Deep Time-series Extracted Features in Credit Scoring Models

Group 40

Ming-Hsiu Hu, Lin-Kuan Chiang, Jeng-Shin Huang

June 19, 2020

Dept. of Information Management and Finance, Dept. of Computer Science
National Chiao Tung University, Hsinchu, Taiwan

Outline

1. Introduction
2. Dataset
3. Exploratory Data Analysis
4. Vanilla Deep Neural Network Models Implementation
5. Deep Time-Series Extraction Models Implementation
6. Results Comparison

Introduction

Introduction

- Traditionally, banks use non time dependency models, such as linear models, to deal with credit scoring problems.
- However, the models would not be robust for more complicated real-world data and applications.
- Thus, we would like to leverage time-related deep learning models to provide better insights for this problem.

Dataset

Dataset

- We use Default of Credit Cards Clients Dataset.
- This dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005, collected by UCI Machine Learning Repository.
- Thus, with features' names for each columns, we believe this dataset would provide us more insights than most of other datasets that only have anonymous features.

Dataset

- Number of Data: 30,000
- Features: 23 features
  - Static Features
  - Dynamic Features
- Label: 0 or 1
  - Binary Formation
  - Indicate whether the clients would default next month.
  - Default = 1, No Default = 0.

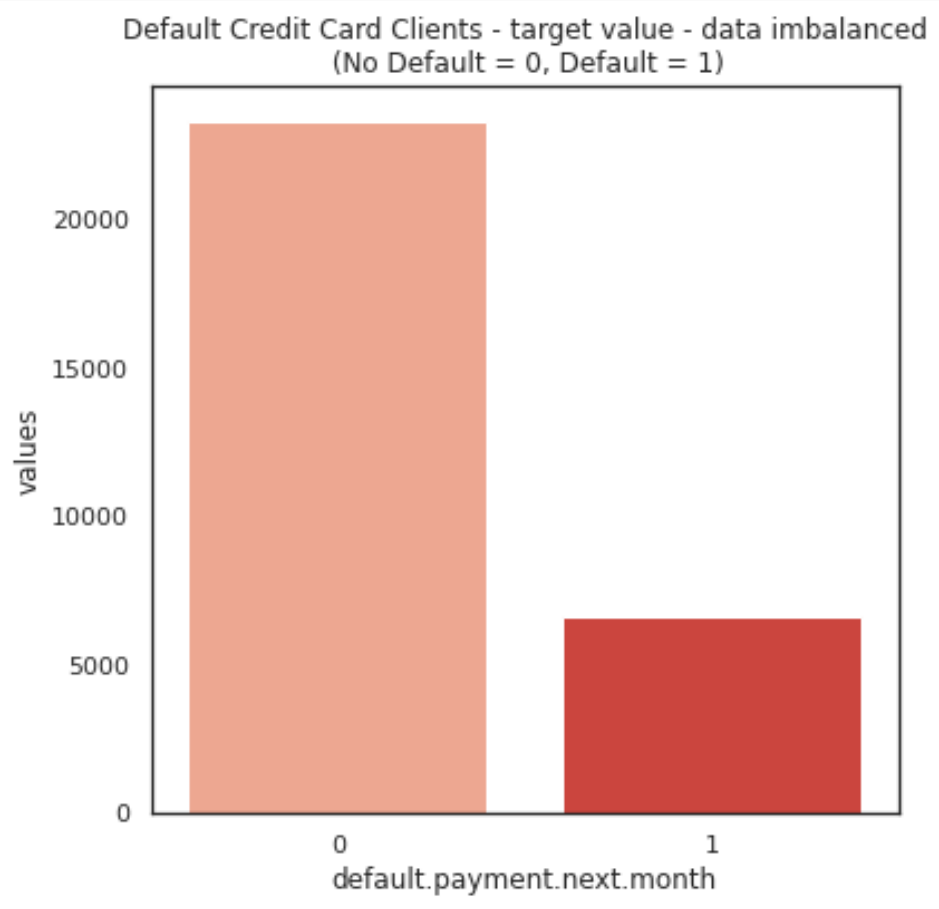
Dataset

- Static Features:
  - Amount of given credit in NT dollars (X1)
  - Demographics: Sex (X2), Education (X3), Marriage (X4), Age (X5)
- Dynamic Features:
  - Customers' past 6-month credit card payment history
    - Repayment Status (X6-X11)
    - Amount of Bill Statement (X12-X17)
    - Amount of Previous payment (X18-X23)

Exploratory Data Analysis

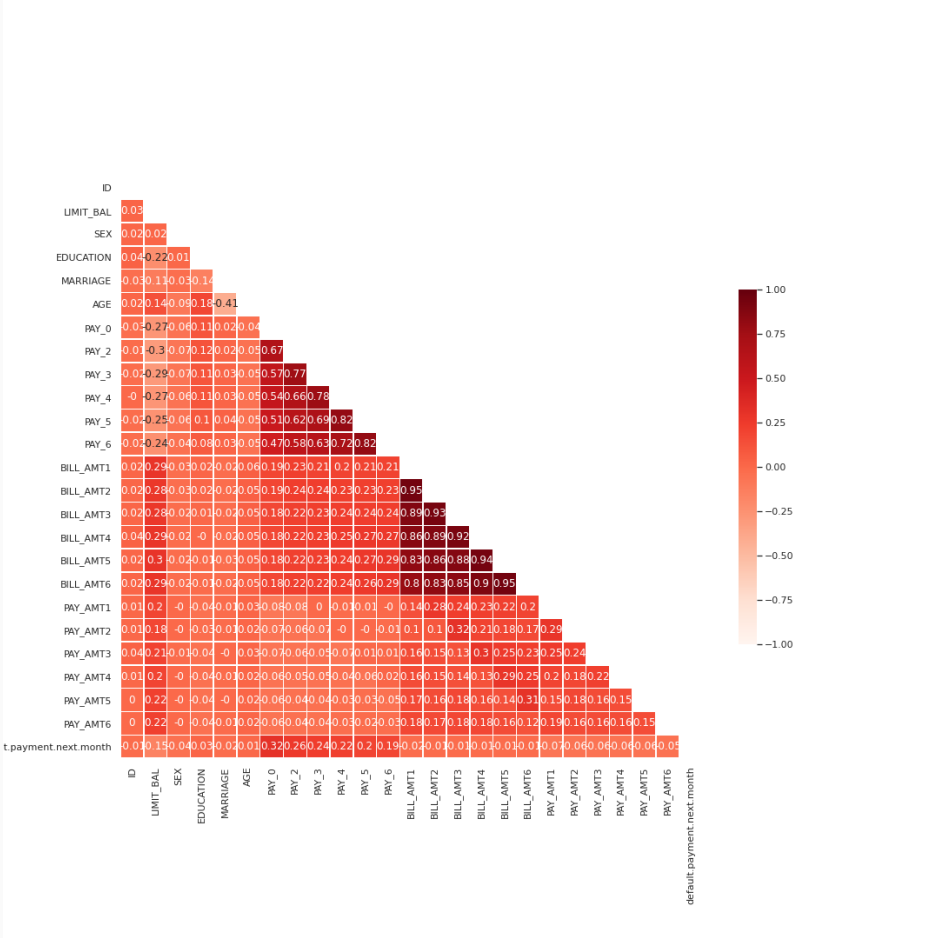
Exploratory Data Analysis

- The target data suffers from an imbalance data problem.



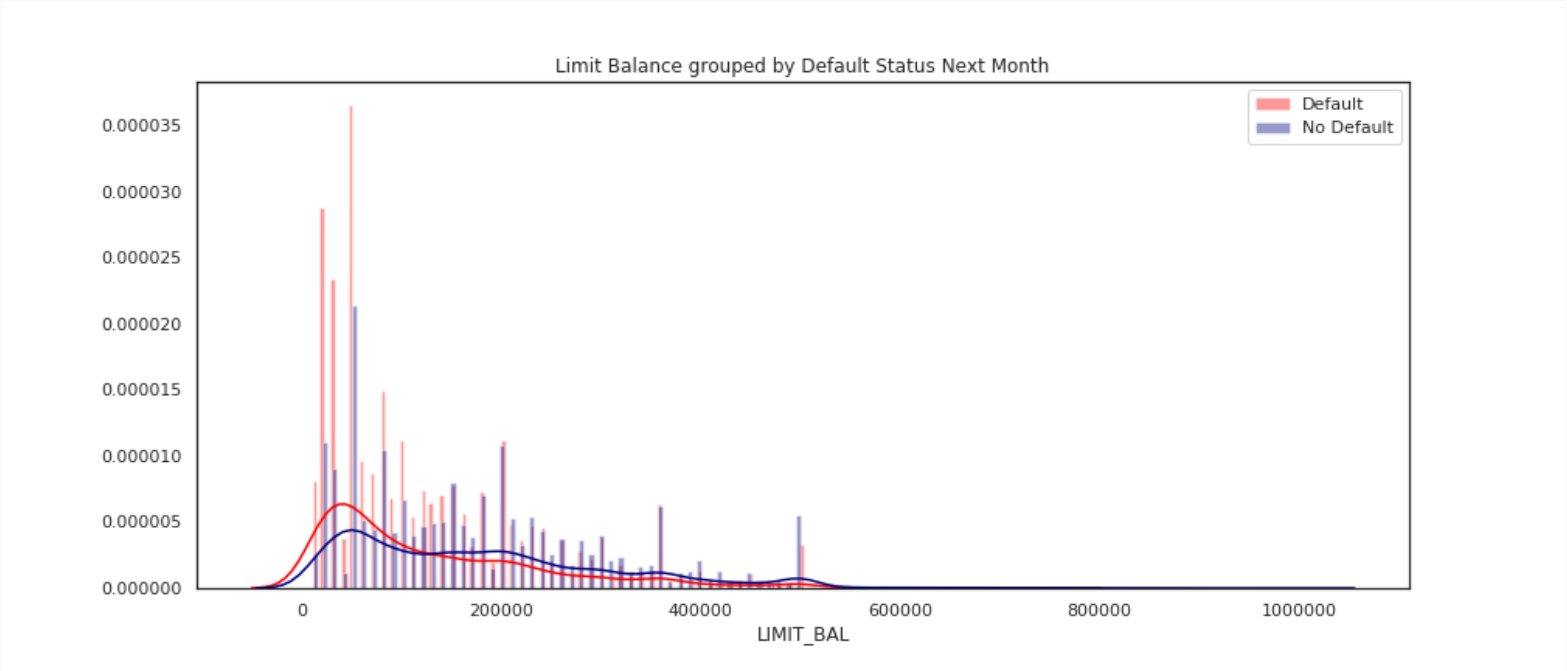
Exploratory Data Analysis

- We plot heatmap to find out the correlations between features.
- Dynamic Features are highly correlated with time.



Exploratory Data Analysis

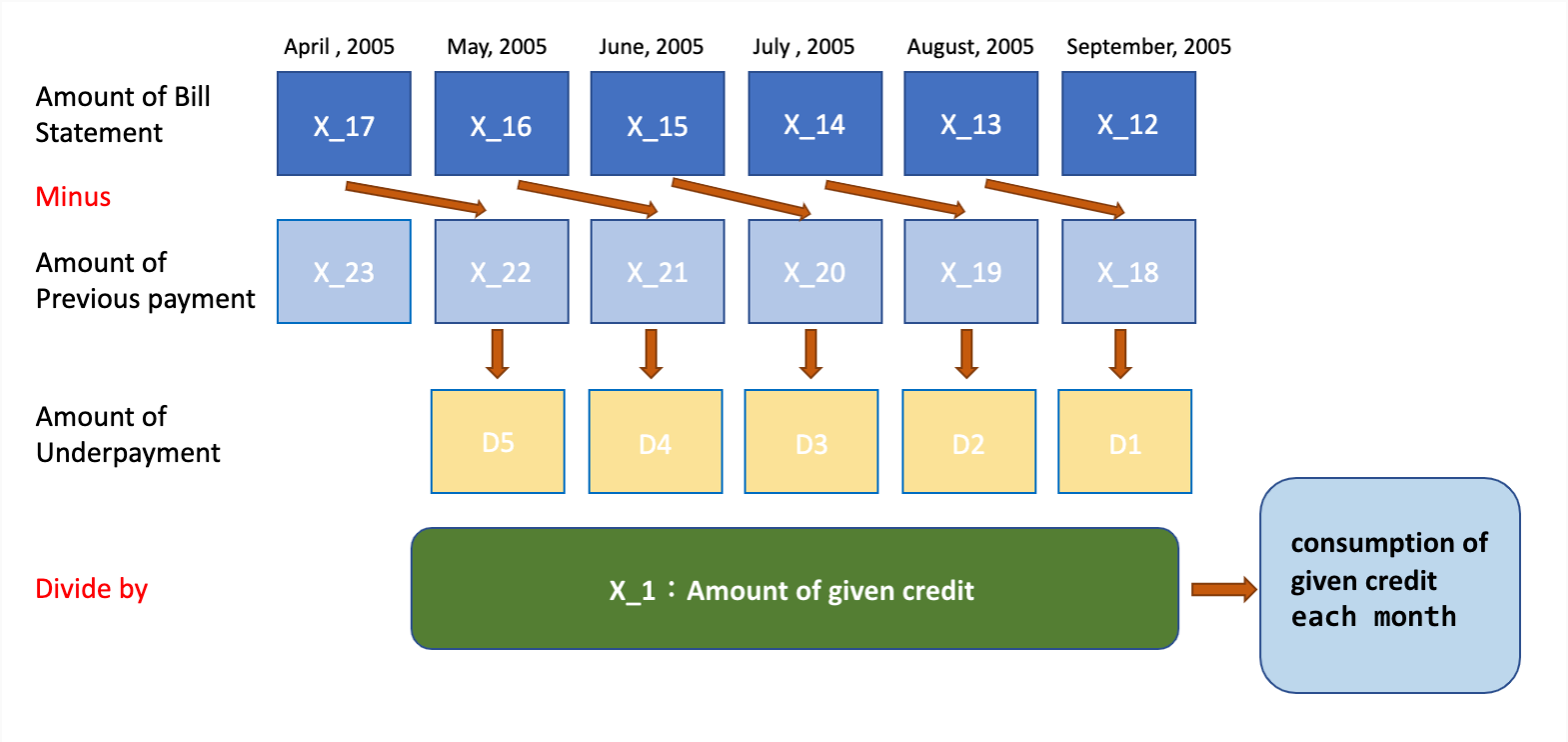
- We plot the limit balance grouped by default status next month.
- Most of defaults are for credit limits between NT 0 - NT 100,000.



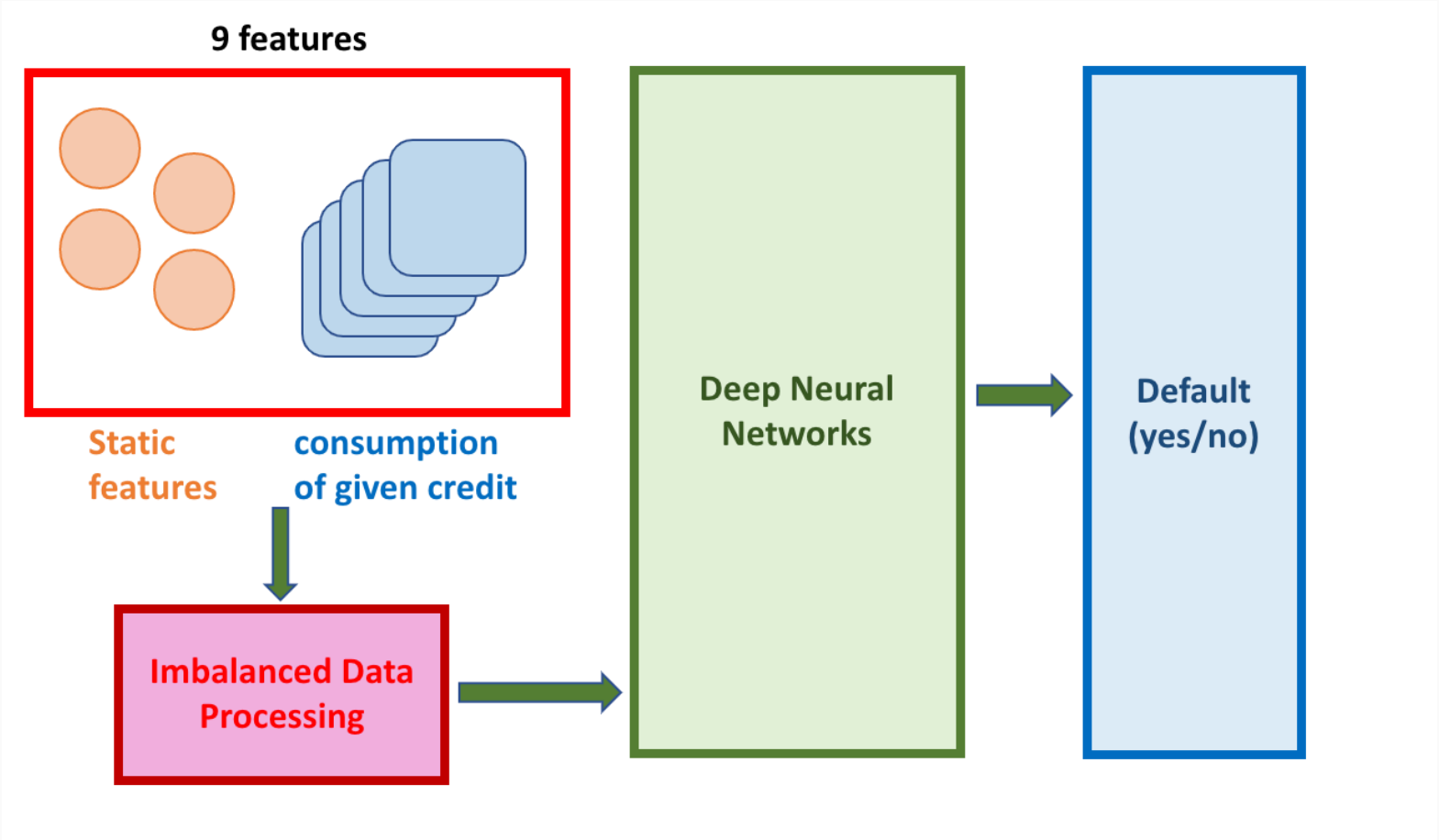
Vanilla Deep Neural Network Models Implementation

Vanilla DNN

- We use the two dynamic features, amount of previous payment and bill statement, to calculate consumption of given credit each month.



Vanilla DNN


9 features
Static features
consumption of given credit
Imbalanced Data Processing
Deep Neural Networks
Default (yes/no)

Hyperparameters Setting for Vanilla DNN

- number of layers = 3
- EPOCH = 1000
- learning rate = 0.0001
- Activation Function = ReLU

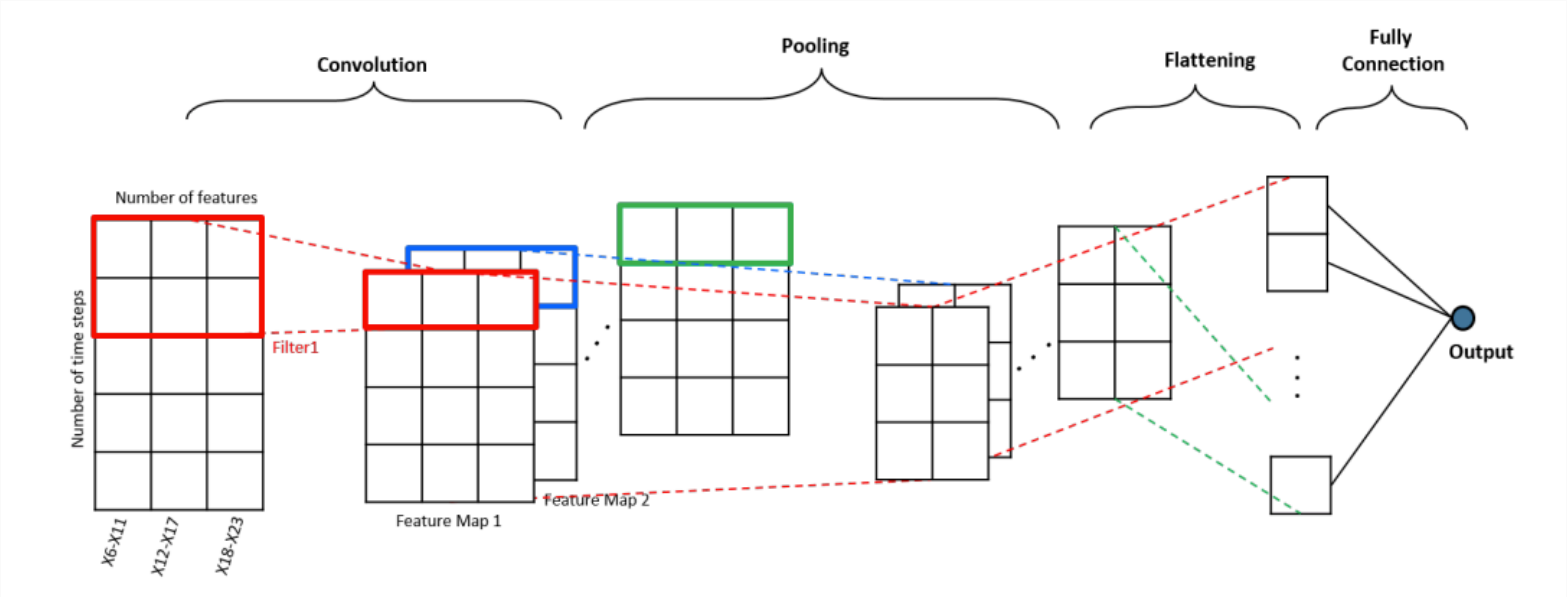
Deep Time-Series Extraction Models Implementation

Time-correlated Features Extraction

- Apparently, some of the dynamic features are highly time-correlated shown in the heatmap.
- Thus, we would like to extract some latent features from dynamic features using RNN(LSTM)/1D-CNN first.

Time-correlated Features Extraction (1D-CNN)

- Arrange dynamic features in the following format.
- Set time-step = 2 months.
- Extract latent features using 1D-CNN.

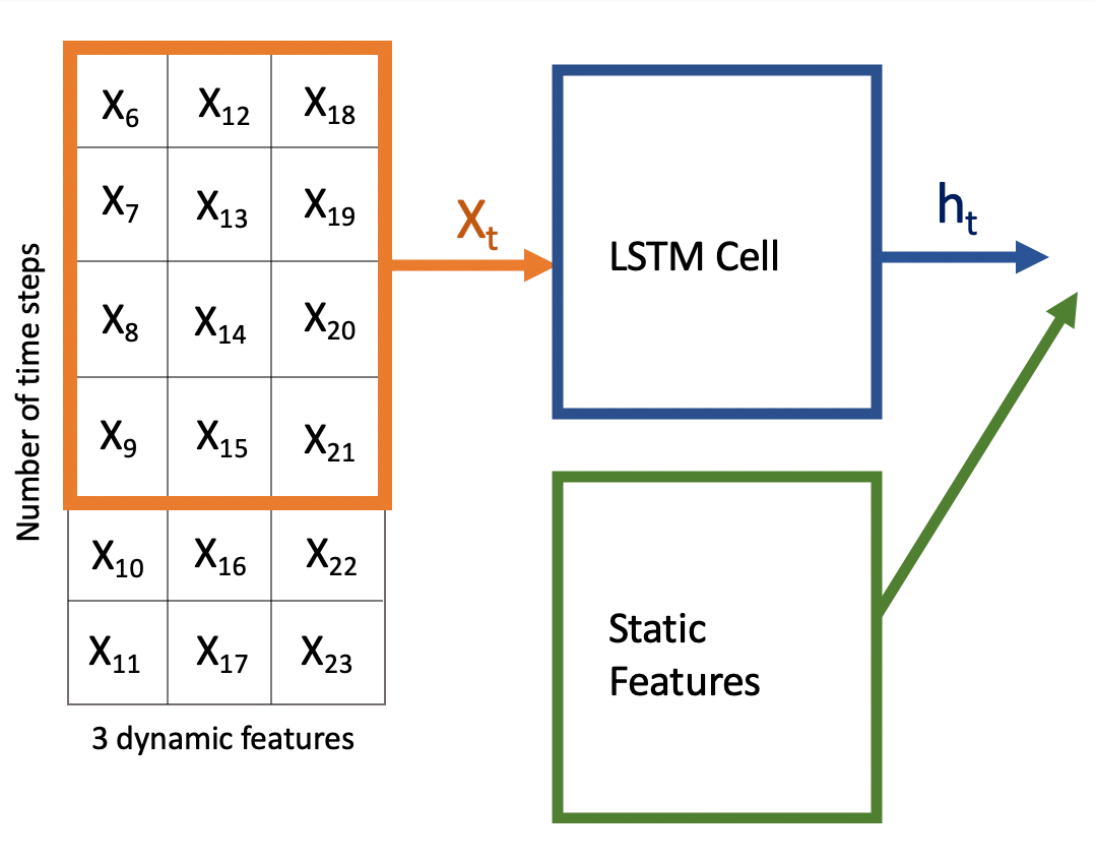


Hyperparameters Setting for 1D-CNN Extraction

- Conv1d (in feature = 3, out feature = 16, kernel size = 2)
- Conv1d (in feature = 16, out feature = 32, kernel size = 4)
- batch size = 128
- EPOCH = 500
- learning rate = 0.0001
- Activation Function = ReLU

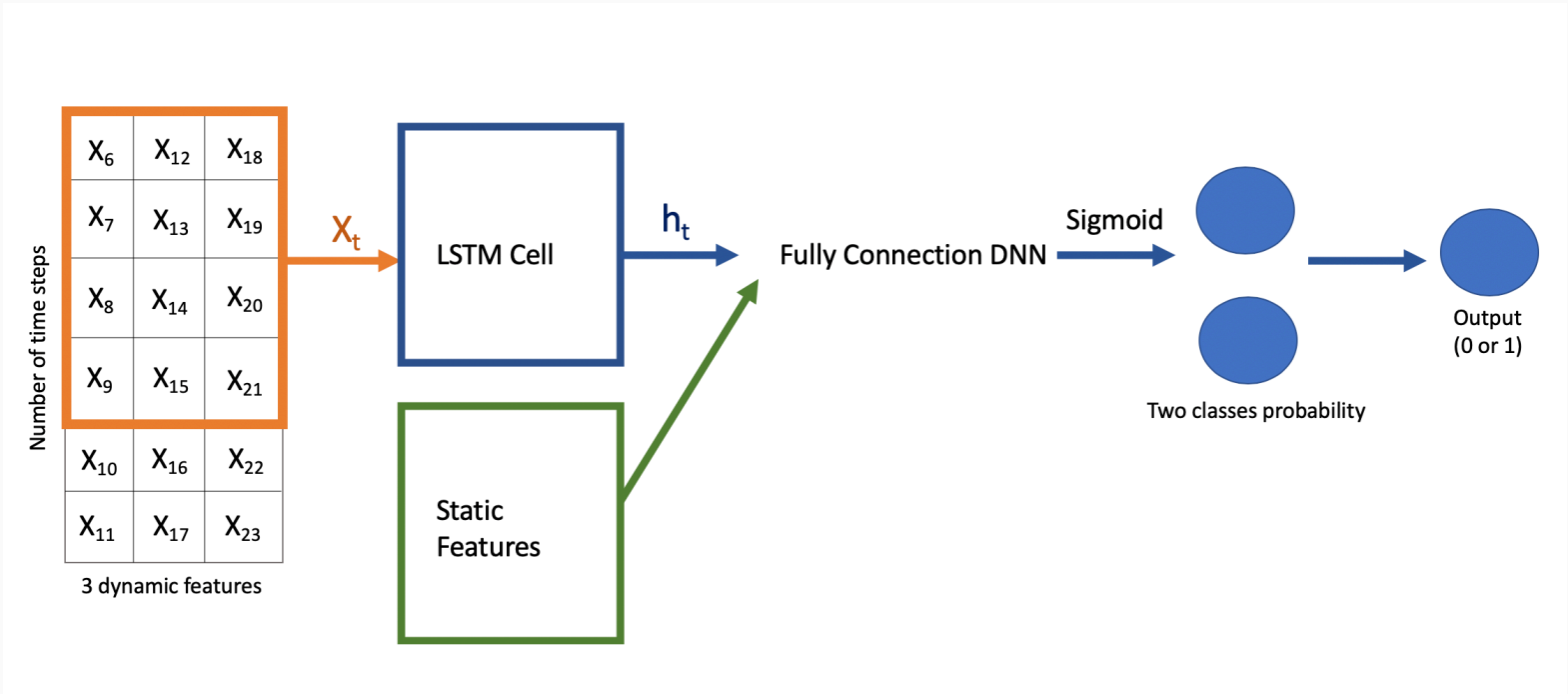
Time-correlated Features Extraction (LSTM)

- Arrange dynamic features in the following format.
- Set time-step = 4 months.
- Extract latent features using LSTM cell.



Time-correlated Features Extraction (LSTM)

- Concatenate latent with static features
- Feed in Fully-connected DNN
- Acquire the predicted output (Default/ No Default)



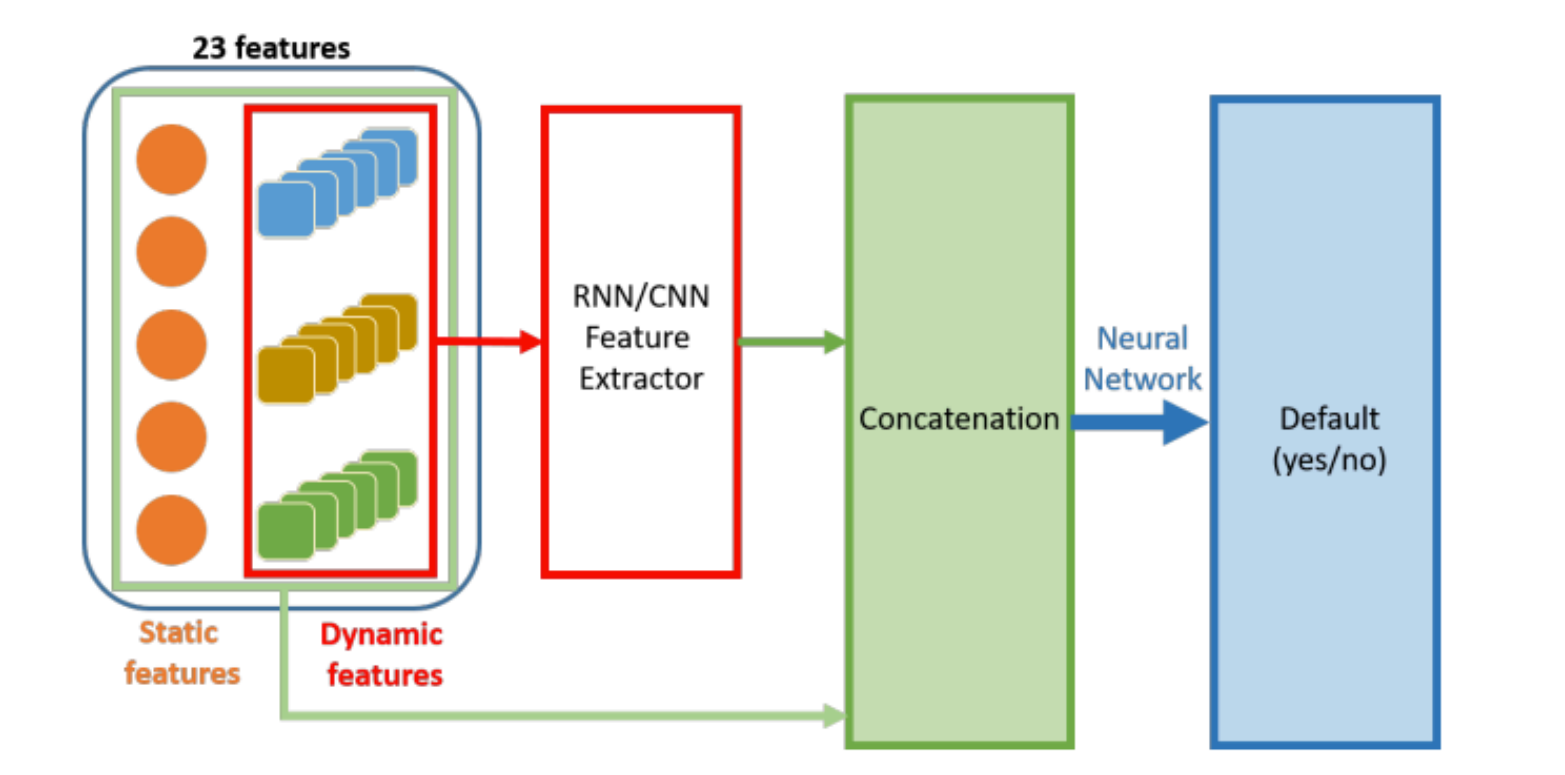
Hyperparameters Setting for LSTM Extraction

- input dimension = 1
- hidden dimension = 512
- number of layers = 2
- EPOCH = 1000
- learning rate = 0.00001
- Activation Function = Sigmoid

Loss function for LSTM Extraction

- We use nn.BCEWithLogitsLoss from pytorch.
- This loss combines a Sigmoid layer and the BCELoss.
- $l(x, y) = L = \{l_1, l_2, l_3 ... l_N\}^T$, where N = batch size
- $l_n = -w_n[y_n * log\sigma(x_n) + (1 - y_n) * log(1 - \sigma(x_n)]$
- Then, $l(x, y) = mean(L)$
- Besides, we also pass in pos weight = [3.54] into the loss function in order to solve the imbalance data problem.

The complete model architecture



Results Comparison

Training Results

Training Results

Model	Accuracy	Precision	Recall	F1-Score	AUC
Logistic Regression	0.63	0.31	0.57	0.41	0.64
Vanilla DNN	0.65	0.59	0.45	0.51	0.61
1D-CNN+DNN	0.65	0.31	0.45	0.36	0.58
LSTM+DNN	0.84	0.86	0.87	0.86	0.78

Testing Results

Testing Results

Model	Accuracy	Precision	Recall	F1-Score	AUC
Logistic Regression	0.62	0.31	0.56	0.40	0.62
Vanilla DNN	0.64	0.60	0.44	0.50	0.61
1D-CNN+DNN	0.66	0.30	0.44	0.36	0.58
LSTM+DNN	0.79	0.84	0.72	0.77	0.79

Conclusion and Future works

- The traditional prediction, such as logistic regression, can be improved by using our data-preprocessing method and time-correlated features extraction model of LSTM+DNN.
- We should check for time-correlated relationship for credit-card default data like this in the future.

Questions?