Pusan National University

Computer Science and Engineering

Technical Report 2023-10

HMM 을 이용한 다음 POI 추천 및 경로 시각화

이성무

정재원

하연지

지도교수 권준호

목 차

1. 서론

1.1. 연구 배경

Global Positioning System(GPS)는 미 국방부가 개발한 위성 항법 시스템으로[1], 상업적 목적으로 사용할 수 없었다. 2000 년 이후, 대중 또한 GPS 를 사용할 수 있게 되면서 위치 기반 서비스(Location Based Services, LBS)는 학계와 산업계 모두에서 큰 관심을 끌었다[2]. GPS 장치, 스마트폰, 사물인터넷(IoT)과 무선 통신 기술이 널리 보급되면서[3] 네비게이션 서비스, 사용자의 이동 정보를 파악하는 건강 관리 서비스, 위치 정보를 활용한 게임 서비스 등 다양한 분야의 위치 기반 서비스(LBS)가 등장했다. 위치 기반 추천 시스템(LBRS)은 위치기반 서비스와 추천 서비스 두 가지 맥락에서 전개되는 서비스로, 위치정보, 사용자 선호도 등을 고려하여 도착지, 활동 등을 추천해주는 것을 말한다[4].

모바일 사용자와 유사 사용자의 장소 선호 정보와, 예측하고자 하는 사용자의 문맥 정보를 이용하여 사용자의 다음 방문 관심 지점(POI)를 추천하는 RNN 기반 추천 시스템을 개발한 연구가 있다[5]. 사용자의 위치 정보를 이용하여 다음 이동 지점을 예측하여 그 지점 주변의 장소 정보를 추천하는 것 또한 위치 기반 추천 시스템(LBRS) 중 하나이다. 이외에도 사용자의 GPS 를 수집할 수 있는 기기들의 수요가 늘어나면서 관련 데이터의 수집 양이 폭발적으로 증가함에 따라 많은 관련 연구가 수행되었다. 앞서 서술한 바와 같이 [5]는 사용자의 장소 선호도, 문맥 정보를 이용하여 다음 방문 관심 지점(POI)을 추천한다. [6]은 식료품 가게에서 RFID 센서로 수집한 고객 카트의 위치 정보를 은닉 마코프 모델(HMM)을 적용하여 사용자 카트의 다음 위치를 예측하고 사용자에게 구매 상품을 추천하는 시스템을 고안하였다. 또한 사용자의 위치와 날씨 정보를 K-Means 클러스터링으로 이산화한 것을 은닉 마코프 모델(HMM)을 이용하여 날씨를 예측하고 구매 트렌드 정보를 고려한 구매 아이템 추천 시에도 은닉 마코프 모델(HMM)을 적용한 연구도 있다[7]. 이와 같이 사용자의 위치를 기반으로 한 추천 서비스는 여러 분야에 적용되고 있다. 본 연구에서는 제주 데이터 허브에서 제공하는 사용자의 렌터카 위치 정보를 활용하여 위치 기반 추천 시스템을 제안하는 것에 그치지 않고 이를 Fast Map Matching(FMM)을 활용하여 시각화 하고자 한다.

1.2. 관련 연구

[8]은 Varied K-Means Clustering 을 통해 사용자의 위치 정보 중 중요한 데이터만을 이용하고자 하였다. 이를 은닉 마코프 모델(HMM)을 이용하여 두 가지 상황에서 연구를

진행하였는데, 하나는 은닉 상태를 요일로 설정하여 어떤 요일에 사용자의 다음 움직임을 예측하는 것이었다. 또한 움직임이 발생한 요일과 시간대를 은닉 상태로 설정하여 사용자가 어떤 요일의 어떤 시간의 움직임을 예측하였다.

또한 [9]은 기존에 사용자의 이동 루트를 추천하기 위하여 다른 사용자의 이동 루트와의 유사성을 비교해 이미 정해져있는 다른 사용자의 이동 루트를 일관적으로 도출하던 것과 달리, 사용자가 방문하고자 하는 관심 지점(POI)의 카테고리 시퀀스를 이용해 개인화된 경로를 제안하였다. 이때 은닉 마코프 모델(HMM)을 적용하여 개인화된 시스템을 고안하였으며, 관심 지점(POI)의 정보를 은닉 상태로 두고 관심 지점(POI)의 카테고리를 관측값으로 설정하였다.

1.3. 연구 목표

앞서 언급한 두 연구 모두 사용자가 방문할 것으로 생각되는 장소를 예측하기 위하여 은닉 마코프 모델(HMM)을 활용하였다. 따라서 사용자의 경로 데이터를 은닉 마코프 모델(HMM)을 이용하여 모델에 학습시킨 후 사용자가 이동할 것으로 예상되는 다음 지점을 추천한다.

2. 연구 배경

사용자의 POI 를 추천하기 위해서 클러스터링, 모델 구성, 가시화를 위한 Fast Map Matching 라이브러리의 사용, 크게 3 가지 과정으로 진행하였으며, 2 절에서는 3 가지 파트로 나누어 알고리즘에 대한 연구와 특징에 대해 서술하고자 한다.

2.1. K-means Clustering

[표 1] 클러스터링 기법의 종류와 특징

Criteria	Hierarchical clustering	K-means++	DBSCAN
Initial condition	No	Yes	Yes
Termination condition	Not precise	Precise	Precise
Arbitrary value	No requirement	Numeric attribute	Numeric attribute
Size of data sets	Not good	Good	Not good
Shape of data set	Arbitrary	Convex	Arbitrary
Granularity	Flexible	K and initial point	Threshold
Result optimization	Optimization	Rebuild optimization	Rebuild optimization
Handling dynamic data	No	Yes	Yes
Behavior on noisy data	No influences	Influences	Not much influences
Distance measurement	Any	Distance at normal space	Density
Implementation	Simple	Simple	Simple

[표 1]에서는 3 가지의 클러스터링 기법인 Hierarchical Clustering, K-means++, DBSCAN 에 대한 특징을 나타낸다.

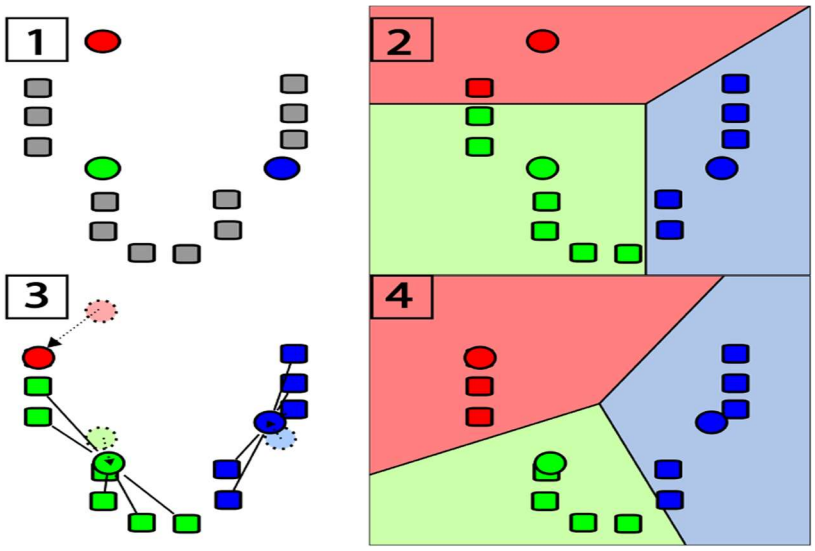
Hierarchical Clustering 은 계층적인 클러스터 구조를 제공하므로, 클러스터 간의 계층적 관계를 시각화하고 이해하는 데 도움이 된다. 이로써 데이터의 구조를 더 잘 이해할 수 있으며, 클러스터 수를 미리 정해주지 않아도 된다. 클러스터 수는 나중에 특정 값으로로 결정할 수 있으며, 클러스터 수에 따라 서로 다른 수준의 계층을 생성할 수 있다. 마지막으로 계층적 구조를 가지므로, 클러스터 간의 크기가 다를 수 있다. 따라서, 비계량적 클러스터링이 필요한 경우 유용하다. 하지만 계층 구조를 만들기 위해 모든 데이터 포인트 간의 거리나 유사성을 계산해야 하므로, 대량 데이터셋에 대해서는 계산 복잡성이 높아질 수 있다.

DBSCAN 은 클러스터 수를 미리 지정할 필요가 없으며, 클러스터 수를 데이터에서 자동으로 결정한다. 이로써 데이터의 복잡성과 밀도에 따라 다양한 크기의 클러스터를 탐지할 수 있다.

또한 데이터에서 이상치를 탐지하고 이상치로부터 클러스터를 분리하는 데 유용하며, 원형, 타원형 또는 비규칙적인 모양과 같이 다양한 형상의 클러스터를 찾을 수 있다. 그러나 대규모 데이터셋에 대해 계산 복잡성이 높을 수 있으며, 메모리 소비량도 크게 증가할 수 있다. 고밀도 데이터의 경우, 클러스터 경계가 일부 포인트에 의해 영향을 받아 클러스터링 결과가 왜곡될 수 있고, 데이터 포인트의 밀도에 따라 클러스터를 형성하므로, 클러스터의 크기가 불균형할 수 있다.

아래의 [그림 1]은 K-means Clustering 의 작동 원리를 보여주고 있다. K-means 는 데이터를 K 개의 클러스터로 그룹화하는 비지도 학습 알고리즘이다. 알고리즘은 초기 중심점을 무작위로 선택한 후, 각 데이터 포인트를 가장 가까운 중심점에 할당하고, 중심점을 재계산하여 클러스터링을 반복하는 방식으로 작동한다.

여기서 K-means++ 는 초기 중심점을 선택하는데 더 효율적인 방식을 사용하여 클러스터링의 수렴 속도와 결과 품질을 향상한 것으로 첫 번째 중심점은 데이터 포인트 중에서 무작위로 선택한 후 다음 중심점을 선택할 때, 이미 선택된 중심점으로부터 먼 데이터 포인트가 다음 중심점으로 선택될 확률을 높인다. 이렇게 함으로써 중심점이 서로 멀리 떨어져 있게 되고, 클러스터 간의 다양성을 증가시키게 된다. K-means++을 사용하면 초기화에 따른 결과의 변동이 줄어들고, 클러스터링 알고리즘의 수렴이 빨라지며, 더 나은 클러스터링 결과를 얻을 수 있다.



[그림 1] K-means Clustering 작동 원리

K-means++ 는 직관적이고 간단한 클러스터링 알고리즘으로, 초기 이해와 사용이 비교적 쉽다. 데이터를 K 개의 클러스터로 그룹화하는 개념은 직관적이며 시각적으로 이해하기 쉽다,

일반적으로 높은 계산 효율성을 가지며, 대규모 데이터셋에서도 상대적으로 빠르게 수행된다. 이는 데이터가 클 수 있어도 비교적 빠르게 클러스터링 결과를 얻을 수 있음을 의미한다. 또한 각 클러스터가 원형 또는 타원형으로 구성되고 데이터가 등방성일 때 잘 작동하며. 이러한 특성은 많은 현실 세계 문제에 적합하다. 이상치에 민감하고 K 의 수를 지정해 줘야 하며 클러스터의 형상에 제한이 있는 단점이 존재하지만 시각적인 사용과 빠른 처리 속도를 위하여 이상치의 경우 전 처리 과정을 통해 모든 제거하였으며, 여러 개의 K 를 지정하여 적절한 K 의 값을 찾아내어 초기값을 설정했다.

2.2. 시계열 데이터 분석을 위한 은닉 마코프 모델

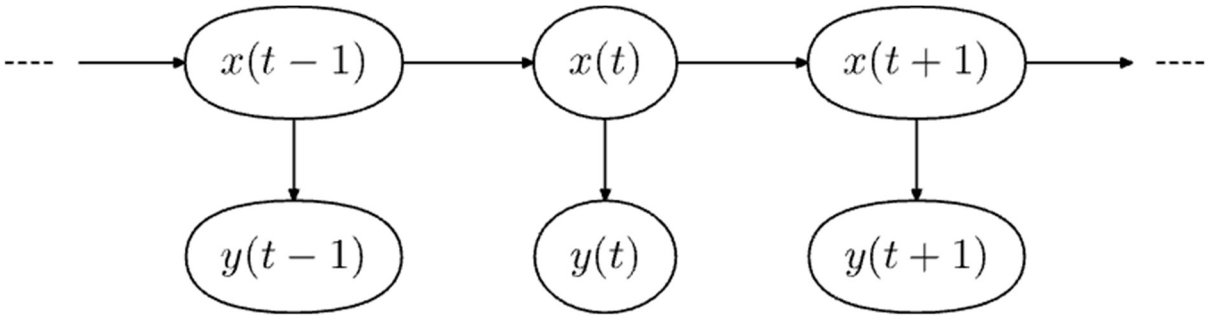
GPS 기반 사용자의 경로 예측 모델은 시계열 데이터 기반 분류 문제로 정의될 수 있다. 분류하기 위한 모델을 선택하기에 시계열 분류에 특화된 모델과 일반적인 분류 알고리즘으로 나눌 수 있다.

[표 2] 대표적인 분류 모델과 개념

CATEGORY	Models	Concept
Time Series Classification Model	DTW (Dynamic Time Warping)	시계열 데이터 간의 비선형 패턴 매칭을 수행하는 알고리즘으로, 패턴의 시간 축을 왜곡하여 매칭
	HMM (Hidden Markov Models)	숨겨진 상태와 관찰 상태 간의 전이 확률을 모델링하는 확률론적 모델
	LSTM (Long Short-Term Memory)	순환 신경망(RNN)의 변종으로, 시퀀스 데이터 처리에 특히 유용한 신경망 아키텍처
General Classification Model	Decision Tree	의사 결정 트리는 의사 결정 규칙을 Tree 구조로 표현하는 분류 및 회귀 알고리즘
	Random Forest	여러 개의 의사 결정 트리를 앙상블하여 분류 또는 회귀 작업을 수행
	SVM (Support Vector Machine)	데이터를 고차원 공간으로 매핑하여 데이터를 분류
	K-NN (K-Nearest Neighbor)	K-NN 은 데이터 포인트의 이웃들을 활용하여 분류 또는 회귀 예측을 수행

[표 2]에서는 다양한 분류 모델이 존재하며, 각 모델별 간단한 개념을 보여주고 있다. HMM 은 주로 시퀀스 데이터 및 숨겨진 상태 모델링에 사용되며, 다른 알고리즘들은 데이터

분류, 회귀 및 패턴 인식과 같은 다른 작업에 중점을 두고 있기에 모델링에 HMM 을 사용하여 구성했다.



[그림 2] Hidden Markov Model 의 구조

다시 말하면 HMM 은 시간적 순서가 중요한 데이터, 예를 들어 음성 신호, 텍스트, 동영상 또는 생물학적 데이터와 같은 시계열 데이터 및 숨겨진 상태를 다룰 때 강력한 도구이다. 데이터의 숨겨진 상태를 추론하는 데 사용되며, 각 상태와 관측치 간의 확률적 관계를 나타내기 때문에 결과를 해석하기 용이하고, 모델의 상태 및 상태 전이를 이해하면 데이터의 동작 및 패턴을 파악하는 데 도움이 된다.

[표 3] HMM 주요 구성요소

- O : 관측값
  - ○ $\{O_1, O_2, \ldots, O_T\}$
- Y : 관측 가능한 값 들의 집합
  - ○ $\{y_1, y_2, \ldots, y_M\}$
- S : Hidden State 의 집합
  - ○ $\{S_1, S_2, \ldots, S_N\}$
- π : 초기 확률
  - ○ $\pi_i = P(S_i), 1 \leq i \leq N$
- A : 전이 확률
  - ○ $a_{ij} = P(S_j \mid S_i), 1 \leq i, j \leq N$
- B : 방출 확률
  - ○ $b_i(O_j) = P(O_j \mid S_i), 1 \leq i \leq N, 1 \leq j \leq M$

HMM 의 5 가지 구성요소로는 관측값, 은닉 상태, 전이 확률, 방출 확률, 초기 확률이 있으며 관측값의 경우 각 상태에서 어떤 관측이 이루어질지를 설명한다. 은닉 상태는 관찰자가 직접 관찰할 수 없는 변수나 정보를 나타낸다. 초기 확률은 초기 상태가 특정 상태일 확률을

나타낸다. 전이 확률은 현재 상태에서 다음 상태로의 확률을 나타내며, 다음 상태는 현재 상태에만 의존하며, 이전 상태의 영향을 받지 않는다. 마지막으로 방출 확률의 경우 특정 은닉상태에 특정 관측값이 나타날 확률을 나타낸다. 본 연구에선 관측값을 POI 와 시간으로 활용했으며 은닉 변수의 경우 POI 이다. 또한 초기 확률, 방출 확률, 전이 확률은 데이터를 카운트하여 직접 구하여 모델을 제작하였다.

2.3. Fast Map Matching 을 활용한 이동 경로 예측의 효과적인 시각화 방법

FMM 은 C++과 Python 의 오픈소스 맵 매칭 프레임워크로 도로망에 노이즈가 많은 GPS 데이터를 매칭하는 문제를 해결하며, 성능, 확장성, 기능성을 극대화하도록 디자인되었다.

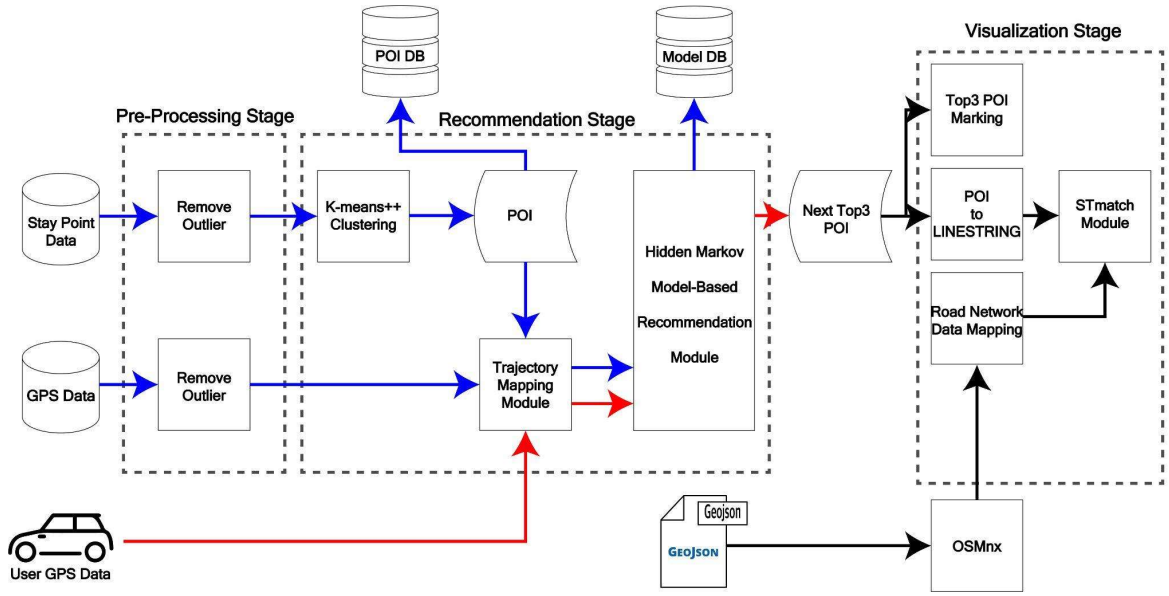
[표 4] Fast Map Matching 의 특징

고성능	Rtree 를 이용한 C++ 구현, 최적화된 라우팅, 병렬 컴퓨팅(OpenMP).
확장성	수백만 개의 GPS 지점과 수백만 개의 도로 Edges.
다중 데이터 형식	OpenStreetMap 또는 ESRI 모양 파일의 도로망입니다.
	점 CSV, 궤적 CSV 및 궤적 형상 파일의 GPS 데이터
상세 매칭 정보	횡단경로, 기하학, 개별 매칭 Edge, GPS 오류
다중 알고리즘	FMM(중소규모) 및 STMatching(대규모 도로 네트워크용)
플랫폼 지원	유닉스(ubuntu), 맥 및 윈도우(cygwin 환경).
헥사곤 매치	uber 의 h3 육각형 계층적 지리공간 색인 시스템과 매치

[표 4]는 FMM 의 특징을 보여주고 있으며, 효율적인 매칭 알고리즘인 STmatching 을 사용한다. FMM 에 관해서는 3.5 장에서 자세히 기술한다.

3. 연구 내용

아래 [그림 3]은 전체적인 구성도이다.



[그림 3] HMM 을 활용한 경로 추천 모델의 전체 구성도

3.1. 제공 데이터셋 설명

앞서 1.1 절에서 설명한 바와 같이 위치 기반 추천 시스템(LBRS)는 위치기반 서비스와 추천 서비스 두 가지 맥락을 고려하여 추천하는 시스템이다. 따라서 위치기반 서비스를 위한 GPS 데이터와 추천 서비스를 위한 사람들이 자주 방문했던 장소의 정보가 필요하다.

위치 기반 서비스를 위해 제주 데이터 허브[1]에서 제공하는 렌터카의 GPS 데이터를, 추천 서비스를 위해 체류거점 데이터를 제공받아 사용한다. "[교통, 안전] 월별 렌터카 위치정보"는 월별 토요일 렌터카의 위치 정보를 나타내며 수집시각 기준 05:00:00 부터 익일 04:59:59 까지의 데이터를 수집한 데이터이다. 총 91,985,676 개의 GPS 데이터로 구성되어있다. "[교통, 안전] 월별 렌터카 체류 거점 위치 정보"는 "[교통, 안전] 월별 렌터카 위치정보"에서 GPS 데이터의 전로그와 후로그의 시각차이가 20 분 이상인 경우, 전로그의 위치 정보를 수집한 데이터이다. 총 699,474 개의 체류거점 데이터로 구성되어있다.

아래 [표 5], [표 6]는 각각 "[교통, 안전] 월별 렌터카 위치정보", "[교통, 안전] 월별 렌터카

[1] 제주데이터허브 (jejudatahub.net)

체류 거점 위치 정보"에 대한 정보이다.

[표 5] 월별 렌터카 위치정보 형식

번호	구분	설명
1	oid	각 렌터카에 부여된 id
2	collection_dt	ms 단위로 수집된 수집시각
3	longitude	경도
4	latitude	위도

[표 6] 월별 렌터카 체류거점 위치정보 형식

번호	구분	설명
1	oid	각 렌터카에 부여된 id
2	collection_dt	ms 단위로 수집된 수집시각
3	longitude	전로그의 경도
4	latitude	전로그의 위도
5	time	collection_dt 를 yyyy-mm-dd hh:MM:ss 형식으로 변경한 수집시각
6	Diff	GPS 데이터 전로그와 후로그 간의 차이

아래 [표 7], [표 8]은 각각 "[교통, 안전] 월별 렌터카 위치정보", "[교통, 안전] 월별 렌터카 체류 거점 위치 정보" 데이터를 나타낸다.

[표 7] 월별 렌터카 위치정보의 예

oid	collection_dt	longitude	latitude
46100c11	20200118050150000	126.4270751	33.251536
46100c11	20200118050220000	126.4297533	33.2513233
…	…	…	…
46100052	20200118051427000	180.0000001	90.0000001
461008ac	20200118051703000	126.2945958	33.4438645

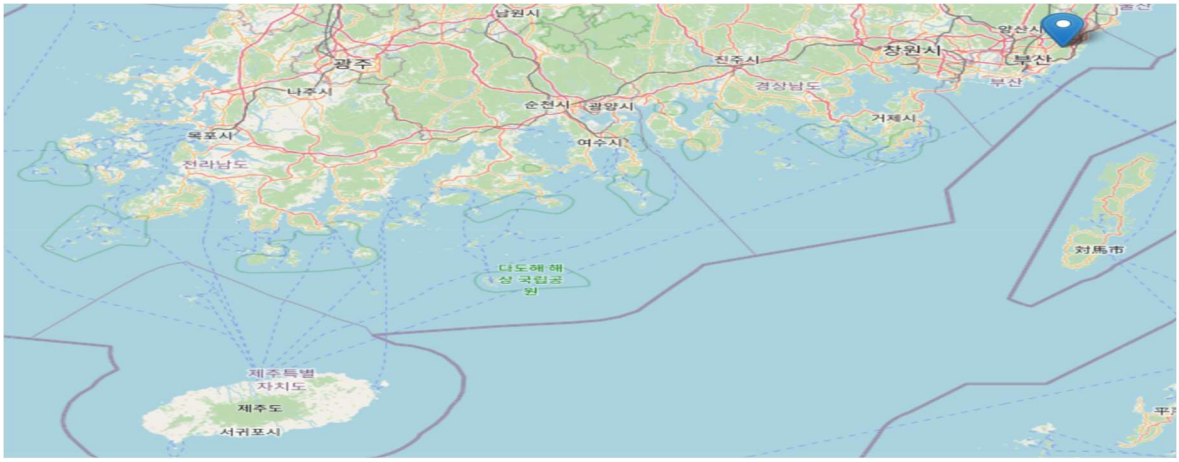
[표 8] 월별 렌터카 체류거점 위치정보의 예

oid	collection_dt	longitude	latitude	time	Diff
0c0000fd	20200111060118	126.4940395	33.493728	2020-01-11 6:01:18	2145
0c0000fd	20200111065334	126.3422986	33.3060038	2020-01-11 6:53:34	1612
…	…	…	…	…	…
46100018	20200104114251	126.4138065	33.2543668	2020-01-04 11:42:51	4715
46100018	20200104130326	126.3528063	33.2817653	2020-01-14 13:03:26	9068

3.2. 데이터 전처리

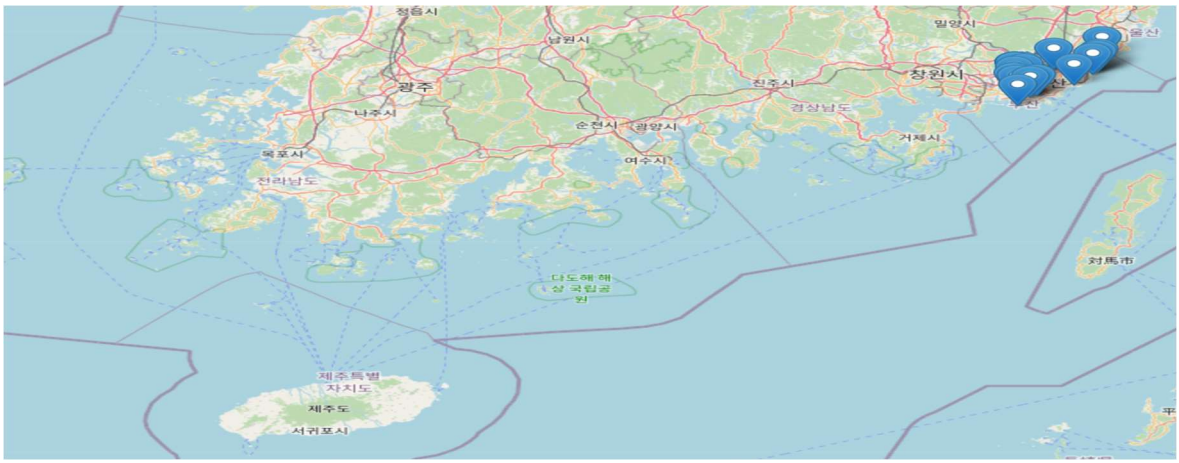
"[교통, 안전] 월별 렌터카 위치정보", "[교통, 안전] 월별 렌터카 체류 거점 위치 정보" 데이터 모두 결측치가 존재하지 않아 결측치를 처리하지 않았다. 이상치는 각 데이터에서 제주도에서 벗어난 좌표라 정의하였다.

"[교통, 안전] 월별 렌터카 위치정보" 데이터 총 91,985,676 개에서 87,950,908 개로 총 4,034,768 개의 이상치가 제거되었다. "[교통, 안전] 월별 렌터카 위치정보"에서 이상치는 아래 그림과 같다.



[그림 4] 월별 렌터카 위치정보 이상치의 예

"[교통, 안전] 월별 렌터카 체류거점" 데이터 총 699,474 개에서 698,841 개로 총 633 개의 이상치가 제거되었다. "[교통, 안전] 월별 렌터카 체류 거점 위치 정보"에서 이상치는 아래 그림과 같다.



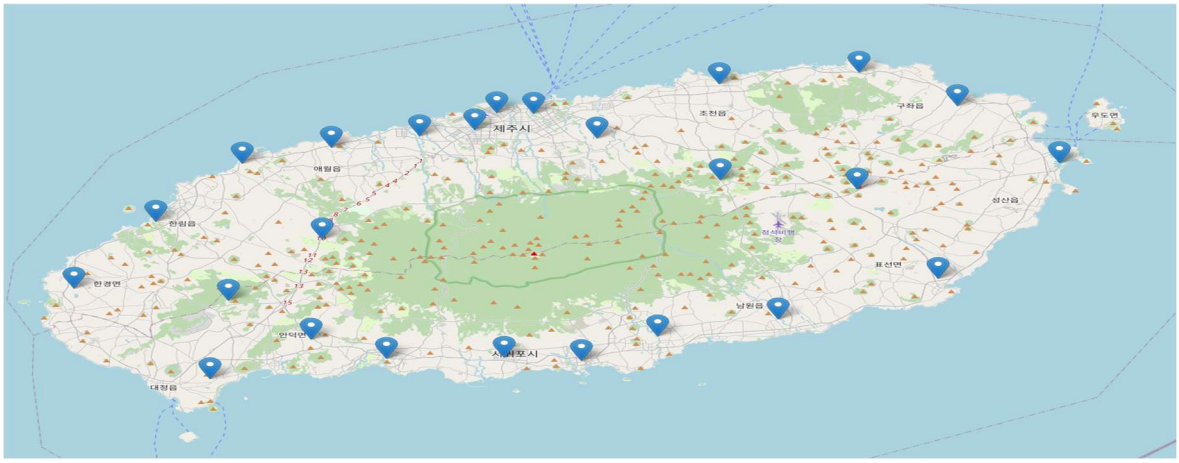
[그림 5] 월별 렌터카 체류거점 이상치의 예

3.3. Recommendation Stage

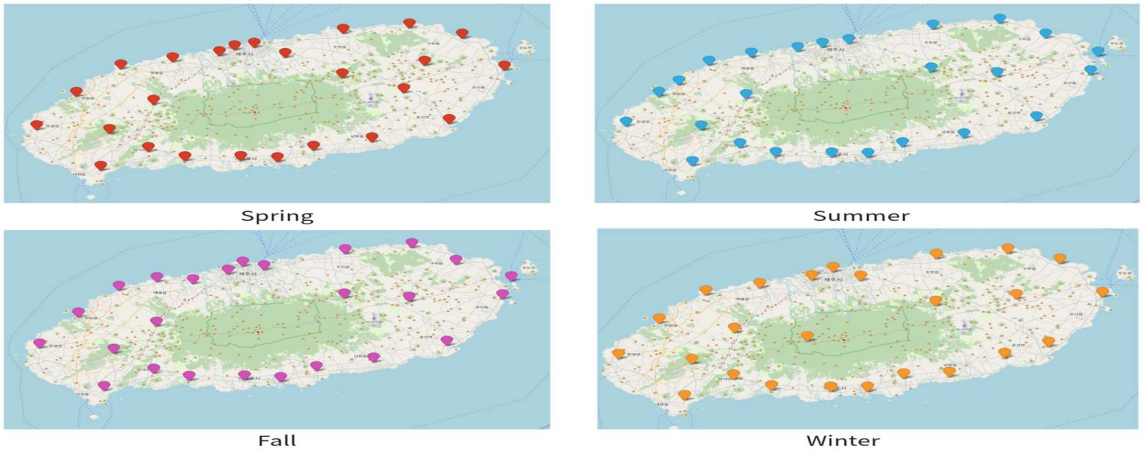
3.3.1. K-means++ Clustering 을 통한 POI 추출

제주도 여행객들의 방문하는 지점들은 계절에 따라 다양한 양상을 보일 것이며 여행객들의 활동 패턴에 직접적인 영향을 끼칠 것이라 가정하였다. 그리하여 POI 를 계절마다 다르게 추출하고 이를 이후 HMM 모델에 적용하여 계절에 맞는 맞춤 서비스를 제공하는 것을 목표로 한다.

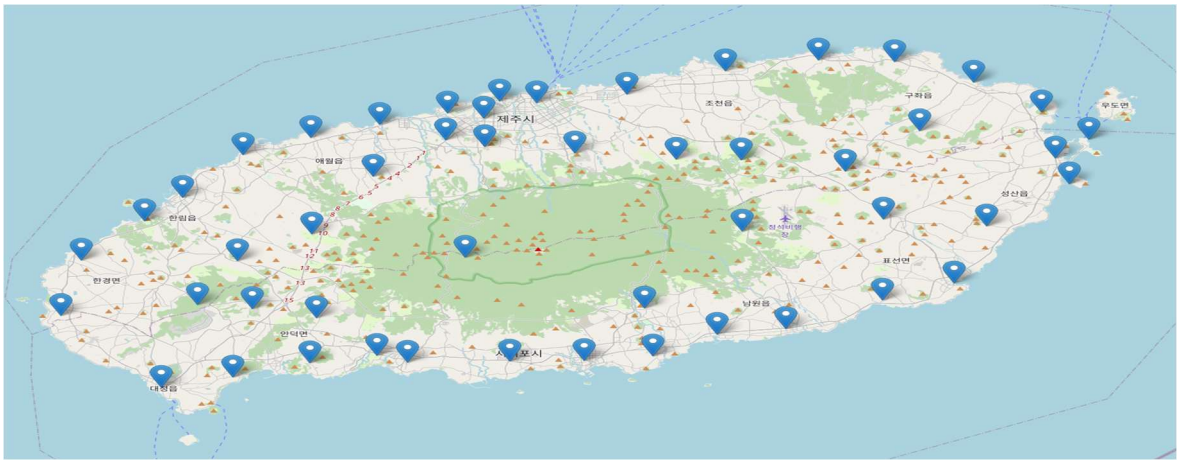
"[교통, 안전] 월별 렌터카 체류 거점 위치 정보" 데이터에 K-means++ clustering 알고리즘을 사용하여 K=25, 50, 75, 100 로 늘려가며 즉, POI 를 25 개, 50 개, 75 개, 100 개로 늘려가며 추출하였다. 전체 POI 추출 결과, 계절별 POI 추출 결과는 아래 그림과 같다.



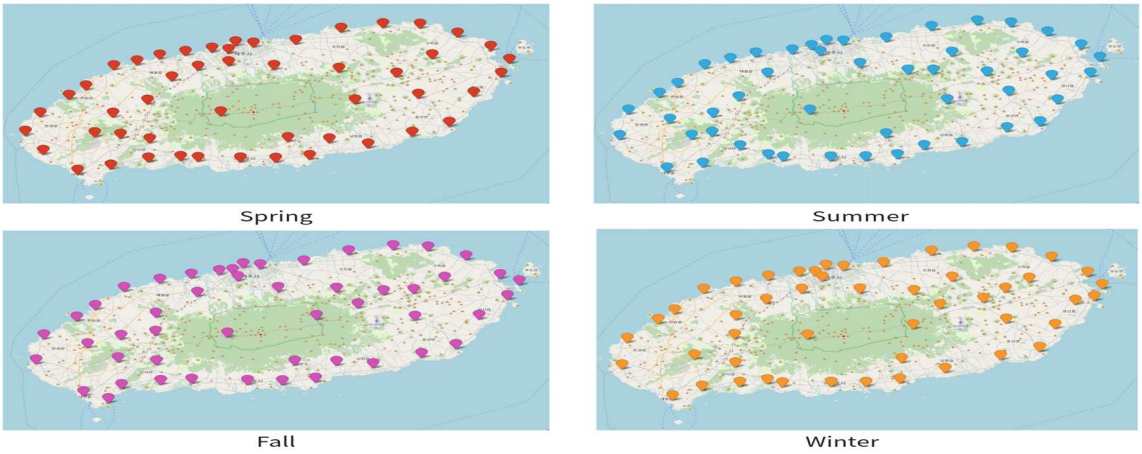
[그림 6] 전체 K = 25 결과



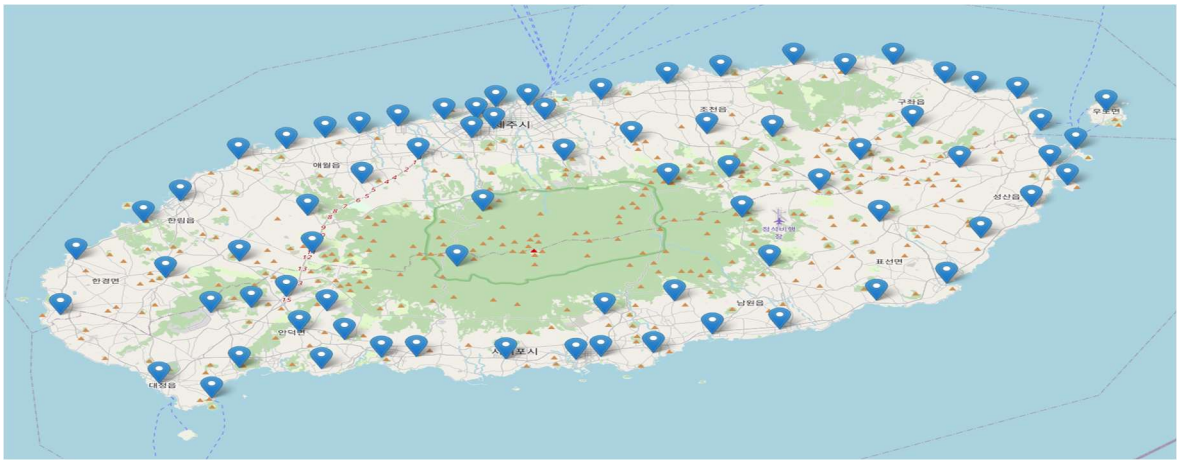
[그림 7] 계절별 K = 25 결과



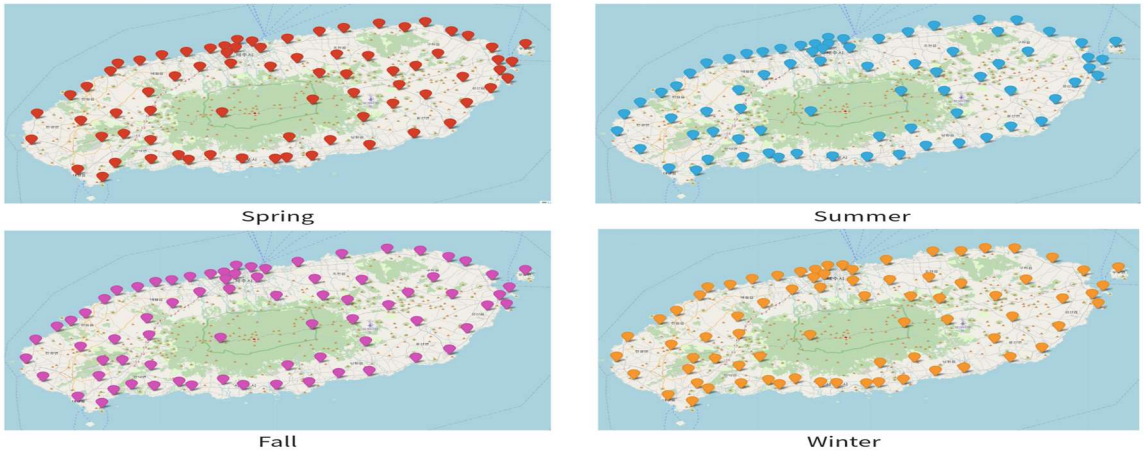
[그림 8] 전체 K = 50 결과



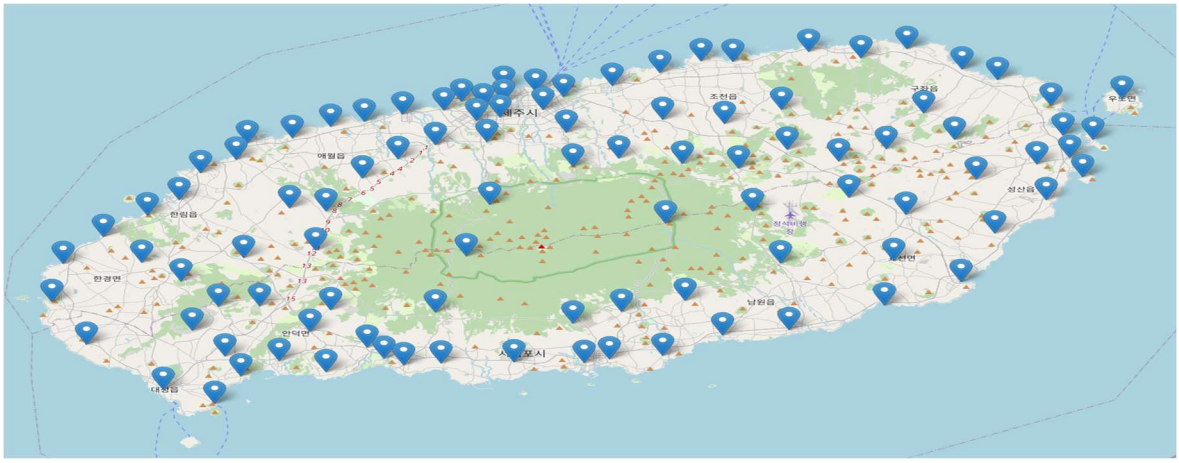
[그림 9] 계절별 K = 50 결과



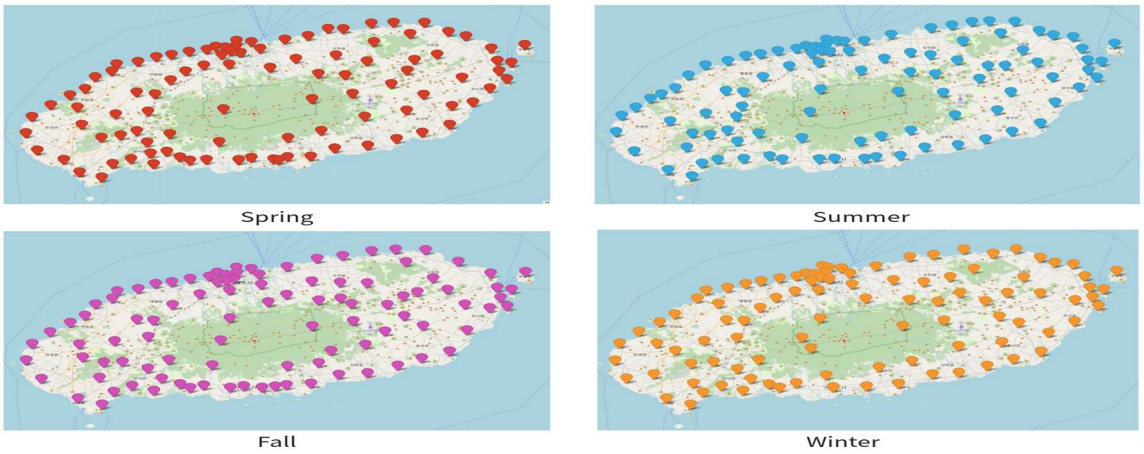
[그림 10] 전체 K = 75 결과



[그림 11] 계절별 K = 75 결과



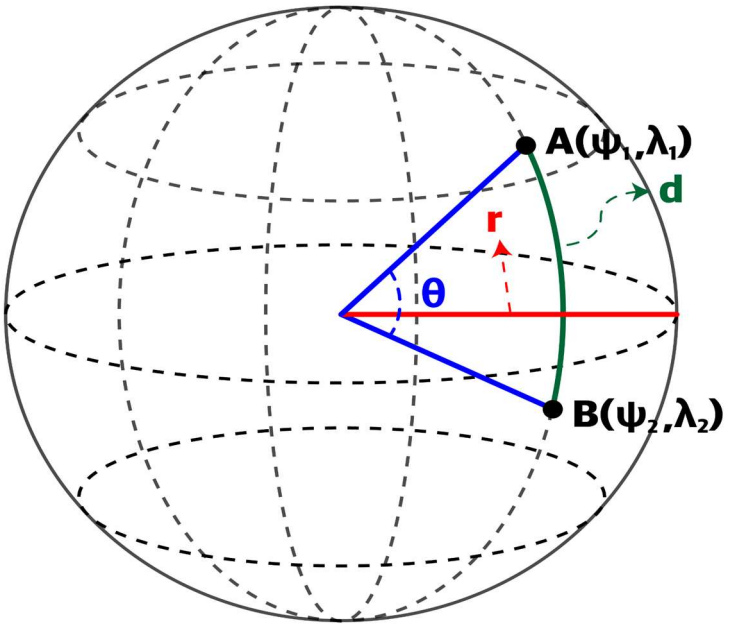
[그림 12] 전체 K = 100 결과



[그림 13] 계절별 K = 100 결과

3.3.2. POI 와 mapping 한 Trajectory 생성

Haversine 공식은 위도와 경도로 표현된 구면 좌표계 상의 두 점 사이의 거리를 구하는 공식이다. 이는 좌표상의 최단거리가 지구의 곡률에 영향에 받기 때문에 사용한다.



[그림 14] haversine 공식을 이용한 두 GPS 포인트 사이의 거리

φ_1과 φ_2 는 점 A 와 B 의 위도를 λ_1과 λ_2 는 점 A 와 B 의 경도를 라디안 단위로 나타낸 것이다.θ는 두 점 A, B 를 잇는 호의 중심각을 라디안 단위로 표현한 각도이다.

$$\theta = \frac{d}{r} \tag{1}$$

$$hav(\theta) = hav(\varphi_2 - \varphi_1) + cos(\varphi_1) \cdot cos(\varphi_2) \cdot hav(\lambda_2 - \lambda_1) \tag{2}$$

$$hav(\theta) = sin^2(\frac{\theta}{2}) = \frac{1 - cos(\theta)}{2} \tag{3}$$

식(2)의 $hav(\theta)$는 하버사인 함수로 식 (3)과 같이 나타낼 수 있다. 두 점 사이의 거리 d 를 구하기 위해 하버사인함수의 역함수를 곱해수면 아래 식 (4)가 유도된다.

$$d = r \cdot hav^{-1}(hav(\theta)) = 2r \cdot arcsin(\sqrt{hav(\theta)} \tag{4}$$

식 (4)에 식 (2)를 대입하면 아래 식 (5)와 같이 두 점의 위도와 경도로 두 점 사이의 거리 d 를 구할 수 있게 된다.

$$d = 2r \cdot arcsin(\sqrt{hav(\varphi_2 - \varphi_1) + (1 - hav(\varphi_1 - \varphi_2) - hav(\varphi_1 + \varphi_2)) \cdot hav(\lambda_2 - \lambda_1)}) \tag{5}$$

$$= 2r \cdot arcsin(\sqrt{sin^2(\frac{\varphi_2 - \varphi_1}{2}) + (1 - sin^2(\frac{\varphi_2 - \varphi_1}{2}) - sin^2(\frac{\varphi_1 + \varphi_2}{2})) \cdot sin^2(\frac{\lambda_2 - \lambda_1}{2})}) \quad (5)$$

$$= 2r \cdot arcsin(\sqrt{sin^2(\frac{\varphi_2 - \varphi_1}{2}) + cos(\varphi_1) \cdot cos(\varphi_2) \cdot sin^2(\frac{\lambda_2 - \lambda_1}{2})})$$

haversine 공식을 활용하여 GPS 데이터 포인트에서 반경 1km 내 가장 가까운 POI 를 찾아 방문한 것으로 간주한다. 이를 각 oid 별 GPS 데이터에 적용하여 POI Sequence 를 생성한다.

또한 수집시간을 dawn, morning, afternoon, night 로 6 시간 간격으로 나눈 후 mapping 하여 time period 라는 새로운 변수를 생성한다. time period 는 이후 Hidden Markov Model 에 적용하기 위한 추가적인 변수로 POI 를 방문했을 때 시간대의 영향을 주기 위해 생성하였다.

생성된 Trajectory 는 아래 [표 9]와 같다.

[표 9] 생성된 Trajectory 형식

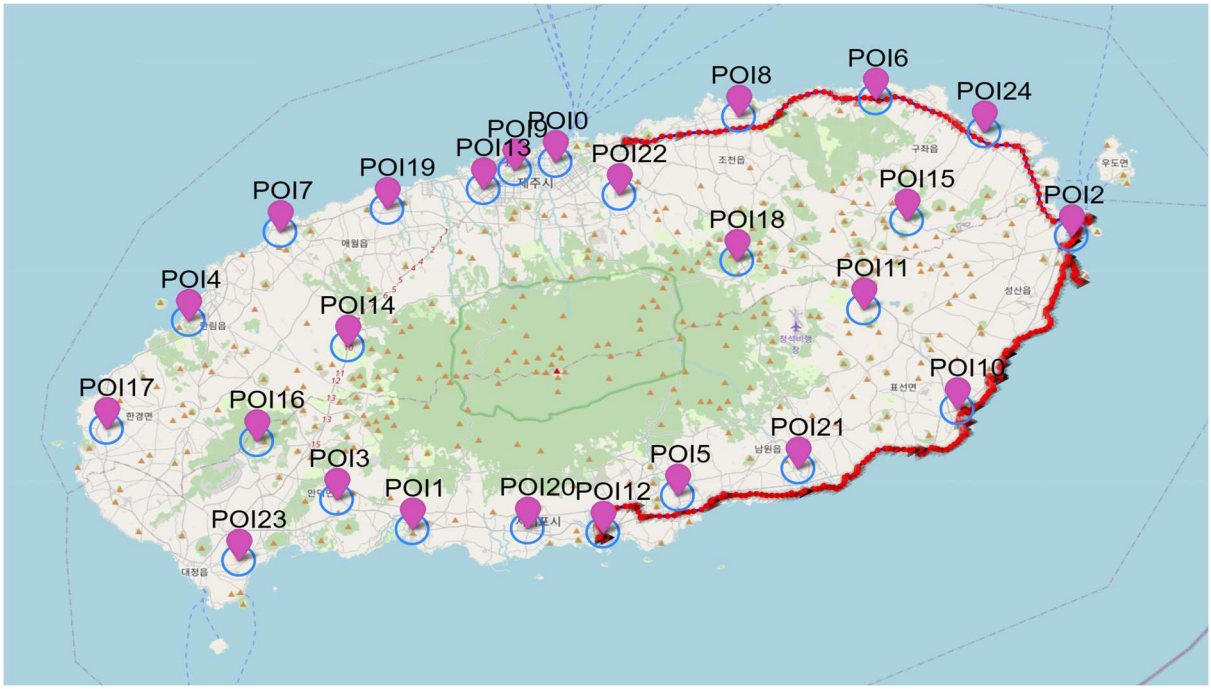
번호	구분	설명
1	trajectory_id	<year, month, day, oid> 형태로 부여된 trajectory id
2	start_point	trajectory 의 첫 방문 POI
3	end_point	trajectory 의 마지막 방문 POI
4	POI_sequence	방문한 POI Sequence
5	time_period_sequence	Mapping 한 time_period Sequence

[표 10] 생성된 Trajectory 의 예

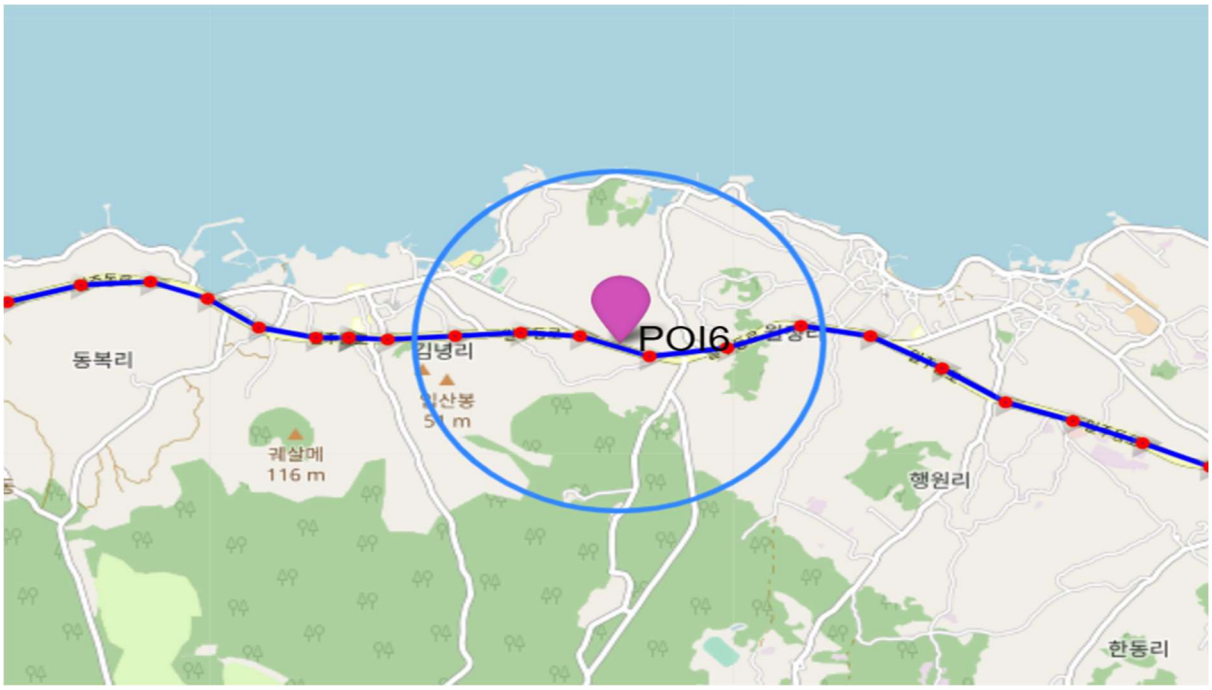
trajectory_id	start_point	end_point	POI_sequence	time_period_sequence
(2020, 3, 7, '46100056')	POI1	POI19	['POI1', 'POI3', 'POI13', 'POI7', 'POI19']	['morning', 'morning', 'afternoon',

trajectory_id	start_point	end_point	POI_sequence	time_period_sequence
				‘afternoon’, ‘afternoon’]
(2020, 3, 7, ‘461000b5’)	POI13	POI24	[’POI13’, ‘POI9’, ‘POI0’, ‘POI8’, ‘POI24’]	[’morning’, ‘morning’, ‘morning’, ‘morning’, ‘morning’]
…	…	…	…	…
(2020, 3, 28, '46100ce2')	POI8	POI2	['POI8', 'POI6', 'POI24', 'POI2', 'POI10', ‘POI12’]	[’morning’, ‘morning’, ‘morning’, ‘morning’, ‘morning’, ‘morning’]

아래 [그림 15]는 [표 10]의 trajectory_id (2020, 3, 28, '46100ce2')에 대해 haversine 공식을 활용하여 GPS 데이터 포인트를 추출된 POI 로 mapping 하여 POI_Sequence 로 생성하는 그림이다.



[그림 15] GPS 데이터를 POI 로 사상하는 예



[그림 16] GPS 데이터를 POI 6 로 사상하는 예

3.3.3. 은닉 마코프 모델(HMM) 정의

은닉 상태(Hidden State) S: 비슷한 생활 양식을 가지는 사용자 그룹의 이동 경로를 관심 지점(POI)의 시퀀스로 표현하고 이를 은닉 마코프 모델(HMM)을 적용해 다음 경로를 예측한 논문이 있다[8]. 해당 논문은 은닉 마코프 모델을 정의할 때 은닉 상태를 클러스터링한 관심 지점(POI)으로 설정하였다. 이를 참고하여 본 연구 또한 관심 지점(POI)을 은닉 상태로 설정하고자 한다. 이때 은닉 상태의 시작 인덱스는 0 이다. 만약 은닉 상태의 개수가 25 개라면 인덱스 0 부터 24 의 은닉 상태가 만들어진다. 즉, 클러스터링한 관심 지점(POI)의 개수가 k 개일 때 은닉 상태는 q_0부터 q_{k-1}까지 정의된다. 사용자의 은닉 상태 시퀀스는 S이며, 이때 i번째 시점의 은닉 상태는 s_i로 정의한다.

관측 가능한 상태(Observable State) O: 관측 가능한 상태는 은닉 상태와는 달리 관찰하거나 측정할 수 있는 상태를 말한다. 본 연구에서는 수집가능한 사용자의 문맥 정보를 반영하여 사용자의 목적지를 예측한 [9]을 참고하여 관측 가능한 상태를 설정하고자 한다. [9]의 경우 수집한 데이터셋에 사용의 위치 정보가 수집된 장소, 이동상태, 요일, 시간이 있어 이들을 모두 활용해 인코딩하였다. 그러나 제주 데이터 허브에서 제공하는 렌터카 위치 정보 데이터의 경우, 토요일에 수집된 데이터만을 제공하고 있다. 뿐만 아니라 개인 정보를 보호하기 위하여 위치 정보와 위치 정보가 수집된 날짜 및 시간만을 제공하고 있기에, 관측값(Observation ID)을 정의할 때 시간 정보만을 활용한다. [9]는 시간 정보 활용을 위해 하루 24 시간을 4 개의 구간으로 나누어 설정하였다. 본 연구 또한 오전 12 시에서 6 시간 간격으로 총 4 개의 시간대를 구성하고 각각 dawn, morning, afternoon, night 으로 대응한다. 이때 dawn 시간대의 ID 는 0 이며 차례로 1, 2, 3 의 값을 가진다. 아래의 식 (6)은 인코딩 방법을 나타낸 것이다.

$$Observation\ ID\ =\ 4\ *\ POI\ ID\ +\ TimePeriod\ ID \tag{6}$$

즉, 예를 들어 사용자가 POI20 을 오전에 방문했다면 해당 관측값은 $4 * 20 + 1 = 81$이 된다. 아래 [표 11]은 하나의 trajectory 에서 인코딩한 결과이다. 관측값의 시퀀스로 표현된 사용자의 관측가능한 상태 시퀀스는 O로 표현되며, 시점 i에서의 관측값은 o_i로 정의한다.

아래 [표 11]은 한 사용자의 경로를 K = 25 일 때 인코딩 된 결과이다.

[표 11] K = 25 일 때 인코딩 한 Trajectory

trajectory_id	start_point	end_point	path	time_period	Observation Sequence
(2020, 3, 7, '46100056')	POI1	POI19	['POI1', 'POI3', 'POI13', 'POI7', 'POI19']	['morning', 'morning', 'afternoon', 'afternoon', 'afternoon']	[5, 13, 44, 30, 78]
(2020, 3, 7, '461000c7')	POI24	POI13	['POI24', 'POI24', 'POI6', 'POI8', 'POI9', 'POI13']	['morning', 'afternoon', 'afternoon', 'afternoon', 'afternoon', 'afternoon']	[97, 98, 26, 34, 38, 54]
(2020, 3, 7, '461000b5')	POI13	POI24	['POI13', 'POI9', 'POI0', 'POI8', 'POI24']	['morning', 'morning', 'morning', 'morning', 'morning']	[53, 37, 1, 33, 97]

초기 확률 π: 초기 확률은 사용자 경로의 시작 지점이 특정 은닉 상태(POI)일 확률을 나타내는 것이다. i 번째 은닉 상태에 대한 초기 확률 π_i를 계산하기 위해 각 사용자 경로의 시작 지점에서 POI_i가 관측된 횟수를 모든 관심 지점(POI)에 대하여 구한다. 그리고 이를 전체 사용자의 수로 나누어 정의하였다. 이를 식으로 나타내면 아래와 같다.

$$\pi_i = P(POI_i), 1 \leq i \leq N \quad (7)$$

전이 확률 A: 전이 확률은 한 은닉 상태(POI)에서 어떤 은닉 상태(POI)로 이동할 확률을 의미한다. 이를 도출하기 위하여 사용자 경로에서 은닉 상태의 전이가 일어나는 횟수를 구하고 이를 전체 상태 전이 횟수로 나누어 준다. 은닉 상태 i에서(q_i) 은닉 상태 j로(q_j) 전이할

확률 a_{ij}를 구하는 식은 아래와 같다.

$$a_{ij} = P(POI_{t+1} = S_j \mid POI_t = S_i), 1 \le i, j \le N \quad (8)$$

방출 확률 B: 방출 확률은 특정 은닉 상태(POI)에서 어떤 관측값이 관측될 확률을 의미한다. 이때 관측값은 위에서 정의한 식으로 인코딩한 것이다. 특정 은닉 상태(q_i)일 때 어떤 관측값($Observation\ ID$)가 관측된 횟수를 전체 관측 횟수로 나누어 구한다. 이를 식으로 나타내면 아래와 같다.

$$b_i(O_j) = P(O_t = O_j \mid POI_t = i), 1 \le i \le N, 1 \le j \le M \quad (9)$$

위의 세 가지 확률은 행렬로써 나타내어 은닉 마코프 모델에 파라미터로 적용하였다.

3.3.4. 다음 POI 추천 알고리즘

Viterbi 알고리즘은 Hidden Markov Model 에 주어진 Observation Sequence $O = (O_1, O_2, \ldots, O_T)$와 모델 파라미터 $\lambda = (S, O, \pi, A, B)$ 로부터 최적의 Hidden state sequence 를 찾는 알고리즘이다. 이는 Hidden Markov Model 의 decode 단계에 가장 일반적으로 사용된다.

$viterbi[i, j]$는 j 에서 Hidden state S_i가 관측값 sequence 를 생성할 가능성이 가장 높은 경로의 확률을 나타낸다. 즉 sequence $\hat{X} = (\hat{x}_1, \hat{x}_1, \ldots \hat{x}_j)$ 발생의 확률이다. 이 때, $\hat{x}_j = S_i$ 이다. $bestpathpointer[i, j]$는 j-1 에서 $\hat{x}_{j-1}$이 Hidden State S_i인 가장 가능성이 높은 경로의 $\hat{x}_{j-1}$을 저장한다. 즉, $\hat{X} = (\hat{x}_1, \hat{x}_1, \ldots \hat{x}_{j-1}, \hat{x}_j = S_i)$를 나타낸다.

Forward 알고리즘은 Observation Sequence $O = (O_1, O_2, \ldots, O_T)$와 모델 파라미터 $\lambda = (S, O, \pi, A, B)$ 로부터 HMM 의 decode 과정에서 도출한 최적의 Hidden state sequence 의 발생 확률 계산하는 알고리즘이다.

아래 Algorithm 1, Algorithm 2 는 각각 Viterbi Algorithm, Forward Algorithm 의 pseudo code 이다. 이때 초기확률 π, 전이확률 A, 방출확률 B는 주어져 있다고 가정한다.

Algorithm1 : Viterbi algorithm

Input : $O_{seq} = [o_1, o_2, \ldots, o_T]$

Output : X

1: ***for each*** *state* $i = 1, 2, \ldots, N$ ***do***

2: $viterbi[i, 1] \leftarrow \pi_i * b_i(o_1)$

3: $bestpathpointer[i, 1] \leftarrow 0$

4: ***end for***

5: **for each** observation $j = 2, 3, \ldots, T$ **do**

6: **for each** state $i = 1,2, \ldots N$ **do**

7: $viterbi[i, j] \leftarrow \max_k \left(viterbi[k, j-1] * a_{ki} * b_i(o_j) \right)$

8: $bestpathpointer[i, j] \longleftarrow \underset{k}{argmax} \left(viterbi[i, j-1] * a_{ki} * b_i(o_j) \right)$

9: ***end for***

10: ***end for***

11: $z_T \leftarrow \underset{k}{argmax}(viterbi[k, T])$

12: $x_T \leftarrow S_{z_T}$

13: ***for*** $j = T, T-1, \ldots, 2$ ***do***

14: $z_{j-1} \leftarrow bestpathpointer[z_j, j]$

15: $x_{j-1} \leftarrow S_{z_{j-1}}$

16: **end for**

17: **return** X

Algorithm2 : Forward algorithm

Input : $O_{seq} = [o_1, o_2, \ldots, o_T]$

Output : *log_likelihood*

1: ***for*** $i = 1, 2, \ldots, N$ ***do***:

2: $forward[0][i] \leftarrow \pi_i * b_i(o_1)$

3: ***end for***

4: ***for*** $i = 2, 3, \ldots, T$ *do*

5: ***for*** $j = 1, 2, \ldots, N$ ***do***

6: $forward[i][j] \leftarrow (forward[i-1] \cdot a) * b_j(o_i)$ */* dot product */*

7: ***end for***

8: ***end for***

9: *log_likelihood =* $log(\sum_{i=0}^{T-1} forward[i])$

10: ***return*** *log_likelihood*

추천 지점을 선정하기 위해 현재까지의 관측값에 관측 가능한 모든 관측값을 덧붙여 Hidden

Markov Model 에 적용한다. HMM 에서 관측 가능한 모든 관측열에 Viterbi 알고리즘을 통해 최적 Hidden state sequence, Forward 알고리즘을 통해 최적 Hidden state sequence 의 Log likelihood 가 구해지면, Log likelihood 가 높은 세가지 경로를 선택하여 이 경로의 Hidden state sequence 의 마지막 Hidden state 를 추천 지점으로 선택한다.

아래 Algorithm 3 은 다음 POI 를 추천하는 알고리즘의 pseudo code 이다.

Algorithm3 : Next top3 POI recommendation

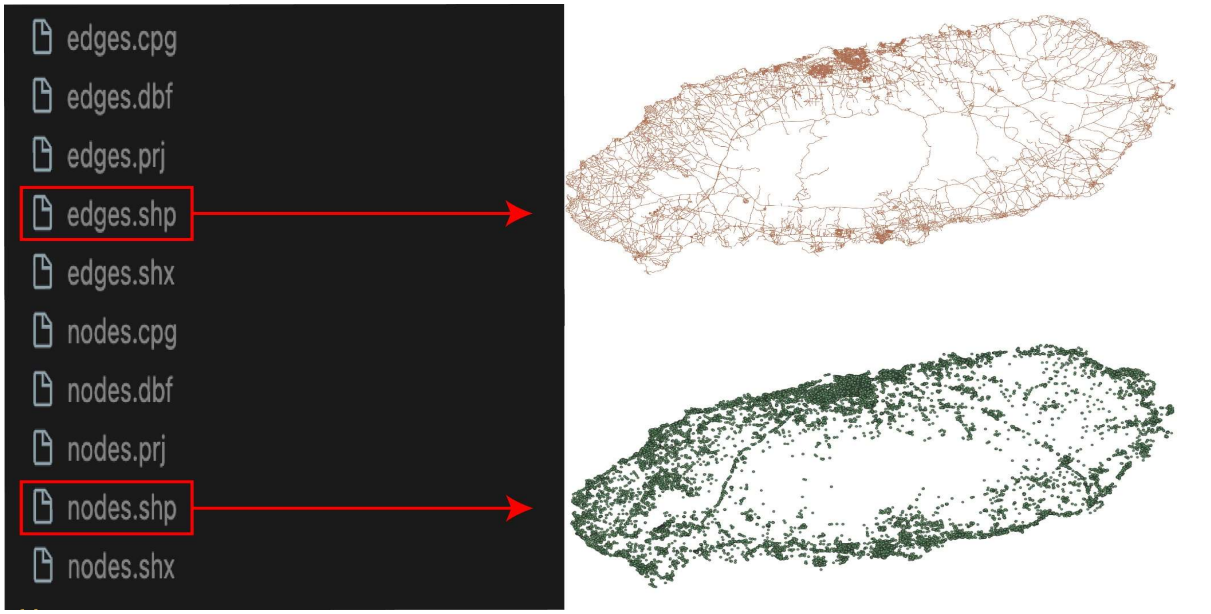
Input : $O_{seq} = [o_1, o_2, \ldots, o_T]$
Output : *top3_POI*

1: $log\_likelihood\_array \leftarrow []$
2: $hidden\_state\_sequence\_array \leftarrow []$
3: ***for*** $i = 1, 2, \ldots, M$ ***do***
4: $can\_sequence \leftarrow O_{seq} + [y_i]$
5: $log\_likelihood\_array\,[i], hidden\_state\_sequence\_array\,[i] \leftarrow HMM\_decode(can\_sequence)$
6: ***end for***
7: $top3\_idx \leftarrow find\_index(log\_likelihood\_array)$ */* find 3 indexes with a high log likelihood */*
8: $top3\_POI \leftarrow hidden\_state\_sequence\_array\,[top3\_idx][-1]$
9: ***return*** *top3_POI*

3.4. 제주도 지형 데이터 처리와 웹 기반 시각화

3.4.1. OSMnx (Open Street Map Network X)

OSMnx 는 Python 패키지로 OpenStreetMap 에서 도로망 및 기타 지형 공간적 특징을 다운로드, 모델링, 분석 및 시각화할 수 있다. 본 연구에서는 제주도 Geojson 파일을 nodes.shp 와 edges.shp 를 얻어낸 후 FMM 의 STMatching 에 활용하여 시각화한다.



[그림 17] OSMnx 를 이용한 제주도 Geojson 파일 변환

3.4.2. Flask

Flask 는 Python 으로 작성된 웹 프레임워크이다. Flask 는 특별한 도구나 라이브러리가 필요 없기 때문에 Micro Framework 라 불리지만 Flask 자체에서 구현된 것처럼 기능을 추가할 수 있는 확장성을 보유하고 있으며 이러한 확장성을 바탕으로 FMM 라이브러리의 STmatching 과 OSMnx 로 얻은 nodes, edges 를 활용하여 Leaflet, OpenMP 등과 같은 라이브러리를 사용하여 웹사이트 내에서 작동한다.

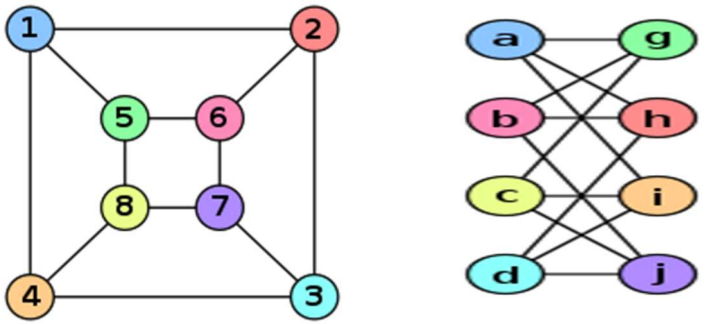
3.4.3. STmatching

STmatching 은 그래프 패턴 매칭을 개선하고 가속화하기 위한 GPU 기반의 시스템으로 그래프 분석 및 그래프 마이닝 애플리케이션에서 중요한 역할을 한다[10]. 그래프 패턴 매칭은 NP-hard 문제로 알려져 있으며, 병목 현상과 메모리 소비 문제가 발생할 수 있다.

먼저 그래프 $G = (V, E, L)$로 구성되며 V는 node 의 집합, E는 edge 의 집합, L은 edge-node 에 label 을 할당하는 labeling function 이다. $G' = (V', E', L')$은 G의 subgraph 이고 V', E'은 각각 V, E의 subset 이다. L'은 G의 labeling function L과 같다. 이 때, 그래프 패턴 매칭 문제는 주어진 Query graph Q와 이와 isomorphic 한 모든 G 의 subgraph 를 찾는 것으로 정의된다.

Isomorphic graph 란 아래 [그림 18]의 (a), (b)와 같이 node 의 순서 차이로 인해 다르게

표현되지만 같은 구조를 가지는 그래프들을 말한다.



(a) (b)

[그림 18] Isomorphic graph

STmatching 은 Backtracking 을 활용하여 그래프 패턴 매칭 문제를 해결한다. 먼저 Query graph Q의 node 에 대한 matching order 인 π를 생성한다. matching order 는 그래프 패턴 매칭을 수행할 때 어떤 순서로 노드를 일치시키는지 나타내며 다음 단계에서 일치시킬 node(즉, $\pi[l+1]$)가 이전 단계($\pi[0], \pi[1], \ldots, \pi[l]$)에서 일치한 node 중 적어도 하나와 연결되어 있도록 한다. Backtracking 알고리즘을 통해 각 단계에서 가능한 모든 matching 을 시도하고, 모든 isomorphic subgraph 를 찾을 때까지 진행한다.

Query graph Q 의 $V' = \{u_1, u_2, \ldots, u_l\}$ 일 때 매칭 알고리즘은 아래와 같다.

Algorithm 3 : Backtracking for graph pattern matching

Input: *a data graph G, a query graph Q, matching order π, recursion level l*
Output: *m (all subgraphs in G that are isomorphic to Q)*

1: ***function*** $Enumerate(G, Q, \pi, m, l)$:
2: ***if*** *l == Q.size* ***then*** *Output m,* ***return;***
3: $u \leftarrow \pi[l]$
4: $C_m(u) \leftarrow getCandidates(G, Q, \pi, m, l)$; /* get candidates for current node u */
5: ***for each*** $v \in C_m(u)$ ***do***
6: *Add v to m;*
7: $Enumerate(G, Q, \pi, m, l+1)$; /* Recursion : matching next node */
8: *Remove v from m;* /* Backtracking */

명시적 함수 호출 스택 - STmatching 은 함수 호출 스택을 명시적으로 유지하면서 재귀 절차를 시뮬레이션한다. 이를 통해 data graph G 와 Query graph Q 간의 패턴 매칭을 수행한다.

스택 기반 루프 최적화 - 스택을 활용하여 재귀적인 접근을 구현하고, 스택 기반 루프 최적화를 사용하여 병렬 처리를 향상시킨다. 이 최적화는 GPU 에서의 성능을 향상시킴으로써 병목을 완화한다.

GPU 병렬 처리 - GPU 에서 실행을 병렬화하기 위해 각 GPU 워프(warp)에 별도의 호출 스택을 할당한다. 각 워프는 독립적으로 작업을 실행하며, 그래프 패턴 매칭을 수행한다. 주요 변수들은 공유 메모리에 할당된다.

데이터 병렬 처리 - 각 워프에서 레벨(재귀 레벨)이 증가할 때, getCandidates 함수를 사용하여 후보 노드를 가져온다. 이때 GPU 워프 간에 병렬 처리가 이루어지며, 이웃 목록의 다양한 요소를 복사하고 동시에 이진 탐색을 수행한다.

3.4.4. 다음 경로 추천 및 시각화

사용자의 경로가 생겨나면서 다음 POI 를 추천하고 시각화 되는 과정은 아래 [그림 19]와 같다.

(a) (b) (c)

[그림 19] 다음 POI 추천 및 사용자의 경로 시각화

[그림 19] (a)는 현재 사용자의 초기 위치에서 Next top3 POI 를 추천하는 모습이다. 이후 (b),(c)의 노란선과 같이 사용자의 경로가 생성되면서 Next top3 POI 가 Recommendation

Module 에 의해 갱신되는 과정을 보여준다. 파란 마커 위의 숫자는 Recommendation 의 순위를 나타낸다.

4. 연구 결과 분석 및 평가

4.1. 성능 평가 방법

Next POI accuracy 는 생성한 Trajectory 를 7 : 3 으로 훈련 데이터 셋, 테스트 데이터 셋으로 분할 후 Test data 에 대해 Encoding 된 Sequence 의 마지막 포인트를 제거한다. 마지막 포인트가 제거된 Sequence 에 Algorithm 1, 2, 3 의 과정을 거쳐 top3_POI 를 Next POI 로 가정하고 실제 방문했던 POI 와 비교하여 Top 3 Accuracy 를 측정한다.

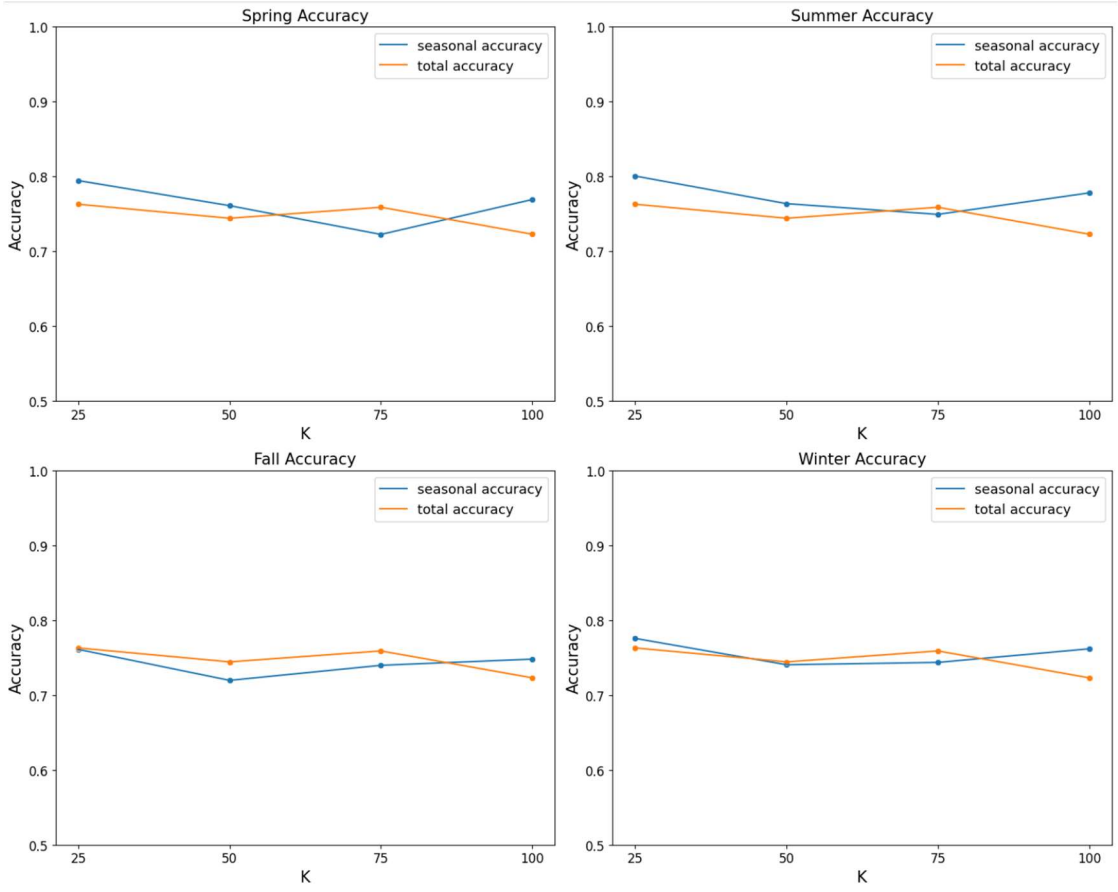
4.2. 모델 비교

앞서 1.2 절에서 언급한 [5]에 따르면 움직임이 발생한 요일과 시간대를 사용하여 HMM 의 파라미터를 구성하였다. 또한, 3.3.1. 절에서 제주도 여행객들의 방문하는 지점들은 계절에 따라 다양한 양상을 보일 것이며 여행객들의 활동 패턴에 직접적인 영향을 끼칠 것이라 가정하였다. 이에 따라 POI 를 계절마다 다르게 추출하고 계절별 Trajectory 를 생성하여 계절별 HMM 를 구성하는 것과 계절을 구분하지 않고 전체 Trajectory 를 HMM 에 구성한 모델을 비교하여 성능을 비교한다. 아래 4.2.1.은 전체 모델과 계절별로 구분한 모델 비교한 결과이다.

4.2.1. 전체 모델과 계절별로 구분한 모델 비교

[표 12] 전체 모델과 계절별 모델 Top 3 accuracy 비교

번호	K = 25	K = 50	K = 75	K = 100
전체 모델	76.29%	74.43%	75.89%	72.29%
spring 모델	79.45%	76.10%	72.26%	76.92%
summer 모델	80.07%	76.37%	74.94%	77.81%
fall 모델	76.10%	71.96%	73.97%	74.79%
winter 모델	77.56%	74.04%	74.35%	76.17%



[그림 20] 전체 모델과 계절별 모델 Top 3 accuracy 비교

두 모델의 성능 대략 75%로 유사하게 나타난 것으로 확인되었다. 즉, 계절별 차이가 존재할 것이라는 가정이 참임을 확인하지 못하였다. 따라서, 계절별로 나누지 않은 전체 Trajectory 를

사용하여 모델 학습에 사용하기로 결정하였다.

5. 결론 및 향후 연구 방향

계절 구분없이 모델을 구성했을 때와 계절별로 모델을 따로 구성했을 때 정확도는 큰 차이가 없었다. 그 이유는 클러스터링 결과 POI 가 계절에 따라 방문 여부나 빈도가 달라지는 것이 아니라, 계절에 관계없이 방문 빈도가 높은 지점으로 선정되었기 때문으로 사료된다. 또한 사용자의 관심 지점(POI)의 개수를 25 개로 선정했을 때가 가장 예측 성능이 높았는데, 이는 클러스터의 개수가 적을 수록 하나의 클러스터 안에 포함된 GPS 정보의 개수가 많아지기 때문인 것으로 생각된다.

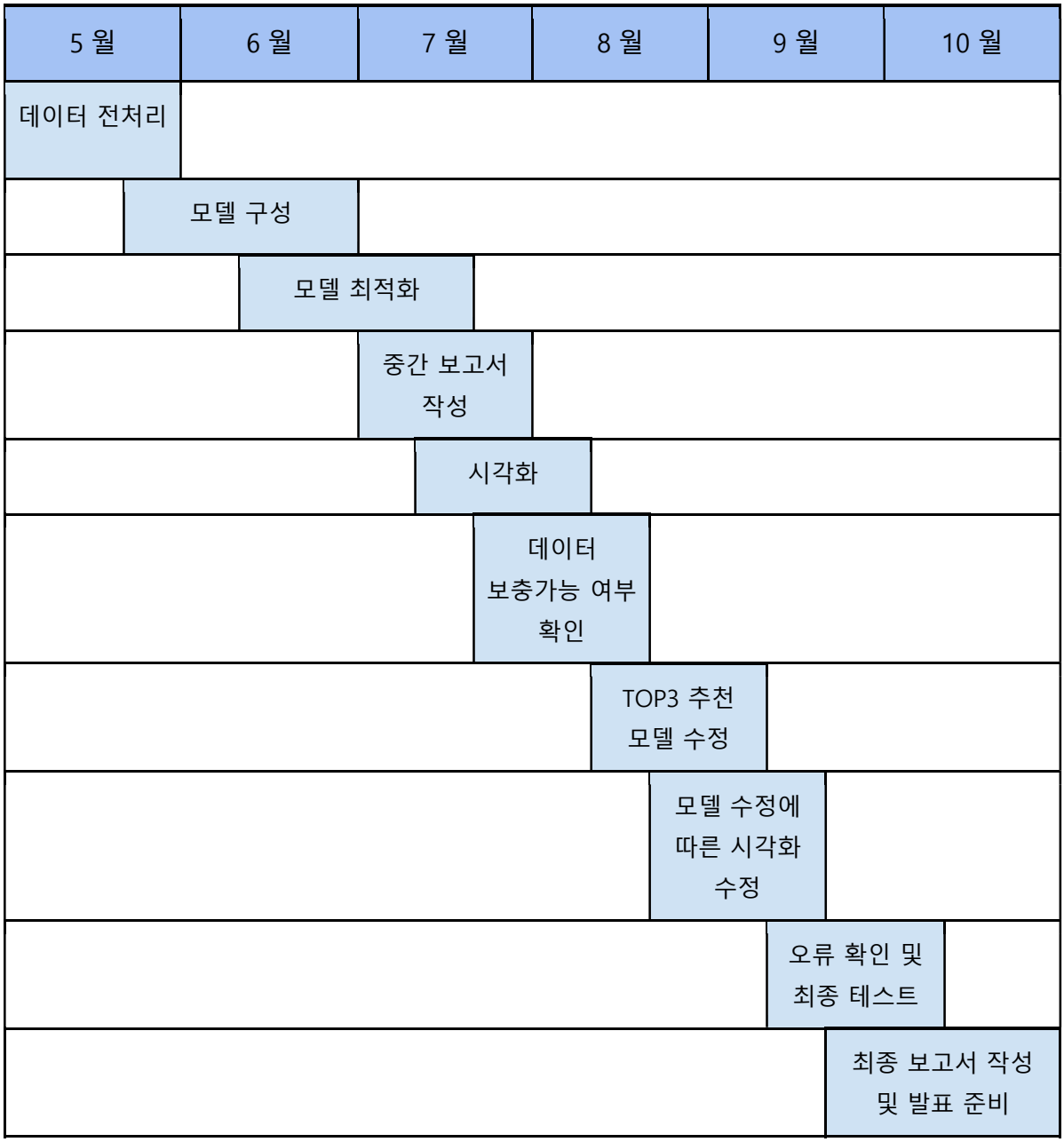
제주 데이터 허브에서 제공하는 데이터는 렌터카 운전자의 성별, 나이 등과 같은 정보는 공개되지 않아 사용할 수 있는 정보는 사용자의 로그 기록이 수집된 날짜 및 시간, 방문 장소의 위도 경도 데이터밖에 없었다. 따라서 사용자의 특성까지 고려하여 모델을 구성할 수 없었기에 여행객의 다음 방문 위치 예측에 사용하는 정보는 현재 방문 위치밖에 없다는 한계가 있다. 또한 본 시스템은 HMM 이라는 확률 모델을 사용했기 때문에 몇몇 사용자의 다음 이동 예상 지점이 현재 위치와는 많이 떨어져 있는 경우가 있었다. 주어진 시간의 한계로 인하여 적용할 수는 없었으나, 앞서 서술한 바와 같은 경우 이를 보정해줄 수 있는 방안을 적용할 필요가 있다.

6. 구성원별 역할 및 개발 일정

6.1. 구성원별 역할

이름	역할
이성무	1. 데이터 전처리 2. TOP3 추천 Hidden Markov Model 구성 3. 예측 정확도 검증
정재원	1. 데이터 전처리 2. 클러스터링을 활용해 관심 지점 군집 선정 3. 전반적인 프로젝트 요약 내용 및 TOP3 추천 시각화
하연지	1. 데이터 수집 및 전처리 2. 결과 분석 및 모델 최적화 3. 예측 정확도 검증

6.2. 개발 일정


5 월
6 월
7 월
8 월
9 월
10 월
데이터 전처리
모델 구성
모델 최적화
중간 보고서 작성
시각화
데이터 보충가능 여부 확인
TOP3 추천 모델 수정
모델 수정에 따른 시각화 수정
오류 확인 및 최종 테스트
최종 보고서 작성 및 발표 준비

7. 참고 문헌

[1] P.K. Enge, "The global positioning system: Signals, measurements, and performance," *International Journal of Wireless Information Networks,* Vol. 1, No. 2, pp. 83-105, Apr. 1994.

[2] H. Huang, G. Gartner, J.M. Krisp, M. Raubal, and N. Van de Weghe, “Location based services: ongoing evolution and research agenda,” *Journal of Location Based Services,* Vol. 12, No. 2, pp. 63-93, Apr. 2018.

[3] P. Rathore, D. Kumar, S. Rajasegarar, M. Palaniswami, and J.C. Bezdek, “A scalable framework for trajectory prediction,” *IEEE Transactions on Intelligent Transportation Systems,* Vol. 20, No. 10, pp. 3860-3874, Mar. 2019.

[4] P. Yochum, L. Chang, T. Gu, and M. Zhu, “Linked open data in location-based recommendation system on tourism domain: A survey,” IEEE Access, Vol. 8, pp. 16409-16439, Jan. 2020.

[5] M. Chen, W.Z. Li, L. Qian, S.L. Lu, and D.X. Chen, “Next POI recommendation based on location interest mining with recurrent neural networks,” *Journal of Computer Science and Technology,* Vol. 35, No. 3, pp. 603-616, May. 2020.

[6] Y. Ding, “A Location-Based Indoor Recommendation System Using A Hidden Markov Model,” *Doctoral dissertation, Auburn University,* 2016.

[7] S. Chakraverty, and A. Mithal, “IoT based weather and location aware recommender system,” *In2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence) 2018,* pp. 636-643, 2018.

[8] N. Pant, and R. Elmasri, “Detecting meaningful places and predicting locations using varied k-means and hidden Markov model,” *InProceedings of the 17th SIAM International Conference on Data Mining (SDM 2017),* pp. 27-29, 2017.

[9] Y. Yang, X. Pan, X. Yao, S. Wang, and L. Han, “PHR: A personalized hidden route recommendation system based on hidden markov model,” *InWeb and Big Data: 4th International Joint Conference, APWeb-WAIM 2020,* pp. 535-539, 2020.

[10] Y. Wei, and P. Jiang, “STMatch: accelerating graph pattern matching on GPU with stack-based loop optimizations,” *InSC22: International Conference for High Performance Computing, Networking, Storage and Analysis,* pp.1-13, 2022.

[11] Q. Lv, Y. Qiao, N. Ansari, J. Liu, and J. Yang, “Big data driven hidden Markov model based individual mobility prediction at points of interest,” *IEEE Transactions on Vehicular Technology,* Vol. 66, No. 6, pp. 5204-5215, Sep. 2016.

[12] Y. Kim, “Path Modeling and Online Learning-based Destination Prediction for Smartphone Users using Hidden Markov Model”, Master’s Thesis, Yonsei University Graduate School in South Korea, 2014