

Mena, Gary; Coussement, Kristof; De Bock, Koen W.; De Caigny, Arno; Lessmann, Stefan

Article — Published Version

Exploiting time-varying RFM measures for customer churn prediction with deep neural networks

Annals of Operations Research

Provided in Cooperation with:

Springer Nature

Suggested Citation: Mena, Gary; Coussement, Kristof; De Bock, Koen W.; De Caigny, Arno; Lessmann, Stefan (2023) : Exploiting time-varying RFM measures for customer churn prediction with deep neural networks, Annals of Operations Research, ISSN 1572-9338, Springer US, New York, NY, Vol. 339, Iss. 1, pp. 765-787,
<https://doi.org/10.1007/s10479-023-05259-9>

This Version is available at:

<https://hdl.handle.net/10419/317737>

Standard-Nutzungsbedingungen:

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

Terms of use:

Documents in EconStor may be saved and copied for your personal and scholarly purposes.

You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.


If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.



<http://creativecommons.org/licenses/by/4.0/>



Exploiting time-varying RFM measures for customer churn prediction with deep neural networks

Gary Mena¹ · Kristof Coussement² · Koen W. De Bock³ · Arno De Caigny² · Stefan Lessmann¹ 

Accepted: 22 February 2023 / Published online: 21 March 2023
© The Author(s) 2023

Abstract

Deep neural network (DNN) architectures such as recurrent neural networks and transformers display outstanding performance in modeling sequential unstructured data. However, little is known about their merit to model customer churn with time-varying data. The paper provides a comprehensive evaluation of the ability of recurrent neural networks and transformers for customer churn prediction (CCP) using time-varying behavioral features in the form of recency, frequency, and monetary value (RFM). RFM variables are the backbone of CCP and, more generally, customer behavior forecasting. We examine alternative strategies for integrating time-varying and non-variant customer features in one network architecture. In this scope, we also assess hybrid approaches that incorporate the outputs of DNNs in conventional CCP models. Using a comprehensive panel data set from a large financial services company, we find recurrent neural networks to outperform transformer architectures when focusing on time-varying RFM features. This finding is confirmed when time-invariant customer features are included, independent of the specific form of feature integration. Finally, we find no statistical evidence that hybrid approaches (based on regularized logistic regression and extreme gradient boosting) improve predictive performance—highlighting that DNNs

✉ Stefan Lessmann
stefan.lessmann@hu-berlin.de

Gary Mena
gary.mena@bdpems.de

Kristof Coussement
k.coussement@ieseg.fr

Koen W. De Bock
kdebock@audencia.com

Arno De Caigny
a.de-caigny@ieseg.fr

¹ School of Business and Economics, Humboldt-University of Berlin, Unter den Linden 6, 10099 Berlin, Germany

² IESEG School of Management, Univ. Lille, CNRS, UMR 9221 - LEM - Lille Economic Management, 59000 Lille, France

³ Audencia Business School, 8 Route de La Jonelière, 44312 Nantes, France

and especially recurrent neural networks are suitable standalone classifiers for CCP using time-varying RFM measures.

Keywords Financial services · Customer churn · Deep learning · Panel data · Time-varying features · RFM · Recurrent neural networks · Transformers · Attention · GRU · LSTM

1 Introduction

Customers are a critical asset for every company operating in a contractual setting. The ability to retain profitable customers is a significant determinant of customer equity, i.e., the total lifetime value of a company's customers (McCarthy et al., 2017). Consequently, customer retention is a strategic imperative (Rust et al., 2004) and customer churn prediction (CCP) models are a crucial tool for data-driven customer relationship management (Wu et al., 2022).

CCP models exploit data at the individual level (e.g., demographic, socio-economic, behavioral data, etc.) to predict whether customers will terminate an existing business relationship within a future time window. These models help companies anticipate and remedy decreases in the stream of cash flows associated with customer churn. Moreover, CCP models can help uncover the underlying drivers of churn or decaying customer loyalty. Such insights are useful to revisit business processes and service offerings and raise customer equity in the long run. Further applications include the estimation of customer lifetime value, which relies on high-quality estimates of customer retention (Schweidel et al., 2014). Finally, CCP models can be deployed to govern the targeting of retention campaigns, either as standalone solutions or as a component of (causal) uplift models (Janssens et al., 2022).

The increasing availability of customer data combined with lower costs of data storage and computational infrastructure fostered the use of supervised machine learning (ML) models to predict customer churn over the past decades (Verbeke et al., 2012). The ability to scale well with high-dimensional data (e.g., an increasing number of customers and/or customer features) and to capture complex, non-linear dependencies between features and the churn event make ML models the tool of choice for CCP in both, industry and academic applications.

The temporal aspects of features have an impact on the performance of customer churn prediction models. Risselada et al. (2010) observed how the predictive performance of different types of CCP models deteriorates quickly over time and suggest the development of dynamic models. To make models more generalizable, Gattermann-Itschert and Thonemann (2021) suggest multi-slicing, an approach for training CCP models on data that is composed of different parts covering different time horizons. This paper takes an orthogonal approach. We develop dynamic CCP models that incorporate time-varying behavioral customer features in the form of recency, frequency, and monetary value (RFM).

The marketing literature emphasizes the predictive power of RFM features (Zhang et al., 2015) and many prior studies on CCP have considered corresponding predictors (Janssens et al., 2022). However, conventional statistical or ML classifiers such as logistic regression or tree-based models do not readily accommodate time-varying features. They assume that observations are independently and identically distributed. This collides with the nature of customer-level time-series data. Hence, the processing of time-varying data requires a non-trivial effort of manual feature engineering or aggregation, which potentially hinders the predictive performance of ML models. The process of mapping time series data to a fixed-length feature vector is not only labor intensive, but it also results almost always in a loss of

information. Recent developments in deep neural networks (DNN) architectures for sequential data have the potential to overcome the problems inherent to shallow ML models to exploit time-varying data. Given a comprehensive space of architectural choices, the design of a DNN-based CCP model that accommodates both, time-varying and time-invariant features is not straightforward. Moreover, recent empirical evidence in related classification tasks in the financial industry suggests that deep learning models might not outperform simpler alternatives based on gradient boosting models (Gunnarsson et al., 2021). Transferring corresponding results to the CCP setting, it is important to examine whether purely DNN-based models are effective and whether the respective merits of DNNs to handle time-varying features and conventional (e.g., gradient-boosting-based) classifiers can be integrated to obtain a hybrid—best-of-breed—solution.

This paper contributes to the literature by empirically testing the potential of DNNs to predict customer churn using time-varying RFM features in the financial services industry. We systematically explore the vast option space of model architectures at the interface of deep vs. shallow and static vs. dynamic churn models and their different forms of integration. This offers churn analysts valuable guidance on how to capitalize on available customer data with both, time-varying and time-invariant features. Our experiments are based on a unique data set from a European financial service provider that encompasses anonymized information of 480 thousand customers collected over 48 months. We find that recurrent neural networks outperform transformer models for CCP using time-varying RFM measures. This finding is confirmed when other, time-invariant customer features enter a CCP model, independent of how different sets of features are integrated. Finally, we find no statistical evidence that hybrid approaches, which integrate DNN predictions in conventional classifiers (based on regularized logistic regression and extreme gradient boosting) improve performance further—highlighting that DNNs are suitable standalone classifiers for predicting churn using time-varying RFM measures.

The paper is organized as follows. Section 2 provides a review of related work on classic, ML based-models, and deep learning approaches to predict customer churn. Sections 3 and 4 describe the data and experimental design, respectively. We report empirical results in Sect. 5 and conclude thereafter.

2 Related literature

A large body of CCP literature comprises different fields of study including operations research, marketing, statistics, and computer science. The promise of increased predictive accuracy and the requirements of operational churn management to handle data sets with a vast number of customer characteristics, multi-collinearity problems, and noisy features have raised a considerable amount of interest in ML-based churn prediction (Janssens et al., 2022; Qi et al., 2009; Wu et al., 2022). Our paper contributes to the existing CCP literature by exploring the beneficial impact of using time-varying over static customer characteristics and investigating the potential of DNN.

To date, the majority of CCP literature employs models for large-scale cross-sectional and static data. Verbeke et al. (2012) provide a comprehensive comparison of the performance of ML models, which highlights that most previous studies focus on cross-sectional data. More recently, Janssens et al. (2022) propose a novel expected maximum profit measure for B2B churn prediction to directly incorporate the heterogeneity of customer values and profit concerns of the company using gradient boosting on a large cross-sectional data set from a North

American B2B beverage retailer. Another recent exemplary study proposes combining PCA analysis with AdaBoost to deliver an enhanced and more stable churn prediction performance in a cross-sectional e-commerce context (Wu et al., 2022). As an exception, Chen et al. (2012) extend the support vector machine model to accommodate time-varying variables and predict customer churn without prior feature aggregation. More generally, the incorporation of static features from time-varying information requires the use of heuristic aggregation procedures such as moving or weighted averages or augmenting the set of features with time-lagged observations of these time-varying variables. These techniques are well-established in the telco-industry (Wei & Chiu, 2002) and e-commerce (Koehn et al., 2020) to predict customer behavior based on call-detail records and click-stream data, respectively. This motivates us to investigate the merit of directly modeling customer churn using time-varying features and DNNs.

DNNs have witnessed rising popularity in the CCP literature to leverage novel data sources like social-network features (Óskarsdóttir et al., 2017) or text data (De Caigny et al., 2020). On the contrary, the use of DNN to extract information from time-varying (RFM) features is sparsely explored in prior work. Table 1 summarizes related work that has used time-varying features and DNN in a CCP context.

Closely related to our work, several studies have experimented with time-varying features and deep learning, but none of these studies include time-varying RFM variables. Tan et al. (2018) and Zhou et al. (2019) obtain a churn model by combining convolutional (CNN) and long short-term memory (LSTM) neural networks. Both studies report that their models outperform benchmarks that do not use time-varying information. Both papers distinguish between static and time-varying features, yet they do not study how to best combine the two types of features in a prediction model. Results in Wangperawong et al. (2016) and Zaratiegui et al. (2015) from applying CNNs for churn prediction provide further empirical

Table 1 Prior work on churn prediction using time-varying features and DNN

Study	Model	Data set size (number of customers)	Industry	Time-varying features
Liu et al. (2018)	Dynamic embeddings	4×10^4	Mobile gaming	Opens, closes, installs, and uninstalls
Tan et al. (2018)	LSTM + CNN	12×10^4 and 15.6×10^4	MOOC, online services	Subscription characteristics
Wangperawong et al. (2016)	CNN	100×10^4	Telecom	Data, voice, sms usage
Yang et al. (2018)	LSTM + k-means	100×10^4	Social network	Daily data usage
Zaratiegui et al. (2015)	CNN	13.2×10^4	Telecom	Call record, Topup
Zhou et al. (2019)	LSTM + CNN	100×10^4	Music stream- ing	Transaction and log activity
This study	GRU/LSTM/attention/transformer	4820×10^4	Fin. services	RFM variables

evidence regarding the merit of deep learning-based churn models and the use of time-varying features. Results in these studies, however, are based on a limited number of variables from a relatively short, few-month timeframe. Wangperawong et al. (2016) also consider time-varying features alone while most real-world CCP applications involve both, time-varying and invariant features. Further applications that combine supervised and unsupervised approaches for churn prediction in dynamic contexts include Liu et al. (2018) and Yang et al. (2018). Liu et al. (2018) combine time-varying and static features but do not distinguish between the two types of features. They also consider a relatively short time frame of four months for their time-varying features. Yang et al. (2018) consider an even shorter time window of two weeks.

In conclusion, our study is the first to examine the potential of behavioral time-varying variables to predict churn using DNN in the financial services industry. Compared to existing work, our study relies on a much longer period of time-varying features from multiple years. Unlike prior work, in which temporal data is represented by some application-specific data, RFM variables are a well-established, widely used, and generic class of features in marketing decision-support (Zhang et al., 2015). Moreover, our thorough assessment of DNN and RFM variables contributes original empirical evidence on which architectures obtain better performance for CCP. For instance, transformer networks, a state-of-the-art deep learning approach, are, to our best knowledge, for the first time tested in a CCP context in this paper. Further, we propose modeling frameworks that can be combined with existing customer churn prediction, and we investigate different options on how to best combine static and time-varying features. Based on the analysis of prior literature, we intend to answer the following research questions:

RQ1: what is the most effective DNN architecture for CCP using time-varying RFM measures?

RQ2: what is the most effective DNN architecture for CCP using mixed data (time-varying RFM measures and static customer variables)?

RQ3: can the performance of a mixed-feature DNN model be improved further by hybridizing it with a conventional CCP classifier?

3 Methodology

The methodology described in this study is graphically depicted in Fig. 1. The models we compare differ on two dimensions: (1) the features on which they are trained and (2) the algorithms deployed to train classifiers. First, we explain the difference between time-varying and static features in Sect. 3.1. Next, in Sect. 3.2.1, we introduce DNNs that can handle time-varying input, followed by strategies to combine static and time-varying inputs in these DNNs. Last, we introduce hybrid approaches that incorporate the output of the DNNs in traditional classifiers in Sect. 3.2.2.

3.1 Static and time-varying features

Our methods presume two categories of features exist. The first category involves *static* customer variables, which remain constant over time (e.g. demographic characteristics). The second category includes *time-varying* measures of recency, frequency, and monetary value. These time-varying RFM variables are available on a monthly level and are derived from

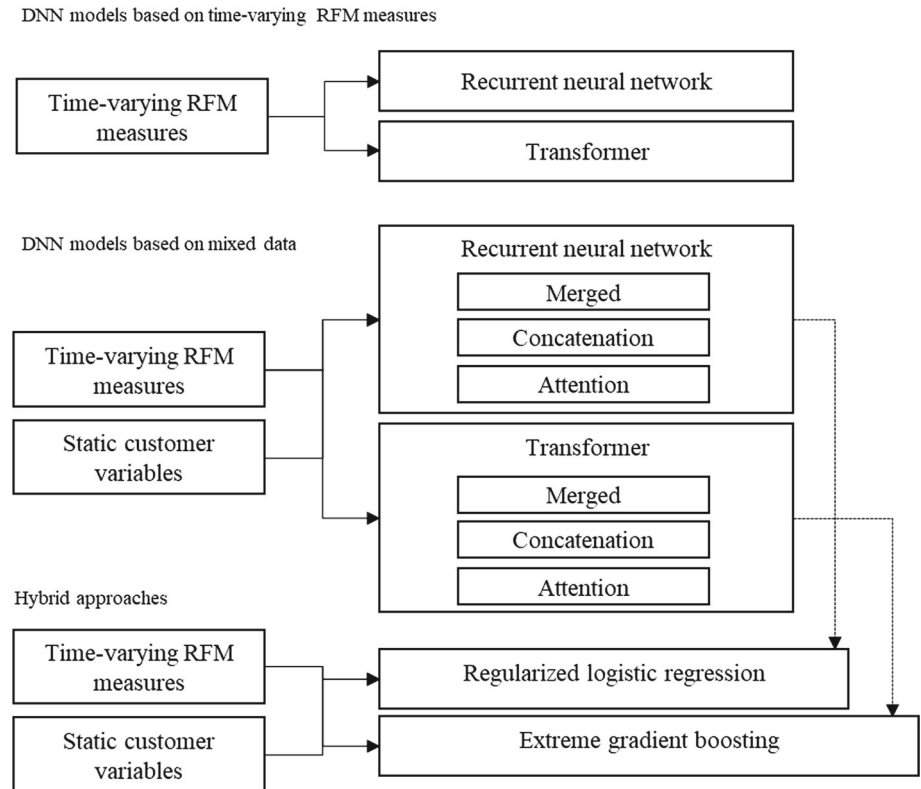


Fig. 1 Visual representation of variable sets and classifiers considered in this study

ongoing product contracts. In line with existing research, we calculate recency as the number of days that have passed since the last new product was subscribed, frequency as the number of open contracts on a given date, and monetary value as the total monthly value associated with a customer's open product portfolio. Figure 2 provides an example of the calculation and the time-varying aspect of RFM variables. Note how, after a contract is opened or closed, or if time passes by, the values of recency, frequency, and monetary value, are modified.

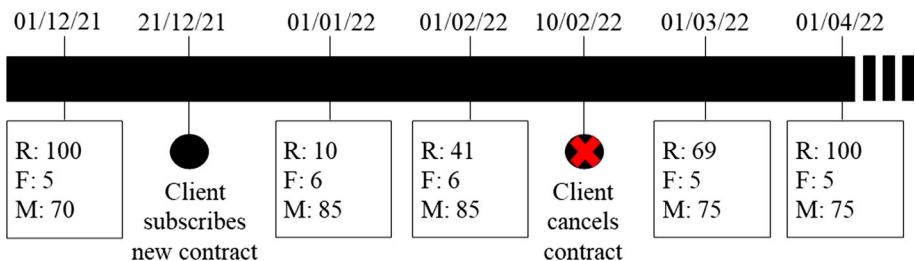


Fig. 2 Example of the outcome and RFM calculation for a fictitious customer

3.2 Classification algorithms

3.2.1 Deep neural networks

We consider recurrent models, with or without attention mechanisms, and transformer-type neural network architectures. In this subsection, we sketch the intuition behind these models while emphasizing the decisions required to model time-varying and static variables.

Recurrent neural networks are, by design, well-suited for time-varying input variables. Consider a vanilla recurrent neural network (RNN) and assume that we observe, for each customer i , a sequence of features X_1, \dots, X_t of fixed length T . Note that this architecture depends on the current value of the variables X_t and is dynamic since the hidden state h_t depends on its past value, which, in turn, incorporates information extracted from previous realizations of the variables, e.g., $h_{t-1} X_{t-1}$. Each hidden state h_t processes the information for time step t . The hidden state for period t , with $t = 1, \dots, T$ is given by:

$$h_t = \tanh(W_x X_t + W_h h_{t-1}) \quad (1)$$

with $\tanh()$ as the hyperbolic tangent function.¹ To predict the probability of churn, we add a dense layer with a sigmoid activation function. This allows for estimating the free parameters, that is the connection weight matrices W_x and W_h , by computing the gradient of the loss L , defined by the binary cross-entropy function, over N customers:

$$L = \sum_{i=1}^N [-y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i)] \quad (2)$$

For notational convenience, the rest of the paper uses the symbol W to refer to a properly shaped matrix of all the weights in the network. For the estimation of the weights W , the hidden state for the first period is usually set to zero to proceed with the estimation ($h_0 = 0$). We refer to Goodfellow et al. (2016) for an overview of the backpropagation algorithm and optimization-specific details.

The vanilla RNN may suffer from the exploding or vanishing gradient problem, which in turn degrades the ability of RNNs to learn long-term dependencies. Gated Recurrent Unit (GRU) (Cho et al., 2014) and Long Short-Term Memory (Hochreiter and Schmidhuber 1997)² neural networks address these issues. Both architectures introduce gating mechanisms that facilitate a different flow of information in the network. There is empirical evidence that both, the GRU and LSTM architecture, offer comparable performance and that both perform better in sequence modeling compared to the vanilla RNN (Chung et al., 2014). Thus, we focus on these models. Formally, the GRU architecture is given by the following components:

$$\begin{aligned} r_t &= \sigma(W_r X_t + W_{hr} h_{(t-1)}), \quad n_t = \tanh(W_n X_t + r_t (W_{hn} h_{(t-1)})) \\ z_t &= \sigma(W_z X_t + W_{hz} h_{(t-1)}), \quad h_t = (1 - z_t) n_t + z_t h_{(t-1)} \end{aligned}$$

where r_t , z_t , and n_t are three elements that represent the reset, update, and new information gates.³ Note how the reset and update gates are similar to the structure of the vanilla recurrent network except for the use of a sigmoid function instead of the hyperbolic tangent. Similarly, the new information gate n_t resembles the vanilla recurrent network with the extra characteristic that the reset gate multiplies the previous hidden state. Despite these similarities, the

¹ Subscript i is suppressed to ease the presentation.

² Several variants of the GRU and LSTM model exist in the literature. We base our explanation on the canonical models for LSTM and GRU.

³ We do not display the associated bias terms in the gates to ease the presentation.

hidden state h_t in the GRU model is computed as a weighted average of the previous hidden state h_{t-1} and the new information n_t where the weight, in turn, depends on z_t —a function of the original inputs X_t and h_{t-1} .

Consider as an alternative the LSTM architecture given in Eq. (4):

$$\begin{aligned} i_t &= \sigma(W_{ii}X_t + W_{hi}h_{(t-1)}), & g_t &= \tanh(W_{ig}X_t + W_{hg}h_{(t-1)}) \\ f_t &= \sigma(W_{if}X_t + W_{hf}h_{(t-1)}), & c_t &= f_t c_{(t-1)} + i_t g_t \\ o_t &= \sigma(W_{io}X_t + W_{ho}h_{(t-1)}), & h_t &= o_t \tanh(c_t) \end{aligned} \quad (4)$$

This architecture uses a cell state represented by c_t and a weight given by the output gate o_t to compute the hidden state h_t . The cell state is a weighted sum of the cell state in the previous period c_{t-1} and an updated cell state proposal denoted by g_t , where the weights are given by the input and forget gates i_t , and f_t .

We consider variations of DNNs for sequential data in terms of (1) using the entire sequence of hidden states h_t or just a subset, (2) using bidirectional models, (3) using separate input layers for static and time-varying data, and the consideration of attention mechanisms.

First, in RNN, the hidden state h_t for time step t summarizes all the information up to that point of the sequence. Hence, we can extract information from the entire sequence using only the last hidden state. Alternatively, we can use the entire sequence of hidden states to estimate churn, which could improve the quality of the estimated probabilities at the cost of adding parameters in the final layer.

Next, RNNs allow for reversing the order of the sequence of X_1, \dots, X_t which facilitates bidirectional modeling. This is relevant in our context because a bidirectional model can better capture differences between the customers that arise earlier in time. Again, it is possible to consider the entire sequence of bidirectional hidden states h_t or just the latest one to compute probabilities.

Last, the inclusion of static variables in the model is not straightforward. First, we could merge the sequential and static variables and include them together in the X_t matrix. We call this approach *Merging* in our experiments. We thus merge the static data with the time-varying data at the input level, having at each timepoint a value of the static feature, similar to the time-varying variables. This implies that the dimension of the weights W increases considerably as the sequence also consists of static features. Moreover, given that static features by definition do not change over time, the static features might preclude the model from fully exploiting the sequential patterns in the data. As a second alternative to consider static and time-varying data, we could divide the features into two subsets and use DNNs designed for each type of data. We refer to this approach as *Concatenation* and consider RNN only with the subset of sequential variables, and concatenate the hidden states from this model with the hidden states from a feedforward neural network for the static data.⁴ In other words, such an approach concatenates the hidden representations of the time-varying features at point T , with the static features. In theory, such an approach would allow the architecture to specialize and better exploit the two types of variables. As a final variation, we consider the use of attention mechanisms with the concatenation approach. Attention allows the architecture to weight different points in the sequence to make the predictions such that the final hidden state is not the only component summarizing the sequence. The literature offers several alternatives to introduce attention mechanisms (Chaudhari et al., 2021; Galassi et al., 2021). We focus on a global attention type as described in Luong et al. (2015). Compared to local attention, global attention is more expensive since it considers

⁴ This is similar in spirit to a sequence-to-sequence architecture (Sutskever et al., 2014) except that we use a feedforward model instead of a recurrent model in the decoder.

all hidden states of the encoder when deriving the context vector instead of only a subset of hidden states. However, if sequences are relatively short, as in the focal study, global attention is relatively easy to train and offers good performance in natural language processing tasks. We refer to this model variation as *Attention*.

The transformer architecture is a different kind of DNN to model sequential data. It is designed to overcome the computational burden of RNNs by relying only on the attention mechanism. RNNs generate a sequence of hidden states, h_t as a function of previous hidden states, h_{t-1} , and the input to the hidden state at position t (Vaswani et al., 2017). As sequences become longer, the inherently sequential nature of RNNs precludes parallelization with a major impact on the computation time. The transformer architecture relies on encoder- and decoder stacks that embed the input and output sequences. The attention mechanism is applied within the layers of the encoder- and decoder stacks. An attention function basically maps a query vector and a set of key-value pairs vectors to an output vector. Explaining the inner works of the transformer model is beyond the scope of this paper and we refer instead to the original implementation in Vaswani et al. (2017), and the notes in Rush (2018). For the objective of this paper, it is sufficient to understand that the self-attention mechanism in the transformer model allows us to encode the sequential data. Like recurrent models, we can incorporate static variables using merge or concatenation approaches.

3.2.2 Hybrid approaches

In addition to evaluating the potential of standalone deep neural networks and comparing alternative architectures, we assess the potential of hybrid approaches, combining DNN and ML models. Specifically, similar to earlier approaches (De Caigny et al., 2020), we construct ML classifiers that incorporate DNN predictions as features. This approach investigates the interest in deploying DNNs that are purpose-built to accommodate time-varying RFM features into existing CCP models. We focus on two ML models with a proven track record in the CCP literature: (1) the regularized (lasso) logistic regression, and (2) the tree-based extreme gradient boosting. The regularized logistic regression is the industry standard model and serves as our benchmark. It can handle high-dimensional data (Hastie et al., 2009) and is easier to interpret compared to other ML models due to its linear form. The regularized logistic model requires manually specifying the interaction terms in the model as well as other transformations to deal with non-linearities. Tree-based gradient boosting models overcome these limitations and offer a strong benchmark. Moreover, previous research for related classification tasks in the financial industry suggests that tree-based gradient boosting models are not outperformed by neural networks (Gunnarsson et al., 2021). In short, the model relies on constructing an ensemble of decision trees sequentially. We refer to Chen and Guestrin (2016) for details on the model estimation.

4 Experimental configuration

4.1 Data set

We obtained a data set through a partnership with a major French provider of financial services for our experiments, containing monthly client records. The provided sample consists of customers for whom the focal company is their primary financial services provider to

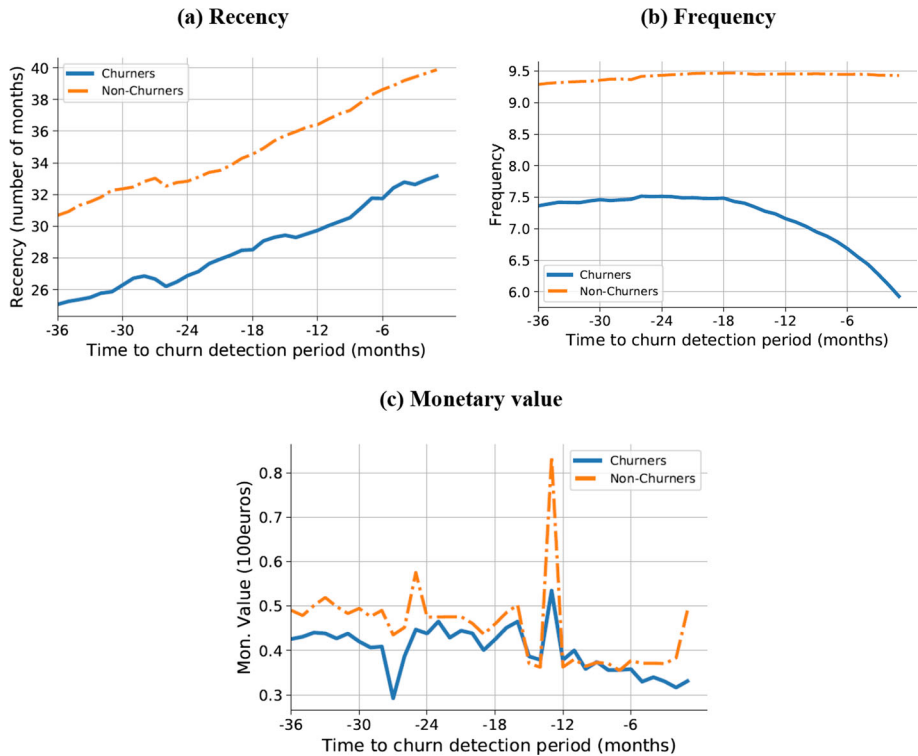


Fig. 3 Evolution over time of mean RFM variables by churn status

ensure the high quality of the behavioral data. The database contains variables that are frequently used to predict churn, like demographic characteristics and behavioral information to compute sequential RFM variables. The time-varying RFM measures are engineered as described in Sect. 3. Figure 3 visualizes how average frequency, recency, and monetary values differ between churners and non-churners over time. In line with the literature, churners are relatively new (lower recency) and have consistently fewer products (lower frequency), and are less valuable (lower monetary value).

Table 2 provides definitions of all the variable categories other than RFM that we use in the experiments. The three categories (i.e. demographic, behavior, and customer-company interaction) group 37 variables, which are frequently used in CCP (De Caigny et al., 2020).

In line with the CCP literature, our main target variable is a dichotomous indicator that equals one if the customer cancels all the contracts with the company within a fixed observation window of 12 months, which in our sample starts on April 1st, 2018. Moreover, the churn definition is aligned with the company's current business processes.

4.2 Data preprocessing and experimental setup

Our data preprocessing and experimental setup are based on conventional practices in recent CCP studies. First, we standardize the features and obtain the parameters for the standardization from the training data. Next, we treat our training samples for the presence of class

Table 2 Overview of static customer variables in our data set

Variable category	Contents	Nbr. of variables
Customer demographics	Standard socio-economic variables: age, gender, income, marital status, etc	14
Customer behavior	Variables that summarize the consumption behavior of customers within the portfolio of the focal company. These variables indicate the possession of specific products, length of the relationship, loan, and mortgage-related information, etc	18
Customer–company interactions	Includes information related to online connections to the company’s website or mobile application, e-mail and telephone contacts, and face-to-face contacts with the advisor	5

imbalance because it can negatively impact the predictive performance. To train the model, we apply an under-sampling approach in line with De Caigny et al. (2018). To focus on our core contributions, exploring alternative sampling approaches is out of the scope of the focal study.

We base the results on holdout test sets on fivefold cross-validation, which is common practice in CPP (Van Nguyen et al., 2020). To tune the hyper-parameters of the models, we rely on a grid search of hyper-parameters and a nested cross-validation procedure. The grid search helps to reduce the variability that would arise from a random search of hyper-parameters. The nested cross-validation provides a clearer picture of the models’ performance and stability compared to a single split. Table 3 provides an overview of the considered parameter settings.

Table 3 Overview of the hyper-parameter settings

Classifier	Hyper-parameter	Candidate settings
DNN only—RNN	Sequential variable encoder	[LSTM, GRU]
	Number of hidden states	[8,16,32,64,128,256,512]
	Batch size	[8,16,32,64]
	Additional layer after input	[Yes, No]
	Last hidden state from RNN	[Yes, No]
DNN only—transformer	Number of layers	[4,8]
Hybrid—logistic regression	Regularization terms λ	50 consecutive terms
Hybrid—XGBoost	Proportion of variables sampled	[0.5,0.7,0.8,0.9,1]
	Minimum loss reduction tree partition	[1,3,5,8]
	Learning rate	[0.05,0.1,0.2,0.3]
	Number of boosting rounds	[1,6,...,46,51]
	Maximum depth tree	[1,2,4,8,16,32,64]

4.3 Evaluation metrics

To evaluate the performance of the models, we report the area under the receiver operating curve (AUC), the top-decile lift (TDL), and the expected maximum profit criterion (EMPC). The AUC assesses a model's performance independent of the decision threshold to convert estimated churn probabilities into dichotomous class labels of churn and non-churn. Decision-makers can easily interpret this metric because it captures the probability that a randomly selected non-churner has a lower predicted churn probability than a randomly selected churner. Next, TDL allows evaluating the performance of a model within the top ten percent of customers with the highest predicted probability of churn compared to a random selection. Hence, this metric describes how much better the classifier can predict churners compared to randomly selecting churners. The top-decile lift expresses the increase in the number of churners in the top-decile relative to the overall churn rate. The EMPC facilitates assessing a model from a profit perspective and finding the most profitable set of customers to target by making assumptions around the expected future revenues and the distribution of misclassification costs (Verbraken et al., 2013). The assumptions for expected future revenues (CLV) and costs (retention offer and contacting costs) are based on De Caigny et al. (2020), who detail this for the financial services industry. We use the default options of the `empChurn` R package for the distribution of the misclassification costs, as often done in prior churn prediction studies (e.g. Verbraken et al., 2013).

4.4 Statistical testing

Our experimental setup results in five performance estimates per model and for each evaluation metric. To statistically compare model performance measures, we rely on the corrected repeated k-fold cv test suggested by Bouckaert and Frank (2004) appropriate for pairwise comparisons of classifiers with multiple performance measures based on experimental cross-validation of an arbitrary number of replications and folds, and on a single data set. The t-statistic they suggest for this purpose is given by:

$$t = \frac{\frac{1}{kr} \sum_{i=1}^k \sum_{j=1}^r x_{i,j}}{\sqrt{\left(\frac{1}{kr} + \frac{n_2}{n_1}\right) \hat{\sigma}^2}} \quad (5)$$

where n_1 is the number of customers used for training models, n_2 is the number of customers in the test set, r is the number of experimental repetitions, k is the number of folds, $x_{i,j}$ is the measured difference in predictive performance for fold i and replication j , and finally, $\hat{\sigma}^2$ is the variance of the $k \cdot r$ performance differences. In comparisons that involve more than two classifiers, we deploy Li's procedure (Li, 2008) to correct p-values and protect against an elevation of the family-wise error.

5 Results

This section discusses the observed results. Table 4 presents the detailed average performance levels of the various models considered in the study, measured in terms of AUC, TDL, and EMPC. The analyses are structured along the research questions introduced above.

RQ1: what is the most effective DNN architecture for CCP using repeated RFM measures?

Table 4 Results (averages and standard errors) in terms of AUC, top-decile lift (TDL) and maximum profit criterion (EMPC) for the classifiers in this study

Algorithm category	Feature set	Algorithm	Architecture variation	Metric		
				AUC	TDL	EMPC
DNN only	Time-varying RFM	RNN	–	0.7671	(0.0211)	4.4892
		Transformer	–	0.7130	(0.0154)	3.2447
	Time-varying RFM and static variables	RNN	Merged	0.8013	(0.2320)	4.6095
			Concatenation	0.8146	(0.0214)	5.1366
			Attention	0.8084	(0.0239)	4.9825
		Transformer	Merged	0.7567	(0.0384)	3.8095
Hybrid	Time-varying RFM and static variables		Concatenation	0.7761	(0.0292)	3.8778
			Attention	0.7839	(0.0292)	3.9115
			Multihead attention	0.7854	(0.0223)	4.0138
		Regularized logistic regression	RNN concatenation	0.8178	(0.0225)	5.1702
		Extreme gradient boosting	RNN concatenation	0.8428	(0.0193)	5.5447

Table 5 Pairwise comparison between the RNN and transformer models estimated on time-varying RFM measures

Algorithm comparison	Metric					
	AUC		TDL		EMPC	
	<i>t</i> -value	<i>p</i> value	<i>t</i> value	<i>p</i> value	<i>t</i> value	<i>p</i> value
RNN versus transformer	3.3592	0.0142**	4.3466	0.0061***	3.5307	0.0121**

Significant differences at the 90%, 95%, and 99% levels are indicated by *, **, and ***, respectively

The first research question is dedicated to a comparison of alternative DNN architectures for accommodating time-varying RFM measures. Table 5 presents the results of the statistical tests comparing RNN and Transformer models estimated on time-varying RFM measures, which clearly indicate that RNNs dominate a transformer model for handling time-varying RFM measures. This holds for all three performance metrics under consideration.

RQ2: what is the most effective DNN architecture for CCP using mixed data (repeated RFM measures and static customer variables)?

Next, we extend the feature set by including static customer variables alongside time-varying RFM measures. Section 3.2.1 presented alternative network architectures to accommodate these additional features in RNN and transformer models. Table 6 presents the results of pairwise model comparisons.

As can be seen in the first column of Table 6, comparisons are grouped into three parts. The first set involves a comparison of the three RNN architecture variants: merged, concatenation, and attention. The second set compares network architectures in the Transformer category: merged, concatenation, attention, and finally, multi-head attention. A final comparison involves a test that compares the best RNN variant versus the best Transformer variant. From the results presented in Table 6, the following conclusions emerge. First, among RNN architectures, concatenation exhibits the best performance in absolute terms. Statistical tests demonstrate its superiority over the merged variant, but not over the architecture based on attention. Second, among transformer variants, none of the architectures is dominant, at least not in statistical terms. The highest performance can be observed for the architectures with multi-head attention. Finally, a comparison of the best variant of both architecture families reveals that RNN with concatenation significantly outperforms the best transformer variant, i.e. a transformer network with multi-head attention. In summary, the results identify recurrent neural networks with concatenation as the most promising DNN architecture for accommodating a mixed set of variables (Table 6).

Table 7 presents the results of a comparison between RNN and transformer networks based solely on time-varying RFM features, and their best-in-class counterparts based on mixed data. From these tests, unsurprisingly, it is clear that the performance for all metrics significantly improves when models are trained on a mix of time-varying RFM features and static customer variables. This shows that the added complexity of these network variants is justified by increased predictive performance. The observed result also supports the argument expressed in Sect. 2 that predicting customer churn using time-varying features alone does not offer a fully-comprehensive picture of the merits of DNNs.

Table 6 Pairwise comparison of DNN architectures estimated on mixed data: time-varying RFM measures and static customer variables

Comparison grouping	Algorithm comparison	Metric					
		AUC		TDL		EMPC	
		<i>t</i> value	Adj. <i>p</i>	<i>t</i> value	Adj. <i>p</i>	<i>t</i> value	Adj. <i>p</i>
(i) RNN architectures	Concatenation versus Merged	3.7697	0.01251**	2.3403	0.0541*	5.5868	0.0029***
	Concatenation) versus Attention	0.8329	0.2259	0.5491	0.3061	1.2422	0.1410
	Merged versus Attention	− 1.6897	0.0970*	− 1.3049	0.1588	− 1.7984	0.0786*
	Multihed-attention versus Concatenation	− 0.9848	0.2513	− 0.3340	0.4051	− 0.7162	0.3278
(ii) Transformer architectures	Multihed-attention versus Attention	− 0.1790	0.4333	− 0.2409	0.4256	− 0.0708	0.4734
	Multihed-attention) versus Merged	− 1.1210	0.2229	− 0.3066	0.4112	− 0.6802	0.3340
	Attention versus Concatenation	− 1.6660	0.1311	− 0.3257	0.4070	− 1.2914	0.2014
	Attention versus Merged	1.3648	0.1772	0.1832	0.4378	0.9005	0.2845
(iii) Best RNN versus Best Transformer architecture	Concatenation versus Merged	1.0288	0.2419	0.1458	0.4455	0.5703	0.3626
	RNN (Concatenation) versus Transformer (Multihed-attention)	5.4758	0.0027***	3.3400	0.0144**	3.3646	0.0110**

The comparison comprises three groups: (i) RNN architectures, (ii) Transformer architectures, and (iii) between the strongest performer of each group. Significant differences at the 90%, 95%, and 99% levels are indicated by *, **, and ***, respectively. The strongest classifier in every comparison is shown in boldface

Table 7 Pairwise comparison between RNN and Transformer models estimated on time-varying RFM measures vs. best-in-class RNN and Transformer architectures estimated on mixed data (time-varying RFM measures with static customer variables)

Algorithm comparison	Metric					
	AUC		TDL		EMPC	
	<i>t</i> value	<i>p</i> value	<i>t</i> value	<i>p</i> value	<i>t</i> value	<i>p</i> value
RNN versus RNN (Concatenation)	3.3592	0.0142**	4.3466	0.0061***	4.6503	0.0048***
Transformer versus Transformer (Multihead attention)	4.0873	0.0075***	2.8628	0.0229**	4.6503	0.0048***

Significant differences at the 90%, 95%, and 99% levels are indicated by *, **, and ***, respectively. The strongest performing classifier in every comparison is shown in boldface

RQ3: can the performance of a mixed-feature DNN model be improved further by hybridizing it with a conventional CCP classifier?

Finally, the potential of hybridized approaches is tested. Specifically, we examine whether incorporating DNN-model predictions in conventional ML classifiers raises predictive performance. Table 8 presents the results of three comparisons. The best overall DNN architecture, i.e. RNN with concatenation, is compared to (1) a regularized logistic regression model, and (2) extreme gradient boosting, both incorporating the predictions of RNN with Concatenation as an additional feature. In addition, both hybrid variants are compared.

The statistical testing results in Table 8 indicate that the interest in hybridizing models depends on the ML learner that serves as a host. In the regularized logistic regression with RNN predictions, the performance is not significantly improved over a standalone RNN model. However, when RNN predictions are embedded in an extreme gradient boosting model, the performance is found to significantly improve over a standalone RNN model. This holds for all three performance measures. Finally, we observe the overall best performance for the hybrid extreme gradient boosting model that incorporates RNN predictions since it outperforms the hybrid regularized logistic regression. Differences are statistically significant for all three performance criteria.

To conclude, Table 9 reports the set of selected hyper-parameters by outer fold for the models with the best performance based on AUC_{outer} . For the regularized logistic model there is a broad range of selected regularization terms λ and there is no clear tendency for the model to perform worse in the outer test folds. Results for the gradient boosting show no variability regarding the learning rate. They also suggest using a larger number of boosting rounds and sample randomly half of all the available variables. The RNN concatenation model performed better on average when using a GRU encoder of time-varying variables, using a unidirectional version, and only taking into account the last hidden state of the sequence. Regarding the transformer model, there is no clear stable pattern of the hyper-parameters except for the number of layers (8).

Table 8 Pairwise comparisons of the RNN (Concatenation) and hybrid models integrating predictions produced by RNN (Concatenation) in (1) regularized logistic regression and (2) extreme gradient boosting

Algorithm comparison	Metric		TDL		EMPC	
	AUC					
	<i>t</i> value	<i>p</i> value	<i>t</i> value	<i>p</i> value	<i>t</i> value	<i>p</i> value
RNN (Concatenation) versus hybrid regularized logistic regression-RNN (Concatenation) model	− 0.50730	0.3193	− 0.2844	0.3951	0.2815	0.3962
RNN (Concatenation) versus hybrid extreme gradient boosting-RNN (Concatenation) model	− 5.8111	0.0022***	− 2.5255	0.0305**	− 4.4457	0.0056***
hybrid regularized logistic regression-RNN (Concatenation) model versus hybrid extreme gradient boosting-RNN (concatenation model)	− 2.8244	0.0238**	− 2.6026	0.02994**	− 3.4328	0.0132**

Significant differences at the 90%, 95%, and 99% levels are indicated by *, **, and ***, respectively. The strongest performing classifier in every comparison is shown in boldface

Table 9 Optimal hyperparameter values per cross-validation fold

Classifier	Parameter	Outer cross-validation fold index				
		1	2	3	4	5
Regularized logistic regression	Regularization parameter λ	0.41	0.015	0.1	0.126	0.012
Extreme gradient boosting	Prop. Sampled variables	0.5	0.5	0.5	0.5	0.7
	Learning rate	0.1	0.1	0.1	0.1	0.1
	Regularization parameter γ	3	5	8	1	5
	Max tree depth	32	32	32	8	8
	Number of rounds	46	41	41	46	46
Recurrent neural networks	RNN type	LSTM	GRU	GRU	GRU	GRU
(concatenation)	Bidirectional	True	False	False	False	False
	Extended	True	False	True	False	False
	Last hidden only	True	True	True	True	True
	Number of hidden units	256	64	256	32	32
	Batch size	16	16	32	8	8
Transformer	Number of layers	4	8	8	8	8
(Multihead attention)	Extended	False	False	False	False	False
	Number of hidden units	512	8	16	16	32
	Batch size	16	8	16	16	32

6 Discussion

Having provided empirical answers to our core research questions, we next discuss the implications of the observed results and reconcile our findings.

First, we find RNN-type networks to provide a more suitable framework for extracting information from time-varying RFM variables than transformer networks. Much literature advocates the advantages of transformers. They have virtually replaced RNNs in language-related problems and become increasingly popular for computer vision tasks. Our findings oppose this trend of the transformer becoming the *lingua franca* of deep learning. Scalability resulting from the ease of parallelizing computations is a major advantage of the transformer. This advantage facilitates pretraining transformer networks on very large data sets, which would be impossible with RNNs. Large pretrained networks lead to the superiority of the transformer in tasks like language or image processing where enormous amounts of data for pretraining are easily available. Marketing applications like CCP do not involve the same masses of data. Moreover, it is not at all clear whether concepts like pretraining and transfer learning are applicable in marketing and/or whether this would be successful. Therefore, scalability advantages, which enable transformers to benefit from richer pretraining in other domains, do not translate into higher performance in the CCP context studied here. Given that we do not consider transfer learning but train our networks from scratch, which is probably the standard setting in many marketing applications, the strict sequentiality of hidden state

updates, the computational bottleneck within the RNN framework, might be the reason for the RNN to extract more useful information from the time-varying RFM variables and, therefore, predicting customer churn more accurately than transformers (as confirmed by Table 4). In the same vein, the observed results shed some light on the sequential dependency structure in RFM variables. Transformers are much credited for their ability to capture long-term dependencies in sequential data. The superiority of RNNs as observed in Table 4 indicates that churn patterns in the focal data set are mainly driven by near- or short-term effects. This is notable because switching costs in the financial services industry are higher compared to many other service businesses. Relatively higher switching costs suggest the willingness to churn to build up over time and some lead time before finally canceling a service. In this regard, finding the RNN to excel in an industry with high switching costs, where one would expect long-term dependencies between RFM data and the churn event, could indicate that RNNs will also outperform transformer networks in other domains where changing service provides is easier than in the financial services.

Second, many customer behavior forecasting settings including CCP require processing static and time-varying covariates. Neural networks facilitate the processing of alternative types of data in a single network architecture. Results associated with RQ2 clarify alternative options for integrating demographic (static) and RFM (time-varying) customer features in a CCP model. Consistent with expectations, we find a simple pooling of static and time-varying features in the input vector to perform less well than multi-modal architectures in which task-specific subnetworks take care of processing static and dynamic inputs, respectively, and subsequent layers concatenate the derived latent representations of each type before producing predictions. Attention mechanisms offer yet another approach to integrating static and time-varying features. Multi-head attention is the key component of transformer networks. Our results echo this characteristic in that transformer networks perform best with multi-head attention. For RNNs, our results warrant recommending marketing analysts to use the concatenation approach when devising a CCP model.

Third, we find further evidence for the power of the gradient boosting framework to aid customer-centric decision tasks. When comparing CCP models derived from a single DNN to hybrid classifiers, in which DNN outputs are transferred to a second-stage classifier, the combination of RNN and gradient boosting facilitates developing the most accurate churn model overall. Much empirical work highlights the outstanding performance of gradient boosting. A recent study in the related field of credit scoring (Gunnarsson et al., 2021), for example, suggests that gradient boosting outperforms DNN in credit scoring. Our results mirror the unmatched performance of gradient boosting and suggest that the combination of RNN for extracting information from time-varying features and a gradient boosting-based post-processor for integrating observed and latent customer characteristics gives a highly powerful CCP model. In appraising this recommendation, it is only fair to mention several previous studies that have applied ML algorithms including gradient boosting to heterogeneous data sets including both, static and time-varying features. Corresponding work uses a range of feature transformations to aggregate the latter before applying an ML-based prediction model (e.g., Koehn et al., 2020). To maintain a clear focus on the research questions that inspired this study, we did not report empirical results from comparing DNN-based models to purely static CCP models with feature aggregations. However, these results are available for the interested reader as supplementary material⁵ and confirm the superiority of sequence learning over feature aggregation when using RNNs, whereby the aggregated time-varying

⁵ The URL to the supplementary material will appear here should the paper be accepted for publication.

RFM features (across different transformations) have been processed by a regularized logistic regression and an extreme gradient boosting classifier.

7 Conclusion

ML has become an increasingly important tool to support decision-making in marketing. Corresponding decision models forecast customer behavior using transactional data related to the recency, frequency, and monetary value of client transactions. Such RFM variables are naturally dynamic and time-evolving. Their incorporation into predictive ML models is nontrivial and offers considerable degrees of freedom. Focusing on the prominent use case of customer churn prediction (CCP), the paper has studied the potential of recently introduced deep learning approaches for sequential data modeling in a CCP context. We have examined the space of modeling options comprising different network architectures for sequence prediction, pooling strategies for blending dynamic and static customer features, and approaches to obtain an overall churn prediction in a single stage CCP model versus a hybrid strategy, which incorporates DNN-based churn predictions into a conventional CCP model.

The empirical results observed in a multi-factorial experiment related to customer churn in the financial services industry provide original insights into the interplay of alternative modeling options and suggest marketing analysts a principled routine of how to integrate time-varying RFM variables into CCP and, more generally, customer behavior forecasting models. In practice, the developed tool could inform business decision-makers about customers' risk of churning. Our experiments present several modeling options that allow financial services providers to create better churn prediction models using sequential data. First, the hybrid approaches serve as a fairly easy way to extend existing customer churn prediction models with sequential data. As our results demonstrate, such an approach could already significantly improve churn prediction models that do not integrate time-varying data. Next, we present a pure DNN-based approach that integrates various types of data. Such an approach could be extended with other data sources that were not included in this experiment.

Our study exhibits limitations, which pave the way for future research. Most importantly, we employ a data set from the financial service industry. Feature-rich, real-world data comprising time-varying and static customer characteristics for customer behavior and specifically churn prediction is not easily available in the public domain. Access to the proprietary data set has been, therefore, a key asset facilitating this research. However, we cannot claim external validity of the observed results beyond the employed data and welcome future work to revisit the observed results, DNN architectures, and CCP models using different data sets. Another limitation concerns the focus on correlational models to predict churn as opposed to causal models predicting retention uplift. When the primary goal of CCP is to target retention campaigns, decision-makers should use uplift models. However, in appraising the limitation of not using uplift modeling, it is important to note that campaign targeting is not the only objective of CCP and that many uplift modeling strategies such as the S-, T-, or X-learner employ conventional (i.e., correlational) ML models as a building block. In this regard, our results are immediately relevant to uplift modeling and inform campaign targeting decisions whenever an uplift model processes time-varying features. Next, we acknowledge that another important goal of CCP beyond campaign targeting is to understand the drivers of customer churn. DNN models are black-boxes and require additional (xAI) tools to reveal the behavioral patterns derived from transactional customer data. While model-agnostic and

DNN-specific xAI tools exist, we see the largest need for future work in this direction, namely building and testing xAI tools that support both, time-varying and time-invariant features. Mixed data is common in CCP and many other applications in marketing and beyond. Having clarified how to leverage corresponding data for CCP in this study, the next necessary step is to develop tools for interpreting the DNN-based prediction approaches we find to excel. Finally, our study provides insights into the impact of time-varying features on churn, but there could also be external influences on the features. An important example is covariate shift, for which more research is needed in the customer churn prediction domain.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s10479-023-05259-9>.

Acknowledgements We wish to extend our gratitude to the anonymous financial services company that kindly provided us with the data analyzed in this study, and the editor and anonymous reviewers for their constructive comments that allowed us to substantially improve the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. No funding was received.

Availability of data and material The data research was obtained under an NDA and cannot be shared publicly. The authors can make available pseudonymized data subsets upon request.

Declarations

Conflict of interest No conflict of interest.

Code availability Not applicable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Bouckaert, R. R., & Frank, E. (2004). Evaluating the replicability of significance tests for comparing learning algorithms—advances in knowledge discovery and data mining. In H. Dai, R. Srikant, & C. Zhang (Eds.), *Proceedings of the Pacific-Asia conference on knowledge discovery and data mining (PAKDD) 2004* (pp. 3–12). Springer.
- Chaudhari, S., Mithal, V., Polatkan, G., & Ramanath, R. (2021). An attentive survey of attention models. *ACM Transactions on Intelligent Systems and Technology*, 12(5), 1–32.
- Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 785–794). Association for Computing Machinery.
- Chen, Z. Y., Fan, Z. P., & Sun, M. (2012). A hierarchical multiple kernel support vector machine for customer churn prediction using longitudinal behavioral data. *European Journal of Operational Research*, 223(2), 461–472.
- Cho, K., van Merriënboer, B., Gulcehre, C., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP 2014)*.
- Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of the NIPS 2014 workshop on deep learning, December 2014*.

- De Caigny, A., Coussement, K., & De Bock, K. W. (2018). A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. *European Journal of Operational Research*, 269(2), 760–772.
- De Caigny, A., Coussement, K., De Bock, K. W., & Lessmann, S. (2020). Incorporating textual information in customer churn prediction models based on a convolutional neural network. *International Journal of Forecasting*, 36(4), 1563–1578.
- Galassi, A., Lippi, M., & Torrioni, P. (2021). Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10), 4291–4308.
- Gattermann-Itschert, T., & Thonemann, U. W. (2021). How training on multiple time slices improves performance in churn prediction. *European Journal of Operational Research*, 295, 664–674.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
- Gunnarsson, B. R., Vanden Broucke, S., Baesens, B., Óskarsdóttir, M., & Lemahieu, W. (2021). Deep learning for credit scoring: Do or don't? *European Journal of Operational Research*, 295(1), 292–305.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
- Janssens, B., Bogaert, M., Bagué, A., & Van den Poel, D. (2022). B2Boost: Instance-dependent profit-driven modelling of B2B churn. *Annals of Operations Research*, 1, 1–27.
- Koehn, D., Lessmann, S., & Schaal, M. (2020). Predicting online shopping behaviour from clickstream data using deep learning. *Expert Systems with Applications*, 150, 113342.
- Li, J. (2008). A two-step rejection procedure for testing multiple hypotheses. *Journal of Statistical Planning and Inference*, 138(6), 1521–1527.
- Liu, X., Xie, M., Wen, X., Chen, R., Ge, Y., Duffield, N., & Wang, N. (2018). A semi-supervised and inductive embedding model for churn prediction of large-scale mobile games. In *Proceedings of the 2018 IEEE international conference on data mining (ICDM)* (pp. 277–286).
- Luong, T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 1412–1421). Association for Computational Linguistics.
- McCarthy, D. M., Fader, P. S., & Hardie, B. G. S. (2017). Valuing subscription-based businesses using publicly disclosed customer data. *Journal of Marketing*, 81(1), 17–35.
- Óskarsdóttir, M., Bravo, C., Verbeke, W., Sarraute, C., Baesens, B., & Vanthienen, J. (2017). Social network analytics for churn prediction in telco: Model building, evaluation and network architecture. *Expert Systems with Applications*, 85, 204–220.
- Qi, J., Zhang, L., Liu, Y., Li, L., Zhou, Y., Shen, Y., et al. (2009). ADTreesLogit model for customer churn prediction. *Annals of Operations Research*, 168, 247–265.
- Risselada, H., Verhoef, P. C., & Bijmolt, T. H. A. (2010). Staying power of churn prediction models. *Journal of Interactive Marketing*, 24, 198–208.
- Rush, A. (2018). The annotated transformer. In *Proceedings of the workshop for NLP open source software (NLP-OSS)* (pp. 52–60). Association for Computational Linguistics.
- Rust, R. T., Lemon, K. N., & Zeithaml, V. A. (2004). Return on marketing: using customer equity to focus marketing strategy. *Journal of Marketing*, 68(1), 109–127.
- Schweidel, D. A., Park, Y. H., & Jamal, Z. (2014). A multiactivity latent attrition model for customer base analysis. *Marketing Science*, 33(2), 273–286.
- Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Proceedings of the 27th international conference on neural information processing systems—volume 2* (pp. 3104–3112). MIT Press.
- Tan, F., Wei, Z., He, J., Wu, X., Peng, B., Liu, H., & Yan, Z. (2018). A blended deep learning approach for predicting user intended actions. In *Proceedings of the 2018 IEEE international conference on data mining (ICDM)* (pp. 487–496).
- Van Nguyen, T., Zhou, L., Chong, A. Y. L., Li, B., & Pu, X. (2020). Predicting customer demand for remanufactured products: A data-mining approach. *European Journal of Operational Research*, 281(3), 543–558.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Proceedings of the Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc.
- Verbeke, W., Dejaeger, K., Martens, D., Hur, J., & Baesens, B. (2012). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218(1), 211–229.
- Verbraken, T., Verbeke, W., & Baesens, B. (2013). A novel profit maximizing metric for measuring classification performance of customer churn prediction models. *IEEE Transactions on Knowledge and Data Engineering*, 25(5), 961–973.

- Wangperawong, A., Brun, C., Laudy, O., & Pavasuthipaisit, R. (2016). Churn analysis using deep convolutional neural networks and autoencoders. *arXiv.org, stat.ML*.
- Wei, C. P., & Chiu, I. T. (2002). Turning telecommunications call details to churn prediction: A data mining approach. *Expert Systems with Applications*, 23(2), 103–112.
- Wu, Z., Jing, L., Wu, B., & Jin, L. (2022). A PCA-AdaBoost model for E-commerce customer churn prediction. *Annals of Operations Research*, 1, 1–18.
- Yang, C., Shi, X., Jie, L., & Han, J. (2018). I know you'll be back: Interpretable new user clustering and churn prediction on a mobile social application. In *Proceedings of the proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 914–922). Association for Computing Machinery.
- Zaratiegui, J., Montoro, A., & Castanedo, F. (2015). Performing highly accurate predictions through convolutional networks for actual telecommunication challenges. In *Proceedings of the international conference on computer vision and pattern recognition* (Vol. abs/1511.0, pp. 1–8).
- Zhang, Y., Bradlow, E. T., & Small, D. S. (2015). Predicting customer value using clumpiness: From RFM to RFMC. *Marketing Science*, 34(2), 195–208.
- Zhou, J., Yan, J., Yang, L., Wang, M., & Xia, P. (2019). Customer churn prediction model based on LSTM and CNN in music streaming. In *Proceedings of the 2019 international conference on advanced electrical, mechatronics and computer engineering (AEMCE 2019)* (pp. 254–261).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.