequence of words) from one language to another, we should allow for the length of the output sequence to differ from the length of the input; e.g. "*We are studying*" translated to French is "*Nous étudions*".

However, in the context of this paper, the encoder-decoder structure serves another purpose. According to Ren et al. (2018), the rationale behind employing an encoder-decoder structure is to alleviate the data-sparsity problem. In the case of ad delivery, the sequence of user actions typically follows a pattern, "impression-click-conversion". However, clicks are less frequent than impressions, and conversions are even rarer than clicks. This leads to a scarcity of click and conversion data compared to impression data. Therefore, an encoder is employed to obtain a shared representation of the user behaviour features, which can then be used to predict clicks. The signal of the click behaviour is then utilised to improve the estimation capacity for the conversion behaviour.

Figure 6 shows an example of a encoder-decoder structure. First, the input sequence is passed to the encoder. The final hidden state of the encoder is used as a summary of the input sequence.

This hidden state is then passed to the decoder to obtain the output sequence. It does this by using the final hidden state as the initial state and generates an output sequence step by step. At each step, it predicts the next element in the sequence based on the previously generated elements and the final hidden state of the encoder.

Figure 6: A visual representation of an encoder-decoder. Source: A. Zhang et al. (2021)

4.1.4 Attention Mechanisms

Attention mechanisms were first introduced by Bahdanau, Cho, and Bengio (2014). They refer to a set of techniques that allow the model to focus on certain parts of the input. In a classic encoder-decoder model, the input is compressed into a context vector, which is of fixed length. However, attention suggests that the decoder should be able to focus on various parts of the input at each step of decoding instead of relying on a single representation. This allows for more flexible and contextually informed decoding, which improves performance and understanding. The attention mechanism in this paper works by taking a weighted sum of the hidden states in the encoder and decoder, where the weights represent the attribution given to each state in a sequence, which can be seen in Figure 7. This way of calculating attention is called additive attention. Various attention mechanisms exist, including (scaled) dot-product attention, content-based, and location-based mechanisms; each has a unique method for computing attention scores. However, this paper will not go into the details of these alternatives.

4.1.5 Auxiliary Components

This section briefly discusses additional techniques employed in the DARNN(+) model that do not require detailed elaboration, but are mentioned for completeness. These include dropout, gradient clipping, batch processing, padding, and regularisation. Dropout, gradient clipping, and regularisation are techniques used to prevent the neural network from overfitting the training data.

Dropout is introduced by Srivastava (2013). It works by randomly dropping out nodes in the neural network. This introduces a bias in the model weights during training, which can enhance performance on the test data. In sequence modelling using RNNs, dropout is applied not only to the input data, but also to the hidden states between the RNN cells at each time step, commonly

referred to as recurrent dropout.

Gradient clipping is a technique that clips the gradients to a specific range, creating a bias in the gradients and consequently in the model weights during training. In addition to preventing overfitting, it is also commonly used to ensure that gradients do not vanish or explode during backpropagation (Pascanu, Mikolov, & Bengio, 2013).

Padding, described in Dwarampudi and Reddy (2019) as Pre-Padding, refers to a sequence modelling technique to ensure uniformity of the input data. Within the context of MTA, journeys have different lengths. Some journeys include only one touchpoint whereas others include dozens. To make sure the neural network can process the data, zeros are added to shorter sequences so that all sequences have the same length.

Batch processing in the context of machine learning involves dividing a data set into smaller subsets or batches, allowing a model to process multiple data samples simultaneously during training, leading to improved computational efficiency and faster convergence (Devarakonda, Naumov, & Garland, 2017).

Lastly, regularisation is another approach to reduce overfitting by shrinking the model weights to zero (Van Laarhoven, 2017). In this paper, $l2$-regularisation is used.

4.2 DARNN+

4.2.1 Notation

For notation, a similar approach will be used as outlined in Ren et al. (2018), which assumes the existence of n users indexed by $i \in \{1, \ldots, n\}$. Each user is denoted as u_i. A user can have multiple interactions with the ad content of an advertiser. Therefore, the sequence of touchpoints to which a user has been exposed is denoted as $\{u_i, \{\mathbf{q}_{ij}\}_{j=1}^{m_i}, y_i, T_i\}$. Here, u_i represents the user, $\{\mathbf{q}_{ij}\}_{j=1}^{m_i}$ represents the set of m_i browsing activities a user has with the advertisements of an advertiser, y_i is an indicator of whether the user converts and T_i represents the conversion time if it occurs (otherwise null). Each touchpoint, denoted as $\mathbf{q}_{ij}$, contains a categorical vector of characteristics $\boldsymbol{x}_{ij}$ and a binary click indicator z_{ij}. The click indicator z_{ij} is equal to 1 if the impression is a click impression and 0 otherwise. The feature vector $\boldsymbol{x}_{ij}$ contains information on the content of the ad. It also contains the channel ID feature, denoted as c_{ij}, which is the channel over which this touchpoint is delivered and t_{ij} which is the time at which the interaction occurred. c_{ij} is a categorical variable.

4.2.2 Architecture

The DARNN model consists of three parts: the encoder for impression-level behaviour modelling, the decoder for click-level behaviour modelling, and the dual attention mechanism to jointly model click and impression behaviour to obtain the final conversion estimation. Figure 1 shows the architecture of the Neural Network.

The model works by first feeding the user sequence to the encoder, which contains all the touchpoints in that sequence. Considering that the side information feature vector consists primarily of categorical variables, an embedding layer is employed to convert the sparse input features into dense representation vectors. A GRU as described in Section 4.1.2 is then used to obtain a latent representation of the input as shown in equation (5)

$$\boldsymbol{h}_{ij} = f_e(\boldsymbol{x}_{ij}, \boldsymbol{h}_{ij-1}), \tag{5}$$

where $\boldsymbol{h_{ij}}$ denotes the hidden vector at each time step j.

To model clicks, the DARNN model uses the decoder to decompose the joint click probability into ordered conditionals as

$$p(\mathbf{z}_i) = \prod_{j=1}^{m_i} p\left(z_{ij} = 1 \mid \left\{z_{i1}, \ldots, z_{ij-1}\right\}, \mathbf{x}_i\right), \tag{6}$$

where $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{im_i})$ and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{im_i})$. Then the conditional probability of a click is modelled as

$$\hat{z}_{ij} = p\left(z_{ij} = 1 \mid \left\{z_{i1}, \ldots, z_{ij-1}\right\}, \mathbf{x}_i\right) = g\left(z_{ij-1}, \boldsymbol{s}_{ij}\right), \tag{7}$$

where g is a multi-layer fully connected perceptron with sigmoid activation function. The sigmoid activation function ensures that the output is between 0 and 1 and can therefore be interpreted as a probability. Furthermore, $\boldsymbol{s}_{ij}$ is a hidden vector specific to the j^{th} touchpoint, containing relevant information on the probability that that touchpoint is clicked. It is computed as

$$\boldsymbol{s}_{ij} = f_d\left(\boldsymbol{s}_{ij-1}, z_{ij-1}, \boldsymbol{h}_{im_i}\right), \tag{8}$$

where f_d is a RNN of which the forward functions are given by equation (9). In addition to equation (5), the last hidden state, $\boldsymbol{h_{im_i}}$, is used from the encoder. Because the decoder uses not only past hidden state and past click data, but also the last hidden state from the encoder, the forward

functions differ from equations (1) to (4). The forward functions are given by

$$\begin{aligned}
\mathbf{r}_{ij} &= \sigma\left(W_{rs}\mathbf{s}_{ij-1} + W_{rz}z_{ij} + W_{rh}\mathbf{h}_{im_i}\right),\\
\mathbf{l}_{ij} &= \sigma\left(W_{ls}\mathbf{s}_{ij-1} + W_{lz}z_{ij} + W_{lh}\mathbf{h}_{im_i}\right),\\
\tilde{\mathbf{s}}_{ij} &= \tanh\left(W_{\tilde{s}s}(\mathbf{r}_{ij}\odot\mathbf{s}_{ij-1}) + W_{\tilde{s}x}z_{ij} + W_{\tilde{s}h}\mathbf{h}_{im_i}\right),\\
\mathbf{s}_{ij} &= (1-\mathbf{l}_{ij})\odot\mathbf{s}_{ij-1} + \mathbf{l}_{ij}\odot\tilde{\mathbf{s}}_{ij}.
\end{aligned} \tag{9}$$

Note that the notation z_{ij} is used to denote the clicks at time j, therefore z_j is replaced with l_{ij} in equation (2). The loss for the click probabilities is then calculated using equation (10)

$$L^c = \sum_{i=1}^{n}\sum_{j=1}^{m_i} -z_{ij}\log\hat{z}_{ij} - \left(1-z_{ij}\right)\log\left(1-\hat{z}_{ij}\right) + \theta\sum|w|, \tag{10}$$

where θ is the $l2$-regularisation parameter for the loss function and w denotes a weight in the model. $\sum|w|$ sums over all trainable weights in the neural network. Finally, the conversion probability is calculated as follows

$$\boldsymbol{c}^{i2v} = A^{i2v}\left(\boldsymbol{h}_{i1},\ldots,\boldsymbol{h}_{ij},\ldots,\boldsymbol{h}_{im_i}\right), \tag{11}$$

$$\boldsymbol{c}^{c2v} = A^{c2v}\left(\boldsymbol{s}_{i1},\ldots,\boldsymbol{s}_{ij},\ldots,\boldsymbol{s}_{im_i}\right), \tag{12}$$

$$\hat{\mathbf{y}}_i = p(\mathbf{y}_i = 1 \mid \mathbf{x}_i, \mathbf{z}_i) = r\left(\mathbf{x}_{im_i}, \boldsymbol{c}^{i2v}, \boldsymbol{c}^{c2v}\right), \tag{13}$$

where c^{i2v} refers to the context vectors that represent the input user behaviour vectors, which capture the patterns of user impressions. c^{c2v} represents the context vector obtained by modelling click patterns for estimating conversions. r contains a weighting function that balances the attribution between impressions and clicks, which is a fully connected multilayer neural network for the prediction of the final conversion. $\boldsymbol{x}_{im_i}$ is the vector of characteristics of the last touchpoint user u_i is exposed to, after being fed to an embedding layer. The impression-to-conversion attention c^{i2v} and the click-to-conversion attention c^{c2v} is calculated according to equation (14), which is a linear combination of the hidden states of the encoder and decoder and the attention weights a

$$\begin{aligned}
\boldsymbol{c}^{i2v} &= \sum_{j=1}^{m_i} a_{ij}^{imp2v}\boldsymbol{h}_{ij},\\
\boldsymbol{c}^{c2v} &= \sum_{j=1}^{m_i} a_{ij}^{clk2v}\boldsymbol{s}_{ij},
\end{aligned} \tag{14}$$

The attention weights are calculated through equation (15) which is the softmax operator,

$$a_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{m_i}\exp\left(e_{ik}\right)}, \tag{15}$$

where

$$\boldsymbol{e}_{ij} = \mathcal{E}\left(\boldsymbol{h}_{ij}, \boldsymbol{x}_{im_i}\right), \tag{16}$$

is an energy model that evaluates the importance or credit of each touch point leading to a final conversion. The energy function $\mathcal{E}$ is a multilayer nonlinear deep neural network with the activation function *tanh*. Figure 7 illustrates the attention mechanism.

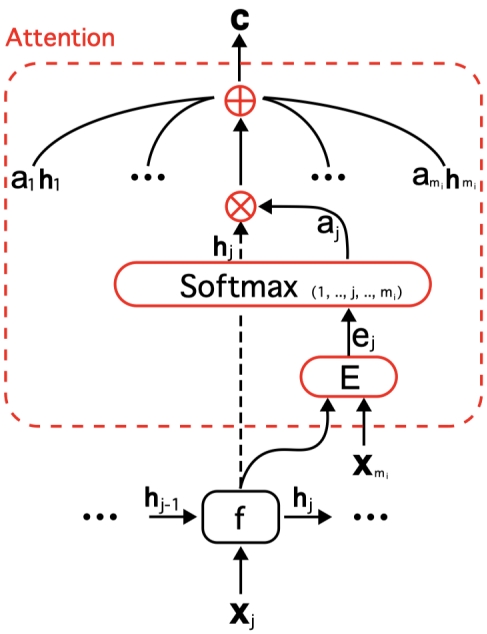


Figure 7: Attention mechanism. Source: Ren et al. (2018)

The final conversion estimations are calculated through equations (17) and (18) as

$$\hat{y}_i = r\left(\boldsymbol{x}_{im_i}, \boldsymbol{c}^{i2v}, \boldsymbol{c}^{c2v}\right) = r_{\text{conv}}\left((1-\lambda)\cdot \boldsymbol{c}^{i2v} + \lambda \cdot \boldsymbol{c}^{c2v}\right), \tag{17}$$

where

$$\lambda = \frac{\exp\left[f_\lambda\left(\boldsymbol{x}_{im_i}, \boldsymbol{c}^{c2v}\right)\right]}{\exp\left[f_\lambda\left(\boldsymbol{x}_{im_i}, \boldsymbol{c}^{i2v}\right)\right] + \exp\left[f_\lambda\left(\boldsymbol{x}_{im_i}, \boldsymbol{c}^{c2v}\right)\right]}. \tag{18}$$

In this context, λ represents the significance or importance of click-level attention compared to impression-level attention. The function f_λ, which is a multilayer perceptron, aims to learn and determine the weight or contribution of these two attention results in estimating the final conversion. The weights in the neural network are learnt by minimising the loss function in equation (19)

$$L^v = \sum_{i=1}^{n} -y_i \log \hat{y}_i - (1-y_i)\log\left(1-\hat{y}_i\right) + \theta \sum |w|. \tag{19}$$

The DARNN+ model differs from the original DARNN model by adding additional recurrent layers, making it a deep recurrent neural network. As previously mentioned, adding more layers could allow the model to capture more intricate relationships. Instead of $\mathbf{h}_{ij}$ and $\mathbf{s}_{ij}$, $\mathbf{h}_{ij}^{(l)}$ and $\mathbf{s}_{ij}^{(l)}$ will be used, where l denotes the depth of the model, i.e. the number of stacked recurrent layers. That means that for $l = 1$ the DARNN+ and the DARNN models are equivalent. Figure 8 gives a visual representation of the DARNN+ model(s).

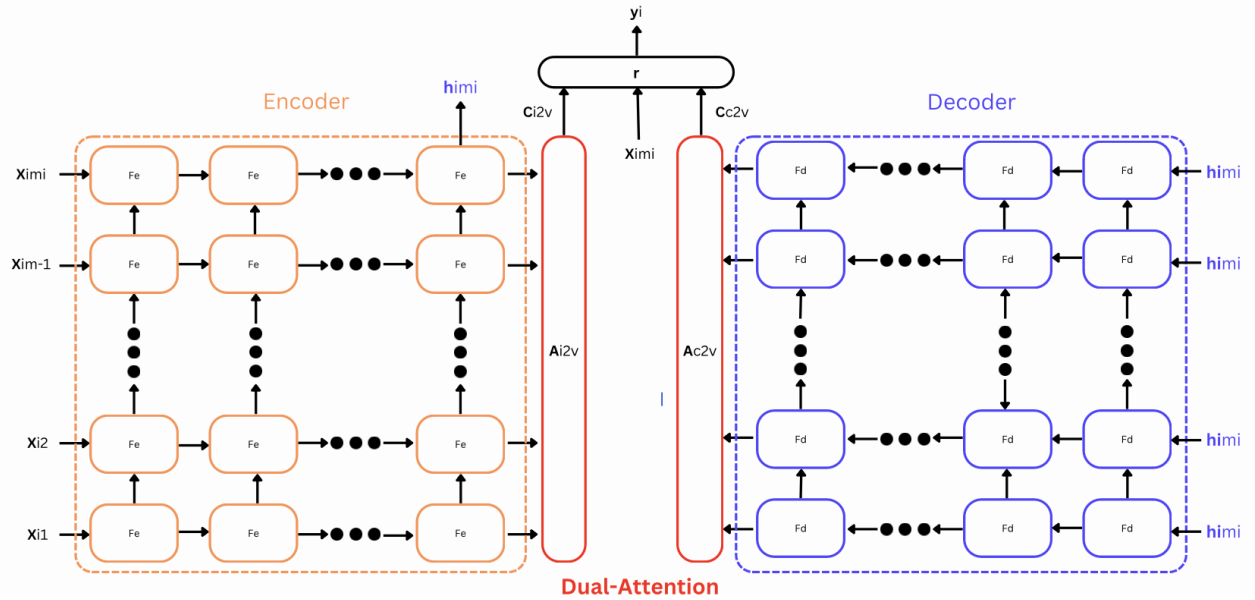


Figure 8: Architecture of the DARNN+ model(s)

4.3 Evaluation & Training Procedure

In this section, the evaluation metrics and the training procedure for the DARNN(+) model are discussed.

4.3.1 AUC Scores

AUC scores are used to assess the model's performance in conversion and click prediction. The AUC, representing the area under the ROC curve, gauges the model's ability to distinguish between conversions and nonconversions (Huang & Ling, 2005). A higher AUC score, closer to 1.0, indicates superior discrimination. A model that performs no better than random guessing would yield an AUC score of 0.5. These scores are calculated for each epoch during both training and testing.

4.3.2 Total Loss

The performance of the model is also evaluated by adding click loss and conversion loss shown in equations (10) and (19) respectively. The training loss incorporates $l2$-regularization, while the test loss omits this term. This ensures consistency with the model's application, where regularisation is not applied during testing. Similarly to AUC scores, losses are computed for each epoch for both training and test data sets.

4.3.3 Calibration Plots

Calibration plots provide an additional means of evaluating the model. They represent the fraction of conversions compared to the mean predicted probabilities. Well-calibrated probabilities align the predictions with the actual data, enhancing the reliability of the probability estimates (Vuk & Curk, 2006). Proper calibration is crucial for accurate attribution measurements described in Section 4.4.

4.3.4 Training Procedure

Following the approach in Ren et al. (2018), the model initiates training by minimising the click loss function as shown in equation (10). Click and conversion AUC scores are calculated for both train and test data during each epoch. The model transitions to joint minimisation of click and conversion loss functions when the click AUC scores rise for three consecutive epochs. This is done by minimising the loss functions as shown in equations (10) and (19) simultaneously. Training ends when conversion AUC scores on test data decrease for three consecutive epochs, indicating convergence.

4.4 Conversion Credit Acquisition

Before delving into various credit allocation methods, it is crucial to define attribution. Following the approach in Dalessandro et al. (2012), attribution is defined as the marginal value created by an ad served to the customer u_i at time j as shown in equation (20)

$$\Psi_{ij} = E\left[y_i \mid \left\|\mathbf{q}_{ij}\right\| > 0, \mathbf{q}_{ij}\right] - E\left[y_i \mid \left\|\mathbf{q}_{ij}\right\| = 0, \mathbf{q}_{ij}\right], \tag{20}$$

where $\left\|\mathbf{q}_{ij}\right\| > 0$ indicates that an ad has been served to customer u_i at time j, $\left\|\mathbf{q}_{ij}\right\| = 0$ indicates that an ad is not being served at time j, and $\mathbf{q}_{ij}$ is a vector of customer attributes and contains contextual features related to the advertisement as mentioned in Section 4.2.2. The assumption is

made that y_i has a causal dependency on $\mathbf{q}_{ij}$, that is, all confounders of y_i are included in $\mathbf{q}_{ij}$. Then the total conversion contribution for customer u_i can be defined as follows:

$$\Psi_i = \sum_{j=1}^{m_i} \Psi_{ij} * I\left(\left\|\mathbf{q}_{ij}\right\| > 0\right). \tag{21}$$

Here the assumption is made that the conversion contribution is additive. To then obtain the total marginal contribution of a channel, Ψ_i is decomposed as follows

$$V_c = \sum_{i \in \mathcal{N}} \sum_{j=1}^{m_i} \Psi_{ij} * I\left(\left\|\mathbf{q}_{ij}\right\| > 0\right) * I(c_{ij} = c), \tag{22}$$

where $\mathcal{N}$ is the set of journeys/users that lead to a conversion, and V_c are the conversion credits for channel $c \in C$, where C is the set of all channels.

4.4.1 Incremental Value Heuristic (IVH)

Following the approach in Arava et al. (2018), the Incremental Value Heuristic calculates the attribution of channel c by comparing the conversion probability before and after removing the touchpoints(s) of the customer's ad journey that were delivered through this channel. IVH can be formulated as follows

$$\begin{aligned}\hat{V}_{c,ivh} &= \sum_{i \in \mathcal{N}} (\hat{\mathbb{E}}(y_i = 1 | \text{ad journey}_i \text{ with channel } c) - \hat{\mathbb{E}}(y_i = 1 | \text{ad journey}_i \text{ excluding channel } c)) \\ &= \sum_{i \in \mathcal{N}} (\hat{P}(y_i = 1 | \text{ad journey}_i \text{ with channel } c) - \hat{P}(y_i = 1 | \text{ad journey}_i \text{ excluding channel } c)).\end{aligned} \tag{23}$$

The Incremental Value Heuristic can be interpreted as the accumulated marginal conversion probability after excluding a channel. It can be negative if the conversion probability is higher when a channel is left out. This means that this attribution measure can capture the negative effects of touch points if these are present.

Under the assumption that all confounding variables are taken into account, that is, the inclusion of channel c captures the partial effect on the probability of conversion, IVH is a causal attribution measure. However, this assumption is unlikely.

4.4.2 Simplified Shapley Value Method

The Simplified Shapley Value Method proposed by Zhao et al. (2018) can be described as follows

$$\hat{V}_{c,shap} = \sum_{S \subseteq C \setminus \{c\}} \frac{1}{|S| + 1} R\left(S \cup \{c\}\right), \tag{24}$$

where C is the set of all channels, S is a subset of C excluding channel c, and the function R is a revenue function that can be thought of as the utility or revenue obtained from a user visiting channels $S \cup \{c\}$. In the case of the DARNN model, revenue is defined as the conversion probability of a user visiting a set of channels that lead to a conversion. Hence, equation (24) can be rewritten as

$$\hat{V}_{c,shap} = \sum_{i \in \mathcal{N}} \sum_{S \subseteq C \setminus \{c\}} \frac{1}{|S|+1} \hat{P}(y_i = 1 | \text{ad journey}_i \text{ with channels } S \cup \{c\}). \tag{25}$$

Attribution for channel c through the Simplified Shapley Value Method can simply be described as a weighted sum of the conversion probabilities of journeys that lead to a conversion and contain channel c. This greatly improves the computational efficiency of the original Shapley Value method, because for each channel it only assesses each coalition at most once. It does not involve calculating the counterfactual scenario $R(S)$, which increases computational efficiency. The Simplified Shapley Value Method is also not able to capture negative effects, because the attribution in equation (25) is always positive by construction.

4.4.3 Attribution through Attention Scores

Adopting the approach in Ren et al. (2018), equations (15) and (18), the attribution for channel c through attention scores can be calculated as

$$\hat{V}_{c,att} = \sum_{i \in \mathcal{N}} \sum_{j=1}^{m_i} \left((1-\lambda) \cdot a_{ij}^{imp2v} + \lambda \cdot a_{ij}^{clk2v} \cdot I\left(c_{ij} = c\right) \right). \tag{26}$$

Here, λ is a mixing parameter learnt through backpropagation, determining the weight assigned to the impression attention weights (a^{imp2v}) and the click attention weights (a^{clk2v}). This attribution measure represents the total weights assigned to a channel during backpropagation by the attention mechanism described in Section 4.2.2. Since the weights are unitless, their interpretation is unclear. This attribution measure cannot also capture negative effects because the weights a^{imp2v} and a^{clk2v} are always positive by construction. This can be seen in equation (15).

4.4.4 Fractional Scores

All the aforementioned attribution measures have different interpretations and/or units. To allow a direct comparison, Fractional Scores are calculated using the approach outlined in Arava et al. (2018)

$$\textit{Fractional Score}_{c,b} = \frac{|\hat{V}_{c,b}|}{\sum_{c \in C} |\hat{V}_{c,b}|} \qquad \text{for } b \in \{ivh, shap, att\}, \tag{27}$$

where $\sum_{c \in C} |\hat{V}_{c,b}|$ represents the total conversion credits using attribution measure b. The Fractional Score indicates the proportion of total conversion credits assigned to channel c using method b, with absolute values considered for potential negative effects when using IVH.

5 Results

This section discusses the performance and findings of the DARNN(+) model(s) on the Criteo Data Set and the Simulated Data Set. The models for the Criteo data set are trained until convergence on Google Colab® using a NVIDIA V100 GPU. The models for the Simulated Data Set are trained on a MacBook Air (2019) equipped with an M1 chip. Models are trained using the hyperparameters shown in Table 1.

Table 1: Hyperparameter settings.

(a) Criteo Data Set

	Learning Rate	Click Learning Rate	Embedding	Hidden Units	Dropout
$l = 1$	$1e^{-6}$	$1e^{-7}$	256	512	0.5
$l = 2$	$1e^{-5}$	$1e^{-6}$	256	512	$1 - \sqrt{0.5}$
$l = 3$	$1e^{-5}$	$1e^{-6}$	256	512	$1 - \sqrt[3]{0.5}$

(b) Simulated Data

	Learning Rate	Click Learning Rate	Embedding	Hidden Units	Dropout
$l = 1$	$1e^{-6}$	$1e^{-7}$	16	256	0.5
$l = 2$	$1e^{-5}$	$1e^{-6}$	16	256	$1 - \sqrt{0.5}$
$l = 3$	$1e^{-5}$	$1e^{-6}$	16	256	$1 - \sqrt[3]{0.5}$

Adopting the methodology outlined in Ren et al. (2018), the gradients are clipped between 0 and 5 for the neural network. The chosen configuration includes a batch size of 256, and a regularisation parameter (λ) of $1e^{-6}$. The models are trained using the Adam optimiser (Kingma & Ba, 2014).

5.1 Criteo Data Set

5.1.1 Model Performance

To investigate the performance of the DARNN(+) model(s) Figure 9 is used, which illustrates the performance of the models for different depths (l). It can be seen that the DARNN+ model for $l = 2$ and $l = 3$ outperforms both the DARNN model on both the conversion AUC and the loss metric. Also note that for the loss metric, the train loss curve is above the test loss curve for the DARNN+ models. This is because the stacking of recurrent layers causes the model to have more weights which inflates the regularisation term in equation (19). A summary of the results are given in Table 2.

Table 2: Performance Metrics of the DARNN(+) model(s) after convergence. Flag indicates the epoch when the model starts optimising clicks and conversions jointly. The last column shows the number of epochs for the model to converge (Sect. 4.3.4). Train & Test losses are calculated using equations (10) and (19).

	Test Conversion AUC	Test Click AUC	Train Loss	Test Loss	Flag	Number of Epochs
$l = 1$	0.954	**0.893**	**0.426**	1.159	10	34
$l = 2$	**0.973**	0.498	1.097	**0.827**	8	28
$l = 3$	**0.973**	0.498	1.101	0.829	**6**	**17**

It follows that adding recurrent layers results in better performance in predicting conversions. This indicates that touchpoints have higher order interactions which cannot be captured by only using one recurrent layer. Models with extra recurrent layers perform worse when predicting clicks. Likely, because for the click-level prediction higher-order interactions are absent, and thus adding extra recurrent layers results in the model not being able to adequately predict clicks.

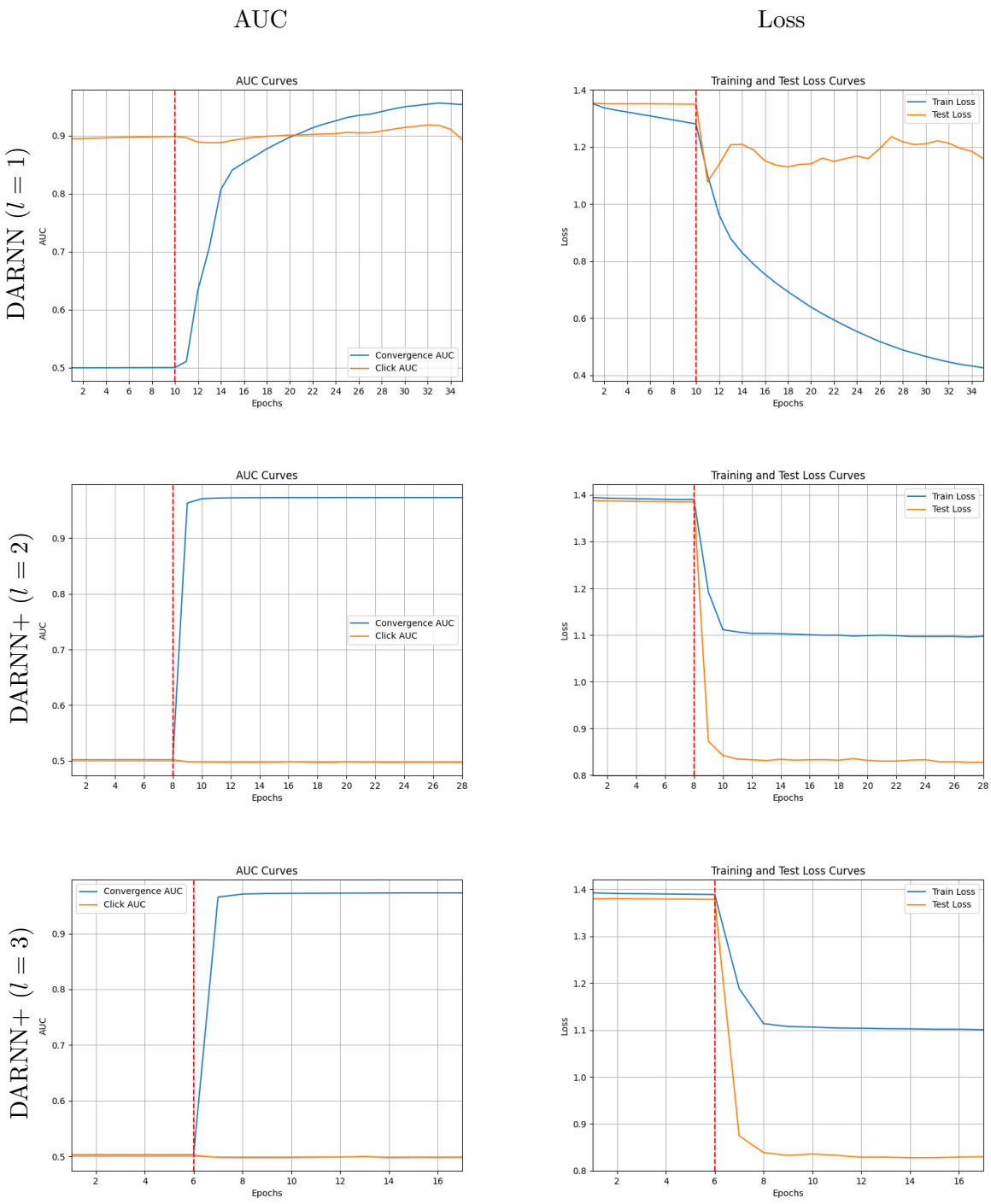


Figure 9: Test AUC scores and loss as described in Section 4.3 plotted against the number of epochs. The red line represents the epoch when the model starts minimising click and conversion loss jointly.

5.1.2 Calibration Plot

Figure 10 is used to investigate whether the predicted conversion probabilities are properly calibrated. Adding extra recurrent layers does indeed result in better calibrated conversion probabilities. This is crucial as the conversion probabilities are used to calculate the attribution measures outlined in Section 4.4.

$l = 1$ $l = 2$ $l = 3$

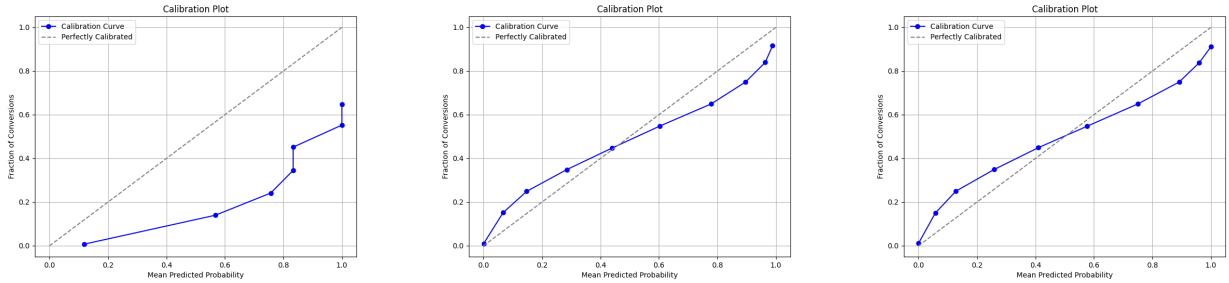


Figure 10: Calibration plots of the conversion probabilities on the test data for different depths (l) of the neural network as described in Section 4.3.

5.1.3 Attribution

Figure 11 illuminates the fractional scores for the attribution measures. In total, there are 599 channels that are part of a journey that leads to conversion. The remaining channels have not been assigned any conversion credits. Both the Simplified Shapley Value Method and the Attention Mechanism agree on the top five channels that are most conducive to a conversion for all depths (l). These channels are chosen for comparison.

The fractional scores assigned to each channel do not change for the different levels of depth of the Attention Mechanism. This is because the Attention Mechanism does not rely on conversion probabilities to calculate the fractional scores and thus is not heavily disturbed by the different conversion probabilities for each depth. For $l = 2$ and $l = 3$ the fractional scores are the same.

IVH does not agree with the Simplified Shapley Value Method and the Attention Mechanism on the five main channels. IVH also assigns negative conversion credits for $l > 1$ to channels that are considered relevant by the other attribution measures, and therefore IVH is not a suitable measure. This is likely due to confounding, which causes biased fractional scores. This becomes clear for $l = 3$. The magnitude of fractional scores and the order in which the channels are deemed relevant in contributing to a conversion are different from the other attribution measures.

The Simplified Shapley Value Method differs in the assignment of the fractional scores for each depth. However, there is a correspondence in the order in which the channels are regarded as important. Channels 15154511, 15398570, and 28351001 are always the third, fourth, and fifth most important channels, respectively. Channels 10341182 and 32368244 are the first and second most important channels for $l = 1$ and $l = 3$, but the opposite is true for $l = 2$. Nevertheless, as mentioned before, the group of these five channels is always in the top five regardless of the depth.

In contrast, having poorly calibrated conversion probabilities thus does not have a strong impact on the relative attribution. This can be explained by looking at equation (25). Even if the conversion predictions for channel c do not resemble the true conversion probabilities, the frequency at which channel c appears in a journey leading to a conversion potentially compensates for the bias in the conversion predictions. For example, the most important channel is expected to have high conversion predictions. If that is not the case, it could still be assigned the most conversion credits because it is more frequent in journeys that lead to a conversion relative to other channels. Therefore, the ranking of channels based on the given attribution could still be correct.

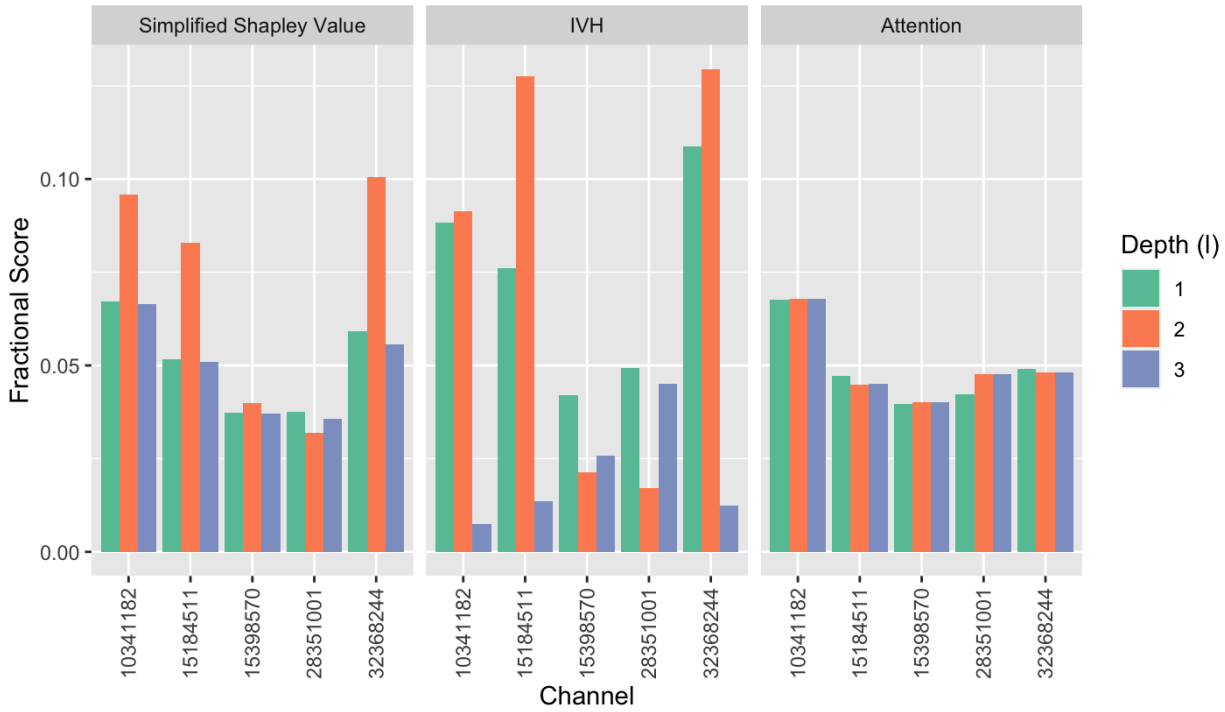


Figure 11: Comparison of Attribution Measures described in Sections 4.4.1 - 4.4.3, using fractional scores described in Section 4.4.4. The channel names are numbers and have no interpretation.

5.2 Simulated Data Set

The same approach described in Shender et al. (2020) is used to simulate users/journeys containing sequences of touchpoints with different browsing activities. The method in Shender et al. (2020) is adjusted to allow for click-activities.

Let N be the number of journeys/users and $C = \{1, 2, 3\}$ be the set of channels/ad types. Let $\mathbf{m}$ be a vector of length N of independent draws from a Poisson distribution with $\lambda_{ads} = 2$. $\mathbf{m}_i$ is the number of touch points per user. Clip $\mathbf{m}$ so that all events are clipped between 1 and 3. This means that each user is exposed to at least 1 ad and at most 3 ads. Therefore, $\mathbf{m}_i \in \{1, 2, 3\}$ for $i = 1, \ldots, N$. For each $i = 1, \ldots, N$ draw $\mathbf{m}_i$ times:

- t from a continuous uniform distribution on a 30-day window i.e. $[0, 30]$,
- u_{click} from a continuous uniform distribution on $[0, 1.0]$
- $u_{channel}$ to decide the ad type, which is drawn from a continuous uniform distribution on $[0, 1.0]$

$$z_{ij} = \begin{cases} 0 & \text{if } u_{click} \in [0, 0.5] \\ 1 & \text{if } u_{click} \in (0.5, 1] \end{cases} \qquad c_{ij} = \begin{cases} 1 & \text{if } u_{channel} \in [0, 0.33] \\ 2 & \text{if } u_{channel} \in (0.33, 0.66] \\ 3 & \text{if } u_{channel} \in (0.66, 1.0], \end{cases}$$

where z_{ij} dictates whether the touch point is click or non-click, and $c_{ij} \in C$ denotes the channel type of the touch point for $i = 1, \ldots, N$ and $j = 1, \ldots, m_i$. Then each touch point can be denoted as $\mathbf{q}_{ij} = (t_{ij}, z_{ij}, c_{ij})$.

For each journey/user draw $\alpha_{u_i} \sim \text{Uniform}(0, 0.025)$ Then similar to the approaches in Shender et al. (2020) and Yao et al. (2022), the log intensity function for each journey/user can be defined as

$$\begin{aligned} \log\big(\lambda_i(t)\big) = \alpha_{u_i} + \sum_{c=1}^{3}\sum_{k=1}^{3}\Big[&\gamma_{ck1} I\{\text{exactly } k \text{ type } c \text{ ads between } 0 < t - t_i \leq 1\} \\ &+\gamma_{ck2} I\{\text{exactly } k \text{ type } c \text{ ads between } 1 < t - t_i \leq 2\} \\ &+\gamma_{ck3} I\{\text{exactly } k \text{ type } c \text{ ads between } 2 < t - t_i \leq 30\}\Big] \\ + \sum_{c=1}^{3}\sum_{k=1}^{3}&[\delta_{ck} I\{\text{exactly } k \text{ type } c \text{ clickable-ads}\}]. \end{aligned} \tag{28}$$

Then the user conversion behaviour in MTA can be viewed as arrivals in an inhomogeneous Poisson counting process with time-varying intensity function $\lambda(t)$. This process can be described as

$$Y_i(t) - Y_i(s) \sim \text{Poisson}(\int_s^t \lambda_i(t)dt).$$

As each journey can contain at most 1 conversion, variable transformation is applied, i.e.,

$$X_i = \begin{cases} 1 & \text{if } Y_i(t) - Y_i(s) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

The probability that a conversion occurs in the $[0, 30]$ window can be formulated as $P(X_i = 1) = P(Y_i(30) - Y_i(0) > 0) = 1 - P(Y_i(30) - Y_i(0) = 0) = 1 - e^{-\Lambda_i}$, where $\Lambda_i = \int_0^{30} \lambda_i(t)dt$.

The simulated data involves generating 1,000,000 user sequences (N = 1,000,000). The parameters for the logarithmic intensity function in equation (28) are shown in Table 3 and Table 4.

Table 3: The gamma parameters ($exp(\gamma_{ckp})$, where p is the period) in equation (28).

		k = 1	k = 2	k = 3
c = 1	$0 < t - t_i \leq 1$	1.0	1.0	1.0
	$1 < t - t_i \leq 2$	0.75	0.75^2	0.75^3
	$2 < t - t_i \leq 30$	0.6	0.6^2	0.6^3
c = 2	$0 < t - t_i \leq 1$	0.75	0.75^2	0.75^3
	$1 < t - t_i \leq 2$	0.6	0.6^2	0.6^3
	$2 < t - t_i \leq 30$	0.55	0.55^2	0.55^3
c = 3	$0 < t - t_i \leq 1$	1.5	1.5^2	1.5^3
	$1 < t - t_i \leq 2$	1.25	1.25^2	1.25^3
	$2 < t - t_i \leq 30$	1.0	1.0	1.0

Table 4: The delta parameters ($exp(\delta_{ck})$) in equation (28).

	k = 1	k = 2	k = 3
c = 1	1.8	1.8^2	1.8^3
c = 1	1.44	1.44^2	1.44^3
c = 3	2.4	2.4^2	2.4^3

This results in a conversion rate of 0.3862. This means that 38.62% of the journeys lead to a conversion. The majority of the true conversion probabilities are between 0.3 and 0.4 and resemble the fraction of true conversions in the simulated data, which can be seen in Figure 12, which shows a histogram of the true conversion probabilities.

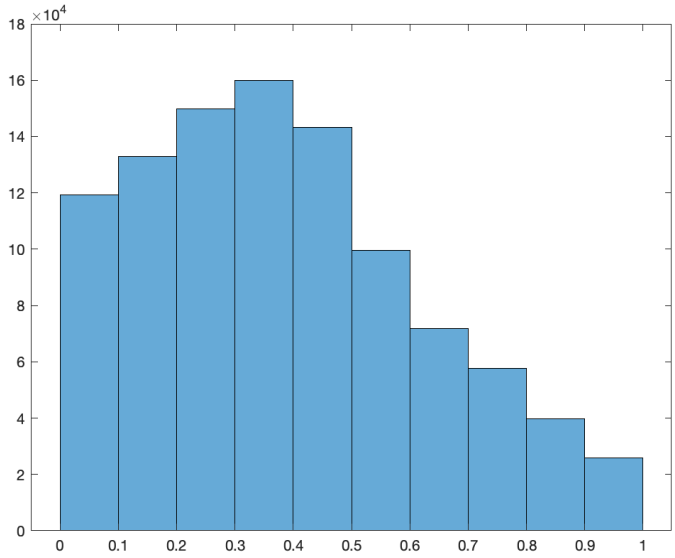


Figure 12: Histogram of the true conversion probabilities on the simulated data.

From the true conversion probabilities, the true conversion credits and fractional scores can be derived using the approaches in Section 4.4. These are shown in Figure 13. From this figure, the conversion credits assigned to each channel become clear. Channel 2 has negative attribution for the purpose of testing whether IVH can detect negative effects. Furthermore, channel 3 receives the most attribution after channel 1. From this figure it also becomes clear that a channel with high fractional scores is not necessarily desirable, as a channel with a high fractional score can potentially have negative conversion credits. True conversion credits and fractional scores are compared with the attribution given by the DARNN(+) model(s).

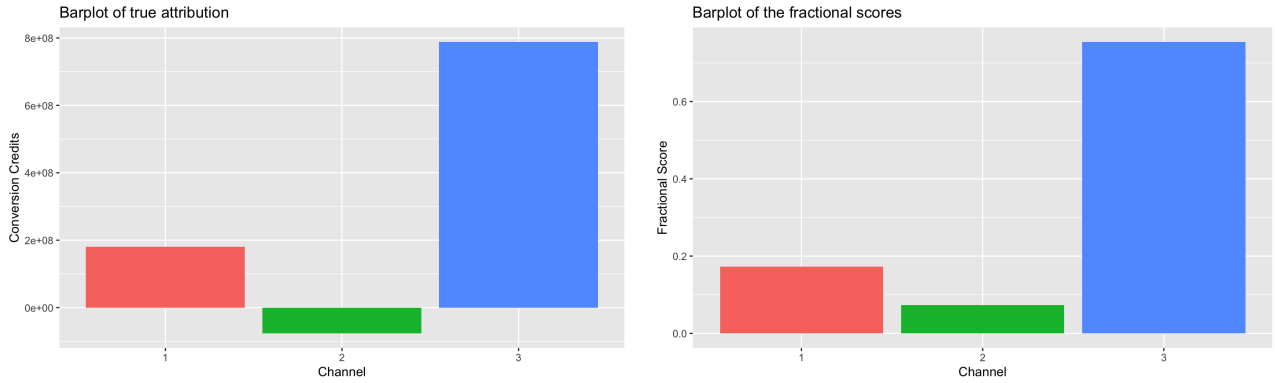


Figure 13: Barplots of the true attribution and fractional scores calculated using the methodology outlined in Section 4.4 on the full data set.

Before the simulated data is split into a train and test set, the t_{ij} of each touchpoint are scaled to ensure consitency with the Criteo data set. For each journey the t_{ij} are scaled using equation (29)

$$\tilde{t}_{ij} = \frac{t_{ij} - min(\mathbf{t}_i)}{30 - min(\mathbf{t}_i)}, \tag{29}$$

where $\mathbf{t}_i$ are all the time occurrences of all the touchpoints of user u_i. Scaling ensures that the first touchpoint in a sequence is always at time zero and that t_{ij} cannot exceed one. Finally, similar to the approach in Ren et al. (2018) for the Criteo data set, the simulated data is divided into a train and test set using a 80%/20% split. The data is simulated using Matlab R2020b.

5.2.1 Model Performance

Similarly to Section 5.1.1, Figure 14 is used to evaluate the performance of the DARNN(+) model(s) on the simulated data. The main findings are displayed in Table 5. The DARNN(+) model(s) perform worse on the simulated data than the Criteo data. An explanation could be that the models are not equipped to work with the assumptions relevant to the DGP, namely that conversions follow an inhomogenous Poisson process with time-varying intensity. Another explanation could be that a relatively small number of features are used to make predictions, namely time, click, and channel. Potentially, these features alone do not have enough explanatory power to make accurate predictions.

The DARNN model ($l = 1$) performs best on the click AUC metric; however, the DARNN+ models perform slightly better on the conversion AUC metric. It is expected that the DARNN model performs relatively well because higher-order interactions between (un)observable variables are absent in the DPG; hence, there is no need for extra recurrent layers. Overall, this is in line with the results in Table 5.

Table 5: Performance Metrics of the DARNN(+) model(s) after convergence on the simulated data. Flag indicates the epoch when the model starts optimising clicks and conversions jointly. The last column shows the number of epochs for the model to converge (Sect. 4.3.4). Train & Test losses are calculated using equations (10) and (19).

	Test Conversion AUC	Test Click AUC	Train Loss	Test Loss	Flag	Number of Epochs
$l = 1$	0.615	**0.850**	**0.821**	**1.081**	10	**23**
$l = 2$	**0.622**	0.499	1.352	1.342	10	76
$l = 3$	**0.622**	0.499	1.348	1.339	**5**	61

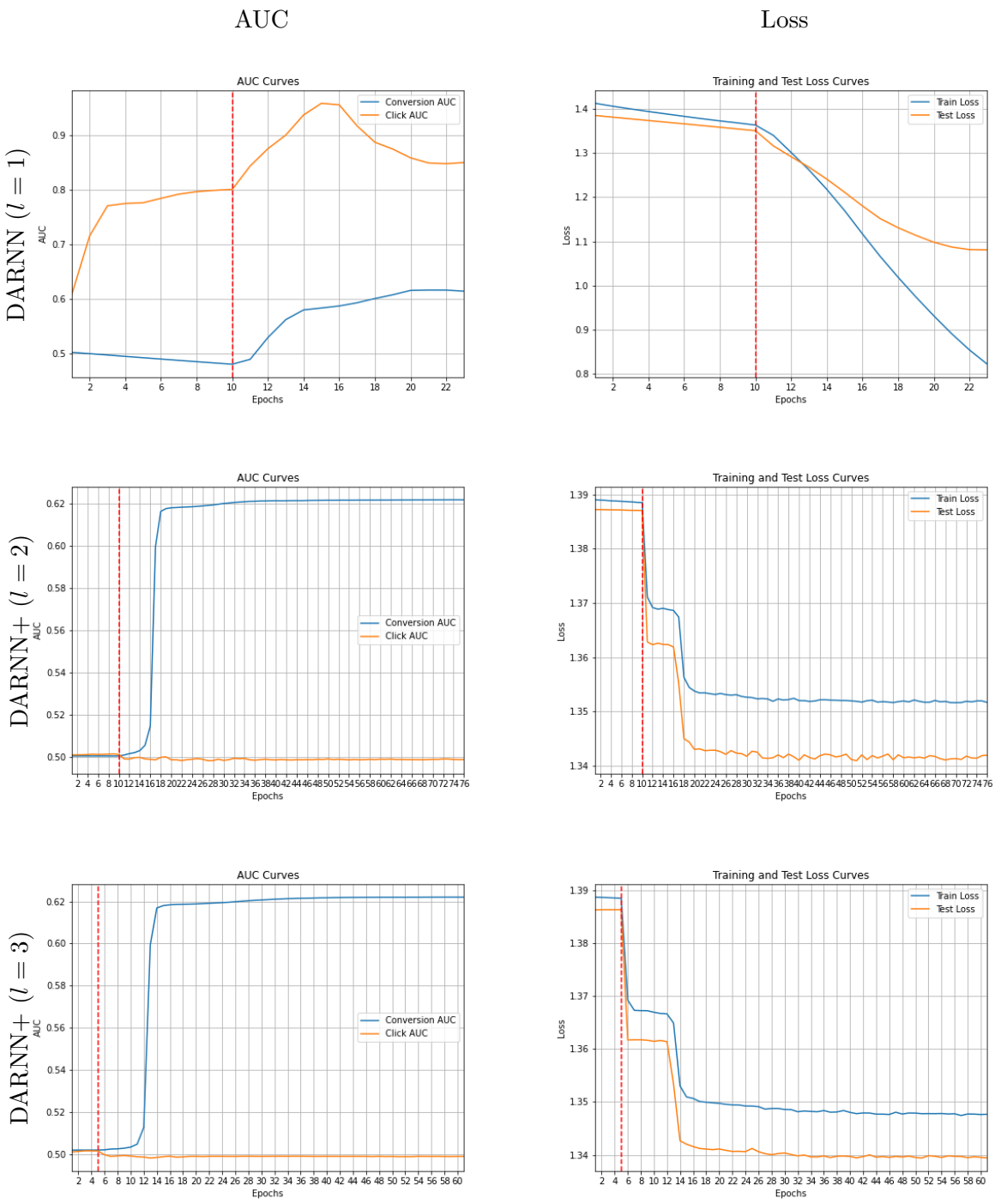


Figure 14: Test AUC scores and loss as described in Section 4.3 plotted against the number of epochs. Red line represents epoch when the model starts minimising click and conversion loss jointly.

5.2.2 Calibration Plot

Similar to the Criteo data set, the addition of extra recurrent layers results in better calibrated conversion probabilities. From Figure 15 it becomes evident that the models do not predict conversion probabilities lower than 0.3 and higher than 0.6 for $l = 1$ and $l = 2$. This could be the result of the model not being able to capture the DGP as described in Section 5.2.1. This also explains why the conversion probabilities are not as well calibrated as the conversion probabilities on the Criteo data set.

$l = 1$ $l = 2$ $l = 3$

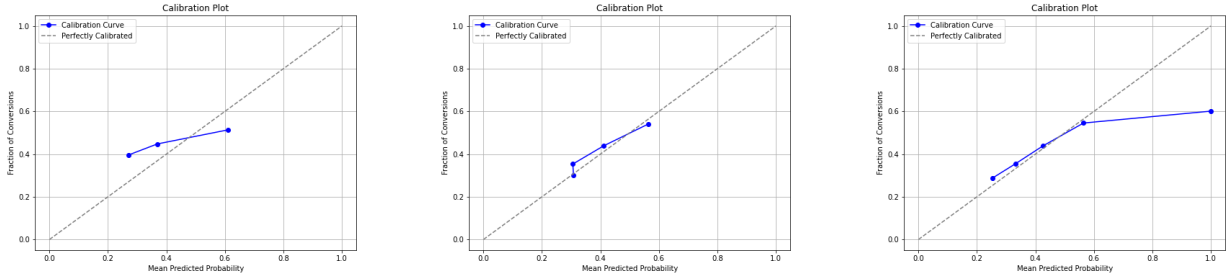


Figure 15: Calibration plots of the conversion probabilities on the simulated test data for different depths (l) of the neural network as described in Section 4.3.

5.2.3 Attribution

Figure 16 is used to investigate the causality of conversion credits acquired by the attribution measures described in Section 4.4. Similarly to the results in Section 5.1.3, the conversion credits acquired through the attention mechanism are not affected by the depth of the model. However, for IVH and the Simplified Shapley Value method, this is not the case. None of the attribution measures is able to detect the negative effects of the second channel. For IVH this is likely due to the model not being able to adequately capture the DGP. Also, the Attention Mechanism gives more conversion credits than the true attribution, whereas IVH gives a lot less. The Simplified Shapley Value method however is in between the Attention Mechanism method and IVH in terms of the total amount of conversion credits which are allocated.

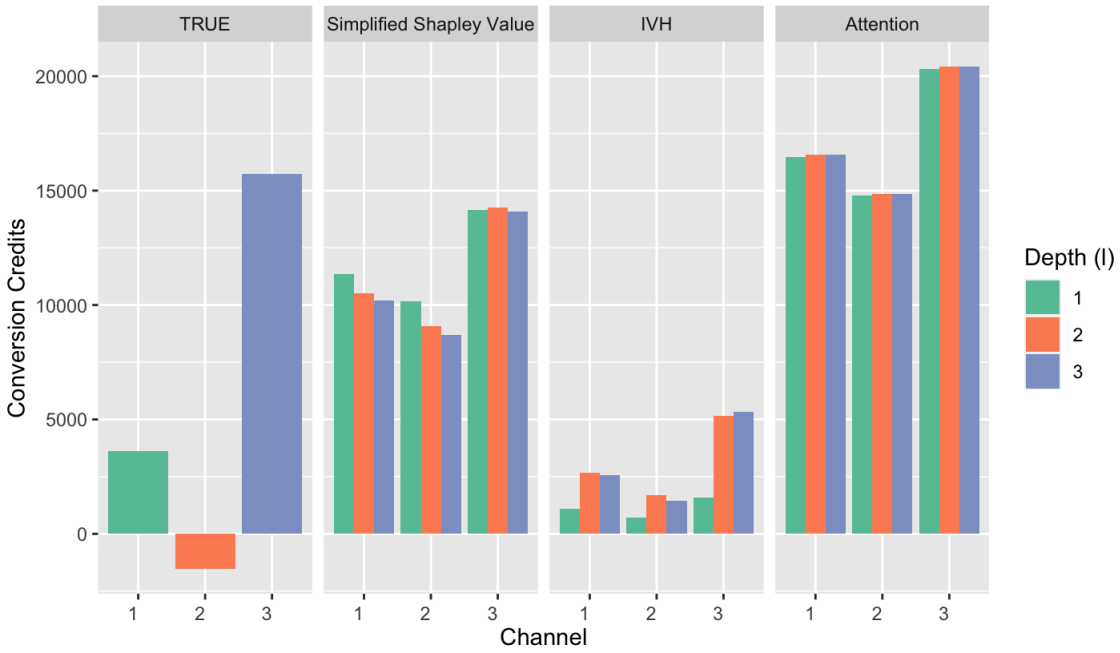


Figure 16: Conversion Credits on the test simulated data following the approach in Section 4.4. The legend does not apply for the TRUE conversion credits.

Figure 17 is used to investigate the relative attribution assigned by the attribution measures by means of fractional scores described in Section 4.4.4. From Figure 17 it follows that none of the attribution measures are correct in terms of the true fractional scores, but are correct in the order in which each channel is deemed important.

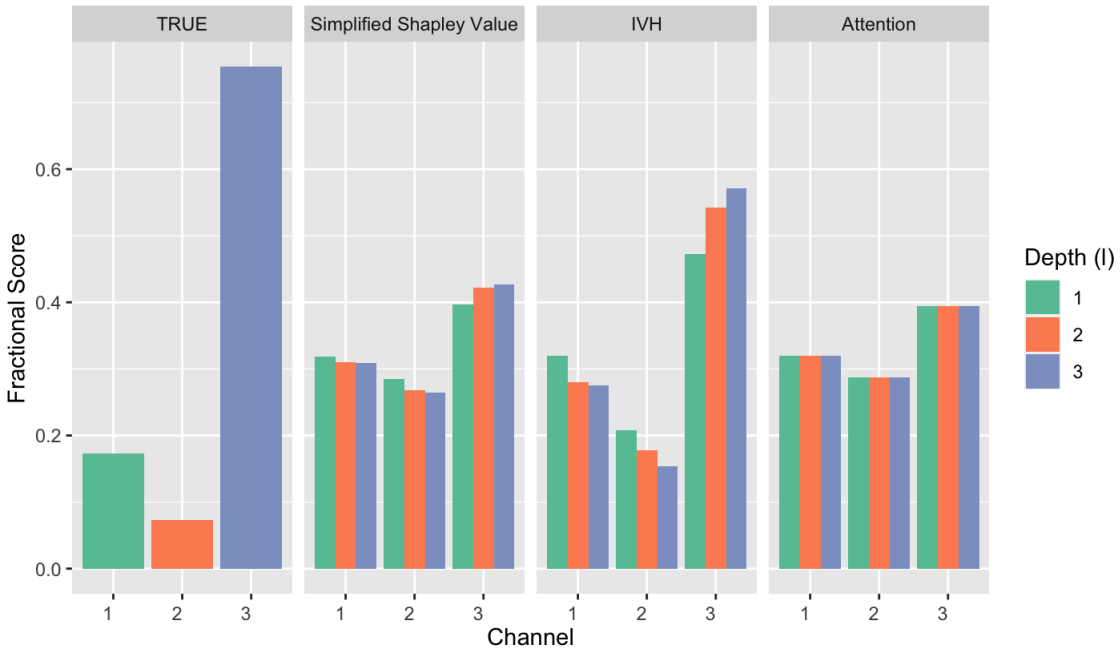


Figure 17: Fractional scores on the test simulated data following the approach in Section 4.4. The legend does not apply for the TRUE fractional scores.

6 Conclusion

This paper aims to investigate the following research question: *"How can a Neural Network be employed to derive causally interpretable conversion credits within the framework of Multi-Touch Attribution?"* The main findings indicate that the addition of stacked recurrent layers within a Dual-Attention mechanism improves the AUC on conversion predictions and results in better calibrated conversion probabilities. To address the issue of interpretability, attribution and attribution measures are defined directly as functions of conversion probabilities. The issue of causality remains evasive. The attribution measures outlined in this paper are not able to give the correct causal estimates when the model is not able to capture the DGP; however, they are able to provide a ranking of the channels which are most effective. Furthermore, the assessment of whether or not an attribution measure provides a causal estimate is heavily influenced by the definition of attribution, which has to be taken into consideration.

The practical and theoretical implications are as follows: marketeers and businesses can predict user conversions with relatively high accuracy, using the model described in this paper. If its attribution definition is based on the conversion probability of users, then using the model in this paper can provide better attribution estimates. Furthermore, if the emphasis is on the ranking of channels and not necessarily correct causal estimates, then the attribution measures discussed in this paper can also be insightful. The conversion credits obtained from the attribution measures can then be used by marketers to make informed decisions about their budget allocation.

Further research can investigate the use of different depths for the encoder and decoder within the Dual-Attention Mechanism. By doing this the model becomes more flexible and hence conversions and click predictions can be optimised and tuned separately. Additionally, further research can examine the use of an attribution measure that takes into account the ordering of touchpoints within a journey. Now, the assumption is made that the order in which touchpoints appear does not affect attribution, which does not necessarily have to be true. Lastly, the use of different proper scoring rules such as the Brier Score could also be investigated. Such scoring rules can be used as a loss function in this model and can also be used to assess the accuracy of the conversion and click predictions.

References

Arava, S. K., Dong, C., Yan, Z., Pani, A., et al. (2018). Deep neural net with attention for multi-channel multi-touch attribution. *arXiv preprint arXiv:1809.02230*.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Chollet, F. (2021). *Deep learning with python*. Simon and Schuster.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Dabre, R., & Fujita, A. (2021). Recurrent stacking of layers in neural networks: An application to neural machine translation. *arXiv preprint arXiv:2106.10002*.

Dalessandro, B., Perlich, C., Stitelman, O., & Provost, F. (2012). Causally motivated attribution for online advertising. In *Proceedings of the sixth international workshop on data mining for online advertising and internet economy* (pp. 1–9).

Devarakonda, A., Naumov, M., & Garland, M. (2017). Adabatch: Adaptive batch sizes for training deep neural networks. *arXiv preprint arXiv:1712.02029*.

Diemert Eustache, M. J., Galland, P., & 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* (p. To appear). ACM.

Du, R., Zhong, Y., Nair, H., Cui, B., & Shou, R. (2019). Causally driven incremental multi touch attribution using a recurrent neural network. *arXiv preprint arXiv:1902.00215*.

Dwarampudi, M., & Reddy, N. (2019). Effects of padding on lstms and cnns. *arXiv preprint arXiv:1903.07288*.

Eustache, D., Julien, M., Galland, P., & Lefortier, D. (2017). Attribution modeling increases efficiency of bidding in display advertising. In *Proceedings of the adkdd and targetad workshop, kdd, halifax, ns, canada.*

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2008). A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, *31*(5), 855–868.

Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural

networks. In *2013 ieee international conference on acoustics, speech and signal processing* (pp. 6645–6649).

Hermans, M., & Schrauwen, B. (2013). Training and analysing deep recurrent neural networks. *Advances in neural information processing systems*, *26*.

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

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, *79*(8), 2554–2558.

Huang, J., & Ling, C. X. (2005). Using auc and accuracy in evaluating learning algorithms. *IEEE Transactions on knowledge and Data Engineering*, *17*(3), 299–310.

Ji, W., & Wang, X. (2017). Additional multi-touch attribution for online advertising. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 31).

Johannemann, J., Hadad, V., Athey, S., & Wager, S. (2019). Sufficient representations for categorical variables. *arXiv preprint arXiv:1908.09874*.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kumar, S., Gupta, G., Prasad, R., Chatterjee, A., Vig, L., & Shroff, G. (2020). Camta: Causal attention model for multi-touch attribution. In *2020 international conference on data mining workshops (icdmw)* (pp. 79–86).

Learning Multi-touch Conversion Attribution with Dual-attention Mechanisms for Online Advertising. (2018). (PDF download)

Leong, L.-Y., Hew, T.-S., Ooi, K.-B., & Dwivedi, Y. K. (2020). Predicting trust in online advertising with an sem-artificial neural network approach. *Expert Systems with Applications*, *162*, 113849.

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*.

Lipton, Z. C., Kale, D. C., Elkan, C., & Wetzel, R. (2015). Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*.

Neil, D., Pfeiffer, M., & Liu, S.-C. (2016). Phased lstm: Accelerating recurrent network training for long or event-based sequences. *Advances in neural information processing systems*, *29*.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International conference on machine learning* (pp. 1310–1318).

Ren, K., Fang, Y., Zhang, W., Liu, S., Li, J., Zhang, Y., ... Wang, J. (2018). Learning multi-touch conversion attribution with dual-attention mechanisms for online advertising. In *Proceedings of the 27th acm international conference on information and knowledge management* (pp. 1433–1442).

Shao, X., & Li, L. (2011). Data-driven multi-touch attribution models. In *Proceedings of the 17th acm sigkdd international conference on knowledge discovery and data mining* (pp. 258–264).

Shapley, L. S., et al. (1953). A value for n-person games.

Shender, D., Amini, A. N., Bao, X., Dikmen, M., Richardson, A., & Wang, J. (2020). A time to event framework for multi-touch attribution. *arXiv preprint arXiv:2009.08432*.

Singal, R., Besbes, O., Desir, A., Goyal, V., & Iyengar, G. (2019). Shapley meets uniform: An axiomatic framework for attribution in online advertising. In *The world wide web conference* (pp. 1713–1723).

Srivastava, N. (2013). Improving neural networks with dropout. *University of Toronto*, *182*(566), 7.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, *27*.

Van Laarhoven, T. (2017). L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*.

Verdinelli, I., & Wasserman, L. (2023). Feature importance: A closer look at shapley values and loco. *arXiv preprint arXiv:2303.05981*.

Vuk, M., & Curk, T. (2006). Roc curve, lift chart and calibration plot. *Metodoloski zvezki*, *3*(1), 89.

Yang, D., Dyer, K., & Wang, S. (2020). Interpretable deep learning model for online multi-touch attribution. *arXiv preprint arXiv:2004.00384*.

Yao, D., Gong, C., Zhang, L., Chen, S., & Bi, J. (2022). Causalmta: Eliminating the user confounding bias for causal multi-touch attribution. In *Proceedings of the 28th acm sigkdd conference on knowledge discovery and data mining* (pp. 4342–4352).

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, *31*(7), 1235–1270.

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into deep learning. *arXiv preprint arXiv:2106.11342*.

Zhang, Y., Wei, Y., & Ren, J. (2014). Multi-touch attribution in online advertising with survival

theory. In *2014 ieee international conference on data mining* (pp. 687–696).

Zhao, K., Mahboobi, S. H., & Bagheri, S. R. (2018). Shapley value methods for attribution modeling in online advertising. *arXiv preprint arXiv:1804.05327*.