ORIGINAL ARTICLE

International Journal of Applied Mathematics in Control Engineering

Journal homepage: http://www.ijamce.com

Tourism Route Optimization Recommendation Based on Big Prediction Model

Haiqing Li[1] | Shanghui Chang[1] | Haowei Liu[1] | Zijun Chen[1] | Xupeng Wang[1]*

[1]Department of Software Engineering, Qingdao University of Technology, Qingdao, Shandong 266520, China

Correspondence
Department of Software Engineering, Qingdao University of Technology, Qingdao, Shandong 266520, China
Email: wxp1130@yeah.net

Article Info
Article history:
Received 14 January 2025
Accepted 20 March 2025
Available online 21 March 2025

Abstract

With the continuous development of the economy, tourism has gradually become a basic necessity in people's lives. However, existing tourism recommendations typically focus on pushing scenario spots, without providing assistance for intelligent and personalized tour route planning or specific location details of the attractions. Therefore, this paper aims to enhance the tourism experience and meet tourists' personalized needs. Based on in-depth research of official introductions and reviews of tourist destinations, the large language model is used for analysis to generate a dictionary of relevant attractions, which is then stored in a knowledge graph. Additionally, tourists' preferences and demands are collected and analyzed for precision tour route planning. Further, through weight analysis, the recommendation index of each attraction is continuously updated, and, based on tourists' preferences, the large language model and natural language processing techniques are used to filter the corresponding attention information, select suitable transportation means, and identify attention types, providing personalized services for the tourists. Finally, a dynamic programming algorithm is used to automatically generate tour routes. The sightseeing sequence can be adjusted based on tourists' needs, and an optical route planning map is generated.

KEYWORDS
Large Language Models, Natural Language Processing Techniques, Scenic Spot Knowledge Graph, Tour Route Optimization

1 | INTRODUCTION

At present, travel itineraries need to be planned in advance. People often prepare based on online articles, blogs, and short videos. This is not only time-consuming and labor-intensive, but also the online recommendations are mixed, and are different from people's experience on the ground . At present, a convenient and fast travel route schedule planning solution is needed to meet people's needs for worry-free travel.

Digital Object Identifiers: 10.62953/IJAMCE.147407

Based on tourists' preferences, schedules and other demands, this article uses large language models and knowledge theories of attraction dictionaries and knowledge graphs to ensure that tourists can fully understand and enjoy the charm of tourist destinations. At the same time, we collect and analyze tourists' needs and preferences, including their travel time, budget, interests and hobbies, etc., to carry out accurate travel route planning. In addition, through weight analysis, the recommendation index of attractions is constantly updated, thereby making real-time adjustments to ensure that each route is generated optimally. Fully consider all aspects such as transportation, travel time, rest space, etc., and use genetic algorithms to simultaneously ensure the recommendation index and maximize the optimal route within the time specified by the tourists, ensuring the comfort and convenience of tourists during the travel process. It uses large language models and natural language processing technology to filter out corresponding scenic spot information, select appropriate means of transportation and types of scenic spots, and provide personalized services to tourists. Through page interaction, it is ensured that tourists can choose tourist attractions independently, automatically generate travel routes and make corresponding changes according to tourists' needs. Finally , the route is optimized through the dynamic planning algorithm, the order of sightseeing is re-adjusted, and the optimal route planning map is generated, truly intelligent and professional, meeting the needs of tourists while reducing tourists' itinerary preparation work, saving worry, time and effort . understanding bedform dynamics is critical in predicting morphodynamics.

2 | RESEARCH STATUS AT HOME AND ABROAD

So far, many foreign research institutions have tried to develop specialized tourism ontology [1]. However, in the actual recommendation process, there is still insufficient consideration of cultural differences. This may lead to recommended travel routes that are not in line with tourists' cultural background and interests, affecting tourists' travel experience.

Domestic travel route recommendations rely more on data analysis, the application effect of recommendation algorithms [2] in specific scenarios, etc., such as tourists' search history, purchase records, and evaluations. However, some studies may rely too much on data analysis and ignore the personalized needs of tourists; some tourist routes may lack innovation and characteristics, making it difficult to attract tourists' interest.

3 | GENERATE SCENIC SPOT KNOWLEDGE GRAPH

For the generation of the scenic spot knowledge map, first use the context learning method to train a large model [3, 4, 5] and generate all the scenic spot dictionaries in the city based on the scenic spot dictionary template. this task trains the model by maximizing conditional probability (1), the model generates the next word by each given partial sequence (2), the output of each position is weighted based on the information of all positions (3).

$$p(r_1,r_2,...,r_m) = \prod_{m=1}^{M} p(r_m|r_1,r_2,...,r_{m-1}) \tag{1}$$

$$\begin{gathered} p(r_m p(r|\hat{r_1},r_2,...,r_{m-1M-1},...,r) = Soft\max \\ (f(r_1 SoftMax(f(R,r_2,...,r_{m-1M-1},...,r)) \end{gathered} \tag{2}$$

$$Attention(M,N,T) = Soft\max(\frac{MN^Q}{\sqrt{d_k}})T \tag{3}$$

r_1 represents the word at time step m; M is the total number of words in the sentence; p_w is the conditional probability given the previous words, representing the model's predicted probability for word r_m; f is a vector computed based on the current context, and the prediction probability for each word is obtained through the Softmax function;In the self-attention formula (3), M is the query matrix, N is the key matrix, and T is the value matrix.

Since the Transformer model does not have built-in sequence information, positional information for each position in the sequence needs to be injected through position encoding (4). Trains a model on a large amount of text data and uses self-supervised learning tasks (such as language model tasks) to learn common language representations (5). Use the vector database [6, 7] to store all the scenic spot dictionaries, and use the offline map API to obtain all scenic spots.

$$\begin{aligned} PE_{(pos,2i)} &= \sin(\frac{pos}{10000^{2i/s}}) \\ PE_{(pos,2i+1)} &= \cos(\frac{pos}{10000^{2i/s}}) \end{aligned} \tag{4}$$

$$Z_{fine-true} = -\sum_{i=1}^{N} \log P(y_i|X_i) \tag{5}$$

pos is the position index, *i* is the dimension index, and *s* is the encoding dimension; The loss function (5) is the cross-entropy loss, which measures the difference between the model's output and the true label.

3.1 | Generate attractions dictionary

In-Context Learning, as a special form of prompts, has become one of the typical methods to utilize LLM. ICL uses formatted natural language prompts, selecting some examples from a task dataset as demonstrations. LLM can identify and execute new tasks without explicit gradient updates. The prediction of the output generated from LLM can be formulated as follows:

$$LIM(I, f(x1,y1),...,f(x_k,x_k), f(x_{k+1},z)) -> y_{k+1} \tag{6}$$

The large language model can be used to the generate each scenic spot dictionary [8, 9, 10] according to the scenic spot dictionary template. In addition, Chain-of-Thought [11, 12] prompts can also be used to enhance learning by incorporating a series of intermediate reasoning steps into prompts. After all generation is completed, use the MySQL database to store all the scenic spot dictionaries, and introduce the city's unique identification ID , linking attractions to cities. As shown in Figure 1:

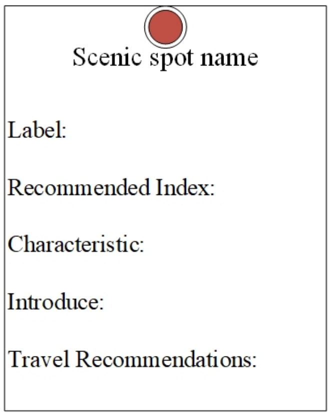


FIGURE 1 Scenic Spot Dictionary Diagram.

3.2 | Offline map API

The offline map API refers to the interface that provides offline map data and functions. Through these APIs, developers can integrate offline maps into their applications, allowing users to access map data even without a network connection.

Loop the above steps to continuously generate a dictionary of all attractions in the city. At the same time, use the above-mentioned offline map API to obtain the best public transportation mode (7) and required time (8) between all attractions, and use SQL to store it in the database table. Each row Store two scenic spots, travel methods and time, and introduce the city's unique identification ID.

$$d(m) = \min(d(m), d(n) + w(n,m))$$
$$d_{ij}^{r} = \min(d_{ij}^{(r-1)}, d_{ir}^{(r-1)} + d_{rj}^{(r-1)}) \tag{7}$$

$$T_{ij} = TravelTime(v_i, v_j, \bmod e)$$
$$T_{ij} = \min(T_{ij}^{1}, T_{ij}^{2}, ..., T_{ij}^{r}) \tag{8}$$

$d(m)$ is the shortest distance from the source node to node m, $w(n,m)$ is the weight of the edge from node nnn to node m, and d_{ij}^{r} represents the shortest distance from node i to node j after considering the first r nodes; $TravelTime(v_i, v_j, \bmod e)$ is the travel time from scenic spot v_i to scenic spot v_j.

3.3 | Generate scenic spot knowledge map

Knowledge graph [13] (shown in Figure 2) is a structured form of knowledge expression that graphically organizes and stores a large number of entities (such as people, places, events, etc.) and their interrelationships. In the knowledge graph, entities serve as nodes, and various semantic associations between entities are connected through edges, forming a huge data network. The core value of the knowledge graph lies in its ability to accurately and intuitively represent knowledge in a complex world and support efficient knowledge query and reasoning. As shown in Figure 2:

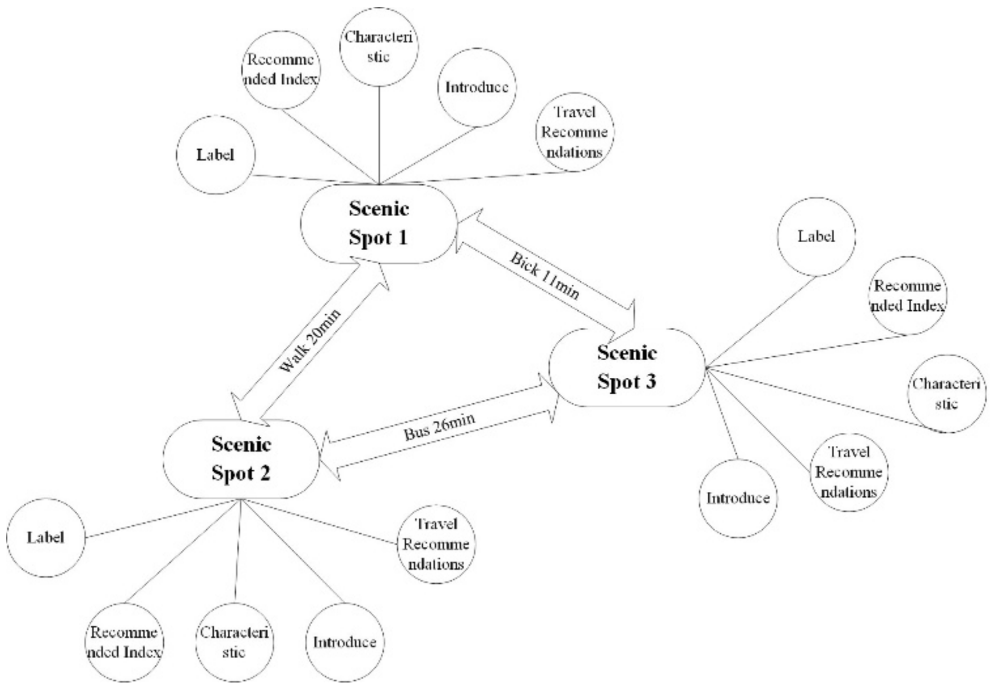


FIGURE 2 Scenic Spot Knowledge Graph Diagram.

3.3.1 | Centrality algorithm

To generate a knowledge graph using a dictionary of scenic spots, first use the centrality algorithm (9-10) to evaluate the importance of scenic spots by calculating the number of edges (hereinafter referred to as edges) generated by distance, travel mode and provide future knowledge Establish the basis for the generation of graphs.

$$\begin{aligned} w_{ij} &= \frac{1}{r_{ij} \cdot t_{ij} \cdot p_{ij}} \\ C_d(i) &= \sum\nolimits_{j \in M(i)} w_{ij} \\ C_c(i) &= \frac{1}{\sum_{j \neq i} r_{ij}} \end{aligned} \tag{9}$$

$$\begin{aligned} C_p(i) &= \sum\nolimits_{s \neq i \neq t} \frac{\sigma_{st}(i)}{\sigma_{st}} \\ C(i) &= \alpha \cdot C_d(i) + \beta \cdot C_c(i) + \gamma \cdot C_P(i) \end{aligned} \tag{10}$$

w_{ij} represents the edge weight, r_{ij} denotes the distance between attraction i and j, t_{ij} is the travel time from attraction i to j, and p_{ij} is a coefficient related to the mode of transportation. C_d stands for centrality degree, with M_i being the set of attractions directly connected to attraction i. Closeness centrality is denoted as C_c, where r_{ij} (alternatively, d_{ij} for clarity) represents the shortest path length from attraction i to j. Betweenness centrality is denoted as C_p, with σ_{st} representing the number of all shortest paths from attraction s to t, and $\sigma_{st}(i)$being the number of those shortest paths that pass through attraction i. Comprehensive centrality is denoted as C_i, which is a weighted combination of different centrality measures with α, β, and γ as the weight coefficients.

3.3.2 | Recursive algorithm to generate knowledge graph

After determining the importance of the scenic spots, a recursive algorithm (11) is used to establish a connection between the more important scenic spots and the scenic spots associated with them. Then, the more important scenic spots are retrieved (12) among the connected scenic spots and continue the recursion. At the same time, find the scenic spots that appear repeatedly in the map and delete them (13). Finally, the information of each scenic spot is retrieved from the database and added to the graph to generate a preliminary scenic spot knowledge graph.

$$\begin{aligned} &connect(r_1, r_2) = 1 \\ &dis\tan ce(r_1, r_2) \leq \mathrm{t} and reach(r_1, r_2) \\ &connect(r_1, r_2) = 0 otherwise \end{aligned} \tag{11}$$

$$S_c(r) = S_c(r) \cup \{c(r, r_i) = 1 and r_i \notin S_c\} \tag{12}$$

$$S_v = S_v \cup \{r_i | r_i \notin S_v\} \tag{13}$$

$connetc(r_1, r_2)$ indicates whether there is a connection between scenic spot r_1 and scenic spot r_2, distance represents the travel distance between the two scenic spots, reach indicates whether there is a public connection, t is the maximum acceptable travel time threshold; S_c is the set of scenic spots that have already been connected, r_i is another scenic spot connected to scenic spot r, and the process proceeds recursively until no more new scenic spots are added; S_v is the set of scenic spots that have been added to the knowledge graph.

3.3.3 | Use PRA algorithm to improve the scenic spot knowledge graph

The PRA algorithm [14, 15] (Path Ranking Algorithm) is used to generate a set of path features (14) using the information in the database through random walks, and then the feature values are calculated (15) to generate a classifier to further determine the specific connections (16) between each connecting attraction. When generating the attractions knowledge map, priority is given to the connections between the same type of attractions, so as to optimize and improve the function of the attractions knowledge map.

$$w = (r_1, r_2, ..., r_n) \tag{14}$$

$$\min_{\theta} \sum_{i=1}^{N} L(s_i, f(F(Y_i), \theta)) + \lambda ||\theta||^2 \tag{15}$$

$$Similarity(r_i, r_j) = \cos(t(r_i), t(r_j)) \tag{16}$$

r_i is the node of the i step of the random walk, n is the length of the path; N is the number of paths, s_i is the label of path Y_i (whether a certain relationship between attractions holds), L is the loss function, f is the output of the classifier, based on path features F and parameters θ; t is the path feature vector of attraction r.

4 | DATA UPDATE

This article combines the Larimar large language model [16] with traditional NLP natural language processing technology [17] to mine and analyze user reviews more efficiently, continuously generate new recommendation indexes, and then provide personalized and dynamic Recommended tourist attractions. Its structure is shown in Figure 3:

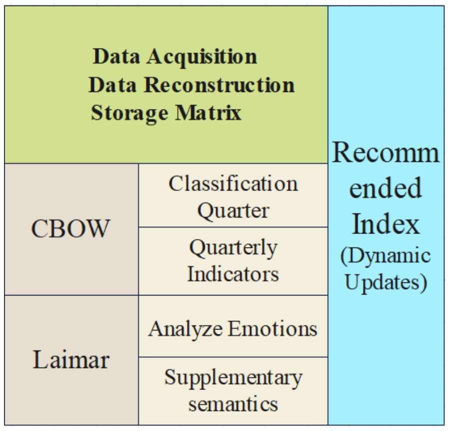


FIGURE 3 Scenic Spot Knowledge Graph Diagram.

4.1 | Data preprocessing

The column indicators are quarterly data and total indicator data. Then perform data cleaning, such as denoising , language recognition: use PaddleOCR [18] to detect the evaluation text, and classify the language through PaddleClass [19], text standardization: perform word segmentation, remove stop words and semantic completion of the evaluation content (Larimar supports). Finally, the timestamps are standardized and the evaluation timestamps are uniformly converted into quarterly labels. As shown in Table 1:

TABLE 1 Quarterly Label.

q1	q2	q3	q4
January–March	April–June	July–September	October–December

4.2 | Data classification and analysis

This article classifies user reviews by quarter and uses them for quarterly forecasts and user travel time preference forecasts [20].

First, input and evaluate the text context (17); then convert labels and target quarters (18) and conduct data training and construction based on evaluation content and generated training data (19).

$$P(t) = [r_1, r_2, ..., r_n] \tag{17}$$

$$q = \arg\max(Model(t)) \tag{18}$$

$$W = -\sum_{i=1}^{m} [q_i \log(\hat{y_i}) + (1 - q_i) \log(1 - \hat{y_i})] \tag{19}$$

r_i is the feature vector of the word; $\operatorname{mod} el(t)$ is a classification model; W is the loss function to minimize; $\hat{y_i}$ is the predicted probability for the i text t_i from the model.

Secondly, predict the evaluation of the new quarter (20) and then output the model; at the same time, use user evaluation history language to conduct model Training (21), extract keywords (such as “spring”, “autumn”, etc.) as time signals; finally output the user’s most likely travel time(22) .

$$f_{new} = \arg\max(Model(t_{new})) \tag{20}$$

$$S_i = keywords(t_i) \tag{21}$$

$$\begin{aligned} &Z(f_{new} = spring) = z_{spring} \\ &Z(f_{new} = summer) = z_{summer}, \cdots \\ &\hat{y} = \arg\max(Z(f_{new})) \end{aligned} \tag{22}$$

S_i is the user feedback for each review; Z is the probability distribution for the target quarter, and y is the user’s most likely travel time.

4.3 | Evaluation index and recommendation index generation

Indicator calculation: Based on user preferences and relevant industry research experience, the indicator weights are set as shown in Table 2:

TABLE 2 Indicator Weight.

Index	User ratings	Transportation convenience	Cultural heritage	Beautiful scenery	Food experience
Weight	50%	20%	10%	10%	10%

Index score calculation: First use Semantic Keyword Extraction(SKE) [21] to extract keywords related to the index (such as “convenient" , “rapid" , “difficult", “slow", etc.) . Then divide the positive keywords and negative keywords (punishment keywords), in which a positive keyword will add 1 point every 100 times it appears, up to 100 points; a negative keyword will deduct 1 point every 100 times it appears, and the minimum will be 0 points . Then the users are rated and ranked , and after converting the user ratings into a percentage system, the weighted contract of all reviews is obtained.

Quarterly recommendation index : For each quarter’s data , the following calculation is performed:

$$\mathrm{t} = 0.5 * a + 0.2 * b + 0.1 * c + 0.1 * d + 0.1 * e \tag{23}$$

Among them, t represents the quarterly index, a represents the score, b represents transportation, c represents culture, d represents landscape, and e represents food.

Comprehensive recommendation index: After calculating the quarterly recommendation index, and then based on the quarterly weight, the comprehensive recommendation index calculation formula is as follows:

$$\mathrm{sum} = 0.55 * f + \sum_{\mathrm{i}} \sum 0.15 * \mathrm{t} \tag{24}$$

sum represents the comprehensive recommendation index, f represents the target quarter index, i represents other quarters, and t represents the quarterly index.

4.4 | Introduction of Larimar large language model

The Larimar model can accurately capture the emotional relationships (25) (implicit positive emotions, negative emotions) in user reviews.

$$\begin{aligned} f_{positive} &= \sum_{r_i \in R} m_{pos}(r_i) \\ f_{negative} &= \sum_{r_i \in R} m_{neg}(r_i) \\ f_{emotion} &= f_{positive} - f_{negative} \end{aligned} \tag{25}$$

m_{pos} is the weight of word r_i in the positive sentiment lexicon, m_{neg} is the weight in the negative sentiment lexicon, $f_{emotion}$ is the difference between the positive and negative sentiment scores.

Semantic completion function for some incomplete evaluations, Larimar can perform contextual supplementation. And intelligent question and answer can provide personalized travel suggestions through Larimar.

Finally, obtain results (27-28) by supplementing SKE (26).

$$f_{emotion\_ske} = \sum_{x_i \in X} m_{emotion}(x_i) \tag{26}$$

$$f_{final} = \alpha \cdot f_{emotion} + \beta \cdot f_{emotion\_ske} \tag{27}$$

$$\begin{aligned} Emotion &= Positive f_{final} > positive \\ Emotion &= Negative f_{final} > negative \\ Emotion &= Neutral f_{final} > Neutral \end{aligned} \tag{28}$$

f_{final} is the final sentiment score.

At the same time, whenever new evaluation data is input , Optical Character Recognition(OCR) [22, 23] is used to identify (29) and extract the text content , and then the keyword statistics in the storage matrix are updated (30) and the recommendation index (31) is recalculated quarterly and stored in the database (32) .

$$t_{new} = OCR(image) \tag{29}$$

$$\begin{aligned} k_{new} &= Ske(t_{new}) \\ w(k,m) &= w(k,m) + p_k \end{aligned} \tag{30}$$

$$f_q = \sum_k w(k,q) \cdot p_k \tag{31}$$

$$DB(quarter, f_q) = f_q \tag{32}$$

t_{new} is the extracted text; k_{new} is the keyword extracted from the text; $w(k,m)$ is the keyword frequency; m is the index for each quarter; k is the keyword index; f_q is the weighted sum of all keyword weights in quarter q; $w(k,q)$ is the frequency of the keyword k; and DB is the database table storing the recommendation index for each quarter.

5 | OPTIMIZE RECOMMENDED TRAVEL ROUTES

The project is based on large language models, genetic algorithms, dynamic programming, information interaction, data visualization to optimize and recommend routes. First, the back-end service is developed through the Flask framework [24] and javaScript [25] to process attraction requests and query attraction information from the database, and then the front-end page displays the attraction information and related functions.

5.1 | Applications of Genetic Algorithm

The genetic algorithm searches based on the time limit entered by the user to maximize the attraction recommendation index and travel routes. In the absence of user preferences, the algorithm will initialize a population containing all attractions (33), arrange the attractions in random order, and calculate the fitness (36). The fitness value takes into account the attraction recommendation index (34) and timeout penalty (35). A new generation of populations is generated through crossover and mutation.

$$N = [q_1, q_2, ..., q_i] \tag{33}$$

$$S_{total} = \sum_{j=1}^{I} S_j \tag{34}$$

$$\begin{aligned} R_{timeout} &= p \cdot (t_{total} - t_{litmit}) \quad t_{total} > t_{\lim it} \\ R_{timeout} &= 0 \quad\quad\quad\quad t_{total} \leq t_{\lim it} \end{aligned} \tag{35}$$

$$f(N) = S_{total} - R_{timeout} \tag{36}$$

Each individual N is a permutation of attractions, where q_i represents the i attraction; S_j is the recommendation index; $R_{timeout}$ is the timeout penalty; and $f(N)$ is the fitness of the individual.

The algorithm is terminated until the maximum number of iterations is reached or the fitness improvement stagnates. As shown in Figure 4:

5.2 | Data visualization and intelligent recommended route

Use Python matplotlib to convert the drawing library attributes and draw a roadmap, and add information annotations to make the information display more intuitive. Output the route map as a web page so that users can intuitively see the generated route planning map. Users can modify travel routes, view and adjust attraction information through the interactive functions of the graphical interface. As shown in Figure 5:

5.3 | Recommendation of special tourist routes

Through natural language processing technology, using the jieba library for text segmentation, and using the before BERT model to analyze user preferences, the system will generate travel routes that meet personal preferences. Users can make modifications based on recommended routes. As shown in Figure 6:

Users can freely select attractions based on interests and time constraints on the travel planning interface, and the system will automatically generate travel routes based on the selected attractions. If time is limited, the intelligent recommendation system

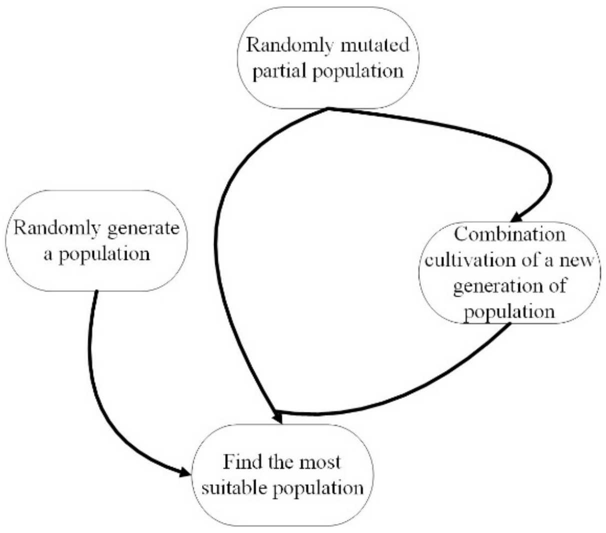

Randomly mutated partial population
Randomly generate a population
Combination cultivation of a new generation of population
Find the most suitable population

FIGURE 4 Genetic Algorithm Analysis Diagram.

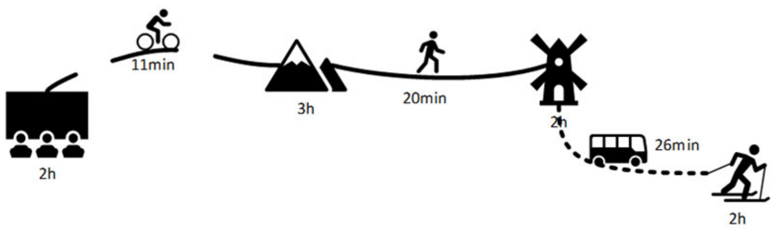


FIGURE 5 Tour Route Recommendation Diagram.

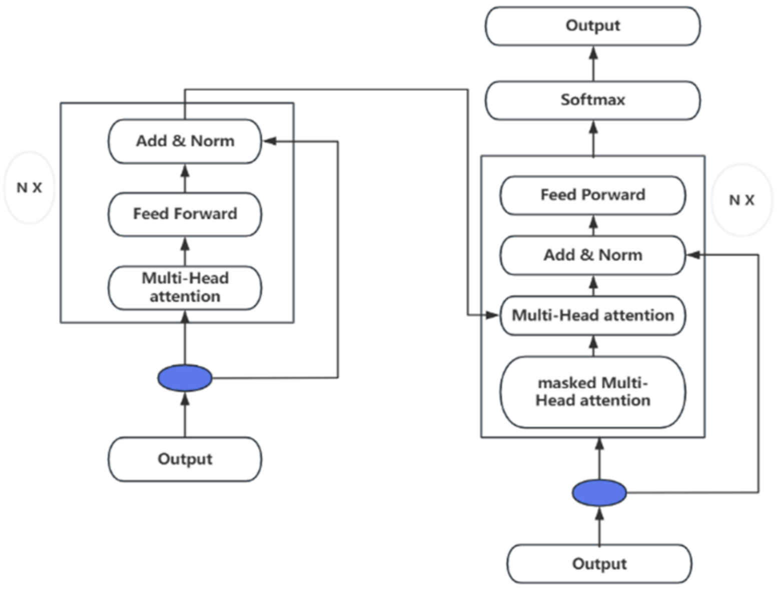


FIGURE 6 Semantic Understanding Function Flowchart of Large Language Models.

will generate a suitable characteristic travel route based on genetic algorithms, and users can also adjust this route to meet personal needs.

5.4 | Tourism route optimization

The system will use natural language processing technology to perform data cleaning, word segmentation and keyword extraction (37), and combine it with the scenic spot database for screening. During the matching process, the system will calculate and sort the similarities (38) between the keywords entered by the user and the attraction tags to select the attractions that best meet the user's needs travel route (39).

$$\begin{aligned} TI(k_i) &= t(k_i) \cdot i(k_i) \\ T(k_i) &= \frac{frequency(k_i)}{totaltimes} \\ I(k_i) &= \log\left(\frac{documents}{keydocuments}\right) \end{aligned} \tag{37}$$

$$Sim(m, p_j) = \frac{\sum_{i=1}^{p} TI(m_i) \cdot TI(l_{ji})}{\sqrt{\sum_{i=1}^{p} (TI(m_i))^2} \cdot \sqrt{\sum_{i=1}^{p} (TI(p_{ij}))^2}} \tag{38}$$

$$f(route) = \sum_{j=1}^{n} S_j - \beta \cdot TravelTime(route) \tag{39}$$

Trepresents the term frequency of keyword k_i in the user's input, I represents the inverse document frequency (rarity) of the keyword across all texts; m is the set of keywords in the user's input, p_j is the tag set and S_j is the recommendation index.

Finally, the optimized route will comprehensively consider all constraints, and the system will generate the final tourist route based on the optimal solution obtained by the dynamic programming algorithm [26], and ensure that the route meets the user's preferences and needs to the greatest extent, while taking into account the overall Smoothness and satisfaction of travel experience.

6 | SUMMARY

This model is different from the tourism recommendation system based on numerical data and location. It adopts a combination of large language model and natural language processing technology. By constructing a knowledge map of scenic spots and analyzing local related indicators, it is easier and more realistic. Data mining analysis is combined with genetic algorithms and dynamic planning to select and optimize routes, giving more comprehensive route recommendation results and allowing users to choose and adjust, making it more flexible and efficient.

ACKNOWLEDGEMENTS

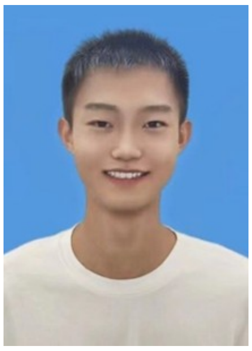
First of all, HaiqingLi would like to express my deepest gratitude to my mentor Xupeng Wang for their valuable guidance, support and encouragement during my research process. At the same time, I would like to express my sincere gratitude to Zijun Chen, HaoweiLiu and ShanghuiChang for their collaborative efforts and assistance in data collection during this research. Finally, I would like to express my special thanks to 2025 Shandong Provincial College Students Innovation and Entrepreneurship Training Program named Large Language Model-Based Travel Route Optimization and Recommendation] for financial support under the Grant [S202410429056].

REFERENCES

[1] Franklin A. Tourism as an ordering: Towards a new ontology of tourism. Tourist studies 2004;4(3):277–301.

[2] Chen C, Zhang S, Yu Q, Ye Z, Ye Z, Hu F. Personalized travel route recommendation algorithm based on improved genetic algorithm. Journal of Intelligent & Fuzzy Systems 2021;40(3):4407–4423.
[3] Dong Q, Li L, Dai D, Zheng C, Ma J, Li R, et al. A survey on in-context learning. arXiv preprint arXiv:230100234 2022;.
[4] Min S, Lewis M, Zettlemoyer L, Hajishirzi H. Metaicl: Learning to learn in context. arXiv preprint arXiv:211015943 2021;.
[5] Zhou S, Zhang W, Huang Y, Gao L. Geochemical Big Data Processing Based on Hadoop Platform. International Journal of Applied Mathematics in Control Engineering 2022;5(1):16–23.
[6] Becker J, Kuropka D. Topic-based vector space model. In: Proceedings of the 6th international conference on business information systems Citeseer; 2003. p. 7–12.
[7] Castells P, Fernandez M, Vallet D. An adaptation of the vector-space model for ontology-based information retrieval. IEEE transactions on knowledge and data engineering 2006;19(2):261–272.
[8] Zhuansun Y, Zhi J, Zhang W, Han Y. Extraction of the Main Features of Pellet by Different Mathematical Algorithms. International Journal of Applied Mathematics in Control Engineering 2020;3(2020):113–120.
[9] Li J. Tourist Attractions Translation Database Aided by Digital Technology. In: International Conference on Innovative Computing Singapore: Springer Nature Singapore; 2024. p. 55–64.
[10] Xiao L, Li Q, Ma Q, Shen J, Yang Y, Li D. Text classification algorithm of tourist attractions subcategories with modified TF-IDF and Word2Vec. PloS one 2024;19(10):e0305095.
[11] Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems 2022;35:24824–24837.
[12] Lyu Q, Havaldar S, Stein A, Zhang L, Rao D, Wong E, et al. Faithful chain-of-thought reasoning. In: The 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (IJCNLP-AACL 2023); 2023. .
[13] Guo T, Yang Q, Wang C, Liu Y, Li P, Tang J, et al. Knowledgenavigator: Leveraging large language models for enhanced reasoning over knowledge graph. Complex & Intelligent Systems 2024;10(5):7063–7076.
[14] You Gw, Hwang Sw. Search structures and algorithms for personalized ranking. Information Sciences 2008;178(20):3925–3942.
[15] Li G, Chen Q. Exploiting explicit and implicit feedback for personalized ranking. Mathematical Problems in Engineering 2016;2016(1):2535329.
[16] Das P, Chaudhury S, Nelson E, Melnyk I, Swaminathan S, Dai S, et al. Larimar: Large language models with episodic memory control. arXiv preprint arXiv:240311901 2024;.
[17] Chowdhary K, Chowdhary KR. Natural language processing. Fundamentals of Artificial Intelligence 2020;p. 603–649.
[18] Patil M. Optimized Scene Text Detector and Paddle Optical Character Recognizer Techniques to Extract Text from Images. In: Proceedings of 4th International Conference on Artificial Intelligence and Smart Energy: ICAIS 2024, Volume 1, vol. 1 Springer Nature; 2024. p. 218.
[19] Cui C, Ye Z, Li Y, Li X, Yang M, Wei K, et al. Semi-supervised recognition under a noisy and fine-grained dataset. arXiv preprint arXiv:200610702 2020;.
[20] Li C, Zhao Y, Li S. Prediction of the yield based on the BP neural network and fitting. International Journal of Applied Mathematics in Control Engineering 2021;4:80–87.
[21] Haggag MH. Keyword extraction using semantic analysis. International Journal of Computer Applications 2013;61(1).
[22] Mithe R, Indalkar S, Divekar N. Optical character recognition. International journal of recent technology and engineering (IJRTE) 2013;2(1):72–75.
[23] Hamad K, Kaya M. A detailed analysis of optical character recognition technology. International Journal of Applied Mathematics Electronics and Computers 2016;(Special Issue-1):244–249.
[24] Mufid MR, Basofi A, Al Rasyid MUH, Rochimansyah IF, et al. Design an mvc model using python for flask framework development. In: 2019 International Electronics Symposium (IES) IEEE; 2019. p. 214–219.
[25] Wirfs-Brock A, Eich B. JavaScript: the first 20 years. Proceedings of the ACM on Programming Languages 2020;4(HOPL):1–189.
[26] Hindelang TJ, Muth JF. A dynamic programming algorithm for decision CPM networks. Operations research 1979;27(2):225–241.

Haiqing Li is currently an undergraduate student at the School of Software Engineering, Qingdao University of Science and Technology, China. Her main research interests include software engineering, artificial intelligence, and big data analytics.



Shanghui Chang is currently an undergraduate student at the School of Software Engineering, Qingdao University of Science and Technology, China. His main research interests include software engineering and related fields.

Haowei Liu is currently an undergraduate student at the School of Software Engineering, Qingdao University of Science and Technology, China. His main research interests include software engineering and related fields.

Zijun Chen is currently an undergraduate student at the School of Software Engineering, Qingdao University of Science and Technology, China. His main research interests include software engineering and related fields.

Xupeng Wang is currently a teacher at at the School of Software Engineering, Qingdao University of Science and Technology, China. His main research interests include deep learning and generative artificial intelligence.