# Building Recognition in Air and Satellite Photos

D.I. Bulatitskiy[1], A.K. Buyval[2], M.A. Gavrilenkov[1]
bulatizkydi@mail.ru, alexbuyval@gmail.com, gavrilenkov@umlab.ru
[1] Bryansk State Technical University, Bryansk, Russia
[2]Innopolis University, Innopolis, Russia

*The paper deals with the algorithms of building recognition in air and satellite photos. The use of convolutional artificial neural networks to solve the problem of image segmentation is substantiated. The choice between two architectures of artificial neural networks is considered. The development of software implementing building recognition based on convolutional neural networks is described. The architecture of the software complex, some features of its construction and interaction with the cloud geo-information platform in which it functions are described. The application of the developed software for the recognition of buildings in images is described. The results of experiments on building recognition in pictures of various resolutions and types of buildings using the developed software are analysed.*

***Keywords:*** *Earth remote sensing, building recognition in photos, convolutional neural networks, semantic picture segmentation.*

## 1. Introduction

At present, the recognition of construction objects in satellite and air photographs, which is a part of operation of many government departments and commercial structures, is often carried out manually. Such processes as cadastral surveys, control over observing the borders of separate and protective zones, use of land as intended, control over the setting of buildings on the state registration and other require considerable cost and labor. Therefore, it is necessary to automate recognition and classification of objects in satellite and air photographs through the use of information technologies, in particular, computer vision and machine learning, which show good results in related fields.

The source data for the problem to be solved are usually GeoTiff files, which contain both the terrain image and information on the spatial resolution of pixels and the image binding to geographical coordinates. As the output, it is necessary to obtain the contours of the detected buildings in vector form in geocoordinates.

Several phases can be distinguished in the solution of the initial problem:

1. Getting a bitmap of the terrain from the original GeoTiff file and direct building recognizing, that is, selecting areas in the picture and classifying them as buildings of a particular type.
2. Polygon boundary detection in vector form and converting bit-mapped coordinates into geographic ones based on geodata from the original GeoTiff file.
3. Post-processing of selected polygons, including the application of rules and heuristics for filtering and classification refinements.

Each processing phase uses its own set of approaches and technologies, but for the convenience of the end user it is advisable to implement the solution of this problem as a single act.

## 2. Selection of Methods and Algorithms for Solving the Problem of Building Recognition

The building recognition problem can be referred to a class of machine vision problems called "semantic segmentation", in which each pixel of the original image must be assigned to one of the predefined semantic classes. If the way of referring pixels to semantic classes corresponds to the human perception of the image, the pixels will be grouped into areas that contain dots of a certain class only or mainly this class. Thus, the whole image will be divided into a finite number of a segment, each of which are an object of one of the required classes or is its background.

To solve the segmentation problem, two types of methods can be distinguished: 1) classical and 2) based on artificial neural networks (ANNs).

Classical methods include such methods as K-means clustering, edge detection, watershed transformation and others. However, classical approaches, as a rule, show good results only on simple images and after careful adjustment of parameters. At the same time, they are extremely unstable to various changes in the image (brightness, contrast, and others). And, probably, the most important drawback is that these methods do not allow to determine the class of the found object.

In their turn, image segmentation methods based on artificial neural networks significantly surpass classical methods in accuracy and stability.

Analysis of some papers [1-6] and competition results of image processing and Earth remote sensing (ERS) [7-10] allowed to conclude that the use of convolutional neural networks for solving the problem of building recognition is reliable, and U-Net and DeepLabV3 architectures are the most attractive for further research and experimental testing.

On the basis of source code libraries provided by the authors of the selected architectures, we tested software for training the corresponding neural network models and quality control of their work on the basis of Jaccard index.

At the first stage of the work there were only the results of satellite photos, air photos using manned and unmanned aircrafts to obtain high-resolution images were only performed by outsourcers, so there were tasks of testing the selected ANN architectures set in the following areas. Firstly, it was necessary to assess the impact of hyper-parameters of networks on their work. Secondly, we were to check the assumption that the shade marking of buildings can have a beneficial effect on the building recognition. Finally, it was necessary to choose one of the architectures for further use in the project.

For the experiments, a set of data on satellite images was prepared, including more than 600 images with different types of buildings. In total, more than 5,000 buildings of various classes were represented in these images. The images were carefully labeled and divided into a training sample (448 images), a validation sample (110 images), and a test sample (59 images). In each sample, images with large buildings, private sector, and no buildings are presented in the same proportions.

Training of each model took up to several days. Therefore, it was too difficult to perform experiments for testing all possible combinations of hyper-parameters and marking variations. Instead, the effect of hyper-parameters on the shade marking was first studied and their best combination was chosen. Then, using the best and worst combination of hyper-parameters ANN work was tested on the marking without shades influence. The result of this check is presented in tables 1-2.

The following conclusions were made according to the analysis of Tables 1-2.

1. Both architectures showed that taking shades into account only makes the result worse. Therefore, in the future, it was decided to perform the marking and all other works without taking shades into account.
2. The best values of Jaccard index were achieved with the help of DeepLabV3 network: 89.5% vs 77% from U-Net. Therefore, it was decided to conduct further research on the basis of DeepLabV network.
3. When we decrease the value of output stride in the studied sample, a slight improvement of the result was observed (especially noticeable on smaller objects), however, it should be noted that source intensity increases multiply (the consumption of time and memory clock for GPU computing, training time). The optimal value of this parameter is 16.
4. Increasing batch_size results in memory consumption during training, but allows to get a much better result and reduces training time. When training, it is recommended to increase this parameter as much as GPU memory allows.

**Table 1.** Results of U-Net Testing

| № | Number of layers | Number of features_root | IoU, (%) |
|---|---|---|---|
| Marking with shades | | | |
| 1 | 3 | 32 | 65 |
| 2 | 3 | 48 | 69 |
| 3 | 3 | 64 | 52 |
| 4 | 4 | 32 | 70 |
| 5 | 4 | 48 | 71 |
| 6 | 4 | 64 | 57 |
| 7 | 5 | 32 | 66 |
| 8 | 5 | 48 | 69 |
| 9 | 5 | 64 | 68 |
| Marking without shades | | | |
| 10 | 3 | 64 | 67 |
| 11 | 4 | 48 | 77 |

**Table 2.** Results of DeepLabV3 Testing

| № | Reduction factor of output_stride | Batch_size | IoU, (%) |
|---|---|---|---|
| Marking with shades | | | |
| 1 | 16 | 2 | 68 |
| 2 | 16 | 4 | 71 |
| 3 | 8 | 2 | 70 |
| 4 | 8 | 4 | 72 |
| Marking without shades | | | |
| 5 | 16 | 2 | 86 |
| 6 | 16 | 4 | 89 |
| 7 | 8 | 4 | 89.5 |

## 3. Software Development for Building Recognition

Simultaneously with the experiments on selecting neural network architecture, software was developed which is intended to function as a part of a cloud geoinformation platform (CGIP). This platform is being created at Innopolis University and should become a comprehensive system for promoting products and services in the field of remote sensing of the Earth. Building recognition in images service (BRiIS) is one of the internal services of the cloud geoinformation platform and does not communicate directly with the users, but the results that the user gets directly depend on the quality of its operation. BRiIS is quite a resource-intensive part of CGIP, and its implementation has a search character: the scenarios are very likely when models and algorithms of BRiIS core may undergo significant changes. These factors have determined the requirements for the organization of BRiIS and CGIP interaction: BRiIS should be as isolated from CGIP as possible; BRiIS should be easy to scale; there should be means of monitoring BRiIS operation. Taking into account these requirements, the architecture of BRiIS service has been developed, shown in Fig. 1.

The following data flows can be identified: task information and source files come from the platform to BRiIS, and resulting files and diagnostic messages - from the service to the platform.

It was decided to organize the first flow through which tasks are transferred on the basis of RabbitMQ queue. The web user interface is on the platform side. The user chooses files for processing and additional recognition options: building classes and images he is interested in, and others. The platform generates processing tasks and sends them to the queue.

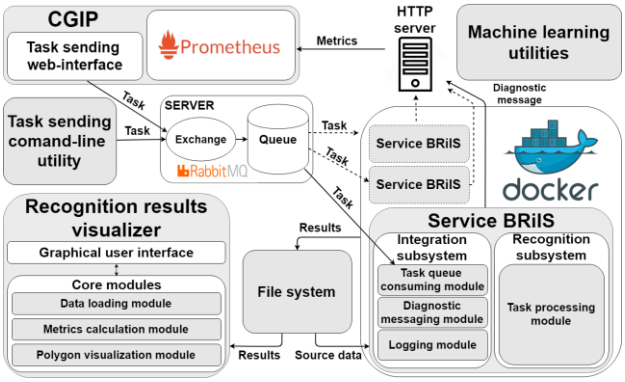


**Fig. 1. Service BRiIS architecture**

Service BRiIS guards the task queue, and as messages appear, the Service de-queues and processes them. This way of transferring tasks provides not only their guaranteed delivery, but also provides scalability of the system. If necessary, several BRiIS will be launched guarding the same task queue and performing the tasks in parallel.

The message contains JSON data structure. It contains the identifier and the task type, the path to the source directory, and the path to the target directory where service BRiIS should write the resulting files.

For debugging purposes, a command-line utility has also been developed that allows you to send tasks to the queue for processing one at a time or in batches.

The second data flow, which provides the feedback from service BRiIS, is organized by sending diagnostic messages by POST over HTTP Protocol in JSON format. Messages are sent when significant task processing events occur: when the task is de-queued, when processing begins, when polygons are formed, and when it is completed.

The third data flow is provided by file exchange. For its successful functioning service BRiIS should have access to the file system of the geographic information platform. During task processing service BRiIS refers to itfor the source files, and writes intermediate and final results. The path to the source directory and the resulting directory is specified in the message de-queued from the task queue.

The central part of the service is a task processing module, which is based on convolutional neural network DeepLabV3. The neural network is surrounded by a processing pipeline of geographical images. First, the image is cut into fragments of the desired size, they are transferred in batches to the neural network, the resulting segmented fragments are combined into the image of the original size. Then, vectoring procedure finds the outlines of buildings, approximate them in polygons with pixel coordinates, and performs the initial filtering of noise and transfers the polygons into geographical coordinates based on the position and scale data of the source geoimages. Finally, the selected polygons are post processed based on heuristics.

BRiIS uses a large number of libraries, many of which are large, require related libraries of certain versions, or involve a non-standard installation process. All this greatly complicates the

environment adjustment for BRiIS. In some cases, with certain combinations of operating system versions and installed software, adjustment may not be possible. Therefore, we decided to run BRiIS in an isolated docker container environment.

In addition to insulation of applications, using docker containers makes it easy to deploy and replicate. The operating environment, all necessary libraries with all dependencies, as well as application modules and scripts are packaged into the image. This image is transferred to other machines, unpacked, and the service container is started.

Machine learning utilities shown in Fig. 1 are not a direct part of CGIP, they are designed to prepare neural network models, which are then used by BRiIS. The utilities allow to generate ground truth labels based on the original GeoTiff files, and the ground truth label, made in the vector form in GIS-systems, then to assemble these training sets in a special format of tf-records and finally to execute the learning procedure itself.

Python was chosen as the language for creating BRiIS and related programs. Neural networks are made with the use of open-source libraries for machine learning TensorFlow from Google. Framework Nvidia from CUDA is used to speed up calculations.

## 4. Evaluation Criteria

Taking into account the objectives of service BRiIS developing, there are two main typical scenarios of its application:

- reconciliation of the building boundaries, recorded within the new session of ERS, with the registered;
- detection of new buildings within the new session of ERS, not previously recorded.

In the first case, it is of paramount importance to determine the boundaries of buildings as accurately as possible. For these purposes, the best are the criteria for calculating the accuracy of the recognition algorithm, based on the quantitative similarity between the ground truth label and the predicted one in the pixel-by-pixel comparison. In this paper one of the strictest criteria was used, that is Jaccard index, which in its finitely multiple version (at a given resolution, the image is a finite set of pixels) can be written as follows:

$$K = \frac{n(A \cap B)}{n(A) + n(B) - n(A \cap B)} = \frac{n(A \cap B)}{n(A \cup B)}.$$

This measure is also called Intersection over Union (IoU), which reflects the essence of the fraction above.

For the second case, more suitable are measures based on counting the number of buildings, which polygons in the predicted label sufficiently intersected with the polygons of the ground truth label. In other words, the comparison is not made by pixels, but by pieces (or buildings). The score is calculated as follows:

$$Fscore = \frac{2 * \text{Precision} * \text{Recall}}{(\text{Precision} + \text{Recall})},$$

$$\text{Precision} = \frac{TP}{(\text{TP} + \text{FP})},$$

$$\text{Recall} = \frac{TP}{(\text{TP} + \text{FN})},$$

where Precision is called algorithm accuracy, Recall is the completeness, TP is the number of true positives, FN is the number of false positives, and FN is the number of false negatives. In this paper, a building is considered to be correctly detected if Jaccard index for it and its label exceeds 50%.

After finishing software development of service BRiIS, including results visualizer modules, it became possible to use not only a bit-mapped metrics based on Jaccard index (IoU), but also F-score objective measure, which allows to assess the results better in the context of the ultimate goal – building recognition.

There is certainly a direct connection between Jaccard index and F-score: the better the image is segmented, the easier it is to find the correct building boundaries. However, there is a significant difference. Bit-mapped metrics is much more loyal to the gaps or, on the contrary, false recognition of small buildings, as well as to situations where two close buildings merge into one or, conversely, one building of a complex configuration breaks apart.

To quantify the quality of building recognition based on F-score and results visualizing, a separate application is developed in Python using Tkinter library.

## 5. Study of Network DeepLabV3 Operation in Images of Various Types

At the second stage of the work, not only space images were ready, but also air photographs: in the urban area with the resolution of 0.05 and 0.1 m/pixel, in rural areas – with the resolution of 0.1 m/pixel. In addition to red-green-blue images (RGB), there were also colored infra-red (CIR) satellite images with the resolution of 0.5 m/pixel. In total tagged images consisted of approximately 50,000 buildings. For evaluation images of different types and with various types of buildings, containing over 7,000 buildings, were left (i.e. not used in training).

Now the task was to test in practice how the combination of training sets affects the final result.

Intuitively, it has been assumed that separate models for datasets of air photos (made by unmanned aerial vehicle, UAV and by manned aircraft, MA) and space photos (made by satellite, SAT) should be trained, as their scales are too different. Similarly, RGB data sets differ from CIR sets, and therefore separate models should also be trained for them. So, the following models were trained: 1) UAV+MA, 2) SAT(RGB), 3) SAT(CIR) and the results were evaluated based on F-score. The results of the evaluation are shown in table. 3 (the second column). Fig. 2 shows an example of the results of building recognition in UAV image.

Then a general model for all RGB images (together UAV, MA, SAT) was trained and a separate one for SAT(CIR) images. The results of evaluation of these models are shown in table. 3 in the third column. As it can be seen from table 3, the result of F-score has not changed for UAV+MA images, but improved for SP images. Probably, this improvement of recognition results of SAT images is due to the training set is too small, and adding UAV and MA images, even differing in scale, beneficially effects learning.

Since the training set for SAT(CIR) is even smaller than SAT(RGB), then a unified model for all types of available images was trained. The results of evaluation of the unified model are shown in table. 3 in the fourth column. As it can be seen from table 3, the result of F-score has not changed for MA images, but improved for UAV images and slightly deteriorated for SAT. However, the overall F-score has slightly improved.

Another advantage of the unified model is that there is no need to prepare separate datasets for different types of recording. Also, the unified model will speed up the work of the service, since no time will be spent on downloading different ANN models in case of recognizing images of different types.

For evaluating the results and forming columns 2-4 of table 3, all objects larger than 2x1m for air photographs and 4x4 for satellite images were taken into account. If we ignore all buildings less than 3x3 and 7x7 respectively, the results are significantly improved (see the fifth column of table 3). This proves the assumption that small objects are the most difficult to recognize.

**Table 3.** Results of Network DeepLabV3 Testing

| Image | F-score | | | |
|---|---|---|---|---|
| | Three separate models | Two models (RGB/CIR) | Unified model | Unified model (3x3, 7x7) |
| **UAV** | | | | |
| 16-1-239-157-B-1 | 0,879 | 0,878 | 0,850 | 0,923 |
| 16-1-239-157-B-2 | 0,846 | 0,871 | 0,885 | 0,940 |
| 16-1-239-157-B-3 | 0,904 | 0,896 | 0,913 | 0,946 |
| 16-1-239-157-B-4 | 0,813 | 0,857 | 0,861 | 0,922 |
| 16-1-239-157-A-7 | 0,846 | 0,826 | 0,884 | 0,927 |
| 16-1-239-157-A-9 | 0,852 | 0,872 | 0,887 | 0,946 |
| 16-1-239-157-A-10 | 0,867 | 0,862 | 0,885 | 0,899 |
| 16-1-239-157-A-11 | 0,868 | 0,915 | 0,906 | 0,965 |
| 16-1-239-157-A-13 | 0,883 | 0,854 | 0,888 | 0,937 |
| 16-1-239-157-A-14 | 0,858 | 0,878 | 0,888 | 0,928 |
| 16-1-239-157-A-15 | 0,930 | 0,853 | 0,888 | 0,930 |
| Konstantinovka | 0,867 | 0,846 | 0,872 | 0,893 |
| **Averagefor UAV** | **0,868** | **0,867** | **0,884** | **0,930** |
| **MA** | | | | |
| 16-33-23-(131-д) | 0,760 | 0,771 | 0,763 | 0,829 |
| mesha_2_12 | 0,790 | 0,783 | 0,808 | 0,865 |
| 16-33-23-(018-е) | 0,777 | 0,772 | 0,770 | 0,798 |
| 16-33-23-(018-b) | 0,761 | 0,781 | 0,770 | 0,721 |
| **Average for MA** | **0,772** | **0,777** | **0,778** | **0,803** |
| **SAT** | | | | |
| Fr4a_RGB | 0,705 | 0,710 | 0,685 | 0,764 |
| Fr7a_RGB | 0,612 | 0,694 | 0,658 | 0,715 |
| Fr7a_CIR | 0,605 | 0,667 | 0,666 | 0,728 |
| **Average for SAT** | **0,641** | **0,690** | **0,672** | **0,740** |
| **Total average** | **0,812** | 0,823 | 0,828 | **0,872** |

The main task of the work was to create a service for building recognition without additional classification by their functional profile or other criteria. However, most part of marking at the second stage was performed according to ten classes: Background, Residential building, House, Industrial or commercial building, Administration or educational building, Other non-residential building, Building under construction, Greenhouse, Garages, Foundation of building. A separate model was trained on these data, but its results were much worse due to frequent errors in the classification of found objects. Tables with the results for ten classes are very bulky, that is why they are not given in this paper.

## 6. Conclusion

Analysis of papers and experimental data obtained when testing software developed by the authors prove the efficient use of convolutional neural networks and, in particular, DeepLabV3 architecture for building recognition in satellite and air photos. The average F-score on the sample of images under study exceeded 80%, which is a very good result, taking into account the fact that the test sample of images had difficult for recognition objects.

These objects hard for recognition include poorly structured clusters of containers and tents in markets, neighborhoods with old low-rise buildings and an abundance of small very close to each other household buildings, as well as industrial facilities of complex shapes with many link buildings and transporters between buildings.

F-score results are much lower than 80% for all these types of complex constructions. This is quite natural, because even a man using semantic context find it difficult to determine where the boundary between such objects are. However, finding such areas in the processed images and the application of separate models and algorithms to them can give a significant increase in the quality of recognition. It is the development of such combined architectures that is a priority for further research within the framework of the project development.

**Fig. 2. Example of recognizing on UAV model, trained only on UAV+MA sets (first is an original image, second is reference marking, third is the result of recognition result on top of the reference marking)**

## 7. References


[1] Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. arXiv:1511.00561v3 [cs.CV] 10 Oct 2016

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. arXiv:1606.00915v2 [cs.CV] 12 May 2017

[3] Liang-Chieh Chen, George Papandreou, Florian Schroff, Hartwig Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv:1706.05587v3 [cs.CV] 5 Dec 2017

[4] Jonathan Long, Evan Shelhamer, Trevor Darrell. Fully. Convolutional Networks for Semantic Segmentation. arXiv:1411.4038v2 [cs.CV] 8 Mar 2015.

[5] Olaf Ronneberger, Philipp Fischer, Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597v1 [cs.CV] 18 May 2015

[6] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. arXiv:1312.6229v4 [cs.CV] 24 Feb 2014

[7] 2015 IEEE GRSS Data Fusion Contest Results http://www.grss-ieee.org/community/technical-committees/data-fusion/2015-ieee-grss-data-fusion-contest-results/

[8] 2016 IEEE GRSS Data Fusion Contest Results http://www.grss-ieee.org/community/technical-committees/data-fusion/2016-ieee-grss-data-fusion-contest-results/

[9] 2017 IEEE GRSS Data Fusion Contest Results http://www.grss-ieee.org/community/technical-committees/data-fusion/2017-ieee-grss-data-fusion-contest-results/

[10] 2018 IEEE GRSS Data Fusion Contest Results http://www.grss-ieee.org/community/technical-committees/data-fusion/2018-ieee-grss-data-fusion-contest-results/