Segmentation of Spatial and Geometric Information from Floorplans using CNN Model

Dr.Anusuya Ramasamy[a], Dr.M.Sundar Rajan[b], Dr.J.R.Arunkumar[c]

[a] Assistant Professor, Faculty of Computing and Software Engineering, Arbaminch Institute of Technology, Arbaminch University, Ethiopia. anusuccess@yahoo.com

[b]Associate Professor, Faculty of Electrical and Computer Engineering, Arbaminch Institute of Technology, Arbaminch University, Ethiopia. msundarrajan84@gmail.com

[c]Associate Professor, Faculty of Computing and Software Engineering, Arbaminch Institute of Technology, Arbaminch University, Ethiopia. arunnote@yahoo.com

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 20 April 2021

Abstract: In automated document analysis, floorplans are a concern for several years and algorithmic approaches have been used until recently. This problem has also improved output with the emergence of Convolutionary neural networks (CNN). In this study, it is the task to retrieve space and geometric data from planes as accurately as possible and the bulk of the information is extracted from a plane image by means of instance segments, such as the Cascade Mask R-CNN. A new way of using keypoint CNN is presented in order to supplement the segmentation, so that precision corner positions can be identified. Then they are coupled to the resulting segmentation in a post-processing stage. With an average IoU of 72.7% compared to 57.5%, the resulting segmentation scores surpass the existing benchmark for CubiCasa5k floorplan datasets. In addition, the mean IoU is increased in almost every class for individual classes. Cascade masks R-CNN have also shown to be more appropriate for this mission than R-CNN masks.

Keywords*: *Segmentation, CNN, Floorplan, IoU, Cascade Mask R-CNN

1. Introduction

In recent years, improved machine learning and computer vision made machines more effective in previously unreachable tasks for a computer system. Computer visualisation research is in the process of detecting objects in an image and of classified pixels, to which object with high precision using instance segmentation[10, 5]. Computer vision research is in progress. Furthermore, models may also figure out how people pose using the detection of the keypoint [13, 24, 4]. The developments are due to the excellent performance of strong Convolutionary Neural Networks (CNNs). In this paper, the use of these two powerful computer vision technologies together will be discussed in order to be complementary in a role where high pixel accuracy is needed.

The topic is to look at the parsing floor plans issue. A floor plan is an image of an apartment or building from above. The layouts are drawn by a CAD programme, but when used in real estate, the planes are rastered into a bitmap that results in lost details. Re-creation is a time-consuming process that involves manual labour, including the room size, the wall length, the location of the window etc. While the issue of automation appears to be trivial at first glance, non-machine algorithms have proved to be difficult to execute well. At present, CNNs[11, 15] are used for the best performing versions.

This research is carried out in partnership with Pythagoras AB who run the facility. As the core part of their company, they use Building Information Models (BIM). A BIM is a building's 2D structures that contain codes for items such as spaces, walls, windows and doors. A previous business project showed that instance segmentation with an R-CNN mask [23] architecture can provide a good prediction for different objects in a picture but still does not have a good enough segmentation to turn them into a BIM. The technique can be further developed, however, as better designs can be tested by the computer vision group. Kalervo 2019 et.al. are also available. [11] A new rich data set with 5000 annotations on the floors has been published. The results were strong, with semantic segmentation and a new representation of their own. But they are also insufficient to be economically viable. This gives incentives to re-examine the parsing issue with the use of improved architecture for CNN instance segmentation to test the new dataset.

2. Background

A tutorial will be available in this section on the computer vision techniques used in this project. The Cascade R-CNN and Keypoint R-CNN in the final floorplan analyser are each compounded to make clear all of the previous components. After a good foundation is built in deep neural networks (DNN), related parsing works are created. Artificial intelligence and machine learning are not anything new. Indeed, A.L Samuel coined the word "machine learning" in 1959[22]. Since the computer was invented, ideas were given about how a machine could learn complex designs and ultimately just like or better that humans are at work. In the past few years, however, there has been a boom in machine learning that can be credited with hardware advancement. Better GPUs have allowed the development of deeper neural networks in particular.

The basic structure blocks are called perceptron in a machine learning system. It is a simple node which can take multiple inputs, use a weight and generate one y. They are programmed to replicate how the brain's neurons act. As several of these perceptrons are positioned in a row, a neural network is produced that can enter and produce a production according to the weights of various connexions. They have one layer, one layer and one output layer. They have one layer. When more cached layers are inserted, the Deep Network (DNN) is now renamed. The production of the first secret layer is incorporated into the second and so forth layer. This encourages the secret representations to become more complex, such that complex tasks can be solved.

Typically the word training occurs as in a model when exploring DNNs. DNN training involves feeding a model data to change the weights of the proceptrons to better respond to the next input. In other words , the model will react to the data. The back-propagation introduced by Rumelhart in 1986 to the AI community[20] is used. The supervised education we use in this study is carried out through the feeding of the model pairs of an input xi and a mark yi. The label is the right output from the input to be generated. In a model there are layers of perceptons and even functions f L to activate. The activation function decides if an incoming XIWI signal is powerful enough to stay in the network. The sigmoid function is a common activation feature. When a model g training, the input xi is taken and g(xi) is given. Then a loss feature is calculated by C (yi, g(xi))). Then the loss feature is calculated. Backpropagation is employed to change G weights with the loss C and "backward propagation" through the neural network. Backpropagation takes error C and the partial derivatives are determined in each direction. This results in the model gradient, and it is possible to calculate the effect of each layer node on the output using the string law. The weights can be changed to allow a slight step in the direction of the gradient, until the same input again, and thus reduce the loss. When training a model, thousands of examples generalise the model to reduce the loss are typically present.

Artificial intelligence and machine learning are not anything new. Indeed, A.L Samuel coined the word "machine learning" in 1959[22]. Since the computer was invented, ideas were given about how a machine could learn complex designs and ultimately just like or better that humans are at work. In the past few years, however, there has been a boom in machine learning that can be credited with hardware advancement. Better GPUs have allowed the development of deeper neural networks in particular.

The basic structure blocks are called perceptrons in a machine learning system. It is a simple node which can take multiple inputs, use a wi weight and generate one y. They are programmed to replicate how the brain's neurons act. By putting several of these perceptrons in line, a neural network is generated that can take input and produce an output based on the weights of the various connexions. They have one layer, one layer and one output layer. They have one layer. When more cached layers are inserted, the Deep Network (DNN) is now renamed. The production of the first secret layer is incorporated into the second and so forth layer. This encourages the secret representations to become more complex, such that complex tasks can be solved.

Typically the word training occurs as in a model when exploring DNNs. DNN training means feeding model data so that percepton weights can be properly reacted to the next input. DNN training means In other words , the model will react to the data. This is accomplished by means of back propagation by Rumelhart in 1986 [20] to the AI community. The supervised education we use in this study is carried out through the feeding of the model pairs of an input xi and a mark yi. The label is the right output from the input to be generated. In a model there are layers of perceptons and even functions f L to activate. The activation function decides if an incoming XIWI signal is powerful enough to stay in the network. The sigmoid function is a common activation feature. When a model g training, the input xi is taken and g(xi) is given. Then a loss feature is calculated by C (yi, g(xi))). Then the loss feature is calculated. The back propagation uses loss C and "reverse spread" through the neuronal network to upgrade the weights of g. The reverse propagation takes the error C in each direction of the inputs and calculates the partial derivatives. This results in the model gradient, and it is possible to calculate the effect of each layer node on the output using the string law. The weights can be changed to allow a slight step in the direction of the gradient, until the same input again, and thus reduce the loss. When training a model, thousands of examples generalise the model to reduce the loss are typically present.

The first basic step in the computer science is the classification of pictures. There are still not trivial tasks in the field of medicine , for example, in which complicated pathological images must be classified to diagnose a patient. However, a natural next stage in computer vision development enables objects to be found and segmented within an image. The main objective in this project, too, is that walls and spaces be well segmented.

Semantic image segmentation means that each pixel is categorised into a predefined label and therefore objects can differentiate from the background in the image. For example, when you make semantic segmentation, all chairs in Figure 1 are treated as chair objects. However, all chairs have their individual segmentation when doing instance segmentation. The latter helps the picture to find different objects, which is beneficial, if, for example, several footpaths or cars in cars are distinguished.

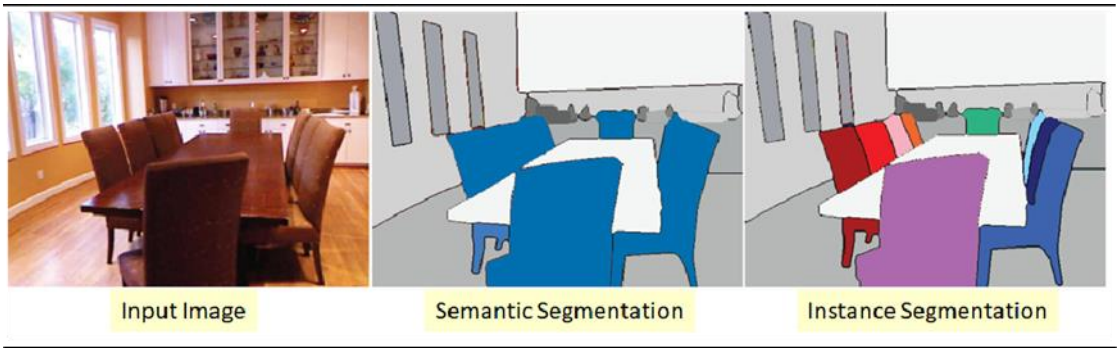

Figure 1 Semantic segmentation and instance segmentation compared. In instance seg- mentation all chairs are segmented individually.

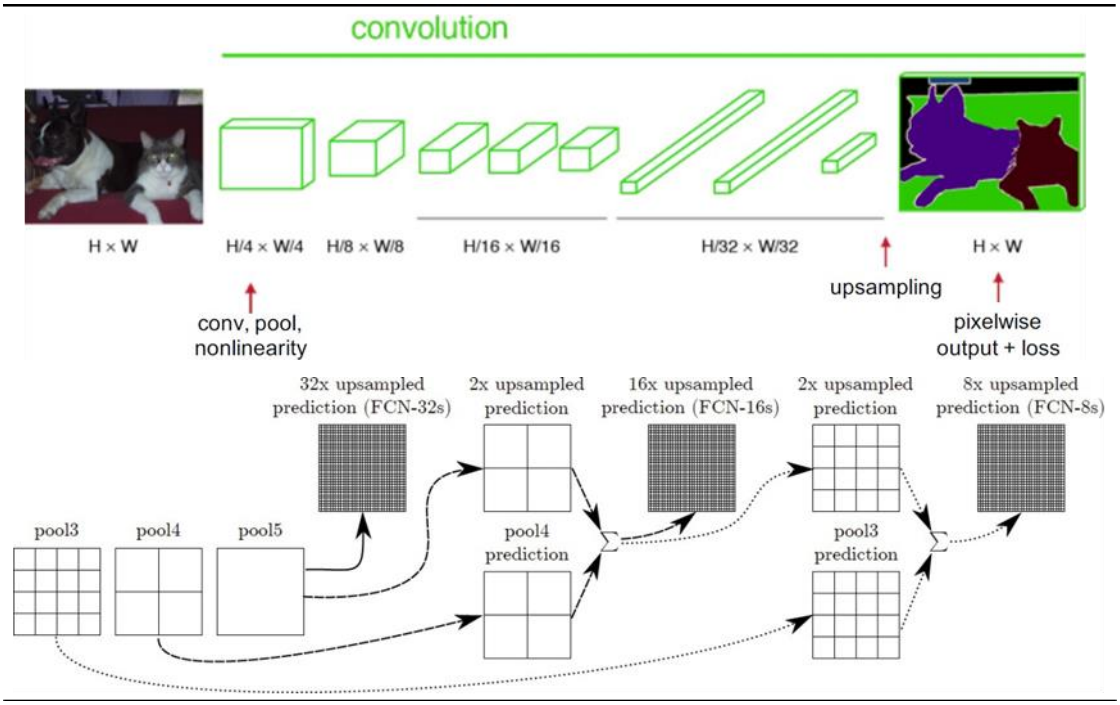

Figure 2 Fully Convolutional Network from Long. Credit: [16]

Long et.al . 2015. Introducing Completely Convolutionary Neural Networks (FCNs) to further advance research[16]. These networks do not eventually generate a gradation of the image level. Instead, an FCN output generates a pixel-wise label map with 1x1 Convolutionary network layers in the last few layers. The practical extraction of convolution layers in the network has now downgraded to a small size the highly detailed output not matching the image size of the original layer. A sampling method can overcome this situation to extend the functionality back to the original image size. In a fundamental example, the up sampling of the functional output is rendered by deconvolutionary layers using interpolation. The FCN proposed by Long (Figure 2) is a typical model of a successful FCN. They also suggested combining the output from different layers to form combined predictions, rather than using only Convolutionary layers. This is useful because the extraction of unique features by adding more Convolutionary layers loses spatial information and the interaction between features. By adding the output of the low layers with the output from deeper layers, the impact of loss of spatial information can be minimised.

3. Methods

3.1 System Overview

The system architecture as a whole comprises similar building blocks and data flows following previous work of Lin and Kalervo. Neural networks are thus used to extract information for crossover and segmentation, then used to generate the final vector representation during a post-processing process. Instead of semanthe segmentation, however, the sections within the structure are different, and the combined style is a new form of representation called Vertex representation. Figure 3 displays the complete device overview of the parser. Two separate neural networks, one for segmentation, and one for extracting vertices are the first sections of this

method. The performance from both networks will then be implemented as an entry into the post-processing algorithm which is a modern method for linking vertical walls and straightening walls between them.

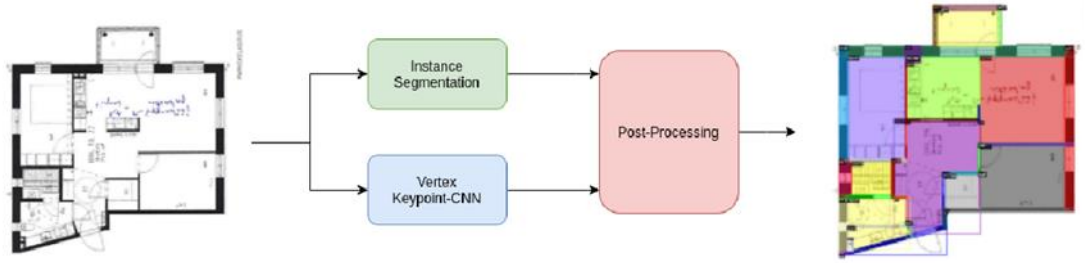

Figure 3 Automatic Floorplan Parser. The rasterized floorplan image is fed to the instance segmentation model and the keypoint model. Then they are used in the post- processing to create the final output.

3.2 Dataset

The dataset consists of 5000 hand-annotated imagery from the CubiCasa5k floorplan. The png format restored floor plans are supported with the vector graphics SVG annotations. In this way, objects such as rooms, walls, windows and doors are represented as polygons in the image and have geometric and spatial link in a plane with other objects. Human annotations are created by people who are qualified to make the pictures consistent. The dataset is divided into 4200, 400 and 400 images for preparation, testing and validation.

3.3 Tools and Detection 2 Backbone

There are several frameworks to choose from that can do the work when constructing a neural network architecture. Linear algebra is at the heart of machine learning, which basically implies that all programming languages are available. There are, however, two popular Python3 deep learning libraries, called Facebook's PyTorch and Google's Tensorflow, both of which are used for various types of implementations of machine learning[1, 18]. Python3 also offers numerous other libraries that help to build image processing and data management pipelines[2, 27]. Therefore, as the programming language to be used in this project, Python3 was a simple choice.

The project set out to explore new ways of segmenting the floor plan that involve rapid implementation in order to test new hypotheses. Detectron2 is a platform developed by Facebook to assist in research into computer vision. It is based on PyTorch and includes several popular network architectures and rapid testing utility tools[29]. Much time was gained by not having to write comprehensive code for the example segmentation and keypoint CNNs by using this system. In this project, Detectron2 is used for all components that include machine learning, while other libraries are used for pipelines and post-processing between the various modules. For stakeholders, it is also beneficial that the project has been tailored to this system for future study.

3.4 Pre-processing

A particular notation called COCO-notation is used by Detectron2, which is a JSON-format used in the COCO dataset. The COCO dataset is commonly used in computer vision research for benchmarking and challenges [14]. However, CubiCasa5k has its own specific notation in the floorplan dataset, which they call a House-representation. The House-representation for a full-image semantic segmentation task is very efficient in constructing a ground reality, but knowledge about individual objects is lost in the pre-processing. Instead of a JSON-file that can be loaded into Detectron2, the House representation also exists as a class at runtime. This involved a re-writing of the COCO-notation representation of the Building.

The COCO-notation is organised such that a list of pictures, a list of objects and a list of class labels are available. Objects are connected via ID to their label and to an image. The artefacts contain segmentation data such as location, bounding box, and polygon points. The ground truths are the segmentations for the artefacts while planning for the segmentation-model training example. A script iterates a JSON-file that is stored on the hard drive through all images in the dataset and structures it. To plan for keypoint-model preparation, the same procedure is used. However, the script needs to extract keypoints from the House representation instead of using polygon segmentation. This is achieved by taking all the walls' end-points and constructing a fixed bounding box around the keypoint of 100x100 pixels. To minimise loading time, the JSON-files are then used as a dictionary by Detectron2.

3.5 Cascade Mask R-CNN + ResNet50

As shown in Figure 9, the Cascade Mask R-CNN model has an expanded Mask R-CNN architecture. The network's bounding box prediction serves as fresh RoI proposals for stage 1 instead of making the final prediction at stage 0 that is performed in Mask R- CNN. In order for proposals to pass stage 0, a confidence threshold of $u > 0.5$ is necessary. They need $u > 0.6$ and $u > 0.7$ respectively for proposals to pass stages 1 and

2. This is what refines the performance so that more high-quality predictions can be made by Cascade Mask R-CNN. The mask branch arrives in stage 2 with the standard implementation of Detectron2.

3.6 Experiments with neural networks

There was a great deal of scope for quick testing of neural network architectures by basing the project on Detectron2. The bulk of the work involved adapting the dataset to the COCO-format and then correctly feeding the data. When that was finished, however, several architectures with various settings could be checked to see what works best. By changing settings, training the network, running inference, and looking at outcomes, the experiments were completed.

The learning rate did not appear to impact a qualified model 's final failure. It did, however, increase training time to lower the rate of learning. The batch-size also had to be changed to match the graphics card 's maximum memory, but it did not affect loss. Adjusting the anchors was one setting that may have been significant, based on my intuition. The scale and aspect ratio of anchors, mainly. From small anchors like [8,16,32,64,128] to big anchors like [64,128,256,512,1024], I did some experiments. But the versions tended to be marginally worse or much worse than the original [16,32,64,128,256] anchor scale. Also, adjusting the aspect ratio was not effective and could in some cases render the model unable to train at all. My inference from this is that to find good matches rather than relying on intuition, the hundreds of settings possibly need to be systematically checked.

3.7 Evaluation Matrix

There are two interesting metrics that can be used to test the accuracy of the built floorplan segmentation method to establish an understanding of how the system works. First, to assess the best performing CNN, the example segmentation CNNs must be compared with each other. In order to get a picture of how well this kind of method works when segmenting floor plans, it is also useful to compare them with example segmentation performed on other tasks. There is a standardised metric called mean average accuracy (mAP) used in challenges such as COCO and Cityscapes[14, 7] to do this. Secondly, it is noteworthy that the completely post-processed floor plan is contrasted with other techniques such as Long, Kalervo and Sandelin 's previous iteration of this project[15, 11]. Since these projects are assessed with semantic segmentation, these metrics must also be used to operate this work.

In papers, the estimate for the competition metric mean average precision mAP is sometimes de-noted AP as it is known by context that in a test dataset several images exist. We need to identify accuracy and recall to obtain the mAP. Precision is a mean certainty about how many items are numbered correctly. The recall is a calculation of how many of the related items have been identified. These are listed as being:

Precision= (TP)/ (TP+FP)

Recal= (TP)/(TP+FN)

Where

TP= True Positive

FP= False Positive

FN= False Negative

We use the Intersection over Union IoU of an object to find TP, FP and FN. The area of ground truth is compared by IoU to the segmented area as shown in Figure 4. It is denoted as TP if the intersected region is greater than a given threshold u. To build a more demanding benchmark, the threshold goes from 50 percent to 95 percent. It is called an FP if the IoU is less than u. If a ground truth object has no detection at all or the marking is inaccurate, it is called a FN.

To calculate the mAP, the formula is as following:

$$AP = \sum (r_{n+1} - r_n) p_{interp}(r_{n+1})$$

Where

$$p_{interp}(r_{n+1}) = max_{\tilde{r} \geq r_{n+1}}(\tilde{r})$$

The r is recall and accuracy is p. Pinterp(r) originates from the interpolation of the recall values from that stage to the largest positive one. A model that perfectly identifies and segments all images will receive a score of 100%, while a model that does not locate a single object will receive a score of 0%. For Mask R-CNN on COCO, the benchmark score is 36.7%.

The precision of semantic segmentation and IoU is measured in raw pixels in order to do this. Taken from the description used by Sandelin and Kalervo in Long[16]. Let n_{ij} be the number of class I pixels that are supposed to belong to class j, where there are distinct classes of ncl, and let ti = j nij be the total number of class I pixels.

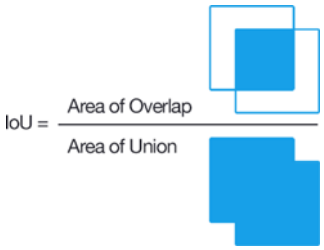

Figure 4 Intersection over union resulting in a value between 0 to 1 which is used together with a threshold to decide which objects gets labelled TP, FP and FN.

4. Results

4.1 Quantitative

The findings presented are from models trained in the system according to the requirements. They are trained on an RTX 2080 Ti GPU for 2 classes (wall and space) until validation loss converges at approximately 35k-50k iterations. The rate of learning was set at 0.00025 and no planned rate of learning was applied. For the models, the training time was about 8-10 hours. The findings are described in two categories where, when doing example segmentation shown in Table 1, the first category compares the different models with each other. The best scores in all categories are recorded by Cascade R-CNN. It is also apparent that as compared to the previous iteration of this project, the availability of a comprehensive and wide dataset has a great effect. Looking at the outcome in Table 2 for bounding boxes, it reveals very close outcomes. In the later results, the Cascade Mask R-CNN-model is then selected for the overall parsing method. An significant relation is also seen in Table 3 where the AP is compared with the model making inference on the COCO dataset as a result of floorplan segmentation. Notice that, instead of 2, the COCO challenge dataset includes 82 classes. It gives an idea, however, of how complex the problem of floorplan parsing.

Table 1 Instance segmentation AP scores comparison between CNN-models. The column denoted AP is an average between the AP threshold ranges [0.5:0.95].

Model	AP	AP_{50}	AP_{75}	AP_{room}	AP_{wall}
Sandelin Mask R-CNN	26.9	46.1	-	46.1	16.8
Mask R-CNN+ResNet50	54.6	77.2	55.9	74.4	34.8
Mask R-CNN+ResNet101	55.3	77.1	56.7	75.4	35.3
Cascade R-CNN+ResNet50	57.4	77.9	59.7	77.2	37.7

Table 2 Bounding Box AP scores comparison between CNN-models.

Model	AP	AP_{50}	AP_{75}	AP_{room}	AP_{wall}
Mask R-CNN+ResNet50	53.6	76.8	55.1	72.3	34.9
Mask R-CNN+ResNet101	54.2	76.4	55.6	73.0	35.3
Cascade R-CNN+ResNet50	57.3	77.5	58.8	76.0	38.7

Table 3 Bounding Box AP scores comparison with baseline trained on COCO challenge dataset. Numbers on COCO reported from [5].

Model	Floorplan AP	AP_{50}	AP_{75}	COCO AP	AP_{50}	AP_{75}
Mask R-CNN+ResNet50 Mask R-CNN+ResNet101 Cascade R-CNN+ResNet50	53.6 54.2 57.3	76.8 76.4 77.5	55.1 55.6 58.8	36.7 39.4 40.9	58.4 61.2 59.0	39.6 43.4 44.6

Cascade Mask R-CNN is selected as the example segmentation network along with the R-CNN Keypoint Mask trained on vertex detection for the semantic evaluation. In semantic assessment, the CNN models need to be optimally designed to gather as much information as possible to do a segmentation afterwards. The threshold score for the model at inference (not when training) is then lowered to get more regions of interest. The R-CNN cascade is reduced to u = 0.3 and the R-CNN keypoint is reduced to u = 0.35. In order to be able to make a

comparison with the baseline as fair as possible, the Cascade R-CNN is trained in their study on all recorded classes except the railing class that was perceived as walls.

Table 4 is a comparison of the baseline of CubiCasa5k recorded in [11]. For the raw segmentation made by both models and then both models with applied post-processing, the evaluation is completed. The context class is recorded by CubiCasa, which can lead to deceptively good performance. The table includes the numbers without the history class in parenthesis for the Cascade-model. The results indicate changes relative to the baseline in mean accuracy and mean IoU. Note also that the model, which is not the case for CubiCasa, is greatly improved by post-processing. A class contrast for the various class marks is in Table 5. Again, the Cascade Vertex model does better than the baseline on nearly every stage.

Table 4 Comparison of Kalervo [11] and this project on semantic segmentation metrics.

Model	Pixel accuracy	Mean accuracy	Mean IoU
CubiCasa	82.7	69.8	57.5
Cascade R-CNN	79.6 (71.5)	75.1 (74.2)	67.9 (58.6)
CubiCasa + Post-processing	80.9	66.6	54.2
Cascade R-CNN + Vertex	83.0 (75.1)	79.0 (76.8)	72.7 (63.3)

Table 4 A comparison between Kalervo[11] and this semantic segmentation metrics project. The raw segmentation is the top portion and the bottom is for both models of applied post-processing. Numbers are for calculation without the back-ground class in parenthesis.

Table 5 Class comparison of pixel accuracy and IoU.

Model	Metric	Background	Outdoor	Wall	Kitchen	Livingroom	Bedroom	Bath	Hallway	Storage	Garage	Other
CubiCasa	PixelAcc	87.3	77.7	85.8	79.9	82.6	86.2	73.4	71.2	53.9	47.2	57.1
Cascade+V		94.8	81.0	87.7	84.2	80.4	86.3	80.7	72.4	68.3	49.1	53.7
CubiCasa	IoU	93.6	64.4	73.0	65.0	66.6	74.2	60.6	55.6	44.8	33.7	41.4
Cascade+V		87.7	71.9	77.3	74.8	75.0	80.3	70.3	65.0	58.7	44.5	44.8

Qualitative

By examining the images and recognising trends, the qualitative evaluation is based on human-assessment. Here we search for system-made structural errors and insightful takeaways. The model usually makes good predictions of the geometry and layout of the floor plan, as can be seen in Figures 5 and 6. Walls are well positioned to connect the room segmentations along with the keypoints. There are times where rooms' bounding boxes do not hit the wall and therefore stop the segmentation too early. This is something in post-processing that can be worked on.

It looks like the model is having trouble correctly marking the spaces. In particular, rooms with little substance, such as bedrooms, hallways, storage and other-class rooms. On all picture sizes, the same error occurs. This may be because classification is confined to a particular region and does not take much into account the surrounding environment.

Looking at diagonal walls, which were an important initial problem for this project to solve, the diagonal walls are not flawless. In order to determine if the wall is diagonal, the post-processing is a series of heuristic checks. This can result in either diagonal labelling of a straight wall or direct labelling of a diagonal wall. As seen in Figure 7, both cases can lead to massive imperfections in the final segmentation. The diagonal wall filling algorithm is also not sufficient, as it does not obey the endpoints well. The last point is that they appear to have wider spaces in wider floor plans where there are no rooms that can be seen in Figure 6. When using example segmentation instead of semantic segmentation, this is an issue since the model is not required to mark all pixels.

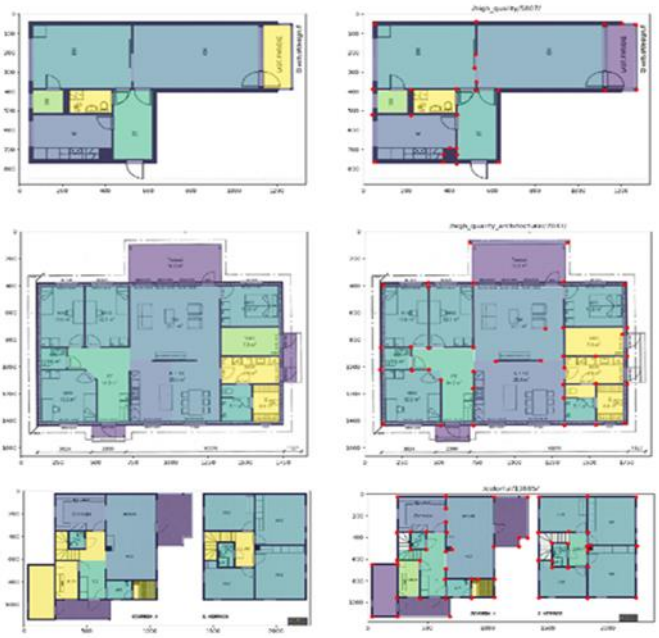

Figure 5 Smaller floorplans with ground truth to the left and inference to the right.

Figure 6 Bigger floorplans with ground truth to the left and inference to the right.

Figure 7 Examples of problems that are caused by missed walls and diagonal wall algorithm failing.

It is evident from the comparisons in Table 4 and 5 that, based on evaluation parameters, the Cascade Vertex method performs better than the baseline. For both overall performance and class performance, the ranking is higher in almost every group. Cascade Vertex does particularly well on average IoU, which may be because when finding objects in an image, the example segmentation network is more conservative. Semantic segmentation allows the device to establish a mark in the image for each pixel, which could lead to the assigning of more pixels or conflict between artefacts in some areas. Instance segmentation networks treat the entire picture as a backdrop and then pick areas to segment with the RPN. Instead of an aggregate class, this results in many separate objects. For example, walls are mostly straight, but can be combined in a number of ways. For example, it is therefore simpler for a segmentation network to locate only straight walls regardless of its surroundings.

By applying post-processing, a great improvement was also seen. Segmentations of instances are always complex, as shown before. The purpose of the post-processing phase was, therefore, to simply render walls and rooms straight and distinct. This, with some points, also improved the overall ranking. Since the post-processing is very undeveloped, if one wishes to further boost the test scores, this is very promising.

In addition, it is evident from the quantitative assessment that examples of segmentation models do a decent job of both positioning bounding boxes and segmenting floor plans. A general trend for Cascade Mask R-CNN is to have identical scores at AP50 to Mask R-CNN, yet to improve the overall AP at higher thresholds with more high-quality predictions. By looking at AP50 compared to AP75 in the tables, this is also evident in this analysis.

Looking at the comparison in Table 3, it reveals that floorplan instance segmentation is a simpler task for a CNN model to perform than the COCO-challenge. This is probably because floor plans with several different kinds of artefacts are less complicated than a noisy picture. Different floor plans have some specified features that they share with others. Design styles and quality, however, vary greatly between floor plans, making it difficult to obtain even higher scores with only segmentation of examples. Another takeaway from the same table is that the R-CNN+R101 deeper mask outperforms the R-CNN+R50 mask, which is in accordance with the COCO benchmark performance. Table 1 shows a comparison with the previous implementation of this project. This project had a model similar to the R-CNN+ResNet50 Mask, but with a much more restricted dataset. These types of models have a significant influence on training the rich and uniformed CubiCasa5k. Better results will possibly come from even more annotated floor plans. Many floor plans in the dataset are from houses, making the model very specific for home-style design prediction. However, if the model were to forecast certain types of floor plans, it would be useful to extend the available training data to also include office buildings, schools and the like.

The qualitative examination shows concerns with the framework as it is today. In post-processing, several aspects can be further enhanced to boost the overall parsing method per- type. Particularly for the diagonal walls that were the motivator for the new representation to be formed. I assume, however, that the information extracted from the segmentation of the example and the detection of the vertex is adequate for good post-processing. It just needs more work and in this project, time was not enough. Overall, the qualitative evaluation

shows good results where rooms are mostly well- segmented which is also reflected in the quantitative evaluation.

5. Conclusion

The first purpose of this study was to examine how well the task of parsing floorplans can be done by example segmentation models. In addition, if the method is feasible to use if one wants to build a completely automatic parser to be used in the real-estate economy as a tool. This project demonstrates that segmentation of instances is feasible because it outperforms the baseline model on almost all semantic metrics. Segmentations performed by an example segmentation model have been found to require additional processing to make floor plan artefacts more distinct. This sparked the idea of using a keypoint model to extract corner data to complement the segmentation of instances in a different way than was done before. The scores for mean IoU were increased by 15.2 percent overall and between 3.4-13.9 percent for individual classes by using the data generated by an instance segmentation model along with post-processing functions. This concludes that segmentation of instances along with vertex-keypoints and post-processing is better than the semantic segmentation techniques previously implemented.

On the new CubiCasa5k dataset, the second objective was to compare different instance segmentation models. It has been shown by training and evaluating various models that the models with a better AP score on the COCO benchmark also perform better on floor plans. This means that by introducing state-of-the-art models as they are published in the future, one will continue to improve the efficiency of the example segmentation model.

References:

1. G. Bradski, “The OpenCV Library,” Dr. Dobb’s Journal of Software Tools, 2000.
2. Bulat, J. Kossaifi, G. Tzimiropoulos, and M. Pantic, “Toward fast and accurate human pose estimation via soft-gated skip connections,” 2020.
3. Bulat and G. Tzimiropoulos, “Human pose estimation via convolutional part heatmap regression,” in European Conference on Computer Vision. Springer, 2016, pp. 717–732.
4. Z. Cai and N. Vasconcelos, “Cascade r-CNN: High quality object detection and instance segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, p. 1–1, 2019. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/tpami. 2019.2956516
5. D. Ciresan, U. Meier, J. Masci, and J. Schmid Huber, “Multi-column deep neural network for traffic sign classification,” Neural Networks, vol. 32, pp. 333 – 338, 2012, selected Papers from IJCNN 2011. [Online]. Tillga¨nglig: http://www.sciencedirect.com/science/article/pii/S0893608012000524
6. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun 2016. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/CVPR.2016.350
7. R. Girshick, “Fast r-CNN,” 2015 IEEE International Conference on Computer Vision (ICCV), Dec 2015. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/ICCV. 2015.169
8. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” 2014 IEEE Conference on Computer Vision and Pattern Recognition, Jun 2014. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/CVPR.2014.81
9. K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask r-CNN,” in 2017 IEEE Inter- national Conference on Computer Vision (ICCV), 2017, pp. 2980–2988.
10. A. Kalervo, J. Lyonias, M. Ha¨ikio¨, A. Karhu, and J. Kannala, “Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis,” Lecture Notes in Computer Science, p. 28–40, 2019. [Online]. Tillga¨nglig: http://dx.doi.org/10.1007/978-3-030-20205-7 3
11. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–

1105. [Online]. Tillga¨nglig: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep- convolutional-neural-networks.pdf

12. W. Li, Z. Wang, B. Yin, Q. Peng, Y. Du, T. Xiao, G. Yu, H. Lu, Y. Wei, and J. Sun, "Rethinking on multi-stage networks for human pose estimation," 2019.
13. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," Lecture Notes in Computer Science, p. 740–755, 2014. [Online]. Tillga¨nglig: http://dx.doi.org/10.1007/978-3-319-10602-1 48
14. Liu, J. Wu, P. Kohli, and Y. Furukawa, "Raster-to-vector: Revisiting floor- plan transformation," in 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2214–2222.
15. J. Long, E. Steelhammer, and T. Darrell, "Fully convolutional networks for semantic segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 4, p. 640–651, Apr 2017. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/TPAMI.2016.2572683
16. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in European conference on computer vision. Springer, 2016, pp. 483–499.
17. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in Advances in Neural Information Processing Systems 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlche´-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Tillga¨nglig: http://papers.neurips.cc/paper/9015-pytorch-an-imperative- style-high-performance-deep-learning-library.pdf
18. S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-CNN: Towards real-time object detection with region proposal networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, p. 1137–1149, Jun 2017. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/TPAMI.2016.2577031
19. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, pp. 533 – 536, 1986. [Online]. Tillga¨nglig: https://doi.org/10.1038/323533a0
20. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Backpropagation applied to handwritten zip code recognition," Neural Computation, vol. 1, pp. 541 – 551, 1989. [Online]. Tillga¨nglig: http://yann.lecun.com/exdb/publis/pdf/lecun-89e.pdf
21. Anusuya Ramasamy, Joseph Wondwosen," Deep Learning Based Ethiopian Car's License Plate Detection and Recognition" International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8 Issue-6, March 2020. DOI: 10.35940/ijrte.F9857.038620.
22. Samuel, "Some studies in machine learning using the game of checkers. ii—recent progress," Annual Review in Automatic Programming, vol. 6, pp. 1 – 36, 1969. [Online]. Tillga¨nglig: http://www.sciencedirect.com/science/article/pii/ 0066413869900044
23. Sandelin, Semantic and Instance Segmentation of Room Features in Floor Plans using Mask R-CNN. Uppsala: UPTEC IT, 2019. [Online]. Tillga¨nglig: urn:nbn:se:uu:diva-393348
24. K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," 2019 IEEE/CVF Conference on Computer Vision and Pattern

Recognition (CVPR), Jun 2019. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/CVPR.2019.00584

25. H. Touvron, A. Vedaldi, M. Douze, and H. Jegou, "Fixing the train-test resolution discrepancy: Fixefficientnet," 2020.
26. J. R. Arunkumar, Tagele Berihun Mengist," Developing Ethiopian Yirgacheffe Coffee Grading Model using a Deep Learning Classifier" International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-9 Issue-4, February 2020. DOI: 10.35940/ijitee.D1823.029420.
27. J. Uijlings, K. Sande, T. Gevers, and A. Smeulders, "Selective search for object recognition," International Journal of Computer Vision, vol. 104, pp. 154–171, 09 2013.
28. P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, ˙I. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," Nature Methods, vol. 17, pp. 261–272, 2020.
29. S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun 2016. [Online]. Tillga¨nglig: http://dx.doi.org/10.1109/CVPR.2016. 511
30. Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.
31. Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," 2019.
32. H. Zhang, C. Wu, Z. Zhang, Y. Zhu, Z. Zhang, H. Lin, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, and A. Smola, "Resnest: Split-attention networks," 2020.
33. Anusuya Ramasamy, M. Sundar Rajan, J.R. Arunkumar, "A 30 Mb Re-Configurable Convolutional Neural Network Processor for High-Performance and Energy-Efficient Operation", Materials Today: Proceedings, 2020, DOI: 10.1016/j.matpr.2020.10.288.