Final Project

(0) 摘要

推薦系統在我們的日常生活十分常見，例如 youtube 的推薦歌單，或是網購等等，但是推薦系統可能會造成個人資料的洩漏，以及讓攻擊者回推這筆資料者的擁有者是誰，所以我們希望可以設計某一種模型，使推薦出來的結果不能回推使用者的個人資料，而這個模型可以應用在健保資料等需要高度對使用者個人資料加密等情形。

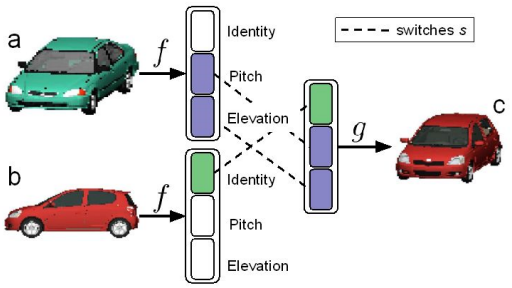
(1) 介紹

為了解決上述擁有者是誰的問題，有題提出了可以加入人為噪音或是加入 Pooling 層來使輸出的結果模糊，進而使得攻擊者不能回推資料的擁有者，但是擁有者個資的問題比較少有人討論。

針對以上個資的問題一開始我們打算使用特徵分解(如下示意圖)的方法讓我們輸出的特徵分成兩個功能讓其中一部分預測個人資料的部分的結果是差的，而去預測推薦的部分是準的。但是我們這樣的結構並不能達到我們的目的，其中有兩個難點第一個部分是如何讓結果是差的，但其實不是刻意的。因為如果只是讓結果準確下降，會使模型故意去挑選錯誤的變數。而不是作出一個好的攻擊者。

其二如果是要使結果變差而去作特徵分解，這樣做是沒有意義的，因為這樣其實就跟在分解的那一層前面在加一層 embedding 是一樣的，雖然如此我還是作出了一個特徵分解的版本，但是這個版本只能讓兩個任務都做好。

Learning a disentangled representation



$$\mathcal{L}_{dis} = \sum_{a,b,c,s \in \mathcal{D}} ||c - g(s \cdot f(a) + (1-s) \cdot f(b))||_2^2$$

為了解決上述問題我們參考了以下這篇的論文，最後嘗試加以實作裡面所提出來得 RAP 模型。一種加入對抗式網路來訓練可以保護個人資料的模型。

Privacy-Aware Recommendation with Private-Attribute Protection using Adversarial Learning

Ghazaleh Beigi, Ahmadreza Mosallanezhad, Ruocheng Guo, Hamidreza Alvari, Alexander Nou, Huan Liu
Computer Science and Engineering, Arizona State University, Tempe, Arizona
{gbeigi,amosalla,rguo12,halvari,asnou,huan.liu}@asu.edu

(2) 使用符號

I : $\{i_1, ..., i_M\}$ 所有物品的集合

U : $\{u_1, ..., u_N\}$ 所有使用者的集合

I_h : 使用者 u_h 評過分的物品集合

R_h :推薦給這個使用者 u_h 的物品集合

P : $\{p_1, ..., p_T\}$ 所有個人資料的集合 e.g. :年齡、工作、性別

R : 使用者評分的矩陣

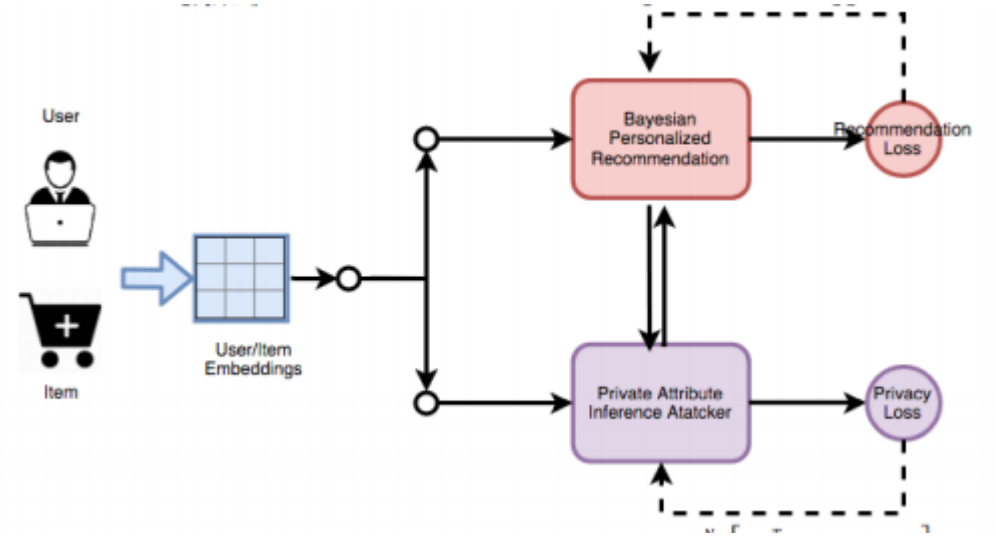
S_h : $\{I_h \cup R_h\}$

(3) 問題描述

給定 (I_h、R、P) 找出一個涵式 f 輸出 R_h ，使得 R_h 同時可以保護 P 且使得 u_h 這個使用者對 R_h 是有興趣的。

(4) RAP 模型介紹

RAP 的模型結構圖如下

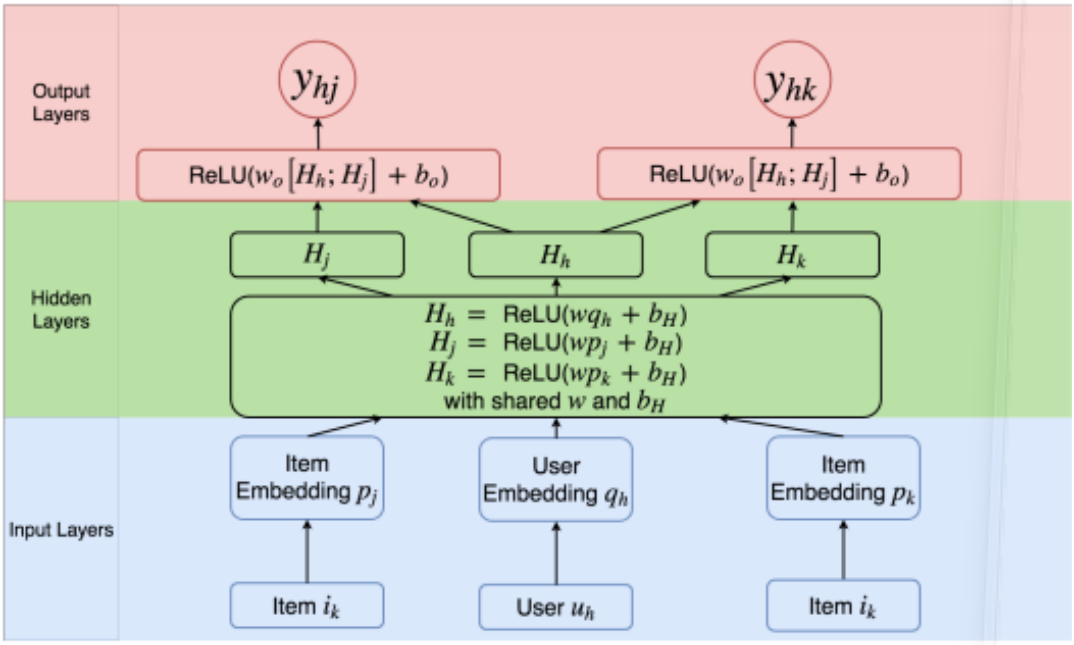


他其實可以分為三個部分，一開始的 Embedding 層，以及在圖的右上方的推薦系統，右下方的攻擊系統的部分，模型特別之處在於加入對抗式網路讓推薦系統跟攻擊系統。

首先是 Embedding 層，因為輸入的使用者向量以及物品向量都是類別向量，所以加入 Embedding 層讓他投影到歐式空間。之後將 Embedding 後的向量輸入到推薦系統。

在這邊論文當中使用的推薦系統模型是 BPR，當然在整個模型架構下想用任何的推薦系統模型都可以。

BPR 的模型如下



首先輸入一個使用者 Embedding 過的向量，一個未評分過物品 Embedding 過後的向量，以及一個已經評分過的物品 Embedding 過後的向量。經過一個相同的 layer，最後會得出三個向量然後將使用者向量跟物品向量作交互(連接、相加)，在經過一層相同的 layer.使用 ReLU 作為 activiation.最後比較出來的兩個值跟真實情況的情形。綜合上述這個部分的 loss function 可以寫為：

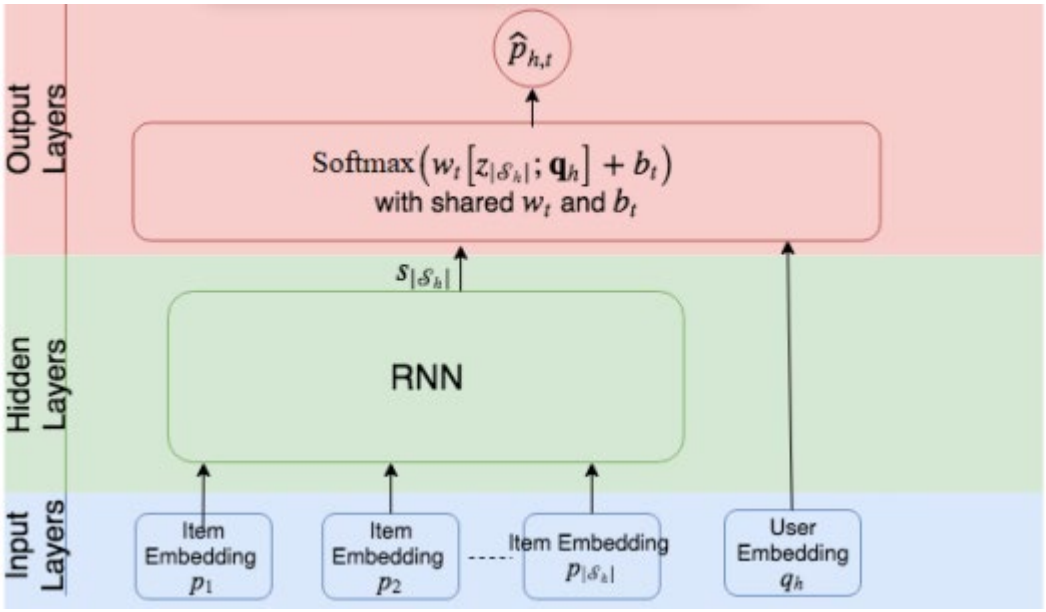
$$\mathcal{L}_{D_R} = \frac{1}{N}\sum_{h=1}^{N}\sum_{(h,j,k)\in\mathcal{D}_h} -\ln\delta((\hat{y}_{hj}(\theta_R)-\hat{y}_{hk}(\theta_R)).g(h,j,k))+\lambda_{\theta_R}\|\theta_R\|^2 \tag{4}$$

where, $g(h,j,k)$ is the ground truth value for our model training:

$$g(h,j,k) = \begin{cases} 1, & \text{if user } u_h \text{ prefers item } i_j \text{ over item } i_k \\ -1, & \text{otherwise} \end{cases} \tag{5}$$

其中最後一項是 regularization.D_h 表示所有 user、此 user 有評分過的樣本、此 user 沒有評分過的樣本所形成的集合。
而這個模型對沒有評分過的樣本預測，是將兩個輸入的向量都放入未評分過樣本的向量。取最後出來的平均，以這個為排序，排序前 K 個物品即為推薦的物品。
最後一個部分為攻擊系統即利用使用者的使用紀錄等去得知使用者的個人資料，這篇的論文當中使用的是 RNN 模型作為攻擊系統的主架構，我覺得值得一提的有兩點：

第一點是在攻擊的時候不僅考慮使用者的過往紀錄，同時還考慮使用者的向量、第二點是這篇論文認為，除了已經有使用的過往紀錄外還必須將推薦的物品向量也放入裡面。理由是攻擊者也可能從過往紀錄推得這部分的資訊，所有這個部分也應該要包含在推論使用者個人資料的模型裡面。當然在整個模型架構下想要使用任何的攻擊方式都是可以的，綜合以上模型大約如下：



這個部分的 loss function 使用簡單的 cross entropy 可以寫為如下：

$$\mathcal{L}_{D_P} = \frac{1}{N}\sum_{h=1}^{N}\left[\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{D_P^t}(\hat{p}_{h,t}, p_{h,t})\right]$$

最後將兩部分的 loss function 寫在一起，形成對抗式網路。
可以寫成：

$$\underbrace{\min_{\theta_R}\left(\mathcal{L}_{D_R} \overbrace{-\alpha \max_{\{\theta_P^t\}_{t=1}^{T}} \mathcal{L}_{D_P}}^{\text{private-attribute attacker}}\right)}_{\text{privacy-aware recommendation system}}$$

$$\min_{\theta_R}\max_{\{\theta_P^t\}_{t=1}^{T}}\left(\frac{1}{N}\sum_{h=1}^{N}\left[\sum_{(h,j,k)\in\mathcal{D}_h} -\ln\delta((\hat{y}_{hj}(\theta_R) - \hat{y}_{hk}(\theta_R)).g(h,j,k))\right.\right. \tag{9}$$

$$\left.\left. -\alpha\left[\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{D_P^t}(\hat{p}_{h,t}, p_{h,t})\right]\right] + \lambda\Omega(\theta)\right)$$

其中 alpha 控制攻擊系統的部分的 loss 的權重 。演算法的部分可以寫成。

Algorithm 1 The Learning Process of RAP model

Input: Items set $\mathcal{I}$, training user data $\mathcal{U}$, training user-item matrix data $\mathbf{R}$, batch size b, θ_R, $\{\theta_P^t\}_{t=1}^T$, α, λ and K.

Output: Trained recommendation with protection RAP.

1: **repeat**
2: Create a mini-batch $\mathcal{U}_b$ of b users with their private-attribute and item-rating information from $\mathcal{U}$
3: Train the recommendation with attribute protection via Eq. 10 w.r.t. θ_R
4: For each user h in $\mathcal{U}_b$, calculate the top-K recommended items $\mathcal{R}_h$
5: Train the private-attribute inference attacker D_P (i.e., $\{\theta_P^t\}_{t=1}^T$) via Eq. 7 given the users' information including their list of items information, i.e., $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$
6: **until** Convergence

(5) 資料前處理及與參考論文中不同的地方比較

我們使用與這篇論文同樣的資料集 ml-100k，資料形式如下。

	user	item	rating	timestamp
0	0	167	5	874965478
1	0	171	5	874965478
2	0	164	5	874965518
3	0	155	4	874965556
4	0	195	5	874965677

對於每個使用者的時間後 20%時間資料，我們將他當成 ground_truth 部分，這個部分這篇論文是直接使用隨機的，但我們覺得評分之間的關係是有時間性的所以採用時間作為切割。

使用者個資如下：

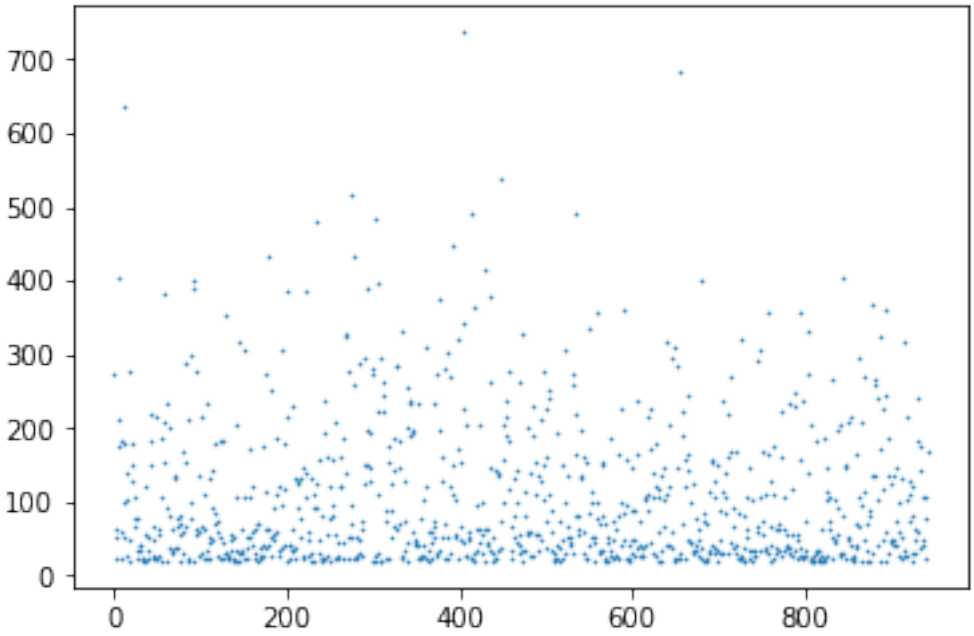
	Age	Gender	Job
0	24	1	19
1	53	0	13
2	23	1	20
3	24	1	19
4	33	0	13

使用者評分矩陣如下：

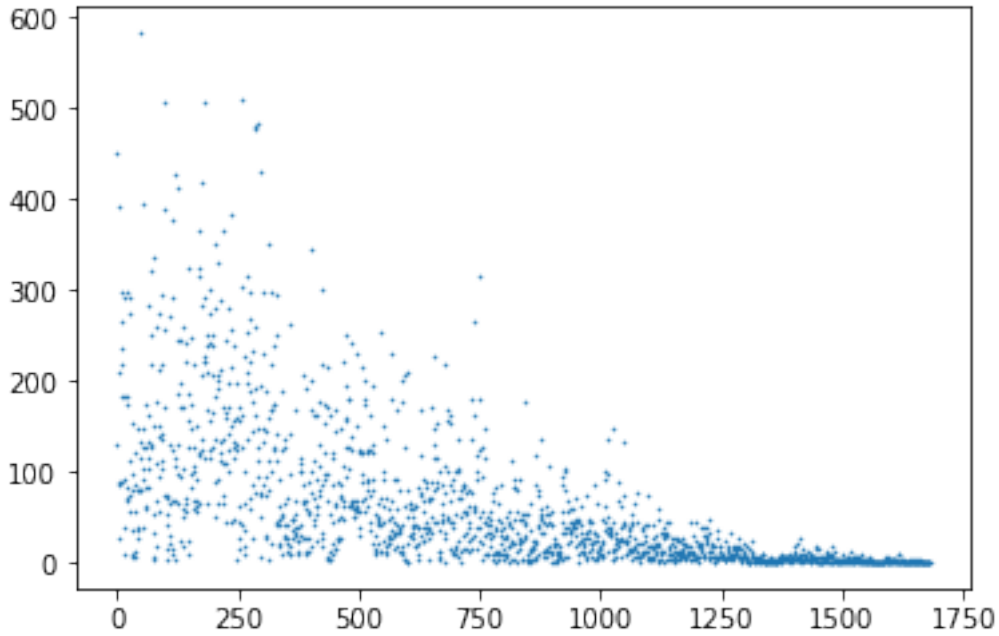
```
array([[5, 3, 0, ..., 0, 0, 0],
       [4, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 5, 0, ..., 0, 0, 0]],
```

因為最後使用 cross_entropy 的關係我們將 Age 分成
(0,25)、 (25,50) 、 (50,inf) 來做預測論文中則是
(0,35)、 (35,45) 、 (45,inf)
因為模型運行時間較久，我們並沒有作超參的優化但在論文中經過超參數優化將第一層的 Embedding layer 設為 70 、中間的隱藏層設為 20、lambda 設為 0.01。
而 RNN 的部分則是輸入維度為 70 。並沒有提到這個部分的 batch_size 設定為 32。最後使用 Adam 優化器。
而這個實驗需要檢驗的部分有三：
分別是
(1)隱私性：這個模型是否保護了使用者的個人資料
(2)有效性：這個模型是否推薦了使用者有興趣的物品
(3)兩者之間的關係：兩者之間是否有消長的情形

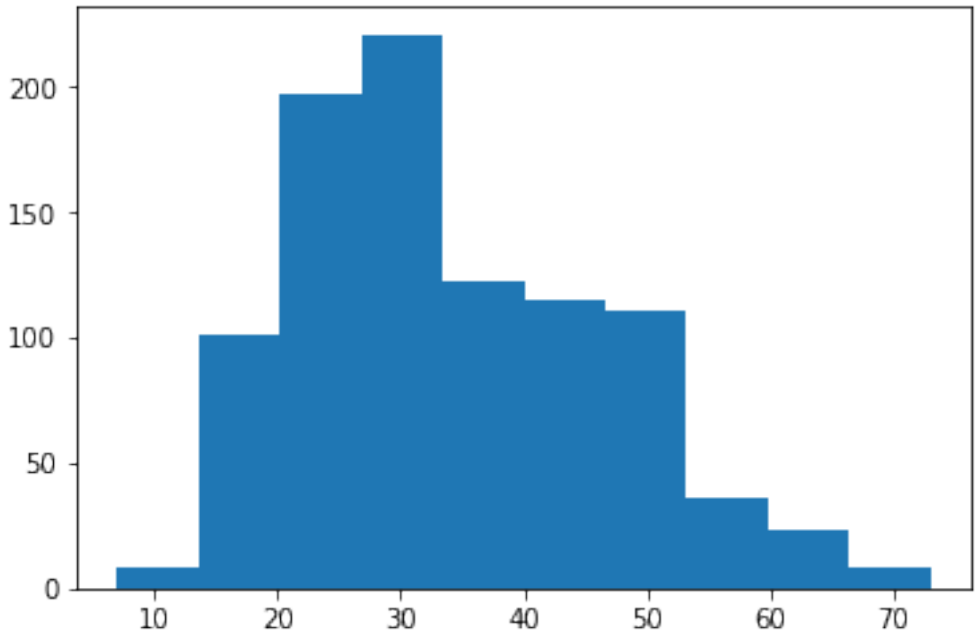
(6) 敘述統計量
使用者的個數：９４３
物品的個數：１６８２
評分的比率的分佈圖：
Ｘ軸：ｕｓｅｒ
Ｙ軸：有評分的個數



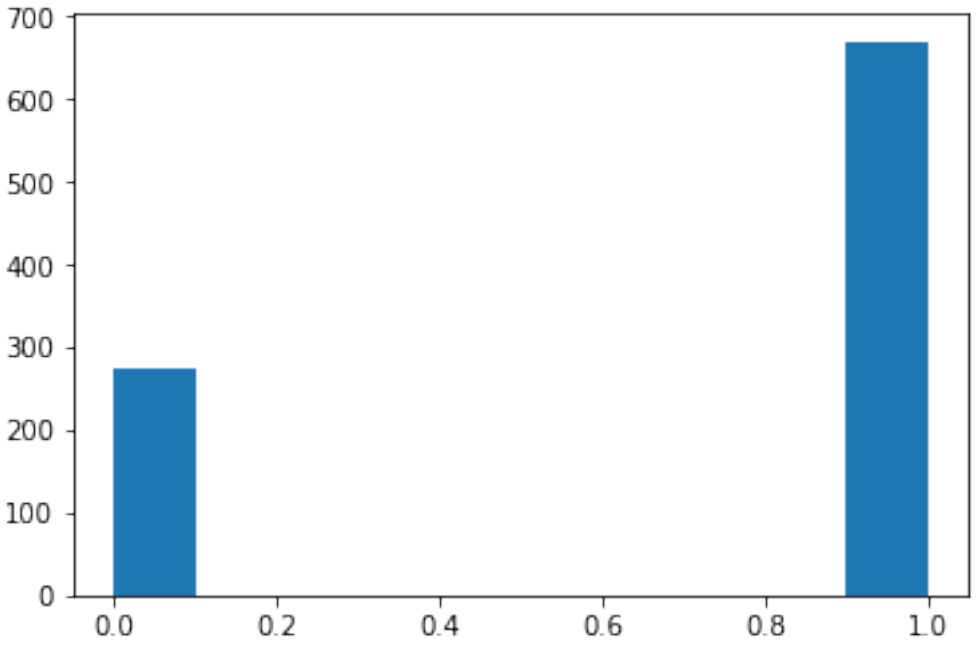
X軸：ｉｔｅｍ
Y軸：有評分的個數



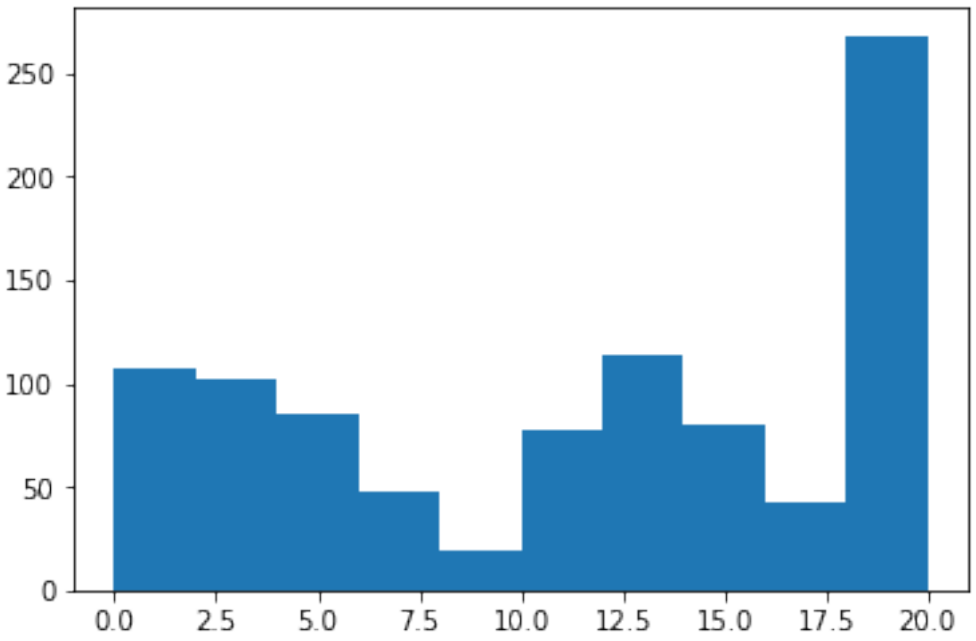
可以看出ｕｓｅｒ的分佈比較平均，ｉｔｅｍ的分佈比較離散，有些被很多人憑分過，但大部分都集中在０附近。平均的評分也成現差不多的情形
年齡長條圖



性別長條圖



工作長條分佈圖



(7) 結果

針對隱私性的部分，這篇論文使用 AUC_CURVE 來作為評斷，因為是希望可以保有隱私所有 AUC_CURVE 是越低越好。

而有效性的部分有兩個評斷分別是 P@K 以及 R@K 公式如下：

$$P@K = \frac{|\{\text{test items}\} \cap \{\text{top-}K \text{ returned items}\}|}{K}$$

$$R@K = \frac{|\{\text{test items}\} \cap \{\text{top-}K \text{ returned items}\}|}{|\{\text{test items}\}|}$$

論文中的結果如下：

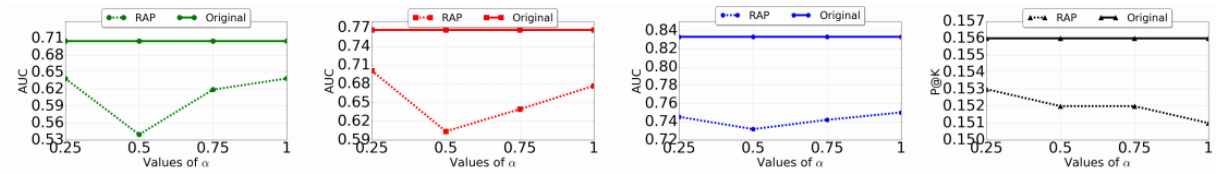
Model	# test items (l)														
	35					40					45				
	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$
ORIGINAL	0.7662	0.7050	0.8332	0.156	0.156	0.7662	0.7050	0.8332	0.151	0.172	0.7662	0.7050	0.8332	0.145	0.187
LDP-SH	0.6587	0.6875	0.8076	0.071	0.071	0.6440	0.6777	0.7954	0.062	0.078	0.6398	0.6732	0.7817	0.055	0.081
BLURME	0.6266	0.6177	0.7614	0.118	0.118	0.6013	0.5949	0.7589	0.109	0.134	0.5884	0.5901	0.7522	0.099	0.150
RAP	**0.6039**	**0.5397**	**0.7319**	**0.152**	**0.152**	**0.5714**	**0.5270**	**0.7315**	**0.147**	**0.168**	**0.5278**	**0.5262**	**0.7312**	**0.142**	**0.183**

比較中 original 即不加入對抗式網路，LDP-SH 是加入噪音的做法，可以看出來 RAP 雖然降低了一點推薦有效性，但是卻提升了很高的隱私保護。而隨著 test-item 的數量上升情況也有上升。

而針對對哪個個人資料最有保護這篇論文也有作實驗，這個部份我們並沒有時間進行。結果如下：

Model	# test items (l)														
	35					40					45				
	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$
RAP	0.6039	**0.5397**	**0.7319**	**0.152**	**0.152**	0.5714	**0.5270**	**0.7315**	**0.147**	**0.168**	0.5278	**0.5262**	**0.7312**	**0.142**	**0.183**
RAPAGE	0.6450	0.5948	0.7528	0.150	0.150	0.5489	0.5938	0.7522	0.146	0.167	0.5475	0.5909	0.7497	0.141	0.182
RAPGEN	**0.5332**	0.6789	0.7558	0.151	0.151	**0.5298**	0.6614	0.7556	0.145	0.166	**0.5211**	0.6415	0.7555	0.141	0.181
RAPOCC	0.6571	0.6949	0.7468	0.147	0.147	0.6485	0.6871	0.7466	0.141	0.161	0.6454	0.6853	0.7438	0.135	0.174

Table 2: Impact of different private-attribute attacker components on RAP in terms of utility and privacy.

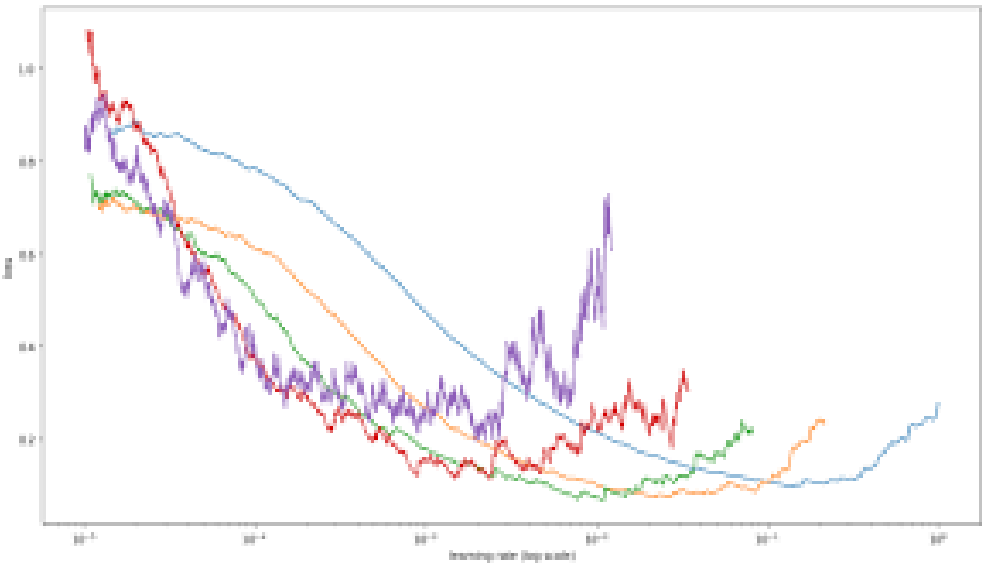


(a) Attribute Age **(b) Attribute Gender** **(c) Attribute Occupation** **(d) Recommendation**

可以看得出來針對性別的保護是最好的，我們個人的猜測是因為性別是一種二元變數所以可以保護得比較好，但因為我們這部分連模型都沒辦法跑出理想結果所以沒有比較。

回到我們自己的實驗，由於我們在傳送 sess 的時候是一筆一筆傳進去的所以我們的 batch_size 就是 1，結果我們最後的結果沒有收斂，我們的推薦有呈現漸漸準確的跡象但是在我們迭代 100 次之後準確率 P@K 跟 R@K 只有 0.05 左右跟論文中的實驗結果差了 3 倍，我們推測可能性有如下：

一些超參的基本設定就不同，例如層數、alpha。其中最重要的我覺得是 batch size 的設定。因為我們 batch_size 設定太小會使他收斂的很慢所以 100 次迭代可能是不夠的



如上圖呈現下降但是不穩定，需要更多的迭代。

至於隱私保護性的部分結果與論文類似但是我們並沒有使用 AUC_CURVE 而是直接使用準確性。準確性的確比什麼都不加的時候有所下降但是因為時間關係我們的結果無法作出比較圖

(8) 結論及建議

明顯的針對這些目標這個模型有作到我們想作的事，我覺得整理的模型架構是沒有什麼問題的，但是我覺得像這樣的資料集比較少，其實除了這個ｍｌ１００Ｋ之外想要找到像這樣的資料集並不容易。不過這個模型的應用很廣我覺得可以在這個模型之下加入社群相關的資訊也許可以做到更好的保護也許可以直接堵絕有可能洩漏個資的社群關係等等。

(9) 自我評論

我覺得這份報告對我來說是十分有意義的，一來我嘗試用ｔｅｎｓｏｒｆｌｏｗ這個套件去手刻一些比較難的模型，其中ｇａｎ模型在這學期一開始的教學中就有講到，但事實際作起來的設計那些卻沒有想像中容易，然後我覺得ＧＡＮ真得很容易就壞掉，可能還需要多一點再深度學習的ｓｅｎｓｅ才能把這些作得更好。例如像我的ＢＡＴＣＨ就設計得很爛，才會讓我們的結果如此的奇怪。

(10) 組員貢獻

陳彥宏：Ｃｏｄｉｎｇ、論文解讀

吳岱傑：題目探索

王耀德：書面