Automatic User Profiling for Intelligent Tourist Trip Personalisation

Liam Attard

Supervisor(s): Dr Josef Bajada

Faculty of ICT

University of Malta

June 2021

Submitted in partial fulfillment of the requirements for the degree of B.Sc. ICT in Artificial Intelligence (Hons.)

L-Università ta' Malta

FACULTY/INSTITUTE/CENTRE/SCHOOL______________________

DECLARATIONS BY UNDERGRADUATE STUDENTS

Student's I.D. /Code ____________________________

Student's Name & Surname ____

Course ____

Title of Long Essay/Dissertation

____

____

____

Word Count ___________

(a) Authenticity of Long Essay/Dissertation

I hereby declare that I am the legitimate author of this Long Essay/Dissertation and that it is my original work.

No portion of this work has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher education.

I hold the University of Malta harmless against any third party claims with regard to copyright violation, breach of confidentiality, defamation and any other third party right infringement.

(b) Research Code of Practice and Ethics Review Procedures

I declare that I have abided by the University's Research Ethics Review Procedures.

______________________ ______________________

Signature of Student Name of Student (in Caps)

____

Date

08.02.2018

Abstract:

Sample abstract.

Acknowledgements:

Sample Acknowledgement

Contents

List of Figures

List of Tables

1 Background Research and Literature Review

This section aims to position our study by describing the technologies we will use in this dissertation. In the first part, we provide an overview of the TTDP research area. We then describe methods of retrieving POIs from a location, existing user-profiling techniques and existing research regarding automatic tourist planning.

1.1 Recommender Systems

We introduce the term Recommender Systems (RS) as a solution for the TTDP and present background knowledge and relevant work that forms this thesis's basis. RSs hold use cases in diverse fields, such as e-commerce, media, and tourism [1]. However, in tourism, RSs offer tourists information in a unified and centralised manner, providing them with a plan for their trip [2–4]. Two domains develop current RSs for the TTDP solutions, which are; methods for obtaining tourist products (such as events and Point of Interests (POI)) and tour recommendation algorithms that create tourist trips [4] as shown in Figure 1. The following sections discuss related work in each field and discuss adding personalisation to RSs.

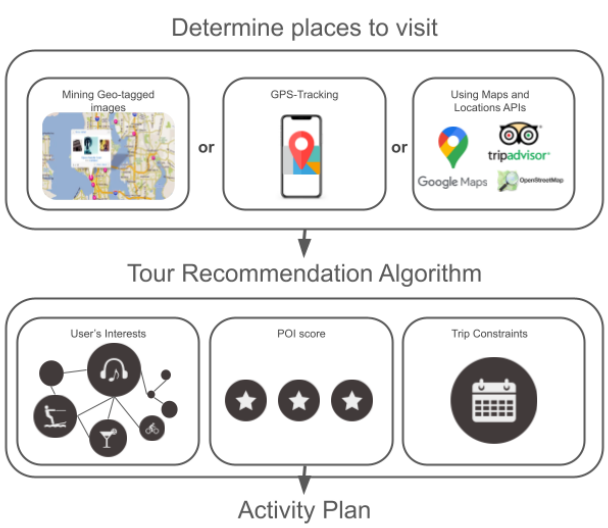

Figure 1: Recommender System Process in Tourism

1.2 Methods of retrieving travel products

Before producing an itinerary, RSs have to formulate a dataset of POIs from some data source. The proposed tour recommendation algorithm will then evaluate a guided path, route or itinerary from this dataset after understanding the users' implicit preferences such as the travel date and activity moderation. There are several ways to identify an appropriate data source representing real-life tourist trajectories.

Geotag mining: One approach is made by gathering tourist products by mining them from geotagged images of Location-Based Social Networks (LSBN) such as Flickr, Facebook or Twitter [5–14]. Lim et al. [4] denote this process into three steps;

1. First, the application assembles an organised series of relevant photographs of the user's destination from the LSBN.

2. The application then maps these pictures with a list of popular places extracted from sites like Wikipedia.

3. Since the photos contain metadata, like the location and the timestamps; the application can calculate an approximate visit duration for each specific POI.

GPS-based data sources: The ubiquitous presence of smartphones and GPS-enabled devices has facilitated another approach to collecting trajectories [15–17]. A system can automatically gather the best POIs to visit based on other users' historical paths providing additional information such as the average time people spend at a specific POI and how many people go there. However, privacy issues are the main caveat towards this approach since it requires people to share their location constantly and publically [4].

Prebuilt dataset: The most straightforward method is done by self-defining the POIs or gathering them from a dataset such as the TSPLIB95 [1]. Manually collecting travel products provides precision and a better understanding of the itinerary that the algorithm will generate. However, the algorithm would be dataset-specific testified and personalised towards what the authors of the dataset think are the best POIs to visit in a location [18–20].

[1]Sample instances for travelling salesman Problem: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

Maps APIs: A prompt and accurate strategy towards gathering essential places in the vicinity is using Mapping & Location APIs such as Foursquare, Google or TripAdvisor. Wörndl et al. [21] use this approach and build a dataset of prominent POIs by querying their API with the user's desired location. In return, they receive a sequence of places and information about each site, including its category, other user's ratings, opening hours, co-ordinates and helpful additional information to use as criteria for the itineraries. However, the API does not return the average amount of time people spend at a specific POI.Wörndl et al. [21] solve this issue by adding a fixed time constant for each category with a variable dependant on the POI's score. For example, suppose a restaurant's time constant is 45 minutes, and the chosen restaurant has a high score (based on its rating and user's characteristics). In that case, the time spent at the restaurant will increase by an additional 15 minutes. A significant advantage of using this approach is that the vast number of POIs that these endpoints return. According to Google's website, the API contains up to 200 million places with 25 million updates daily which an application can achieve with a few REST requests [22, 23].

1.3 User Profiling for Travel Preferences

In 2018, Lim et al. [8] demonstrated how implementing personalisation in their algorithm, which they called PersTours, helped their algorithm to portray real-life scenarios more accurately. The authors built a system where the tourist's level of interest in a specific category is dependant on their time spent at such POIs, relative to the average user. They gathered information from the user's past trips from the social media platform Flickr.

Nguyen et al. [24] produced an android chat application called STSGroup that gathers user's preferences and resolves conflicts between tourists by understanding the messages sent in a group chat. They provided an example of students travelling to South Tyrol (Italy), which gathered information such as the users' mood and recommended POIs from their conversations. Other users in the group chat rate their suggestions through a voting system as the system uses raking lists and logistics to calculate the ideal group preferences in the background.

The average internet user has gone from being a passive content absorber to a content producer through the rise in social media [25]. TTDP RSs can use this as an advantage and provide a fully automated activity plan based on the user's characteristics. This section describes several methods for user profiling and information gathering from the user's social media.

1.3.1 User Profiling based on natural language processing

In 2013, A sentiment analysis by Ikeda et al. [25] based on a hundred thousand Japanese user profiles managed to perform a demographic estimation. This study shows how social media posts can be helpful to gather information about a user, and in fact, Hung et al. [26] demonstrated a user profiling technique based on tag correlation.

1.3.2 User Profiling based on images

Instagram has a significant influence on the tourism industry. Sharing photos of amazing sights and landscapes have impacted the way people choose their POIs [27]. Therefore, a system that uses tourist's social media photos could impact automatic user preference gathering.

Chen et al. [28] produced a system for automatically retrieving tags from images and incomplete tags called *FastTag*. The algorithm can be trained in $O(n)$ time and uses two simple linear mappings. Figure 2 shows an example of an input image used alongside the incomplete input tags *snow, lake, feet*. Given these two inputs, the algorithm produced the following additional tags, mountain, water, legs, boat, trees.

Figure 2: Example of an image input for the FastTag algorithm

Images classification techniques could provide a preference gathering system. Deep neural networks play an essential role in this field of Image Classification [29, 30]. We will describe how this technology could gather information for a tourist's user preferences in the upcoming sections.

1.4 Travel Planners for both individual and grouped travellers

This section will focus on breaking down existing meta-heuristic approaches, notably swarm-based, trajectory-based and evolutionary algorithms, towards the vast number of variants of the TTDP. Gavalas et al. (Gavalas 2014) classify these variants into two; Systems that produce a single route and systems that can handle multiple days.

1.4.1 Single Route Problems

The Orienteering Problem (OP), introduced by Tsiligirdes [31], in observance of the sport, orienteering, is the foundation of single route planners. Figure 3 shows the variants of the OP that will be discussed in this section and how they relate with the original problem. TTDPs [1]. Vansteenwegen et al. [32] describe OP as a travelling salesman problem with profits. In OP, several nodes V representing POIs where $\{V = v_1, \ldots, v_n\}$, are designated in a space G and edge set E with a starting and an ending point. Each node holds a score s calculated from the tourist's constraints and the distance from node v_i to v_j. The objective is to visit a subset of these locations, maximising the s by obeying the tourist's constraints and minimising the travel time [33]. Vansteenwegen et al. [34] mathematically formulate the OP and the methodology contains a mathematical formulation of this instance of the TTDP.

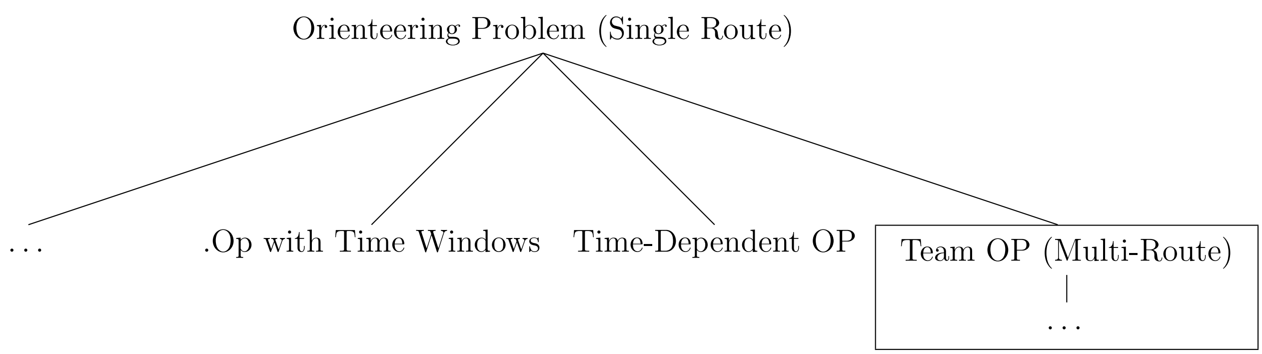

Figure 3: Graph showing the variants that will be discussed in the section

Particle Swarm Optimisation-based (PSO) systems provide prevalent OP solutions with fast computing time. These are bio-inspired meta-heuristic approaches in which, in the TTDP, a particle represents a travel path. The particles aim to optimise themselves by communicating with each other and using their velocity property to move to the most optimal solution [35]. In 2010, Sevkli et al. [36, 37] tested out two PSO variants: Strengthened Particle Swarm Optimization (StPSO) and Discrete Strengthened Particle Swarm Optimization (DStPSO). These two algorithms introduce pioneering particles, which first perform a local search-based technique called Reduce Variable Neighborhood Search (RVNS) between all the particles and then assign a random velocity. These PSO algorithms obtains either the best or competitive solutions compared with other algorithms such as ant colony and genetic algorithms when tested on the Tsiligirides [17, 31] dataset.

There are numerous Evolutionary Algorithms (EA) proposed to solve OP. [38, 39].

EAs are algorithms based on natural evolution which use a fitness score to get to the best solution of a problem, in this case, the TTDP [40]. A novel approach in 2018 by Kobeaga et al. [38] was able to find ambitious solutions for over 400 POI nodes using the steady-state genetic algorithm. The algorithm also implements a local search, which aims to reduce travel time.

In 2019, Santini et al. [41] introduced a heuristic algorithm based on adaptive extensive neighbourhood search. They evaluated their system by comparing it with Kobeage et al.'s EA. The results showed that both algorithms find suitable solutions in a reasonable amount of time. However, the EA finds slightly more suitable solutions, while the extensive neighbourhood search has a lower average gap.

In real-life scenarios, POIs have time constraints that allow them to be visited only during specific hours, such as opening and closing hours or public holiday constraints. Traditional OP is not able to cater for such problems. A single route variant of the OP which solves these issues is the Orienteering Problem with Time Windows (OPTW) [42].

Kantor et al. [43] provided the first attempt towards the OPTW [34]. They developed two heuristics; Insertion and depth-first search. The former algorithm solves the path by selecting a POI with the highest score over-insertion cost incrementally. On the other hand, the depth-first search algorithm gathers parallel tree-based solutions simultaneously and iteratively adds new POIs as long as they follow a set of constraints. Their evaluation showed significant improvements of the second algorithm over the insertion. Most of the novel solutions of OPTW are for the multiple route problems discussed in the upcoming sections.

When travelling between two POIs, the travel time may depend on certain variable time constraints such as the traffic levels and waiting time [1]. The Time-Dependent Orienteering Problem (TDOP) introduced by Fomin et al. [44] is the single route variant of OP, which considers these scenarios since traditional OP does not [40]. In 2011, Abbaspour et al. [45]provide a solution for the Time-Dependent Orienteering Problem with Time Windows, which combines the two previously mentioned OP variants (TDOPTW). They propose two adaptive genetic algorithms and multi-modal shortest pathfinding evaluated in the city of Tehran.

In 1998, Glover et al. [46] introduced a meta-heuristic approach called the Tabu Search, and several RSs used this algorithm [47–49]. This optimisation technique is advantageous when trying to escape from a local optimum [49]. A novel approach by Chou et al. [49] aims at tackling the Probabilistic Orienteering Problem (POP) [50], which is another variant in which every path contains a cost, and the system can access every node within a specific probability. Moreover, each node will be available for a visit only with a certain probability. When evaluated, a simple tabu search could compete with complex meta-heuristics showing its potential in this field.

1.4.2 Multiple Route Problems.

The RSs available from what we discussed in the previous sections can only generate a single efficient path for a tourist's holiday. The Team Orienteering Problem (TOP) [51] is a variant of the OP, which allows for solving the TTDP with multiple days [33]. The system generates a full itinerary for the tourist, with a maximum total score of all routes [1].

Several Recommender Systems use PSO-based solutions to solve the TOP [19, 52, 53] Muthuswamy et al. [52] developed a discrete version of the PSO (DPSO) which can generate n routes where n can be between two to four. The algorithm consists of two procedures; Random initialisation of n-1 routes with a calculated initialisation of the nth route based on partial randomness and the current score divided by the current distance of the particle. Updating the current velocity of each particle. The particles use RVNS and 2-opt techniques to communicate with each other as local search techniques. The authors evaluated their work by comparing the algorithm to seven TOP heuristics in which DPSO performed competitively across all applied benchmark data sets [42].

A few years later, Dang et al. wrote another PSO inspired algorithm (PSOiA) for the TOP. They evaluated their work using an interval graph model, which showed how to examine a more extensive search space faster [40].

Besides swarm-based algorithms, A RS by Sylejmani et al. [48] used the trajectory-based tabu search to solve a Multi Constrained Team OPTW. Their system followed three steps in order to generate an activity plan: a new activity is added as a node to the trip using *Insert*, a node is exchanged with a new activity using *Replace* and two nodes swap with each other using *Swap*.

Several RSs also use PSO-based solutions in novel approaches. For example, in 2019, Yu et al. [53] developed a system for the Team OPTW variant based on selective DPSO. In 2020, Wisittipanich [19] presented an application of a metaheuristic called Global Local and Near-Neighbour Particle Swarm Optimization (GLNPSO). Wisittipanich evaluated their results using LINGO, an optimisation program and showed excellent results.

Recently, Gama [54] et al. compared their reinforcement learning with top-performing heuristics of the TOP, such as Vansteenwegen's Iterated Local Searchl [55]. The authors use a Pointer network as this has been previously to solve TSP-related problems. This study opened a different way of tackling the TTDP and achieved production-level performance and inference times. An advantage of this approach is that the results are probabilistic. So it is possible to retrieve the top-n solutions and use them in a more generalised route recommendation system.

2 Methodology

This section will elaborate on collecting travel products, user-profiling methods, the itinerary generator and the implementation used to build this application. Figure 4 outlines the overall process of our personalised itinerary generation framework.

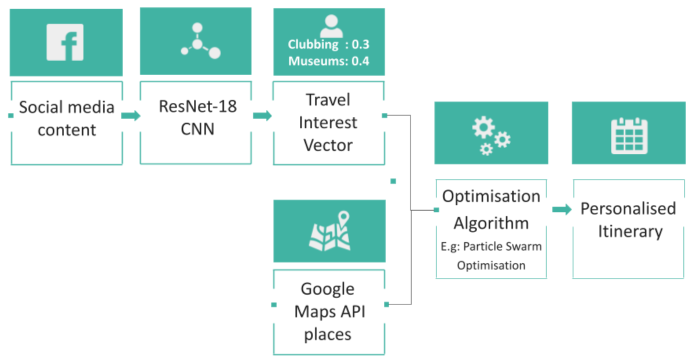

Figure 4: Personalised itinerary generator

2.1 Retrieving travel products

We implemented the Google Maps API as the data source for our application because of its real-time accuracy and massive dataset compared with the other approaches and other APIs that we discussed in the literature review [22, 23]. In addition, the nearby search endpoint allows the app to search for places of a given category within a specified area. In order to retrieve the places for the application, eight requests are made, each requesting places of different categories. To solve the issues with time windows, we split the endpoints into two categories. Five of the requests represent places shown as part of the itinerary during the day:

beaches, natural sights, museums, shopping malls and cafeterias.

and the rest represent places shown during the night:

nightclubs, bars and restaurants.

Figure **??** shows the query parameters used to gather cafeteria related places in Malta. These categories are based on the ones used by Wörndl et al. [21] for their application.

Table 1: Sample Query being made to the google maps nearby search endpoint

Key	**Value**
location	35.93575, 14.3754
radius	50000
type	cafe
keyword	must visit tourist

In return, the API returns a list of places of the specified area and category and attributes about each place. The attributes used by our application include the place's name, rating, the number of reviews and the coordinates. All of these attributes help the application further optimise the algorithm to find the perfect itinerary.

2.2 Generating the User Profile

Social media's effect on the world is something significant [56]. That is why this application builds a user profile from the user's social media.

The application built by Lim et al. [8] allowed the user to connect the application with their Flickr profile to scan their past trips. However, Facebook provides an API that would allow users to connect both their Facebook and Instagram accounts and request content from the user with their permission. A significant advantage is that the API allows the application not to limit the results to mimic only past user's trips like the application by Lim et al. [8] and gather preferences from his complete profile. The app requests two things from the potential tourist's social media, the photos and the liked pages and tries to classify these into six categories that make up the user's travel interest vector;

[1 Beach, 2 Nature, 3 Shopping, 4 Museums, 5 Clubbing, 6 Bars]

These categories are the same categories that we requested from the google maps API except 'cafeterias' and 'restaurants'. These two categories were not included because the application tries to suggest the best places to eat as part of the timetable, irrelevant to the user's profile. At the start of the application, the app initialises all vector values to zero and increments a value whenever the user's content matches a category. We will describe how the app classifies both the user's liked pages and the user's photos separately in the upcoming subsections.

2.2.1 Transforming the liked pages into the travel interest vector

The Facebook API allows the application to request each category of the user's liked Facebook pages. The API's documentation contains a whole list of possible categories.

The app iterates through all of these user's likes categories and increments a value in the user's vector whenever the Facebook result matches one of the six travel interest vector values. For example, if a user likes a page with class 'DJ', the user's clubbing vector value is incremented, and if a page is labelled as a 'Mountain', the app increments the user's nature vector value.

2.2.2 Transforming the user's photos into the travel interest vector

Convolutional Neural Networks have become a standard for classifying an image because of their high accuracy [57]. Therefore, we decided to test out two approaches for classifying the photos into the app's six categories.

Zhou et al. [57] trained several CNNs for scene recognition and generic deep scene features for visual identification. However, the places365 models are not explicitly trained on the six categories of our application. Therefore, we need to carefully map the 365 categories with our six application's categories. That is why we introduced a Tensorflow Keras sequential model, explicitly trained on the six application's categories to compare.

Pretrained Places365 models: These models are trained on the places365-standard dataset of about 1.8 million images to classify an image into 365 different scene categories. We used the Resnet places365 models, Resnet-18 and Resnet-50 since they achieved the highest top-5 validation accuracy on the places365 dataset. The Resnet 18 comprises 18, and the Resnet 50 comprises 50 convolutional layers. They both converge to an output layer representing the 365 output categories. Figure 7 shows a summary of the whole Resnet 18 model.

Trained Tensorflow Keras model: This model contains three convolutional layers with a rectified linear unit (ReLu) activation function. A pooling layer follows each to lower the input volume's spatial dimension for the upcoming layers. The final layer represents a flattening layer and two dense layers to reduce the outputs to the six application categories, and another representing the 'None' classification. Figure 6 shows a summary of the whole model.

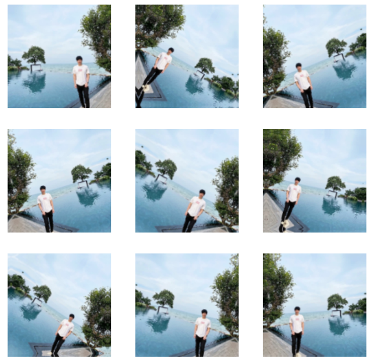

Figure 5: Data augmentation to reduce overfitting

The dataset contains 3600 public internet images representing the seven classes: Beach, Nature, Museums, Shopping, Clubbing and Bars and None. The tensor library provides tools to Split the dataset into a training and validation set Distribute the photos into batches of 32 Cache the dataset to memory to prevent I/O blocking All of the images were resized to 180x180 pixels, and the RGB values were normalised from zero to one. Since the dataset is small compared to the places 365 models, the training process is prone to overfitting. Data augmentation generates additional samples using random transformations on the dataset. Figure 5 shows an example of data augmentation on a photo. We also added a dropout layer to the model randomly drops sets the input values of the neuron. These two techniques help the model avoid overfitting.

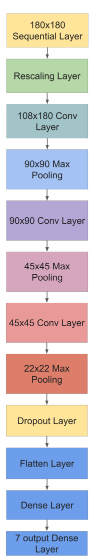

Figure 6: Keras Sequential Architecture Summary

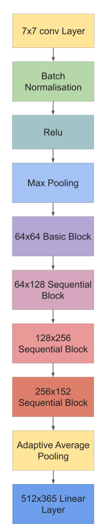

Figure 7: Resnet 18 Architecture Summary

2.3 Producing the activity plan

After the app generates the dataset of POIs and the user's travel interest vector, we formulate an efficient activity plan using these two inputs. This itinerary generator is based on the existing state of the art activity planners [19, 33] with some adjustments: We wanted the trip's output to take the form of an itinerary. The problem takes the form of the 'day' and 'night' category split discussed in the literature review. The scoring of itineraries is adjusted with the travel interest vector.

The problem definition of our novel itinerary planner is mathematically formulated as follows. A tourist trip is made up of some pre-defined user constants alongside the travel interest vector. The predefined constants are:

M:	The number of travelling days
C:	The moderation of activities (ie. the greater the C value, the more activities are generated in a day)

The objective function of our itinerary planner is:

$$\text{MAX} \sum_{m=0}^{M} (S_{D_m} + S_{E_m})$$

where:

m	Travelling day (m=1,2,..., textitM)
D_m	Morning section of day number m
E_m	Evening section of day number m
S_{D_m}	Score of the morning section D_m
S_{E_m}	Score of the evening section E_m

A day is made up of the morning D_m section and the evening E_m section. The morning section is made up of $C + 2$ tourist attractions whilst the evening section is just made up of 2.

$$D_m = Y_i + Y_f + C(Y_i)$$

$$E_m = Y_f + Y_j$$

i	Morning Tourist attraction (i = 1, 2, 3,. . . , n_1)
j	Evening Tourist attraction (j = 1,2, 3,. . . , n_2)
f	Food Place (f = 1,2, 3,. . . , n_3)
$Y_{i\|f\|j}$:	1 if a tourist visit attraction i, j or f and 0 if otherwise

Constraints

$\sum_{m=0}^{M}\sum_{i=0}^{n_1} Y_i \leq 1$	Ensures that all morning tourist attractions are not visited more than once throughout the whole itinerary
$\sum_{m=0}^{M}\sum_{j=0}^{n_1} Y_j \leq 1$	Ensures that all evening tourist attractions are not visited only once throughout the whole itinerary

2.3.1 Calculation of Score

The score S_{D_m} or S_{E_m} is calculated using

$$S_{D_m|E_m} = \frac{1}{T} + R + V$$

where:

T	Total distance between each tourist attractions in the morning/evening of day m
R	Average rating of the tourist attractions in the morning/evening of day m
V	how much the tourist attractions of the morning/evening of day m match with the user's travel interest vector

2.3.2 Particle Swarm optimisation algorithm

Kennedy et al. [58] proposed the original PSO algorithm in 1995 designed to solve optimisation problems. The algorithm is a population-based technique that uses n elements called particles. Each particle has a d-dimensional *position* vector representing a solution and a d-dimensional *velocity* vector expressing the direction of the particle during its search period.

When a PSO program initialises all of the particles, they are usually set to a random or predetermined value. In our algorithm, we introduce a method of randomisation bias. Although the initial particles are generated randomly, the randomness is weighted and affected by three things:

1. The user's travel interest vector
2. the place's rating
3. the place's number of ratings.

We implemented the randomness bias to give a head start to the algorithm rather than just starting optimising from purely random itineraries. Figure X shows an example of a sample place with its probability of being chosen as part of the initial particles alongside a sample tourist interest vector. At each iteration in the algorithm, the velocity of each particle is calculated based on the inertia constant and how well it is doing compared with its personal best score and the global best score. The inertia constant helps the particle explore new solutions and escape the local minima. After a few iterations have passed, particles use this velocity and move towards the optimum position. We demonstrate the framework of our PSO algorithm in Figure 8.

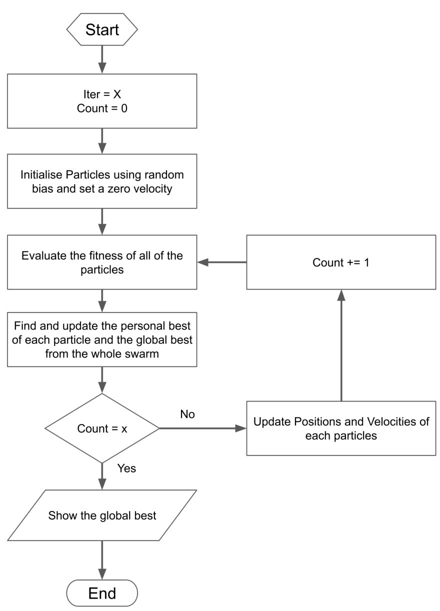

Figure 8: Framework of PSO algorithm

2.4 Web application implementation and user interface

We built the application using several technologies where each communicates with each other to provide a user-friendly website for the potential tourist. Figure 9 shows the tech stack diagram of the website. The website is accessible through the URL `https://www.touristplanner.xyz`. We built the front end of the website using HTML, CSS and javascript and hosted it on a cloud Vultr server. The website is responsive to be accessible from both a mobile phone and a laptop. The website communicates with the back end of the application using REST endpoints, hosted on a separate dedicated server provided by Hetzner using the Java Spring Boot framework. Another Python Gunicorn server is used to generate the itinerary and calculate a travel interest vector which sends the information directly to the Spring boot server. Finally, a local instance of an Open Source Routing Machine server calculates the distances from one tourist attraction to another used by the Gunicorn server to optimise the itinerary.

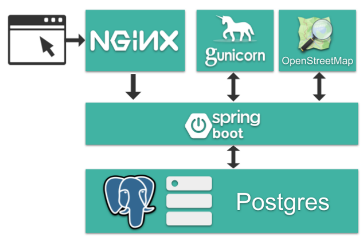

Figure 9: Tech Stack implementation of the application

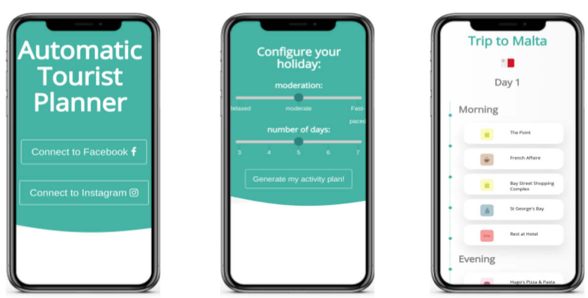

Figure 10: User Experience Timeline

Figure 10 shows screenshots of the website portraying the user's timeline. The user navigates to the homepage, accepts terms and conditions and connect his social media profiles. The user selects the number of days M and the activity moderation C. The website navigates to the final page of the application exhibiting their personalised itinerary.

3 Results and Evaluation

This section aims to position our study by describing the technologies we will use in this dissertation. In the first part, we provide an overview of the TTDP research area. We then describe methods of retrieving POIs from a location, existing user-profiling techniques and existing research regarding automatic tourist planning.

References

[1] D. A. Herzog, “A User-Centered Approach to Solving the Tourist Trip Design Problem for Individuals and Groups,” Tech. Rep., 2020.

[2] L. Santamaria-Granados, J. F. Mendoza-Moreno, and G. Ramirez-Gonzalez, “Tourist Recommender Systems Based on Emotion Recognition—A Scientometric Review,” *Future Internet*, vol. 13, no. 1, p. 2, dec 2020. [Online]. Available: https://www.mdpi.com/1999-5903/13/1/2

[3] P. Di Bitonto, F. Di Tria, M. Laterza, T. Roselli, V. Rossano, and F. Tangorra, “Automated generation of itineraries in recommender systems for tourism,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6385 LNCS. Springer, Berlin, Heidelberg, 2010, pp. 498–508. [Online]. Available: www.expedia.com

[4] K. H. Lim, J. Chan, S. Karunasekera, and C. Leckie, “Tour recommendation and trip planning using location-based social media: a survey,” *Knowledge and Information Systems*, vol. 60, no. 3, pp. 1247–1275, dec 2018. [Online]. Available: https://link.springer.com/article/10.1007/s10115-018-1297-4

[5] M. De Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu, “Automatic construction of travel itineraries using social breadcrumbs,” in *HT’10 - Proceedings of the 21st ACM Conference on Hypertext and Hypermedia*, 2010, pp. 35–44. [Online]. Available: http://www.munmund.net/pubs/ht{_}10{_}long.pdf

[6] I. Memon, L. Chen, A. Majid, M. Lv, I. Hussain, and G. Chen, “Travel Recommendation Using Geo-tagged Photos in Social Media for Tourist,” vol. 80, pp. 1347–1362, 2015.

[7] C. Lucchese, R. Perego, F. Silvestri, H. Vahabi, and R. Venturini, “How Random Walks can Help Tourism,” 2012. [Online]. Available: http://www.flickr.com

[8] K. H. Lim, J. Chan, C. Leckie, and S. Karunasekera, “Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency,” *Knowledge and Information Systems*, vol. 54, no. 2, pp. 375–406, feb 2018. [Online]. Available: https://doi.org/10.1007/s10115-017-1056-y

[9] K. Hui Lim, J. Chan, C. Leckie, and S. Karunasekera, “Personalized Tour Recommendation based on User Interests and Points of Interest Visit Durations,” Tech. Rep., 2015.

[10] K. Hui Lim, S. Karunasekera, C. Leckie, and J. Chan, “Recommending Tours and Places-of-Interest based on User Interests from Geo-tagged Photos.” [Online]. Available: http://dx.doi.org/10.1145/2744680.2744693.

[11] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura, “Travel route recommendation using geotagged photos,” *Knowledge and Information Systems*, vol. 37, no. 1, pp. 37–60, oct 2013. [Online]. Available: https://link.springer.com/article/10.1007/s10115-012-0580-z

[12] T. Kurashima and K. Fujimura, *Travel Route Recommendation Using Geotags in Photo Sharing Sites*, 2010.

[13] I. Brilhante, J. Antonio Macedo, F. Maria Nardini, R. Perego, and C. Renso, “Where Shall We Go Today? Planning Touristic Tours with TripBuilder,” 2013. [Online]. Available: http://dx.doi.org/10.1145/2505515.2505643.

[14] I. R. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, and C. Renso, “On planning sightseeing tours with TripBuilder,” *Information Processing and Management*, vol. 51, no. 2, pp. 1–15, mar 2015. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0306457314000922

[15] Y. Zheng and X. Xie, “Learning Travel Recommendations from User-Generated GPS Traces,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 1, 2011. [Online]. Available: https://doi.org/10.1145/1889681.1889683

[16] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, “Mining Interesting Locations and Travel Sequences from GPS Trajectories,” in *Proceedings of the 18th International*

Conference on World Wide Web, ser. WWW '09. New York, NY, USA: Association for Computing Machinery, 2009, pp. 791–800. [Online]. Available: https://doi.org/10.1145/1526709.1526816

[17] Z. Chen, H. T. Shen, and X. Zhou, "Discovering popular routes from trajectories," in *Proceedings - International Conference on Data Engineering*, 2011, pp. 900–911.

[18] X. Chou, L. M. Gambardella, and R. Montemanni, "A Tabu Search algorithm for the Probabilistic Orienteering Problem," *Computers and Operations Research*, vol. 126, p. 105107, feb 2021.

[19] W. Wisittipanich and C. Boonya, "Multi-objective Tourist Trip Design Problem in Chiang Mai City," in *IOP Conference Series: Materials Science and Engineering*, vol. 895, no. 1. Institute of Physics Publishing, jul 2020, p. 012014. [Online]. Available: https://iopscience.iop.org/article/10.1088/1757-899X/895/1/012014https://iopscience.iop.org/article/10.1088/1757-899X/895/1/012014/meta

[20] E. Erbil and W. Wörndl, "Generating Multi-Day Round Trip Itineraries for Tourists," Tech. Rep., 2021.

[21] W. Wörndl, A. Hefele, and D. Herzog, "Recommending a sequence of interesting places for tourist trips," *Information Technology and Tourism*, vol. 17, no. 1, pp. 31–54, mar 2017. [Online]. Available: http://link.springer.com/10.1007/s40558-017-0076-5

[22] H. Iltifat, "Generation of paths through discovered places based on a recommender system," Ph.D. dissertation, Master's Thesis, Department of Computer Science, Technical University of˜. . . , 2014.

[23] "Geo-location APIs — Google Maps Platform — Google Cloud." [Online]. Available: https://cloud.google.com/maps-platform

[24] T. N. Nguyen and F. Ricci, "A chat-based group recommender system for tourism," *Information Technology and Tourism*, vol. 18, no. 1-4, pp. 5–28, apr 2018. [Online]. Available: https://doi.org/10.1007/s40558-017-0099-y

[25] K. Ikeda, G. Hattori, C. Ono, H. Asoh, and T. Higashino, "Twitter user profiling based on text and community mining for market analysis," *Knowledge-Based Systems*, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.knosys.2013.06.020

[26] C.-C. Hung, Y.-C. Huang, J. Yung-jen Hsu, and D. Kuan-Chun Wu, “Tag-Based User Profiling for Social Media Recommendation,” Tech. Rep., 2008. [Online]. Available: http://www.flickr.com/

[27] A. Terttunen, “The influence of Instagram on consumers’ travel plan-ning and destination choice,” Tech. Rep., 2017. [Online]. Available: http://www.theseus.fi/handle/10024/129932

[28] M. Chen, A. Zheng, and K. Q. Weinberger, “Fast Image Tagging,” Tech. Rep., 2013. [Online]. Available: http://tinyurl.com/9jfs7ut

[29] K. Balaji and K. Lavanya, “Medical Image Analysis With Deep Neural Networks,” in *Deep Learning and Parallel Computing Environment for Bioengineering Systems*. Elsevier, jan 2019, pp. 75–97.

[30] A. Cufoglu, “User Profiling-A Short Review,” Tech. Rep. 3.

[31] T. Tsiligirides, “Heuristic methods applied to orienteering,” *Journal of the Operational Research Society*, vol. 35, no. 9, pp. 797–809, sep 1984. [Online]. Available: https://www.tandfonline.com/doi/full/10.1057/jors.1984.162

[32] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, “The orienteering problem: A survey,” pp. 1–10, feb 2011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0377221710002973

[33] K. Sylejmani, J. Dorn, and N. Musliu, “Planning the trip itinerary for tourist groups,” *Information Technology and Tourism*, vol. 17, no. 3, pp. 275–314, sep 2017. [Online]. Available: https://link.springer.com/article/10.1007/s40558-017-0080-9

[34] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, “The orienteering problem: A survey,” pp. 1–10, feb 2011.

[35] A. Rezaee Jordehi and J. Jasni, “Parameter selection in particle swarm optimisation: A survey,” *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 25, no. 4, pp. 527–542, dec 2013. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/0952813X.2013.782348

[36] A. Z. Şevkli and F. E. Sevilgen, “StPSO: Strengthened particle swarm optimization,” *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 18, no. 6, pp. 1095–1114, nov 2010.

[37] Z. Sevkli and F. E. Sevilgen, "Discrete particle swarm optimization for the orienteering problem," in *2010 IEEE World Congress on Computational Intelligence, WCCI 2010 - 2010 IEEE Congress on Evolutionary Computation, CEC 2010*. IEEE, jul 2010, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/document/5586532/

[38] G. Kobeaga, M. Merino, and J. A. Lozano, "An efficient evolutionary algorithm for the orienteering problem," *Computers and Operations Research*, vol. 90, pp. 42–59, feb 2018. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0305054817302241

[39] X. Wang, B. L. Golden, and E. A. Wasil, "Using a genetic algorithm to solve the generalized orienteering problem," *Operations Research/ Computer Science Interfaces Series*, vol. 43, pp. 263–273, 2008.

[40] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering Problem: A survey of recent variants, solution approaches and applications," pp. 315–332, dec 2016.

[41] A. Santini, "An adaptive large neighbourhood search algorithm for the orienteering problem," *Expert Systems with Applications*, vol. 123, pp. 154–167, jun 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417418308182

[42] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou, "A survey on algorithmic approaches for solving tourist trip design problems," *Journal of Heuristics*, vol. 20, no. 3, pp. 291–328, jun 2014. [Online]. Available: http://link.springer.com/10.1007/s10732-014-9242-5

[43] M. G. Kantor and M. B. Rosenwein, "The orienteering problem with time windows," *Journal of the Operational Research Society*, vol. 43, no. 6, pp. 629–635, 1992.

[44] F. V. Fomin and A. Lingas, "Approximation algorithms for time-dependent orienteering," *Information Processing Letters*, vol. 83, no. 2, pp. 57–62, jul 2002.

[45] R. A. Abbaspour and F. Samadzadegan, "Time-dependent personal tour planning and scheduling in metropolises," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12 439–12 452, sep 2011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417411005410

[46] F. Glover and M. Laguna, "Tabu Search," in *Handbook of Combinatorial Optimization*. Boston, MA: Springer US, 1998, pp. 2093–2229. [Online]. Available: http://link.springer.com/10.1007/978-1-4613-0303-9{_}33

[47] H. Tang and E. Miller-Hooks, "A TABU search heuristic for the team orienteering problem," *Computers and Operations Research*, vol. 32, no. 6, pp. 1379–1407, jun 2005. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0305054803003265

[48] K. Sylejmani, J. Dorn, and N. Musliu, "A Tabu Search approach for Multi Constrained Team Orienteering Problem and its application in touristic trip planning," in *Proceedings of the 2012 12th International Conference on Hybrid Intelligent Systems, HIS 2012*, 2012, pp. 300–305.

[49] X. Chou, L. M. Gambardella, and R. Montemanni, "A Tabu Search algorithm for the Probabilistic Orienteering Problem," *Computers and Operations Research*, vol. 126, p. 105107, feb 2021.

[50] E. Angelelli, C. Archetti, C. Filippi, and M. Vindigni, "The probabilistic orienteering problem," *Computers and Operations Research*, vol. 81, pp. 269–281, may 2017. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0305054816303331

[51] I. M. Chao, B. L. Golden, and E. A. Wasil, "The team orienteering problem," *European Journal of Operational Research*, vol. 88, no. 3, pp. 464–474, feb 1996.

[52] S. Muthuswamy and S. S. Lam, "Discrete particle swarm optimization for the team orienteering problem," *Memetic Computing*, vol. 3, no. 4, pp. 287–303, dec 2011. [Online]. Available: http://link.springer.com/10.1007/s12293-011-0071-x

[53] V. F. Yu, P. A. A. N. Redi, P. Jewpanya, A. Gunawan, V. F. . Yu, P. A. A. N. . Redi, P. . Jewpanya, V. F. Yua, A. A. N. Perwira Redia, and P. Jewpanyaa, "Selective discrete particle swarm optimization for the team Selective discrete particle swarm optimization for the team orienteering problem with time windows and partial scores orienteering problem with time windows and partial scores Citation Citation S," Tech. Rep., 2019. [Online]. Available: https://ink.library.smu.edu.sg/sis{_}research/4469

[54] R. Gama and H. L. Fernandes, "A REINFORCEMENT LEARNING APPROACH TO THE ORIENTEERING PROBLEM WITH TIME WINDOWS A PREPRINT," Tech. Rep., 2020.

[55] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden, "Iterated local search for the team orienteering problem with time windows," *Computers and Operations Research*, vol. 36, no. 12, pp. 3281–3290, dec 2009.

[56] D. Miller, J. Sinanan, X. Wang, T. McDonald, N. Haynes, E. Costa, J. Spyer, S. Venkatraman, and R. Nicolescu, *How the World Changed Social Media*. UCL Press, feb 2016. [Online]. Available: www.ucl.ac.uk/ucl-press

[57] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 Million Image Database for Scene Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1452–1464, jun 2018.

[58] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4. IEEE, pp. 1942–1948. [Online]. Available: http://ieeexplore.ieee.org/document/488968/