Holistic Multi-View Building Analysis in the Wild with Projection Pooling

Zbigniew Wojna,1,3 Krzysztof Maziarz,2 Łukasz Jocz,1 Robert Pałuba,1 Robert Kozikowski,1 Iasonas Kokkinos3

1 Tensorflight
2 Jagiellonian University
3 University College London

zbigniew@tensorflight.com, krzysztof.maziarz@microsoft.com

Abstract

We address six different classification tasks related to fine-grained building attributes: construction type, number of floors, pitch and geometry of the roof, facade material, and occupancy class. Tackling such a remote building analysis problem became possible only recently due to growing large-scale datasets of urban scenes. To this end, we introduce a new benchmarking dataset, consisting of 49426 images (top-view and street-view) of 9674 buildings. These photos are further assembled, together with the geometric metadata. The dataset showcases various real-world challenges, such as occlusions, blur, partially visible objects, and a broad spectrum of buildings. We propose a new projection pooling layer, creating a unified, top-view representation of the top-view and the side views in a high-dimensional space. It allows us to utilize the building and imagery metadata seamlessly. Introducing this layer improves classification accuracy – compared to highly tuned baseline models – indicating its suitability for building analysis.

Introduction

This work aims to develop a deeper understanding of scenes containing buildings based on aerial (top-view) and multiple side view (street-view) images. This problem is both technically challenging and of great practical importance. It allows for automatic pricing of an insurance policy, mechanical claims analysis, risk detection, understanding the environment for self-driving cars, or socioeconomic statistics extraction. The onset of new computer vision techniques and growing large-scale datasets of urban scenes naturally lead to various new scientific challenges related to holistic building understanding, much deeper than simple single-task classification. We investigate a multi-task classification problem of automated building analysis. The goal is to determine the following set of building attributes: construction type, i.e., the material the building is made of; number of floors, roof pitch, roof geometry, facade material, and occupancy type. Those features are crucial for catastrophic risk estimation (Stone 2018). The real-world unconstrained environment is often poorly recognizable based on a single image. Rather, it requires more complex analyses of multiple images taken from different angles. Our model uses both the top-view image and numerous street-view photographs of the same building to understand its fine-grained details.

In this work, we propose a new fusion technique, leveraging both the scene’s geometric structure and the high-dimensional features extracted from the top and street-view images. As a result, we obtain a single unified top-view representation of the scene, including information from side views, building outline, and imagery metadata. Based on the street-view photo location, direction, and the field of view, i.e., angle representing the visible range, we construct a projection of the street-view features onto the building walls outlined in the top-view image.

Understanding of physical structures, such as buildings, may require integrating information from all possible input sources. An example of a building scene from our dataset is presented in Figure 1. The four camera marks on the top-view image represent four different locations where the street-view images were taken. The green mark and the blue boundary on the top-view image point to the building of interest. The violet boundaries represent other visible buildings, which can be a source of confusion for the model. We can easily see how different images complement each other. The top-view image allows us to see that the roof is not flat as there is a ridge joining the two opposite sides and a skewed shadow. The roof geometry is only visible from the top-view image. The construction type, number of floors, facade material, and occupancy class can be determined only based on the street-view images.

Some decisions may require more sophisticated reasoning. A possible overlap of the features from different street-view photos might mitigate the errors caused by occlusions and inaccuracies present in a single image. In the example in Figure 1, the first street-view image points towards the center of the building but is occluded by an adjacent building and could lead to classification errors. However, when looking at all four images, it is possible to classify the building’s attributes correctly by combining several clues. The side-attached structure made of bricks may correctly suggest that all buildings’ construction type along the street is masonry. We are not able to estimate the exact roof pitch by looking exclusively at the street-view images. We can only assume that the roof is flat or has a low slope as it does not stick out from the front.
To summarize, our contributions are the following:

(1) We propose a new layer called \textit{projection pooling}, which benefits from the building scene geometry and the relationship between images from different views and perspectives. We then integrate it into a deep learning architecture. It results in a new, unified, high-dimensional representation and suits well the classification task.

(2) We develop a new deep convolutional model, fusing multiple inputs to understand building characteristics better. It includes the projection pooling layer and achieves results that are superior to highly tuned baseline models. These results indicate that one can design substantially more accurate models by incorporating information from multiple images.

(3) We build a new real-world multi-view multi-task dataset of building images, annotations, and attributes. Buildings are heterogeneous in architectural style, size, age, and come from all around the world. It sets a new benchmark of detailed building understanding to spark further research in this area.

\textbf{Related Work}

In this section, we discuss the ideas which inspired this work. These include (a) the use of street-view images, (b) building modeling, (c) the fusion of street-view and top-view imagery, and (d) building attribute classification.

(a) As street-view imagery is becoming more ubiquitous, it brings new research opportunities. One can assess socioeconomic statistics (Gebru et al. 2017a,b), evaluate the safety, beauty and popularity of the neighborhood (Anderson, Birck, and Araujo 2017; Dubey et al. 2016), estimate the road safety (Song et al. 2018), perform road scene segmentation (Cordts et al. 2016), and determine precise geolocalization of the car (Armagan et al. 2017a; Hirzer et al. 2017; Armagan et al. 2017b). Combining house rental ads and street-view imagery allows for performing 3D building reconstruction (Chu et al. 2016). In our work, we show that street-view images can also help with understanding the state of urban structures.

(b) Building facade segmentation is a well-studied problem on 2D images (Yang and Förstner 2011; Tyleček and Šára 2013; Riemenschneider et al. 2012; Teboul et al. 2011; Martinovic and Van Gool 2013; Kozinski et al. 2015; Mathias, Martinovic, and Van Gool 2016; Liu et al. 2017), through combination of 2D and 3D (Gadde et al. 2017; Riemenschneider et al. 2014) and directly using 3D data from scanners (Serna, Marcotegui, and Hernández 2016; Li et al. 2016). Many methods rely on assuming symmetry of the building facades (Cohen et al. 2017; Mitra et al. 2012; Musialski et al. 2009; Wu, Frahm, and Pollefeys 2010; Zhang et al. 2013). Facade datasets are usually collected at a specific location, consist only of a few hundred images, are homogeneous in style and well rectified. Therefore, models can fail when tested on different architectural styles (Lotte et al. 2018). Our work introduces a diverse and large-scale dataset, with buildings coming from all over the world, along with potential noise, and without any assumption about rectification, symmetry or style.

(c) A fusion of street-view and top-view imagery may further improve performance. One can fuse top-view and ground-level imagery for detailed city reconstruction (Bódis-Szomorú, Riemenschneider, and Van Gool 2016), use cross-view matching with top-view images to improve street-view geolocalization (Workman, Souvenir, and Jacobs 2015; Hu et al. 2018), or use cross-view matching with street-view to retrieve latent representation of top-view images (Workman et al. 2017; Cao et al. 2018). In (Zhai et al. 2017), the authors transform a semantic top-view scene into a semantic street-view scene. Detecting trees can be achieved through merging both of these sources (Wegner et al. 2016). Projecting the street-view latent representation onto an orthographic projection allows for 3D object detection (Roddick, Kendall, and Cipolla 2019). MVSNet (Yao et al. 2018) introduces ‘differentiable homography’,
our layer is a special case of theirs, but we can utilize multiple images in a single training iteration and encode problem constraints directly in the model. Our solution also takes advantage of both input modalities but focuses on detailed building understanding. As opposed to the previous works, our fusion strategy directly utilizes the building geometry to create a 2D latent representation through the projection pooling layer.

(d) Closely related to our study is the classification of building age, condition, and land use from a single street-view image (Zeppelzauer et al. 2018; Koch et al. 2018; Kang et al. 2018; Zhu, Deng, and Newsam 2019). Urban zone classification was also explored in multi-view settings (Srivastava et al. 2018; Hoffmann et al. 2019; Srivastava, Vargas-Muñoz, and Tuia 2019), which we use as our multi-view multi-task baseline models. We are the first to study buildings from all around the world, where multiple recognition tasks are trained together. Most importantly, we propose a new fusion strategy based on building geometry, which boosts our models’ classification accuracy.

Dataset

One of the goals of this study is to create the first large-scale benchmark of buildings that represent a diverse spectrum of architectural styles, locations, and building attributes. Building sizes vary: from small wooden houses, through churches made of brick, to extensive concrete manufacturing facilities. Thus, models trained on such a dataset should apply to any building in the world. We present the world heatmap of building sites in Appendix A.

Our dataset of top-view images was collected from Google Maps and Bing Maps, while the street-view photos come from Google Street View and Bing StreetSide. The resolution of top-view images is around 30 cm per pixel; see Appendix B for more information on aerial images’ resolution. We built a custom tool and annotated the georeferenced top-view images with the following set of classes: buildings, temporary structures, trees. We selected the georeferenced street-view photos based on the building location and other visible objects on the top-view image to account for potential occlusions. The images were selected to have the best building visibility and cover all sides of the building. For every building, we have one top-view image and between 1 and 9 street-view photos. Knowing the street-view images’ locations, we compute the field of view required for every image to capture the entire building. Since these locations can be imprecise, we set the actual field of view to be 20% broader to make sure the whole building is present in the photo. Top and street-view images usually come from different dates, but our annotators verified that both images refer to the same structure. We annotated construction type, number of floors, roof pitch, roof geometry, facade material, and occupancy type for every building of interest. Hired contractors double-checked all the annotations and building characteristics. Additionally, construction types were checked by a professional architect.

We classified the building attributes as follows (Stone 2018):

- the construction type describes the predominant material used for the building construction, and it consists of four classes: masonry, metal, reinforced concrete, and wood;
- the number of floors includes five categories: one, two, three, four and five or more;
- the roof pitch was divided into four categories: flat (0° - 9.5°), low (9.5° - 22.5°), medium (22.5° - 37°) and steep (> 37°);
- the roof geometry classes are: flat, gable, hip and shed;
- facade material was grouped into brick, cement block, concrete, glass, metal, plaster, plastic, stone, and wood;
- the occupancy classes are the following: agriculture, commercial, industrial, mercantile, public, and residential.

Our dataset consists of 6477 training and 3197 testing building scenes, split by stratified sampling. The total number of street-view images is 29350 in the training dataset and 10402 in the testing dataset. Most street-view photos are of size 640 × 640, but around 25% of them are taller. This is because we extended some of the images upwards – by stitching together multiple photos – to obtain a better view of taller structures. The size of the top-view image (one per building) depends on the building size.

In summary, our dataset includes the following: top-view images, street-view images, the location, direction and field of view of the street-view images, and the georeferenced footprint of the building.

In Figure 1 we present a full example, showing all images and metadata available for a single building, along with the correct output classes.

In Appendix C, we present more information about the dataset: an example of each class for each attribute, a sample of full scenes with descriptions, and a comparison with existing datasets from the literature.

Multi-View Multi-Task Approach

We start by introducing the basic definitions and then present increasingly strong models for building attribute classification.

We denote the top-view image by \( I_{0} \), and the \( n \)-th street-view input image by \( I_{n} \), for \( n \in \{1, \ldots, N\} \), where \( N \) is the total number of street-view images for the given building. We transform the input images to obtain spatial feature maps \( f_{0}, f_{1}, \ldots, f_{N} \) using feature extractor networks CNN\(_{TV}\) (top-view) and CNN\(_{SV}\) (street-view), with \( d \) output channels for street-view features and \( d_{0} \) output channels for top-view features:

\[
f_{0} = \text{CNN}_{TV}(I_{0}) \quad f_{n} = \text{CNN}_{SV}(I_{n}).
\]  

(1)

Feature vectors obtained from these feature maps by mean-pooling along spatial dimensions are denoted as \( v_{0}, \ldots, v_{N} \). We use six linear heads with softmax activation to generate the class probabilities for each of the tasks. We train the models to minimize the total cross-entropy loss summed over all of the classification tasks. We consider the following baselines.

Top-view (TV) Only the top-view input image is used. The input to the six task-specific classification layers is \( v_{0} \).
Street-view (SV) To fuse the street-view input images, we compute feature vectors \( v_1, \ldots, v_N \), and use their average as input to the task-specific classification layers.

Street-view + Top-view separately (SV+TV separately) The different tasks achieve varying results depending on which kinds of images are given as input: street-view or top-view. Based on the results of the baselines that use a single input type (either street-view or top-view), we choose the best baseline model for each task separately. We train the top-view model to classify the roof slope and roof geometry, while the street-view model to classify other attributes. The two networks are completely separate and are trained separately.

Street-view + Top-view (SV+TV) We use two independent feature extractors: one for street-view and one for top-view images. The vectors \( v_1, \ldots, v_N \) are averaged as in the street-view baseline, and then this average is concatenated with \( v_0 \) to form the input to the task-specific classification layers. This fusion is shown visually in Appendix D. We also consider a modification where the same feature extractor network is used both for the top-view and street-view images (i.e. \( \text{CNN}_{TV} = \text{CNN}_{SV} \)).

Our Model
The baseline models described in the previous section take into account neither the relatively simple cuboid-like geometry of a typical building nor the source positions of the street-view images. These factors, especially the distance between the building and the source of the street-view photo, may impact what is visible on the input image. We design a fusion strategy to leverage the building geometry, street-view photo location, and its direction. It results in a single, unified, top-view representation of the building. We extract the feature maps from all street-view photos as defined in Equation (1), and map these features (we call it a projection in this work) onto relevant parts of the building polygon, as seen on the top-view image. Those projected features are overlapped and concatenated with the top-view aerial representation to let the model infer the entire scene.

The final representation encompasses the following information: all street-view images, top-view image, building footprint, street-view image positions, directions, and fields of view.

Projection Pooling
We describe the construction of the unified building representation following the explanatory Figure 2. For each street-view image \( \text{Im}_i \) of size \([H_i, W_i, 3]\), we extract the feature map \( f_i \) of shape \([h_i, w_i, d]\), and average out the vertical dimension, while keeping the horizontal dimension intact. It results in a feature stripe \( s_i \) of shape \([w_i, d]\), which will be projected onto the building outline. Since all the street-view images in our dataset are of the same width, we can replace the \( w_i \)'s by one value \( w \), so all the stripes \( s_i \) are in fact of shape \([w, d]\). We initialize a zero tensor \( T \) of shape \([h_0, w_0, d]\), where \( h_0 \) and \( w_0 \) are the height and width of the feature map \( f_0 \). \( T \) represents a top-view 2D-grid of \( d \)-dimensional neurons. Neurons in \( T \) are projected from each stripe \( s_i \). By concatenating \( T \) of shape \([h_0, w_0, d]\) with the top-view feature map \( f_0 \) of shape \([h_0, w_0, d_0]\), we construct the final unified representation.

In this paragraph, we explain the details of how we project a stripe \( s_i \) onto the tensor \( T \). Using the building outline, source location, and direction of street-view image \( \text{Im}_i \), we
compute the visible parts of the building polygon (represented by colored segments on tensor $T$ in Figure 2). An efficient way of computing them is described in Appendix E. These segments correspond to parts of the edges of the building wall: more feature vectors are summed if the building wall is in proximity. Therefore, we propose a variant of the Sum strategy i.e. $s[6]$. The ‘sum’ strategy takes the weighted sum between pixel boundaries $X s[5] + Y s[6]$. The ‘avg’ normalizes the ‘sum’ strategy i.e. $(X s[5] + Y s[6]) / (X + Y)$.

for deriving the neuron value from the $s_i$ stripes. While we describe these strategies below, we also presented them visually in Figure 3. In all strategies, $FoV$ represents the field of view angle for the entire street-view image, $w$ is the width of the associated stripe $s$, and $a_s = w \frac{a}{\pi d}$ for $x \in \{l, r, c\}$.

Nearest (N) uses the angle, measured clockwise, between the beginning of the field of view of given street-view image and the center of the neuron. We denote this angle by $a_c$. In this strategy, we use a single feature vector $s$ to calculate the value of neuron $p$:

$$P_{\text{nearest}} = s(a_s').$$

(2)

In other words, we draw a ray from the location of the street-view photo towards the neuron’s center, calculate the position $p \in \{0, \ldots , w - 1\}$ of its intersection with the stripe $s$, and take the $d$-dimensional feature vector at this position (recall that $s$ is of shape $[w, d]$).

Sum (S) uses two rays pointing towards the pixel’s ends instead of one ray pointing towards the pixel’s center. Denote the angles of these rays by $a_l$ and $a_r$ for the left and right one, respectively. Then, the value of the projected feature vector is defined below, where $\{x\} = x - \lfloor x \rfloor$.

$$P_{\text{sum}} = s(a_l') \cdot \{-a_l\} + s(a_r') \cdot \{a_r\} + \sum_{i=[a_l']}^{[a_r']-1} s(i) \quad \text{(3)}$$

Average (A) follows a similar strategy to Sum — i.e. it takes into account feature vectors from multiple positions along the feature stripe $s$. However, a potential problem with the Sum strategy is that the magnitude of values in the resulting vector may vary significantly with the distance to the building wall: more feature vectors are summed if the building wall is in proximity. Therefore, we propose a variant of

Algorithm 1 Pool($h', w', \text{prt}, s_i, \text{src}_i, \text{dir}_i, \text{FoVi})$

\[
\begin{align*}
p &= (h', w') \\
\text{if } p \text{ not in ConeOfVisibility}(s_i, \text{dir}_i, \text{FoVi}) &\text{ then return } 0 \\
\text{end if } \\
\text{if } p \text{ not in Boundary(prt)} &\text{ then return } 0 \\
\text{end if } \\
\text{if } p \text{ in OccludedByOtherWall(prt, src}_i, \text{dir}_i, \text{FoVi}) &\text{ then return } 0 \\
\text{end if } \\
\end{align*}
\]

\[
\begin{align*}
\text{if } \alpha_1, \alpha_c, \alpha_r = \text{PixelToStripeRange}(p, \text{prt}, s_i, \text{dir}_i, \text{FoVi}) &\text{ then } \\
\text{return } \text{StripeSample}(\alpha_1, \alpha_c, \alpha_r, w, s_i, \text{FoVi}) \\
\end{align*}
\]
the Sum strategy, where the output is additionally normalized. We choose to simply divide the resulting feature vector by the width of the part of \( s \) corresponding to the neuron \( p \):

\[
\text{p}_{\text{average}} = \frac{\text{p}_{\text{sum}}}{a_r - a_l}
\]

Visualizing Projected Features

We visualize the features extracted from the street-view photos and how they are projected to create the final unified building representation. We obtain features from the test set pictures corresponding to multiple buildings and project them to three dimensions using principal component analysis, resulting in RGB coordinates. It allows us to visualize a single feature vector as a color.

We visualize the street-view feature maps both before and after their height dimension is averaged out. Then, we look at the feature vectors projected onto the building polygon from each street-view photo separately. We show this in Figure 4. Note that the south wall looks substantially different from the others for the selected building, and this difference can be seen in the visualizations. Moreover, feature maps in positions where no structure is visible get projected to the same uniform color (pink), showing that the feature extractor ignores irrelevant information. Finally, we examine the unified representation vectors, i.e., after max-pooling the features projected from different views. Again, we see that the side of the building providing different visual information has a substantially different projection.

Experiments and Results

Experimental Setup

We benchmark the proposed approach on our newly collected dataset of building attributes. Street-view images are resized to size \( 500 \times 500 \). The top-view image is cropped to the bounding box containing the building of interest and resized to keep the aspect ratio so that the length of the longer side is 500. As the feature extractor network, we use the ResNet-50 model pre-trained on the ImageNet dataset, as available in PyTorch (Paszke et al. 2017). We discard the final fully connected output layer. We freeze the parameters of the stem and the first block in the pre-trained ResNet-50 feature extractors and do not fine-tune the batch normalization layers. We train the network using stochastic gradient descent with momentum, on a single GPU (see Appendix G for details), and with a batch size of one building. The effective number of images in a single batch is equal to 1 for the top-view branch, while for the street-view branch, it varies from 1 to 9.

We use the learning rate of 0.0001 for ten epochs and then 0.00001 for one more epoch. We set the momentum to 0.9. We apply L2 regularization with a weight decay of 0.001 and augment the dataset with random color jittering. We compare the models using average classification accuracy over the six tasks. For details about hyperparameter tuning, see Appendix H.

Image dropout (ID) Inspired by dropout (Srivastava et al. 2014), we regularize the training by randomly dropping entire street-view images, which we call image dropout. During training, every street-view image in the batch of a given building is omitted with probability \( p \). We make sure that there is always at least one street-view image in a batch. We do not rescale the values of the features as done in the original dropout. With image dropout, we force the model to use a different set of images in each iteration. We do not apply image dropout at test time, allowing the model to use all of the street-view images available for the given building.

Projection thickness (TH) When rounding up the top-view building polygon to a set of neurons in \( T \), we obtain a thin building outline, where each side is discretized as a one-pixel wide line. We consider using an outline wider than one. Projection thickness of \( k \) means that the building outline is discretized to line segments, which are \( k \) neurons wide. We assume that pixels belonging to the same side do not occlude each other. Widening the polygon makes it easier for the convolutional layer applied on top of the unified representation to capture features from adjacent walls.

Cutout (CU) To further regularize the training over multiple potentially redundant street-view images, we apply
Cutout (DeVries and Taylor 2017). For every street-view image, with probability $q$, we blackout 40% of pixels, by covering the image with a randomly placed black rectangle. Using this approach, we force the model to rely on multiple parts of an image when performing classification.

**Splitting the street-view images (SS)** To perform projection pooling, we average the street-view feature maps $f_i$ along the height dimension, which gives a rough feature stripe $s_i$. We also investigate a different strategy, which allows the model to take into account the differences between lower, middle, and upper parts of the street-view images. We split the feature representation into $k$ different tensors along the height dimension, and concatenate them along the depth dimension. In other words, we shift the height dimension into the depth dimension. From a feature map $f_i$ of size $[h, w, d]$, we obtain one of size $[h, w, d \cdot k]$ and apply mean-pooling along height dimension with projection pooling.

**Results**

In this subsection, we discuss the experimental results.

In Table 1, we present a comparison between competitive methods and the new proposed model with the projection pooling layer. Using only the top-view images gives worse results than using only the street-view photos. It is still better than one would expect, given that most parts of the building are not visible on the top-view image. Unsurprisingly, using the street-view rather than the top-view gives the most substantial gains when counting the number of floors (+15%) and predicting facade material (+10.4%). On the other hand, top-view images are more informative when it comes to roof geometry (+3.9%) and roof pitch (+2.5%).

Combining both top-view and street-view images gives better results than using a single source, but only by 0.35% than when using street-view alone. Separate networks for the top and street-view images yield a small improvement. We obtain the best baseline results when training the top-view and street-view networks independently on different tasks.

Incorporating the projection pooling layer results in about 3% accuracy improvement over the best baseline.

In Appendix I, we present the results of an ablation study, which examines the impact of all components of our final model. First, we train a vanilla projection pooling model with different stripe sampling strategies, and find that using the averaging strategy gives the best results (75.93%). Adding image dropout with probability $p = 50\%$ further regularizes the model, and increases average accuracy (+0.23%). We then tested values higher than 1 for projection thickness, and found that using 3 improves performance (+0.38%), while increasing beyond this number did not bring further gains. Applying cutout with probability $q = 50\%$ improves our model even further (+0.33%), which shows the importance of using a wide variety of regularization techniques. The final improvement (+0.41%) comes from splitting the street-view feature maps into three tensors, and concatenating them along the depth dimension. In this way, lower, middle, and upper parts of the building are separately projected, which is especially helpful for discrimination of the number of floors.

**Impact of multiple images** We investigate the impact of varying the number of street-view images per building. For $k \in \{1, 2, 3, 4\}$, we test our best model with a restriction to use at most $k$ street-view images.

The results of this comparison are shown in Appendix J. We see a substantial (+1.36%) gain from using more than one street-view image, suggesting that photos from multiple angles are often necessary for correct classification. On the other hand, the benefits of adding more street-view images quickly plateau, as there is a considerable overlap of information provided by the different street-view photos.

**Conclusions**

Our study presents a novel solution to a practical problem of building understanding. For the first time, this problem is approached using the building geometry inside a deep neural architecture to create a unified high-dimensional representation. We propose a new way to integrate the features from multiple views called projection pooling. It is a general method for creating a unified representation of 3D objects from orthogonal projections and is particularly well suited for building analysis. In the future, it can be tested against analogical setups, such as mammogram analysis (Morrell et al. 2018). We propose a model for building feature recognition, which incorporates the projection pooling layer, and its results are superior to highly tuned baseline models.

We build a new dataset with fine-grained building attributes and analyze techniques for integrating information from multiple views. The dataset establishes a demanding benchmark for state-of-the-art deep learning methods. It requires reasoning about numerous images at once to give accurate results. We plan to expand the dataset to detect objects, such as doors and windows.
References


