AI Dining Suggestion App

By Bao Pham

Advisor: Dr. Chris Pollett
Committee Member: Dr. Mike Wu
Committee Member: Dr. Kevin Montgomery
Agenda

1. Introduction
2. Preliminaries
3. Implementation
4. Experiments
5. Conclusion
Introduction
Trying to decide what to eat can sometimes be challenging and time-consuming for people. Google Maps and Yelp are two main applications of the web that provide lists of restaurants. One’s taste at a certain time does not always depend on the reviews of a restaurant. Google Maps and Yelp apps still offer the traditional way of typing keywords, and the apps show the list of potential matches.
Introduction

Issues:

- Google Maps and Yelp do not really learn about a user’s taste.
- Lack of simplicity with regard to functionality.
Proposed Solution

- Use AI models to learn the user's dining pattern over time to help make restaurant suggestions.
- Create friendly UI app with simplicity to change the way of how data is presented to user.

Goal: to reduce time when it comes to food decision making process.
Related Work

- Several papers discuss improvements to Yelp’s recommendation systems using neural networks and machine learning algorithms.
- These models are static and would only provide the same output if given the same input of restaurants because they are trained on Yelp data.
- Not much work found in terms of how the restaurant data is presented to the user.
- They indicated that machine learning methods can be useful for recommendation systems.
Related Work

- Youtube’s recommendation system paper (2016):
  - It has two layers
  - Candidate Generation filters the options that are irrelevant for an average user
  - Ranking ranks the options that are filtered by the first layer for a specific user
Related Work

- UI was designed following Tinder’s format:
  - Tinder is a dating app which has gained lots of traction and has become a success due to its simplicity
  - Options are presented as a deck of cards
Preliminaries
Technology Stack

- React Native
  - Javascript
  - Multi-platform (iOS & Android)
  - Strong community support with lots of plugins

- MongoDB
  - NoSQL, flexible schema
  - JSON objects
  - mLab

- NodeJS
  - Javascript
  - RESTful
  - Amazon Web Services
Technology Stack
AI Technology

- Tensorflow
  - Common neural network framework
- Keras
  - Neural network library
  - Different implementations of commonly used NNs
- Flask
  - Web framework for Python
Implementation
App Design

- Swiping left or right to show like or dislike
- When finalizing the decision, the app should show name and address of the restaurant
Implementation

- RESTful API with each component structure as follows:
  - Router
  - Controller
  - Service
  - Object
  - Model

- Data collection with Google and Yelp APIs
Implementation

```json
{
  "geometry": {
    "location": {
      "lat": 37.34459,
      "lng": -121.838002
    },
    "viewport": {}
  },
  "icon": "https://maps.gstatic.com/mapfiles/place_api/icons/restaurant-71.png",
  "id": "022beae10df8f3de57a303f0e0805479c758cef",
  "name": "La Mejor Taqueria",
  "opening_hours": {
    "open_now": true
  },
  "photos": [],
  "place_id": "ChIJG798FjJ4AR7Th0RBwZ5W8",
  "plus_code": {
    "compound_code": "85V6+QQ San Jose, California",
    "global_code": "849W85V6+QQ"
  },
  "price_level": 2,
  "rating": 4.3,
  "reference": "ChIJG798FjJ4AR7Th0RBwZ5W8",
  "scope": "GOOGLE",
  "types": [
    "restaurant",
    "point_of_interest",
    "food",
    "establishment"
  ],
  "vicinity": "2003 Story Rd # 975, San Jose"
}
```
Implementation
App Design

● Challenges:
  ○ Yelp query limit
  ○ Only 3 images/restaurant from Yelp
  ○ Duplicate restaurants from both Yelp and Google
  ○ Large amount of queries made to Yelp and Google APIs -> surpass the free tiers

● Solutions:
  ○ Switch all of the queries to Google except initial query
  ○ Map restaurants using location to avoid duplicates
  ○ Create a cache system to avoid repetitive queries
External Services for API

- Food decisions rely on some other factors, not just the restaurants themselves.
- Three extra types of information:
  - Temperature
  - Travel time
  - Busy hours
Front-end Implementation

- Authentication (Unique ID for each user)
- React Native Deck Swiper
- React Native Image Viewer
Filtering and Processing Data for AI Training

- Filtering data:
  - Yelp’s challenge dataset
  - 12 Gigabytes (business, user, review, photo data)
  - Not all businesses are restaurants
  - Filtered by categories, attributes

- Processing data:
  - Tensorflow can’t train string typed data
  - Converted data to integer and float type using a mapping system

- 1,418,418 reviews with ratings, 32,516 restaurants, 191,904 users and 69 restaurant categories for ~3 Gigabytes of data
Filtering and Processing Data for AI Training

```json
{
    "business_id": "Apn5Q_b6Nz61Tq4XzPdf9A",
    "name": "Minhas Micro Brewery",
    "neighborhood": "",
    "address": "1314 44 Avenue NE",
    "city": "Calgary",
    "state": "AB",
    "postal_code": "T2E 6L6",
    "latitude": 51.0918130155,
    "longitude": -114.031674872,
    "stars": 4.0,
    "review_count": 24,
    "is_open": 1,
    "attributes":{
        "BikeParking": "False",
        "BusinessAcceptsCreditCards": "True",
        "BusinessParking": "{garage": False, 'street': True, 'validated': False, 'lot': False, 'valet': False}",
        "GoodForKids": "True",
        "HasTV": "True",
        "NoiseLevel": "average",
        "OutdoorSeating": "False",
        "RestaurantsAttire": "casual",
        "RestaurantsDelivery": "False",
        "RestaurantsGoodForGroups": "True",
        "RestaurantsPriceRange2": "2",
        "RestaurantsReservations": "True",
        "RestaurantsTakeOut": "True"
    },
    "categories": "Tours, Breweries, Pizza, Restaurants, Food, Hotels & Travel",
    "hours":{
        "Monday": "8:30-17:00",
        "Tuesday": "11:0-21:00",
        "Wednesday": "11:0-21:00",
        "Thursday": "11:0-21:00",
        "Friday": "11:0-21:00",
        "Saturday": "11:0-21:00"
    }
}
```
Filtering and Processing Data for AI Training

<table>
<thead>
<tr>
<th>user_id</th>
<th>business_id</th>
<th>category</th>
<th>rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.065</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.222</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.365</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.425</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.565</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.604</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1505</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1652</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1705</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.18143</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1902</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2005</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2195</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Training Static AI Model

- Matrix Factorization as a baseline model
- Using business, user IDs and ratings
Training Static AI Model

- Improvements on baseline model
- Replaced Dot Layer by three fully connected layers: 128, 64, 1 with two Dropout Layers with dropout rate 0.2
- Dataset was sparse
- Some restaurants have one rating
- Some users only rates once
- Optimal architecture at this point given training time trade-off
How to prepare data when each action recorded from the app is an object? (LSTM)

- Unfolded each object into a string of field
- Unlike video games when feedback is learned immediately from each action -> training stack is 20 actions
- Used Label Encoding and One Hot Encoding
Training Reinforcement Learning Model

<table>
<thead>
<tr>
<th>Food Name</th>
<th>Categorical #</th>
<th>Calories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>1</td>
<td>95</td>
</tr>
<tr>
<td>Chicken</td>
<td>2</td>
<td>231</td>
</tr>
<tr>
<td>Broccoli</td>
<td>3</td>
<td>50</td>
</tr>
</tbody>
</table>

A day value = 0 -> 6 for Mon -> Sun
69 restaurant categories = 0 -> 68

Label Encoding

<table>
<thead>
<tr>
<th>Food Name</th>
<th>Categorical #</th>
<th>Calories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>1</td>
<td>95</td>
</tr>
<tr>
<td>Chicken</td>
<td>2</td>
<td>231</td>
</tr>
<tr>
<td>Broccoli</td>
<td>3</td>
<td>50</td>
</tr>
</tbody>
</table>

One Hot Encoding

<table>
<thead>
<tr>
<th></th>
<th>Apple</th>
<th>Chicken</th>
<th>Broccoli</th>
<th>Calories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>95</td>
</tr>
<tr>
<td>Chicken</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>231</td>
</tr>
<tr>
<td>Broccoli</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>50</td>
</tr>
</tbody>
</table>
Training Reinforcement Learning Model

- One action string was converted into:

\[0, 4, 11, 16, 20, 51, 1, 0, 0, 0\]
App Experiment

- Five testers is sufficient as a study’s result from Nielsen Norman Group
- The formula only holds for comparable users who will be using the app in fairly similar ways
- A/B testing
- Great influence from Tinder
- “Dislike” and “Like” were confusing
App Experiment
AI Experiment

- Used an average user from Yelp’s dataset
- First layer filtered the low rated restaurants
- At first, the testers did not notice the difference from the reinforcement learning model (accuracy only 70%)
- As more data came in, the accuracy gradually went up
- Model learned some restaurants that users had disliked to push down to the stack
AI Experiment

<table>
<thead>
<tr>
<th>layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>lstm_1 (LSTM)</td>
<td>(None, 28)</td>
<td>1760</td>
</tr>
<tr>
<td>dense_1 (Dense)</td>
<td>(None, 4)</td>
<td>84</td>
</tr>
</tbody>
</table>

Total params: 1,844  
Trainable params: 1,844  
Non-trainable params: 0

2019-04-26 15:03:01.026312: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2

2019-04-27 14:56:05.425934: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2

Epoch 1/5

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>Loss</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>0.8267</td>
<td>0.3083</td>
</tr>
<tr>
<td>250</td>
<td>0.6900</td>
<td>0.4533</td>
</tr>
<tr>
<td>300</td>
<td>0.6356</td>
<td>0.6742</td>
</tr>
<tr>
<td>300</td>
<td>0.5989</td>
<td>0.7017</td>
</tr>
<tr>
<td>300</td>
<td>0.5750</td>
<td>0.7017</td>
</tr>
<tr>
<td>300</td>
<td>0.5750</td>
<td>0.7017</td>
</tr>
<tr>
<td>300</td>
<td>0.5750</td>
<td>0.7017</td>
</tr>
<tr>
<td>300</td>
<td>0.5750</td>
<td>0.7017</td>
</tr>
<tr>
<td>300</td>
<td>0.5750</td>
<td>0.7017</td>
</tr>
</tbody>
</table>

Epoch 2/5

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>Loss</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>0.7159</td>
<td>0.4567</td>
</tr>
<tr>
<td>250</td>
<td>0.6325</td>
<td>0.5512</td>
</tr>
<tr>
<td>300</td>
<td>0.5808</td>
<td>0.6988</td>
</tr>
<tr>
<td>300</td>
<td>0.5491</td>
<td>0.7532</td>
</tr>
<tr>
<td>300</td>
<td>0.5287</td>
<td>0.8034</td>
</tr>
<tr>
<td>300</td>
<td>0.5287</td>
<td>0.8034</td>
</tr>
<tr>
<td>300</td>
<td>0.5287</td>
<td>0.8034</td>
</tr>
<tr>
<td>300</td>
<td>0.5287</td>
<td>0.8034</td>
</tr>
<tr>
<td>300</td>
<td>0.5287</td>
<td>0.8034</td>
</tr>
</tbody>
</table>
address: "408 Barber Ln, Milpitas"

busy_hours: (7) [{...}, {...}, {...}, {...}, {...}, {...}, {...}]
dislikes: null
distance: {distance: {...}, duration: {...}, durationInTraffic: {...}}
favored: null
id: "8KayUocIVMfgfEuqP1Q-rw"
in_db: true
likes: null
location: {lat: 37.4210596148519, lng: -121.916425418683}
name: "Pepper Lunch USA"
open_now: true
photos: (10) [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}]
place_id: "ChIJQfdhqCbJ1j4ARE9zXHEhMh-Q"
predicted_action: 2
predicted_rating: 3.4051642417907715
price: 2
rating: 3.5
temperature: {temp: 292.59, pressure: 1015, humidity: 72, temp_min: 289.15, temp_max: 295.93}
Conclusion
Conclusion

- The simplicity of the app reduced the time taken to search for restaurants.
  - Typing keywords and searching take ~10 minutes
  - After loading a deck of restaurants, only take few seconds to swipe (3-5 minutes)
- With more data and time, reinforcement learning can really enhance the recommendation systems
  - Accuracy increased from 70% -> 80% from 500 records -> 1800 records
Future Work

- Other AI architectures can be experimented
  - More layers, other type of neural networks (DFF, RNN, etc.)
  - Other types of data
- Group functionality can be derived from individual’s preferences
  - When forming suggestions for a group, different lists of suggestions for individuals can be combined.
References


Acknowledgement

- Advisor: Professor Chris Pollett
- Committee members:
  - Dr. Mike Wu
  - Dr. Kevin Montgomery
- The great testers
Demo

“A demo is worth a thousand pictures”
Thank You!