國 立 中 正 大 學

資訊工程研究所碩士論文

基於深度學習之美食推薦平台設計與實作

The Design and Implementation of Cuisine Recommendation Platform Based on Deep Learning

研 究 生：簡明旋

指導教授：吳　昇 博士

中華民國 107 年 7 月

國立中正大學碩士學位論文考試審定書

資訊工程學系

研究生 簡明旋 所提之論文

基於深度學習之美食推薦平台設計與實作
經本委員會審查，符合 碩 士學位論文標準。

學位考試委員會
召集人 _____ 簽章

委員

指導教授_____ 簽章

中華民國 109 年 7 月 28 日

致謝

在中正大學研究所求學期間，非常感謝吳昇教授在這段時間的關心及悉心指導，也謝謝老師給予我們許多空間及自由，讓我們能盡其所長，而在我遇到困難時，老師也會給予許多幫助及建議，使我於研究所期間受益匪淺。

在 GAIS Lab 這段期間，非常感謝學長姐的照顧及幫忙，謝謝實驗室的夥伴們的互相陪伴，我們一起玩樂、出遊、煮飯度過了許多歡樂時光，也謝謝實驗室的學弟妹們一起分擔事務，讓我能專注於論文上。

在論文期間，感謝邵為、昱傑、思璇、翊荃、祥慶、家羽、柏賢、佳欣、怡靜、亦淳、耀德、宗穎、智皓、睿宏及所有支持和幫助我的人，因為你們，才能讓我的論文更順利，研究生活更精彩充實。

最後，僅將此論文獻給最關心和支持我的家人及朋友，我願與你們分享這份喜悅及榮耀，謝謝你們。

基於深度學習之美食推薦平台設計與實作

研究生：簡明旋　　指導教授：吳昇 博士

國 立 中 正 大 學 資 訊 工 程 研 究 所

摘要

隨著科技發展，現今社群網路充斥著生活，人們於社群網路上互相分享資訊造就社群網路扮演著大量資訊提供者的角色，同時也間接形成資訊過載的問題，導致人們取得感興趣資訊的成本提高，因此推薦系統也成為了網路平台重要的發展趨勢。

本論文實作一個美食推薦平台，平台擁有社群概念並結合推薦服務，使用者可透過此平台記錄美食食記、搜尋餐廳及收藏食記等。平台會依照內文進行分類從而得知使用者偏好類別，並結合圖片辨識提高分類準確性。除此之外，對於推薦系統的建構，除使用傳統協同過濾（Collaborative Filtering），也結合基於會話行為訓練的模型進行時間序列預測，來解決協同過濾的冷啟動（Cold-start）問題。

在整個系統中運用了深度學習（Deep Learning）的技術，來更準確的推薦使用者有可能感興趣的美食店家。根據實作出來的美食平台實測結果顯示，透過結合不同推薦方法可以有效提升點擊率。

關鍵字：推薦系統、協同過濾、深度學習

The Design and Implementation of Cuisine Recommendation Platform Based on Deep Learning

Student：Ming-Hsuan Chien Advisor：Dr. Sun Wu

Department of Computer Science and Information Engineering

of National Chung Cheng University

ABSTRACT

With the development of technology, people share information on the social network, which makes it a massive information provider. However, the problem of information overload occurs simultaneously and makes it hard for people to acquire the information they are interested in. Therefore, the recommendation system has become a major trend on the Internet.

This paper describes the implementation of the cuisine recommendation platform with social network concept. Users can search restaurants, post and keep other people's dining briefs on this platform. Subsequently, we can obtain the categories those users prefer by classifying the articles and enhance the accuracy of classification through image recognition technology. Beside using traditional collaborative filtering to construct the recommendation system, we employ the training model of user behaviors based on the session to forecast the time-series, which solves the cold-start problem of collaborative filtering.

This system applies deep learning to precisely recommend users the restaurants in which they have interests. The experimental results show that the combination of different recommendation methods can effectively improve click-through rate.

Keyword: Recommendation System, Collaborative Filtering, Deep Learning

目錄

圖目錄

表目錄

第一章 緒論

隨著科技與網際網路的發展，如今的人類社會已是一個高度資訊化的時代，多數人的生活皆與社群軟體有著密切關係，網際網路上的資訊與日俱增，使得在網路中尋找有用的資訊成本也隨之提高，也因此產生了所謂「資訊過載（Information Overload）」的問題。

為了解決資訊過載的問題，「搜尋引擎」及「推薦系統」皆是解決此問題的技術。使用者在網路上尋找資訊時，提供關鍵字透過搜尋引擎來找出最相關之結果，例如我們使用搜尋引擎找美食，關鍵字為「美食」，搜尋引擎就會列出與美食相關的結果，倘若使用者提供關鍵字資訊不足，例如要找嘉義的美食但關鍵字並沒有提及嘉義，搜尋引擎便無法提供精準的結果；推薦系統與搜尋引擎不同，推薦系統可以以使用者為基礎，透過分析使用者的行為，為此計算出特徵進行建模，從而主動推薦給使用者感興趣的資訊，例如我們知道該位使用者偏好中式料理，我們則可以主動推薦中式料理的餐廳。對於使用者來說搜尋引擎由使用者主動搜尋，推薦系統則為被動的接受系統推薦，兩者之間也能起到互補的作用。

本論文實作一推薦系統，使用數據為使用者撰寫的美食食記、瀏覽餐廳店家和美食喜好等資訊來進行美食推薦，透過協同過濾及結合深度學習的圖片辨識、基於會話（Session-based）的混合推薦結果來提供使用者感興趣的美食、餐廳店家。

本論文的章節編排如下：第二章將探討論文中使用到的核心技術之背景知識；第三章系統架構與設計說明平台的架構及核心功能設計；第四章說明系統的設計與實作方式主要針對推薦部分進行探討；第五章為系統呈現；第六章為實驗結果，包含模型評估及針對前端蒐集使用者點擊推薦結果，最後分析結果並探討是否能有效的提升使用者點擊推薦項目的意願；第七章為結論；最後為本論文的參考文獻。

第二章 背景知識

2.1 協同過濾（Collaborative Filtering）

2.1.1 簡介

協同過濾（Collaborative Filtering）最早於 1992 年應用在 Xerox 公司作為個性化郵件系統，1994 年 GroupLens 將協同過濾應用於新聞篩選，也是協同過濾重要的里程碑，自此協同過濾迅速於推薦系統中大放異彩。協同過濾著重於使用者（Users）及評分（Rating），以此推測興趣相投的使用者所感興趣的資訊，其主要推薦演算法可分為兩大類，分別為以模型為基礎（Model-based）及以記憶為基礎（Memory-based）的協同過濾。

2.1.2 以模型為基礎（Model-based）

以模型為基礎的協同過濾（Model-based Collaborative Filtering）的核心演算為先將使用者及評分進行處理再訓練建模，常見的方法為矩陣分解（Matrix Factorization），評分矩陣透過矩陣分解成兩個矩陣相乘，這個目的就是為了學習到使用者的隱向量矩陣 U 及項目的隱向量矩陣 I，我們也可以透過深度學習來學習這些隱向量。

2.1.3 以記憶為基礎（Memory-based）

以記憶為基礎的協同過濾（Memory-based Collaborative Filtering）又細分為基於項目的協同過濾（Item-based Collaborative Filtering）及基於使用者的協同過濾（User-based Collaborative Filtering），前者依照使用者歷史行為將曾喜歡的項目與其他項目計算相似性，舉例來說得知使用者曾喜好的項目 A 與陌生的項目 B 有較高的相似度，則可以選擇給予推薦；後者依照使用者進行相似性計算，使用者 乙、丙皆喜歡項目 B ，而使用者甲未曾接觸過項目 B ，因使用者甲、乙、

丙 若皆喜歡項目 A ，我們則可透過 K-近鄰演算法（K-Nearest Neighbor，KNN）[12]，得出使用者甲、乙、丙很相似像鄰居為同一群，即可選擇給予使用者甲推薦項目 B 。

無論是基於項目或使用者都有適合使用的地方，「物以類聚，人以群分」一言也很好的解釋了這兩種協同過濾。

2.2 遞歸神經網路（Recurrent Neural Networks）

2.2.1 簡介

遞歸神經網路（Recurrent Neural Networks，RNN）[23] 或稱循環神經網路，不同於傳統神經網路，RNN 是一種神經元（Neural）上帶環的神經網路，神經元上的環可以保存著不同狀態作為下一個神經元的一部分輸入（Input）。

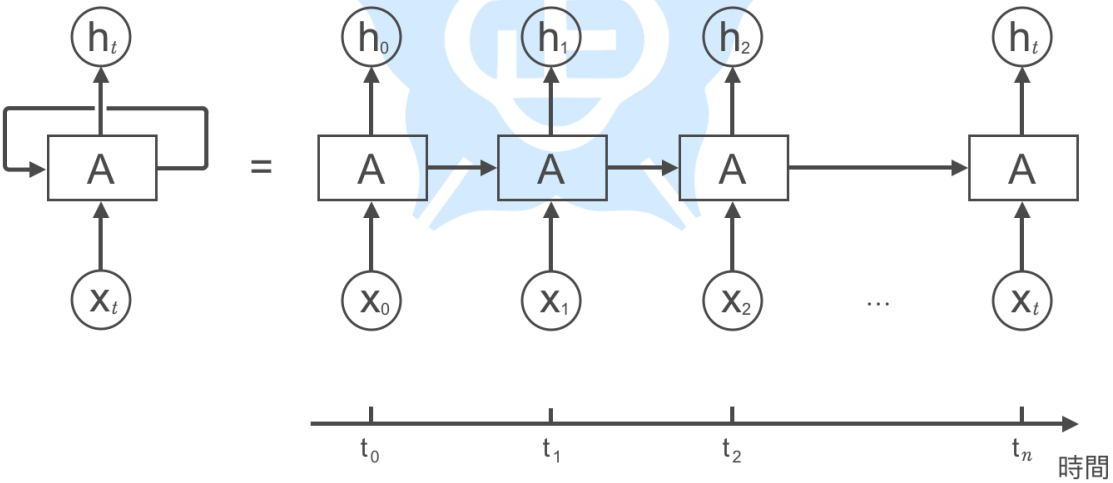

圖 2.1、遞歸神經網路

於時間序列下的不同時間點，RNN 的神經元輸入皆與先前的時間狀態有關，在 t_n 的時間點下的輸出即是該時間點的輸入和所有歷史狀態共同計算出的結果，這也就使得 RNN 可以達到對時間序列建模的目的。

2.2.2 長短期記憶（Long Short-Term Memory）

長短期記憶（Long Short-Term Memory，LSTM）[14] 是一種 RNN 的特殊類型，在 LSTM 中每個神經元皆是一個記憶細胞，LSTM 的關鍵就是細胞狀態（Cell State），每個細胞內皆包含了輸入門（Input Gate）、遺忘門（Forget Gate）、輸出門（Output Gate）三個門，根據當前的資訊以及這些門來控制細胞是否接受影響來更新這個細胞的記憶和資訊。

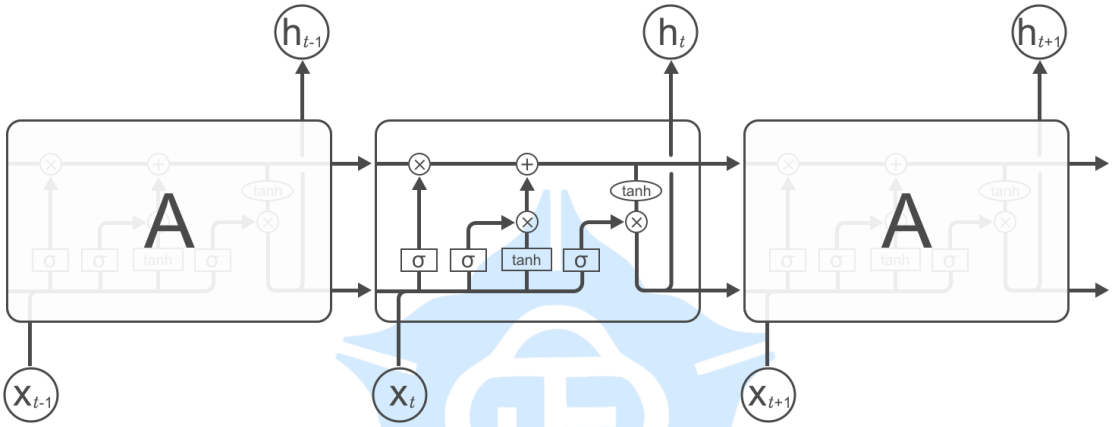

圖 2.2、LSTM 結構

LSTM 中的「遺忘門」因能夠控制訓練時梯度的收斂性，不僅可以避免傳統 RNN 中梯度消失（Gradient Vanishing）與梯度爆炸（Gradient Exploding）的問題，同時也能保持長期的記憶性。

2.2.3 門控遞歸單元（Gated Recurrent Unit）

門控遞歸單元（Gated Recurrent Unit，GRU）是 LSTM 的變種，GRU 模型只有兩個門，更新門（Update Gate）和重置門（Reset Gate），更新門為 LSTM 中的輸入門（Input Gate）和遺忘門（Forget Gate）的合併，GRU 也將細胞狀態（Cell State）和隱狀態（Hidden State）做了合併。

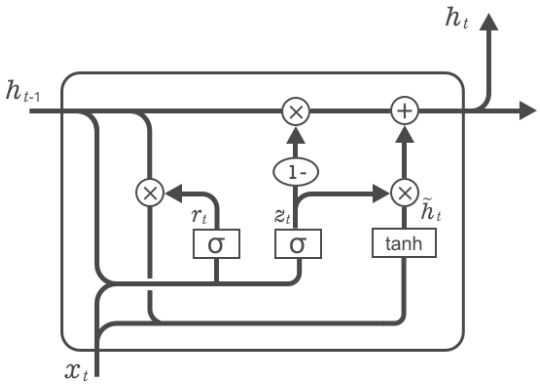

圖 2.3、GRU 結構

重置門控制是否遺忘前一時刻的記憶狀態，重置門的值越小說明忽略的越多；更新門則控制之前記憶留存的比例，更新門的值越大說明前一時刻的狀態留存越多，GRU 除了保持了 LSTM 的效果且結構更加簡單，GRU 也因此流行。

2.3 標記式隱含狄利克雷分布（Labeled Latent Dirichlet Allocation）

隱含狄利克雷分布（Latent Dirichlet Allocation，LDA）[13] 是一種主題模型（Topic Model），可將文檔集（Documents）中的每篇文檔的主題按照機率分佈的形式給出，文檔都是由詞（Words）所構成，LDA 通過分析文檔後計算文檔的主題分布，我們僅考慮詞的出現與否，而不考慮順序也屬典型的詞袋模型（Bag-of-Words Model）[6]。LDA 可用於文本分類或主題類聚，是一種非監督式學習（Unsupervised Learning）。

標記式隱含狄利克雷分布（Labeled Latent Dirichlet Allocation，Labeled-LDA）由 Daniel Ramage 等四位作者於 2009 年提出[2]，Labeled-LDA 基於 LDA ，於文檔到主題分布之間，新增了一個監督項，Labeled-LDA 透過人工標記主題的方式，使得訓練時只從文檔對應的主題中採樣，從而得到最後的主題分佈，是一種監督式學習（Supervised Learning）。

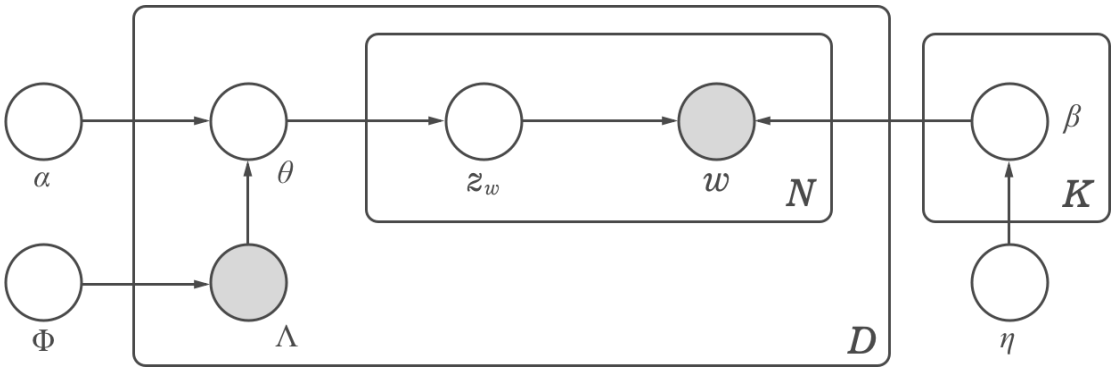

圖 2.4、Labeled-LDA 模型

第三章 系統架構與設計

3.1 系統架構

本系統架構為前端 UI 和後端 API 及搭配機器學習實作推薦系統，使用者透過前端可以進行食記撰寫、店家搜尋等操作，而使用者撰寫美食食記及瀏覽店家等行為皆會回傳於後端系統進行分析，透過後端實作推薦演算法來推薦使用者感興趣的美食店家。

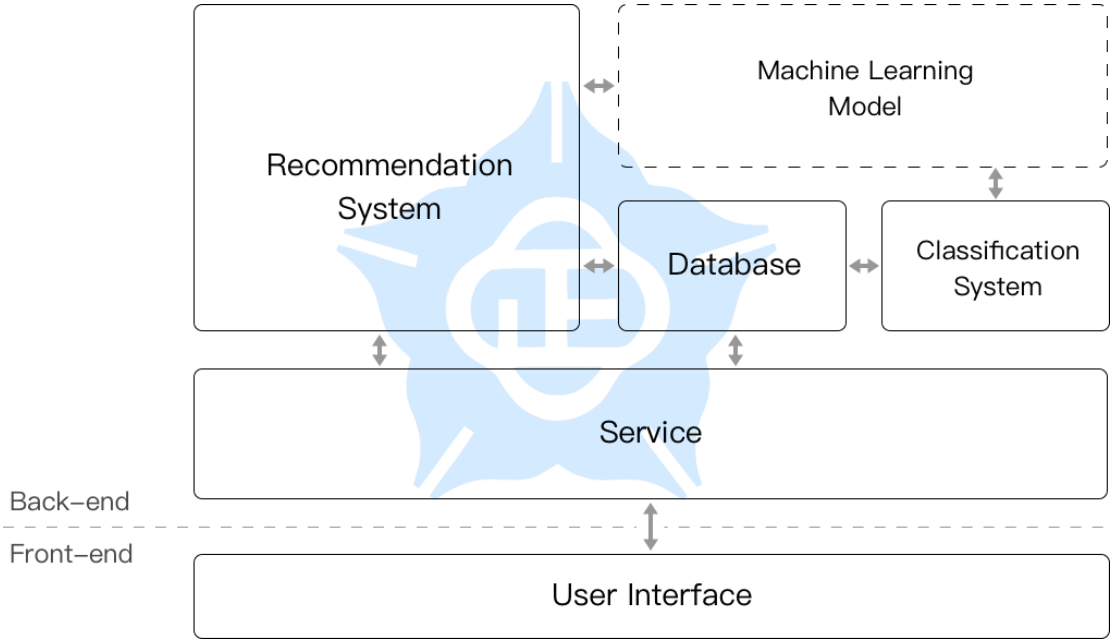

圖 3.1、美食推薦平台架構圖

3.1.1 Service

為整個美食平台的服務入口，串接著資料庫及推薦系統為系統核心之一，處理並提供使用者可以在前端 Web 或行動裝置 App 上取得所需資料，前端提供的發布貼文、搜尋、推薦等功能皆透過後端提供之 RESTful API[24] 進行資料傳輸。

3.1.2 Database

我們使用 PostgreSQL[20] 保存平台上使用者、美食、店家等資料，而使用者行為數據保存於 MongoDB[16]，這些資料將提供平台服務或推薦系統欲使用時取其相關數據資料。

3.1.3 Recommendation System

推薦系統為整個平台上核心之一，主要為使用者推薦可能感興趣的店家美食，主要使用協同過濾並結合深度學習來進行混合推薦，其目的為提高使用者可能感興趣項目點擊率。

3.1.4 Machine Learning Model

包含分類系統及推薦系統所使用的機器學習演算法，所建立的模型將被保存下來，推薦系統包含協同過濾及遞歸神經網路，當需要分類或使用者需要推薦時，系統可透過已保存的模型進行預測（Predict）。

3.1.5 User Interface

串接後端 API 提供完整的使用介面，如前端 Web 或行動裝置 App，前端介面提供使用者可以進行美食照片上傳、評論評分、文章撰寫、餐廳搜尋等功能操作。

3.2 系統設計

系統於前端 UI 提供使用者進行操作，並設計會員系統用來區分使用者，使用者進行相關操作時會回傳給後端，後端需設計一服務進行平台上的相關操作及設計推薦系統來進行使用者數據分析從而達成推薦目的。以下列出服務為系統核心功能。

3.2.1 註冊及登入

為了區分不同使用者及保存相關食記、行為等資訊，需要實作一會員系統，使用者登入系統後，方能使用更完整功能，亦可針對使用者進行後續的推薦動作。

3.2.2 美食食記

使用者可以進行美食食記照片的上傳、編輯及發布，這些食記會於後端進行分類處理，也會成為使用者的特徵之一，作為推薦系統的參考數據。

3.2.3 行為蒐集

使用者瀏覽食記、搜尋美食和收藏店家時的行為，皆會保存於後端資料庫，作為推薦系統的參考數據。

3.2.4 搜尋推薦

使用者搜尋店家美食時，除透過關鍵字或類別篩選，其結果也包含了針對使用者的推薦，這些推薦目的為讓使用者更快搜尋到感興趣的店家，抑或是單純提供附近的推薦美食。

3.2.5 推薦系統

使用者資料經由前處理後，應用協同過濾演算法進行推薦，並針對使用者的行為蒐集每次的會話（Session）數據，我們利用這些時間序列數據並透過深度學習進行遞歸神經網路訓練，使得協同過濾下使用者資訊過少無法獲得較佳結果時，可以透過會話數據來進行基於會話行為的推薦。

第四章 系統實作

4.1 主要服務

本論文以 Node.js[18] 實作平台上主要服務，透過與資料庫溝通交互所需資料，並建立 API 供前端使用，除了基本功能之外，主要核心為推薦功能，推薦實作細節會於此章節進行探討。

4.1.1 註冊及登入

使用者使用平台服務時需先進行註冊登入動作，註冊及登入服務為平台實作用於區分使用者及保存管理使用者相關數據。使用者於前端進行登入動作，透過 API 於後端驗證，若確認無誤則核發一組 token ，前端會保存這組 token 進行後續請求驗證，登入後即可使用平台上完整的服務。

4.1.2 美食食記

提供類似於社群軟體上的圖片貼文功能，使用者可於平台上撰寫美食食記，步驟如下：

1. 選取欲上傳的美食照片
2. 選擇打卡地標以添加店家資訊
3. 食記評分，分數為 0~5 分
4. 主文撰寫完畢後即可發布食記

4.1.3 行為蒐集

使用者操作前端系統時，會將特定行為記錄透過 API 傳至後端，前端的行為蒐集包括瀏覽食記、搜尋美食、瀏覽及收藏店家，後端可以針對使用者行為記錄分析出偏好的特定美食類別及店家等，這些行為數據也會應用於推薦系統。

4.1.4 搜尋服務

平台上已預先蒐集臺灣約 3 萬筆店家資訊，包含店名、地址、電話及 GPS 經緯度座標等資訊，使用者可以透過關鍵字的方式來尋找自己感興趣的店家，並提供篩選類別、距離及評價的功能，搜尋的頁面除了顯示搜尋的結果，另會針對不同的使用者提供推薦的店家。

4.2 食記分類

使用者所撰寫的食記保存於後端之後，會經過平台上的分類系統進行文章分類的動作，並給予分類標記，我們訂定了美食分類有吃到飽、中式、日式、鍋物和早午餐等共 23 種類別，並使用標記式隱含狄利克雷分布（Labeled-LDA）將食記的主文作為文章輸入進行分類，並將分類結果保存至資料庫中。考慮到部分使用者僅會上傳照片，撰寫的主文過短導致分類結果不精準，甚至主文為空白導致無法分類，因此另實作基於深度學習的圖片辨識來輔助分類結果的判斷。

4.2.1 標記式隱含狄利克雷分布

我們使用 Python[21] 實作 Labeled-LDA ，我們訂定了共 23 種食記的類別如表 4.1，使用的訓練資料為 1 萬筆已完成人工標記類別的美食文章，並將此模型結果保存下來用於新文章分類預測。

表 4.1、美食食記分類

吃到飽	中式	西式	日式	韓式	港式
美式	泰式	義式	法式	南洋料理	鍋物
燒肉	素食	早點	早午餐	餐酒館	小吃
飲料	咖啡	甜點	複合式餐廳	酒吧	

4.2.2 卷積神經網絡

美食食記在分類時總有預測錯誤的時候，然而部分的使用者在撰寫食記時主文內容過短甚至留空，僅有美食照片及評分，這些稀少內容資訊無法有效的利用 Labeled-LDA 的方式來進行分類預測，既然如此我們將使用者所上傳的圖片透過圖片辨識來做分類的預測，將圖片辨識出來的結果用來輔助以文章內容分類時內文過短的不足，來提升整體分類的準確率。

卷積神經網絡（Convolutional Neural Network，CNN）[8] 是一種多層的神經網路架構，透過深度學習的方式建立圖片辨識模型來達到美食分類的效果。CNN 的實作採用 TensorFlow[27] 搭配 Keras[11]，使用的模型為 Inception V3[10]，我們預先蒐集了特定美食圖片，其中包含 50 個食物類別及 3 個非食物類別人物、環境和菜單，每個類別約 1,000 張，共約 5 萬張圖片，並於 GCP（Google Cloud Platform）上租用虛擬機器進行訓練，訓練時使用 CUDA[9] 搭配 NVIDIA Tesla P100 GPU 進行運算加速，加快訓練時間，模型評估準確率約為 88.46%。

4.3 協同過濾

我們使用 Python 實作協同過濾（Collaborative Filtering），經測試達到了不錯的效果，後續的優化我們透過正規化及依照時間調整使用者給予店家的評分（Rating）的權重及交替最小二乘法（Alternating Least Squares）來達到更好的推薦效果。

4.3.1 基於使用者的協同過濾

將使用者所發布過的店家食記評論作為基於使用者的協同過濾（User-based Collaborative Filtering）輸入，食記中的店家為項目（Item）及食記的評分（Rating）作為協同過濾所需的矩陣數據，有了這些資訊後，我們可以尋找與使用者相似的

其他使用者，這個步驟也稱為最鄰近搜索（Nearest Neighbor Search, NNS）[17]，有了鄰近集合我們就可以針對使用者進行預測，推薦使用者可能感興趣的店家。

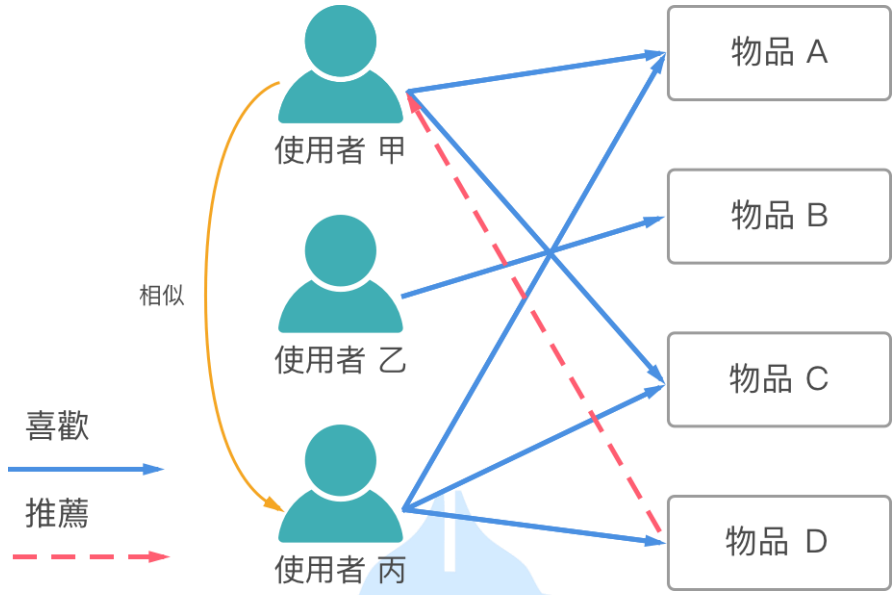

圖 4.1、基於使用者的協同過濾概念

4.3.2 評分正規化

使用者給予的評分並非遵循同一標準，每位使用者皆有自己的評分習慣，例如有些使用者評分時通常給予很高的分數，有些使用者則反之，我們在使用者的評分上可以做正規化（Normalization）的處理，使用的方法為標準分數（Standard Score，又稱 z-score）[26]，參考公式 4.1 我們透過將使用者對於這間店家的評分 r_{ui} 減去使用者曾經評分過的所有店家的平均評分 $\bar{r}_u$ 最後再除以一個標準差，透過正規化將使用者評分轉換為整體評分標準來達到提升預測準確性。

$$h(r_{ui}) = \frac{r_{ui} - \bar{r}_u}{\sigma_u} \tag{4.1}$$

4.3.3 遺忘曲線

使用者的興趣偏好是會隨著時間而有所改變，店家與實際的評論參考價值也應該根據時間有所變化，舉例來說一篇三天前發布的食記較於一年前發布的食記評論更具有參考價值。

遺忘曲線（Forgetting curve）是用於表述記憶中的中長期記憶的遺忘率的一種曲線，根據赫爾曼·艾賓浩斯（Hermann Ebbinghaus）所提出如圖 4.2 的艾賓浩斯記憶遺忘曲線（The Ebbinghaus Forgetting Curve）[28] 得知，遺忘的過程是不均勻的，最初遺忘速度很快並逐漸緩慢，我們想將其應用於食記參考價值，距離發布時間的長短之參考價值，並非固定逐漸減少，我們可以設近幾週的食記的參考價值較高，而已經過數個月甚至一年以上的食記參考價值逐漸降低最終平緩，依照經過時間的多寡給予評分相應的權重，如此進行協同過濾計算時，經過時間越長的評分紀錄則會稍為降低一些分數。

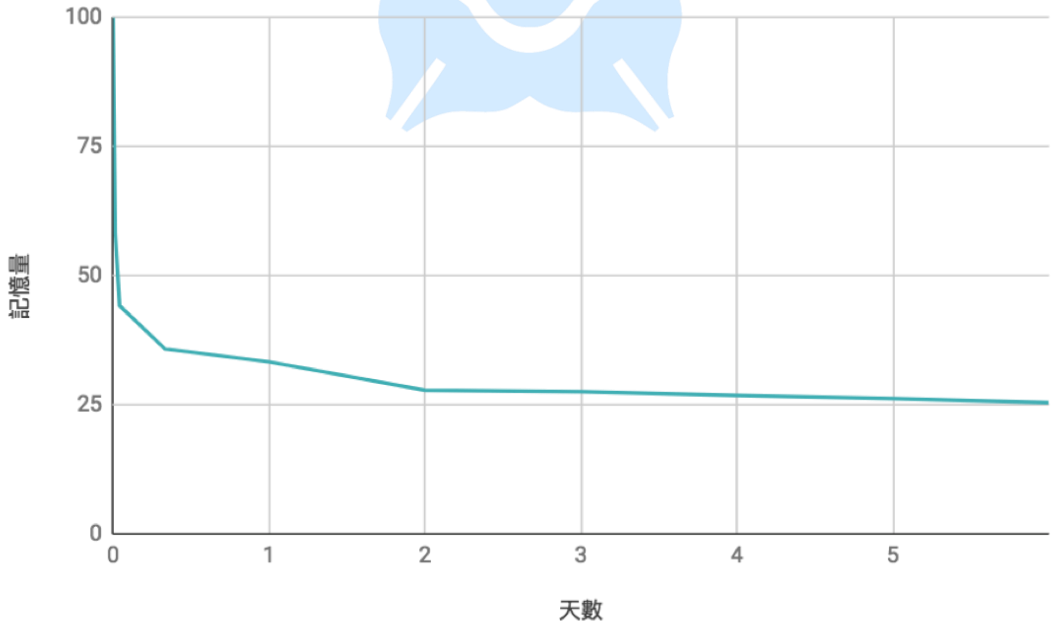

圖 4.2、艾賓浩斯記憶遺忘曲線

我們參考遺忘曲線近似公式定了一個評分權重公式 4.2 ，s 為記憶強度，t 是時間，時間 t 每一單位為三個月並設記憶強度 s 為 3，且我們希望權重不要低於 0.8，依據食記的發布時間將評分乘上對應的權重如表 4.2，如此距今時間較近的食記則會有較高的評分數值，即有更高的影響價值。

$$w = 0.8 + 0.2e^{-\left(\frac{t}{s}\right)} \tag{4.2}$$

表 4.2、食記評分經過時間與權重比例

經過時間	權重
三個月	0.9433
六個月	0.9027
一年	0.8527
兩年	0.8139

4.3.4 交替最小二乘法

協同過濾在實務上確實為一個簡單有效的方式，但仍存在著幾個缺點，我們嘗試使用交替最小二乘法（Alternating Least Squares，ALS）[5] 針對協同過濾其缺點稀疏問題（Sparsity）及系統延伸性問題（Scalability）進行改善。

協同過濾中評分(Rating)矩陣是由使用者(User)及食記中評分的店家(Item)所構成，當新使用者、店家的產生會因為沒有足夠資訊而導致矩陣稀疏問題如圖 4.3，同時使用者及店家數量的成長也會導致評分矩陣越來越巨大，因為使用者不可能將每個店家進行評分，這也就是所謂的系統延伸性問題如圖 4.4，矩陣越大

計算時間當然也會越久，此時矩陣分解（Matrix Factorization）有時候也無法應付如此巨大的數據量，而 ALS 演算法正是一項矩陣分解的優化技術。

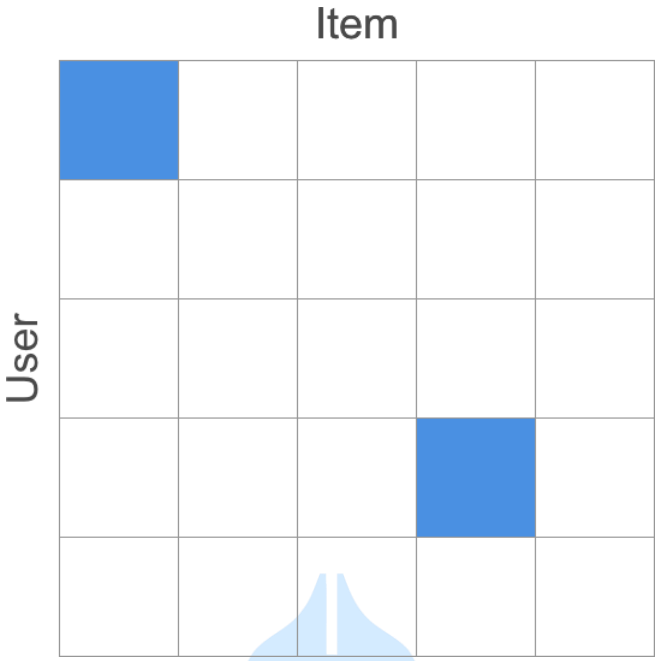

圖 4.3、協同過濾矩陣稀疏問題

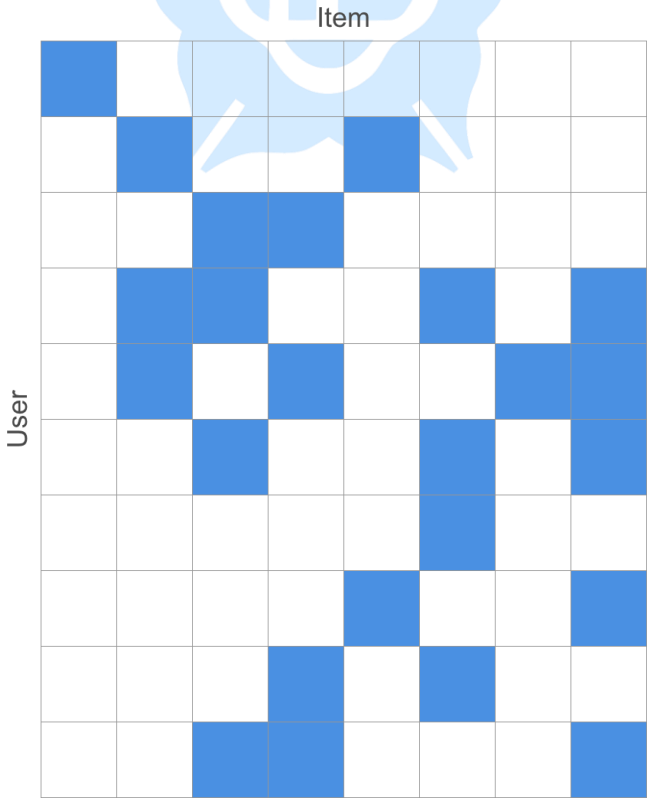

圖 4.4、協同過濾系統延伸問題

ALS 的核心假設說明評分矩陣是近似低秩矩陣（Low-rank Matrix），意指一個評分矩陣 r 可以分解為兩個矩陣的乘積，這兩個矩陣也就是所謂的使用者特徵矩陣 x 及店家特徵矩陣 y，為了得到這兩個矩陣我們需要對公式 4.3 進行最小化求值，並加入正規化項（Regularization item）來避免過擬合（Overfitting）的問題，λ 為正規化係數，最後透過矩陣分解的結果來填充評分矩陣的缺失值（Miss value），因此我們可以基於這個填充值的預測來對使用者進行推薦。

$$\min_{X,Y} \sum_{r_{ui}} (r_{ui} - x_u^T y_i)^2 + \lambda \left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right) \quad (4.3)$$

如圖 4.5 所示，我們可以訓練模型並進行評估，並依照需求視情況進行參數調整以達到較佳的結果，最終在使用較優異的模型進行推薦預測。

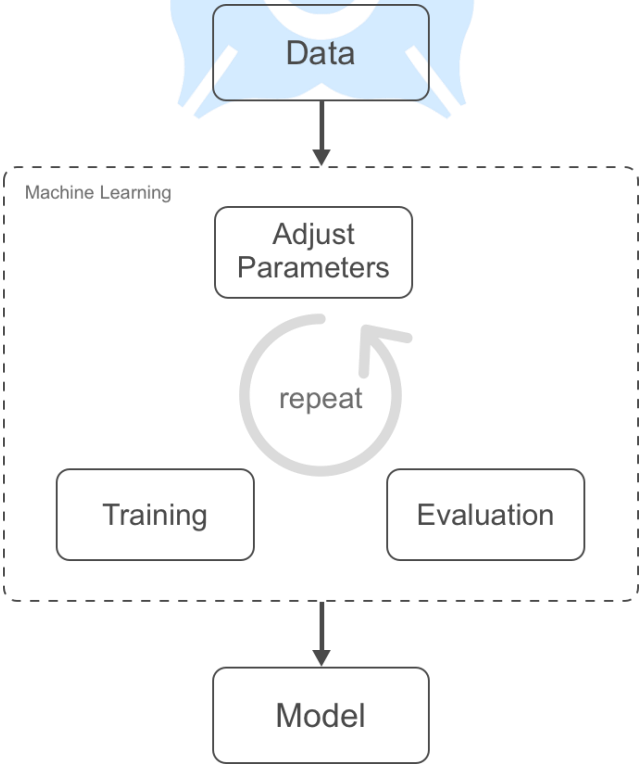

圖 4.5、推薦模型訓練流程

4.4 基於會話的推薦

協同過濾除了有稀疏（Sparsity）及系統延伸性（Scalability）的問題，還有存在著冷啟動（Cold-start）問題，協同過濾的推薦基於著使用者的數據，我們才能依此進行類聚計算相似性，進而依照評分給予推薦預測，如圖 4.6 所示若系統出現了一位新使用者，從沒有店家進行評分過，也缺乏其他數據，那麼系統並無法有效的對新使用者進行推薦，又或者是有新的店家，但沒有相關評分數據，結果亦同。

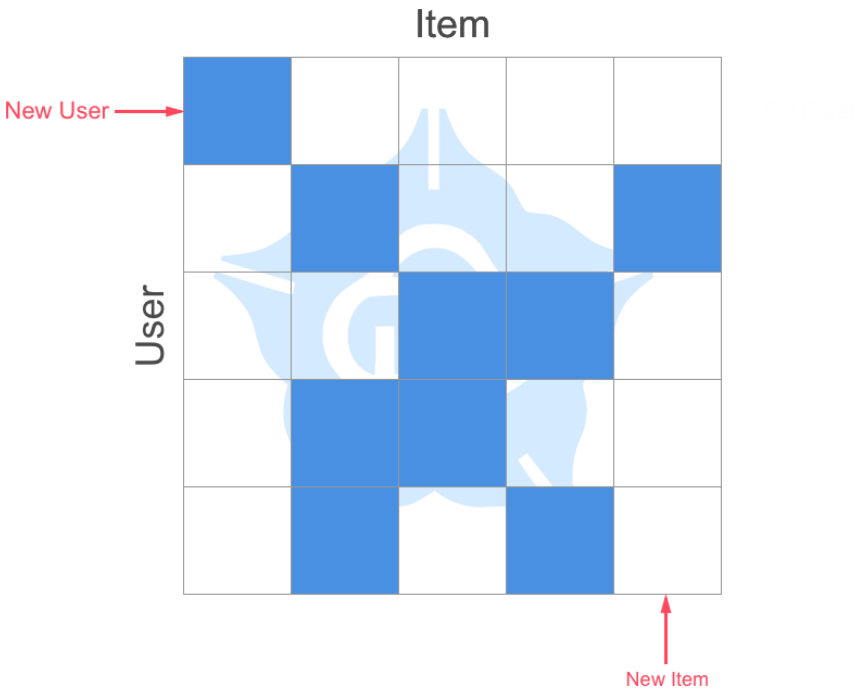

圖 4.6、協同過濾冷啟動問題

為了對新使用者進行推薦，我們嘗試分析舊有使用者操作系統時的行為並記錄其行為發生時間，使用者於使用系統前端時會有一系列的操作，我們將大量的使用者會話行為進行處理並做訓練，新的使用者在操作時將其當前發生行為當作輸入，讓模型預測使用者下一個可能會點擊的店家（Item）。

4.4.1 會話行為

我們在系統前端設計了行為記錄功能，當使用者使用搜尋系統或瀏覽店家時，會將其行為回傳至後端，記錄使用者、點擊的店家和點擊的時間，同一個使用者可能會有很多次的行為，但可能每次都是為了不同的目的，我們可以將其切分開來當作不同的會話（Session），例如使用者在這次會話搜尋了「中式」做了一系列的操作，下一次會話可能經過了幾小時並針對了「鍋物」有了一系列的操作，這些都是可以切分開來區分不同的會話，有了大量的會話行為及時間，我們就可以嘗試用深度學習進行時間序列的預測。

表 4.3、一段會話中瀏覽食記的行為

ID	Article	Timestamp
3	347935981	1529917097324
3	186989405	1529917146813
3	66269037	1529917289963
3	174007938	1529917412519
3	171218298	1529917583910
3	186874958	1529917680233
3	45971325	1529917713610
3	186921327	1529917741593
3	117762277	1529917937791

4.4.2 遞歸神經網路

Balázs Hidasi 等人於 2016 年提出「Session-based Recommendations with Recurrent neural network」[1]，這篇論文將遞歸神經網路（RNN）應用於推薦系

統中，論文中採用了門控遞歸單元（Gated Recurrent Unit）作為基本結構，建立了一個深度神經網路，針對會話來預測下一個項目（Item）的點擊機率，我們根據該論文進行實作來預測下一個點擊的店家的機率。

實作採用了 TensorFlow 來搭建神經網路，模型的輸入即為會話中的點擊序列（Sequence）經過 1-of-N coding 的編碼方式（One-hot Encoding）[19]，通過嵌入層（Embedding）壓縮成低維度的連續向量作為 GRU 的輸入，最後模型輸出每一個項目（Item）被點擊的預測機率。

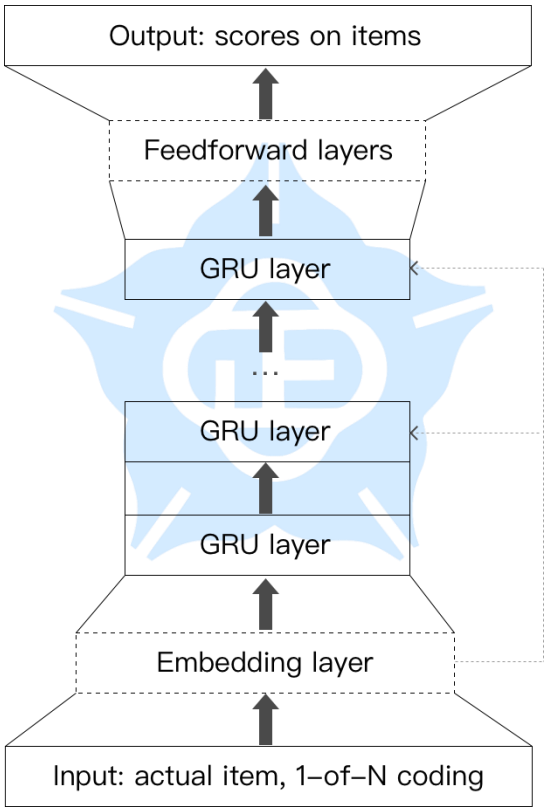

圖 4.7、基於會話推薦模型架構

我們使用單層 GRU 進行及表 4.4 作為訓練參數來訓練我們的推薦模型，同時可以透過模型評估來調整訓練參數達到較佳的效果，數據集的大小也會影響訓練的結果，我們要時常保持新數據進來時進行重新訓練。

表 4.4、遞歸神經網路訓練參數

參數	參數值
GRU Units	100
Batch size	64
Learning rate	0.001
Dropout	0.2
Loss function	cross-entropy
Activation function	tanh
Final Activation function	softmax

4.5 混合推薦

我們使用了協同過濾與基於會話行為的遞歸神經網路推薦，兩者推薦的結果會依照分數或機率由高至低排序，通常我們會選取較高的項目作為結果，並將兩者選取的項目進行混合。經測試協同過濾有著較高的點擊意願，我們可以將協同過濾選取比例設高一點，讓最終的混合推薦結果協同過濾佔有較高比例。

4.5.1 推薦系統架構

深度學習模型及推薦都是要定期維護的，隨著時間推進，使用者、食記及店家等數據都會增加，使用者的偏好和熱門的美食都是會變動的，因此推薦模型也需要定期評估維護並進而調教參數及週期性訓練，模型的生命週期透過評估結果的方式來調整，系統將資料庫數據進行前處理後進行協同過濾及遞歸神經網路的訓練，保存的模型供後續預測使用，我們約每經過一週會加入新數據並重新訓練模型，使得新數據的影響能應用於推薦系統之中，最終推薦系統架構如圖 4.8。

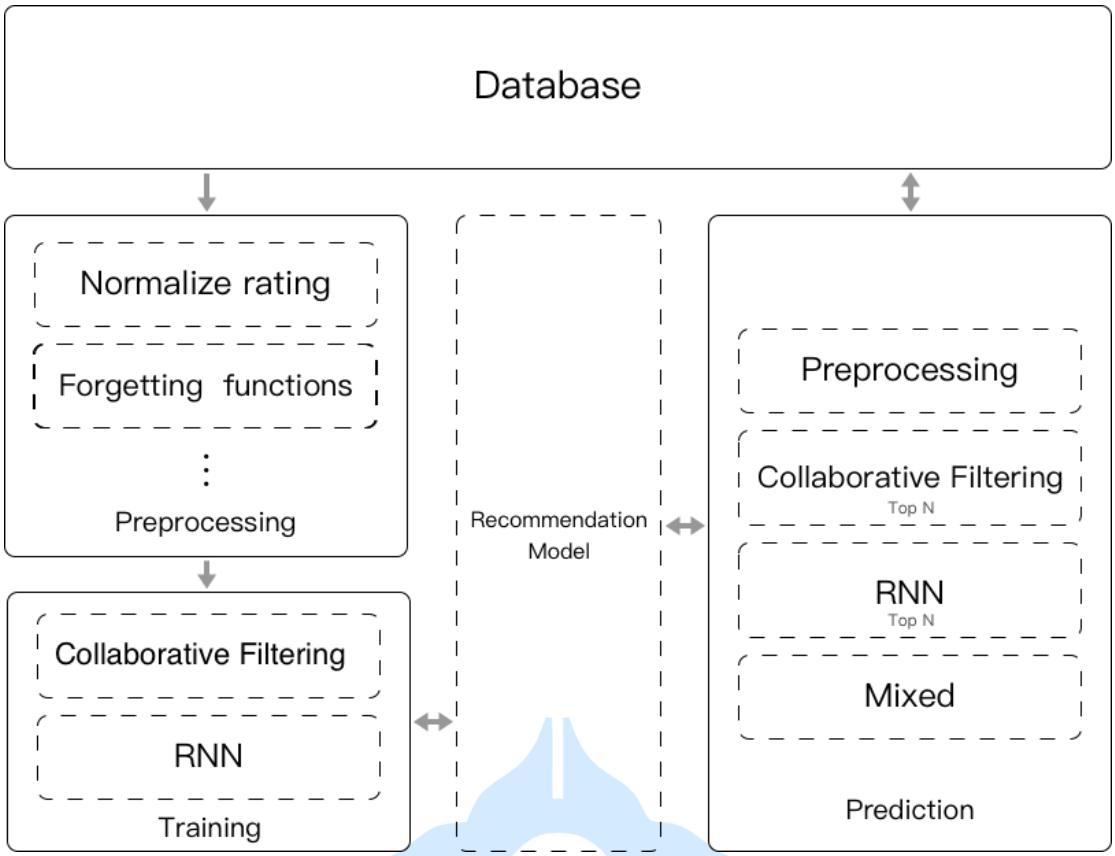

圖 4.8、推薦系統架構

4.5.2 推薦結果選取

我們採用共 12 筆混合推薦結果，混合推薦是由不同推薦方法的結果經特定比例進行混合，若使用者數據足夠協同過濾使用，則協同過濾的結果將優先選取，反之若新使用者因數據過少，因協同過濾的結果不佳，則選取較高基於會話行為的推薦，最後混合偏好類別高評價的結果，推薦結果的選取比例可參考表 4.5。

正常情況下我們參考表 4.6 ，針對協同過濾獲得前 5 筆推薦結果，基於會話行為的遞歸神經網路推薦結果取 4 筆，依照類別隨機推薦結果取 3 筆。若協同過濾的推薦結果不足，則以會話行為的推薦結果遞補，至多不超過 6 筆上限，剩餘則以類別隨機填滿上限，最後推薦集合隨機排列後即為推薦結果。

表 4.5、各推薦方法選取比例及上限

推薦方法	選取比例	選取上限
協同過濾（CF）	5	6
遞歸神經網路（RNN）	4	6
類別隨機（Random category）	3	12

表 4.6、各推薦方法選取比例範例

推薦方法	選取比例			
	一般情況	CF 缺少	RNN 缺少	CF & RNN 缺少
CF	5	2	6	0
RNN	4	6	3	0
Random	3	4	3	12

第五章 系統呈現

5.1 系統呈現

我們對平台主要服務及推薦進行呈現探討，說明於前端提供何種功能及如何給予使用者推薦結果及呈現方式。

5.1.1 瀏覽食記

美食平台上的使用者除了進行食記撰寫，也可以像社群軟體一樣打卡，標註美食位於哪間店家，食記的評分除了提供自己及其他使用者參考，評分及使用者常撰寫的食記類型、打卡店家及收藏偏好等數據皆為推薦系統的重要依據。

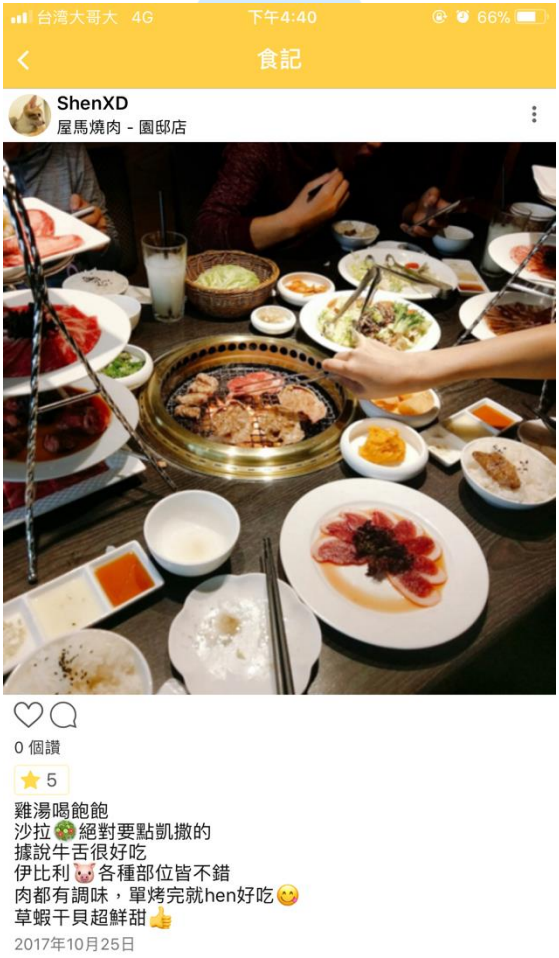

圖 5.1、App 瀏覽食記

5.1.2 店家資訊

當使用者尋找附近有什麼好吃的美食，會想查看店家的詳細資訊，我們實作了一個店家資訊頁面，提供所有食記評論給予的評分及地址等資訊，讓使用者可以迅速掌握店家資訊，也提供使用者可以進行店家的收藏。

圖 5.2、App 店家資訊

5.1.3 首頁推薦

我們提供使用者透過美食平台尋找附近有什麼好吃的美食，於前端 App 建置一首頁，讓使用者啟動 App 時即可看見推薦的美食店家，如圖 5.3 「為您推薦」為推薦系統給予使用者的推薦結果，透過左右滑動可以瀏覽更多的推薦店家。

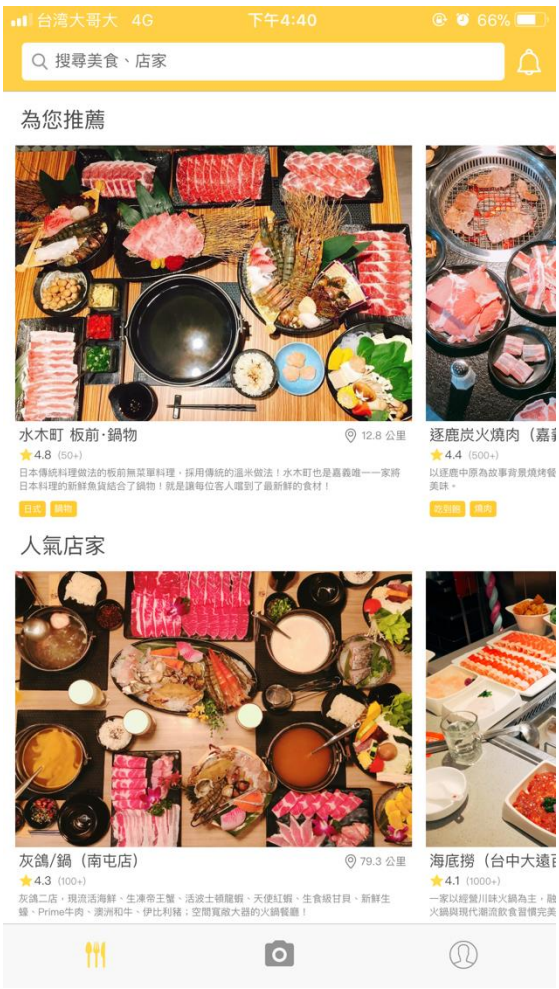

圖 5.3、App 首頁推薦

5.1.4 搜尋推薦

使用者也可以透過搜尋功能來進行美食店家的搜尋，於搜尋結果頁面上方提供「您可能會喜歡」的項目清單，透過左右滑動可以瀏覽系統推薦的店家，推薦的結果除了協同過濾推薦，後端也會針對此次的會話行為來進行推薦預測並將推薦結果進行混合。

圖 5.4、App 搜尋推薦

第六章 系統實驗

6.1 實驗方法

為了驗證推薦結果是否對於使用者有所幫助，我們進行了模型的評估並針對推薦進行前端實驗，前端實驗為後端隨機選擇一種方式進行推薦，使用者獲得推薦列表後進行點擊行為蒐集，從單一推薦方法到混合三種推薦演算法，來分析從單一類別到混合類別、協同過濾、遞歸神經網路的推薦結果是否有助於提升使用者點擊率，分別依照表 6.1 及表 6.3 做了兩次實驗，選取的比例分別為表 6.2 及 6.4，兩次實驗分別在不同的時間區間蒐集，實驗一蒐集了約三星期的行為數據，實驗二蒐集了約兩星期的行為數據，每次實驗採用 A/B Testing[4] 的方式來隨機給予使用者不同的推薦方法的結果。

表 6.1、實驗一使用推薦方法

推薦方法	說明
Random category	從單一類別隨機選取進行推薦。
CF	使用協同過濾的推薦結果。
RNN	使用遞歸神經網路的推薦結果。
CF & RNN	協同過濾、遞歸神經網路經特定比例來混合推薦結果。
Random category & CF	單一類別、協同過濾經特定比例來混合推薦結果。
Random category & CF & RNN	單一類別、協同過濾、遞歸神經網路經特定比例來混合推薦結果。

表 6.2、實驗一各方法推薦結果選取比例

推薦方法	選取比例		
	Random category	CF	RNN
Random category	12	-	-
CF	12	-	-
RNN	12	-	-
CF & RNN	-	6	6

Random category & CF	6	6	-
Random category & CF & RNN	3	5	4

表 6.3、實驗二使用推薦方法

推薦方法	說明
Random category	從單一類別隨機選取進行推薦。
Random category & CF	單一類別、協同過濾經特定比例來混合推薦結果。
Random category & CF & RNN	單一類別、協同過濾、遞歸神經網路經特定比例來混合推薦結果。

表 6.4、實驗二各方法推薦結果選取比例

推薦方法	選取比例		
	Random category	**CF**	**RNN**
Random category	12	-	-
Random category & CF	6	6	-
Random category & CF & RNN	3	5	4

6.2 實驗數據

我們使用美食平台上的資料，其中包含使用者、食記、照片、喜好、收藏店家、瀏覽及搜尋等行為數據，針對店家有效數據約 6,593 筆，涵蓋使用者針對店家食記及收藏，有效的會話行為數據為 38,617 筆。

6.3 模型評估

我們將針對協同過濾的推薦模型進行評估計算均方根誤差（Root-Mean-

Squared Error，RMSE）[25]，並將數據集做 K 折交叉驗證（K-Fold Cross Validation）分別計算均方根誤差，參考公式 6.1 評分總數為 n，d_i 為實際評分，p_i 為預測評分，我們取 K 為 10 做 10 折交叉驗證，並將所得到均方根誤差進行平均得到值約為 0.34684525293531，最後計算均方誤差的值可應用於交替最小二乘法中調整參數的依據，透過重複訓練以達到較好的推薦，最終協同過濾的模型使用表 6.5 作為訓練參數。

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i - p_i)^2} \tag{6.1}$$

表 6.5、交替最小二乘法訓練參數

參數	參數值	說明
Rank	10	欲使用特徵因子數量
Lambda	0.0001	正則化因子
Iterations	20	最大迭代次數

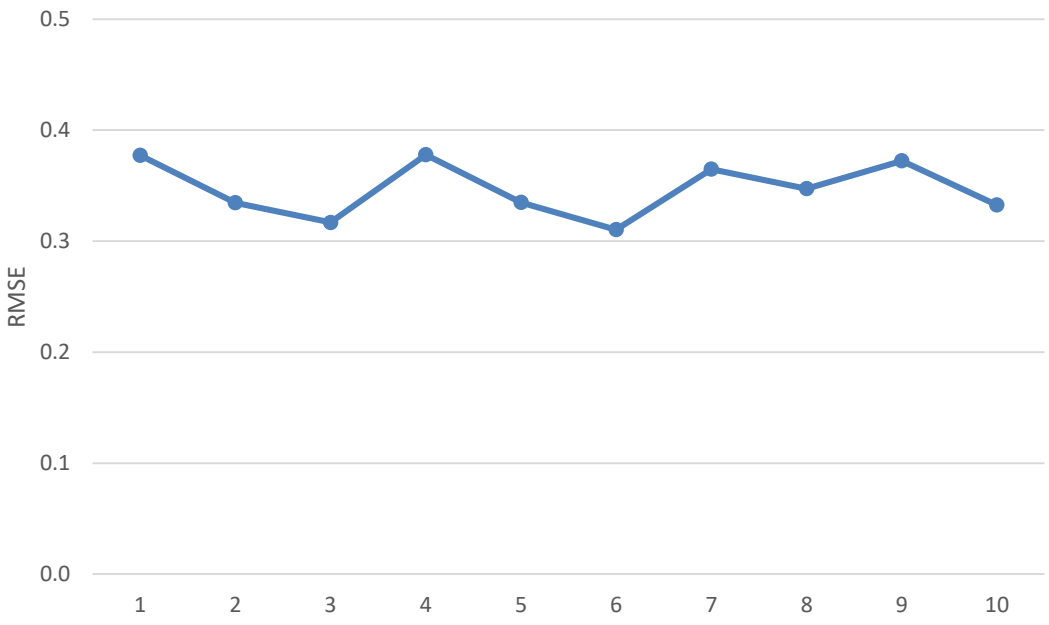

圖 6.1、ALS 訓練結果 10 折交叉驗證

針對會話行為推薦的遞歸神經網路模型評估，我們使用 MRR（Mean Reciprocal Rank）[15] 及召回率（Recall）[22] 來進行評估，參考公式 6.2 計算 MRR，$|Q|$ 為預測數量，$Rank_i$ 為預測出來的結果序列於正確答案排在第幾位，參考公式 6.3 計算 Recall，每個預測取前 20 個結果 MRR 及 Recall 計算結果如表 6.6。

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{Rank_i} \tag{6.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{6.3}$$

表 6.6、基於會話行為推薦的模型評估結果

MRR@20	Recall@20
0.2743	0.5966

6.4 實驗結果

前端顯示的推薦為後端進行混合的推薦結果，我們透過後端標記推薦是透過協同過濾、遞歸神經網路或隨機的方式來進行統計，分別統計使用者點擊的店家及是否對於傳統推薦有顯著提升。

表 6.7 我們將使用者點擊推薦的行為隨機抽取並進行分析，統計了混合推薦結果下使用者於推薦中點擊的筆數，使用協同過濾的推薦，佔了一半以上的點擊數，可表示協同過濾的點擊率高於該類別下隨機推薦及遞歸神經網路的推薦結

果，意即相較於類別下隨機，推薦使用者更願意點擊協同過濾及遞歸神經網路的推薦結果。

表 6.7、針對推薦有效點擊隨機 1,000 筆數據

推薦方法	點擊筆數
協同過濾（CF）	576
遞歸神經網路（RNN）	289
類別隨機（Random category）	135

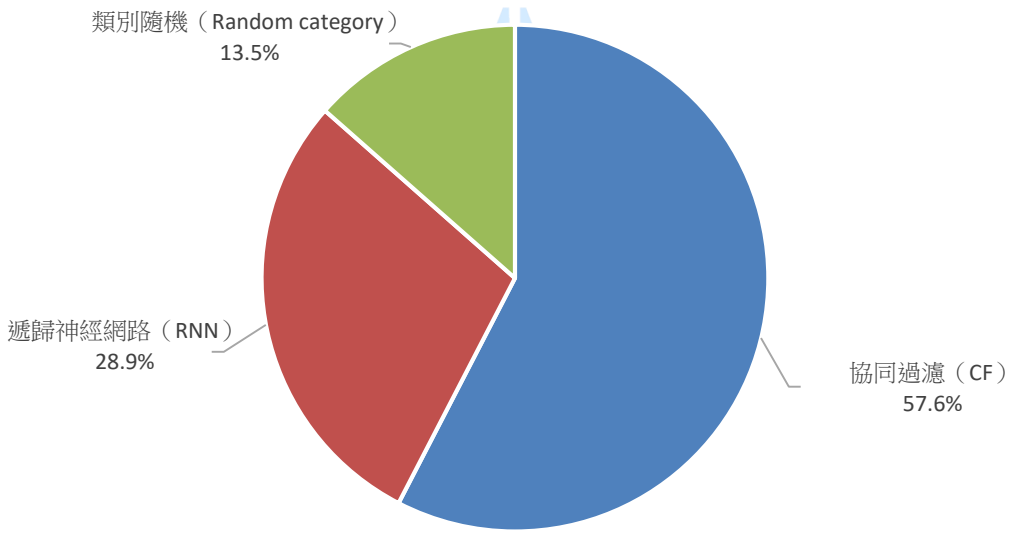

圖 6.2、各推薦方法點擊比例

我們使用了 A/B Testing 對使用者進行不同方法的測試，最後將其點擊結果進行分析，統計了使用者從各個推薦方法中的推薦結果點擊率，實驗一我們採用了個別單一推薦演算法、混合協同過濾與遞歸神經網路及混合類別隨機等共六種混合推薦方法，各推薦方法的點擊率參考表 6.8。

表 6.8、實驗一的各推薦方法點擊率

推薦方法	點擊率
Random category	14.9%
CF	31.8%
RNN	23.7%
CF & RNN	38.1%
Random category & CF	32.6%
Random category & CF & RNN	48.2%

我們發現採用隨機推薦還是有部分的使用者進行點擊，而單純採用協同過濾也有不錯的效果，至於單純採用遞歸神經網路則較類別隨機小幅提升，其中我們也使用了混合協同過濾及遞歸神經網路有取得近 38% 的點擊機率，而將三種方法進行混合，得到的結果表示有約 48% 點擊率。

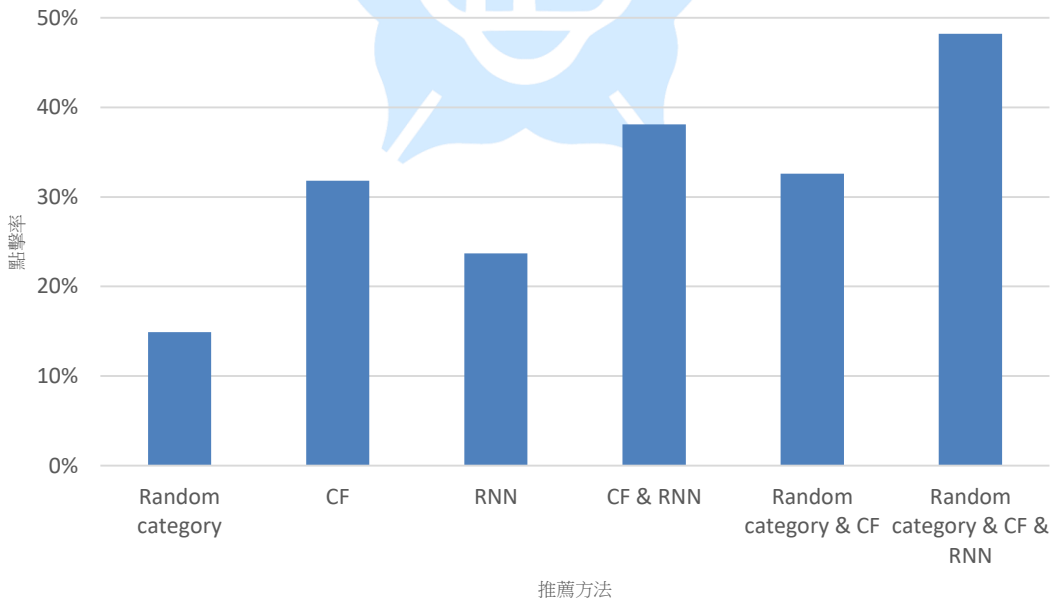

圖 6.3、實驗一使用者於不同推薦推薦方法的點擊率

實驗二我們採用了逐一混合推薦演算法，來觀察混合推薦是否有助於點擊率的提升，其方法包括單一推薦演算法、混合單一推薦及協同過濾與混合類別隨機、協同過濾與遞歸神經網路的推薦結果共三種推薦方法，各推薦方法的點擊率參考表 6.9。

表 6.9、實驗二的各推薦方法點擊率

推薦方法	點擊率
Random category	17.9%
Random category & CF	36.4%
Random category & CF & RNN	43.1%

我們發現單一方法的推薦使用者願意點擊推薦結果有近 18%，結合了協同過濾後點擊率提高至 36% ，最後混合了基於會話行為的遞歸神經網路推薦讓整體點擊率上升到了 43% ，可以得知將推薦結果進行混合能提升使用者願意點擊的意願。

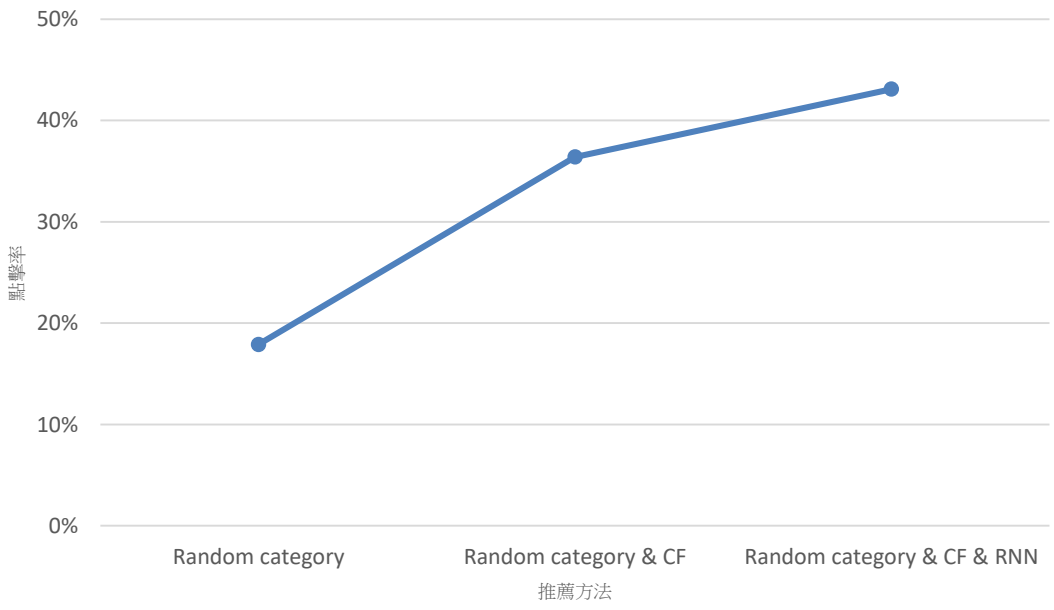

圖 6.4、實驗二使用者於不同推薦推薦方法的點擊率

第七章結論

本論文建立一美食平台提供使用者分享食記，並透過使用者的相關數據來進行店家美食推薦，推薦系統中應用了使用者評分的這種顯性資訊來做協同過濾，並將評分依照使用者的評分習慣做正規化處理，此外也考慮時間因素對於使用者偏好的影響訂定了遺忘函數將評分權重進行調整，有了這些使用者相關數據，我們使用了交替最小二乘法進行模型的訓練，其協同過濾推薦結果根據統計，在使用者點擊率擁有不錯的推薦效果。

在協同過濾的基礎下雖得到了不錯的推薦效果，但針對新使用者導致的冷啟動問題，我們嘗試將使用者的行為依照時間序列進行遞歸神經網路的模型建置，在平台上的進行混合推薦，並透過前端提供給使用者。

我們利用了深度學習的技術來進行推薦模型的建置，本論文的混合推薦系統，輔助單純協同過濾推薦的不足，也將各推薦方法依設置的比例混合其推薦結果，根據前端實際測試，其結果顯示混合推薦方法能有效提升使用者點擊推薦結果的點擊率。

在未來，平台也有可以更精進的地方，如協同過濾我們可以蒐集更多使用者相關數據來訓練隱藏特徵來持續優化模型，基於會話的遞歸神經網路推薦模型也可以增加考慮其他因素，例如將使用者停留的時間當作考慮因素，若停留時間越長則表示使用者對此越感興趣，透過持續的優化調整來達到越好的推薦成效。

參考文獻

[1] Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2015). Session-based recommendations with recurrent neural networks. *CoRR*, abs/1511.06939, 2015.

[2] Ramage, D., Hall, D., Nallapati, R., & Manning, C. D. (2009). Labeled LDA: A supervised topic model for credit attribution in multi-label corpora. *EMNLP* 2009.

[3] 吳思璇（2017）。**旅遊搜尋系統之設計與實作**。國立中正大學資訊工程研究所碩士論文，未出版，嘉義縣。

[4] A/B Testing, https://en.wikipedia.org/wiki/A/B_testing

[5] Alternating Least Squares, https://spark.apache.org/docs/latest/mllib-collaborative-filtering.html

[6] Bag-of-Words Model, https://en.wikipedia.org/wiki/Bag-of-words_model

[7] Collaborative Filtering, https://en.wikipedia.org/wiki/Collaborative_filtering

[8] Convolutional Neural Network, https://en.wikipedia.org/wiki/Convolutional_neural_network

[9] CUDA, https://en.wikipedia.org/wiki/CUDA

[10] Inception V3, https://ai.googleblog.com/2016/03/train-your-own-image-classifier-with.html

[11] Keras, https://keras.io/

[12] K-Nearest Neighbor, https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[13] Latent Dirichlet Allocation, https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

[14] Long Short-Term Memory, https://en.wikipedia.org/wiki/Long_short-term_memory

[15] Mean Reciprocal Rank, https://en.wikipedia.org/wiki/Mean_reciprocal_rank

[16] MongoDB, https://www.mongodb.com/

[17] Nearest Neighbor Search, https://en.wikipedia.org/wiki/Nearest_neighbor_search

[18] Node.js, https://nodejs.org/

[19] One-hot Encoding, https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/

[20] PostgreSQL, https://www.postgresql.org/

[21] Python, https://www.python.org/

[22] Recall, https://en.wikipedia.org/wiki/Precision_and_recall

[23] Recurrent Neural Networks, https://en.wikipedia.org/wiki/Recurrent_neural_network

[24] RESTful API, https://en.wikipedia.org/wiki/Representational_state_transfer

[25] Root-Mean-Squared Error, https://en.wikipedia.org/wiki/Root-mean-square_deviation

[26] Standard Score, https://en.wikipedia.org/wiki/Standard_score

[27] TensorFlow, https://www.tensorflow.org/serving/

[28] The Ebbinghaus Forgetting Curve, https://en.wikipedia.org/wiki/Forgetting_curve