2022.01.21玉山成果發表會

基於智慧取樣與自監督技術之信用卡盜刷偵測2.0

指導教授:黃思皓

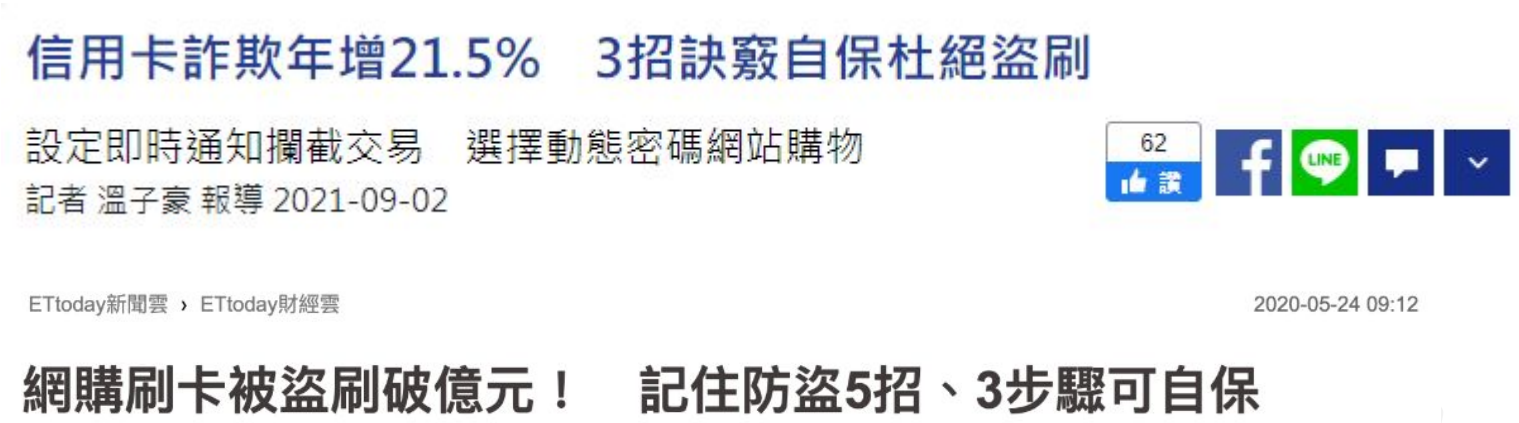
交大研究團隊:陳巧莛、呂姿瑀、陳成峯、李齊、周信廷

玉山研究團隊:葉倚任、陳品瑜

Financial Technology Innovation

問題定義

- 透過信用卡交易的資料, 刷卡發生須及時判斷是否為盜刷交易, 若為盜挖則由人工介入做後續處理
- 判斷盜刷與否可視作**二元分類問題, 盜刷的類別是1;非盜刷的類別則是0**



問題定義

信用卡盜刷偵測2.0需要解決玉山提出的四個問題:

1. **Sampling / Concept Drift**
   - 要使用那些交易資料作為訓練資料？長期資料會更好？
2. **Noise Label**
   - 盜刷不一定能即時發現, 會在一段時間後才回報
3. **Dynamic Threshold**
   - 每筆刷卡需即時判定, 一週僅能人工處理5000筆疑似盜刷的交易
4. **花費時間**
   - 每筆交易需要在0.1秒內即時反應

資料介紹

- 使用2020/05/01~2020/11/30玉山信用卡交易的資料共133,688,582筆

其中包含盜刷資料: 57,571筆

非盜刷資料:133,631,011筆

盜刷資料佔所有資料的0.043%

- 每一筆資料在交易後會透過玉山的盜刷偵測機制或是由使用者向玉山回報, 來判斷該筆交易是否為盜刷
- 通常盜刷的紀錄只有80%會在交易結束後兩個禮拜內被發現

前情回顧

動機:現有交易資料表達能力不足，盜刷模型無法有效學習有用的資訊

- 從**金額、時間、地點**發想，為現有資料設計新的特徵

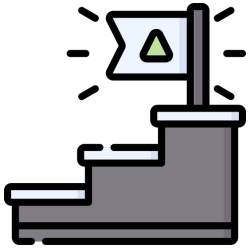
(交易資料特徵共125維)

金額	時間	地點
與前一筆交易的金額差	與前一筆交易的時間差	是否與前一筆交易的消費地點相同
過去時間內消費的累積金額	時間範圍內交易次數	是否曾經在該國家、地點消費過
過去時間內在相同交易地點消費的累積金額	交易時間是否坐落於高風險區間	同一交易地點(國家、特店)佔過去消費次數的比例
與過去消費金額的差距是否大於1個標準差	交易時間差是否小於1分鐘	過去消費過的地點(國家、特店)總數

前情回顧 結論

- 目前盜刷模型 Recall@5000有**40%以上**，符合預期
- 針對2.0盜刷偵測模型需要解決的問題，提出看法：
  - **Sampling**：參考研討會、期刊論文，提出Negative Sample構想
  - **Noise Label**：實驗不同的Label方式
  - **Dynamic threshold**：統計Validation Set盜刷佔比，得到盜刷資料的分配情形
  - **花費時間**：單一筆資料Testing十分快速，需要考慮Feature Engineering時間
  - **深度學習技術應用在盜刷偵測議題上**：Self-supervised Learning

實驗過程



盜刷模型
第一次成果發表會提出，作為本次的Baseline

Negative Sampling
挑選有幫助的訓練資料改善Noise Label

Self-supervised Learning
增強序列資料的Representation

Dynamic Threshold
使資源能最有效的利用

Clipping
解決極端值

Positive Accumulated
學習更多盜刷資訊

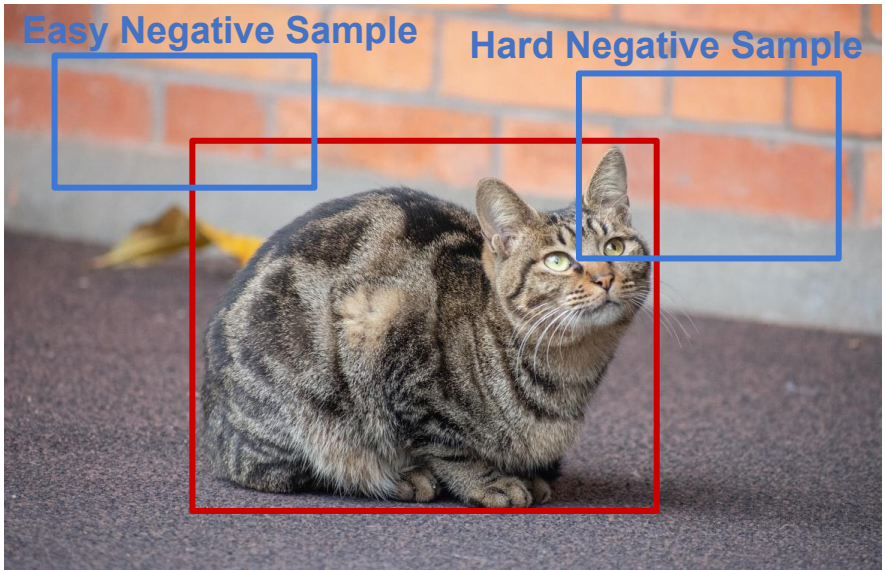
Hybrid
最終盜刷模型

01

Negative Sampling 實驗

Sampling

目標: 雖然放入越多的訓練資料效果越好，但非盜刷交易高達133,631,011筆，太過消耗訓練的資源，且盜刷交易存在Noise Label問題



- Reliable Negative Sample
  - 對模型而言**容易**辨識的Negative Sample
- Representative Negative Sample
  - 對模型而言**難以分辨**的Negative Sample，潛在的**Noise Label**

Rule-based Sampling

目標:使用自行設計的規則, 定義Reliable/Representative Sample

- 統計欄位中不同特徵分別的「盜刷率」
- 使用統計結果作為區分Reliable/Representative Sample的依據
  - 使用欄位:特店國家(stocn)、特店類別(mcc)、交易型態(etymd)
- 以Reliable/Representative Sample各自訓練盜刷模型, 並進行預測

Embedding-based Sampling

目標:

Rule-based方法找出的Sample不具代表性, 對盜刷模型幫助不大

- 透過DNN學習, 將原始資料轉換成具有代表性的Embedding
- 透過Under-sampling演算法移除部分非盜刷交易, 解決Noise Label問題
  - NearMiss-1 : 計算每個非盜刷交易與**K個最近的盜刷交易**的平均距離, 移除**N筆**距離最短的非盜刷交易 (移除Representative Sample)

Embedding-based Sampling 結果

- 嘗試不同的Embedding Size與移除不同比例的Negative Sample

Testing Recall @5000	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
沒有 Sampling	0.375	0.312	0.480	0.476	**0.552**	0.321	**0.419**
Emb 64 維 (移除10%)	0.379	0.320	0.476	0.484	0.019	**0.363**	0.340
Emb 32 維 (移除10%)	**0.392**	0.323	0.501	0.503	0.311	0.362	0.399
Emb 32 維 (移除20%)	0.375	0.046	0.495	**0.505**	0.522	0.034	0.330
Emb 32 維 (移除30%)	0.375	**0.331**	0.495	**0.505**	0.564	0.123	0.399
Emb 32 維 (移除40%)	0.227	0.029	0.497	0.486	0.506	0.347	0.348
Emb 32 維 (移除0.1%)	0.387	0.319	**0.511**	0.476	0.313	0.351	0.393

Embedding-based Clipping 結果

Testing Recall @5000	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
沒有 Sampling	0.375	0.312	0.480	0.476	0.552	0.321	0.419
Emb 32 維 (移除0.1%)	0.384	0.348	0.480	0.509	0.538	**0.303**	0.427
Emb 32 維 (移除0.01%)	0.397	**0.375**	0.480	0.521	0.541	0.283	**0.433**
Emb 32 維 (移除0.001%)	**0.403**	0.352	0.488	0.510	0.543	0.268	0.427
Emb 32 維 (移除0.0001%)	0.375	0.346	**0.498**	**0.523**	**0.551**	0.280	0.429

Embedding-based Sampling 觀察

- 在最難分類的 Representative Sample 中，特定欄位與 Positive Sample 特徵極為相似
  - 類別型：特店類別(mcc)、消費地國別(stocn)、交易類別(contp)、支付型態(hcefg)、交易型態(etymd)
  - 數值型：交易金額(flam1)
- Representative Sample txkey：
  - VS0I00120200714AADAJ
  - MC0I00120200707ABHXC

Embedding-based Highlight 實驗

目標:

讓模型更有效地學習有幫助的訓練資料

- Embedding-based方法已經移除最難分類的 Representative Sample, **解決 Noise Label 問題**
- 以兩種方式調整選出的次Representative Data的資料權重
  - 直接複製
  - 更動資料權重

Embedding-based Highlight 結果 (複製資料)

Testing Recall @5000	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
沒有 Sampling	0.375	0.312	0.480	0.476	0.552	0.321	0.419
Emb 32 維 (移除0.1%)	0.395	0.350	**0.501**	**0.517**	0.535	0.286	0.431
Emb 32 維 (移除0.01%)	0.403	**0.378**	0.482	0.513	**0.560**	0.286	**0.437**
Emb 32 維 (移除0.001%)	0.372	0.367	0.483	0.494	0.501	**0.298**	0.419
Emb 32 維 (移除0.0001%)	**0.409**	0.324	0.341	0.489	0.538	0.289	0.425

Embedding-based Highlight 結果 (調整權重)

Testing Recall @5000	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
沒有 Sampling	0.375	0.312	0.480	0.476	0.552	0.321	0.419
Emb 32 維 (移除0.1%)	0.386	**0.364**	**0.505**	0.498	0.548	**0.309**	0.435*
Emb 32 維 (移除0.01%)	**0.394**	0.356	0.501	**0.519**	0.546	0.301	**0.436**
Emb 32 維 (移除0.001%)	0.375	0.348	0.497	0.510	**0.576**	0.282	0.431*
Emb 32 維 (移除0.0001%)	0.390	0.347	0.497	0.508	0.566	0.292	0.433*

Embedding-based Sampling 結論

Testing Recall @5000	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
沒有 Sampling	0.375	0.312	0.480	0.476	0.552	0.321	0.419
Emb 32 維 (移除10%)	0.392	0.323	0.501	0.503	0.311	0.362	0.399
Emb 32 維 (移除0.1%)	0.387	0.319	0.511	0.476	0.313	0.351	0.393
Emb 32 維 (移除0.1%)(Clipping)	0.384	0.348	0.480	0.509	0.538	0.303	0.427
Emb 32 維 (移除0.01%)(Clipping)	0.397	0.375	0.480	0.521	0.541	0.283	0.433
Emb 32 維(移除0.001%)(Clipping)	0.403	0.352	0.488	0.510	0.543	0.268	0.427
Emb 32 維 (移除0.01%)(Clipping、調整權重)	0.394	0.356	0.501	0.519	0.546	0.301	0.436
Emb 32 維 (移除0.01%)(Clipping、複製資料)	0.403	0.378	0.482	0.513	0.560	0.286	0.437

02

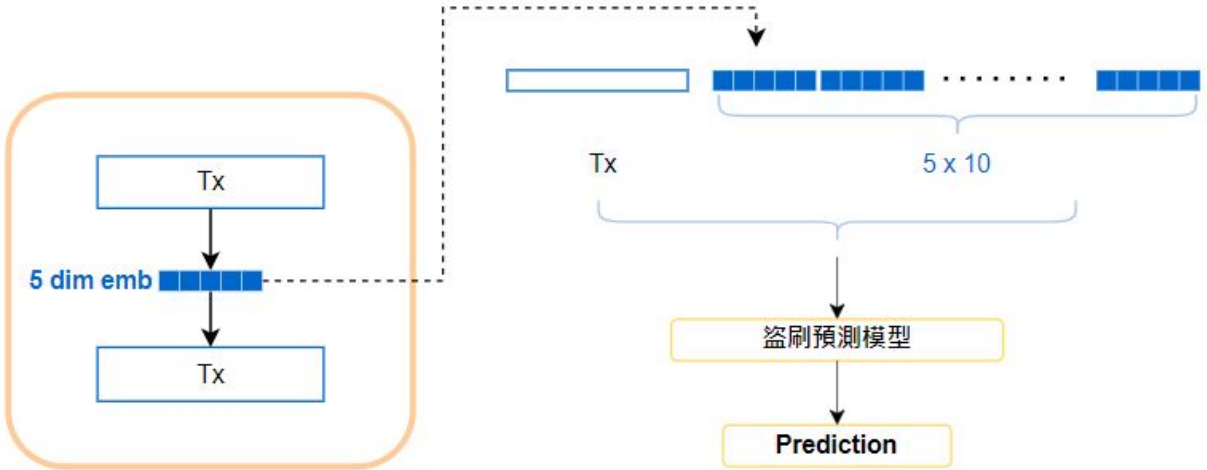
Self-Supervised Learning (SSL) 實驗

User Behavior

目標:將User Behavior加入模型，讓網路可以學習到更多隱含的資訊

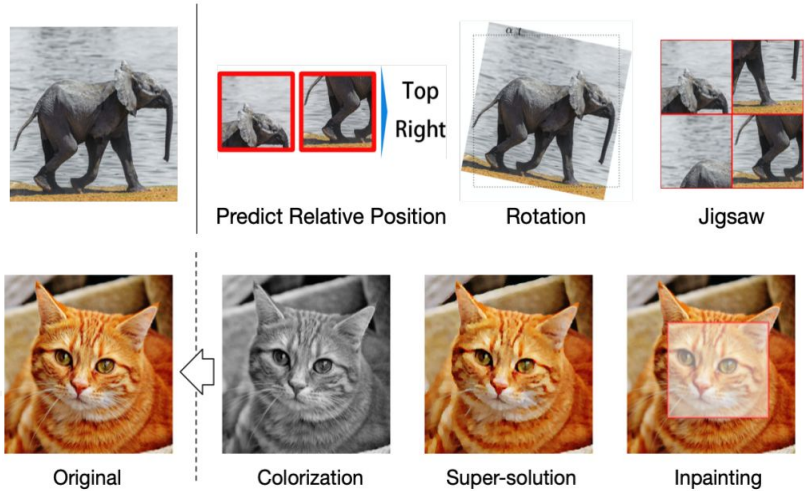
問題:Sequence Data維度過高，容易產生記憶體問題

解決方法:訓練AutoEncoder壓縮Sequence Data



Self-Supervised Learning(SSL)

- 從原始的資料中透過**資料擴充**或設計**新定義的任務(Auxiliary Task)**來挖掘更豐富的資料特徵, 取得更通用的Data Representation
- 幫助**主任務**學習到更全面的資訊



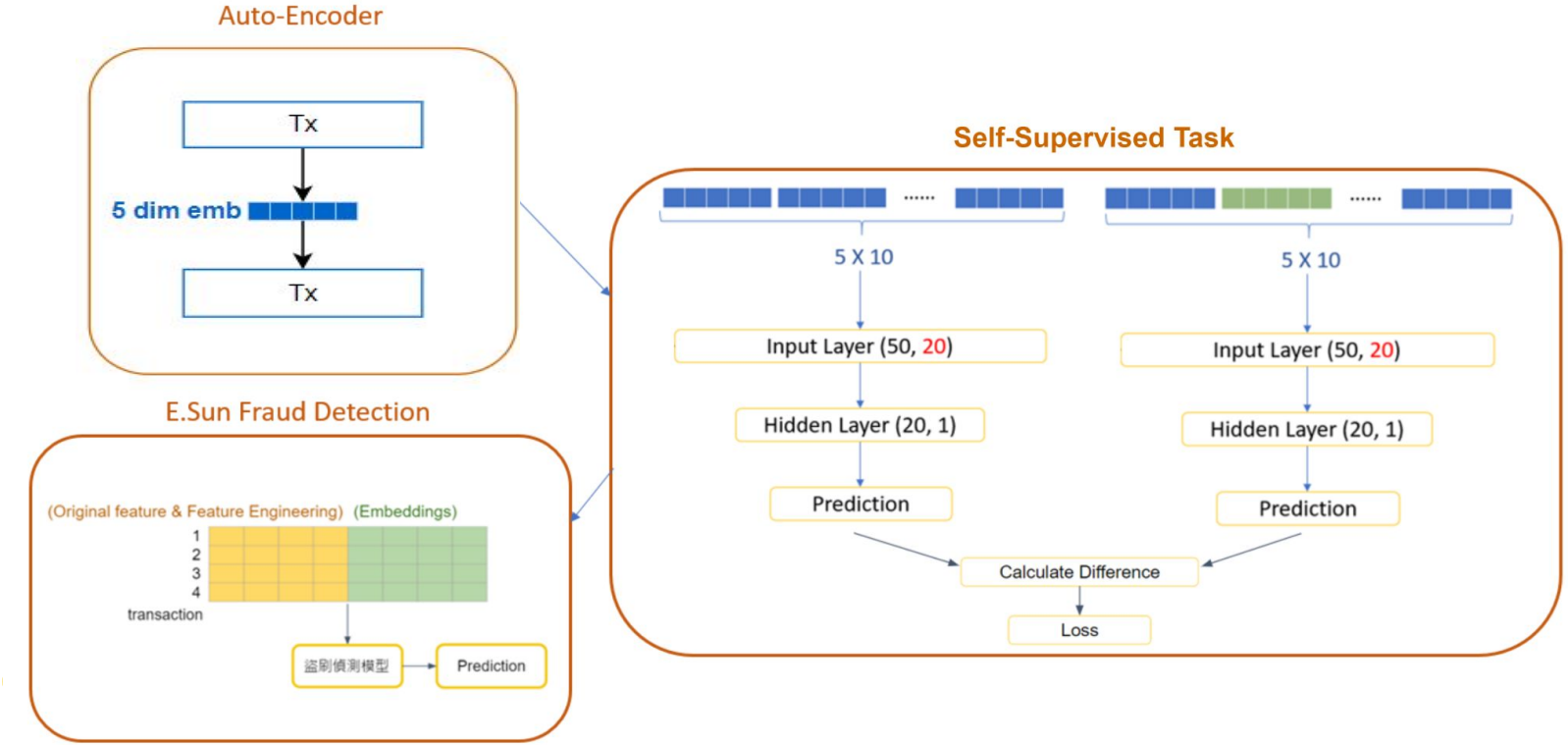
User Behavior

問題:AutoEncoder雖然改善資料維度過高的問題，卻因為壓縮維度過小，可能造成資訊不足的問題。

解決方法:設計Self-Supervised Tasks，藉此取得更具代表性的Sequence Data Representation

- 訓練模型判斷真實與偽造的交易Sequence
- Task 1：Binary Cross Entropy Loss
- Task 2：Pair Wise Learning to Rank

AutoEncoder with SSL實驗架構



AutoEncoder with SSL 結果

	Sep W3	Sep W4	Oct W1	Oct W2	Oct W3	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
A	0.597	0.353	0.311	0.504	0.424	0.410	0.423	0.448	0.404	0.322	0.421
B	0.525	0.164	0.269	0.362	0.281	0.348	0.424	0.413	0.425	0.254	0.347
B 1	0.570	0.258	0.320	0.455	0.397	0.362	0.399	0.412	0.467	0.280	0.392
B 2	0.572	0.253	0.333	0.404	0.378	0.446	0.361	0.418	0.446	0.307	0.392

A: 單筆交易
B: Auto-Encoder 壓縮Sequence Data
B 1 / B 2: Auto-Encoder + SSL，Task1 / Task2

User Behavior

問題:AutoEncoder加上Self Supervised Learning雖然有稍微改善，但盜刷預測結果仍然不佳

解決方法:將完整Sequence放進設計的Self Supervised Tasks，讓模型學習到完整的交易資訊，並取得具代表性的Sequence Data Representation

實作:因為完整Sequence容易產生記憶體問題，因此將資料進行Sampling

User Behavior - Self-Supervised Learning

	Sep W3	Sep W4	Oct W1	Oct W2	Oct W3	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
A	0.597	0.353	0.311	0.504	0.424	0.410	0.423	0.448	0.404	0.322	0.421
C' 1	0.640	0.378	0.331	0.460	0.403	0.406	0.436	0.461	0.455	0.356	0.433
C' 2	0.640	0.341	0.320	0.483	0.389	0.422	0.442	0.493	0.490	0.356	0.438

A: 單筆交易

C' 1 / C' 2: Sampling Sequence Data + SSL，Task1 / Task2

User Behavior

	Sep W3	Sep W4	Oct W1	Oct W2	Oct W3	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
A	0.597	0.353	0.311	0.504	0.424	0.410	0.423	0.448	0.404	0.322	0.421
C' 1	0.640	0.378	0.331	0.460	0.403	0.406	0.436	0.461	0.455	0.356	0.433
C' 2	0.640	0.341	0.320	0.483	0.389	0.422	0.442	0.493	0.490	0.356	0.438
B 1	0.570	0.258	0.320	0.455	0.397	0.362	0.399	0.412	0.467	0.280	0.392
B 2	0.572	0.253	0.333	0.404	0.378	0.446	0.361	0.418	0.446	0.307	0.392

A: 單筆交易
B 1 / B 2: Auto-Encoder + SSL，Task1 / Task2
C' 1 / C' 2: Sampling Sequence Data + SSL，Task1 / Task2

Positive Accumulated

目標:

盜刷樣本的資料珍貴，且盜刷模式可能會繼續保留

問題:

- 不同期間的盜刷Pattern不完全相同，保留太長期的盜刷資料，可能衍生**Concept Drift**問題
- 先前實驗:使用更長期、更多筆交易資料訓練模型，反而讓實驗效果下降

Positive Accumulated

實驗設定:

- 除了訓練區間的盜刷之外, 額外保留所有的盜刷交易
- 使模型可以學習更長遠的盜刷特徵
- 避免保留太長期的盜刷資料, 期望解決**Concept Drift**的問題

Positive Accumulated

	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
A	0.410	0.423	0.448	0.404	0.322	0.401
A(全部累積)	0.420	0.431	0.448	0.415	0.320	0.407
A(累積五期)	0.420	0.425	0.452	0.416	0.322	0.399
A(累積四期)	0.430	0.439	0.457	0.418	0.333	0.436
A(累積三期)	0.420	0.436	0.438	0.412	0.326	0.406
A(累積二期)	0.418	0.434	0.441	0.415	0.322	0.406
A(累積一期)	0.418	0.421	0.438	0.408	0.311	0.399

User Behavior 結論

	Sep W3	Sep W4	Oct W1	Oct W2	Oct W3	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
A	0.597	0.353	0.311	0.504	0.424	0.410	0.423	0.448	0.404	0.322	0.421
B 1	0.570	0.258	0.320	0.455	0.397	0.362	0.399	0.412	0.467	0.280	0.392
B 2	0.572	0.253	0.333	0.404	0.378	0.446	0.361	0.418	0.446	0.307	0.392
C' 1	0.640	0.378	0.331	0.460	0.403	0.406	0.436	0.461	0.455	0.356	0.433
C' 2	0.640	0.341	0.320	0.483	0.389	0.422	0.442	0.493	0.490	0.356	0.438
Final	0.645	0.4	0.34	0.55	0.466	0.461	0.452	0.461	0.43	0.371	0.458

A: 單筆交易
B 1 / B 2: Auto-Encoder + SSL，Task1 / Task2
C' 1 / C' 2: Sampling Sequence Data + SSL，Task1 / Task2
Final: Sampling Sequence Data (累積四期positive) + SSL，Task2

玉山產學框架

目的: 協助整合深度學習實驗流程

- 包含資料預處理、參數選擇與最佳化、模型訓練、模型測試, 以及其他實驗步驟
- 基於PyTorch、PyTorch Lightning開發
- 目前於玉山研發雲中實作
  - 需選擇指定環境 Python 3.7; Tensorflow 2.4; CUDA 11 (MLaaS Develop)
  - 需安裝指定版本套件 (PyTorch、TensorBoard)

玉山產學框架 實作

目前實作完成:

- 範例實驗實作與測試
- 與盜刷專案套件相容性測試
- 實作基礎盜刷模型，並使用交易資料進行測試

03

Dynamic Threshold 實驗

Dynamic Threshold統整

- **目標:** 由於資源有限，目標是模型每週預測出最有可能是盜刷的5000筆資料，希望設計出會隨每週情況而改變的Threshold

版本一:透過Validation set，取得當期的Threshold

版本二:依據已確認的盜刷筆數、該週剩餘時間決定Threshold

版本三:加入週期性盜刷量統計資訊決定Threshold

版本四:同版本三，且依據已確認的盜刷筆數、該週剩餘時間調整Threshold

版本五:依據已確認的盜刷筆數、該週剩餘時間調整Threshold

Dynamic Threshold 版本五

- Threshold的初始值會參考Validation Set**盜刷量統計資訊**
- 盜刷量佔比統計量的計算, 除了盜刷量之外, 也會參考當日的交易數量
- Threshold會依據**已確認的盜刷筆數、該週剩餘時間**動態的調整

	星期一	星期二	星期三	星期四	星期五	星期六	星期日	總和
當日盜刷量	100	200	50	150	100	200	200	1000
當日交易量	60000	80000	50000	60000	50000	65000	60000	4250000
盜刷量佔比	0.17%	0.25%	0.10%	0.25%	0.20%	0.31%	0.33%	0.24%
盜刷量佔比(標準化)	10.39%	15.56%	6.22%	15.56%	12.45%	19.10%	20.72%	100%

Dynamic Threshold 計算方法

- 盜刷 Quota 數量為 7天 5000筆，依據Validation Set統計盜刷量佔比如下表

	星期一	星期二	星期三	星期四	星期五	星期六	星期日	總和
盜刷量佔比 (標準化)	10.39%	15.56%	6.22%	15.56%	12.45%	19.1%	20.72%	100%

1. 假設Testing 第一天是週一，應該分配10.39%的 Quota 的盜刷量
2. 依照確認的盜刷筆數，使用統計結果分配下一天的Threshold

- 假設實際上第一天預測600筆盜刷，剩餘盜刷Quota為4400筆
- Testing第二天可分配到的盜刷量：
  - (第二天盜刷佔筆) / (剩餘總盜刷比例)
  - 0.1556 / (0.1556+0.0622+0.1556+0.1245+0.191+0.2072) ≒ 0.1736 = 17.36%

Dynamic Threshold 實驗結果 (Recall@5000)

	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
直接Testing	0.385	0.386	0.510	0.480	0.549	0.360	0.445
版本一	0.388	0.450	0.484	0.456	0.551	0.377	0.451
版本二	0.397	0.458	0.516	0.476	0.341	0.364	0.425
版本三	0.389	0.453	0.487	0.509	0.472	0.435	0.457
版本四	0.389	0.421	0.519	0.452	0.305	0.361	0.408
版本五	0.378	0.416	0.522	0.464	0.415	0.366	0.426

Dynamic Threshold 實驗結果 (預測誤差)

	Aug W3	Aug W4	Sep W1	Sep W2	Sep W3	Sep W4	Average
版本一	308	288	1266	813	49	851	595.83
版本二	93	41	193	158	28	84	99.50
版本三	549	2518	733	1477	256	2752	1380.83
版本四	101	20	264	112	5	11	85.41
版本五	53	139	94	129	104	218	122.93

04 Hybrid

Hybrid

目標:

整合Sampling, Self-supervised, Dynamic Threshold方法的盜刷偵測模型

- 解決以下問題
  - Noise Label
  - Sequence Data資料量過大
  - Sampling導致資訊量不足
  - 對應不同期資料，動態調整盜刷Threshold

Hybrid 實驗結果 (Recall@5000)

	Sep W3	Sep W4	Oct W1	Oct W2	Oct W3	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
基礎盜刷模型	0.443	0.309	0.261	0.448	0.357	0.371	0.358	0.471	0.408	0.297	0.372
Hybrid	0.497	0.298	0.267	0.446	0.398	0.354	0.381	0.390	0.381	0.265	0.368
Hybrid Dynamic Threshold	0.488	0.321	0.271	0.412	0.424	0.405	0.369	0.402	0.375	0.329	0.380

Hybrid 實驗結果 (Dynamic Threshold)預測誤差

	Sep W3	Sep W4	Oct W1	Oct W2	Oct W3	Oct W4	Nov W1	Nov W2	Nov W3	Nov W4	Average
Hybrid	91	136	3	13	59	114	172	103	273	207	117.1

結論

- 提出Hybrid盜刷模型，Recall@5000達到預期成效
- 針對盜刷偵測2.0模型需要解決的問題提出解決方案
  - **Sampling**：Negative Sampling方法找出最有效的訓練資料
  - **Noise Label**：篩出影響訓練的交易資料，並加以處理
  - **Dynamic Threshold**：結合盜刷統計量與實際預測情形，有效分配有限的資源

感謝各位長官及先進的聆聽
請不吝以下列方式與我聯繫

黃思皓 Szu-Hao Huang
szuhaohuang@nycu.edu.tw
Thanks for your Listening