



UNIVERSITÀ DI PISA

Dipartimento di informatica

Corso di Laurea in Informatica

TESI DI LAUREA

**Implementazione e valutazione di un modello
probabilistico di data coverage per scenari di
Mobile CrowdSensing**

Relatori

Prof. Stefano Chessa

Dr. Michele Girolami

Candidato

Tommaso Colella

Anno Accademico 2019/2020

A mia nonna Paola

RIASSUNTO

I sistemi di Mobile CrowdSensing permettono di creare data set utilizzando le rilevazioni prodotte dai sensori presenti su dispositivi personali, indossabili, etc. Uno degli aspetti di interesse per tale paradigma è misurare la qualità, ovvero la rappresentatività, dei dati che possono essere acquisiti. Tale problema prende il nome di Data Coverage.

In questa tesi ci prefiggiamo di implementare e applicare un modello probabilistico di Data Coverage in un contesto urbano basandoci sullo stato dell'arte. Applichiamo il modello al dataset Microsoft Geolife, dopo averlo opportunamente arricchito per ovviare alla mancanza di dati. Mettiamo inoltre a confronto il data set ottenuto con l'originale per valutare la bontà dell'arricchimento effettuato. Nel seguito, studiamo la Data Coverage calcolata dal modello per una serie di scenari di Mobile Crowdensing nell'area metropolitana di Pechino. Gli scenari selezionati si legano a specifiche problematiche proprie delle smart cities, siano esse esigenze di monitoraggio ambientale, controllo dei flussi di traffico o di raccolta dati da punti di interesse dislocati nella metropoli. Per ciascuno scenario mostriamo i risultati del nostro esperimento per mezzo di opportune mappe di calore.

Tramite questo studio otteniamo alcuni data set di Coverage per ciascuno scenario considerato, adatti a pianificare future campagne di raccolta dati di vario genere.

Indice

Introduzione	5
1 Stato dell'arte e strumenti utilizzati	9
1.1 Mobile Crowdsensing	9
1.2 Data set di mobilità urbana	11
1.3 Librerie Utilizzate	12
1.3.1 Scikit-mobility	12
1.3.2 Pandas	13
1.3.3 OSMnx	13
1.3.4 PyYaml	13
1.3.5 Altre librerie	14
2 Un modello probabilistico per la Data Coverage	15
2.1 Data Coverage	15
2.2 Il modello probabilistico	16
3 Dataset di mobilità urbana	20
3.1 Microsoft Geolife	20
3.2 Augmented Geolife	26
3.2.1 Simulazione delle traiettorie	30
3.3 Analisi del dataset aumentato	34
4 Sperimentazioni e analisi dei risultati	41
4.1 Tre scenari applicativi	41

4.2	Monitoraggio ambientale	43
4.2.1	Generazione della griglia	44
4.2.2	Risultati del modello di coverage	45
4.3	Flussi di traffico	49
4.3.1	OSMnx e metropolitane di Pechino	49
4.3.2	Visualizzazione delle stazioni della metropolitana	49
4.3.3	Risultati del modello di coverage	50
4.4	Punti di interesse	54
4.4.1	Punti di interesse considerati	54
4.4.2	Risultati del modello di coverage	55
4.5	Analisi dei risultati ottenuti	59
5	Conclusioni	61

Elenco delle figure

1	Esempio di Data Coverage	6
1.1	Esempio di architettura MCS	10
2.1	Esempio di mappa di Coverage	16
2.2	Esempio di funzione di densità di probabilità	19
3.1	Mappa di calore del dataset	22
3.2	Alcune traiettorie di Geolife	23
3.3	Numero di punti GPS per mese	24
3.4	Punti GPS per settimana nel periodo giugno 2008/giugno 2009	24
3.5	Mappa di Calore dei punti per giorno su scala annuale	25
3.6	Punti di fermata prima e dopo il clustering	27
3.7	Medoidi	29
3.8	Grafi stradali di Pechino	31
3.9	Traiettoria interpolata	33
3.10	Grafico a barre dei dataset originario e aumentato	35
3.11	Grafico lineare settimanale dei dataset originario e aumentato	35
3.12	Mappa di calore per giorni del dataset aumentato	36
3.13	Mappa di calore del dataset aumentato	37
3.14	Number of visits e utenti attivi a confronto	38
3.15	Radius of gyration paragonato col numero di visite	39
3.16	Joint plot dei frequency e recency ranks	40
4.1	Pipeline di sperimentazione	43

4.2	Griglia visualizzata con folium	45
4.3	Risultati griglia, $\lambda = 1/100$	46
4.4	Risultati griglia, $\lambda = 1/300$	47
4.5	Risultati griglia, $\lambda = 1/700$	48
4.6	Mappa delle stazioni della metropolitana	50
4.7	Risultati metropolitana, $\lambda = 1/100$	51
4.8	Risultati metropolitana, $\lambda = 1/300$	52
4.9	Risultati metropolitana, $\lambda = 1/700$	53
4.10	Mappa dei punti di interesse	55
4.11	Risultati POIs, $\lambda = 1/100$	56
4.12	Risultati POIs, $\lambda = 1/300$	57
4.13	Risultati POIs, $\lambda = 1/700$	58
4.14	Esempio di barriera fisica	60
5.1	Posizionamento di UAV Base Stations	63

Introduzione

La diffusione massiccia di dispositivi indossabili, e la loro sempre maggiore potenza di calcolo, hanno dato origine ad un recente paradigma computazionale denominato Mobile Crowdensing (o MCS). Tale paradigma permette di raccogliere dati sull’ambiente che ci circonda e di analizzarli. Le applicazioni del MCS sono molteplici. Nel caso delle Smart Cities, il MCS può contribuire a migliorarne la vivibilità, la sicurezza e la interconnettività, sfruttando le capacità sensoriali e di calcolo dei dispositivi.

Grazie al Crowdensing è possibile creare data set utilizzando rilevazioni ottenute dai dispositivi personali degli utenti. Tali dataset possono poi essere inviati ad un server remoto che svolga analisi approfondite su di essi e restituisca delle metriche utili. Molti gruppi di ricerca (accademici e industriali) si sono interessati al Crowdensing e hanno prodotto negli anni una moltitudine di articoli che esplorano i vari aspetti di questa metodologia di raccolta dati. Dedicheremo la Sezione 1.1 del Capitolo 1 a presentare il Crowdensing con la dovuta profondità, mostrando i principali ambiti in cui è stato impiegato e presentando gli articoli seminali che ne hanno determinato l’origine. Successivamente, nella Sezione 1.2, daremo una prima descrizione dei data set di mobilità urbana, sottolineandone l’utilità nell’ambito del Crowdensing e presentando i più noti tra di essi.

Uno dei problemi strettamente legati al MCS è quello di misurare la copertura dei dati che possiamo aspettarci per una determinata campagna di raccolta. Per probabilità di copertura, o Data Coverage, intendiamo la probabilità che gli utenti che partecipano volontariamente alla nostra campagna di Crowdensing si avvicinino alle locazioni di interesse per la campagna di MCS in modo da poter acquisire i dati. Nel

seguito ci riferiremo ad essa chiamandola alternativamente "Data Coverage", "probabilità di copertura", o anche solo "Coverage", quando il significato risulti chiaro dal contesto.

Per modellare la probabilità di copertura utilizzeremo una funzione di densità di probabilità che andremo a descrivere nel Capitolo 2, in cui approfondiremo maggiormente il concetto di Data Coverage. La Figura 1 mostra una rappresentazione grafica della Data Coverage ottenuta per mezzo di una meshgrid.

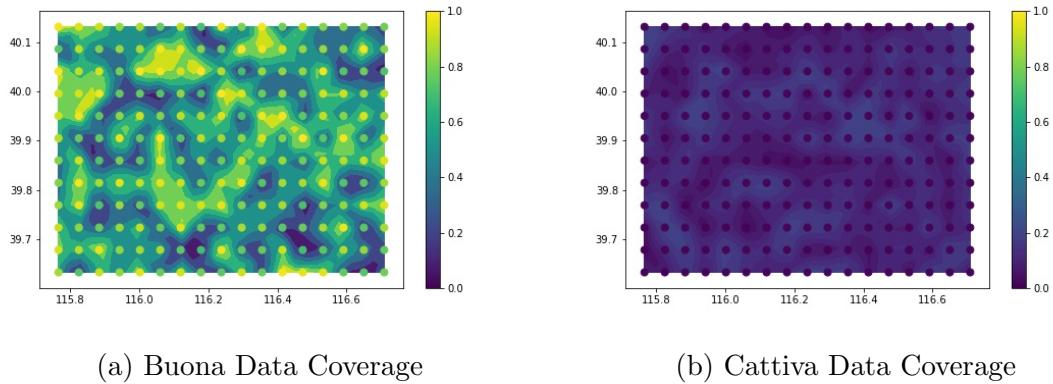


Figura 1: Esempi di Data Coverage su una griglia di locazioni

La presente tesi ha come obiettivo quello di implementare una misura di copertura utilizzando un modello probabilistico che tenga conto della mobilità degli utenti che partecipano alla campagna. Tale modello deve essere facilmente utilizzabile in scenari di raccolta dati differenti. Al fine di misurare la Data Coverage, è stato analizzato ed esteso il data set GeoLife [1, 2, 3]. Il data set offre tracciati GPS raccolti su base volontaria in un periodo temporale molto ampio; questo porta a problematiche di vario tipo: la campionatura dei punti gps non è regolare e vi sono intere settimane nel nostro data set che mancano di dati significativi per la scarsa partecipazione degli utenti; nel corso della tesi tale data set è stato esteso, generando in questo modo Augmented Geolife, di cui daremo una descrizione approfondita nel Capitolo 3, dopo aver parlato del data set originale.

Obiettivi della Tesi

La presente tesi ha due obiettivi principali:

1. Analizzare ed estendere un data set di mobilità
2. Sviluppare un modello di Data Coverage

In merito al primo obiettivo la tesi ha permesso di ottenere un data set aumentato per la città di Pechino, contenente una quantità di dati sufficienti al prosieguo delle analisi. In merito al secondo obiettivo, lo sviluppo del modello di Data Coverage e la sua applicazione a differenti scenari applicativi, ci ha permesso di ottenere delle mappe di probabilità di copertura dell'area oggetto della campagna di Crowdsensing. Tali mappe ci possono aiutare a pianificare future campagne di acquisizione dati, e sono sufficientemente chiare da permettere una identificazione rapida ed efficace delle zone che risultano essere scoperte, zone per cui dovremo probabilmente prevedere campagne di raccolta ad hoc.

La misurazione della Data Coverage è di estrema importanza per una serie di motivi: oltre a permetterci di valutare in modo quantitativo l'efficacia di una campagna di crowdsensing, ci fornisce anche un modo per pianificare ulteriori meccanismi di raccolta dati che siano complementari al paradigma di MCS. Uno tra questi potrebbe essere l'utilizzo di droni da inviare su richiesta nelle zone scarsamente coperte, in modo da migliorarne il parametro di coverage. Un altro dei vantaggi della misurazione della Data Coverage è quello di dare informazioni su quali siano le aree urbane scarsamente popolate o trafficate. Grazie alla visualizzazione grafica dei data set di probabilità ottenuti per i vari scenari, è possibile comprendere facilmente tali risultati sperimentali.

Metodologia e Risultati

Metodologia usata

Per raggiungere i nostri scopi ci siamo avvalsi di una serie di librerie software all'avanguardia nel campo della mobilità, dell'analisi dati e della visualizzazione di questi

ultimi. Nella Sezione 1.3 presenteremo le librerie che ci hanno aiutati maggiormente nella nostra analisi. Tra di esse è sicuramente importante ricordare scikit-mobility [4], strumento grazie al quale abbiamo elaborato con semplicità ed efficacia i dati spaziali. Nel corso della tesi abbiamo progettato e sviluppato processi simulativi per arricchire il data set Geolife con ulteriori traiettorie. L'utilizzo di strumenti di visual analytics ci ha infine permesso di rappresentare le proprietà dei dati spaziali che abbiamo analizzato.

Risultati ottenuti

Lo studio ci ha condotti allo sviluppo di opportune analitiche per misurare le proprietà del data set GeoLife, tra di esse citiamo il numero di utenti attivi in un determinato periodo temporale, i punti di partenza e di arrivo delle traiettorie degli utenti, i flussi di spostamenti sulla città di Pechino.

Abbiamo inoltre sviluppato un simulatore di tracce che, per mezzo di interpolazioni e calcoli di distanze, permette di generare serie temporali di traiettorie sintetiche con una certa precisione basandoci su punti di partenza e di arrivo.

Altro importante risultato ottenuto è l'implementazione di un modello di Data Coverage basato sulla probabilità di detour degli utenti.

I risultati della applicazione del modello ai vari scenari sono presentati nel Capitolo 4 assieme a delle mappe di calore che ne riassumono graficamente il significato. La rappresentazione grafica ci mostra intere zone completamente scoperte per ciascuno dei tre scenari che abbiamo preso in esame, e ci invita a indagare soluzioni future atte a migliorare la Data Coverage nella zona che abbiamo preso come riferimento. Di tali sviluppi futuri facciamo menzione nel Capitolo 5.

Capitolo 1

Stato dell'arte e strumenti utilizzati

In questo capitolo, dopo aver presentato il paradigma del crowdsensing e i data set di mobilità urbana, ci dedicheremo alla descrizione delle librerie che abbiamo impiegato per il nostro studio.

1.1 Mobile Crowdsensing

Il paradigma del Mobile Crowdsensing si è affermato nell’ultima decade per la raccolta dati in ambiente urbano. Grazie a tale paradigma si possono raccogliere dati dai sensori presenti nei dispositivi mobili e indossabili che ogni giorno portiamo con noi. Il concetto di Crowdsensing è apparso per la prima volta nel 2006, ma senza il nome che lo contraddistingue al giorno d’oggi [5]. Sebbene spesso il termine sia strettamente correlato all’utilizzo di dispositivi cellulari e simili, è interessante notare come in realtà il primo articolo in cui esso appare ne faccia un uso più generico, non strettamente legato a telefoni cellulari e dispositivi mobili [6]. La Figura 1.1 mostra un esempio di architettura per il Mobile Crowdsensing.

Gli ambiti applicativi di questa tecnologia sono molteplici: si spazia da utilizzi di tipo medico a monitoraggio ambientale, raccolta di dati sulle abitudini alimentari, monitoraggio del traffico e molti altri [5].

Dal momento che i dati devono essere raccolti da privati cittadini che usano i propri dispositivi, e che ne serve una grande quantità ai fini della successiva ricerca scientifica,

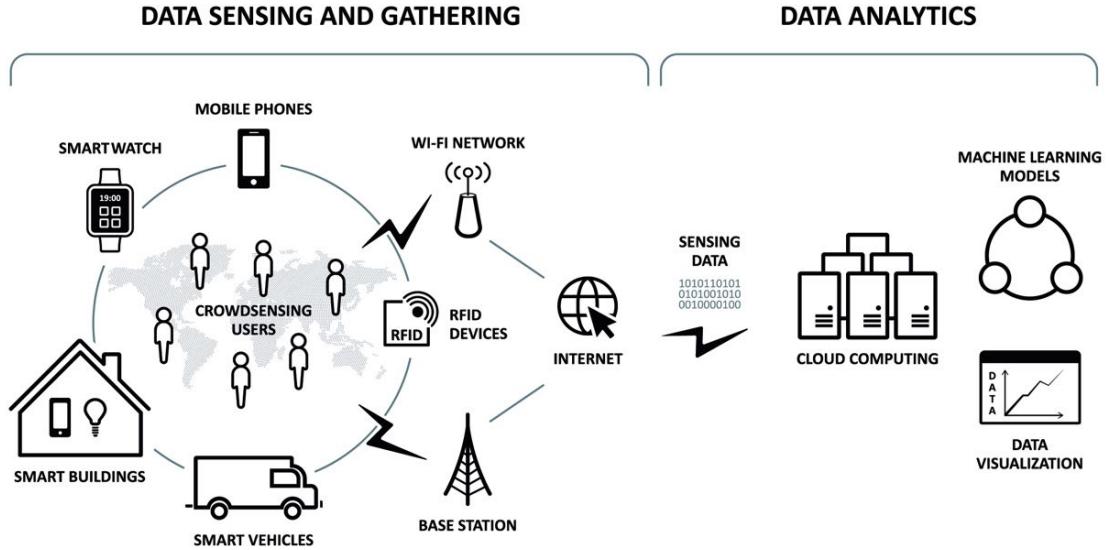


Figura 1.1: Un esempio di architettura per MCS

si rende spesso necessaria l'erogazione di opportuni incentivi per invogliare gli utenti a partecipare alla campagna di raccolta dati. La comunità scientifica ha proposto varie tipologie di incentivo, siano essi a carattere monetario, sociale o di intrattenimento puro[7].

Parallelamente a questa problematica, è sorta anche quella della privacy: I dati vanno opportunamente anonimizzati e trattati con la dovuta cautela: le persone sono spesso preoccupate dal modo in cui aziende private e governi nazionali gestiscono tali dati, ed è necessario rassicurarle circa il loro impiego e i meccanismi con cui vengono immagazzinati. Al tempo stesso è essenziale che la comunità scientifica sottolinei le problematiche principali che possono sorgere in questo ambito e si occupi di risolverle adeguatamente[8].

Molti studi hanno mostrato come il Crowdsensing possa contribuire sensibilmente all'aumento del benessere dei cittadini e della loro vita quotidiana, contribuendo allo sviluppo delle smart cities del futuro: il Crowdsensing può infatti permettere di integrare reti di sensori già presenti a livello cittadino, oltre ad aprire a scenari di ogni tipo in ambito urbano, come ad esempio il monitoraggio delle vibrazioni dei ponti per mezzo degli accelerometri presenti sui telefoni[9, 10, 11].

Un progetto scientifico di MCS significativo e di lunga durata in Italia è stato

il ParticipAct Living Lab, progetto dell’Università di Bologna lanciato nel 2014 allo scopo di raccogliere dati dagli smartphone di circa 200 studenti. ParticipAct è stato uno dei primi esperimenti di MCS su larga scala ed ha permesso di creare un data set di mobilità per la città di Bologna [5, 12].

Negli ultimi anni, molti studi si sono focalizzati sul combinare il MCS con il paradigma del Multiacces Edge Computing (MEC). Il paradigma risultante da questa combinazione è chiamato Human-enabled Edge Computing. Grazie a questo nuovo paradigma è possibile usare i device interconnessi in modo opportunistico, utilizzandoli come se fossero nodi fissi nella rete di raccolta dati. Queste possibilità sono state esplorate utilizzando il data set di ParticipAct[13, 14, 15] e hanno permesso di teorizzare modelli di Data Coverage per questo specifico paradigma[16].

1.2 Data set di mobilità urbana

Visti gli sviluppi recenti del MCS, assumono sempre di più un ruolo essenziali i data set di mobilità urbana; senza di essi, infatti, sarebbe impossibile studiare modelli di Data Coverage come quello che abbiamo implementato.

Tali data set, tra cui ricordiamo i già citati ParticipAct[12] e Geolife[1, 2, 3], sono composti da serie temporali contenenti la posizione dell’utente coinvolto, il timestamp e altre informazioni accessorie più o meno utili a seconda dell’analisi che si sta effettuando. Dato il particolare momento storico, segnato dalla pandemia globale di COVID-19, il mondo accademico si è affrettato a raccogliere dati di mobilità legati a questa situazione eccezionale; sono così stati prodotti alcuni data set di mobilità specifici a seguito del COVID-19[17, 18, 19], grazie ai quali sono stati portati avanti innumerevoli studi accademici a livello italiano[17] e internazionale[20].

Di seguito è riportato un estratto di una delle traiettorie registrate nel data set GeoLife a titolo di esempio. Per brevità sono stati omessi alcuni campi inutili o ridondanti.

Datetime	Latitude	Longitude	Altitude
2009-01-03 01:21:34	39.974294	116.399741	492.000000
2009-01-03 01:21:35	39.974292	116.399592	492.000000
2009-01-03 01:21:36	39.974309	116.399523	492.000000
2009-01-03 01:21:38	39.974320	116.399588	492.000000
2009-01-03 01:21:39	39.974365	116.399730	491.000000

Abbiamo impiegato i dati estratti da GeoLife come input per il modello di Data Coverage che abbiamo implementato, ricavandone risultati sperimentali per diversi scenari applicativi. Per una analisi approfondita di GeoLife si rimanda al Capitolo 3, in cui studieremo il data set per ottenere metriche sulla partecipazione degli utenti e sulla quantità dei dati.

1.3 Librerie Utilizzate

Durante la stesura della presente tesi, abbiamo utilizzato una varietà di librerie software che rappresentano lo stato dell'arte nei rispettivi campi. Il linguaggio che abbiamo scelto per l'implementazione del modello e le analisi è Python 3.8 [21], per via della sua brevità e per la facilità con cui vi si riesce a scrivere codice anche molto complesso.

L'ambiente di sviluppo impiegato è Jupyter Lab [22], grazie al quale si può sviluppare codice interattivamente, visualizzando subito gli effetti della computazione. Contestualmente a Jupyter è stato utilizzato anche Anaconda [23], che permette di gestire in modo compartimentale le dipendenze per evitare conflitti che potrebbero scaturire da un'errata installazione dei suddetti nel filesystem.

Di seguito riportiamo le librerie impiegate.

1.3.1 Scikit-mobility

Scikit-mobility è una libreria specializzata per l'analisi, la generazione e la stima del rischio di dati di mobilità [4]. Tale libreria è lo stato dell'arte in quanto ad analisi

dei data set di mobilità, e permette di svolgere con facilità analisi approfondite che altrimenti richiederebbero molto più tempo. Grazie a scikit-mobility abbiamo potuto calcolare una moltitudine di metriche sul data set di GeoLife per poter valutare la bontà dei dati e il loro miglioramento dopo l'arricchimento. Abbiamo inoltre impiegato scikit-mobility nella nostra pipeline di arricchimento dei dati, utilizzandola soprattutto per calcolare i punti di partenza e di fermata delle traiettorie.

1.3.2 Pandas

Pandas [24, 25] è una libreria per l'analisi e la manipolazione di dati. Grazie ad essa si può operare con semplicità su tabelle e serie temporali, senza rinunciare alla velocità propria dei linguaggi compilati: gran parte della libreria è infatti scritta in C e permette tempi di esecuzione molto rapidi. Abbiamo usato pandas in tutto il codice che abbiamo scritto per la tesi, sfruttandone più volte le capacità di reshaping dei dati, unione e pulizia delle tabelle.

1.3.3 OSMnx

OSMnx [26] è una libreria che permette di recuperare, visualizzare e analizzare le reti stradali in forma di grafo memorizzate nel database di OpenStreetMap [27]. Tramite OSMnx abbiamo inserito nel nostro codice la rete stradale di Pechino, calcolando il cammino minimo tra punti di partenza e di arrivo e estraendone i nodi necessari per la costruzione delle traiettorie sintetiche.

1.3.4 PyYaml

PyYaml [28] è un parser ed emitter per file di tipo .yaml scritto appositamente per Python. Lo abbiamo sfruttato per il parsing dei file di configurazione del nostro codice, onde evitare l'hardcoding di valori che possono essere così facilmente modificati.

1.3.5 Altre librerie

Tra le innumerevoli altre librerie che abbiamo impiegato, vogliamo citare folium [29], usata per produrre mappe interattive, scikit-learn [30], che ci ha aiutati durante la generazione delle traiettorie sintetiche, seaborn [31] e Datashader [32], utilizzate per generare alcune delle figure riportate nella tesi. NetworkX [33] è stata impiegata per il calcolo dei cammini tra punti del grafo di Pechino. Abbiamo sfruttato GeographicLib [34] per l'interpolazione delle traiettorie tramite il calcolo della distanza geodetica. Nella generazione della griglia per il monitoraggio ambientale abbiamo sfruttato le librerie Shapely [35] e PyProj [36].

Capitolo 2

Un modello probabilistico per la Data Coverage

In questa sezione presentiamo un modello probabilistico per calcolare la Data Coverage. Cominciamo spiegando il concetto di Data Coverage, sottolineando l'importanza che assumono i punti del dataset nel calcolarla e teorizzando tre diversi scenari applicativi. Infine parliamo nello specifico del modello probabilistico che abbiamo implementato nel corso della tesi.

2.1 Data Coverage

La Data Coverage è uno dei concetti più importanti per il paradigma del MCS. Nel nostro caso, con Data Coverage per una locazione l_h intendiamo la probabilità p di riuscire a collezionare dati da quella specifica locazione, ovverosia $p = 1$ se almeno un utente ha già visitato la nostra locazione di e vi ha raccolto dei dati. In tutti gli altri casi la Data Coverage assume un valore p e vale $0 \leq p < 1$. Cercare di massimizzare il valore di Data Coverage per tutte le locazioni di raccolta è una delle sfide più interessanti della ricerca nel campo del MCS, e vi sono molti modi di procedere in tal senso. La Figura 2.1 mostra una mappa di calore rappresentante la Data Coverage calcolata grazie al dataset aumentato. La analizzeremo più in dettaglio nel Capitolo 4, dedicato alle sperimentazioni. Nel grafico possiamo osservare le locazioni di raccolta. Quelle

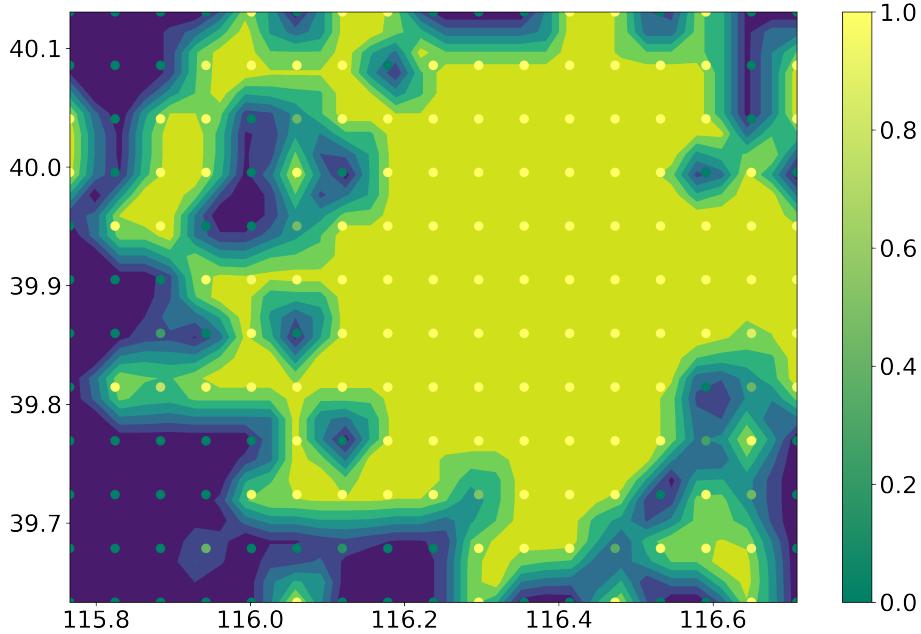


Figura 2.1: Un esempio di mappa di Coverage.

posizionate nelle zone più calde sono chiaramente quelle con una maggiore probabilità di copertura.

I dati di mobilità giocano un ruolo di primaria importanza per valutare la Data Coverage delle locazioni, come vedremo nel prosieguo della tesi.

2.2 Il modello probabilistico

Per ciascuno degli scenari applicativi che vedremo in seguito, abbiamo implementato e utilizzato lo stesso modello probabilistico. Il modello descritto nel seguito del capitolo è stato sviluppato dal gruppo di ricerca WNLab del CNR-ISTI. Riportiamo tale modello al fine di chiarire l'obiettivo della tesi e descrivere la sua implementazione e valutazione.

Il modello generico Definiamo adesso il concetto di copertura di una locazione. Le regioni coperte saranno per noi quelle dalle quali potremo raccogliere dati, grazie agli utenti che le attraversano. Viceversa, le locazioni scoperte saranno quelle non attraversate dagli utenti.

Sia $|L| = H$ la cardinalità dell'insieme delle locazioni. Ciascuna locazione $l_h = (x, y)$ è identificata dal centroide del cerchio con centro in (x, y) e raggio r . Assumiamo l'esistenza di K utenti distinti $U = u_1, \dots, u_k$, con $|U| = K$, che si muovono nella regione di interesse per la campagna di MCS. Ciascuno di essi segue delle traiettorie, ovvero delle sequenze ordinate di punti geo-referenziati. Sia adesso T_i l'insieme delle traiettorie dell'utente u_i , in cui con $t_{ij} \in (x, y_t, \dots, (x, y)_{t+\delta})$ indichiamo l'insieme delle coordinate georeferenziate. Data la locazione l_h e il punto $(x, y)_t \in t_{ij}$, $\delta(l_h, t_{ij})$ è una misura di distanza. Modelliamo quindi la Data Coverage di una locazione come la probabilità che un utente accetti di fare una deviazione verso tale locazione. Assumiamo che tale probabilità aumenti man mano che la distanza tra l'utente e la locazione l_h diminuisce, ovverosia che l'utente sia tanto più propenso ad accettare la deviazione quanto più si trova in prossimità di l_h .

Definiamo adesso la variabile aleatoria $X_{ij}^h : \Omega \mapsto \mathbb{R}^+$, in cui Ω è definito come l'insieme di eventi nella forma: la locazione l_h è coperta dall'utente u_i a distanza $x = \delta(l_h, t_{ij})$ per un certo punto geo-referenziato di $t_{ij} \in T_i$. La variabile aleatoria X_{ij}^h associa ad ogni evento in Ω un valore numerico che è la distanza x tra l_h e un punto geo-referenziato della traiettoria. I valori assunti dalla variabile aleatoria sono continui su \mathbb{R}^+ , di conseguenza la probabilità $P(X_{ij}^h \in [x, \Delta x])$ per un certo incremento Δx è data da:

$$\int_x^{\Delta x} f_{X_{ij}^h}(x) dx \quad (2.1)$$

Ora definiamo la probabilità che la locazione l_h sia coperta dall'utente u_i ad una qualunque distanza da l_h . Più precisamente misuriamo la probabilità che l'utente u_i effettui una deviazione fino a l_h a partire da una qualsiasi delle sue traiettorie. Definiamo la variabile aleatoria $X_i^h : \Gamma \mapsto \mathbb{R}^+$, con Γ insieme degli eventi nella forma: la locazione l_h è coperta dall'utente u_i ad una qualsiasi distanza x da l_h . Assumiamo che le variabili aleatorie X_{ij}^h siano tutte indipendenti, dal momento che la probabilità di deviazione per un utente è indipendente dalla traiettoria che sta seguendo. La probabilità che un

evento in Γ accada è data da:

$$P(X_i^h \in [x, \Delta x]) = 1 - \prod_{\forall x \in D_i^h} 1 - P(X_{ij}^h \in [x, x + \Delta x]), \forall \Delta x \quad (2.2)$$

Con $D_i^h = \{\delta(l_h, t_{ij}), \forall x \in t_{ij} \in T_i\}$, insieme delle distanze tra le traiettorie di u_i e l_h .

Dalla equazione (2.2) possiamo derivare la probabilità che una locazione l_h sia coperta da uno qualsiasi degli utenti in U . Definiamo la variabile aleatoria $X^h : \theta \mapsto \mathbb{R}^+$, in cui θ è definito come l'insieme degli eventi nella forma: la locazione l_h è coperta da un qualunque utente $u_i \in U$, a una qualsiasi distanza x da l_h .

La probabilità degli eventi in θ è data da:

$$\begin{aligned} P(X^h \in [x, x + \Delta x]) &= 1 - \prod_{\forall i \in U} P(X_i^h \in [x, x + \Delta x]) \\ &= 1 - \prod_{\forall i \in U} \left(1 - \prod_{\forall x \in D_i^h} (1 - P(X_{ij}^h \in [x, x + \Delta x]))\right) \end{aligned} \quad (2.3)$$

Distribuzione esponenziale Uno dei possibili modi per definire la probabilità che un utente accetti di fare una deviazione è usare una funzione di densità esponenziale. Possiamo quindi specializzare il nostro modello generico assumendo che la densità di $P(X_{ij}^h \in [x, \Delta x])$ segua un andamento di tipo esponenziale di scala $1/\lambda$:

$$f_{X_{ij}^h}(x) = \lambda \frac{1}{e^{\lambda x}}, x \geq 0 \quad (2.4)$$

Così facendo possiamo modellare l'incremento della probabilità di Data Coverage con la diminuzione della distanza tra la traiettoria e la locazione di interesse.

La Figura 2.2 riporta un esempio di funzione di densità di probabilità esponenziale. La scala è l'inverso del tasso di distribuzione λ , e corrisponde al valor medio per la distribuzione esponenziale.

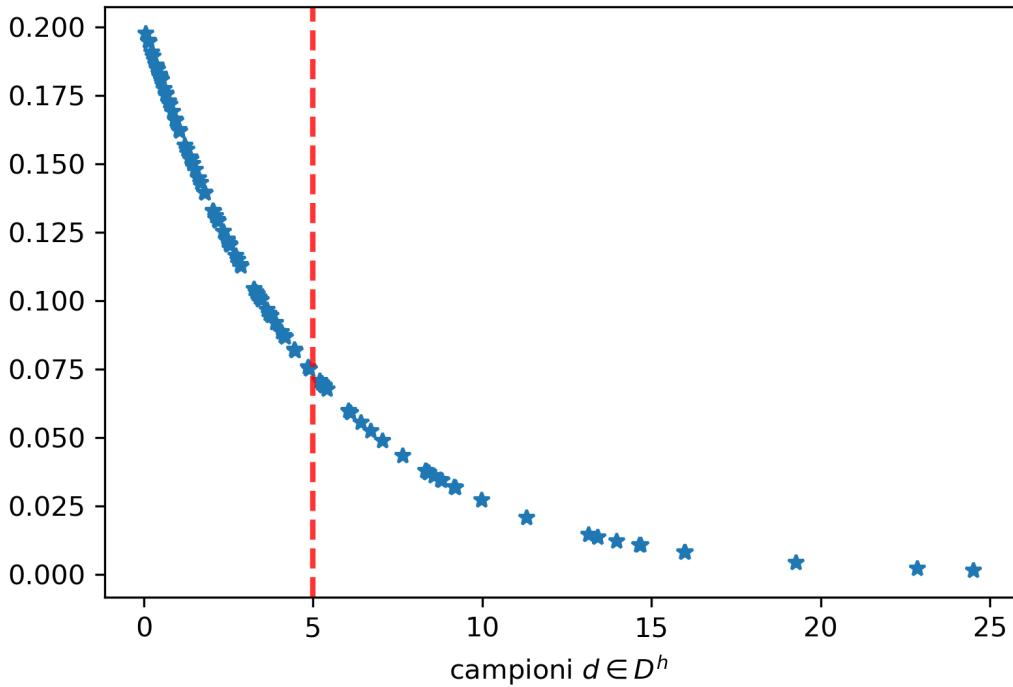


Figura 2.2: Un esempio di funzione di densità di probabilità esponenziale di scala $1/\lambda = 5$. Per valori di $x \leq 5$, la densità sale.

Dato il modello generico (2.3), con 2.4 otteniamo:

$$\begin{aligned}
 P(X^h \in [x, x + \Delta x]) &= 1 - \prod_{\forall i \in U} \left(1 - \prod_{\forall x \in D_i^h} \left(1 - \int_x^{x+\Delta x} \lambda \frac{1}{e^{\lambda x}} dx \right) \right) \\
 &= 1 - \prod_{\forall x \in D_h^i} \left(1 - \left(\frac{1}{2\lambda} e^{-\lambda x} + c \right) \right)
 \end{aligned} \tag{2.5}$$

con $D_i^h = \{\delta(l_h, t_{ij}), \forall x \in t_{ij} \in T_i\}$.

Il modello descritto da (2.5) è infine quello che abbiamo utilizzato per la nostra sperimentazione.

Capitolo 3

Dataset di mobilità urbana

In questo capitolo descriviamo i risultati della nostra analisi sul dataset di mobilità urbana di Microsoft Research Asia chiamato GeoLife [1, 2, 3], precedentemente introdotto nel Capitolo 1. Per poter valutare la probabilità di Data Coverage di una locazione, è infatti essenziale avere accesso a tracce di mobilità con una certa granularità temporale, in modo da poter valutare secondo un criterio opportuno quale sia la probabilità che gli utenti accettino di fare una deviazione dal proprio percorso per andare a raccogliere i dati da una delle locazioni oggetto della campagna di MCS.

Nel prosieguo del capitolo, analizziamo il processo di estensione del data set tramite la generazione di traiettorie di mobilità sintetiche. Il capitolo illustra il processo di generazione ed i risultati ottenuti.

3.1 Microsoft Geolife

Il contesto del progetto GeoLife GeoLife è stato un progetto di Microsoft Research Asia volto a creare un servizio di social networking collaborativo [3] dai vari scenari applicativi:

1. Condividere esperienze di vita degli utenti sulla base delle loro traiettorie GPS
2. Dare raccomandazioni di viaggio di tipo generico
3. Dare raccomandazioni personalizzate su nuove amicizie o luoghi da visitare

Il progetto è durato per oltre 5 anni, dall'aprile del 2007 all'agosto del 2012. Come specificato dagli autori [1], gli utenti erano provvisti di vari dispositivi GPS-enabled personali, come telefoni e logger, con i quali hanno registrato le proprie traiettorie, eventualmente etichettandole con il mezzo di trasporto utilizzato per lo spostamento. Grazie a tale social network è stato possibile raccogliere e aggregare le traiettorie GPS degli utenti allo scopo di utilizzarle per successivi studi. Per mezzo delle traiettorie si possono derivare molte relazioni utili tra gli utenti e l'ambiente che li circonda. Il gruppo di ricerca di Microsoft Asia guidato da Yu Zheng ha prodotto una serie di articoli sulle possibilità offerte dal dataset, in termini di analisi della mobilità degli utenti [1, 2] e di possibili relazioni tra essi [3].

Dal momento che questo dataset raccoglie molte traiettorie di utenti diversi, tutte geo-referenziate e provviste di timestamp, è particolarmente adatto ad essere impiegato nel nostro modello, descritto nel Capitolo 2.

I numeri di GeoLife Nel dataset sono raccolte un totale di 17621 traiettorie appartenenti a 182 utenti, per un totale di circa 2.48×10^7 singoli punti georeferenziati. Il 91% delle traiettorie registrate ha una frequenza di campionamento elevata nell'ordine di pochi secondi o pochi metri, al contrario del rimanente 9%. I punti GPS registrati appartengono principalmente all'area metropolitana di Pechino, sebbene alcuni siano stati registrati in varie località europee e asiatiche. Per i nostri scopi ci siamo chiaramente limitati a prendere in considerazione i punti nella zona di Pechino. La Figura 3.1 mostra una rappresentazione grafica dei punti presenti nell'area della metropoli cinese ottenuta grazie alla libreria Datashader [32].

Gli autori hanno scelto di utilizzare il formato *.plt*, proprio dei file di plotting e usato da molti strumenti professionali tra i quali quelli di Autodesk Inc [37], per serializzare le traiettorie. I file *.plt* sono a tutti gli effetti assimilabili a dei file *.csv* eccetto un header di 6 righe che può essere rimosso in quanto completamente inutile, su ammissione stessa degli autori [38]. I file delle traiettorie contengono una serie di metriche associate a ciascun punto:

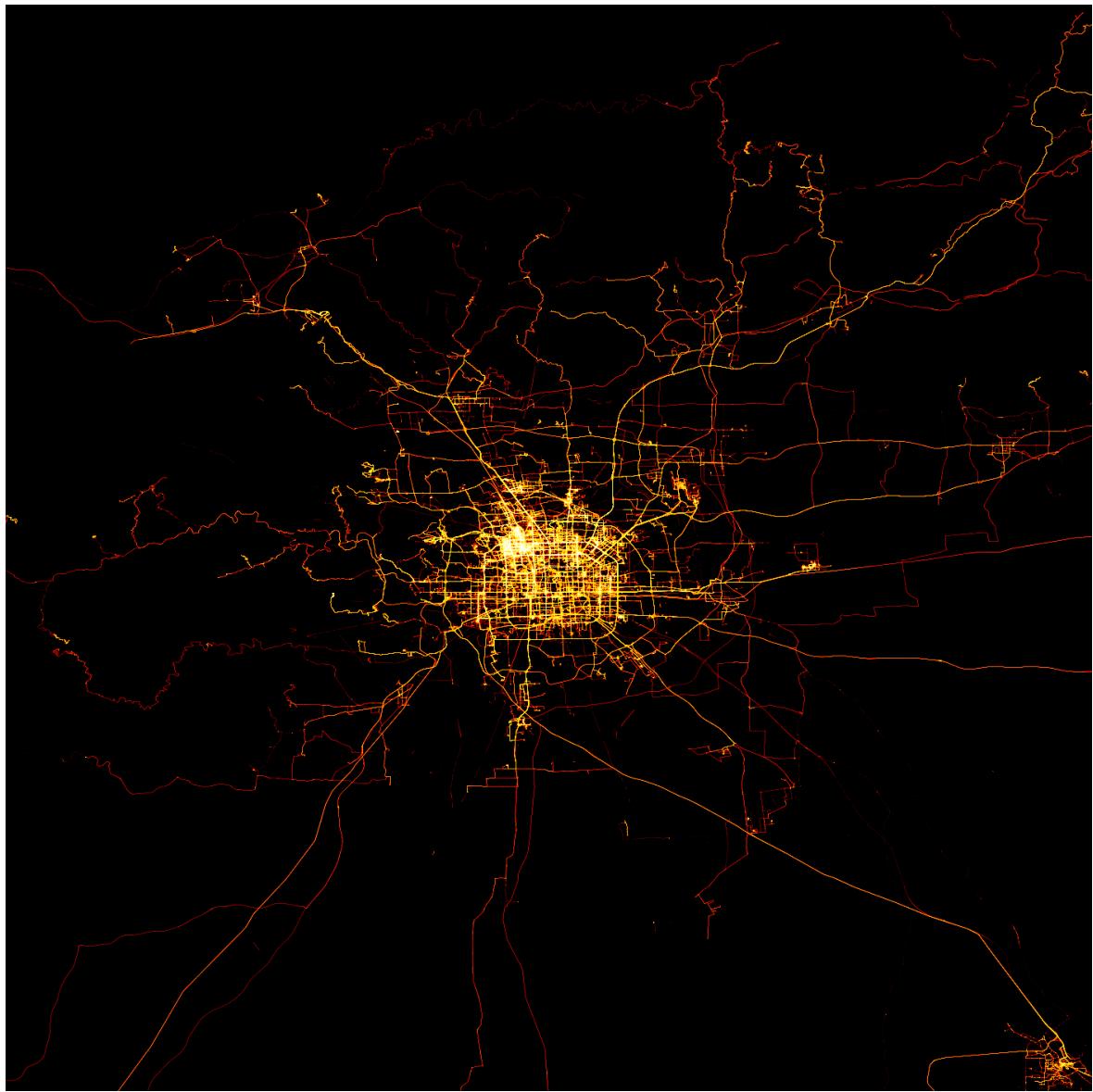


Figura 3.1: Una mappa di calore ottenuta con Datashader relativa a tutti i dati presenti nel dataset GeoLife, ristretta all'area di Pechino.

1. Latitudine e longitudine espresse in gradi decimali
2. Altitudine
3. Data espressa in numero di giorni da un epoch (il 30/12/1899)
4. Data in formato di stringa
5. Tempo in formato di stringa

Vi è inoltre un campo inutile, settato a 0 in tutto il dataset [38]. In Figura 3.2 sono presenti alcune traiettorie appartenenti a vari utenti, stampate grazie a Folium[29] e a scikit-mobility[4].

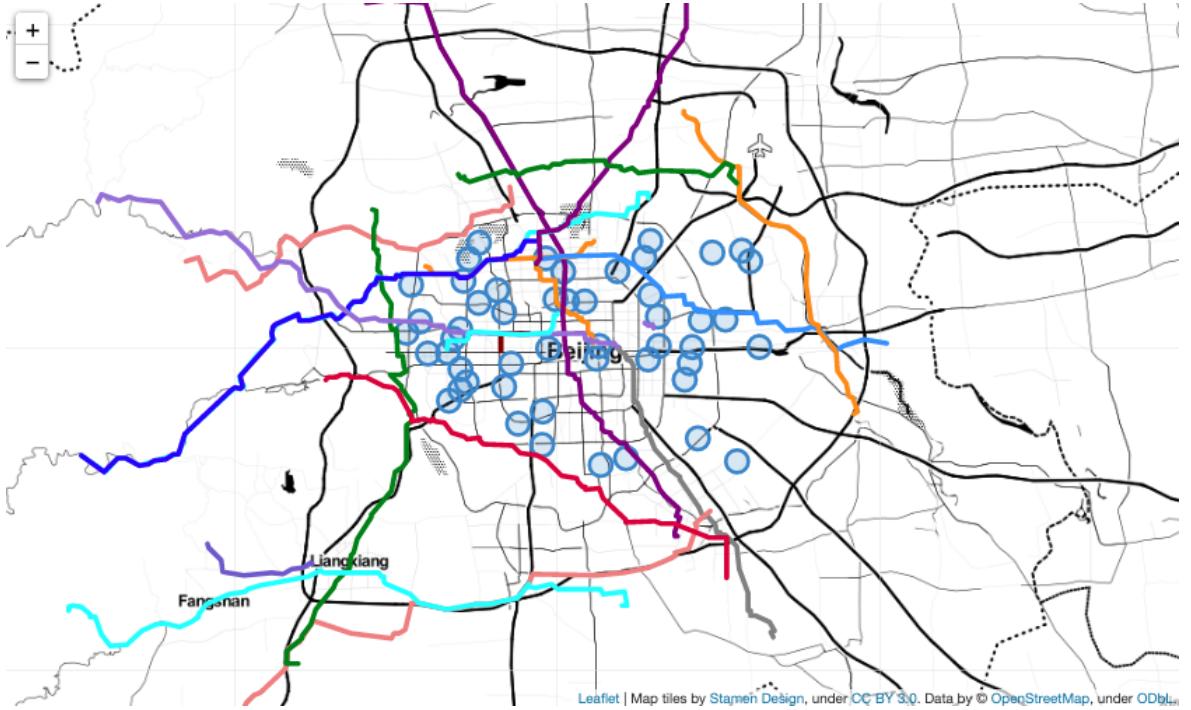


Figura 3.2: Delle traiettorie stampate sulla mappa di Pechino. I cerchi blu rappresentano delle locazioni di raccolta scelte casualmente.

Per studiare il dataset abbiamo provveduto al parsing dei file *.plt* per mezzo della libreria scikit-mobility [4], che possiede un metodo apposito per caricare le traiettorie del dataset GeoLife. Dopo il caricamento abbiamo proseguito con una pulizia del dataset, rimuovendo i campi inutili precedentemente citati, altitudine compresa, e tenendone uno soltanto, in formato Datetime, come timestamp. Successivamente abbiamo assegnato manualmente degli identificativi univoci (UIDs) agli utenti, poiché ne erano sprovvisti. Nel dataset, infatti, cartelle diverse corrispondono ad utenti diversi, e all'interno di ciascuna di esse sono presenti altre cartelle ad indicare gruppi di traiettorie del singolo utente. Una volta assegnati gli UIDs abbiamo proceduto a concatenare in un dataframe le traiettorie e a serializzarle su disco in formato *.csv*, basando le nostre successive analisi sul file così ottenuto.

Come specificato dagli autori nella guida allegata al dataset, solo alcuni degli utenti hanno partecipato per più di qualche settimana al progetto [38] e solo in 73 hanno

etichettato le proprie traiettorie coi mezzi di trasporto utilizzati. Questi fatti sono particolarmente evidenti se si analizza la quantità di punti presenti nel dataset. Il dato che emerge è una grossa fluttuazione nella quantità di traiettorie disponibili, a seconda del periodo di riferimento preso in considerazione. Questo fatto è mostrato in Figura 3.3, dove si può osservare una rappresentazione lineare dei punti presenti in GeoLife.

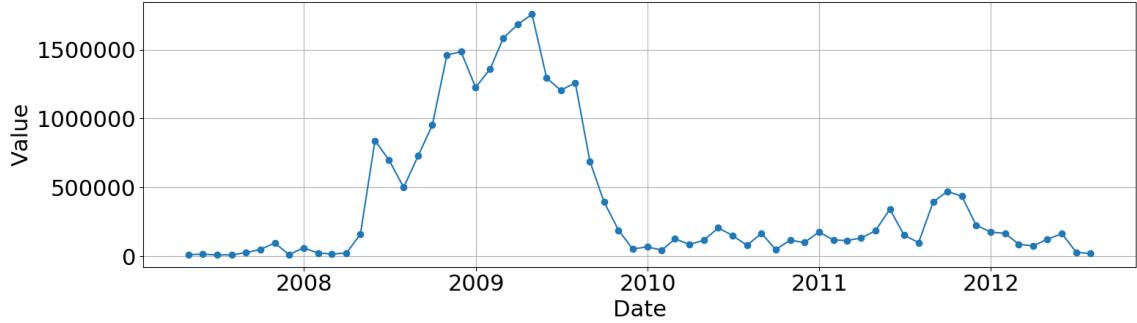


Figura 3.3: Serie temporale del numero di punti aggregati per mese.

Conseguentemente a ciò, abbiamo ritenuto opportuno restringere le nostra analisi ad un periodo temporale con la maggiore quantità di dati disponibile, ovvero il periodo di 12 mesi che va dal giugno 2008 al giugno 2009. Tale periodo è mostrato in Figura 3.4. Le mappe di calore in Figura 3.5 per il biennio 2008/2009, ottenute con seaborn, danno una ulteriore conferma di questa scarsità di dati.

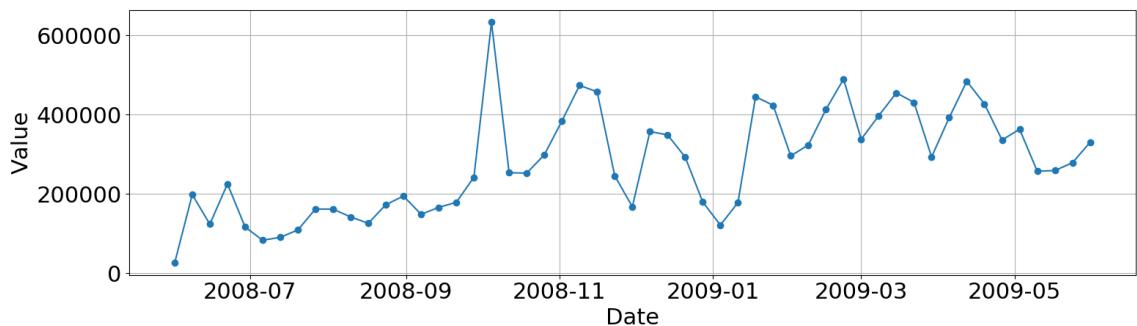
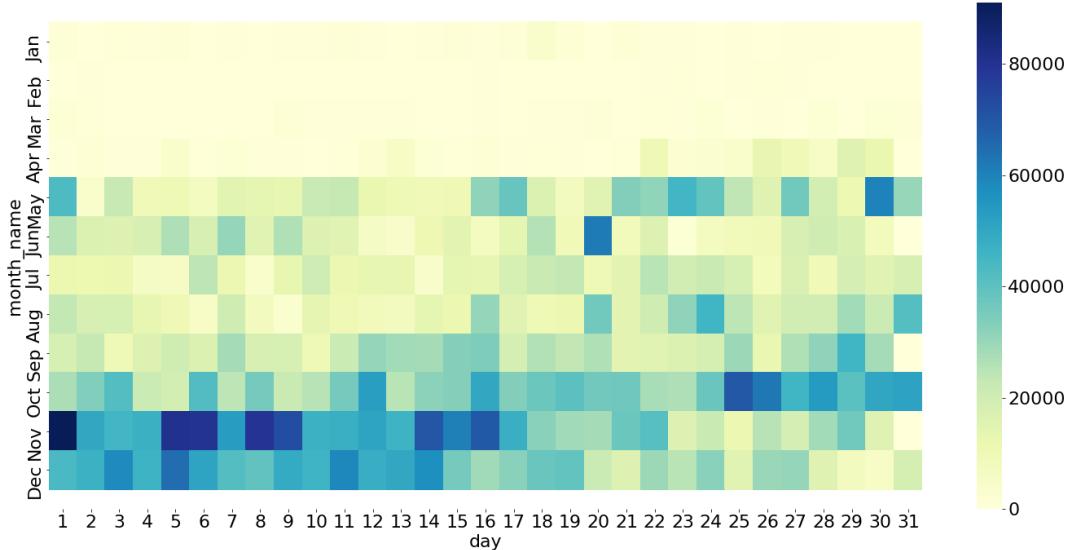
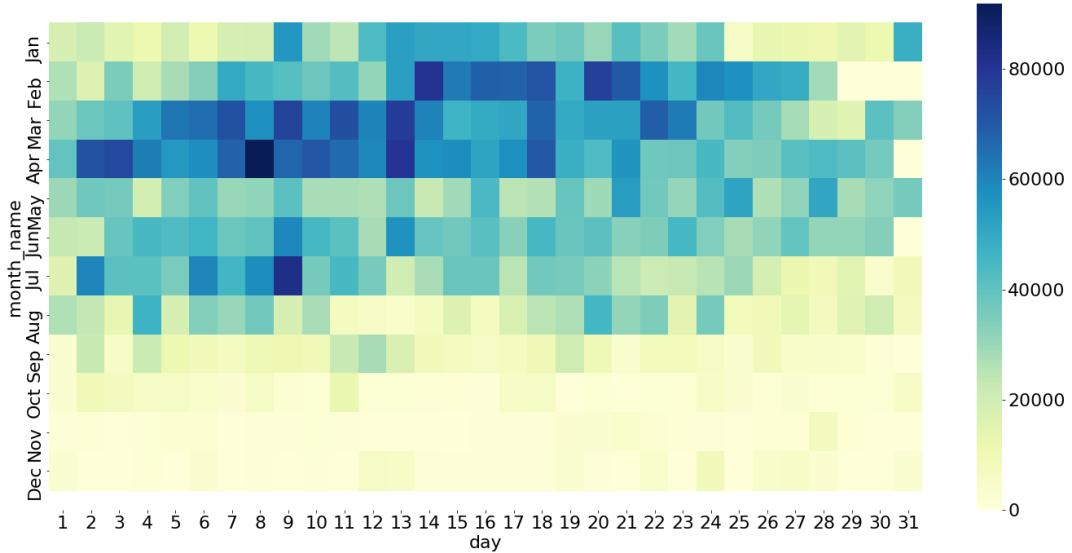


Figura 3.4: Serie temporale del numero di punti aggregati per settimana nel periodo giugno 2008/giugno 2009



(a) Quantità di punti GPS aggregati per mese nel 2008. L'asse x riporta i giorni del mese, l'asse y i mesi considerati



(b) Quantità di punti GPS aggregati per mese nel 2009. L'asse x riporta i giorni del mese, l'asse y i mesi considerati

Figura 3.5: Numero di punti registrati per giorno su scala annuale, graficati grazie a seaborn.

Nonostante il periodo preso in esame sembrasse inizialmente promettente, gli utenti attivi sono comunque pochi: solo 44 per l'estate 2008. Dal momento che questa scarsità di utenti condiziona il calcolo della Data Coverage, abbiamo sviluppato una metodologia per estendere GeoLife.

3.2 Augmented Geolife

Al fine di generare Augmented GeoLife, ovvero la nostra versione del dataset GeoLife estesa per mezzo di tracce generate sinteticamente, sono stati necessari una serie di passi preliminari e di simulazione, che riportiamo di seguito:

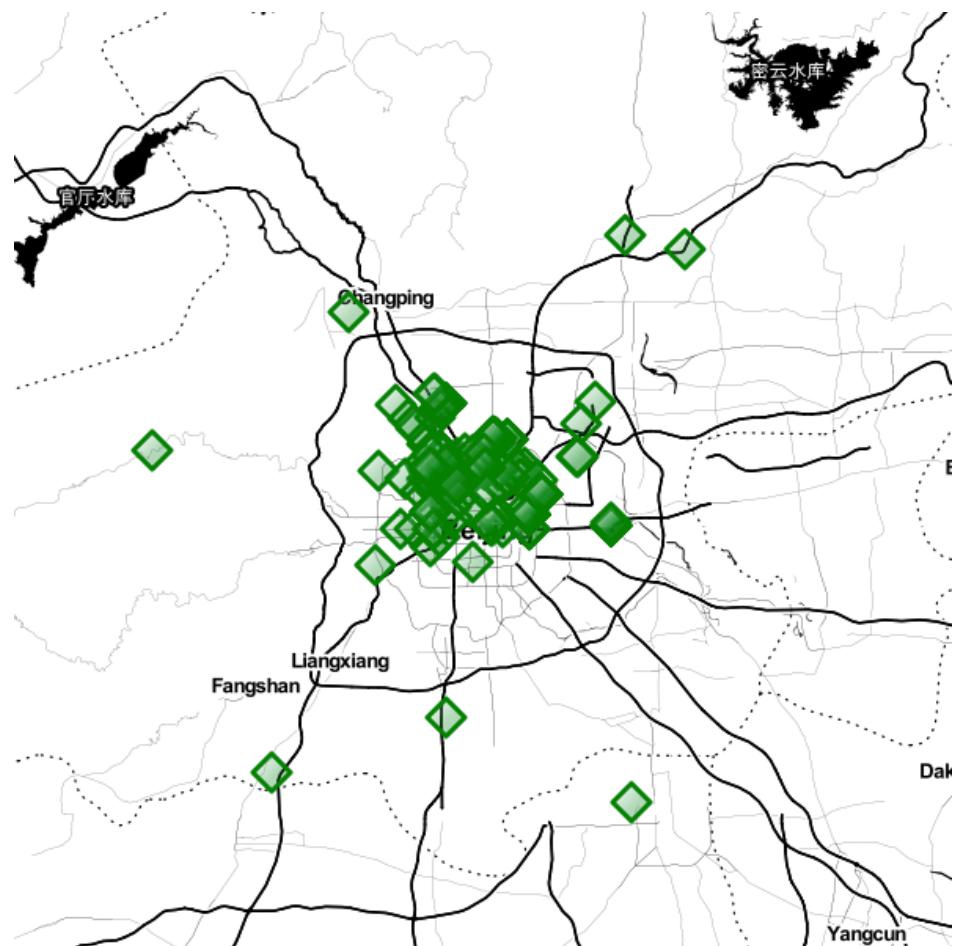
1. Rilevamento e clustering dei punti di fermata
2. Calcolo dei mediodi di ogni cluster
3. Simulazione delle traiettorie
4. Interpolazione e timestamping dei punti di ogni traiettoria

Rilevamento e clustering dei punti di fermata Abbiamo cominciato con il rilevamento dei punti di fermata per ciascuna delle traiettorie di GeoLife, e il loro successivo clustering. Questo passo, e il seguente di clustering, sono stati eseguiti grazie alla libreria scikit-mobility; [4], il processo implementato da essa per il rilevamento dei punti di fermata è basato su Lachesis Project [39]. Grazie a scikit-mobility è stato possibile eseguire questa fase preliminare con due semplici chiamate a metodi presenti nella libreria, rispettivamente a

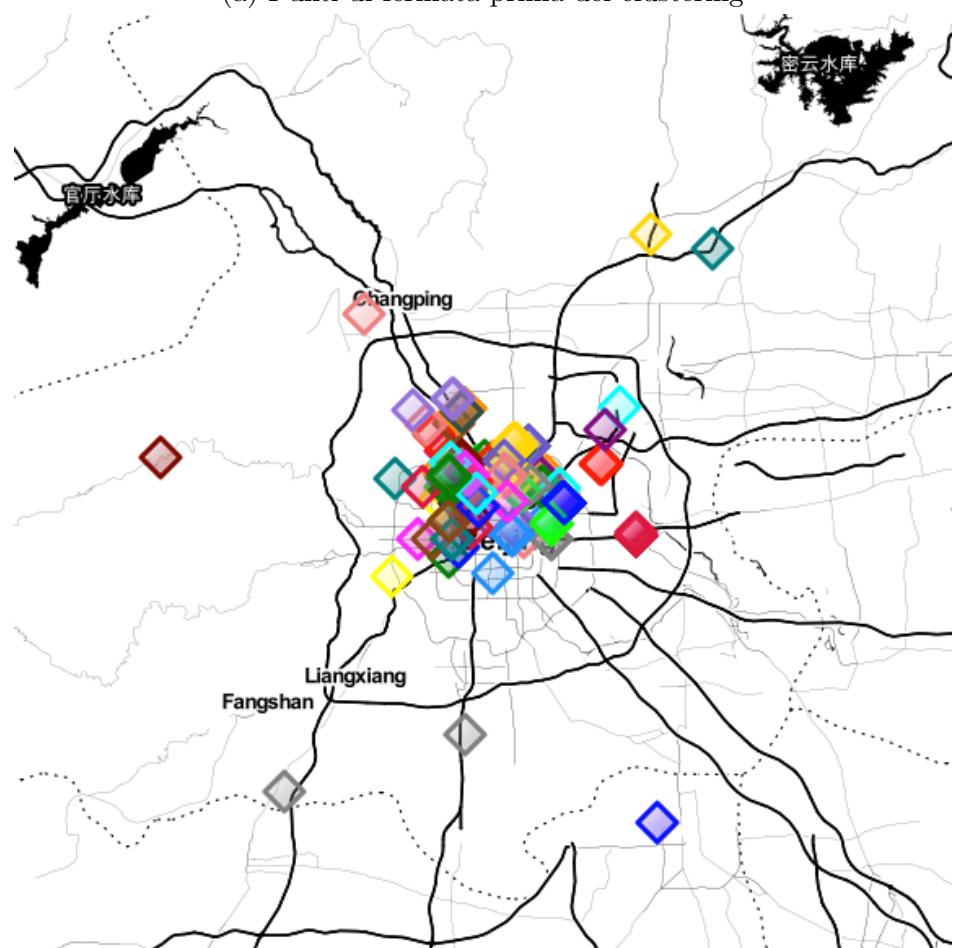
- *skmob.preprocessing.detection.stops()*
- *skmob.preprocessing.clustering.cluster()*

In Figura 3.6 si possono vedere i punti di fermata prima e dopo il clustering.

La tabella sottostante mostra invece la struttura dei punti di fermata calcolati da scikit mobility.



(a) Punti di fermata prima del clustering



(b) Punti di fermata dopo il clustering

Figura 3.6: I punti di fermata di GeoLife prima e dopo il clustering.

index	datetime	lat	lng	uid	leaving`datetime
85	2008-06-22 01:53:33	39.9754283	116.33049495	0	2008-06-22 04:01:07
59	2008-06-15 11:14:26	39.9848275	116.303761	0	2008-06-15 12:50:11
6	2008-06-03 05:22:48	39.984988	116.352642	0	2008-06-03 08:08:27
300	2008-08-11 16:21:21	40.006256	116.320507	0	2008-08-11 23:34:28
311	2008-08-13 19:02:24	40.03055245	116.40909325	0	2008-08-13 20:33:27

Calcolo dei medoidi Con medoide intendiamo un punto rappresentativo di un dataset o di un cluster che minimizzi la differenza media da tutti gli altri oggetti appartenenti al cluster [40].

Abbiamo calcolato i medoidi per ciascun gruppo di punti di fermata, tramite la formula

$$x_{medoid} = \arg \min_{y \in X} \sum_{i=1}^n d(y, x_i) \quad (3.1)$$

dove $X = x_1, x_2, \dots, x_n$ è un insieme di n punti in uno spazio con una funzione di distanza d .

Come funzione di distanza abbiamo utilizzato la distanza euclidea, calcolata per mezzo del metodo `sklearn.metrics.pairwise_distances(metric="euclidean")` appartenente alla libreria scikit-learn [30]. Grazie alla funzione abbiamo ottenuto una matrice di distanze a coppie (pairwise distance matrix) grazie alla quale abbiamo applicato l'equazione 3.1 per il calcolo dei medoidi. Sebbene questo metodo abbia una complessità di $O(n^2)$, per i nostri scopi risulta essere sufficientemente rapido. Ciononostante, qualora si dovesse ripetere l'esperimento con un dataset molto più ampio, si potrebbero impiegare metodi subquadratici molto più efficienti per il calcolo dei medoidi [41, 42] che si avvicinano alla linearità.

La Figura 3.7 mostra i medoidi calcolati per i punti di fermata.

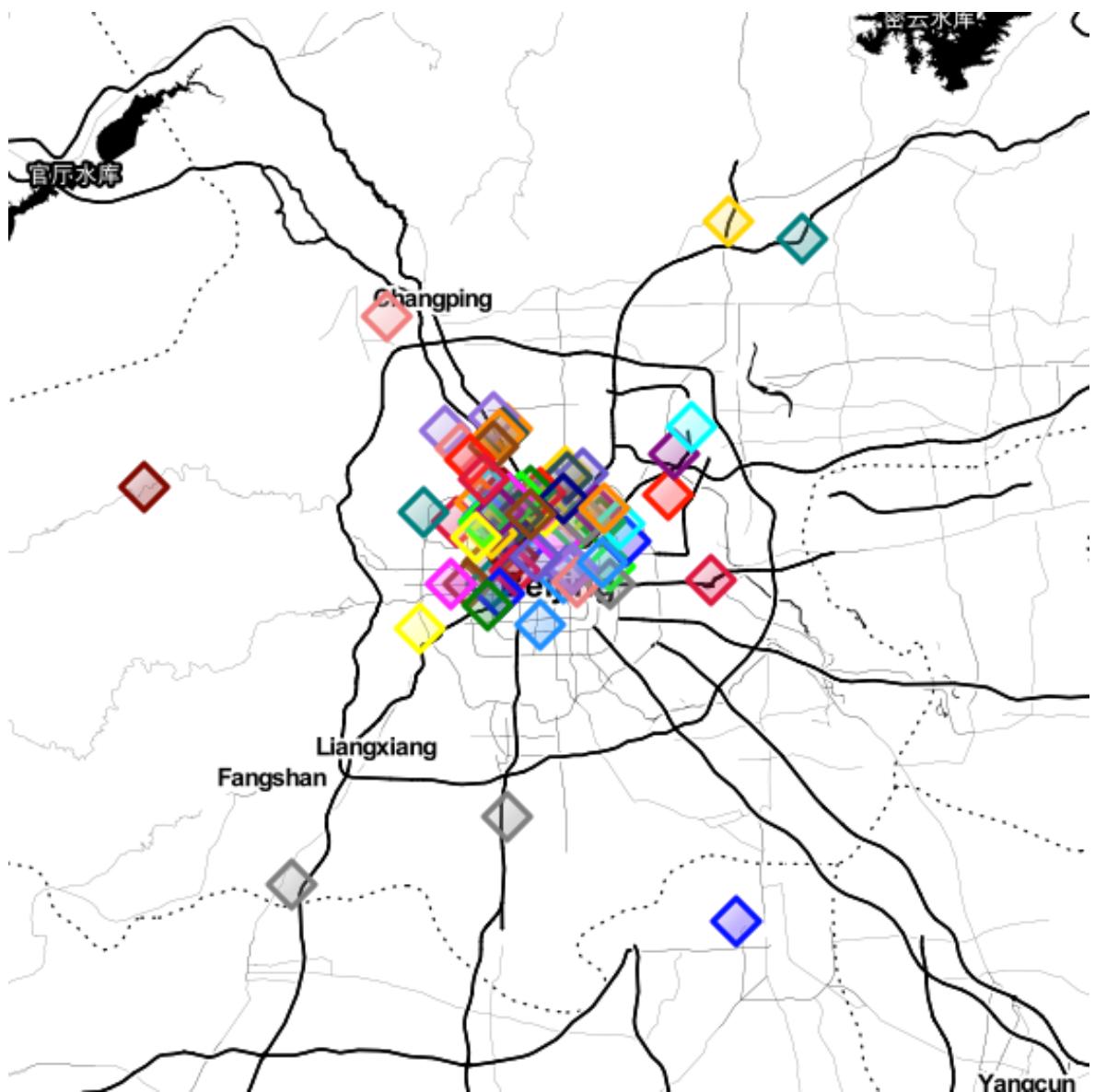


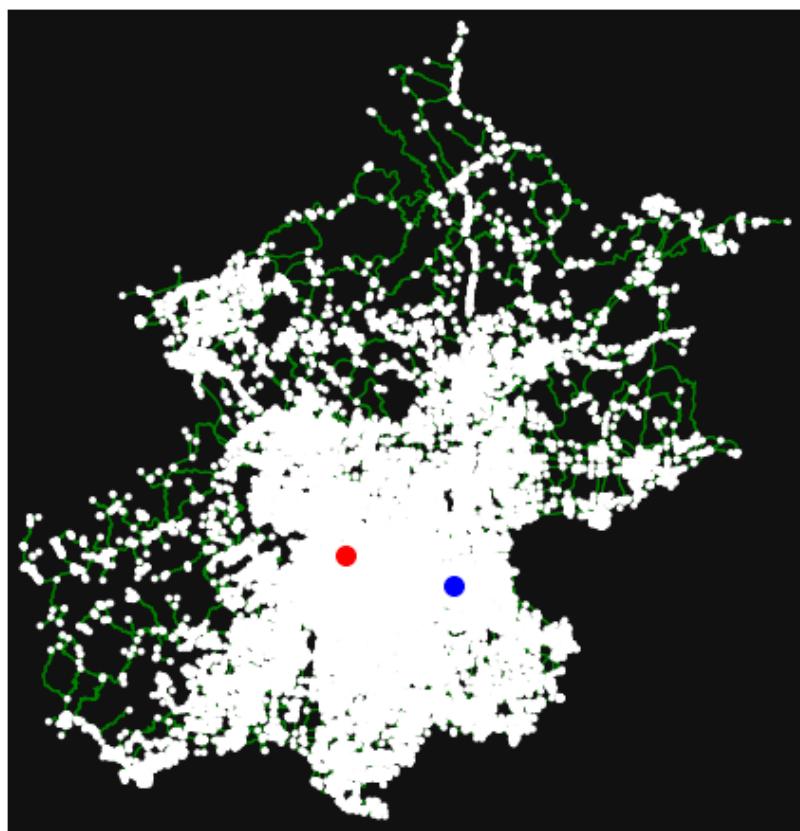
Figura 3.7: I medoidi calcolati a partire dai punti di fermata

Dalla mappa risulta un numero di medoidi relativamente basso, a nostro avviso insufficiente per l'utilizzo come punti di partenza ed arrivo per le traiettorie che in seguito saremmo andati a generare; per questo motivo abbiamo deciso di aggiungere dei punti random all'interno del perimetro di coordinate GPS della regione di Pechino. A tale scopo abbiamo utilizzato la funzione `random.uniform()` della libreria standard di python. Così facendo abbiamo arricchito l'insieme, arrivando ad un totale di 181 punti da utilizzare come partenze e arrivi per le traiettorie sintetiche. Fatto ciò abbiamo provveduto alla serializzazione dell'insieme di punti su disco, al fine di utilizzarlo nel simulatore di traiettorie.

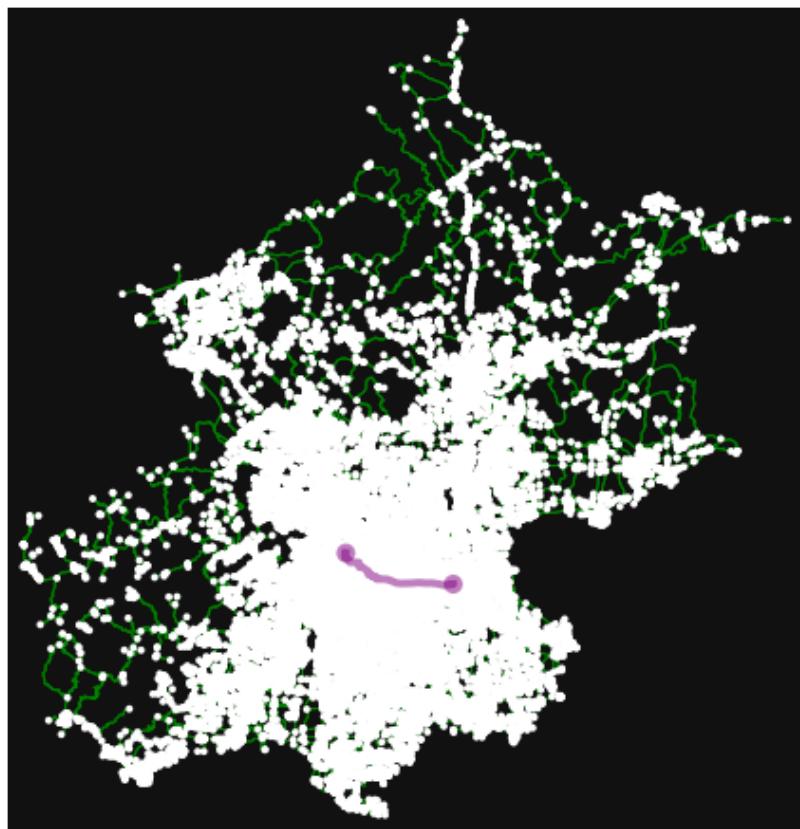
3.2.1 Simulazione delle traiettorie

Al fine di simulare le traiettorie abbiamo utilizzato una serie di librerie, prima fra tutti OSMnx. Grazie a tale libreria abbiamo scaricato la rete stradale di Pechino dal database di OpenStreetMap, nelle tre versioni pedonabile, ciclabile e percorribile in auto. Successivamente abbiamo ricercato nel grafo della rete stradale i nodi più vicini ai nostri punti di arrivo e partenza precedentemente calcolati a partire dai medoidi e dai punti di arricchimento random; per farlo ci siamo serviti della struttura dati denominata *KD Tree*, specializzazione multidimensionale di un albero binario di ricerca inventata negli anni 70 da Jon Bentley [43], grazie al quale abbiamo ottenuto una complessità asintotica di $O(n \log n)$ nella ricerca di un punto nel grafo. Una volta trovati i punti di partenza e arrivo più vicini, abbiamo calcolato il cammino minimo tra di essi nel grafo della città, usando a tale scopo la funzione `shortest_path()` della libreria NetworkX [33] che implementa al suo interno l'algoritmo di Dijkstra [44]. Nella Figura 3.8 si mostra un esempio del grafo ottenuto per la rete stradale con sovraimpressi due punti di partenza ed arrivo assieme ad una traiettoria generata.

Analizzando il cammino ottenuto sul grafo, ci siamo resi conto che non era sufficientemente denso di punti da essere utilizzato con successo nel nostro modello; OSMnx, infatti, onde evitare sprechi di memoria, rappresenta il grafo della città usando come nodi svincoli e incroci stradali, e come archi le strade stesse, in formato di linee tra due



(a) Punti di partenza ed arrivo



(b) Traiettoria generata

Figura 3.8: I grafi stradali ottenuti per mezzo di OSMnx.

punti estremi, evitando in questo modo di inserire i punti intermedi [26]. Per questa ragione abbiamo usato una strategia di interpolazione lineare per uniformare la distanza tra i punti del grafo. L’interpolazione è servita ad evitare situazioni di carenza di dati significativi in zone con strade dritte e lunghe, come ad esempio arterie extraurbane e viali di scorrimento. Per interpolare le traiettorie abbiamo fatto uso della libreria *GeographicLib* [34]. Tale libreria implementa al suo interno una funzione per il calcolo della distanza geodetica tra due punti, grazie alla quale abbiamo interpolato i nodi del grafo ottenuto con OSMnx senza introdurre errori che sarebbero stati causati da misure di distanza più semplici. Siamo così riusciti ad ottenere delle traiettorie di punti ad una distanza arbitraria, da utilizzare nella nostra pipeline di arricchimento di GeoLife. In Figura 3.9 si mostra un esempio di traiettoria interpolata tramite GeographicLib.

La funzione generatrice Avendo preparato la nostra pipeline per l’arricchimento, abbiamo provveduto a creare una funzione generatrice, che prendesse in input i profili di diversi tipi di utenti (pedoni, ciclisti, automobilisti) e restituisse in output traiettorie credibili generate utilizzando i punti di partenza e arrivo precedentemente calcolati, che vengono poi selezionati casualmente come estremi per ogni nuova traiettoria da generare. Il corpo della funzione si occupa di deserializzare i grafi di Pechino dal disco e richiama al suo interno due componenti principali:

- **interpolator**

Il cui ruolo è quello di interpolare le traiettorie generate, secondo i criteri descritti sopra.

- **put_datetime**

Che si occupa di assegnare un timestamp progressivo a ciascun punto delle traiettorie, evitando casi di ubiquità per il singolo utente, ovvero controllando che non vi siano collisioni temporali tra traiettorie diverse dello stesso utente. Per fare ciò utilizza al suo interno la classe *Interval* di pandas, grazie alla quale si possono rappresentare intervalli numerici e verificare se un numero ricade tra uno di essi, per mezzo del metodo *overlaps*.

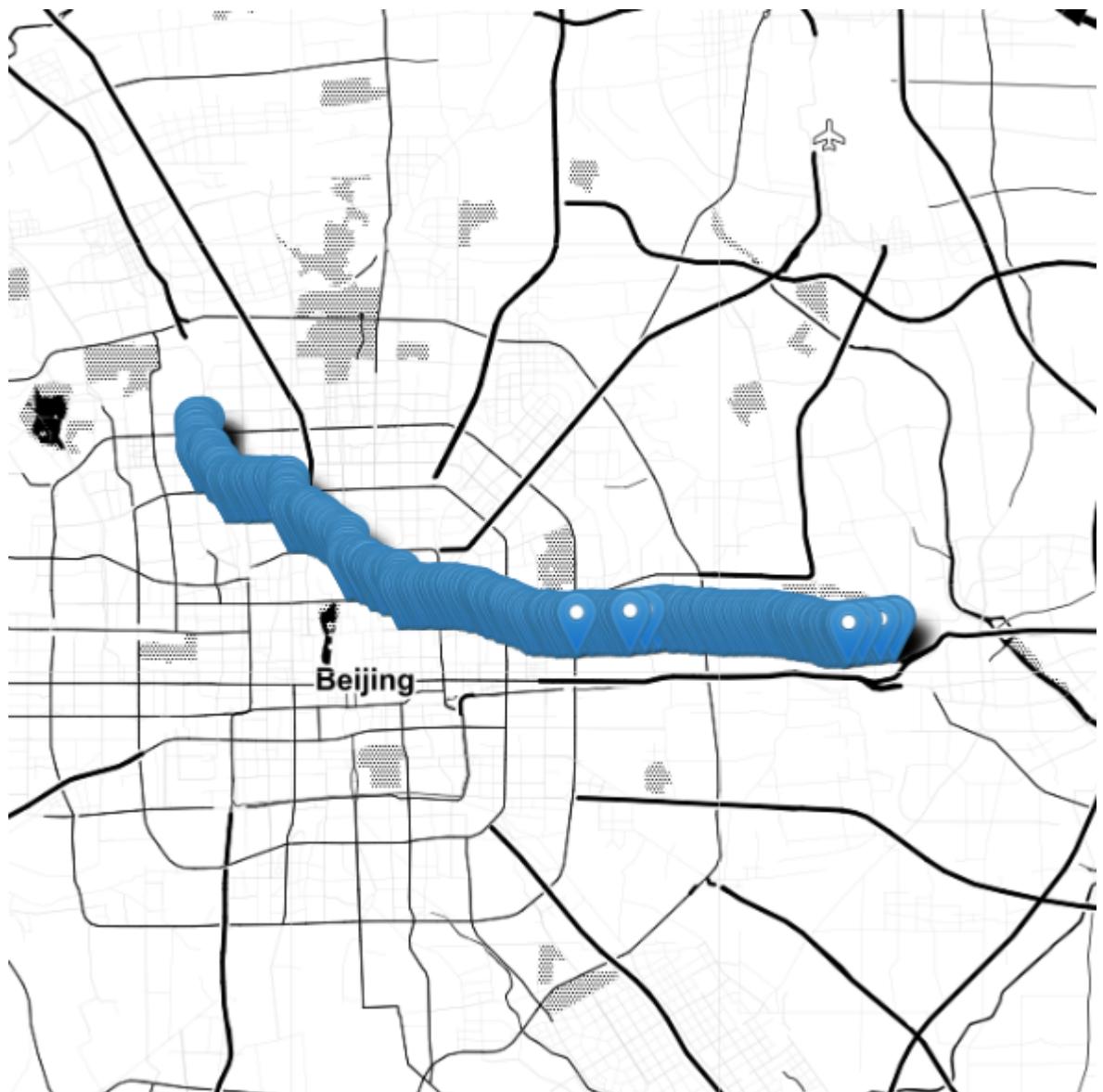


Figura 3.9: Una traiettoria interpolata e visualizzata tramite folium. Ciascuno dei pin rappresenta uno dei punti della traiettoria.

Il formato delle traiettorie rese in output dalla funzione generatrice è un DataFrame del tutto indistinguibile da quello delle reali traiettorie registrate in Geolife, eccetto il campo *uid*, identificativo dei singoli utenti, che è settato ad un valore molto più alto dell'uid massimo di GeoLife. Riportiamo di seguito un esempio di punti tratti dal dataframe ottenuto.

lat	lon	uid	tid	date`time
39.978994	116.347534	1000	1	2008-06-24 03:48:48
39.978949	116.347536	1000	1	2008-06-24 03:48:48.450114
39.978904	116.347538	1000	1	2008-06-24 03:48:48.900228
39.978859	116.347540	1000	1	2008-06-24 03:48:49.350342
39.978814	116.347543	1000	1	2008-06-24 03:48:49.800456

Infine, abbiamo concatenato le traiettorie sintetiche con quelle presenti nel dataset originale, ottenendo in questo modo il dataset che abbiamo denominato *Augmented GeoLife*.

3.3 Analisi del dataset aumentato

Passiamo adesso a valutare la quantità di dati presenti in Augmented GeoLife rispetto al dataset originale. Dal grafico a barre in Figura 3.10 è possibile osservare l'aumento di utenti attivi e di traiettorie nel periodo preso in esame: siamo passati da 38 utenti unici per un totale di 1580 traiettorie del dataset originario, a rispettivamente 237 utenti e 2957 traiettorie in Augmented Geolife. In media, per ogni utente, sono presenti 53849.36 punti georeferenziati, per un totale di 1.27623×10^7 punti.

Il paragone tra punti aggregati per settimana presente in Figura 3.11 è altrettanto interessante per mostrare la bontà della nostra funzione generatrice; si nota infatti come le traiettorie del dataset aumentato siano spalmate su tutto il periodo preso in esame senza particolari picchi positivi o negativi.

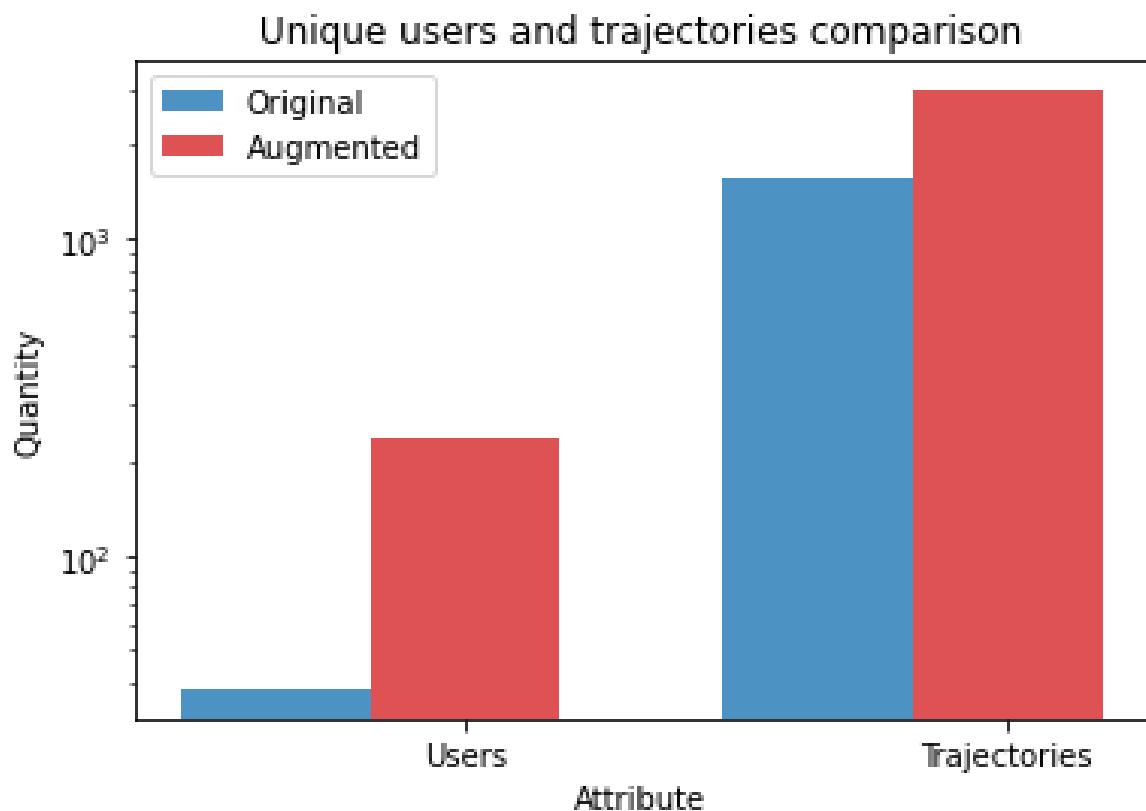


Figura 3.10: Un paragone tra utenti e traiettorie del dataset originario e di quello aumentato, graficato per mezzo di un bar chart.

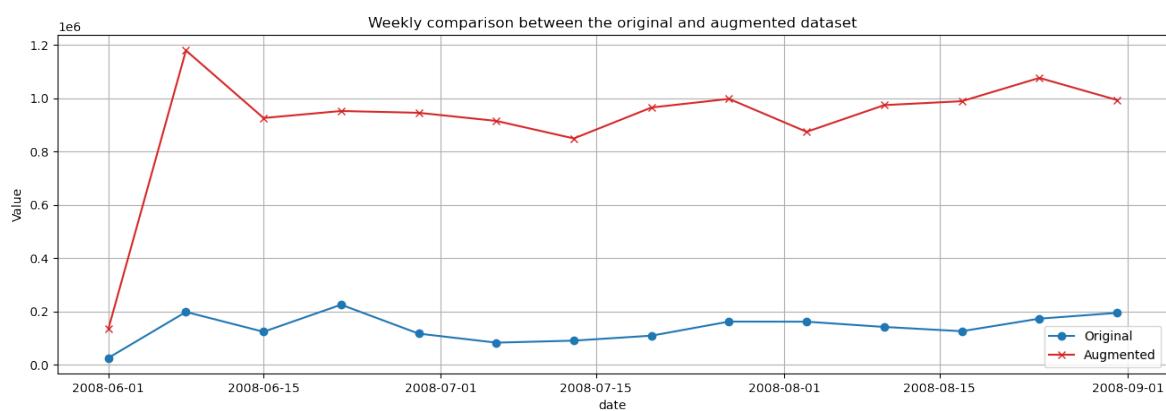


Figura 3.11: Il numero di traiettorie è cresciuto in modo abbastanza regolare, senza particolari picchi positivi o negativi, come evidenziato da questo grafico lineare.

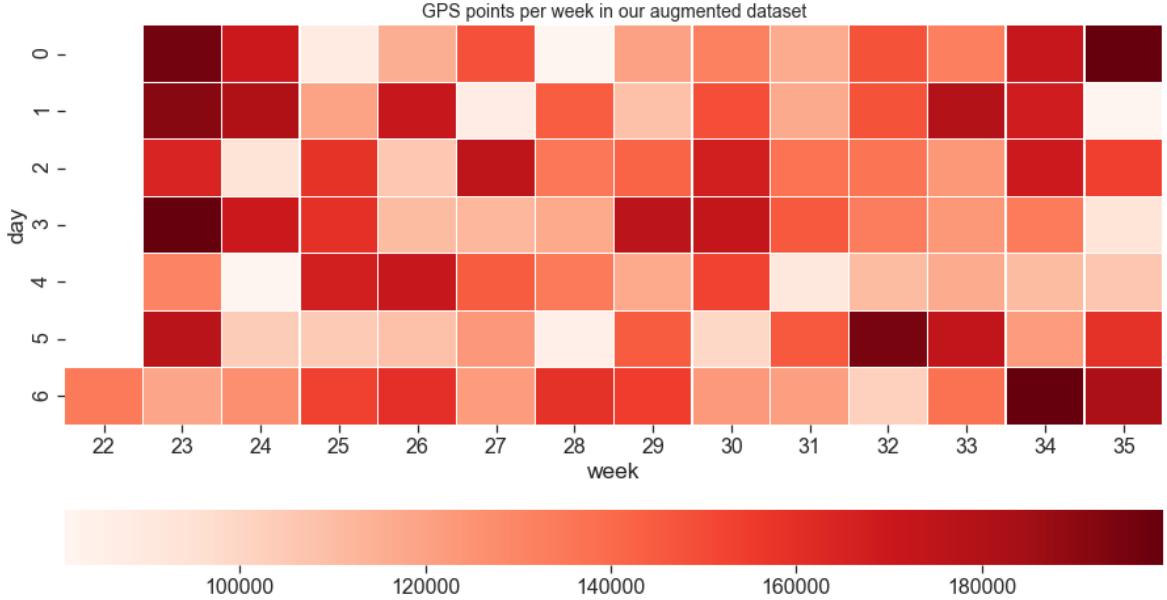


Figura 3.12: In questa mappa di calore, ottenuta con seaborn, è possibile vedere la quantità di punti gps per ciascuno dei giorni della settimana del dataset aumentato.

La mappa di calore presente in Figura 3.12 è un ulteriore modo per visualizzare la densità di punti nel nostro dataset aumentato, che risulta essere sufficientemente realistica, se paragonata alle mappa di calore di Figura 3.5.

La Figura 3.13, ottenuta con datashader, mostra la densità dei punti del dataset aumentato, in modo analogo a quanto fatto con la Figure 3.1 col dataset originario.

Altre metriche Per studiare il dataset aumentato abbiamo calcolato su di esso altre metriche utili a valutarne le caratteristiche. Le seguenti misure possono essere collettive, ovvero riferite ai pattern di mobilità dell’intera popolazione degli utenti del dataset, siano essi reali o simulati, oppure individuali, riferite ai pattern di mobilità ogni singolo utente. Per ciascuna di esse chiariremo questo fatto e daremo una descrizione dell’utilità della metrica. Tutte le metriche sono implementate all’interno di scikit-mobility e ne costituiscono una delle funzionalità principali [4].

Numero di visite e utenti attivi settimanalmente Il numero di visite (number of visits) si riferisce alla quantità di punti georeferenziati registrati per ciascun utente all’interno del DataFrame di traiettorie. Dalla Figura 3.14 è chiaro come le fluttuazioni negative nella quantità di utenti attivi, visibili nella mappa di calore in-

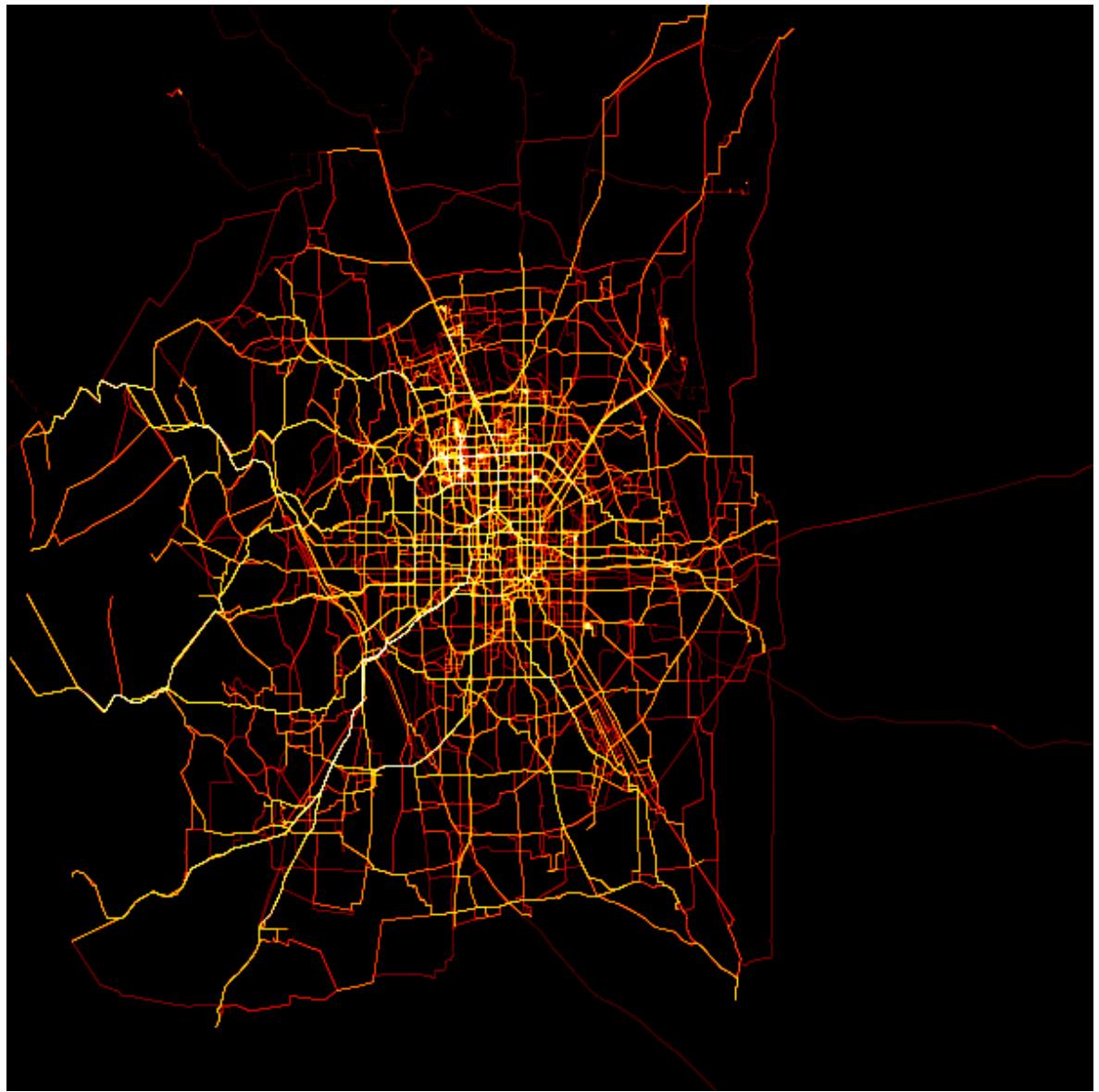


Figura 3.13: Questo grafico ottenuto con datashader, mostra la densità dei punti presenti nel dataset aumentato.

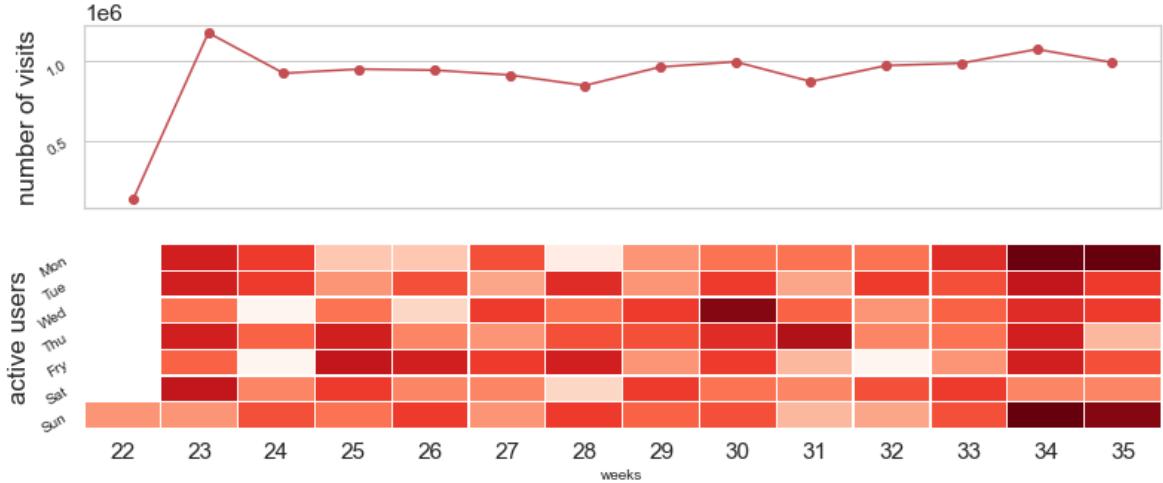


Figura 3.14: In questi grafici si mettono in relazione il numero di visite aggregate per settimana e gli utenti attivi per la durata del nostro dataset aumentato.

feriore, non influiscono sulla quantità di punti registrati per ciascuna settimana, che resta nell’ordine di 1×10^6 .

Radius of gyration e numero di visite il Radius of Gyration [45, 46] è una misura di mobilità individuale definita da

$$r_g(u) = \sqrt{\frac{1}{n_u} \sum_{i=1}^{n_u} dist(r_i(u) - r_{cm}(u))^2} \quad (3.2)$$

in cui $r_i(u)$ rappresenta le n_u posizioni registrate per l’utente u , e $r_{cm}(u)$ è il centro di massa delle traiettorie dell’utente u . Tale metrica indica la distanza caratteristica percorsa dall’utente u , espressa in chilometri. Dal grafico in Figura 3.15 si osserva una elevata distribuzione nell’intervallo [10, 25]. La PDF mostra una densità stabile nell’intervallo 0 - 12 Km, dopodichè osserviamo una tipica campana gaussiana nell’intervallo 15 - 30 Km. Questo è dovuto al fatto che alcuni degli utenti sintetici si sono mossi relativamente poco, avendo effettuato traiettorie brevi, qualora i punti di partenza e di arrivo scelti in modo casuale risultassero molto vicini.

Frequency e recency ranks Il frequency rank $K_f(r_i)$ di una locazione r_i per un utente u è $K_f(r_i) = 1$ se la locazione r_i è la più frequentemente visitata dall’utente, $K_f(r_i) = 2$ se è la seconda più visitata e così via.

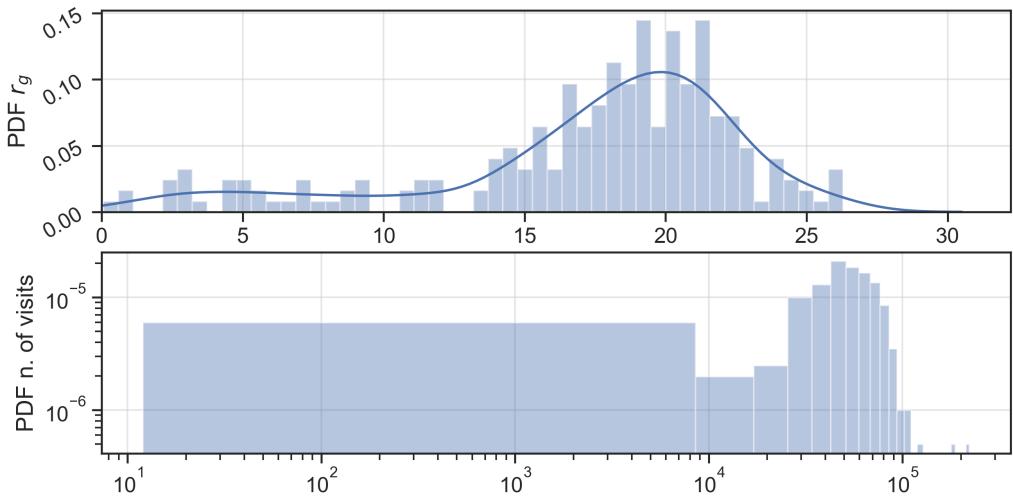


Figura 3.15: In questa figura si mostrano le densità di probabilità del Radius of gyration, in alto, e del numero di visite, in basso, messe in relazione tra loro.

Allo stesso modo, il recency rank $K_s(r_i)$ per una locazione r_i di un utente u , vale $K_s(r_i) = 1$ se tale locazione è quella visitata più recentemente dall’utente, $K_s(r_i) = 2$ se è la seconda visitata più recentemente e così via.

Frequency e recency rank sono entrambe misure di mobilità individuali.

La Figura 3.16 mostra un joint plot di queste due metriche. In basso a sinistra è possibile osservare come alcune delle locazioni siano state visitate molto recentemente e con frequenza, mentre il resto del grafico risulta essere non troppo significativo, dal momento che la natura casuale dell’algoritmo di generazione delle traiettorie fa sì che non vi sia una correlazione tra le locazioni visitate con più frequenza e quelle visitate più recentemente nel caso degli utenti sintetici.

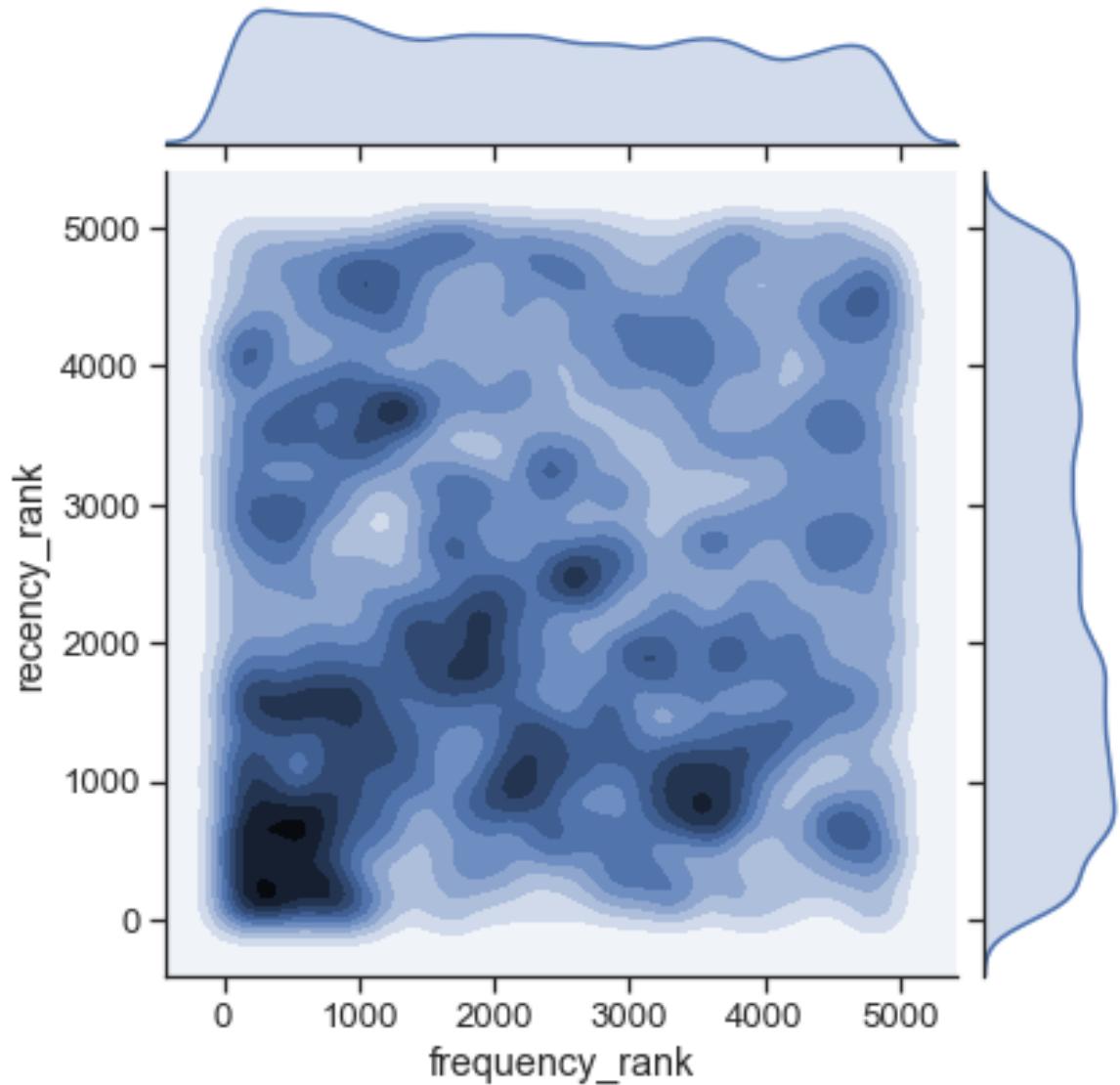


Figura 3.16: Relazione tra frequency e recency rank.

Capitolo 4

Sperimentazioni e analisi dei risultati

In questo capitolo descriviamo la sperimentazione effettuata grazie al modello di Data Coverage e alle traiettorie di Augmented Geolife. Daremo una descrizione per ciascuno degli scenari applicativi del modello, e successivamente presenteremo i risultati sperimentali, fornendone anche una opportuna visualizzazione grafica. Dapprima parleremo del monitoraggio ambientale, spostandoci poi sul monitoraggio dei flussi di traffico e trattando infine il sensing basato su punti di interesse.

Nell'ultima sezione del capitolo, cercheremo di interpretare i risultati ottenuti per ciascuno scenario.

4.1 Tre scenari applicativi

In questa tesi abbiamo preso in esame tre diversi scenari applicativi. Ciascuno di essi rappresenta un diverso ambito di applicazione della Data Coverage e, più in generale, del paradigma del Mobile Crowdsensing. Con scenario applicativo intendiamo un diverso posizionamento delle locazioni di raccolta dati, guidato dall'obiettivo che vogliamo raggiungere. Tali differenze nei posizionamenti sono state considerate principalmente per mostrare l'efficienza del modello di copertura in diversi contesti di raccolta dati.

Nel seguito del capitolo evidenzieremo le peculiarità di ciascuno scenario, fornendo contestualmente i risultati dell'applicazione del modello di Data Coverage.

Il procedimento sperimentale seguito per ciascuno scenario è il seguente:

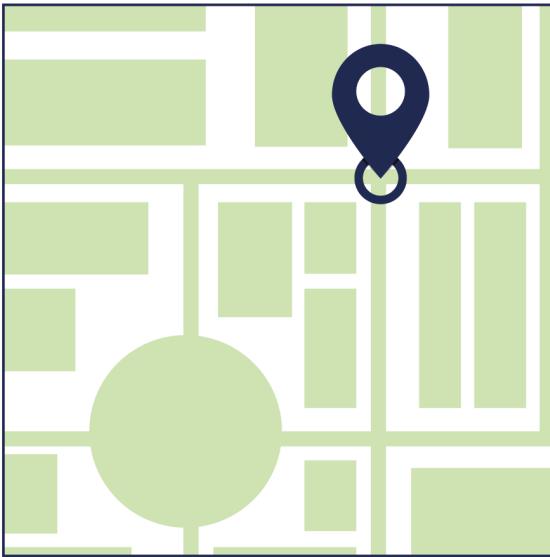
1. Dato uno scenario, ne abbiamo identificato le locazioni tramite generazione o estrazione dal database di OpenStreetMap.
2. Abbiamo considerato tre diversi valori di λ per modellare la probabilità che un utente sia disposto ad accettare un detour, come descritto nel Capitolo 2.
3. Applicando il modello alle locazioni dello scenario abbiamo ottenuto i valori di coverage per ciascuna di esse.
4. Utilizzando grafici di vario tipo abbiamo dato una visualizzazione della Data Coverage.

La Figura 4.1 presenta la nostra pipeline di sperimentazione, utilizzata per ogni scenario.

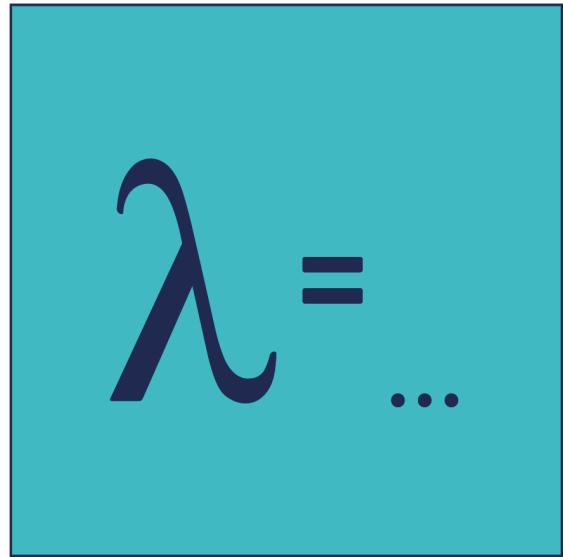
Gli scenari presi in esame sono i seguenti:

- Monitoraggio dei flussi di traffico
- Sensing basato su punti di interesse
- Monitoraggio ambientale basato su griglia

1 Identificazione delle locazioni



2 Setup dei parametri del modello



3 Applicazione del modello

```
1001000001110100110101011100001110  
100101101100110011001000000110111011  
01110010011001010010000001100110011011  
0101011101000111010001101001011011100110  
0110010100100000011000100110100101011011  
000001110110011000010110110001110101011  
1100110001000000011010010110100001011011  
00110010000001101001011010010100001011011  
0100000011100100110000010110110011000110  
0001100101010111000011101000100000001  
001110010000001101110110111000100000001  
0100100000011001100110111011101100100010  
1101000110100101101100110011101001000000  
000011000100110100010110111001100001011  
10000101101100011010101100101011100110  
000110101011010001011100101010001011011  
0101001010110100000011100110110111101110  
0011000010110111001100011011110110111011  
1000011101000010000000110011101101111011  
0110111101110001000000110100001100110001101  
0100110111011001000000110100001100110001101
```

4 Visualizzazione dei risultati

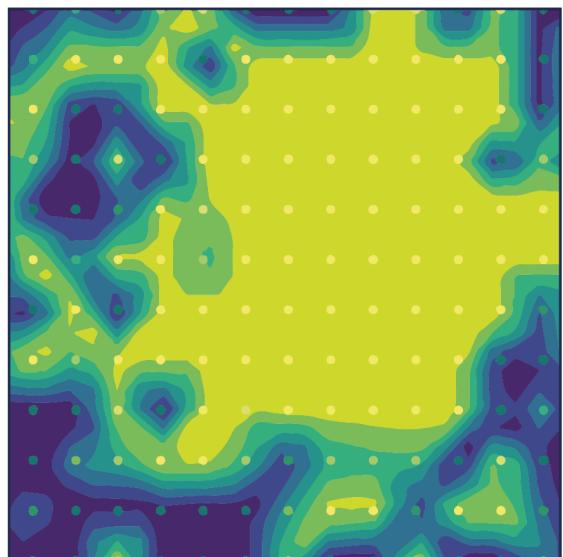


Figura 4.1: L'immagine mostra la pipeline di sperimentazione utilizzata

4.2 Monitoraggio ambientale

Lo scenario applicativo di monitoraggio ambientale si riferisce alla raccolta di dati utili al monitoraggio di parametri ambientali, come ad esempio: la temperatura, l'inquinamento acustico, la copertura e la qualità dei segnali wireless e molti altri.

Per questa tipologia di MCS abbiamo preso in esame locazioni disposte a griglia ed equispaziate l'una dall'altra. Per fare ciò abbiamo opportunamente generato una

griglia rettangolare più grande possibile che ricadesse all'interno dei confini geografici della macroarea di Pechino. L'idea è che i dati di tipo ambientale varino in modo molto meno netto rispetto a dati di altro tipo come possono essere quelli del traffico, per questa ragione la distanza tra le locazioni è nell'ordine dei chilometri, così da poter captare variazioni significative nei dati che devono essere raccolti.

4.2.1 Generazione della griglia

Per la generazione della griglia ci siamo avvalsi di *Shapely* [35] e *PiProj* [36]. Dopo aver scelto gli estremi dell'area su cui generare la griglia, abbiamo impiegato queste librerie per ottenere le locazioni corrette. Grazie alla classe Point di Shapely e alle proiezioni fornite da PyProj, abbiamo iterativamente selezionato i punti della griglia prendendoli a distanza regolare sulla proiezione bidimensionale ottenuta. Successivamente li abbiamo riproiettati a coordinate GCS sempre grazie a PyProj.

Una volta ottenuta la griglia abbiamo provveduto a serializzarla per la successiva applicazione del modello.

Grazie a folium [29], in Figura 4.2, possiamo visualizzare la griglia ottenuta. Le locazioni sono identificate dai cerchi blu. In sovraimpressione sono presenti alcune delle traiettorie di Augmented Geolife.

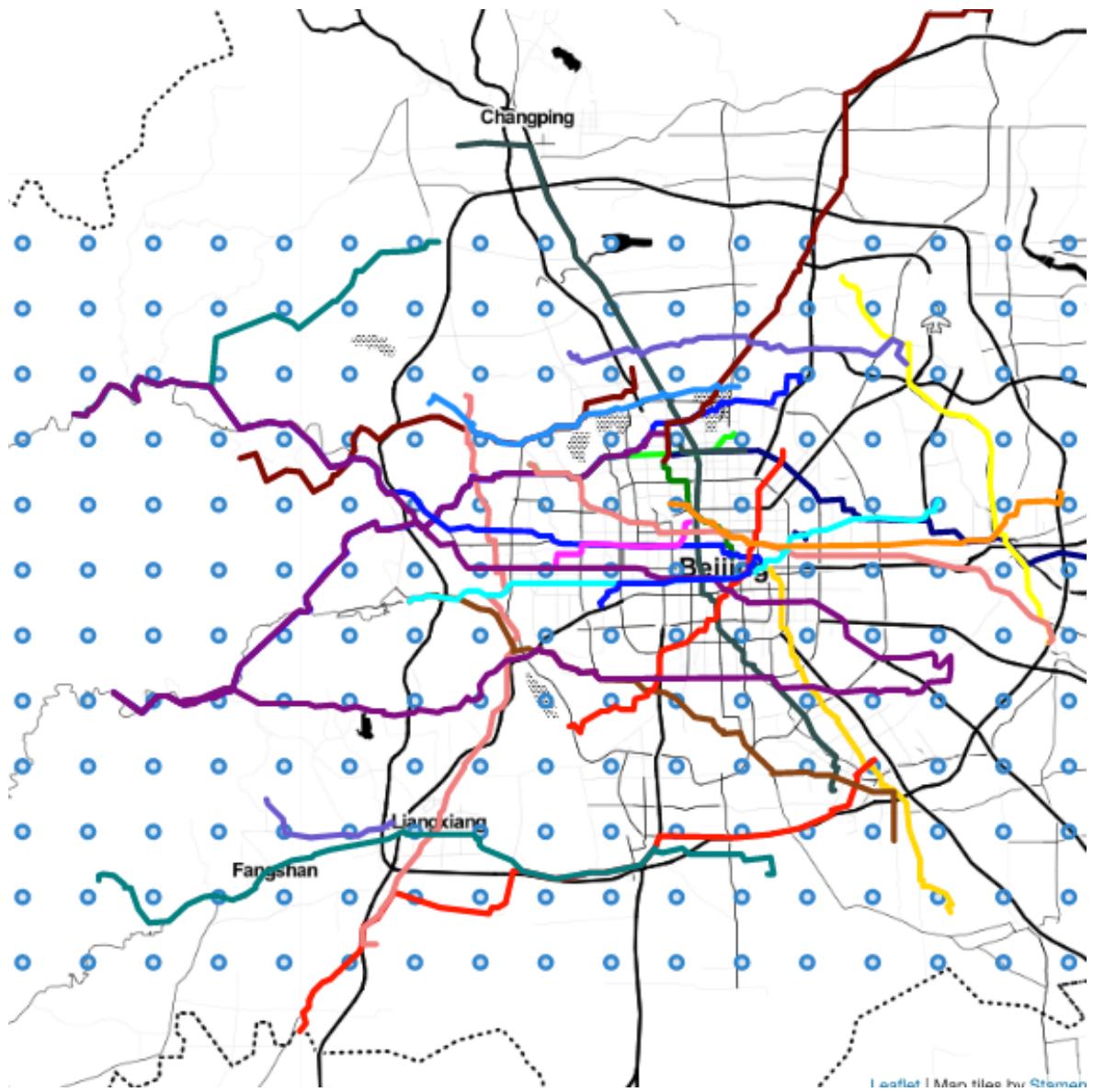
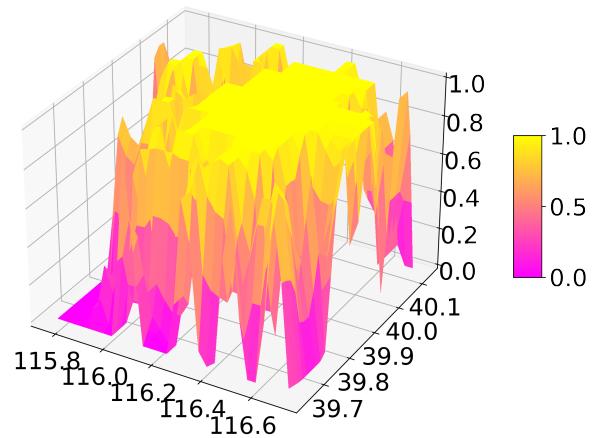


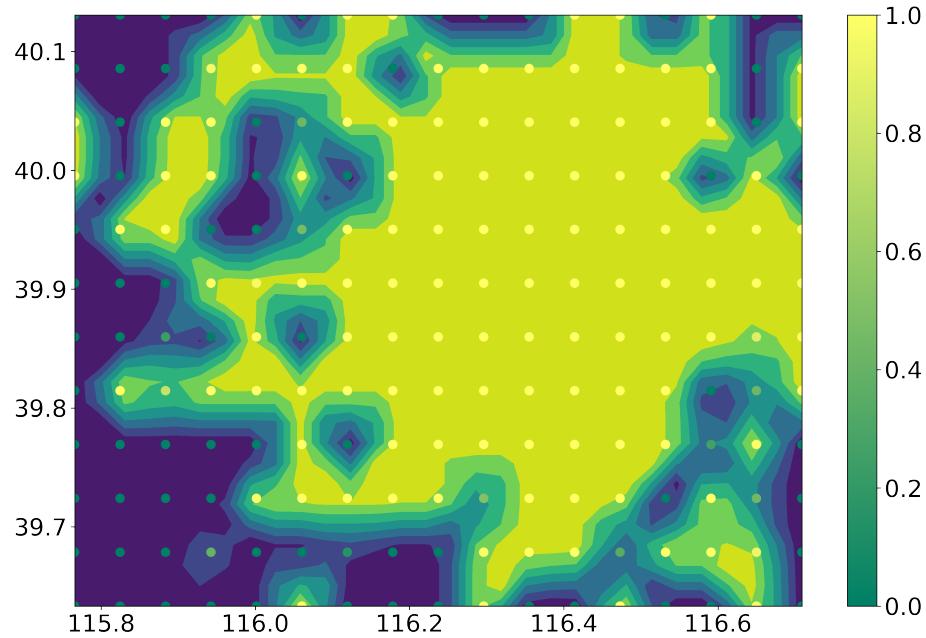
Figura 4.2: La griglia visualizzata per mezzo di folium

4.2.2 Risultati del modello di coverage

Le Figure 4.3, 4.4, e 4.5, mostrano i risultati ottenuti applicando il modello con differenti valori di lambda sulla griglia ottenuta. Dalle mappe di calore e meshgrid ottenute, si può notare come la Data Coverage vari al variare del paramentro λ , che modella la probabilità che gli utenti siano disposti ad accettare il detour verso le locazioni di raccolta dati. Al crescere di λ la Data Coverage migliora. Le meshgrid e le mappe di calore sono state ottenute tramite interpolazione lineare sui risultati di Data Coverage ottenuti applicando il modello ad ogni singola locazione.

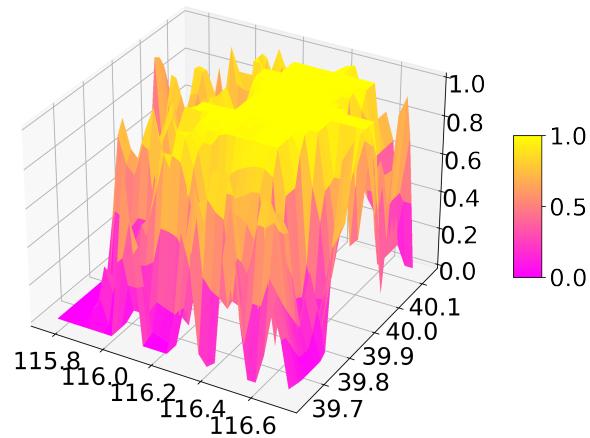


(a) Meshgrid ottenuta per $\lambda = 1/100$

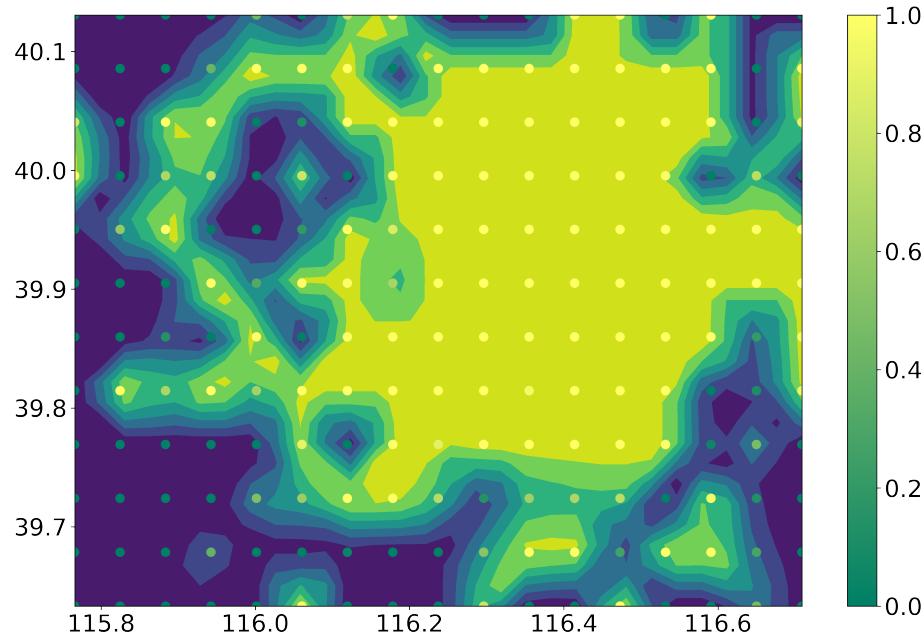


(b) Mappa di calore ottenuta per $\lambda = 1/100$

Figura 4.3: I risultati ottenuti applicando il modello alla griglia ambientale con $\lambda = 1/100$

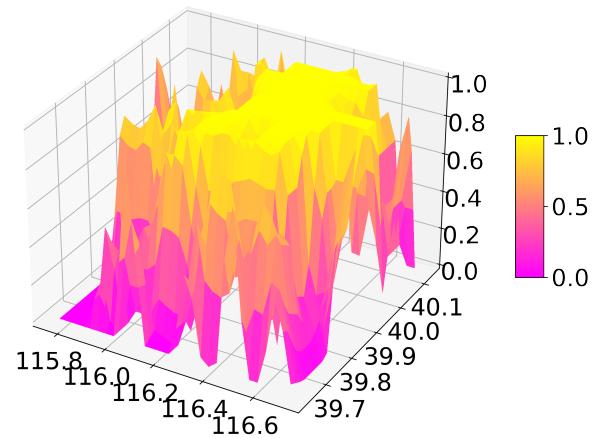


(a) Meshgrid ottenuta per $\lambda = 1/300$

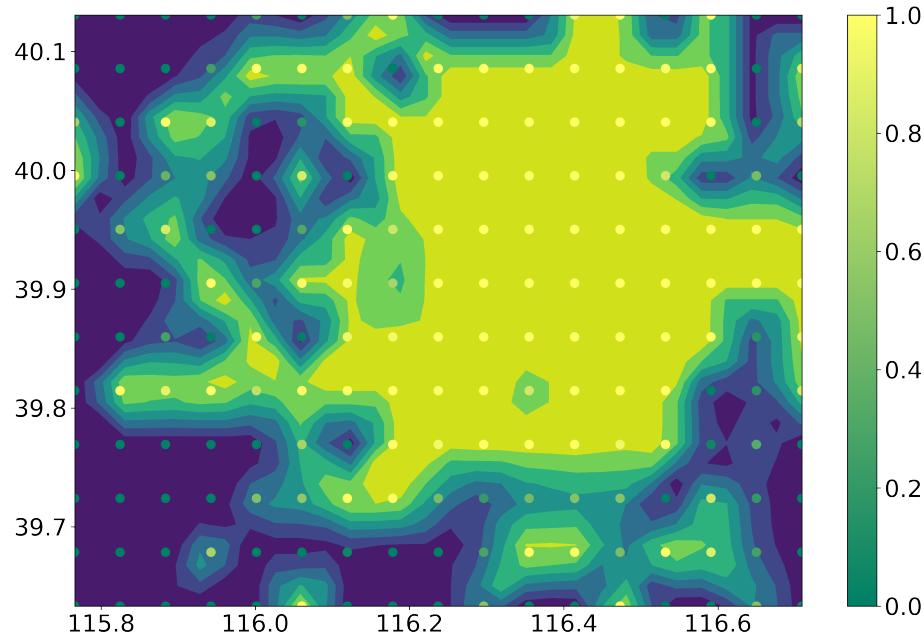


(b) Mappa di calore ottenuta per $\lambda = 1/300$

Figura 4.4: I risultati ottenuti applicando il modello alla griglia ambientale con $\lambda = 1/300$



(a) Meshgrid ottenuta per $\lambda = 1/700$



(b) Mappa di calore ottenuta per $\lambda = 1/700$

Figura 4.5: I risultati ottenuti applicando il modello alla griglia ambientale con $\lambda = 1/700$

4.3 Flussi di traffico

Con flusso di traffico si intende l'interazione tra viaggiatori, siano essi a piedi o su un qualsiasi altro mezzo di trasporto, e l'infrastruttura su cui essi si muovono. I flussi di traffico vengono studiati per migliorare e rendere più efficienti le reti stradali, minimizzando congestioni e massimizzando il throughput della infrastruttura in termini di capacità effettiva di spostamento.

Riguardo al monitoraggio di tali flussi, ci siamo avvalsi di un dataset con le coordinate geografiche di tutte le stazioni della metropolitana di Pechino estratto per mezzo di OSMnx [26]. Ipotizziamo che tali snodi principali siano molto trafficati e siano visitati da una moltitudine di persone. Questo specifico scenario ci permette di valutare quale sia la portata del traffico umano attraverso questi luoghi.

Il monitoraggio tramite MCS dei flussi di traffico, permette di raccogliere dati reali dalle infrastrutture di una Smart City permettendo a chi la gestisce di applicare nuove e migliori politiche urbanistiche e di gestione delle infrastrutture.

4.3.1 OSMnx e metropolitane di Pechino

Grazie alla funzione *pois_from_point()* appartenente a OSMnx [26], abbiamo estratto dal database di OpenStreetMap le coordinate GPS delle stazioni della metropolitana in un raggio di 50Km dal centro di Pechino. Dopo aver ripulito il dataset estratto, abbiamo ottenuto un totale di 331 punti che abbiamo poi serializzato per il successivo calcolo della Data Coverage.

4.3.2 Visualizzazione delle stazioni della metropolitana

La Figura 4.6 mostra le stazioni della metropolitana considerate, stampate grazie a folium [29]. Analogamente alla griglia, la mappa ha in sovraimpressione alcune delle traiettorie del dataset. I cerchi blu rappresentano le singole stazioni della metropolitana.

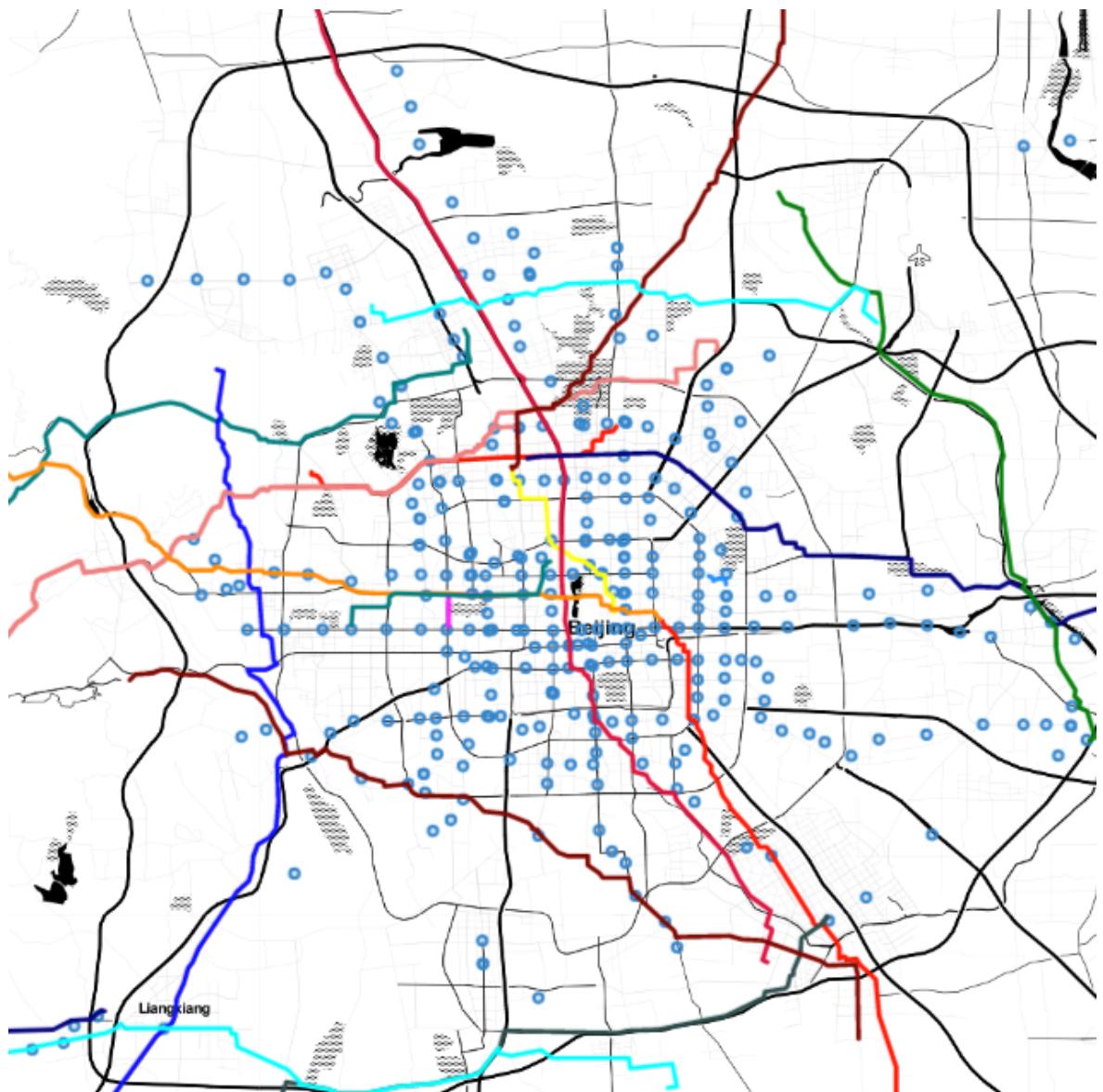
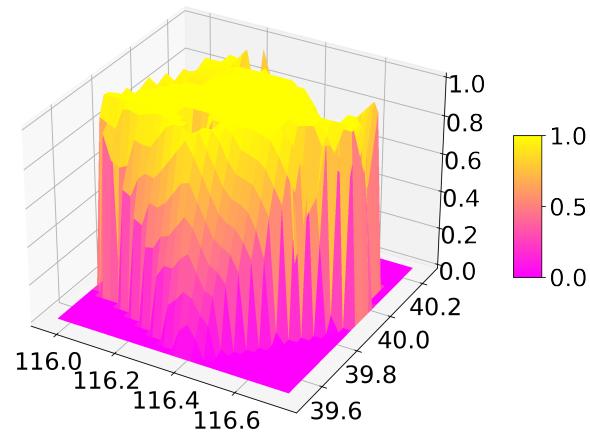


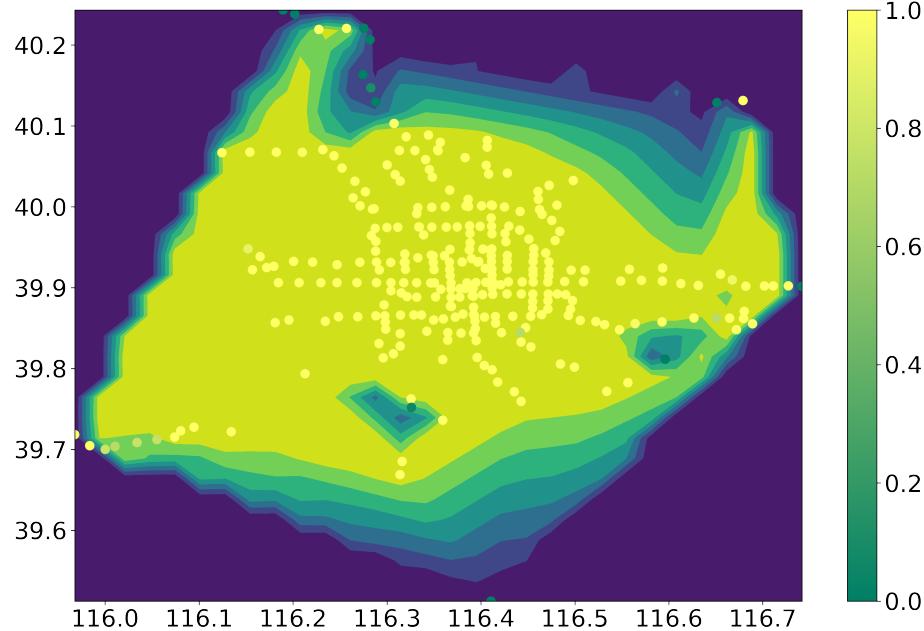
Figura 4.6: Le stazioni della metropolitana stampate grazie a folium

4.3.3 Risultati del modello di coverage

Le Figure 4.7, 4.8, e 4.9, mostrano i risultati ottenuti applicando il modello con differenti valori di lambda sulle uscite della metropolitana estratte per mezzo di OSMnx [26].

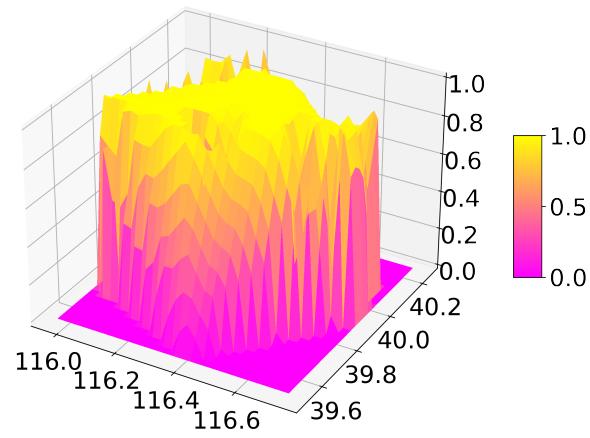


(a) Meshgrid ottenuta per $\lambda = 1/100$

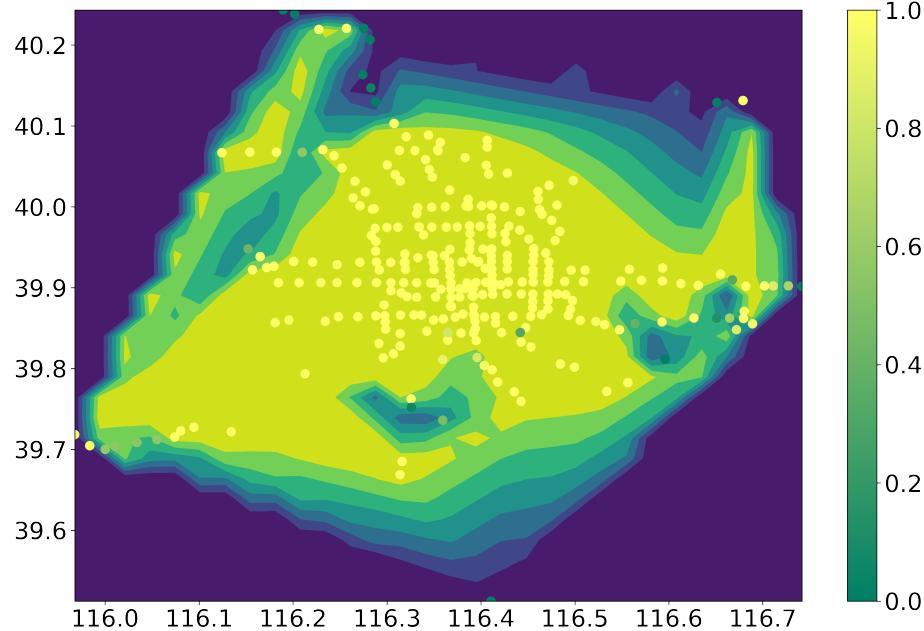


(b) Mappa di calore ottenuta per $\lambda = 1/100$

Figura 4.7: I risultati ottenuti applicando il modello alle uscite della metropolitana con $\lambda = 1/100$

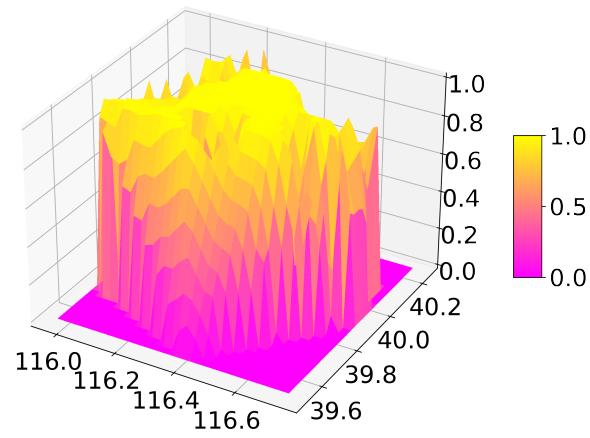


(a) Meshgrid ottenuta per $\lambda = 1/300$

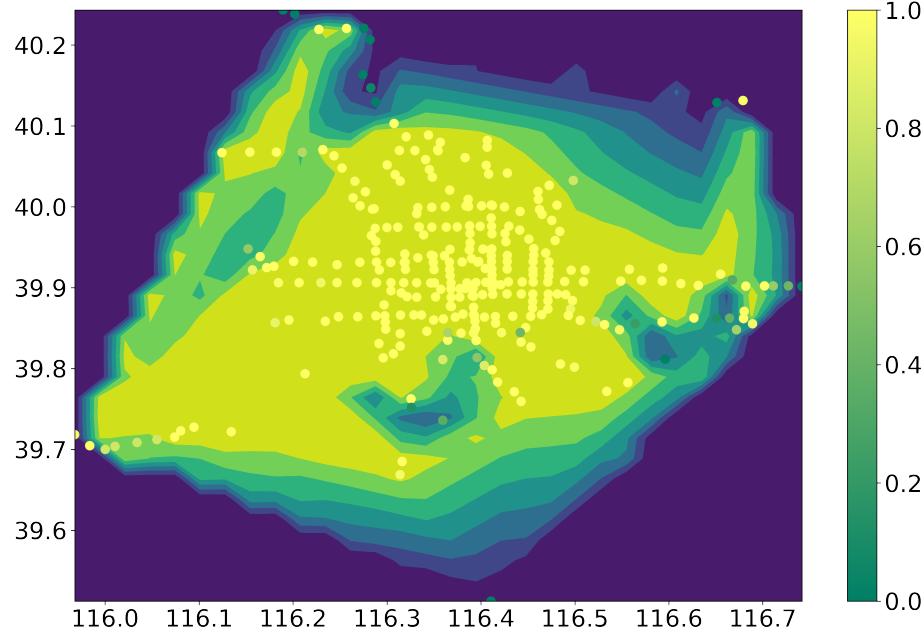


(b) Mappa di calore ottenuta per $\lambda = 1/300$

Figura 4.8: I risultati ottenuti applicando il modello alle uscite della metropolitana con $\lambda = 1/300$



(a) Meshgrid ottenuta per $\lambda = 1/700$



(b) Mappa di calore ottenuta per $\lambda = 1/700$

Figura 4.9: I risultati ottenuti applicando il modello alle uscite della metropolitana con $\lambda = 1/700$

4.4 Punti di interesse

I punti di interesse dislocati in tutta la città di Pechino ci permettono di verificare quanto e dove le persone siano più interessate ad andare. Il data set dei punti di interesse è stato estratto utilizzando OSMnx analogamente a quello delle metropolitane.

Questo tipo di MCS ha un'utilità nel verificare quali siano i luoghi più popolari, e potrebbe essere interessante per studi di tipo turistico e culturale.

4.4.1 Punti di interesse considerati

I punti che sono stati scelti sono tutti monumenti, piazze, edifici storici o luoghi di aggregazione sociale come ad esempio centri commerciali. Analogamente a quanto fatto con le uscite della metropolitana, abbiamo estratto le posizioni grazie a OSMnx [26] che permette un filtraggio a grana fine basato su tags per trovare determinate tipologie di punti di interesse.

Dopo la pulizia abbiamo serializzato il dataset, contentente un totale di 956 punti di interesse sparsi in tutta la città.

La Figura 4.10 mostra i punti di interesse presi in considerazione, stampati per mezzo di folium [29]. Anche in questo caso i cerchi blu rappresentano le singole locazioni e in sovraimpressione sono presenti alcune delle traiettorie di Augmented Geolife.

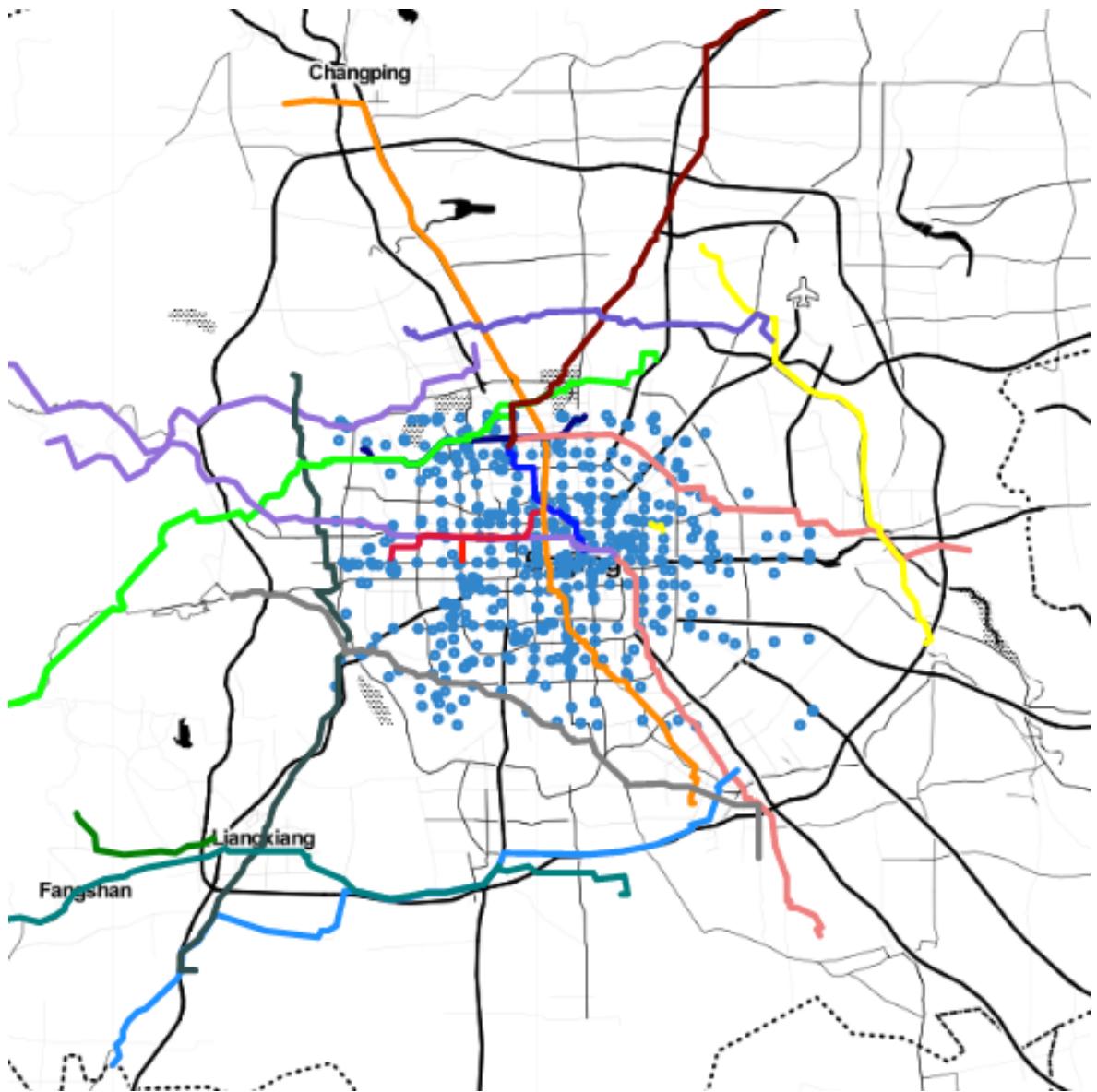
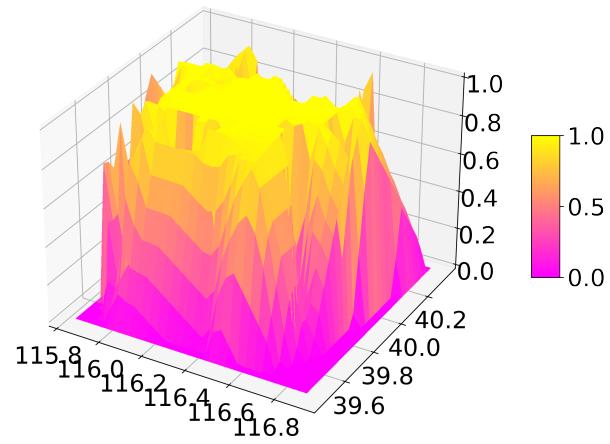


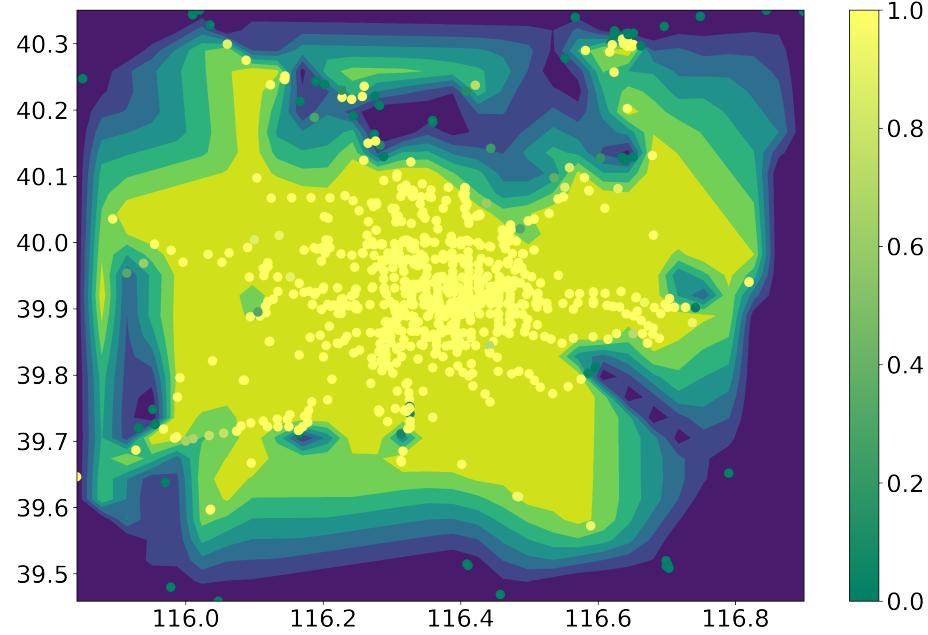
Figura 4.10: I punti di interesse estratti visualizzati grazie a folium

4.4.2 Risultati del modello di coverage

Le Figure 4.11, 4.12, e 4.13, mostrano i risultati ottenuti applicando il modello con differenti valori di lambda sui punti di interesse estratti per mezzo di OSMnx [26].

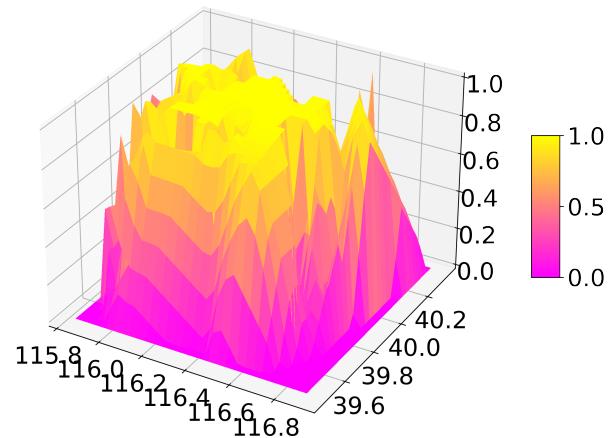


(a) Meshgrid ottenuta per $\lambda = 1/100$

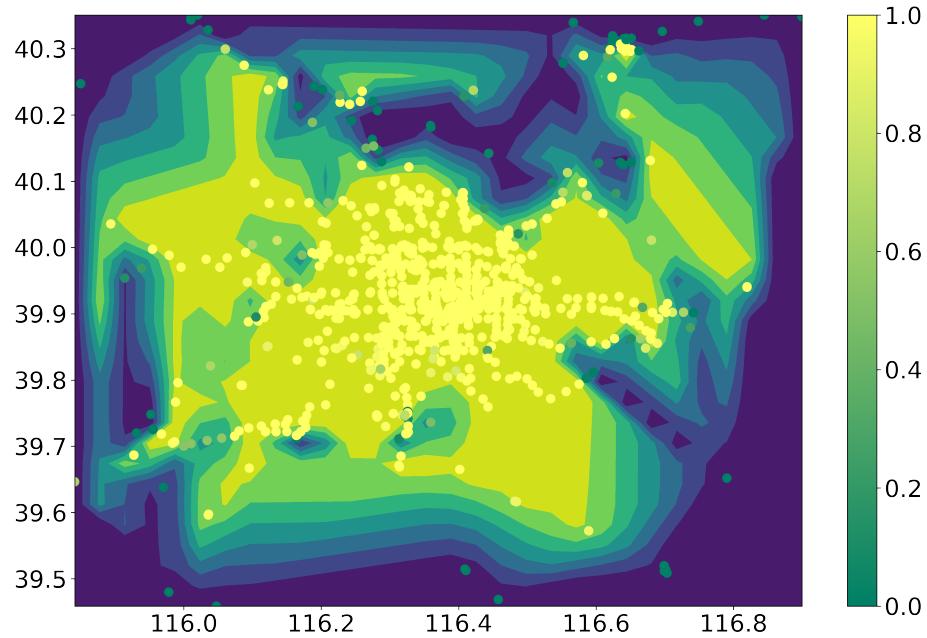


(b) Mappa di calore ottenuta per $\lambda = 1/100$

Figura 4.11: I risultati ottenuti applicando il modello ai punti di interesse con $\lambda = 1/100$

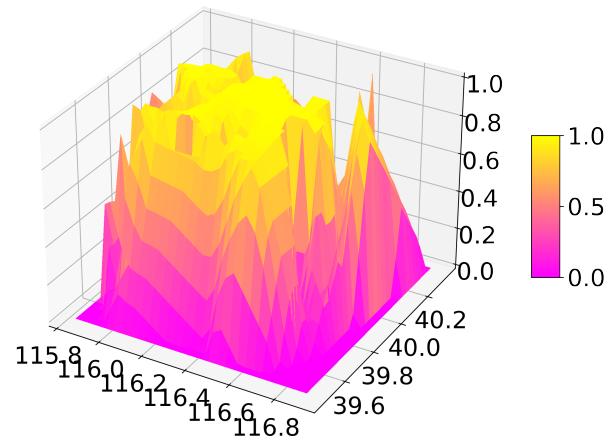


(a) Meshgrid ottenuta per $\lambda = 1/300$

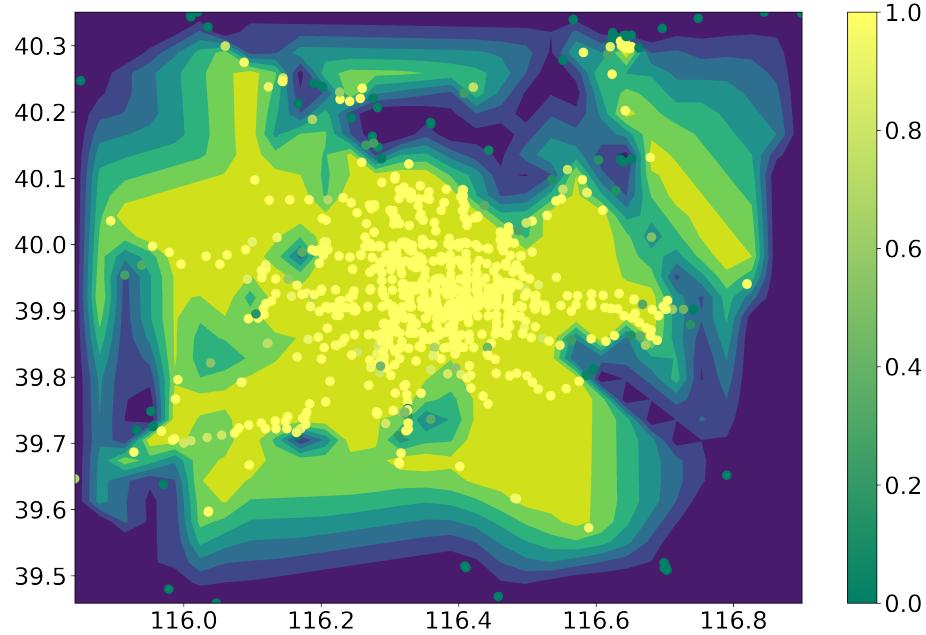


(b) Mappa di calore ottenuta per $\lambda = 1/300$

Figura 4.12: I risultati ottenuti applicando il modello ai punti di interesse con $\lambda = 1/300$



(a) Meshgrid ottenuta per $\lambda = 1/700$



(b) Mappa di calore ottenuta per $\lambda = 1/700$

Figura 4.13: I risultati ottenuti applicando il modello ai punti di interesse con $\lambda = 1/700$

4.5 Analisi dei risultati ottenuti

Procediamo adesso con una analisi dei risultati ottenuti per ciascuno scenario e, più in generale, per tutta la sperimentazione.

Le meshgrid e le mappe di calore mostrano la Data Coverage in relazione alle diverse locazioni di raccolta dati. La differente scelta delle locazioni ha chiaramente influenzato la mappa di Data Coverage risultante; questo è particolarmente evidente nel caso degli scenari sui flussi di traffico e sui punti di interesse, in cui la relativa sparsità delle locazioni, e la loro assenza nelle zone periferiche della città, porta a dei risultati peculiari. Ciononostante è bene ricordare che, per i fini specifici del monitoraggio dei flussi di traffico e dei punti di interesse, non siamo interessati a zone con assenza di traffico umano, di conseguenza il risultato ci sembra soddisfacente, ed in linea con le aspettative per ciascuno scenario. Nelle mappe di calore riguardanti lo scenario di monitoraggio ambientale, si riescono a distinguere bene zone periferiche affatto trafficate, poiché la natura delle locazioni, posizionate a griglia equispaziata, ci permette di valutare con efficacia la Data Coverage in tutta la zona di sensing. Il centro di Pechino ha una Data Coverage praticamente certa e ben distribuita, come si nota dalle figure.

Fatte queste considerazioni, risulta normale che, ove le locazioni diventino più sparse e distanti dalle traiettorie, la Data Coverage decresca molto rapidamente fino ad arrivare a valori nulli, nelle zone in cui non sono presenti locazioni. Nei grafici sono evidenti zone con picchi negativi di Data Coverage racchiusi all'interno di aree con una Data Coverage pressoché perfetta. Questi picchi negativi sono dovuti alla mancanza di traiettorie in quelle specifiche zone, e ci suggeriscono la possibilità di ampliare la strategia di raccolta con piani ad hoc per le suddette zone.

In tutti e tre gli scenari si nota una differenza minima al variare di λ . Questo è dovuto probabilmente al fatto che le traiettorie, pur aumentate, risultano essere insufficienti a modellare con precisione i pattern di spostamento di persone reali. Per aggiungere realismo alle traiettorie si potrebbero introdurre dei punti di fermata intermedi tra il punto di partenza e quello di arrivo di ciascuna traiettoria sintetica.

Infine, ci sembra opportuno soffermarci brevemente sui limiti del modello. Poiché

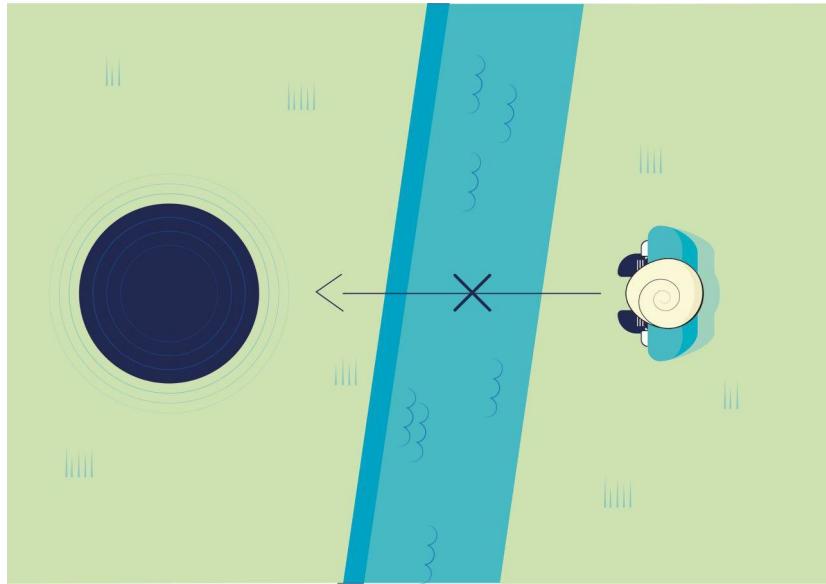


Figura 4.14: Una barriera fisica, in questo caso un corso d'acqua, impedisce all'utente di raggiungere la zona di sensing.

λ deve modellare il valore di collaboratività degli utenti, è evidente che tale parametro debba essere necessariamente una semplificazione della realtà: con i dati forniti da GeoLife [1, 2, 3] non possiamo sapere veramente se un utente sarebbe disposto o meno ad accettare una deviazione. Un altro limite degno di nota è l'aver usato una distribuzione esponenziale nel modello di coverage. Nulla vieta che la probabilità di Data Coverage segua un andamento di tipo diverso, per questo sarebbe interessante provare il modello con altri tipi di distribuzione. Un'ultima considerazione riguarda le barriere architettoniche e fisiche: nel modello utilizziamo la distanza da una locazione per calcolare la probabilità che un utente sia disposto a raggiungerla e raccogliere i dati, ma glissiamo sull'esistenza di eventuali barriere architettoniche e fisiche che potrebbero impedire all'utente di raggiungere agilmente la locazione. Si pensi ad esempio ad una locazione che si trova sulla riva sinistra di un fiume senza ponti: il modello restituirebbe una probabilità di copertura per l'utente che si trova sulla riva destra, tuttavia la raccolta dati sarebbe impossibile, poiché l'utente non avrebbe modo di attraversare il fiume. La Figura 4.14 mostra questo tipo di problematica.

Pur con tutte le semplificazioni citate, i risultati dell'applicazione del modello ci sembrano soddisfacenti, soprattutto in vista di ulteriori sviluppi nella tecnologia del MCS e del miglioramento dei dataset disponibili.

Capitolo 5

Conclusioni

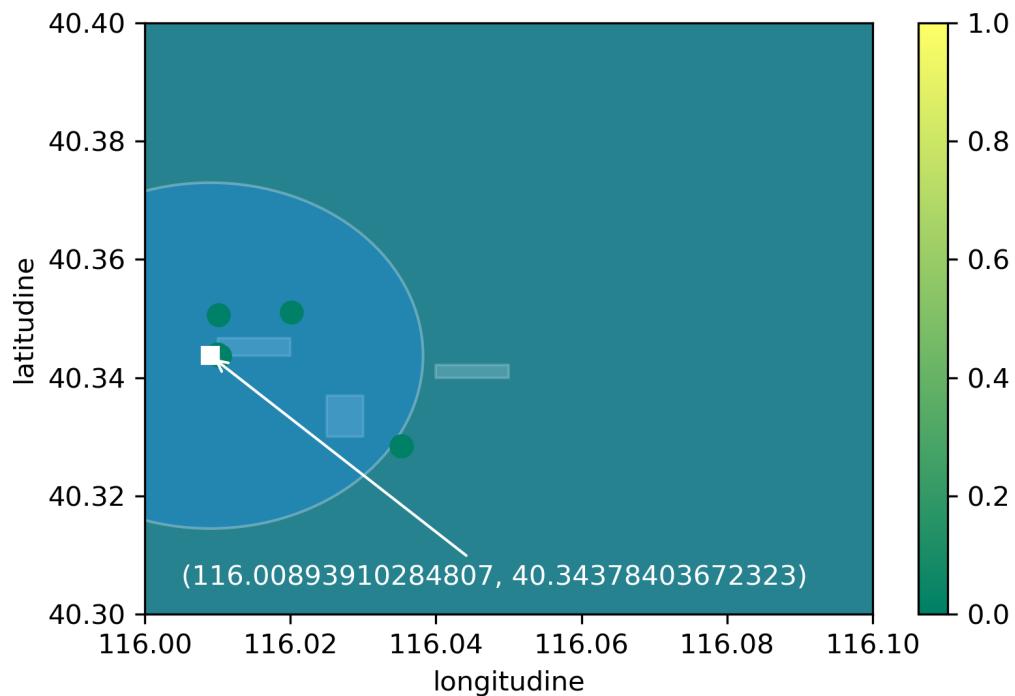
Il nostro obiettivo principale è quello di applicare un modello di Data Coverage facilmente replicabile a vari scenari di Mobile Crowdsensing e, contemporaneamente, di sviluppare una strategia efficace per l'arricchimento sintetico dei dataset di mobilità disponibili pubblicamente. Librerie versatili e complete come scikit-mobility [4] ci hanno aiutato a svolgere compiti affatto banali e a ottenere metriche utili sul nostro dataset. I tool grafici poi, ci hanno permesso di rappresentare risultati di Data Coverage ottenuti grazie al modello descritto nel Capitolo 2 in modo chiaro e diretto. Grazie a tali librerie, e agli strumenti esplorativi della Data Science, siamo riusciti nel nostro intento di modellare la Data Coverage e di calcolarla sul dataset arricchito, producendo così delle mappe probabilistiche di Coverage che possano essere utilizzate in lavori futuri. L'implementazione del modello e la sua successiva applicazione a scenari diversi per finalità e composizione è, a nostro avviso, il risultato più interessante di questa tesi. Ci riteniamo soddisfatti dei risultati che abbiamo raggiunto in questi ambiti, tuttavia ci sono alcuni aspetti del nostro studio che potrebbero essere ampliati e approfonditi in futuro; grazie alla Data Coverage che abbiamo calcolato, infatti, è possibile pianificare uno sviluppo del lavoro che sia guidato da tale metrica. Vorremmo di seguito descrivere brevemente tre dei possibili sviluppi in questo senso.

Utilizzo della Data Coverage per il posizionamento di UAV Base Stations
Grazie alla Data Coverage calcolata, sarà possibile scegliere il posizionamento ottimale

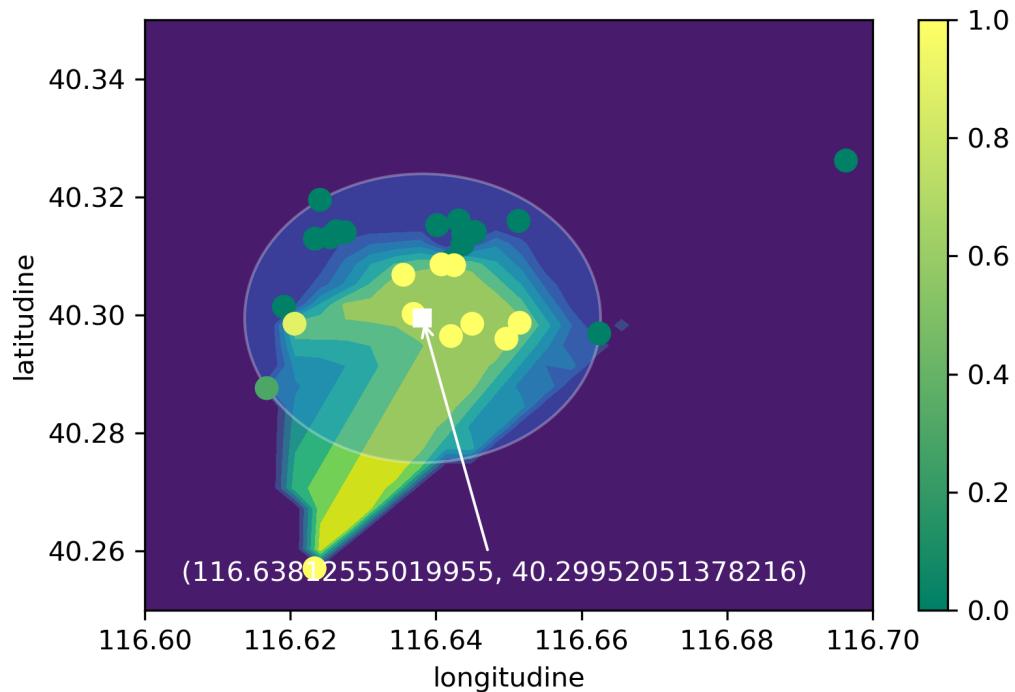
per una o più stazioni di ricarica per UAV (unmanned aerial vehicles). Queste stazioni fungeranno da punti di partenza e arrivo per droni a pilotaggio remoto (possibilmente automatico, e schedulato sulla base della evoluzione dinamica della data coverage nella area di sensing) che sorvolino le aree con una Data Coverage pessima o inesistente, al fine di raccogliere loro stessi i dati mancanti. Tale attività rappresenta la naturale continuazione di questo lavoro di tesi su cui altri studenti potranno condurre ulteriori attività di ricerca. Le immagini presenti in Figura 5.1 mostrano un possibile posizionamento per le stazioni di ricarica.

Calcolo della Coverage a partire dalla mobilità degli utenti tramite strumenti predittivi Partendo dalle traiettorie note degli utenti, è possibile sviluppare strumenti predittivi che utilizzino metriche quali la frequenza e la periodicità degli spostamenti dei singoli individui al fine di predirne il posizionamento futuro. Questo permetterebbe lo sviluppo di modelli di Data Coverage predittivi, utili nella pianificazione di campagne di MCS continuative e dinamiche, che sfruttino queste metriche per ottimizzare l'allocazione di risorse umane a materiali in relazione alle necessità future.

Uso della Data Coverage per ottimizzare la progettazione di architetture di MCS adattive alla mobilità ed alla quantità di dati Questo punto si collega strettamente ai due precedenti. La possibilità di predire la Data Coverage futura, assieme alla versatilità dei droni, sono strumenti utili alla progettazione di architetture MCS il più possibile adattive e dinamiche. Nel recente passato, ad esempio, le nostre abitudini giornaliere sono cambiate molto a causa della pandemia di Covid-19 [47]: le persone tendono a muoversi meno e a evitare assembramenti, riducendo al tempo stesso l'utilizzo di trasporto pubblico. Date queste premesse, è senz'altro interessante teorizzare e sviluppare architetture MCS resilienti a questo tipo di modificazioni strutturali nel comportamento umano.



(a) Posizionamento su mappa con cattiva Data Coverage. I rettangoli chiari rappresentano zone in cui non è possibile posizionare la stazione di ricarica.



(b) Posizionamento su mappa con Data Coverage accettabile.

Figura 5.1: Esempi di posizionamento di una stazione di ricarica per UAV su una mappa di Data Coverage. I quadrati bianchi rappresentano la posizione della stazione di ricarica, il cerchio attorno ad essi rappresenta il raggio della zona coperta dal drone.

Ringraziamenti

Grazie al Prof. Stefano Chessa per avermi dato fiducia, al Dott. Michele Girolami, per la sua pazienza e per avermi insegnato le basi della Data Science, a Talia che mi sostiene ogni giorno, ai miei genitori, Andrea e Cristina, a mio fratello Edoardo, ai miei zii Stefano e Barbara e al resto della mia famiglia, a Giacomo per i consigli sempre utili, a Leonardo per 22 anni di amicizia, a tutti i miei amici.

Bibliografia

- [1] Y. Zheng, X. Xie, and W.-Y. Ma, “Mining interesting locations and travel sequences from gps trajectories,” in *Proceedings of International conference on World Wide Web 2009*, April 2009. WWW 2009.
- [2] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, *Understanding Mobility Based on GPS Data*, p. 312–321. New York, NY, USA: Association for Computing Machinery, 2008.
- [3] Y. Zheng, X. Xie, and W. Ma, “Geolife: A collaborative social networking service among user, location and trajectory,” *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010.
- [4] L. Pappalardo, F. Simini, G. Barlacchi, and R. Pellungrini, “scikit-mobility: a python library for the analysis, generation and risk assessment of mobility data,” 2019.
- [5] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, “A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities,” *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2419–2465, 2019.
- [6] R. K. Ganti, F. Ye, and H. Lei, “Mobile crowdsensing: current state and future challenges,” *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

- [7] L. G. Jaimes, I. J. Vergara-Laurens, and A. Raij, “A survey of incentive techniques for mobile crowd sensing,” *IEEE Internet of Things Journal*, vol. 2, no. 5, pp. 370–380, 2015.
- [8] D. A. G. L. Pournajaf, L. Xiong and V. Sunderam, “A survey on privacy in mobile crowd sensing task management,” Tech. Rep. TR-2014-002, Department of Mathematics and Computer Science, Emory University, 2014, 2014.
- [9] O. Alvear, C. Calafate, J.-C. Cano, and P. Manzoni, “Crowdsensing in smart cities: Overview, platforms, and environment sensing issues,” *Sensors*, vol. 18, p. 460, Feb 2018.
- [10] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of things for smart cities,” *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.
- [11] T. J. Matarazzo, P. Santi, S. N. Pakzad, K. Carter, C. Ratti, B. Moaveni, C. Osgood, and N. Jacob, “Crowdsensing framework for monitoring bridge vibrations using moving smartphones,” *Proceedings of the IEEE*, vol. 106, no. 4, pp. 577–593, 2018.
- [12] G. Cardone, A. Cirri, A. Corradi, and L. Foschini, “The participact mobile crowd sensing living lab: The testbed for smart cities,” *IEEE Communications Magazine*, vol. 52, no. 10, pp. 78–85, 2014.
- [13] P. Bellavista, S. Chessa, L. Foschini, L. Gioia, and M. Girolami, “Human-enabled edge computing: Exploiting the crowd as a dynamic extension of mobile edge computing,” *IEEE Communications Magazine*, vol. 56, no. 1, pp. 145–155, 2018.
- [14] P. Bellavista, D. Belli, S. Chessa, and L. Foschini, “A social-driven edge computing architecture for mobile crowd sensing management,” *IEEE Communications Magazine*, vol. 57, no. 4, pp. 68–73, 2019.
- [15] D. Belli, S. Chessa, L. Foschini, and M. Girolami, “The rhythm of the crowd: Properties of evolutionary community detection algorithms for mobile edge selection,” *Pervasive and Mobile Computing*, vol. 67, p. 101231, 2020.

- [16] D. Belli, S. Chessa, L. Foschini, and M. Girolami, “A probabilistic model for the deployment of human-enabled edge computing in massive sensing scenarios,” *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2421–2431, 2020.
- [17] E. Pepe, P. Bajardi, L. Gauvin, F. Privitera, B. Lake, C. Cattuto, and M. Tizzoni, “Covid-19 outbreak response, a dataset to assess mobility changes in italy following national lockdown,” *Scientific Data*, vol. 7, p. 230, Jul 2020.
- [18] Apple, “Report on mobility trends.” <https://covid19.apple.com/mobility>, 2020.
- [19] Facebook, “Covid-19 mobility data network.” <https://dataforgood.fb.com/docs/covid19/#covid-19-mobility-data-network>, 2020.
- [20] H. S. Badr, H. Du, M. Marshall, E. Dong, M. M. Squire, and L. M. Gardner, “Association between mobility patterns and covid-19 transmission in the usa: a mathematical modelling study,” *The Lancet Infectious Diseases*, 2020.
- [21] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.
- [22] K. Thomas, R.-K. Benjamin, P. Fernando, G. Brian, B. Matthias, F. Jonathan, K. Kyle, H. Jessica, G. Jason, C. Sylvain, and et al., “Jupyter notebooks; a publishing format for reproducible computational workflows,” *Stand Alone*, vol. 0, no. Positioning and Power in Academic Publishing: Players, Agents and Agendas, p. 87–90, 2016.
- [23] “Anaconda software distribution,” 2020.
- [24] T. pandas development team, “pandas-dev/pandas: Pandas.” <https://doi.org/10.5281/zenodo.3509134>, Feb. 2020.
- [25] Wes McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.

- [26] G. Boeing, “Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks,” *Computers Environment and Urban Systems*, vol. 65, pp. 126–139, 07 2017.
- [27] OpenStreetMap contributors, “Planet dump retrieved from <https://planet.osm.org> .” <https://www.openstreetmap.org>, 2017.
- [28] K. Simonov, “PyYAML - The next generation YAML parser and emitter for Python..” <https://github.com/yaml/pyyaml>.
- [29] R. Story, “Python Data, Leaflet.js Maps.” <https://github.com/python-visualization/folium>.
- [30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Pas-sos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [31] M. Waskom, O. Botvinnik, D. O’Kane, P. Hobson, S. Lukauskas, D. C. Gemper-line, T. Augspurger, Y. Halchenko, J. B. Cole, J. Warmenhoven, J. de Ruiter, C. Pye, S. Hoyer, J. Vanderplas, S. Villalba, G. Kunter, E. Quintero, P. Bachant, M. Martin, K. Meyer, A. Miles, Y. Ram, T. Yarkoni, M. L. Williams, C. Evans, C. Fitzgerald, Brian, C. Fonnesbeck, A. Lee, and A. Qaleh, “mwaskom/seaborn: v0.8.1 (september 2017),” Sept. 2017.
- [32] “Datashader is a data rasterization pipeline for automating the process of creating meaningful representations of large amounts of data. .” <https://github.com/bokeh/datashader>.
- [33] A. A. Hagberg, D. A. Schult, and P. J. Swart, “Exploring network structure, dynamics, and function using networkx,” in *Proceedings of the 7th Python in Science Conference* (G. Varoquaux, T. Vaught, and J. Millman, eds.), (Pasadena, CA USA), pp. 11 – 15, 2008.

- [34] C. F. F. Karney. <https://geographiclib.sourceforge.io/>.
- [35] “Manipulation and analysis of geometric objects in the Cartesian plane. .” <https://github.com/Toblerity/Shapely>.
- [36] “Python interface to proj (cartographic projections and coordinate transformations library)..” <https://github.com/pyproj4/pyproj>.
- [37] Autodesk, “To work with plot files.” <https://knowledge.autodesk.com/support/autocad/learn-explore/caas/CloudHelp/cloudhelp/2020/ENU/AutoCAD-Core/files/GUID-6405BB6F-4350-4F0A-AC09-3E237F20FBD6-htm.html>.
- [38] Y. Zheng, H. Fu, X. Xie, W.-Y. Ma, and Q. Li, *Geolife GPS trajectory dataset - User Guide*, geolife gps trajectories 1.1 ed., July 2011. Geolife GPS trajectories 1.1.
- [39] R. Hariharan and K. Toyama, “Project lachesis: Parsing and modeling location histories,” in *Geographic Information Science* (M. J. Egenhofer, C. Freksa, and H. J. Miller, eds.), (Berlin, Heidelberg), pp. 106–124, Springer Berlin Heidelberg, 2004.
- [40] “Medoid.” <https://en.wikipedia.org/wiki/Medoid>, Sep 2020.
- [41] T. Z. Baharav and D. N. C. Tse, “Ultra fast medoid identification via correlated sequential halving,” *CoRR*, vol. abs/1906.04356, 2019.
- [42] V. Bagaria, G. M. Kamath, V. Ntranos, M. J. Zhang, and D. Tse, “Medoids in almost linear time via multi-armed bandits,” 2017.
- [43] “K-d tree.” https://en.wikipedia.org/wiki/K-d_tree, Sep 2020.
- [44] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numer. Math.*, vol. 1, p. 269–271, Dec. 1959.

- [45] M. C. Gonzalez, C. Hidalgo, and A.-L. Barabasi, “Understanding individual human mobility patterns,” *Nature*, vol. 453, pp. 779–82, 07 2008.
- [46] L. Pappalardo, S. Rinzivillo, Z. Qu, D. Pedreschi, and F. Giannotti, “Understanding the patterns of car travel,” *The European Physical Journal Special Topics*, vol. 215, pp. 61–73, Jan. 2013.
- [47] L. D. Renzo, P. Gualtieri, F. Pivari, L. Soldati, A. Attinà, G. Cinelli, C. Leggeri, G. Caparello, L. Barrea, F. Scerbo, E. Esposito, and A. D. Lorenzo, “Eating habits and lifestyle changes during COVID-19 lockdown: an italian survey,” *Journal of Translational Medicine*, vol. 18, June 2020.