A clustering algorithm to organize satellite hotspots data for the purpose of tracking bushfires remotely

by Weihao Li, Emily Dodwell, Dianne Cook

Abstract An abstract of less than 150 words.

Introduction

- What is the data, generic structure
- Lit review: Spatio-temporal clustering. Algorithms for tracking movement.
- Bushfire literature review?

Algorithm

Data pre-processing

To track bushfires in Australia remotely, we used hotspots data taken from the Himawari-8 satellite. The hotspots data is available on the JAXA FTP site in CSV file format, and only the data during October 2019 to March 2020 was downloaded. It contains records of 1989572 hotspots for 5 months in the full disk of 140 °east longitude. We only kept records of hostpots within the boundary of Australia, which reduced to 1526080 records. Besides, a threshold (irradiance over 100 watts per square metre) for fire power was used to filter hotspots data, which can limit the influence of radiation from other objects. For the convention of this algorithm, a sequence of discrete timestamps was needed. We calculated the hourly difference between each record and the earliest record, then rounded them to integers. The end result was a 1010794×4 dataset. The four fields were the unique identifier for each row, the longitude, the latitude and the indicator of timestamps respectively. The code to implement this process is in "main.R". Read in CSV files was done by using package `readr`. Data manipulation was done by using package `dplyr`. High resolution Australia vector map was obtained from package `rnaturalearth`. Operation of geometric intersection between hotspots and Australia map was done by using package `sf`.

Steps

After the data pre-processing, this algorithm ran in a time-series manner. It first selected entries of the first timestamps, which was the first hour in the hotspots data. The algorithm then calculated the matrix of pairwise geodesic distances between all points being selected. With the geodesic distances matrix, an "adjacent distance" as one of the hyperparameters in this algorithm was used to determine the adjacency matrix. If a geodesic distance between two points was less than the "adjacent distance", the corresponding entry in the adjacency matrix would be assigned with integer 1, otherwise it would be assigned with integer 0. Normally, this "adjacent distance" would be set between 0 to 100000 meters. Using the adjacency matrix, the algorithm then constructed a undirected unweighted graph. For each connected component in this graph, a unique integer was assigned as the membership. In our hotspots data, components could be recognised as bushfires. Points in the same component shared with the same membership. Meanwhile, the longitude and the latitude of centroids in each component were calculated by taking the average of longitude and the average of latitude for all points in the corresponding component. Those centroids along with memberships would then be recorded and labelled as active groups. In other words, their "active" attributes were assigned with integer 0.

When the algorithm moved to the next timestamps, it subtracted 1 from "active" attributes. Another hyperparameter "group active time" was used for selecting active groups. Conventionally, "group active time" was set to be 24 hours. If any centroid had an "active" attribute greater than the negative of "group active time", it would be selected as active groups.

For the second and the later timestamps, the algorithm first combined centroids of active groups with the hotspots data in the corresponding timestamps. It then calculated geodesic distances matrix, filled adjacency matrix and constructed graph as before. There was an additional step which was to find the nearest active group within the same component for each point. If a point shared the same component with active groups, it would be assigned with the membership of the nearest active group.

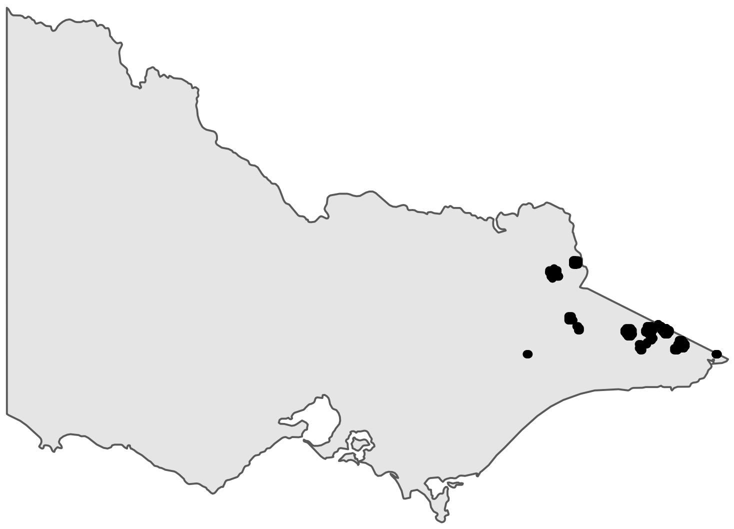

Figure 1: Hotspot locations in Victoria during2019-2020 season.

Otherwise, points shared with the same component would be assigned with a new membership. Therefore, points in one component would not necessary had the same membership if there were more than one active groups within a component. All centroids of active groups and new group would then be recalculated and updated using only the current timestamps hotspots data. Their "active" attributes were set to be 0.

This algorithm worked till the last timestamps. The end result was a vector of memberships with length equal to number of rows in hotspots data and a time-series record of all groups.

For computational performance, we stored the hostpots data in a SQLite database. Relevant operation was done by using packages `DBI` and `RSQLite`. Geodesic distances matrix was calculated using package `geodist`. Graph operation was done by using package `igraph`.

The code implemented this algorithm is "clustering.R".

(Code in clustering.R does this, needs cleaning up)

1. Divide hotspots by hour
2. Start from the first hour
3. Connect adjacent hotspots and active centroids (3km)
4. For each point, if there is a connected nearest active centroid, join its group
5. Otherwise, create a new group for each connected graph
6. Compute centroid for each group
7. Keep the group active until there is no new hotspots join the group within 24 hours
8. Repeat this process to the last hour

Effects of parameter choices

Using the resulting data

Determining the ignition point and time for individual fires

Tracking fire movement

Allocating resources for future fire prevention

Merging data with camp sites, CFA, roads, . . .

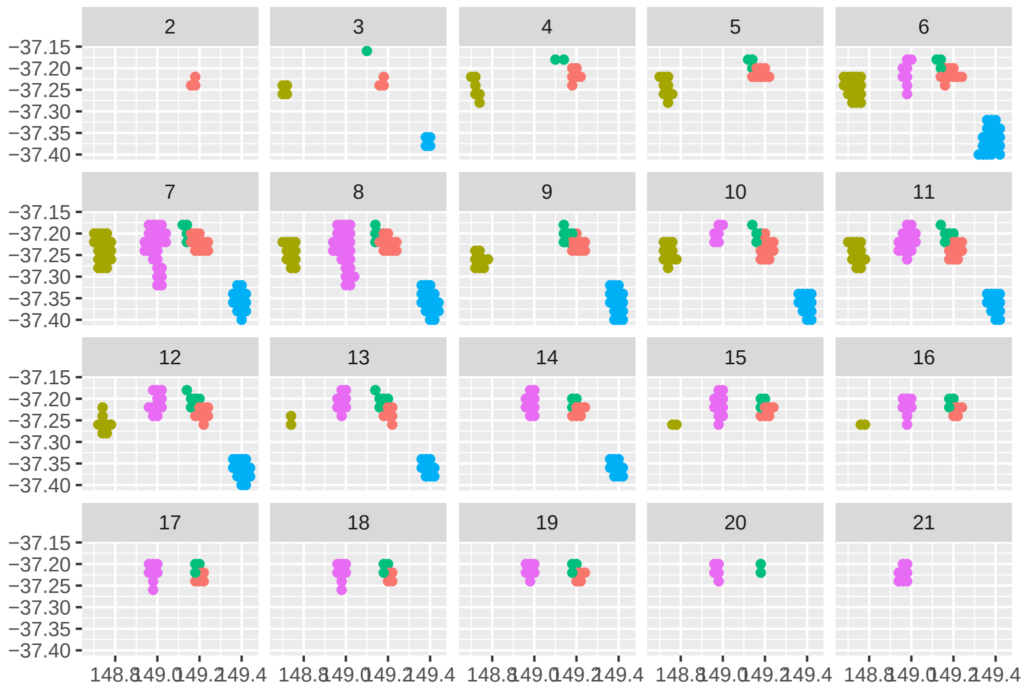

Figure 2: Main clusters in one area over time.

Summary

Acknowledgements

- The code and files to reproduce this work are at XXX
- Data on hotspots can be downloaded from XXX

Weihao Li
Monash University
line 1
line 2
wlii0039@student.monash.edu

Emily Dodwell
AT&T
line 1
line 2
emily@research.att.com

Dianne Cook
Monash University
line 1
line 2
dicook@monash.edu