Segmenting and tracking populations in raster data from Google Earth Engine

Chanon Olley, Radu Cimpeanu*

Mathematics Institute, University of Warwick, Coventry, UK

chanon.olley@warwick.ac.uk
radu.cimpeanu@warwick.ac.uk

WARWICK

UNDERGRADUATE RESEARCH SUPPORT SCHEME

Introduction

Image segmentation is the process of dividing an image into subsets of pixels based on a partitioning scheme (hard assignment) or probability distribution (soft assignment). In this project, we consider the task of image segmentation for identifying communities of population in population count raster data from Google Earth Engine [1]. Furthermore, we devise a method of inferring inter-layer connections between partitions in consecutive frames.

We outline a pipeline for image segmentation and tracking as follows:

- Determining the velocities of features in an image via optical flow.
- Discovering latent features of pixels via an autoencoder neural network with reconstruction MSE of 2e-4 ($\pm$7e-5) on normalised windows of size $7 \times 7 \times 3$.
- Clustering latent features via KMeans and DBSCAN.
- Inferring inter-layer connections between partitions in consecutive frames via cosine similarity of optical flow vectors.

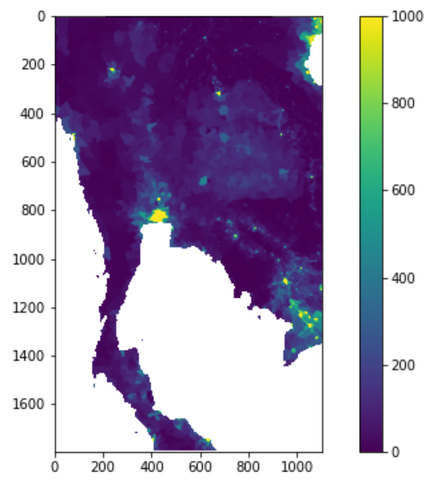

Figure 1: GPWv411 Population Count for year 2000 over Thailand. Raster data clipped to 1000 per pixel (30 arc-seconds).

1) Optical flow

Optical flow [2] is a numerical scheme to infer the velocities of pixels in an image. It assumes that there is a differentiable brightness pattern $E(x, y, t)$ and that the brightness of a particular point in the pattern is constant, so that

$$\frac{\mathrm{d}E}{\mathrm{d}t} = 0,$$

which by chain rule gives the equation

$$\frac{\partial E}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial E}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} + \frac{\partial E}{\partial t} = 0.$$

We can approximate the partial derivatives either symmetrically, forwards or backwards in the time component. Furthermore, we want to minimize the departure from smoothness in the velocity flow, so we have two minimization objectives ϵ_b and ϵ_c,

$$\epsilon_b = E_x u + E_y v + E_t$$
$$\epsilon_c^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2,$$

where E_x, E_y and E_t are the partial derivatives of E, and u and v are the velocity fields we want to calculate. Via calculus of variations we can derive an iterative scheme.

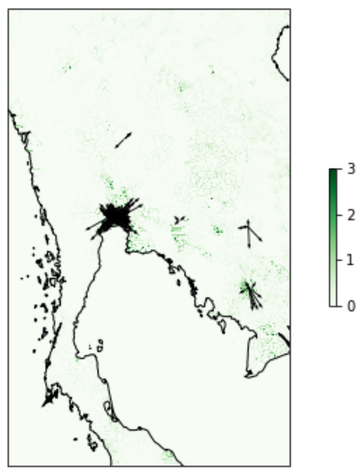

Figure 2: Optical flow calculated for year 2005 of Population Count raster data over Thailand using symmetric derivative estimates. Only vectors in the top 0.01% percentile of magnitudes shown. Growth in area of Bangkok is most prominent feature.

2) Latent representation of pixels with an autoencoder

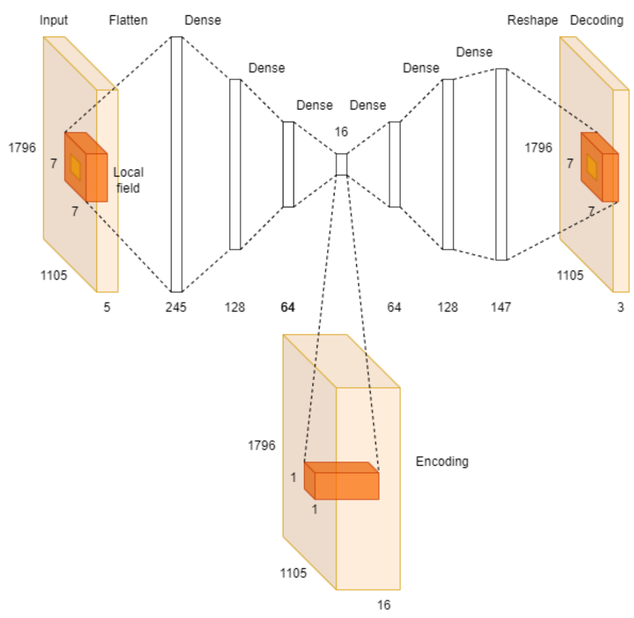

Figure 3: Autoencoder for finding latent representations of individual pixels. A sliding window of size 7×7 is considered the input to the network. At each epoch, a random sample of pixels is chosen as the central pixels of the windows. The optical flow velocities and positions of the pixels are concatenated to the image to form windows of size $7 \times 7 \times 5$. The reconstruction is of size $7 \times 7 \times 3$, with position discarded. With all information discarded except the central pixel, the reconstruction and encoding of each pixel is embedded into a 2D grid of the same shape as the input.

Autoencoders [3] are neural networks that are designed to generate a low-dimensional representation of input data and reconstruct the input data from the encoding. An autoencoder is trained to produce efficient representations that capture the key features of the dataset so that the reconstructions are as close to the input data as possible. In this project, we propose a method to encode pixels locally (see Figure 3).

3) Clustering and image segmentation

KMeans [4], a popular data clustering algorithm, is used to cluster the encodings of each pixel. The labels are assigned to each pixel position and are embedded in a 2D grid, representing a segmentation of the image. These clusters are based on significant features of population count and optical flow velocities, however, they are not necessarily connected sets. Therefore we use the **DBSCAN** [5] algorithm on pixel positions to partition the clusters further. We choose DBSCAN over other clustering algorithms because it is suitable for high-complexity non-convex partitions.

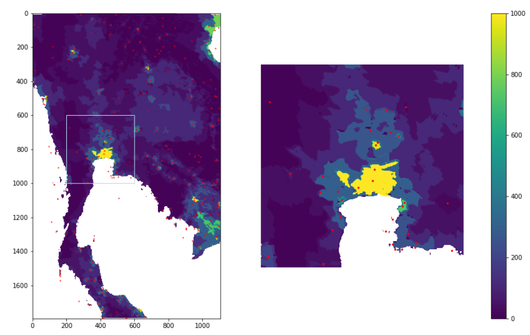

Figure 4: (Left) Population count averaged over each cluster in the partitioning formed by clustering of encodings by KMeans and clustering of pixel positions by DBSCAN. (Right) Zoomed in over Bangkok. Red dots represent mean positions of clusters.

3) Tracking by cosine similarity

The tracking of the image segmentation across frames is achieved by comparing each cluster pair in consecutive frames and finding the best fit for cosine similarity of optical flow vectors. We define inter-layer connections by the relation $\alpha \sim \beta \iff (\alpha \rightsquigarrow \beta) \vee (\alpha \leftsquigarrow \beta)$ where

$$\alpha \rightsquigarrow \beta \iff \alpha = \max_{c \in C^{(1)}} \left[\sum_{i \in \beta \cap c} \frac{\left\langle v_i^{(1)}, w_i^{(2)} \right\rangle}{\|v_i^{(1)}\| \|w_i^{(2)}\|} \leq |\beta \cap c| \right],$$

where $\beta \in C^{(2)}$, $C^{(k)}$ is the partitioning of frame k, and $v_i^{(1)}$ are the *forwards* optical flow vectors for frame 1 and $w_i^{(2)}$ are the *backwards* optical flow vectors for frame 2. The backwards relation $\alpha \leftsquigarrow \beta$ is defined similarly, so that for each α we find a β by a maximization task. The relations $\rightsquigarrow$ and $\leftsquigarrow$ account for cases where clusters in frame 1 either grow, shrink, merge or split into clusters in frame 2.

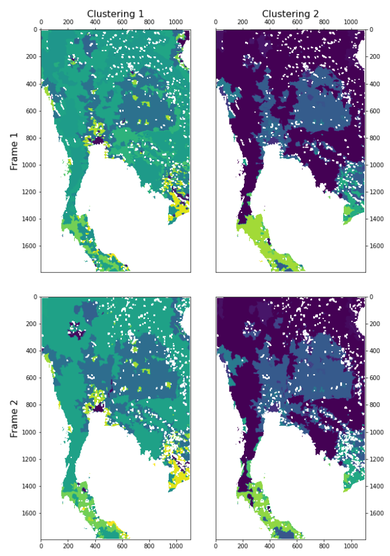

Figure 5: Tracking of cluster assignments from frame 1 (2000) to frame 2 (2005). The diagonal plots represent the cluster assignments for each frame in a colour scheme that matches the order of assignment, which is not associated with anything physical. The off-diagonal plots show each frame's clustering shape in the colour scheme of the other frame's clustering. Frame 1 Clustering 2 ($\leftsquigarrow$) is in the shape of frame 1's clustering and in the colour scheme of frame 2's clustering.

References and Acknowledgements

(1) *GPWv4: Population Count*, DOI: 10.7927/H4JW8BX5 (accessed 25 July 2022), Palisades, NY: NASA SEDAC, 2018.

(2) B. K. Horn and B. G. Schunck, *Artif. Intell.*, 1981, **17**, 185–203.

(3) Y. Lecun, *PhD thesis: Modeles connexionnistes de l'apprentissage (connectionist learning models)*, Universite P. et M. Curie (Paris 6), 1987.

(4) S. P. Lloyd, *Inf. Theory*, 1982, **28.2**, 129–137.

(5) M. Ester, H.-P. Kriegel et al., Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, Portland, Oregon, 1996, pp. 226–231.

*Research supervised by Radu Cimpeanu and supported by Undergraduate Research Support Scheme.