Transfer Learning in Credit Risk[s]

Hendra Suryanto[1[0000−0001−9318−8572]], Charles Guan[1[0000−0002−6406−2599]], and Andrew Voumard[1[0000−1111−2222−−3333]]

Rich Data Corporation, Singapore info@richdataco.com
http://www.richdataco.com
{hendra.suryanto,charles.guan,andrew.voumard}@richdataco.com

Abstract. In the credit risk domain, lenders frequently face situations where there is no, or limited historical lending outcome data. It generally results in limited or unaffordable credit for some individuals and small businesses. Transfer learning can potentially reduce this limitation, by leveraging knowledge from related domains, with sufficient outcome data. We investigated the potential for applying transfer learning across various credit domains, for example, from the credit card lending and debt consolidation domain into the small business lending domain.

Keywords: First keyword · Second keyword · Another keyword.

1 Introduction

We studied a new domain where no or limited historical lending outcomes are available, for example: offering credit to un-banked or under-banked populations or micro to small businesses, where limited historical data is available. Currently, lenders rely mainly on expert rules for credit scoring. Due to high uncertainty in the performance of such scoring models, lenders charge a high fee or simply don't offer credit. Transfer learning from related domains is a potential solution to augment this lack of information and improve financial inclusion. For instance, transferring knowledge from credit card/debt consolidation loans to more risky small business loans or from utility bill payments to loan repayments could potentially deliver a more accurate scoring model.

We investigated the application of transfer learning during the initial stage of a credit risk model implementation, where there was limited historical labelled data available. In the credit risk domain, business priorities are stability and accuracy of model performance, in order to predict the probability of default. We present our approach, that enabled us to combine the outcome of the transferred model from related credit risk domains, with new models based on newly acquired labelled data from new domains. Using this approach, we were able to achieve a higher accuracy and maintained stability of the overall model. Experiments on real-world commercial data showed that combining the transferred models and the new models can achieve these goals by using an incrementally transitioned

[s] Supported by Rich Data Corporation.

approach. To allow us to publish the results and comply with the privacy require- ments of our client's data, we reproduced our experiment using lendingclub.com data, https://www.lendingclub.com/info/download-data.action, which is publicly available.

When a lender expands into new market segments, a new credit risk model is required to assess the credit risk of loan applications. The current approach is based on expert rules, where the credit risk expert builds business rules based on data and available derived data, combined with the expert's experience and knowledge. Lenders initially used an expert model to gather sufficient labelled data, to build a supervised learning model. The expert model is compared against the supervised learning model. If one model performs substantially better than the other, the better model is used. Alternatively, if both models complement each other, they can be combined into an ensemble model. In real-world commercial lending systems, lenders normally charge a higher price or limit credit offerings as there is no (or limited) labelled data to validate the expert models. The result is many individuals and businesses are excluded from these formal lending systems. Organizing data access for a suitable expert to perform analysis may be difficult. For example, when data can only be accessed on site by authorized persons, it may be hard to organize access.

Credit card and debt consolidation loans are typical unsecured consumer loans. Their scoring model depends mainly on an individual's credit rating, income, expenses and other attributes like stability of their employment and residence. This type of lending was among the first areas penetrated by fintechs, and became a crowded and competitive market in countries like the UK, US and China. Lenders have accumulated sufficient historical lending outcomes in this domain and have developed many scoring models that are stable and accurate. Small business lending is a relatively new market for fintechs, since it is more risky and diverse, and more challenging to predict the outcomes. As we can see in the lendingclub.com data, the quantity of historical lending outcome for small business loan is far less and insufficient to develop a stable and accurate model using traditional supervised learning. With less competition and higher margin for small business lending (compared to consumer lending) it is more valuable for lenders to find ways to predict loan outcomes and serve this market. Furthermore, Micro, Small and Medium Enterprises (MSMEs) are one of the strongest drivers of economic development, innovation and employment. Access to finance is frequently identified as a critical barrier to growth for MSMEs. Creating opportunities for MSMEs in emerging markets is a key way to advance economic development and reduce poverty. 65 million (or 40% of formal MSMEs) in developing countries have unmet financing needs. The MSME finance gap in developing countries is estimated at $5.2 trillion - 1.4 times the current level of MSME lending [4].

2 Related Work

We have seen increasing interest in transferred supervised models - from one domain to another. Most published works in this area cover image processing, for example: Yang proposed transferring parameters on SVM [12], Pan proposed domain adaptation using transfer component analysis [6]. Pan and Yang grouped transfer learning into four approaches: instance-transfer, feature-representation- transfer, parameter-transfer, and relational-knowledge-transfer [7].

Our experimentation combines the reuse of features and derivation of new features from the source (existing) domain. Source domain labels are available; limited target (new) domain labels are available. We also focus on classification. Our experimentation is similar in those ways to Transductive Transfer Learning [6]
- one key addition, is to the target classification task optimization. In Transductive Transfer Learning, the source and target tasks must be the same (classification in this case). In our experimentation though, we took a new step in optimizing the target model accuracy, by introducing and experimenting with an extra optimization variable: the level of relative source/target feature data contribution proportions into the target model.

Many papers focus largely on making optimal choices of parameters, features, and source(s), to transfer learning to the target model, as summarized in [10] - which examines homogeneous and even heterogeneous data domains, symmet- ric and asymmetric feature transformation, for instance-based, feature-based, parameter-based, and relational-based related transfer learning. [9], [11], [2], [5] make specific efforts to minimize 'negative transfer' (a transfer that has a negative impact on the target model). While these approaches help to improve target model results - and can (in some cases) reduce target model build times, our focus was centered on optimizing the target model configuration / composition and design's use of the transferred features, after they were already chosen to be inputs to the target model.

3 Credit Risk

Lenders seek to optimize the risk return ratio across their lending portfolios. Accurately and consistently measuring credit risk is the foundation of this optimization. Lenders commonly use the concept of Expected Loss (EL) to measure credit risk. In an unsecured lending scenario, EL is mainly determined by the Probability of Default (PD). Credit scoring models are used to calculate PD. Inputs of a credit scoring model are normally attributes of the loan applicant and their application. In this paper we use a few attributes from lendingclub.com data to illustrate our approach. In credit risk, the most common metrics to assess the quality of credit scoring model are Gini, Kolmogorov-Smirnov statistics (KS), Lift, the Mahalanobis distance and information statistics [8]. In this paper we use Gini for this purpose.

The scoring model output is a score from 0 to 1; it is an estimated probability of default. Usually some part of the data is pre-allocated for calibration of the score. Lenders use a set of decision processes and rules to make an optimal decision with the derived PD and loan application data as inputs. A decision process generally starts with an eligibility test. PD is calculated for the eligible applicants, and then used to group applicants as different decision groups. For instance, the interest rate could vary for different decision groups, and the loan amount as a percentage of net income could vary too.

In this investigation, our focus is credit scoring for unsecured lending. We measure the performance of our credit scoring model using Area Under Receiver Operating Curve (AUC) or $ttiniROC$ which is $2AUC$ 1 [3]. This GiniROC shares the same concept as Gini, for splitting criteria in CART [1]. Gini and GiniROC usages are, however, different. The metric GiniROC is used to allow the assessment of model quality, based on PD, without needing to convert PD into binary classifications, since the threshold to do those classifications is defined in the credit decisioning.

3.1 Credit Scoring

Credit scoring produces a PD, which is used to predict binary outcomes, loan- paid or loan-defaulted. In real-world scenarios, there are additional outcomes, such as late payment or partial payment. In credit scoring, we need a metric to assess the quality of the model without defining a threshold to convert the PD into a classification. When we have classifications, we can use a metric such as $Fscore$. In credit risk, this decision is deferred to the credit decisioning step, where expert rules are utilized to decide whether the loan is approved or not.

3.2 Credit Decisioning

Credit decisioning consumes PD and produces a decision to approve or decline a loan application. The conversion from PD to a decision is usually driven by a mapping table to map ranges of PD to decisions. The decision is not only to approve or to decline, it may also update the loan amount, interest and term. This model is usually based on expert rules, since the data is usually too sparse and/or the search space is too large for building supervised learning models

4 Experimentation Approach

The experiments were set up to empirically assess the effectiveness of our transfer learning algorithm. Table 1 shows six example network configurations, that have an increasing Progressive Shift Contribution (PSC) from the source domain to the target domain. Model No 1 is developed by training using source domain data only. The domain contribution in Models No 2, 3, 4 and 5 are then progressively shifted from source to target domain data. The last model, Model No 6 is trained using target domain data only. The algorithm can be generalized to any size of network configuration. Further details on the algorithm will be discussed in Section 6. Source code and data for all experiments is provided.

Table 1. Six Network Configurations with *PSC*

No	Model Name	Source Do- main Con- tribution	Target Do- main Con- tribution	Non Train- able Lay- ers	Trainabl e Layers	Figure
1	$M(v)_e$			4	0	2
2	$M(w)_{transfer}$			3	1	4
3	$M(wx)_{transfer}$			5	2	5
4	$M(wxy)_{transfer}$			6	4	6
5	$M(wxyz)_{transfer}$			6	7	3
6	$M(u)_n$			0	4	1

We discover the optimum network configuration by shifting the PSC from the source to target domain and measure the Gini performance on the target domain test data. The model performance is conceptually influenced by a) the modelling techniques (e.g. deep learning, gradient boosting machine, generalized linear model), hyper parameters[1], b) the signal strength in the data and c) feature engineering; Informally, the relationship between $ttini$ and these factors canbe written as follows:

$$ttini = g(test(M_e, s_e)) \tag{1}$$

where s_e is test data from the source domain, M_e is the model trained using training data from the source domain, $test()$ is an activity to test a model on the test data producing the test results and $g()$ is a function to calculate the Gini of the results. M_e is defined as follows:

$$M_e = train(M_0, P_e, t_e, F_e) \tag{2}$$

where M_0 is a deep neural network configuration with initial random weights,

P_e
is a set of hyper parameters to train M_e, t_e is the training data from the source domain, F_e is a set of features derived from t_e, $train()$ is an activity to train a model based on these four factors. The result of $train()$ is a trained model.

To explain how we perform the PSC, we define a function $split()$ to conceptu- ally split M_e into two segments: $Mfix_e$ and $Mfree_e$. $Mfix_e$ is the segment where the layers were trained using t_e and these layers are not retrainable. $Mfree_e$ is the segment where the layers were also trained using t_e, but these layers will be trainable using the training data from the target domain t_n.

$$(Mfix_e, Mfree_e) = split(M_e) \tag{3}$$

[1] the hyper parameters optimization has been done before this step

The inverse function of $split()$ is $c()$, to combine $Mfix_e$ and $Mfree_e$ back into M_e

$$M_e = c(Mfix_e, Mfree_e) \tag{4}$$

To create a mix model based on both the source and target domain data, we developed a model for the target domain $M_{transfer}$, by transferring the structure and weights of $Mfix_e$ layers and retraining the structure and weights of $Mfree_e$.

$$Mfree_n = train(Mfree_e, P_n, t_n, F_n) \tag{5}$$

Finally, we combined the target model $Mfree_n$ with $Mfix_e$. The result is the transferred model $M_{transfer}$

$$M_{transfer} = c(Mfix_e, Mfree_n) \tag{6}$$

The overall goal is to maximize:

$$ttini_{transfer} = g(test(M_{transfer}, s_n)) \tag{7}$$

by monitoring $ttini_{transfer}$ as we shift the PSC from the source to target domain data. Finally, we discover the maximum $ttini_{transfer}$ by testing the performance of all six network configurations outlined in Table 1.

5 Model Development

We started the model development by creating base models - training them from scratch, without transfer learning. We applied a grid search to discover the set of hyper parameters for the Deep Learning (*DL*) structure. We validated the performance on Credit Card/Debt Consolidation (*CD*)and Small Business (*SB*) data extracts by developing comparison models that used Gradient Boosting Machines (*ttBM*). The comparison of performances is shown in Table 3.

Name	Sampling	Gini *ttBM*	Gini *DL*
CreditCard/DebtConsolidation	random	0.43 ±0.01	0.43 ±0.01
SmallBusiness	random	0.30 ±0.05	0.31 ±0.02

Table 2. Performance of Gradient Boosting Machine (*ttBM*) and Deep Learning (*DL*) on Credit Card and Debt Consolidation (*CD*) and Samll Business Loan (*SB*) datasets, evaluated using 10 fold cross validation

5.1 The base model

The base models were configured based on network structures, beginning with the first, illustrated in Figure 1. It has 16 input nodes on the input layer, 3 hidden layers, each layer has 32 nodes and has 1 output node on the output layer.

One factor that influenced model performance was the strength of signal [2] from the data. For our experiments, we used data based on lendingclub.com data, which is similar to our clients data, with a time range of 2012 to 2018. For the experiments, we extracted four data subsets.

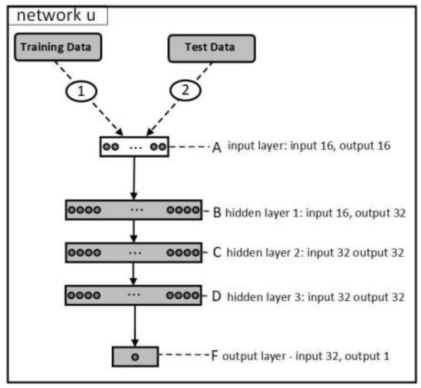

Fig. 1. Network u: the base model

The first extract was dataset $CD4$ with time range of 2012 to 2018. The size was 100,000 records, extracted randomly from 940,948 records where the loan purpose was paying Credit Card and Debt Consolidation. The bad debt rate from this dataset was 21%. The next dataset was dataset $SB4$ extracted from 13,794 records where the loan purpose was for investing in Small Business; this type of loan is riskier; the bad debt rate is 30%. The time range is between 2012 and 2018. No outlier filtering was performed for these two datasets. Datasets $CD1$, $CD2$, $CD3$ are subsets of dataset $CD4$, filtered based on different time ranges. Similarly datasets $SB1$, $SB2$, $SB3$ are subset of dataset $SB4$.

All experiments are based on the data in Table 3. They were performed based on 10 fold cross validation, repeated 5 times. The base model to be transferred was developed using the dataset $CD1$, $CD2$, $CD3$, $CD4$, PDL and network configuration u, as illustrated in Figure 1 and defined in Equation 2.

5.2 Comparison models (Model u)

The comparison models were developed using datasets $SB1$, $SB2$, $SB3$, $SB4$ and ISL. They were also based on network configuration u as shown in Figure 1. Similar to Equation 2, the model built using target domain data can be defined as follows:

$$M(u)_n = train(M(u)_0, P_n, t_n, F_n) \tag{8}$$

[2] associated with the outcome being predicted

ID	Dataset	Time Range	Size	Type	Gini
CD1	*CreditCard/DebtConsol.*	2012-2012	40,000?	Source	0.55 ±0.01
SB1	*SmallBusinessLoan*	2012-2012	1,000?	Target	0.55 ±0.01
CD2	*CreditCard/DebtConsol.*	2012-2014	60,000?	Source	0.55 ±0.01
SB2	*SmallBusinessLoan*	2012-2014	4,000?	Target	0.55 ±0.01
CD3	*CreditCard/DebtConsol.*	2012-2016	80,000?	Source	0.55 ±0.01
SB3	*SmallBusinessLoan*	2012-2016	8,000?	Target	0.55 ±0.01
CD4	*CreditCard/DebtConsol.*	2012-2018	100,000 ?	Source	0.55 ±0.01
SB4	*SmallBusinessLoan*	2012-2018	13,794	Target	0.55 ±0.01
PDL	*PayDayLoan*	2016-2017	140,000 ?	Source	0.55 ±0.01
ISL	*InstalmentLoan*	April 2017	1,023?	Target	0.55 ±0.01

Table 3. List of datasets for transfer learning experiments, the type column indicates whether the dataset is used as the source or the target of the transfer learning process.

where $M(u)_n$ is a model developed using data from the target domain based on network configuration u, $M(u)_0$ is the initial model based on network configuration u with all weights initialized randomly, P_n, t_n, F_n are parameters, training data and features respectively, used to develop the model $M(u)_n$.

$$ttini = g(test(M(u)_n, s_n)) \tag{9}$$

where s_n is test data from the target domain. The Gini for $SB1$, $SB2$, $SB3$, $SB4$ and ISL shown in Table 3 and Table **??** is extracted from the test result of model $M(u)_n$ by applying the function $g()$ on the test results, as defined in Equation 9.

6 *PSC* Models

In Section 4, we discussed six models where the PSC shifts between the target and source domain data. To perform the PSC, we extended the split function defined in Equation 3 with an additional parameter to define the proportion of PSC. The value of this parameter is either v, w, wx, wxy, or $wxyz$. Each value results in a different PSC between the target and source domain. Using these five values, we can develop five PSC models. In addition to these five models, we include the Comparison Model discussed in Subsection 5.2. Thus, we have six PSC models. The following subsections explain the five models in detail.

6.1 Model v

Model v is only created from source domain data (no target domain contribution whatsoever). To create this model, we started by training model $M(v)_e$, based on Equation 10, using configuration shown in Figure 2

$$M(v)_e = train(M(v)_0, P_e, t_e, F_e) \tag{10}$$

Then the model was tested on target domain data, and the Gini value was calculated from the test results.

$$ttini = g(test(M(v)_e, s_n)) \tag{11}$$

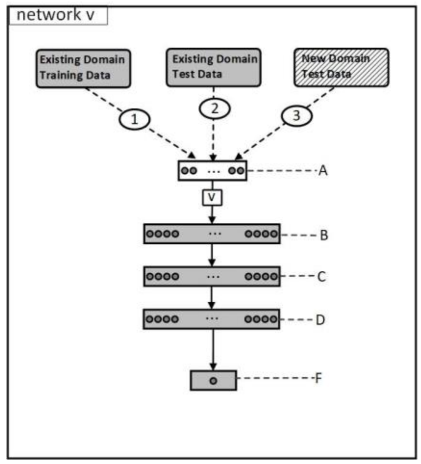

Fig. 2. Network v

6.2 Model wxyz

This model was created based on four parallel networks - each with three hidden layers, connected to the input layer and the output layer. To create this model, we initially copied hidden layers of network v (both the structure and the weights) into networks w, x, y and z. Conceptually, we illustrate the transformation using Equation 12.

$$M(wxyz)_e = transform(M(v)_e) \tag{12}$$

Networks w, x, y and z were setup as illustrated in Figure 3.

After the structure and weights were set (as illustrated in Figure 3), we then set the 3rd hidden layer of Network x as trainable, by the target domain data. Similarly, we set the 2nd and 3rd hidden layers of Network y as trainable by the target domain data. We then set all hidden layers of Network z as trainable by the target domain data. The next three steps are indicated in numbers 1, 2, 3 within ellipses in Figure 3:

1. Weights for networks w, x, y, z were derived from training using t_e. Some layers in networks x, y, z and the output layer are set to trainable by t_n.
2. Train these layers using t_n.

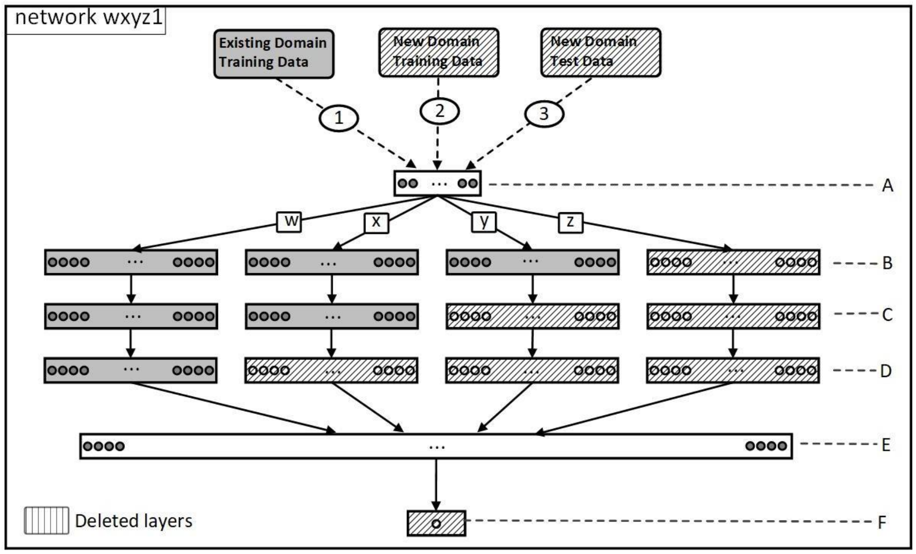

Fig. 3. Network wxyz

3. Test the performance of the whole parallel network (w, x, y, z) on s_n, then calculate the Gini value from the test result.

The development of Model wxyz can be summarized by three equations: Equation 13, Equation 14, Equation 15.

$$(Mfix(wxyz)_e, Mfree(wxyz)_e) = split(M(wxyz)_e) \tag{13}$$

$$Mfree(wxyz)_n = train(Mfree(wxyz)_e, P_n, t_n, F_n) \tag{14}$$

$$M(wxyz)_{transfer} = c(Mfix(wxyz)_e, Mfree(wxyz)_n) \tag{15}$$

In Model wxyz, six hidden layers were trained using the source domain data and seven layers were retrained using the target domain data, i.e. six hidden layers and the output layer were retrained.

6.3 Model w

Model w is developed based on Model wxyz, where Network x, Network y and Network z are deleted. This network configuration is illustrated in Figure 4. In Model w, three hidden layers were trained using the source domain data and only the output layer was retrained using the target domain data.

The development of model w (as illustrated in Figure 4) is shown in Equation 16, Equation 17, Equation 18

$$(Mfix(wxyz)_e, Mfree(wxyz)_e) = split(M(wxyz)_e) \tag{16}$$

$$Mfree(wxyz)_n = train(Mfree(wxyz)_e, P_n, t_n, F_n) \tag{17}$$

$$M(wxyz)_{transfer} = c(Mfix(wxyz)_e, Mfree(wxyz)_n) \tag{18}$$

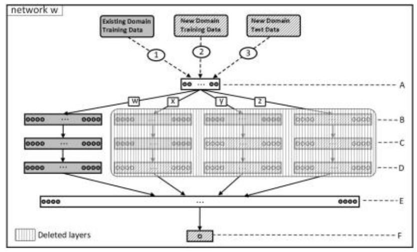

Fig. 4. Network w

6.4 Model wx

Model wx is developed based on Model wxyz, where Networks y and z are deleted. This network configuration is illustrated in Figure 5. In Model wx, five hidden layers were trained using the source domain data. One hidden layer and the output layer were retrained using the target domain data.

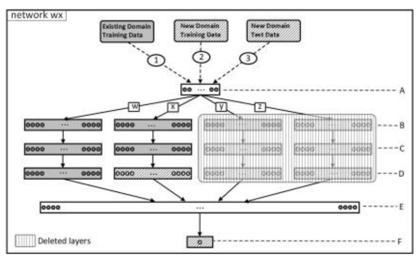

Fig. 5. Network wx

The development of model wx (as illustrated in Figure 5) is shown in Equation 19, Equation 20, Equation 21

$$(Mfix(wx)_e, Mfree(wx)_e) = split(M(wx)_e) \tag{19}$$

$$Mfree(wx)_n = train(Mfree(wx)_e, P_n, t_n, F_n) \tag{20}$$

$$M(wx)_{transfer} = c(Mfix(wx)_e, Mfree(wx)_n) \tag{21}$$

6.5 Model wxy

Model wxy is developed based on Model wxyz, where only Network y is deleted. This network configuration is illustrated in Figure 6. In Model wxy, six hidden layers were trained using source domain data. Three hidden layers and the output layer were retrained using target domain data.

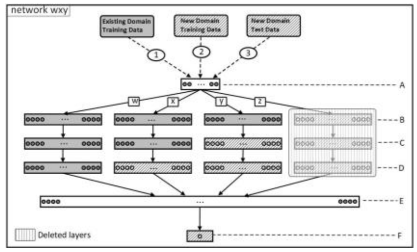

Fig. 6. Network wxy

The development of model wxy (as illustrated in Figure 6) is shown in Equation 22, Equation 23, Equation 24

$$(Mfix(wxy)_e, Mfree(wxy)_e) = split(M(wxy)_e) \tag{22}$$

$$Mfree(wxy)_n = train(Mfree(wxy)_e, P_n, t_n, F_n) \tag{23}$$

$$M(wxy)_{transfer} = c(Mfix(wxy)_e, Mfree(wxy)_n) \tag{24}$$

The experiments have not completed yet, the reference to EDG will be removed. The discussion and conclusion will change based on the experiments. All texts in blue fonts below will change.

6.6 Experimentation results

The experiment results of six models on datasets are as follows:

Table 4. Experimentation Results, the best performing models are marked with *

Source Dataset	$CD1$	$CD2$	$CD3$	$CD4$	PDL
Target Dataset	$SB1$	$SB2$	$SB3$	$SB4$	ISL
$M(v)_e$	0.35 ±0.06	0.21 ±0.04	0.33 ±0.04	0.19 ±0.05	0.31 ±0.01
$M(w)_{transfer}$	0.49 ±0.05	0.24 ±0.04	0.43 ±0.04	0.20 ±0.03	0.39 ±0.04
$M(wx)_{transfer}$	0.55 ±0.09	0.29 ±0.05	0.46 ±0.04	0.25 ±0.05	*0.40 ±0.03
$M(wxy)_{transfer}$	0.58 ±0.05	0.26 ±0.04	0.49 ±0.05	0.24 ±0.03	0.38 ±0.04
$M(wxyz)_{transfer}$	*0.63 ±0.07	0.25 ±0.04	0.49 ±0.03	0.25 ±0.05	0.36 ±0.03
$M(u)_n$	0.58 ±0.06	0.30 ±0.03	*0.49 ±0.02	0.30 ±0.03	0.40 ±0.04

As shown in Table **??**, the best performing model on dataset SB was $M(w)_{transfer}$ with Gini 0.36 and Standard Deviation ±0.03. In this model, only the output layer is retrainable.

For dataset SB_{A1}, $M(wxyz)_{transfer}$ was best, with Gini 0.63 ±0.08. This model was developed by retraining seven layers using target domain data. The remaining six layers were trained using source domain data.

For dataset SB_{B1}, the best was $M\ (u)_n$ with Gini 0.49 ±0.02. This model was developed by training all layers using target domain data. $M\ (wxyz)_{transfer}$ and $M\ (wxy)_{transfer}$ have the same Gini but higher standard deviation.

For dataset SB_{C1} the best was $M\ (wx)_{transfer}$ with Gini 0.49 ±0.03. This model was developed by retraining one layer plus the output layer using target domain data.

SB_{A2}, SB_{B2}, and SB_{C2} have lower maximum average Gini results than their corresponding $EDtt$ datasets. The best performing models are Model $M\ (u)_n$.

6.7 Additional Experiments

We investigated the hypothesis that the Gini performance improvement was due to the complexity of the network structure. We did experiments as described in Equation 25 and Equation 26. The model with network configuration was trained and retrained on source domain data. The performance of this model was 0.39 ±0.01, which is lower than the base model Gini 0.43 ±0.01. It shows that additional complexity of network configuration wxyz does not improve Gini performance.

$$Mfree(wxyz)_e = train(Mfree(wxyz)_e, P_e, t_e, F_e) \tag{25}$$

$$M(wxyz)_{retrain} = c(Mfix(wxyz)_e, Mfree(wxyz)_e) \tag{26}$$

6.8 Discussion

Table **??** summarizes the PSC between the source and target domain data. We found that $M(wxyz)_{transfer}$ had the highest Gini of 0.63 for $EDtt$[5, 0.20]. As $EDttT$ increases, the maximum average Gini decreases, and contribution of the source domain generally increases on the best performing model. The contribution is based on how many layers were trainable using the source and target domain data. In some cases, $M\ (u)_n$ had the best performance.

7 Conclusion

We propose an algorithm to progressively shift the contribution between the source and target domains. The PSC algorithm lets us evaluate incremental complements of target domain data with source domain data. To further validate our PSC algorithm, we also tested on datasets that we created, based on Eucledian distance similarity (discussed earlier). While we undertook some activities manually, the underlying goal has been to devise a framework that can automatically search for the optimum balance between the source and target domain data, resulting in the highest Gini score for that combination. Six PSC models were built, ranging from Model v (using source domain data only) all the way to Model u (using target domain data only).

7.1 Results

After using PSC models on the full dataset, Gini increased from 0.33 to 0.36 for the DL model. Splitting by different $EDttT$ significantly increased the Gini of $EDtt$; $EDtt$[5, 0.20] to 0.63; $EDtt$[5, 0.24] to 0.49, while the respective $IEDtt$ Gini were all at 0.30. Although $EDtt$[5, 0.20] only covers 28% of the full dataset, $EDtt$[5, 0.24] covers 53%. A Similarity Based Bias Splitting pre-filter would be used for commercial applications.

7.2 Future Work

The presented research is part of a larger effort to develop a transfer learning knowledge based system. The presented experiment and results are the first of a series of experiments which will be used to discover and formulate a stream of rules. The rules will be incrementally incorporated in a knowledge based following the Ripple Down Rule framework specifically geared at incremental construction of rule-based systems (Beydoun et al. 2001; Suryanto et al. 2004).

To realise the knowledge based system, an appropriate representation of the transfer context and the transfer recommendations is first needed to enable appropriate encoding of the rules in the system. Towards formulating the representation, we will need identify an adequate set of features of the context transfer. This requires further experiments with additional source data, such as utility payment, taxation, etc. These experiments will also seek ways to accommodate different PSC levels from each data source, and assess target model Gini impact. The representation will also need to account for articulating the recommendations output from the rule-based system. We will also require new features to represent the following:

- Selection of optimization approaches and assessment of target model Gini impact.

- Description of Similarity Based Bias Splitting filter to support the use of $EDtt$ models in commercial setups.

References

1. Breiman, L.: Some properties of splitting criteria. Machine Learning 24(1), 41–47 (1996)
2. Chattopadhyay, R., Ye, J., Panchanathan, S., Fan, W., Davidson, I.: Multi-source domain adaptation and its application to early detection of fatigue. In: KDD (2011)
3. Flach, P.A.: The geometry of roc space: understanding machine learning metrics through roc isometrics. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03). pp. 194–201 (2003)
4. Forum, S.F.: Msme finance gap (Jan 2019), https://www.smefinanceforum.org/ data-sites/msme-finance-gap/
5. Gao, J., Fan, W., Jiang, J., Han, J.: Knowledge transfer via multiple model local structure mapping. In: KDD (2008)
6. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks 22(2), 199–210 (2011)
7. Pan, S.J., Yang, Q., et al.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22(10), 1345–1359 (2010)
8. Řezáč, M., Řezáč, F.: How to measure the quality of credit scoring models. Finance

a úvěr: Czech Journal of Economics and Finance 61(5), 486–507 (2011)
9. Sun, S., Xu, Z., Yang, M.: Transfer learning with part-based ensembles. In: MCS (2013)
10. Weiss, K.R., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. Journal of Big Data 3, 1–40 (2016)
11. Beydoun G and Hoffmann A (2001). "Theoretical Basis for Hierarchical Incremental Acquisition." International Journal of Human Computer Studies 54(3): 407-452.
12. Suryanto, H. and P. Compton (2004). Invented predicates to reduce knowledge Acquisition. 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW04), Whittlebury Hall, Northamptonshire, UK.
13. Beydoun G, Hoffmann A (2000) Monitoring knowledge acquisition instead of evaluating knowledge bases. 12th European conference on knowledge acquisition and knowledge management (EKAW2000). Springer, France

14. Xiao, J., Wang, R., Teng, G.E., Hu, Y.: A transfer learning based classifier ensemble model for customer credit scoring. 2014 Seventh International Joint Conference on Computational Sciences and Optimization pp. 64–68 (2014)
15. Yang, J., Yan, R., Hauptmann, A.G.: Cross-domain video concept detection using adaptive svms. In: Proceedings of the 15th ACM international conference on Multimedia. pp. 188–197. ACM (2007)