Citizen Visual Search Engine: Detection and Curation of Urban Objects

Immanuel Koh(✉) and Jeffrey Huang

École polytechnique fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland
{Immanuel.koh, jeffrey.huang}@epfl.ch

Abstract. Increasingly, the ubiquity of satellite imagery has made the data analysis and machine learning of large geographical datasets one of the building blocks of visuospatial intelligence. It is the key to discover current (and predict future) cultural, social, financial and political realities. How can we, as designers and researchers, empower citizens to understand and participate in the design of our cities amid this technological shift? As an initial step towards this broader ambition, a series of creative web applications, in the form of visual search engines, has been developed and implemented to data mine large datasets. Using open sourced deep learning and computer vision libraries, these applications facilitate the searching, detecting and curating of urban objects. In turn, the paper proposes and formulates a framework to design truly citizen-centric creative visual search engines – a contribution to citizen science and citizen journalism in spatial terms.

Keywords: Deep learning · Computer vision · Satellite imagery · Citizen science · Artificial intelligence

1 Introduction

The collection and analysis of satellite imageries have been increasingly used by organizations or companies as a means to infer new spatial and temporal insights at the local and global scales. What was once only accessible to the military intelligence is now made almost ubiquitous. On the one hand, there are the new satellite players who send private micro-satellites into space to capture the Earth's images; on the other hand, there are the new startups using machine learning to gather actionable intelligence from these images to predict market movements. In near real-time, cultural, social, financial and political insights can be gained by simply harnessing the visual data of the Earth's conditions from above. In addition, we are also witnessing a surge in the resolution of satellite imagery – from the 60 m/pixel in the 1970s, to 30 m/pixel in the 1980s, to 20 m/pixel in the 1990s, to 2.5 m/pixel in 2000s and now as much as 0.5 m/pixel [1]. Yet, the public is denied access to very high-quality satellite imagery often controlled by commercial entities and political organizations. We argue that this denial is not only technological and monetary, but fundamentally political. It is a loss of the citizen's power to detect, search and make sense (curate) of the world. It is what Eyal Weizman refers to as *the threshold of detectability*, a condition between what is and is not detectable [2]. In the past, this detectability is dependent on the grain size of a single

J.-H. Lee (Ed.): CAAD Futures 2019, CCIS 1028, pp. 168–182, 2019.
https://doi.org/10.1007/978-981-13-8410-3_13

silver salt particle on analogue film and now, in our case, the pixel size of today's digital satellite imagery.

How can we, as designers and researchers, empower citizens to understand and participate in the design of our cities amid this new technological shift? As an initial step towards this broader ambition, a series of creative web applications, in the form of visual search engines, has been developed and implemented to data mine large datasets using open sourced deep learning libraries to facilitate in the detection, recognition, tracking and classification of a selection of visible urban artifacts. These applications are in fact the computational apparatus for our envisioned citizen science and citizen journalism. By gathering insights from the curated urban data and presenting them as playful interactions, citizens will be empowered to explore and reflect on their current territorial conditions and implications. A total of 6 different projects will be discussed in varying detail to highlight the opportunities found and strategies used. All projects used Switzerland as their site, with the exception of one in Africa and one in Colombia. All projects are also full stack web developments, implementing different deep neural network architectures and image processing algorithms, on both raster and vector tiles as their raw datasets. The collective contribution of this design research is the prototyping of creative visual applications to search for a specific territorial reality and framing its potential urban meaning and values, while democratizing GIS for all.

2 Definitions

Urban Objects. What are urban objects? In the Sydney Urban Objects Dataset, there are altogether 26 classes of objects 3D scanned (LIDAR), ranging from umbrella, bench, tree to bus and buildings [3]. In the SpaceNet dataset on Amazon Web Services (AWS) [4], probably the largest dataset of this sort, a corpus of prelabelled satellite imagery of up to 0.3 m/pixel resolution featuring selected cities, such as Paris, Shanghai and Rio de Janeiro, has recently been made publicly available for the machine learning research community. In our paper, we define an urban object as any entity that is visible on a satellite image. Our concept of urban objects is closely related to Google's Data Center Mural project by artist Odell [5]. This 2016 project featured the ways in which Odell manually cropped objects found on Google Maps and organized them in an artistic fashion according to their types, shapes, colors and other visual criteria. Her artworks from this series are predominantly an aesthetical endeavor and suggest to us that when urban objects are removed from their contexts, a comparative analysis and an alternative reading of their meanings could emerge in surprising ways. Another strand of our motivation comes from the work of Forensic Architecture [2] whose use of satellite maps and crowdsourced datasets provide the basis to critically interrogate society's political, economic and social issues in visual and temporal terms.

3 Detection with Deep Learning

There are 4 different projects in this section that have used deep neural network architecture to implement their creative visual search engine. Each project discussion is organized in the following order – objective, data, model and engine, to better facilitate a common reading structure. Inspired by Google Map route planning capabilities, the first project ('Roads') uses a publicly available dataset of Africa that contains a higher resolution of satellite imagery to develop an app that will plot the list of safest routes from one location to the other, according to the prediction of a deep neural network trained with pre-labelled data of 'non-road', 'paved-road' and 'dirt-road'. The second project ('Photovoltaics') trained a neural network to recognize and detect every solar panel found on any building in the entirety of Switzerland. As a timely example of citizen science and journalism, it then uses this as a proxy to measure and visualize the solar energy production in each municipality and canton during the May 2017 Swiss Energy Voting. The third project ('Roofs') uses deep learning for ethnographic purposes, as well as to experiment with the process of human-machine collaboration, specifically in the task of data labelling and object classification. This application detects the indigenous Colombian houses in the tropical forest of La Sierra Nevada, and in the process provides users the opportunity to both visually discover and understand its specific spatial and social configurations, while verifying the accuracy of the machine's own predictions. The last project ('Trees'), concerned with the effects of deforestation, trained a convolutional neural network to classify and calculate tree coverage from satellite imagery, according to their pixel spatial density and sparsity. This interactive app not only allows users to pick, compare and visualize any city or canton, but also see their ecological ranking in Switzerland.

3.1 Urban Object: *Roads*

Objective. The urban object of concern in this project is the *road*, more specifically the quality of roads in the African city of Ouagadougou in Burkina Faso – paved road versus dirt road. According to the World Economic Forum and World Health Organization [6], 40 of the 50 countries with the highest road traffic deaths are all African. Compared to Europe, Africa has 10 times fewer cars and yet 3 times higher road traffic deaths. The prototype is akin to a Google Map interface, except that the recommended routes are ranked based on inferred safety, rather than duration of travel. In addition, the generated data from our deep learning model could serve as a proxy to measure economic development of African city neighborhoods, thus creating awareness to both citizens and other stakeholders.

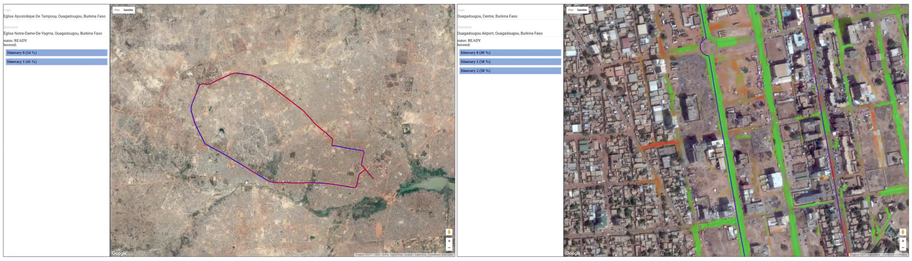
Data. The pre-labelled dataset is available at Kaggle [7] consisting of 1 km $\times$ 1 km satellite images in 3-band and 16-band formats. For our purposes, we only utilize the 3-band format and the 2 class types – *poor_dirt_cart_track* and *good_roads*. To ensure a balanced training set of our 3 classes ('non-road', 'good road', 'bad road'), tiles without roads are removed from our training set. To improve the training process, we have also applied normalization to the images by removing the mean and dividing them by the standard deviation. For our supervised learning process, each RGB satellite image tile is labelled according to the 3 masks of each class. The class labels are originally in the vector format of the given GeoJSON file but are converted as raster format corresponding to the image pixels distribution. Each mask is a black and white image with the same number of pixels as the tile. Each pixel is white when it is associated with the given category.

Model. After several trials, the U-Net with two phases of upscaling and downscaling is chosen as the most appropriate convolutional neural network architecture. Our U-Net consists of 4 blocks in each phase. Each block in turn consists of 2 convolutions (kernel 3×3) with a RELU activation function and a bias. For the downscaling (1st phase), each block ends with a max pooling layer; while for the upscaling (2nd phase), each block begins with a upsampling layer. Finally, the model ends with a 2-dimensional convolutional layer using a sigmoid activation function, the binary cross entropy as loss function and Adam as the optimizer. This last layer is to predict the final classification of each pixel according to the combined 3 classes of binary masks ('non-road', 'good road', 'bad road') given a RGB satellite imagery input. Thus, it is similar to an image segmentation classification problem. We then generate an infinite stream of training data for our neural network model with the following data augmentation procedure:

- Picking a random 4-uples (satellite image and the 3 masks)
- Select a random rotational angle and position displacement
- Crop the 4-uples as 160×160 images according to the new rotation and displacement
- Check if at least 1 pixel corresponds to a road label and return the 4 crops. Otherwise, restart the procedure with a new randomly selected 4-uples.

The training lasted 10 days on a NVIDIA GTX Titan (GK110) with 3000 epochs of 212 batches (24 tiles measuring 160×160 pixels each). The inference process proceeds by first reading a given satellite image, cutting it into 160×160 pixels tiles (with overlaps and normalization), reconstructing as 512×512 pixel tile, merging all three classes with 'non-road' pixel as transparent channel, 'good road' pixel as green channel and 'bad road' pixel as red channel.

Engine. (Figure 1) The user interface (React application) consists of a simple web map visualization and a sidebar allowing the user to type in the desired origin and destination with the additional autocomplete functionality from Google Maps Places API. Multiple itineraries are then generated with the same API and serialized to JSON and sent to the 1st server (Node.js). Upon receiving the routes, the server determines the tiles traversed by these routes and trigger a 2nd server to classify the pixels in the tiles. The 1st server then receives the inferred tiles for reading and extracting all the pixels along the itineraries before computing a score that represents the safety measure (blue for good and red for bad parts of the trajectories). The inferred tiles are also displayed as an overlay on the web map and red-blue trajectories.


Origin
Eglise Apostolique De Tampouy, Ouagadougou, Burkina Faso
Destination
Église Notre-Dame-De-Yagma, Ouagadougou, Burkina Faso
status: READY
hovered:
Itinerary 0 (54 %)
Itinerary 1 (41 %)

Origin
Ouagadougou, Centre, Burkina Faso
Destination
Ouagadougou Airport, Ouagadougou, Burkina Faso
status: READY
hovered:
Itinerary 0 (49 %)
Itinerary 1 (58 %)
Itinerary 2 (58 %)

Fig. 1. (LEFT) Screenshot of the 'ROADS' project interface. The side menu lists the user's origin and destination in Ouagadougou, followed by the sorted generated itineraries according to the computed safety measures in percentages. The web map visualizes the generated itineraries (red = unsafe route, blue = safe route). (RIGHT) Another screenshot with zoom-in details. The web map shows the satellite image tiles being overlaid with pixels mask (red = bad road, green = good road) inferred by our deep learning model. (Color figure online)

3.2 Urban Object: *Photovoltaics*

Objective. The urban object to be searched in this project is the *photovoltaic panels* installed on the roofs of buildings in the entirety of Switzerland. Related works, such as the recent Google's Project Sunroof [8] and the Swiss Sonnen Dach [9] projects, have demonstrated the use of satellite imagery in predicting the amount of sunlight a given building receives on its roof and facade per year, in order to then calculate the optimal solar plan for economical energy production and savings. However, our project is closer to the 2013 Big Atlas of Los Angeles Pools [10], where the focus is about the searching and classifying of urban objects. Instead of mapping LA pools with crowdsourcing mechanism (e.g. Amazon Mechanical Turk services) and traditional computer vision algorithms, our project maps and classifies all the buildings in Switzerland based on the presence or absence of solar panels using deep neural networks and web scraped datasets. Apart from the non-existent tag category of 'solar panels' on OpenStreetMap, the varying shapes and colors of solar panels makes using a standard image processing algorithm ineffective for our project. More philosophically speaking, the project is concerned with a citizen science approach of mapping and measuring urban objects, thus making visible their correlations with the invisible forces of economy, politics and ecology. For instance, in the case of the 2016 Swiss vote for/against "Exiting Nuclear Energy", our project could be used to predict the result based on the distribution of solar panels in each Swiss canton.

Data. There exists no pre-labelled training dataset for our classification problem, thus necessitating both the collection of our own datasets and the creation of their corresponding labels. Using the Overpass Turbo API, we extract the WGS84 coordinates of every building footprint polygon in the city of Lausanne from OpenStreetMap formatted as GeoJSON. We then download every satellite tile at zoom level 20 (the maximum allowable resolution via the Google Maps Static API), with its center according to our computed barycenter of each building polygon. The square bounding box of the intersection between the building polygon and the satellite tile (640 $\times$ 640) is used to crop and resize the final image to 224 $\times$ 224 pixels, which is the default input size of our chosen deep learning model. Since this dataset is still unlabeled, we created a minimalist web interface showing a pre-labelled satellite image and 3 different buttons (‘yes’, ‘no’, ‘redo’) for the user to verify if a solar panel is visible. With our manual labelling interface, we manage to label 15000 satellite images by simply clicking the corresponding buttons at a rate of 5000 images per hour. However, only 1000 out of these 15000 images actually contain solar panels and we end up using a balanced training set of 900 positive and 900 negative images, and another 100 positive and 100 negative images for the validation set of our neural network. In addition to the training set, we need to download the remaining satellite tiles for the rest of Switzerland to be used for our prediction. The same workflow is used, except with 4 API keys to overcome the API daily limits and with 50 downloads in parallel, thus achieving a rate of downloading 100,000 images per day in approximately 1 h. The cropping process takes 2 h while the classification rate is 12000 images per hour. Due to Overpass Turbo API’s internal memory limit of 2 GB when processing queries, the queries are done canton by canton and even split into successive queries for big cantons.

Model. We use a transfer learning approach with an existing pre-trained convolutional neural network called ResNet50 for our classification problem. The advantage of using such an approach is the capability to train a model with little data, in our case, a labelled dataset of 2000 images. Although more complex than the VGG-16 architecture due to its residual architecture and 50 layers of convolutions, the ResNet50 is much lighter and faster to train. We first remove the last fully connected layer (FC) and do a forward pass through this network to obtain the output features and save them as.npy files. We then train our own 1-layer fully connected neural network with a binary output by loading the previously saved.npy files and achieve an accuracy of 85%. For the final prediction model, we replace the last fully connected layer (FC) of the original ResNet50 model with our newly trained small neural network.

Engine. (Figure 2) In total, 1.5 million buildings have been classified with the initial 500 GB of downloaded satellite images. Each canton is represented with a single JSON file containing the number of buildings with and without solar panels, as well as all the building coordinates. An interactive map (rendered as SVG) is made with D3js for the data visualization. This interactive exploration can be experienced at the level of country, canton, municipality and individual building. All the regions are shaded and mapped on a color scale, representing their respective percentages of buildings with solar panels, such that users could compare their statistical differences at any zoom level. Additionally, the interface provides the mechanism for users to verify the presence or absence of the solar panels when viewing the satellite imagery, thus crowdsourcing from the users to improve the accuracy and revealing the potential biasness of our trained model during their online interactions. Due to the huge number of SVG objects, to facilitate a smooth and fluid interaction, the web client only loads on-demand. In other words, without loading all the files in the browser, the SVG elements from previous canton are removed when clicking on another canton during the interactive session.

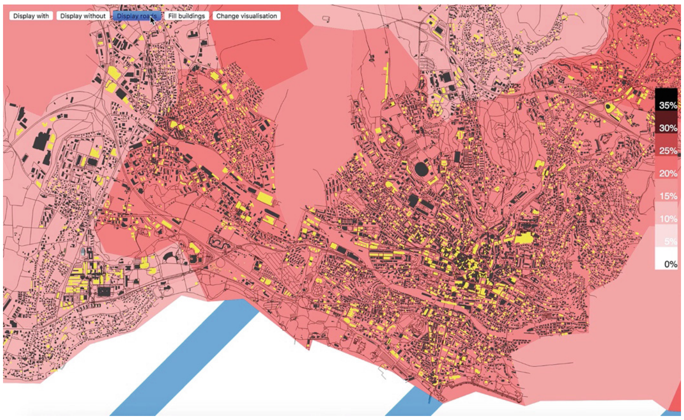


Fig. 2. Screenshot of the 'PHOTOVOLTAICS' project interface. A web map application showing the distribution of buildings with solar panels (in yellow) and those without (in black), based on the prediction of a trained deep learning neural network. The different Swiss municipalities are shaded based on a color scale (from light red to dark red) that maps the percentages of buildings with solar panels within their respective regions. (Color figure online)

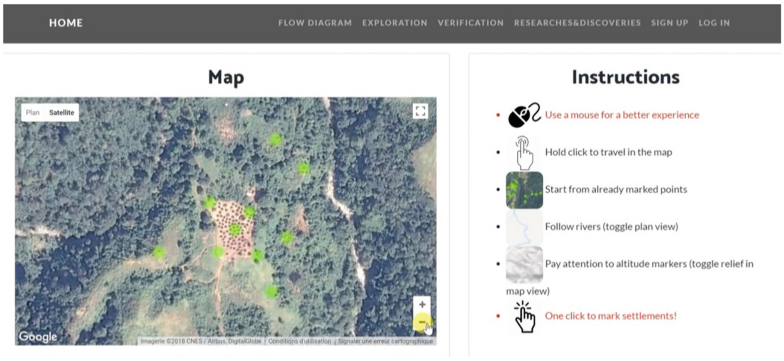
3.3 Urban Object: *Roofs*

Objective. The urban object to be searched in this project is the *roof* itself, more specifically the roofs of the Colombian indigenous house distributed among the tropical forests of La Sierra Nevada that are visible from satellite imagery. Recent online projects like GlobalXplorer [11], which allow citizen explorers to identify culturally important ruins with high-resolution satellite images, serves as our initial inspiration. The additional features of our project are that the user will have the choice to do some guided exploration, as well as verify data labeled by our algorithm. More precisely, the focus is on images containing (or not) Colombian indigenous houses. This way, the user will participate in improving an algorithm that could potentially detect global settlements from different tropical forests.

Data. We use the Google Map API to fetch 1700 satellite tiles as the training set (1500 as positive and 200 as negative labels), and another 360 tiles as the validation set (100 as positive and 260 as negative labels). Each of the satellite tiles is 224 × 224 in size and is cropped to remove any unnecessary overlaid annotations by Google Maps. The tile colors are also normalized before feeding them to our deep learning model – using transfer learning with a pre-trained convolutional neural network (CNN).

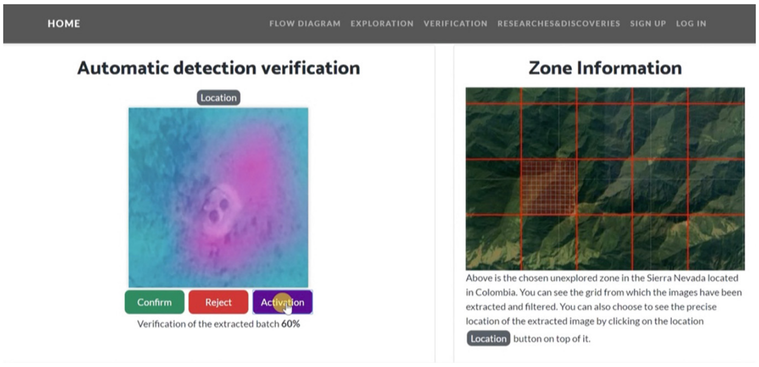
Model. The VGG-16 that we are using for our transfer learning has already been pre-trained with the ImageNet dataset, containing millions of images, to classify 1000 classes of objects. The VGG-16 model consists of 5 blocks of convolutional layers and 1 block of fully-connected layers. We have removed this top block of 3 fully-connected layers, as well as, the last layer (max-pooling layer) of the last convolutional block. Such that we could use their output features as inputs for our smaller fully convolutional network (FCN) of 4 layers to predict if an image contains (or not) any indigenous settlement. Our trained model has an accuracy of close to 98% (validation set). The advantage of using FCN is the ability to visualize the regions of the image that have activated the last convolutional layer in the form of a heatmap, thus serving our purpose in visually understanding how the neural network has perceived the given satellite imagery.

Engine. (Figures 3 and 4) In addition to crowdsourcing the locations of potential indigenous settlements, this project also provides a means for users to confront the predictive accuracy of our deep learning model. Here, users are able to see what our 'black box' machine is 'seeing' (activations) and are empowered to make their own data verification as a community. Technically, it also helps to improve our deep learning model with human curated data.



- Use a mouse for a better experience
- Hold click to travel in the map
- Start from already marked points
- Follow rivers (toggle plan view)
- Pay attention to altitude markers (toggle relief in map view)
- One click to mark settlements!

Fig. 3. Screenshot of the 'ROOFS' project interface. A web map application guiding the user in exploring and detecting the presence of Colombian indigenous house settlements distributed among the tropical forests of La Sierra Nevada.



Verification of the extracted batch **60%**

Above is the chosen unexplored zone in the Sierra Nevada located in Colombia. You can see the grid from which the images have been extracted and filtered. You can also choose to see the precise location of the extracted image by clicking on the location Location button on top of it.

Fig. 4. Another screenshot of the 'ROOFS' project interface. On the left panel, users are empowered to verify the accuracy of our deep learning model and at the same time understand how the model sees via the visualization of its neural networks' activations. The panel on the right displays the unexplored areas, while providing users with the geographical context in which the satellite image is taken from.

3.4 Urban Object: *Trees*

Objective. The urban object to be computed in this project is the *tree* itself. By conceptualizing trees as pixels found on satellite imagery, a measure of their spatial density and sparsity per satellite tile (fixed at a specified zoom level) can be framed as a measure of deforestation. The curation is comparatively visualized: displaying 2 regions, side-by-side, at a selected hierarchical level of the canton and the municipality in Switzerland.

Data. In order to have a realistic representation of trees as pixels, we have opted for a zoom level of 16, which gives a resolution of 2.387 m/pixel with a tile size of 256 × 256, or 512 × 512 if we upscale it with MapBox's added functionality. For a supervised learning approach like ours, the only available source of training labels is OpenStreetMap (OSM) via the Overpass Turbo API, such as querying for the tag *"natural = tree"* which is an OSM node that represents a single tree or *"landuse = forest*" which represents wooded plantations. The tree density could then be calculated and each tile labelled. However, it soon became clear that the labels are very incomplete and thus no longer an option for our purposes.

Model. 3 different Deep Neural Networks architectures were initially considered – the Convolutional Neural Networks (CNN) where the input is an image tile and the output is a number representing tree density; the Fully Convolutional Networks (FCN); and the U-Nets where the input is also an image tile and the output is a categorically segmented version of the same image. However, none of these architectures which each requires a huge number of pre-labelled training dataset was eventually found appropriate and a novel approach of Per-Pixel classification was adopted instead. Our approach only requires 15 manually labelled satellite tiles which translates to almost 4 million inputs (15 × 512 × 512). After balancing the ratio of tree/non-tree pixels in this dataset, 768622 inputs were used for the training model. Two different machine learning models were tested and both yield good results with around 92% accuracies. The first model is an AdaBoost with 512 decision trees as base classifiers, while the other is a simple fully connected Neural Network with 3 hidden layers of 256 nodes each. The input feature vectors of length 10 is used which consists of the pixel colors in CIELAB color space, entropy values and illumination invariant values.

Engine. (Figure 5) After the tile classification, both the density and sparsity of the trees are calculated. Several other measures based on the statistics fetched from the Federal Statistical Office of Switzerland are used as well, namely cars per inhabitant, relative area and relative population. Together, these 5 features with their corresponding weights of 160, 240, 30, 2, 1 form the basis for the curatorial ranking of the most/least air polluted cities and cantons in Switzerland. Since this is a prototype, only all the tiles from the cantons of Vaud and Geneva are classified, which already amounts to 20 GB of 5 million records. The size could have been in the billions if each pixel and its coordinate is saved. Instead, what we have done here is to save them as tile of 32 × 32 pixels each. We are aware that our Per-Pixel approach is probably contrary to the region-based visual perception of objects by humans. Thus, 2 image processing filters (erosion and dilation) are used on the predicted outputs to take into consideration the neighboring pixels while removing any potential noise. The final interface allows a side-by-side visualization, measurement and comparison of different spaces in Switzerland at any scale, from individual cities to cantons and even arbitrarily user drawn areas. Their respective rankings and scores are also indicated according to our custom-made scale index of the 5 weighted features.

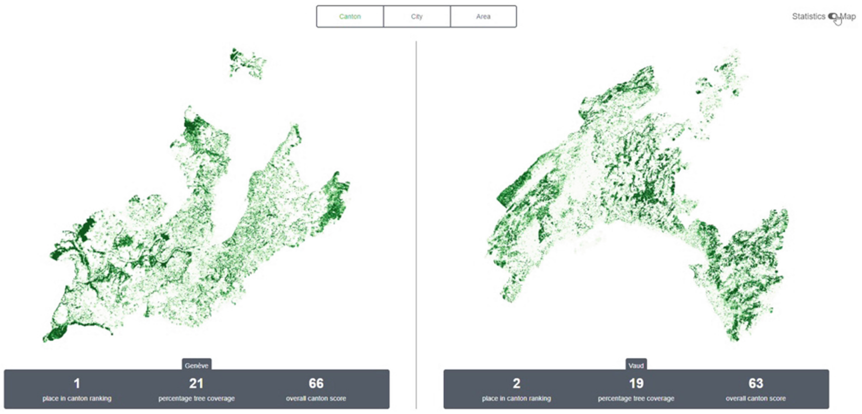


Fig. 5. Screenshot of the ‘TREES’ project interface. A web application showing side-by-side 2 different cantons (Geneva vs Vaud) in Switzerland with their respective statistics and predicted tree coverage.

4 Curation with Computer Vision

Even though the use of deep neural networks has proved to be highly effective in the previously discussed projects, the following projects take on a non-deep learning approach in the implementation of their curatorial applications. Instead, computer vision and image processing algorithms are used which allow more explicit feature selection and finer-tuned differentiation beyond a binary one. Here, there is also a greater emphasis on alternative user interaction in curating the visual data. One project (‘Pitches’) visually ranks artifacts from satellite imagery by looking at ways to grade the quality of football pitches and tennis courts found in Switzerland. Instead of using deep learning, OpenCV (an open source computer vision library) is used for template matching, to detect and score key parts of the pitches, such as the end lines, sidelines and goal lines. This application thus not only provides a means for the respective city council to ensure the maintenance and quality of their sport infrastructure, but also to potentially notify OpenStreetMap (OSM) of any missing or mislabeled vector tiles in their databases. For the pleasure in playful curation, one project (‘Lakes’) rethinks the process of visual search as one of simply sketching a line on screen. With this simple gestural input, the application will search for lakes in Switzerland with the best matching geographical boundaries and rank them accordingly. Again, each project discussion is organized in the following order – objective, data, algorithm and engine, to better facilitate a common reading structure.

4.1 Urban Object: *Pitches*

Objective. The urban object to be curated in this project is the *pitch*, more specifically the quality of football pitches and tennis courts found in the entire Switzerland. It is a curatorial engine where users can search and explore pitches according to their qualitative score – in our case, the measured clarity of the field line markings as seen on the satellite imagery. In doing so, the application also indirectly illustrates the economic well-being and social infrastructural investment of different neighborhoods, cities and cantons in Switzerland.

Data. During the initial dataset investigation, we found that the existing OSM vector dataset shows a considerable amount of discrepancies, namely inaccurate outlining and incorrect tagging of fields. Further data irregularity is also found in the Google Map raster dataset, mainly occlusions of the fields by other urban objects, such as humans, shadows from nearby buildings and trees, as well as other conjoining sport fields. Fortunately, unlike other urban objects discussed thus far, the dimensions of these sport field urban objects are generally standardized – football (1110 by 970, ratio of 0.87) and tennis (530 by 280, ratio of 0.53). The data preprocessing consists of the extraction of field location coordinates, extraction of images and their analysis with computer vision algorithms. First, we use the simple OSM query by specifying the relation type as “facility” and the subtypes as “tennis”, “football” or “soccer”, to obtain the necessary GeoJSON files describing the corner coordinates of all the fields. Next, for each field polygon, we fetch the raster satellite image with Google Maps API and fit their corresponding corners. A duplicate of this image is made but overlaid with a fill color, thus acting like a mask, such that a bitwise operation could be applied on both images to crop the field. Since all the fields have different orientations, we extract their median axis to reorient them vertically for our subsequent image analysis.

Algorithm. Various computer vision algorithms have been tested to extract the field line markings in order to measure their quality. These include applying Canny edge detector with Hough transform on either grayscale or HSV images and Viola-Jones algorithm with Adaboost. The former’s result is not satisfying, while the latter requires more and cleaner data. The final solution is to use a template-matching approach by searching and scoring the field according to a set of key patterns representing various field line markings. More concretely, there are 5 different templates (‘full’, ‘frame’, ‘center,’ ‘top’ and ‘bottom’), each varyingly scaled and consists of thick white outlines. The higher the confidence score implies the more vivid the line markings are and therefore the higher the quality of the field is. Lastly, for each field, a vector is computed that represents all the 5 confidence scores, as well as, another hue color distance score. Using L1 distance function between each pair of fields, we can compute their similarity score and save them in a database for our curatorial engine.

Engine. (Figure 6) The interactive web app is made with the React framework which provides easy implementation of interactivity, as well as, easy communication among the client, the server and the mongo database. The various components of the user interaction include the map navigation, search curation, field inspection, similar fields display and scoring weights adjustment. The user could begin by choosing the type of pitch (either tennis or football), setting the score interval (desired quality of the pitch) and the range of distances to search other similar pitches. Once a pitch (polygon) is clicked on the web map, its corresponding satellite image is displayed on the right panel, alongside its matched templates as overlays. These scoring templates are displayed as color-coded frames with different thicknesses according to their respective numerical scores along the indicative sliders. At the same time, the 6 most similar pitches are also displayed at the bottom panel, allowing the user to seamlessly zoom in and out between different pitches as they explore them on the web map. Lastly, the user could use the bottom right panel to reset the template weights that are used to compute the similarity score. By laying out the logic of our algorithmic curation on the interface, we are opening up creative ways of visually searching and curating urban objects.

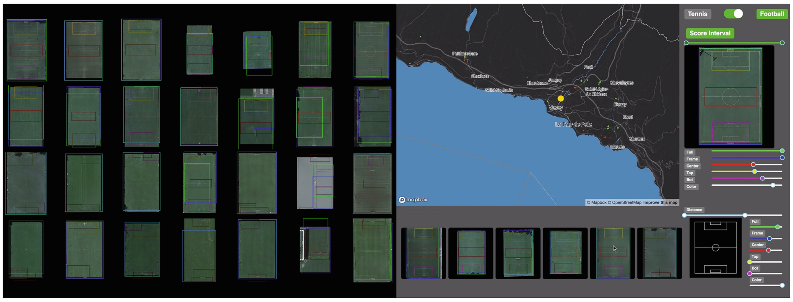


Fig. 6. (LEFT) Screenshot showing the OpenCV algorithm at work to compute the quality of football pitches from the web-crawled satellite imageries. (RIGHT) Screenshot of the 'PITCHES' project interface.

4.2 Urban Object: *Lakes*

Objective. The urban object to be curated in this project is the *lake*, more specifically the shapes and colors of lakes found in the entire Switzerland. The early inspiration for the project is Google Chrome Experiment's Land Lines [12] which is a web app running on the smartphone that allows users to explore Google Earth satellite imagery simply through the gesture of the fingers - searching by drawing lines. In our case, we sketch a shape and our engine will present us with the most similar Swiss lakes. It is a simple search and curation engine that uses visual inputs instead of words. Admittedly, this project seems to lack the degree of clarity with regard to its targeted audience and stated value, as compared to most of the other projects discussed in the paper. However, in the spirit of Land Lines [12], we foresee the advantages of such an open-ended application, in terms of their potential for aspiring future appropriation by other citizen-developers or even experts in lake morphology. For example, by knowing the lake's length-to-width ratio and its shoreline profile, one could already infer the nature of the local weather, as well as the biological diversity of the lake and its surrounding areas.

Data. Using the tags *"natural = water"* and *"water = lake"* in our OpenStreetMap query, we downloaded all the lakes' vector shapes in GeoJSON format with their names included. We then crop the corresponding raster satellite images fetched from Google Maps API within the vector shape boundary and store them in Google Cloud Storage.

Algorithm. Given all the extracted data, the task is to have an algorithm to calculate their similarity in shape and color. An image processing pipeline is adopted as follows:

- Convert each vector shape to a 50 $\times$ 50 pixels binary image
- Perform medial axis skeletonisation on the binary image
- Find principal axis of the medial axis
- Rotate the shape such that the principal axis is aligned to the x-axis
- Calculate a distance function between the pixels of every pair of lakes using L1-norm, while satisfying non-negativity, identity of indiscernible, symmetry and triangle inequality.

Once the above image processing is done, we implement a Vantage-point tree structure to organize all the lakes according to their similarity as neighbors using the value computed with the distance function. The color similarity function is eventually not implemented due to the visible color discrepancy and glitch when lakes are composed of satellite images aggregated from varying temporal and cloud coverage conditions.

Engine. (Figure 7) Implemented in the Flask framework, the application interface is done with a mix of D3js and plain javascript. The user is first presented with an empty canvas to sketch in the browser. Instead of a single line, the user draws a self-closing polygon to query lakes with a similar polygonal shape, regardless of orientation and size. The user is then presented a sorted list of 5 most similar lakes on the right panel. By hovering over any one of these lakes, an expanded view of the satellite image is displayed. By clicking on the name of the lakes shown, the application will open up a new browser window with an OSM view showing the lake's location in context.

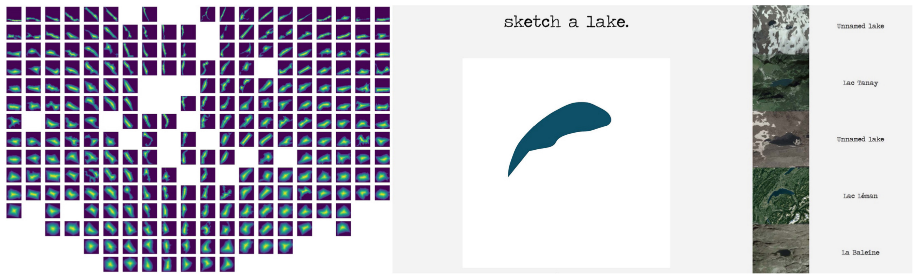


Fig. 7. (LEFT) Poster showing the curatorial organization of different lakes in Switzerland based on their computed visual features. (RIGHT) Screenshot of the 'LAKES' project interface. The user draws a shape on the left panel, while the engine computes and presents the sorted 5 most similar lakes on the right panel, regardless of their actual geographical orientations and sizes.

5 Discussion

All the design research projects discussed thus far, have directly or indirectly, contributed to a broader understanding of what might constitute a citizen visual search engine. In formulating such a conceptual framework for creating meaningful urban values, we have identified the following key characteristics:

- The visual satellite imagery data is often publicly available for download or for querying via free APIs. For example, the use of Kaggle competition dataset and Google Maps API.
- The inaccuracy and inconsistency of open source dataset, as a result of human error, needs to be recognized and considered in the design of the creative visual search engine. For example, the polygon and tag errors in OpenStreetMap and the color glitch in Google's satellite imagery.
- The common nonexistence of specific labelled dataset often necessitates the design of interfaces to rapidly crowdsource their corresponding labels. For example, the creation of minimalistic web interface to manually label satellite images by simply clicking a set of 'yes' and 'no' buttons.
- The desire of citizens to see, understand and interact with algorithmic 'black boxes' is a form of empowerment. For example, the visualization of deep learning model's neural activation layers allows user to intuitively discover and critique their potential data biasness.
- The nature of the urban object to be searched can have a direct impact on the type of computer vision algorithms and learning models used. For example, urban objects with invariant sizes or shapes or colors could use less data-intensive approaches by designing explicit high-level feature extraction mechanisms, instead of deep learning models.
- The selection and modification of deep neural network architecture is often dependent on the type of classification problem and the size of dataset available. For example, the use of transfer learning is extremely effective when only higher-level features classification is needed, thus opportunistically reusing weights already trained with lower-level visual features.
- The frontend design of the user interaction is of paramount importance and is a determining factor in the creative experience of search and curation. For example, even without having a clearly defined practical use for the project, through the process of gestural play and visual exploration, users are given the space for imagination.
- The framing of a citizen visual search engine often begins with a simple or native pairing of an urban object and a societal issue. Yet, the result often suggests multiple and unforeseen potential appropriation or extension of existing applications. For example, the simple classification of safe/unsafe satellite image pixels eventually becomes the basis to extend Google Map's itinerary generation logic.

6 Future Work

The lack of pre-labelled datasets encountered by many of the deep learning projects discussed here poses a glaring concern. Data annotations, either manually labelled by us or made available occasionally at Kaggle, is essential for any supervised learning models like ours. Our future research in creative visual search engine needs to move beyond the paradigm of supervised learning and incorporate both unsupervised deep learning models and self-supervised machine learning algorithms. In addition, instead of the single urban object type, we plan to implement applications that provide the capabilities to detect, search and curate multiple urban objects at different scales. We hope this paper will provide the first step towards the formulation and design of a truly citizen-centric creative search engine, one that is liberated from the current epistemology that is often motived politically and commercially.

References

1. Kurgan, L.: Close up at a Distance: Mapping, Technology and Politics. Zone Books, New York (2013)
2. Weizman, E.: Forensic Architecture: Violence at the Threshold of Detectability. Zone Books, New York (2017)
3. Quadros, A., Underwood, J., Douillard, B.: Sydney Urban Objects Dataset, Australian Centre for Field Robotics (2013). http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjects-Dataset.shtml
4. SpaceNet on Amazon Web Services (AWS). "Datasets." The SpaceNet Catalog. https://spacenetchallenge.github.io/datasets/datasetHomePage.html. Accessed 30 Apr 2018
5. Odell, J.: Mayes County. Google (2016). https://datacentermurals.withgoogle.com/mayes-county
6. Worse than Malaria. The Economist. https://www.economist.com/middle-east-and-africa/2015/10/24/worse-than-malaria. Accessed 24 Oct 2015
7. Dstl Satellite Imagery Feature Detection: Can you train an eye in the sky? Kaggle (2016). https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection/data
8. Project Sunroof - Solar Calculator. Google. https://www.google.com/get/sunroof. Accessed 29 Dec 2018
9. Wie Viel Strom Und Wärme Kann Mein Dach Produzieren? Bundesamt für Energie BFE, Sonnendach.ch. http://www.sonnendach.ch. Accessed 29 Dec 2018
10. Groß, B.: The Big Atlas of LA Pools, June 2013. https://benedikt-gross.de/projects/the-big-atlas-of-la-pools
11. GlobalXplorer° (2016). https://www.globalxplorer.org/
12. Lieberman, Z., Felsen, M., and Google Data Arts Team: Land Lines (2016). https://lines.chromeexperiments.com/