Using Deep Learning to maximize *cash returns* for credit card customers.

1. Introduction:

Credit cards are extensively used in the US to make day-to-day purchase, not just because it helps build credit history, but also because it gives customers the flexibility to make a purchase without worrying about the bank balance. [1] shows the statistic of credit card usage based on store types as of July, 2017. Most credit card companies try to lure customers by providing cash return benefits based on their credit card usage. A popular feature offered by major credit cards is to provide cashbacks on future purchases (most commonly for the next quarter). As a customer, one would want to choose categories that would maximize the cash returns for the upcoming quarter, however customers make this choice mostly out of intuition.

Proposed in this report is a predictive model based on *Deep Learning* that would make category recommendations to customers, based on their past credit card transaction history. The approach can be further extended to also predict the approximate amount likely to be spent for those categories. This recommender model would assist the customers to choose categories to maximize their cash returns, thus improving their card usage experience.

2. Customer Recommendations using AI:

Artificial Intelligence (AI) has impacted many industries, with banking being one among them [2]. Market opportunities are constantly changing, and there is fierce competition among banking service providers. With customer empowerment the need of the hour, banks are under constant pressure to renovate the customer experience they provide by adapting to recent technologies. Banks are constantly working to enhance user experience, not just to attract new customers but also to retain their customer base. For this it is important to give a customer recommendations and insights about spendings, in order for them to build confidence in their ability to maximize savings. This has a direct application in credit card service.

Let us take the example of the *cash+* credit card offered by the US Bank. Cardmembers get to pick two categories in which they can earn 5% cash back on the first $2,000 each quarter, and one category where they earn an unlimited 2% [3]. As a user of this feature, I am always in a dilemma to choose among a vast range of categories. However, similar customers (in terms of age, demographic, income etc) generally have a similar pattern of usage every year. For example, working people tend to spend on travelling during December (due to the holiday season) and on clothing/recreation during the thanksgiving month while young college students tend to spend on books and stationary during september or january (when the academic session begins). While these are very evident spending

patterns, there might be other very subtle patterns that can only be identified through advanced data analytics methods.

Banks handle huge volume of unstructured credit card usage data on a daily basis. While the data is beyond the scope of human analysis, machine learning models help analyze them with ease. Deep learning is one such advanced machine learning concept that excels at identifying patterns in unstructured data. It is currently being heavily used internally for fraud detection, performance evaluation, risk management etc but has not been used to improve the customer's experience. I propose a model where Recurrent Neural Network is used to predict credit card usage based on the customer's past usage history. The idea has been visually represented in Fig.1.

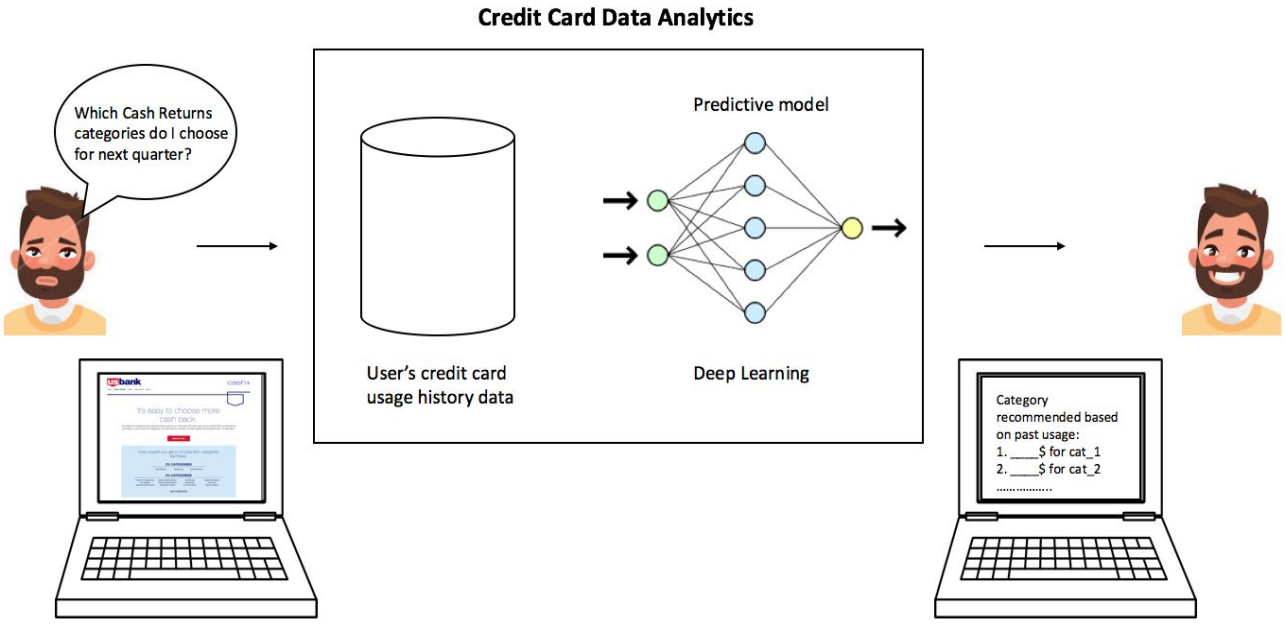

Fig.1 Visual Representation of the analytics process flow.

Written Analytics Plan

3. Recurrent Neural Network (RNN):

A neural network takes a numerical valued feature vector as input, processes the data through multiple hidden units and predicts an output. For details about the Neural Network algorithm, please refer to [4]. Neural networks are powerful machine learning models that achieve state-of-the-art results in a wide range of tasks. Recurrent Neural Network (RNN) is a class of neural network where connections between units form a directed cycle (*folded* over time), that can be used when we are dealing with

sequential data which might be or arbitrary length and have time dependencies. It has been extensively used in the domains of image processing, speech recognition and music generation. For details about RNNs, please refer to [5]. An architectural overview is shown in Fig.2.

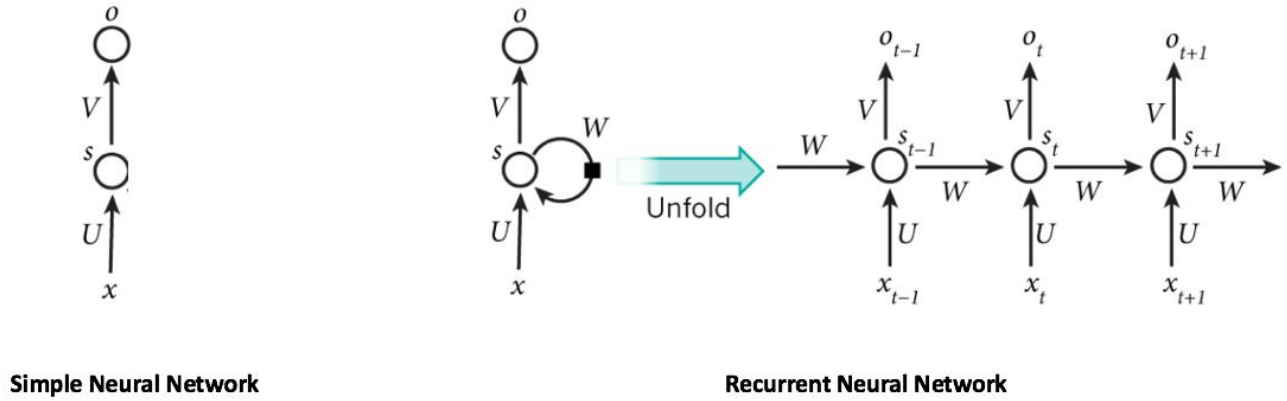

Fig.2 Simple Neural Network vs Recurrent Neural Network: Architectural Overview [6]

4. RNNs on credit card data:

We use the customer's credit card usage history as input to the RNN and predict future purchase categories as output. The RNN code submitted with the report is modelled to predict only the purchase categories but they can also be used to predict the amount associated with that purchase. To test the model, we shift the input sequence n-steps forward to test the accuracy of the results. The RNN model for our sequence prediction problem is represented below in Fig.3.

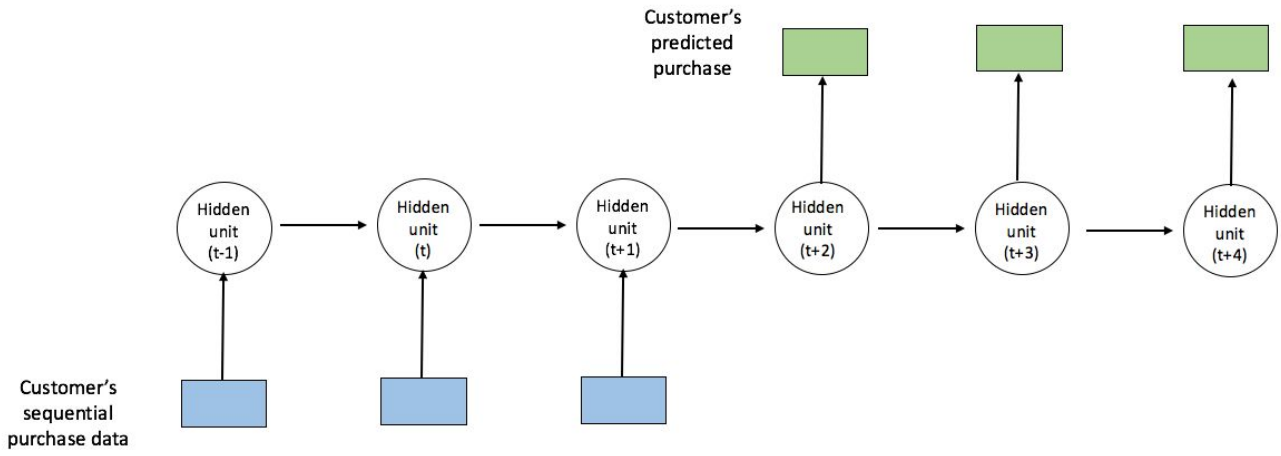

Fig.3 RNN model to predict customer's purchase based on past credit card usage data (t=time)

5.Overview of Model:

Sequential credit card usage data is first split into batches, which lets train multiple samples simultaneously. To predict sequence for a customer, one could employ two approaches: Firstly, we could train over only the customer's batched data. Secondly, we could consider historical data of similar customers as batches and train the network. The second approach would generate more accurate results considering the training dataset size is larger. For the provided fake dataset we stick to generating batches for individual sequences.

Small subsets of the batch data are sequentially used to capture historical purchase patterns. Python's TensorFlow library was used to build up a *computational graph,* that specifies what operations will be done. The input and output of this graph is typically multi-dimensional arrays, also known as *tensors*. Two basic data-structures used in Tensorflow are *placeholders* and *variables*.

On each run the batch data is fed to the placeholders, which are starting nodes of the computational graph. The weights and biases of the network (hidden-unit parameters) are declared as TensorFlow variables, which makes them persistent across runs and enables them to be updated incrementally for each batch.

When a RNN is trained, it is actually treated as a deep neural network with re-occurring weights in every layer. These layers will not be unrolled to the beginning of time, that would be too computationally expensive, and are therefore truncated at a limited number of time-steps (called Truncated Back-propagation Length). The figure below shows the input data-matrix, and the batch is in the dashed rectangle (Batch Size = 3). The batch window (Truncated Back-propagation Length = 3) is slided Truncated Back-propagation Length steps to the right at each run, hence the arrow. The dashed box is represented as '*batch data*' in the code. Schematic representation of this process is illustrated below in Fig.4.

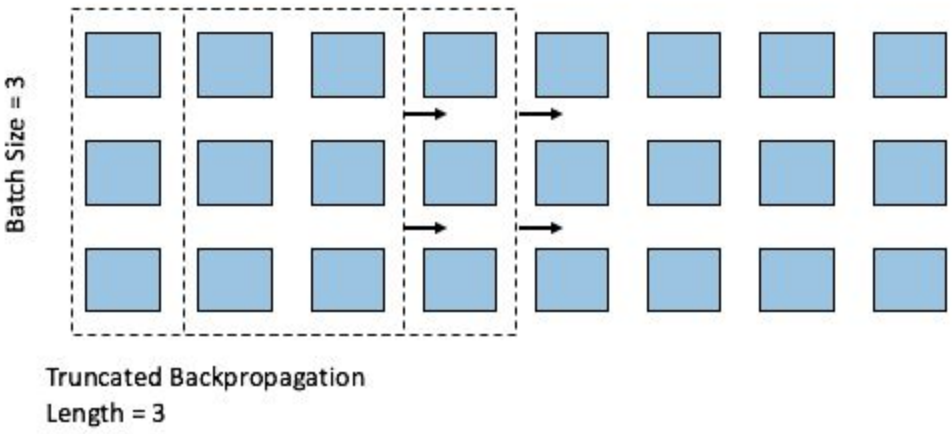

Fig.4. Schematic of training of batched data

We then build RNN Computation graph for the window. Within this graph, the *forward pass* is first made to compute the loss (error) for the batch. Then optimization is performed in the *backward pass*, where the goal is to minimize the loss by modifying the hidden weight matrices. Loss quantifies how far are the predicted results from the actual results. The final output is the predicted probability of all categories (maximum valued category is the prediction). TensorFlow performs the above operations automatically. Details of the process can be found in [7]. Results of the model can be denoted in terms of the loss computed for every batch data run. The general trend is that the loss value decreases for every run. A sample plot is attached below.

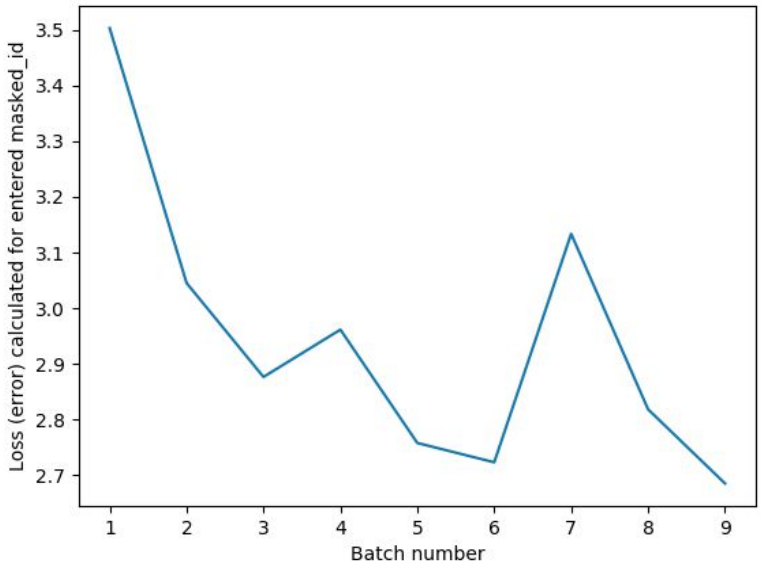

Fig.5.Sample plot for loss calculated for batch data

Neural networks generally perform better if the input data is very large. At least 1 year's historical data is recommended to provide better predictions. One could also use more advanced versions of RNN like LSTM (Long Short Term Memory) or GRU (Gated Recurrent Unit) for better prediction accuracy.

7. REFERENCES:

[1] https://www.creditcards.com/credit-card-news/payment-method-statistics-1276.php
[2] https://thefinancialbrand.com/63322/artificial-intelligence-ai-banking-big-data-analytics/
[3] https://www.valuepenguin.com/us-bank-cash-plus-credit-card
[4] https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html
[5] Lipton.Z, Berkowitz.J, Elkan.C. *A Critical Review of Recurrent Neural Networks for Sequence Learning.*
[6] http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/
[7] https://medium.com/@erikhallstrm/hello-world-rnn-83cd7105b767