Uncertainty-aware Interpretable Deep Learning for Slum Mapping and Monitoring: Supplementary Material

THOMAS FISHER, HARRY GIBSON, YUNZHE LIU, MOLOUD ABDAR, MARIUS POSA, GHOLAMREZA SALIMI-KHORSHIDI, ABDELAALI HASSAINE, YUTONG CAI , KAZEM RAHIMI, AND MOHAMMAD MAMOUEI

A. Dataset Creation

A.1. Creation of spatial data from published maps

In cases where we have a published map of some form showing slums, but not the actual underlying spatial data (i.e. we have a "picture" of the data rather than the data themselves), we may need to reconstruct the spatial data by georeferencing the view of the data that is in the published version. This was the case with all of the images produced from the PKDas Associates Mumbai Slums map [1]. The map of Mumbai slums was obtained in PDF format from the PKDAS associates website. To reconstruct a spatial dataset of the mapped slums for use in training data creation, we first converted the PDF to a GeoTIFF image by rasterising at 600dpi resolution using the gdal_translate utility and then georeferenced the resulting image using the QGIS Georeferencer tool with reference to Google Satellite imagery.

We then converted this colour image to a binary slum/not-slum mask by identifying an approximate RGB value of the slums, which are shown in a reddish colour, and selecting all pixels with RGB values within +-25 of this on each channel. (This threshold was empirically determined to give the most complete selection without excessive additional areas being included.) Next, to clean up gaps in the thresholded image caused by gridlines, text etc on the published map, we conducted a morphological closing operation on the initial binary image. We used the ESRI ArcScan extension with a pixel tolerance of 4 pixels; again this threshold was empirically determined to result in the closure of most erroneous gaps in the selected polygons without closing true gaps due to roads etc. Finally, the cleaned binary image was vectorised to polygons using standard GIS functionality.

A.2. GIS processing of slum maps

We have decided to model the slum-mapping problem as an image segmentation task, whereby each pixel in an image is allocated to a single class from a choice of 2 or more options. As a minimum we therefore need to train the model to recognise two classes; slum and not-slum, and thus need to prepare training data with both these classes labelled. Production of these training data varied depending on whether or not the available map of slums in an area could be deemed to be complete, as follows:

In Mumbai, the map of slum areas was deemed to be essentially complete; that is we assume that all true slum pixels are labelled as slum. The selection of not-slum areas for incorporation into label data was therefore automated as follows. First, all slum polygons were buffered using a distance of 250m (this threshold was empirically determined based on an approximation of the average size and distance between the slum polygons). The buffers were dissolved to remove overlaps and to ensure that no slum polygon was covered by the buffer of another. Next, to ensure that areas not-adjacent to slums were also represented, we created 1000 points at random falling within the bounding box of the slum polygon dataset, but not within the slum polygons themselves. Each of these points was also buffered by 250m and dissolved with all the other buffers as before. Taken together all of these buffer polygons gave the areas to be labelled as not-slum in the training data.

In cities where the slum data are not thought to be complete, this automated approach for choosing areas to label as not-slum could not be used, and a more manual approach was required: Areas which were mapped as slum were labelled as such in the training data, whereas not-slum

areas were in this case selected by manual digitisation of areas which could be readily identified from aerial imagery as not being slum developments.

However we were severely restricted in which areas we could confidently identify as not-slum: open spaces, water, etc could of course be labelled as not-slum, but very few built-up areas could with confidence be labelled as not-slum in the absence of any further information. It is important that we do not erroneously train the model with pixels that are really slum but are labelled as not-slum, or vice versa, and so in any cities for which the slum mapping is not known to be complete, this poses a severe limitation on the usefulness of the models we can produce. From this point of view, a map of slums that is not known to be complete is unfortunately of little use as training data, because we simply have no other way of systematically creating not-slum training samples where we are confident that they truly are not-slum. In either case, the net result of the process is a polygon dataset with an attribute flagging the slum/not-slum status of the polygon. Training samples in the form of "image chips" were then created to cover all these polygons.

A.3. Production of training samples

The training data creation process was based on the production of a map of known slum and not-slum areas in the form of a vector GIS dataset. The area covered by these polygons was then divided into fixed-size tiles (with a degree of overlap) and for each tile intersecting any polygon two images or "chips" were created, one containing the imagery data for that tile and the other containing a rasterised version of the label polygons at the same resolution as the imagery. The tiling scheme we used was the Descartes Labs (https://descarteslabs.com/) DLTile scheme, which for any unique combination of tile size (in pixels), resolution (in metres), and overlap (in pixels), divides the world into uniquely-identified square tiles each in a locally-appropriate projected (UTM) coordinate system. The Descartes Labs data refinery service then allows for the retrieval of imagery mosaics clipped and aligned to any such tile. Python code was developed to identify the required DLTiles, retrieve the imagery from the Descartes Labs API, and to rasterise the labels data for the matching area. In each dataset, training samples were created for each DLTile intersecting a slum and/or not-slum polygon in the training data. The label polygons were rasterised to a unique value for either slum or not-slum.

REFERENCES

1. PKDas, http://www.pkdas.com/maps/3-Mumbai's-Slums-Map.pdf (2011).