# A framework for recommending tourist attractions using deep learning and association rule mining-based methods

*Hanjo* Jeong*

Mokpo National University, Department of Software Convergence Engineering, 1666 Yeong san-ro, Cheonggye-myeon, Muan-gun, Jeollanam-do, 58554, Korea

**Abstract.** Many of tourism recommendation researches are based on the user rating and review data on the tourism platforms, and these approaches might be only suitable for a discrete recommendation for the tourist attractions. It is because each rating and review data on the platforms is created for a tourist place, not for multiple places on a travel itinerary. A travel blog data often contains information about the multiple places on a travel itinerary, but it is difficult to analyse the data compared to the rating and review data since it is like a text document having longer text than the review. In this paper, we introduce a framework consisting of a deep learning-based tourist-attraction extraction method from the blog text and an association rule mining-based recommendation method to recommend a list of tourist attractions that might be favourable to visit together in a travel itinerary.

## 1 Introduction

There have been many researches on the tourism recommendation problems. Many of them use user rating data and text reviews on the tourism platforms such as TripAdvisor. Such approaches are based on collaborative or hybrid filtering [1-3], sentimental analysis [4-6], and semantic clustering or classification [7-8] algorithms. However, the approaches based on the ratings and review data might be only suitable for a discrete recommendation for the tourist attractions. It is because they recommend the discrete list of the tourist attractions in the order of their reputations and/or the expected user preference on them, i.e., they do not consider the associations among the recommending tourist attractions.

Many of tourists do not visit only a place at a time, they want to visit more places and it might be better if the related places are recommended together. Such demand can be fulfilled with tourist route recommendation approaches. The most of the researches on the tourist route recommendation are focused on the location prediction based on the smart device and IoT data such as smart phones, RFID, and WI-FI tracking data [9-11]. In general, these approaches recommend the tourist route based only on the findings of the most co-visited tourist attractions, and they do not consider the user reviews on the tourist attractions.

In this paper, we introduce a framework for recommending a set of the tourist attractions, which might be better to visit together. The framework includes a deep learning-based automatic extraction of tourist attractions and an association rule mining (ARM) -based recommendation of the related tourist attractions. The tourist attractions for the recommendation include tourist spots, famous restaurant, and accommodations.

Fig. 1 shows the overall process of the recommendation framework. We collect the travel blog data from Naver open API (openapi.naver.com). We use the blog data instead of the ratings and the review data from TripAdvisor. It is because the rating and the review data is only for one tourist attraction thereby impossible to extract the associations among the tourist attractions. A travel blog generally introduces the tourist spots, restaurants, and accommodations visited in a travel itinerary, so places frequently mentioned in common are highly likely to be good places to visit together in a travel itinerary. Thus, a deep learning-based method is introduced to extract the tourist places from the collected blog data and an ARM-based tourist-attraction recommendation method is also introduced to find the favourable associations of the extracted tourist attractions.

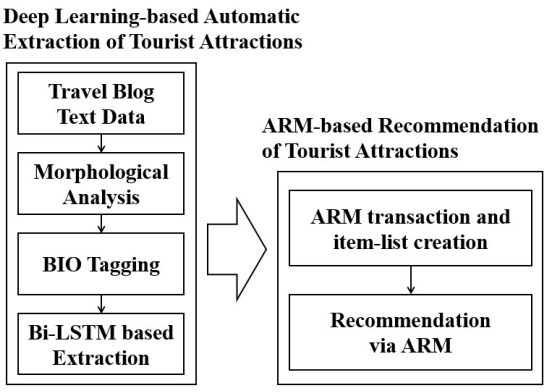


**Fig. 1.** Overall process of the recommendation framework

* Corresponding author: hanjojeong@mnu.ac.kr

## 2 Deep learning-based tourist attraction extraction

This section describes a deep learning-based method of automatic extraction of the names of tourist attractions, restaurants and accommodations from the travel blog text data.

### 2.1 Data collection

A total of 22,000 travel blog data were collected through search terms with "여행_travel" and "관광_tourism" including the names of 22 major cities in Jeonnam province through Naver Open API.

For example, the search terms are generated as "전남_Jeonnam 강진_Gangjin 여행_travel 관광_tourism", "전남_Jeonnam 고흥_Goheung 여행_travel 관광_tourism", "전남_Jeonnam 곡성_Gokseong 여행_Travel 관광_Tourism", etc.

### 2.2 Data preprocessing and training data creation

Fig. 2 shows the examples of the travel blog data collected from Naver open API and only the description column is used to extract tourist attractions, restaurant, and accommodations.

| Title | Link | Description |
|---|---|---|
| 전남 강진 여행 | https://blog.nav | 개별 ? 소규모 ? 맞춤형' 관광 |
| 전남 강진 여행 | https://blog.nav | 남도답사 1번지 관광택시 발대 |
| 힐링여행 | https://blog.nav | 생활하며 강진의 문화를 이해 |
| 전남 가볼 | https://blog.nav | 한 농가에서 농촌 생활관광을 |
| 전남( | https://blog.nav | 강진의 맛 전남 |
| 전남 강진 | https://blog.nav | 전남 강진 병영여행 |
| [지역N문화 테마<b | https://blog.nav | 전남 강진 여행코스 강진 |
| 전남 강진여행< | https://blog.nav | 전남 강진 가볼만한곳으로 <b |
| 전남 강진 | https://blog.nav | 명의 여행객이 방문하는 서남 |

**Fig. 2.** Examples of travel blog data collected from Naver open API.

Html tags are removed on the description and the description sentences are split into morphemes via a Korean morphological analysis. Lastly, we tagged the morphemes with the BIO tags shown in Table 1 in order to use them for a deep learning-based training.

**Table 1.** BIO tags denoting the word/s for tourist attractions, restaurants, and accommodations

| Tag | Description |
|---|---|
| B-TA | Beginning word of tourist attractions (TA) including tourist spot, historical place, etc. |
| I-TA | Intermediate word of TA |
| B-RS | Beginning word of restaurant (RS) |
| I-RS | Intermediate word of RS |
| B-HT | Begging word of Hotel (HT) |
| I-HT | Intermediate word of HT |
| O | Other words besides of TA, RS, HT |

Followings are an example training data consisting of the paired list of the morphemes of a sentence and their corresponding BIO tags as defined Table 1:

['원시/Noun', '체험/Noun', '의/Josa', '섬/Noun', '시호도/Noun', '와/Josa', '가족/Noun', '의/Josa', '섬/Noun', '우도/Noun', '를/Josa', '거닐다/Verb', '과역/Noun', '터미날/Noun', '앞/Noun', '에/Josa', '있는/Adjective', '해주/Noun', '식당/Noun', '에서/Josa', '갈치/Noun', '반/Noun', '으로/Josa', '점심식사/Noun', '를/Josa', '마쳤다/Verb', './Punctuation']

```
['O', 'O', 'O', 'O', 'B-TA', 'O', 'O', 'O', 'O', 'B-TA', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'B-RS', 'I-RS', 'O', 'O', 'O', 'O', 'O', 'O', 'O']
```

### 2.3 Deep learning-based tourist attraction extraction

#### *2.3.1 Long short-term memory (LSTM)*

Long short-term memory (LSTM) algorithm [12] is a recurrent neural networks (RNNs)-based algorithm, which alleviates the vanishing gradient problem in the RNNs. LSTM uses the memory cell consisting of input gate, forget gate, output gate as the hidden node and the forget gate is lastly introduced to reset the memory cell with the more relevant memory contents [13]. The LSTM's update equations are as follows:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \quad (1)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$

$$g_t = tanh\left(W_g x_t + U_g h_{t-1} + b_g\right) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

where f, i, o represents the forget gate, input gate, and output gate respectively, and operator $\odot$ represents the Hadamard product [14], which is the element-wise multiplication between two matrices.

The hidden state, $h_{t-1}$of the LSTM's memory cell memorizes all of the information until the $t-1$ time, and it combines the current input information and transfers the merged information to next state. The forget gate controls which previous information is reset or not, and the combined information with current input information, which calculated by $i_t \odot g_t$ as shown in Equation (5), will be transferred to the next cell's hidden state $h_t$ by combining the value of output state calculated by $o_t \odot \tanh(c_t)$ as shown in Equation (6).

### *2.3.2 Bidirectional LSTM CRF (Bi-LSTM-CRF) based tourist attraction extraction*

While the LSTM alleviates the vanishing gradient problem, it only takes account of the previous information because is recurrent only in one way. Bidirectional LSTM is introduced to use both previous and next information by recurring in both backward and forward [15], i.e., it can apply the effect of the later words to determine the previous words. Bidirectional LSTM with a CRF layer is also introduced to utilize the sentence level information for the sequence tagging [16].

We use the Bi-LSTM-CRF network to have the advantages of the sentence level information since the travel blogs consist of multiple sentences. Fig. 3 represents how we perform the named-entity recognition of the tourist attractions, restaurants and accommodations using the Bi-LSTM-CRF. Each morpheme consisting of a word and a part of speech pair is defined as a term in vocabulary to form the input vector as in Fig. 3. The output vector consists of the seven BIO tags defined in Table 1 including the 'o' tag.

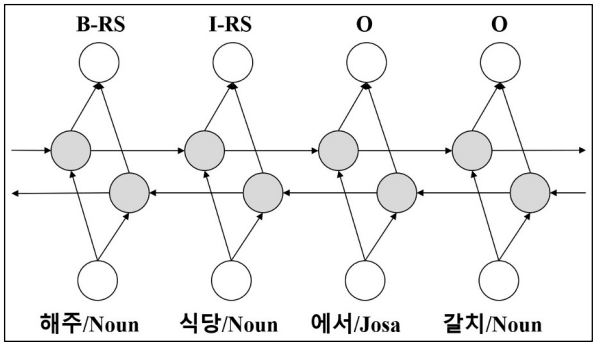


**Fig. 3**. Overall process of tourist attractions recommendation.

## 3 ARM-based tourist attraction recommendation

This section describes an association rule mining (ARM)-based recommendation method for recommending the tourist attractions, restaurants, and accommodations with the extracted information using the deep learning-based method introduced in Section 2.

### 3.1 ARM transaction and item-list creation

Each blog is created as a transaction and the extracted tourist attractions from the blog via the deep learning method in Section 2 are created as the containing items of the transaction.

The blogs containing none of the extracted tourist attractions are excluded from the transaction dataset, and also the blogs containing less than two items are also excluded because it does not indicate the correlation of co-occurrence among the items.

Additionally, the named entity having exact syllables with general words such as "Gwangju, restaurant, Jeong, moon, production, sunset, park, map, painter, space, etc." are ignored when extracting the item set. These general nouns are excluded from the itemset because these general terms can be incorrectly extracted, i.e., it is unclear whether the extracted word represents the name of the attraction visited by the blogger or the general word just used in a sentence.

**Table 2.** Examples of extracted transaction dataset

| TR | List |
|---|---|
| 1 | {'Naru restaurant', 'Nam river', 'BunhongNaru restaurant', 'Saeojae historic-place', 'Haeyoun restaurant'} |
| 2 | {'Nam river', 'Jindo island'} |
| 3 | {'Gaudo island', 'Dasanchodang historic-place', 'baekryon temple', 'Udo island'} |
| 4 | {'Nam river', 'Haesang hotel'} |
| 5 | {'Gaudo island', 'Nam river', 'Daschodang historic-place', 'baekryon temple', 'Saeojae historic-place', 'Udo island'} |

### 3.2 Recommendation via ARM

Each travel-blog post is defined as a transaction and the list of tourist attractions extracted from a blog is defined as an itemset of a transaction. The association rules for tourist attraction recommendation is extracted in the form of X => Y while X and Y denotes a set of the attraction items. Thus, the three major measures of ARM can be defined as follows:

- The support (X) represents the ratio of the number of transactions containing X itemset to the total number of entire transactions:

$$Support(X) = \frac{|\{t \in T;\ X \subseteq t\}|}{|T|} \quad (7)$$

- The confidence (X => Y) represents the conditional probability of the occurrence of Y itemset given the occurrence of X itemset:

$$Confidence(X => Y) = \frac{Support(X \cup Y)}{Support(X)} \quad (8)$$

- The Lift (X => Y) represents independence of X and Y and it also prevents incorrect rules, which can be generated by common items:

$$Lift(X => Y) = \frac{Support(X \cup Y)}{Support(X) \times Support(Y)} \quad (9)$$

## 4 Experiments and Results

ARM is performed using the apriori [17-18] method in Python mlxtend package. The support value is set to a small number, 0.001 even it is important to measure how much the rule itself is reliable. It is because the number of the tourist attraction items are relatively large in the size of blog data and there are a few items that can have a large number of the support value.

Confidence and lift values are also set to a small number, 0.01 because our proposed system performs recommendation incrementally based on user-selected tourist attractions. Thus, we find all of the rules having the higher values of the minimum threshold values and sort the rules based on lift value. Lastly, the top N rules are used to recommend the tourist attractions, which is mostly visited by the bloggers based on the user-selected tourist attractions.

As shown in Table 3, the lift value is also used to sort the rules because it enables to generate better recommendations by penalizing the famous tourist attractions, i.e., the famous tourist attractions tend to get higher possibilities of higher confidence values since they can appear at many of the blog posts. Using the lift values can alleviate such biases and make it lower values for the famous items.

Table 4 represents the Top 10 association rules with items {'Gatbawi rock', 'Yudal mountain'} at left hand sides of rules, i.e., if a user wants to be recommended places with the given two places, the ARM-based method can recommend the most associated places according to the user-provided places as shown in Table 4.

**Table 3.** Top 10 association rules among the entire rules

| Rule# | LHS | RHS | support | confidence | lift |
|---|---|---|---|---|---|
| 60 | {'Kwangju temple'} | {'Kwangju park'} | 0.001222 | 1.000000 | 818.500000 |
| 446 | {'Jayounae restaurant'} | {'Seomji farm'} | 0.001222 | 0.615385 | 503.692308 |
| 1132 | {'Jangsung lakefront street'} | {'Backryondong cypress forest'} | 0.002443 | 1.000000 | 385.176471 |
| 1228 | {'Sinan monument', 'Jeungdo island'} | {'Taiping salt garden'} | 0.001680 | 0.846154 | 369.374359 |
| 182 | {'Bada garden'} | {'Dajungwon garden'} | 0.002902 | 1.000000 | 344.631579 |
| 1024 | {'Jangsung lakefront street', 'Backryondong cypress forest'} | {'Nambaek restaurant'} | 0.002443 | 1.000000 | 284.695652 |
| 670 | {'Chuam farm'} | {'Chuamgol inn'} | 0.001069 | 0.304348 | 284.695652 |
| 1174 | {'Sanbada restaurant', 'Bulgabsa temple'} | {'Seonunsa temple'} | 0.001069 | 1.000000 | 272.833333 |
| 1060 | {'Daegijeomdo island', 'Jeungdo island'} | {'Sogijeomdo island'} | 0.002749 | 0.900000 | 267.872727 |
| 780 | {'Gosado island', 'Daegijeomdo island'} | {'Sogijeomdo island'} | 0.002291 | 0.882353 | 262.620321 |

**Table 4.** Top 10 association rules with items {'Gatbawi rock', 'Yudal mountain'} at LHS

| Rule# | LHS | RHS | support | confidence | lift |
|---|---|---|---|---|---|
| 14348 | {'Gatbawi rock', 'Yudal mountain'} | {'Nojeokbong peak'} | 0.000153 | 0.0625 | 68.072917 |
| 14412 | {'Gatbawi rock', 'Yudal mountain'} | {'Street of culture'} | 0.000153 | 0.0625 | 45.381944 |
| 14434 | {'Gatbawi rock', 'Yudal mountain'} | {'Samhakdo island'} | 0.000459 | 0.1875 | 40.843750 |
| 14538 | {'Gatbawi rock', 'Yudal mountain'} | {'Chonsa bridge'} | 0.000306 | 0.1250 | 30.254630 |
| 14324 | {'Gatbawi rock', 'Yudal mountain'} | {'Kimdaejung nobel-peace-prize memorial'} | 0.000153 | 0.0625 | 24.025735 |
| 14520 | {'Gatbawi rock', 'Yudal mountain'} | {'Ullimsanbang historical place'} | 0.000306 | 0.1250 | 6.188447 |
| 14351 | {'Gatbawi rock', 'Yudal mountain'} | {'Dadohae national park'} | 0.000153 | 0.0625 | 5.236378 |
| 14531 | {'Gatbawi rock', 'Yudal mountain'} | {'Jindo island'} | 0.000459 | 0.1875 | 1.708944 |

| | | | | | |
|---|---|---|---|---|---|
| 14541 | {'Gatbawi rock', 'Yudal mountain'} | {'Dokchon restaurant'} | 0.000153 | 0.0625 | 0.557975 |
| 14542 | {'Gatbawi rock', 'Yudal mountain'} | {'Hangukwan restaurant'} | 0.000153 | 0.0625 | 0.557975 |

## 5 Conclusion

In this paper, we presented a framework consisting of a deep learning-based tourist attraction extraction and an ARM-based recommendation method to recommend a list of tourist attractions that might be favourable to visit together in a travel itinerary. Firstly, we used an RNN-based deep learning framework, Bi-LSTM-CRF network, to train and extract tourist attractions based on the travel-blog text data.

In addition, an ARM-based recommendation method is also presented in this paper to recommend the favourable tourist attractions incrementally, i.e., if a user select one place in a travel itinerary and the list of the places preferable to co-visit with the given place can be recommended based on the ranks of the rules. If a user adds one more place in the travel itinerary, and the new list of favourable places are recommended by finding the rules containing all of the currently selected places at the LHS without re-generating the entire rules.

For future work, a more elaborated deep learning-based tourist-attraction extraction method should be addressed to improve the precision of the automatic extraction from the text data.

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021-0058).

## References

1. Q. Shambour, M. Hourani, S. Fraihat, *An item-based multi-criteria collaborative filtering algorithm for personalized recommender systems,* International Journal of Advanced Computer Science and Applications, **7(8)**, 274-279 (2016).
2. M. E. B. H. Kbaier, H. Masri, S. Krichen, *A personalized hybrid tourism recommender system,* In 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), 244-250 (2017).
3. M. Nilashi, A. Ahani, M. D. Esfahani, E. Yadegaridehkordi, S. Samad, O. Ibrahim, E. Akbari, *Preference learning for eco-friendly hotels recommendation: A multi-criteria collaborative filtering approach,* Journal of Cleaner Production, **215**, 767-783 (2019).
4. A. Valdivia, M. V. Luzón, F. Herrera, *Sentiment analysis in tripadvisor*, IEEE Intelligent Systems, **32(4)**, 72-77 (2017).
5. V. Taecharungroj, B. Mathayomchan, *Analysing TripAdvisor reviews of tourist attractions in Phuket, Thailand*, Tourism Management, **75**, 550-568 (2019).
6. N. Z. Dina, *Tourist sentiment analysis on TripAdvisor using text mining: A case study using hotels in Ubud, Bali*, African Journal of Hospitality, Tourism and Leisure, **9(2)**, 1-10 (2020).
7. Z. Abbasi-Moud, H. Vahdat-Nejad, J. Sadri, *Tourism recommendation system based on semantic clustering and sentiment analysis*, Expert Systems with Applications, **167**, 114324 (2021).
8. Z. Abbasi-Moud, H. Vahdat-Nejad, J. Sadri, *Tourism recommendation system based on semantic clustering and sentiment analysis*, Expert Systems with Applications, **167**, 114324 (2021).
9. Y. Su, X. Li, W. Tang, J. Xiang, Y. He, *Next check-in location prediction via footprints and friendship on location-based social networks,* In 2018 19th IEEE International Conference on Mobile Data Management (MDM), 251-256 (2018).
10. T. Arreeras, M. Arimura, T. Asada, S. Arreeras, *Association rule mining tourist-attractive destinations for the sustainable development of a large tourism area in Hokkaido using Wi-Fi tracking data*, Sustainability, **11(14)**, 3967 (2019).
11. C. Bin, T. Gu, Y. Sun, L. Chang, L. Sun, *A travel route recommendation system based on smart phones and IoT environment*, Wireless Communications and Mobile Computing (2019).
12. S. Hochreiter, J. Schmidhuber, *Long short-term memory*, Neural computation, **9(8)**, 1735-1780 (1997).
13. F. A. Gers, J. Schmidhuber, F. Cummins, *Learning to forget: Continual prediction with LSTM*, Neural computation, **12(10)**, 2451-2471 (2000).
14. R. A. Horn, *The hadamard product*. In Proc. Symp. Appl. Math, **40**, 87-169 (1990).
15. A. Graves, J. Schmidhuber, *Framewise phoneme classification with bidirectional LSTM and other neural network architectures*, Neural networks, **18(5-6)**, 602-610 (2005).
16. Z. Huang, W. Xu, K. Yu, *Bidirectional LSTM-CRF models for sequence tagging*, arXiv preprint arXiv:**1508**.01991 (2015).
17. R. Agrawal, R. Srikant, *Fast algorithms for mining association rules*, in Proceedings of 20th international conference very large data bases, VLDB, **1215**, 487-499 (1994).
18. M. Al-Maolegi, B. Arkok, *An improved Apriori algorithm for association rules*, arXiv preprint arXiv:**1403**.3948 (2014).