



Deep

Architecture

Machine learning and the future of architecture

Deep Architecture

Machine learning and the future of architecture



CHALMERS

Examiner: Kengo Skorick
Tutor: Jonas Lundberg

Fabian Sellberg
Master Thesis
Chalmers school of architecture
Department of Architecture and civil Engineering
Architecture and Urban design, MPARK
Spring 2020

Table of contents

Page	
4	Abstract
5	Student Background
6	Introduction
12	Theory
20	Design tools
32	Experiments
56	Interviews
64	Conclusion
70	Codebase
74	Bibliography

Introduction

Abstract

This thesis explores the use and potential application of machine learning in the field of architecture. It aims to provide a framework for continued research as well as to explore and advance neural networks use in practical and conceptual stages of design processes. This paper will go in depth on a select few emerging systems that will be trained and applied on architectural drawings.

Machine learning is a rather old concept derived from mathematics and statistical methods. But due to new technological advances in processing and computing this technology is now more readily available and can be processed

on most home computers this provides the base of the thesis.

As a starting point unsupervised learning will be explored which means that the human input is kept low. Examples from supervised and reinforcement learning will also be touched upon and tested within an architectural framework.

When a sample of systems have been developed a series of interviews with architects investigated in generative design and automation is carried out. These interviews both test the methods developed and provide insight on the thoughts on machine learning

by architectural practitioners.

These interviews provide the base of discussion material where I look at how machine learning affects the profession today and what is needed from the technology to be able to integrate itself into the architectural design process.

The focus of this thesis is thus on the process and method of the creation of these systems and not in a typical end result. The goal of the project is that the systems will generate relevant and inspiring results. From the results a discussion will form on machine learning and the future of architecture.

Introduction

Student Background

Fabian Sellberg

Chalmers University of Technology

Material & Turn

Education

Chalmers - Architecture and Urban Design

2018 - present

Umeå School of Architecture - Bachelor of Fine arts

2013 - 2016

Employment

PS arkitektur - Architect

2017 - 2018 Stockholm

Axelot - Intern

2016 - 2017 Stockholm

Vizzit - UX Designer/Back end Developer

2011 - 2013

Introduction

Background & Context

We are living in a time where computers we use everyday can do heavy computational tasks able to match super computer of the early 2000s. Nearly every two years the amount of transistors in a processor doubles and has been since the 1970s. This revolution in computational power is something most fields have already started to exploit.

In agricultural science a harvester can learn how a ripe tomato looks and make it harvest only those, in the medical industry they reconstruct 3d replicas from X-ray photos of organs and injuries. But in architecture this abundance of computational power is

used mainly as an assistance for manual drawing. Architecture is known for its slow progress and adaptation of new technology. Even though neural networks are already starting to be used within some fields such as city planning. Where there is a greater need to simulate and quantify agents to greater degree.

But using machine learning as a tool in an architectural process is quite unexplored. There are quite a bit of scholarly projects done on the subject but most of them are from computer science majors and not with a pure architectural focus. So instead of only looking at the technical part on how these

systems are built I have the opportunity to really start to adapt these already created systems and see how they can be adapted to an architectural framework.

As for context the project will not be manifested as a building but rather as a repetition of a system. Since a dataset of existing drawings is the base of the project, the general context of the dataset will be transferred to the system. Thus creating a digital pseudo context, this will be interesting to study and see exactly what conclusions can be drawn from this generalization of context.

Introduction

Research Questions & Delimitations

How can the architectural profession adapt machine learning technologies to its workflow in both early and later stages.

As more professions adapt to becoming increasingly digitized the same is happening to architecture, how can architectural practice tackle this evolution. This thesis aims to explore this and discuss how the future of architecture could be shaped by the technologies emerging today.

Is machine learning able to produce coherent and working drawings while being able to adapt to shifting boundaries.

While machine learning is easily applicable to a multitude of architectural scenarios such as facade design and space organization are these results relevant to the profession. All of the experiments in the thesis work towards this question. While there are no direct objective criteria specified to reach these results it still remains as a goal.

What are the views from practicing architects on how machine learning might change the profession in the future?

Architecture and especially the building industry is one of the slower industries to adapt new technology. A series of interviews will be carried out with both progressive architects and traditional architects that are in different ways involved in the building industry. From this a discussion will form on how architects can adapt and how the building industry will have to adapt as well.

Introduction

Method

The main method of this thesis is to develop a series of Generative adversarial networks (GANs) which are created in the project. From this system the design is generated, this program is then trained and refined through the project in incremental steps. A largely predictable method where the progress will be incremental and definable.

There are two main methods of coding a GAN mainly C-sharp (C#) or Python. Both have clearly defined libraries that can help in the programming, Python is the more popular approach and will be the one used in this thesis. The most used libraries are pyTorch

and Tensorflow, Tensorflow is the one used the most in this thesis but pyTorch is used in some parts where the functions contain in that shell suited better.

Many of the projects working with neural networks share their code and work strictly with open source material so there is a stable ground to build upon. So all the progress that already has been made in the field can easily be adapted and integrated. For processing this large amount of information a common home computer is used to show how this technology is open and can be used by almost everyone with a computer at home.

There is also a series of qualitative interviews with architectural practitioners where a discourse for further discussion will form. This discussion will regard both the present situation in architecture and the future and how machine learning might change the profession and its design processes.

Introduction

Reading instructions

Introduction Introduction to machine learning and the purpose of this thesis.

Theory Short discourse on machine learning and the systems used in the thesis.

Design tools Tools developed to aid in discussion on machine learning and architecture.

Experiments Selected experiments on machine learning in an architectural framework.

Interviews Interviews with architects regarding machine learning and digital processes in architecture.

Conclusion Conclusion and discussion around the subjects brought up in thesis.

Codebase & Bibliography Links to codebase and references used through the project.

Theory

Machine Learning

When looking at machine learning and neural networks it can be classified into three different categories, supervised learning, unsupervised learning and reinforcement learning. These differ in how they are trained to process and learn from a given dataset in figure 1 you can see examples of uses for these networks.

Supervised learning is tasked to learn a function or relationship

between an input and an output based on predefined input-output pairs. This method is the one we will use mostly in the studies below, as it provides the best framework for creative/generative output.

Unsupervised learning is in a way the opposite of supervised as the network will look for functions and relationships between a dataset with no predefined labels.

This method is very flexible and doesn't require much input from the user but instead requires a large amount of data.

Reinforcement learning works by giving the network a task or objective and a reward system. When the network is executing these tasks it is given rewards when achieving the objective that was defined thus reinforcing the patterns taken by the network.

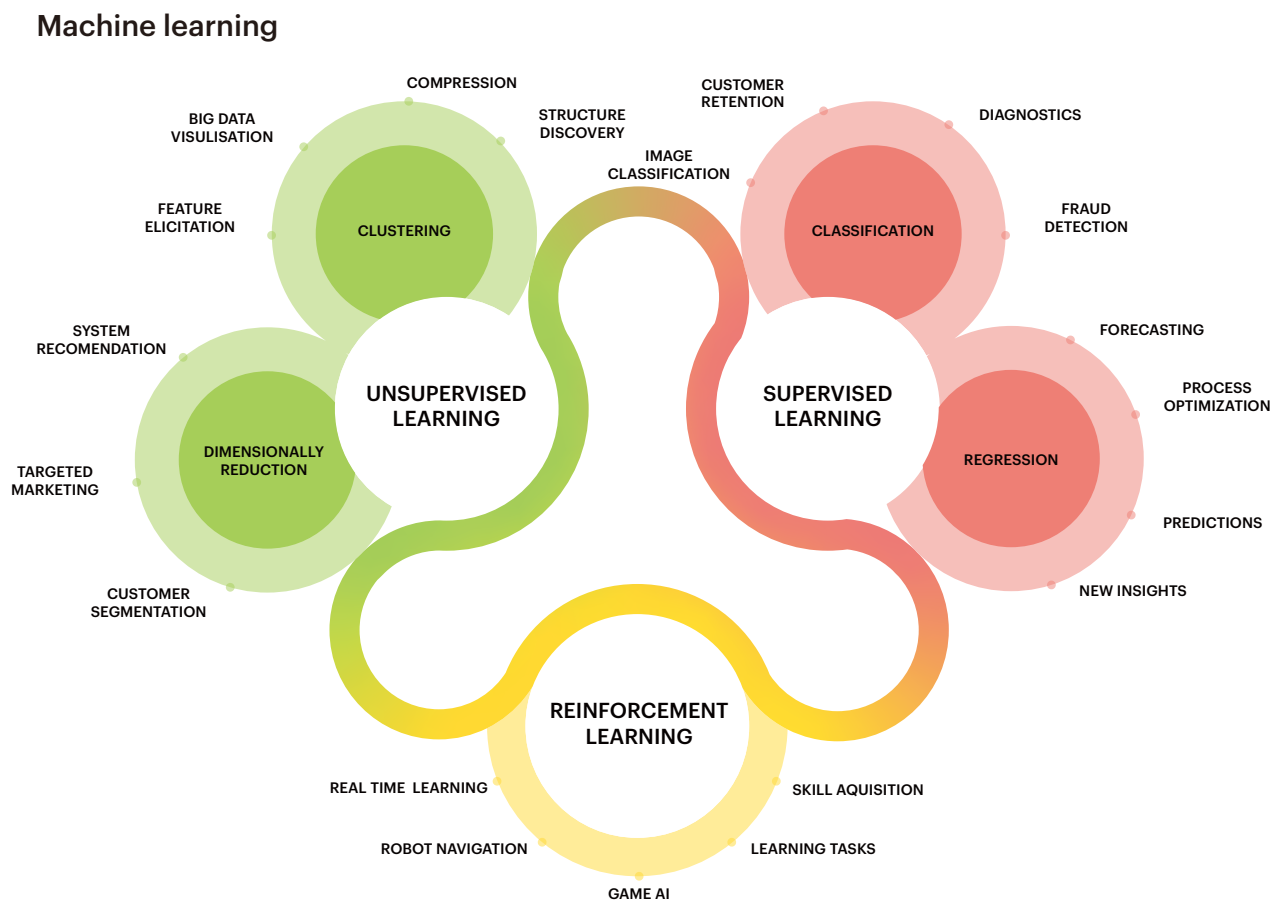


Figure 1

So when deciding what method of machine learning that is the most suitable for this thesis we need to look at two things. First and foremost is what task do we want it to perform, for this thesis we want it to generate architectural drawings. The second parameter is what kind of data is available and how can it be modified.

Architectural drawings have very

clear principles on how they work so it can easily be analyzed and processed. This enables us to work with both unsupervised- (non labeled data) and supervised- (labeled data) learning.

In figure 2 we see how an Generative adversarial network works, this method is great for generating many variations of a dataset. The network consists of two parts, a generator and a discriminator

which work against each other. The generator creates new images that are then fed to the discriminator which tries to distinguish between the output given to it and data from the given dataset. Both then get data from each other on how well they performed and improve. This process is then looped for a set amount of 'epochs' which define how many times this loop will run.

Generative adversarial net (GAN)

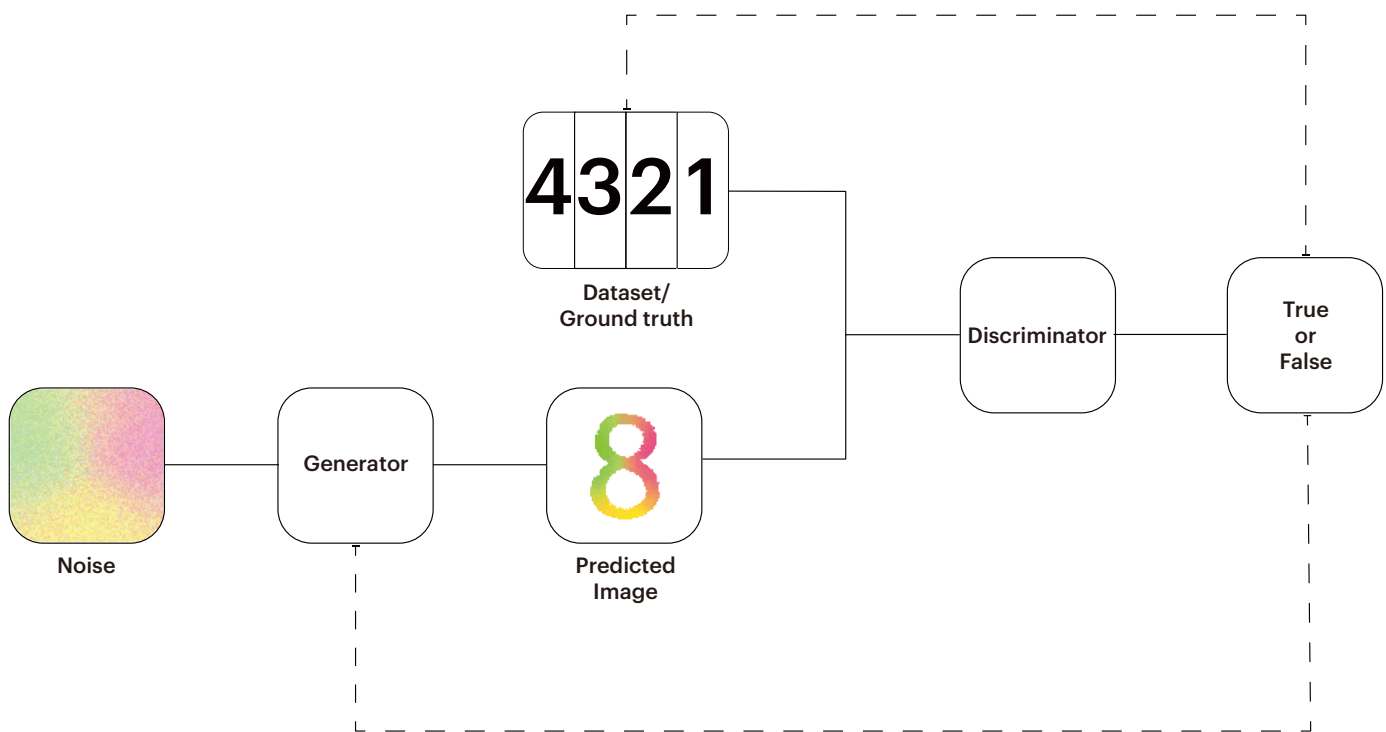


Figure 2

Theory

Convolutional neural network (CNN)

Style transfer is a form of convolution neural networks that works by using three main images: first a content image, then a style image and finally a input image. It then tries to preserve the content of the content image while the style of the style image is applied upon it. The results from this are often nonsensical but works in a nice way as an introduction to how machine learning manifests itself in image generation.

Leon A. Gatys outlined this process in his paper (Gatys et al., 2016) which goes into detail explaining the technical aspects and the intent when creating the network. It's main purpose was to define how the human mind experiences art and what defines an artistic style.

As a process it is very fast since it uses a low dataset of images (most often three) and therefore

the breakdown of the content and style of the images can be processed and applied in only a few generations and epochs.

In the tests made here I gave the network an content image in the form of a pine-cone and a style in the form of a plan-drawing. We use the same content image as the input image (the input is the image that is modified). The output is not something that can

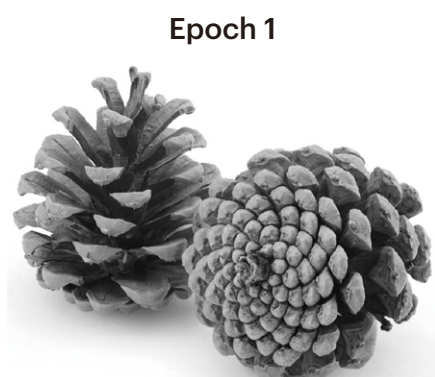


Figure 1

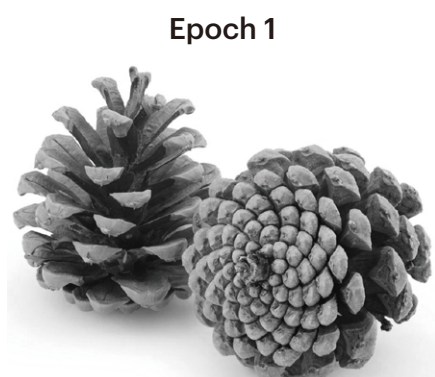
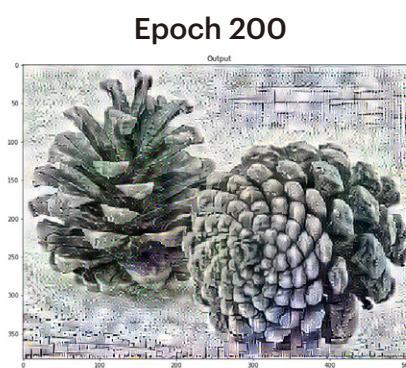
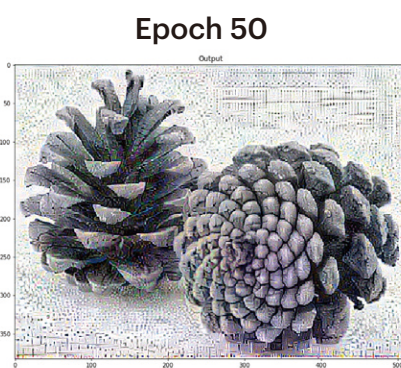
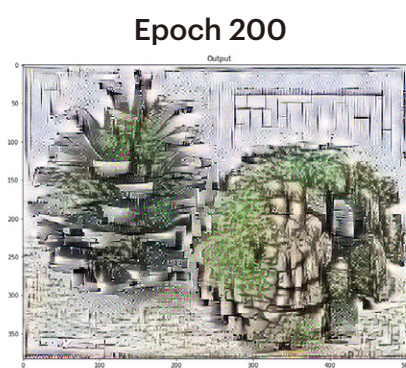
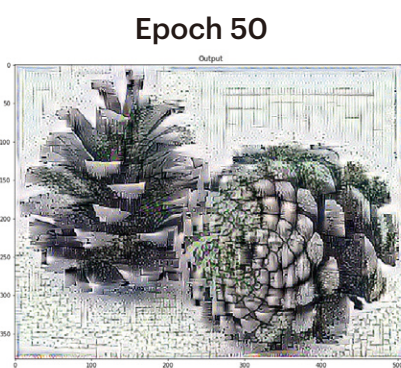


Figure 2

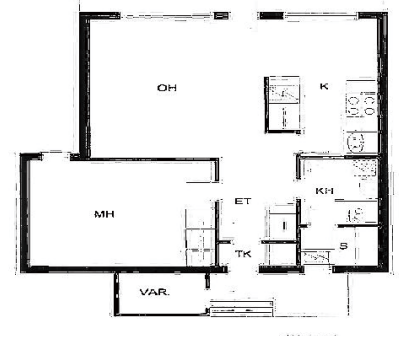


be directly used but provides a nice insight to how these processes are applied in a very concrete example.

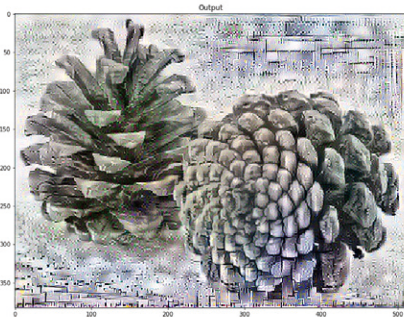
Base image (content)



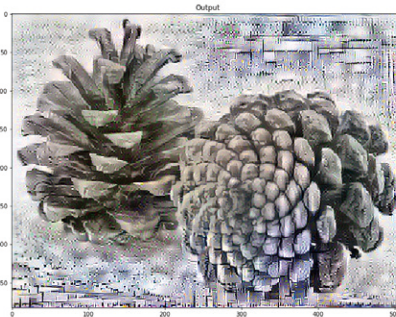
Style image



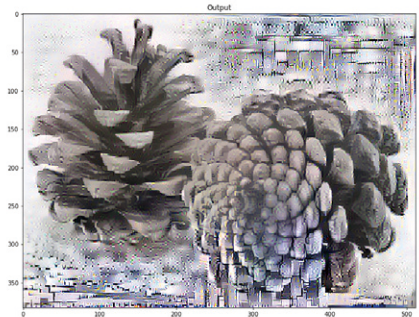
Epoch 350



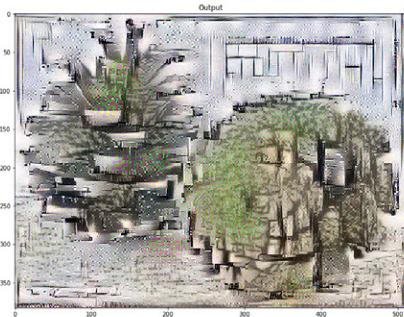
Epoch 500



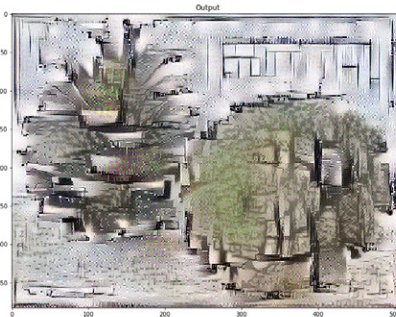
Epoch 1500



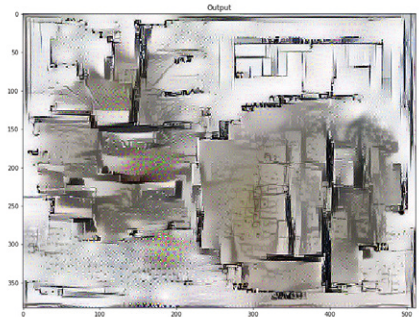
Epoch 350



Epoch 500



Epoch 1500



Theory

Generative adversarial networks (GAN)

Generative adversarial networks were first defined in 2014 (Goodfellow et al. 2014) and is a system designed to produce or identify believable results. It does this by using two adversarial networks that both work against and with each other.

As stated in the name the first part is a generator that produces material for the second part, the discriminator to identify if it's real

or not. To be able to distinguish between real and not the discriminator needs to be trained on a dataset consisting in the example to the right of handwritten digits. While the discriminator is learning to distinguish the patterns of the digits the generator is starting to produce, in the case of images this starts out as noise.

The produced noise is then sent to the discriminator in training,

the discriminator is now starting to form labels of patterns found in the dataset it is given. So when the discriminator is handed the generated output it tries to define it and put a label on it, in the example these are digits so the labels would probably be the numbers ranging from one to ten.

When the discriminator is done with this process of labeling it responds to the generator with

1. $256 \times 256 \times 3$
2. $64 \times 64 \times 128$
3. $32 \times 32 \times 256$
4. $16 \times 16 \times 512$
5. $8 \times 8 \times 512$
6. $4 \times 4 \times 512$
7. $2 \times 2 \times 512$
8. $1 \times 1 \times 512$

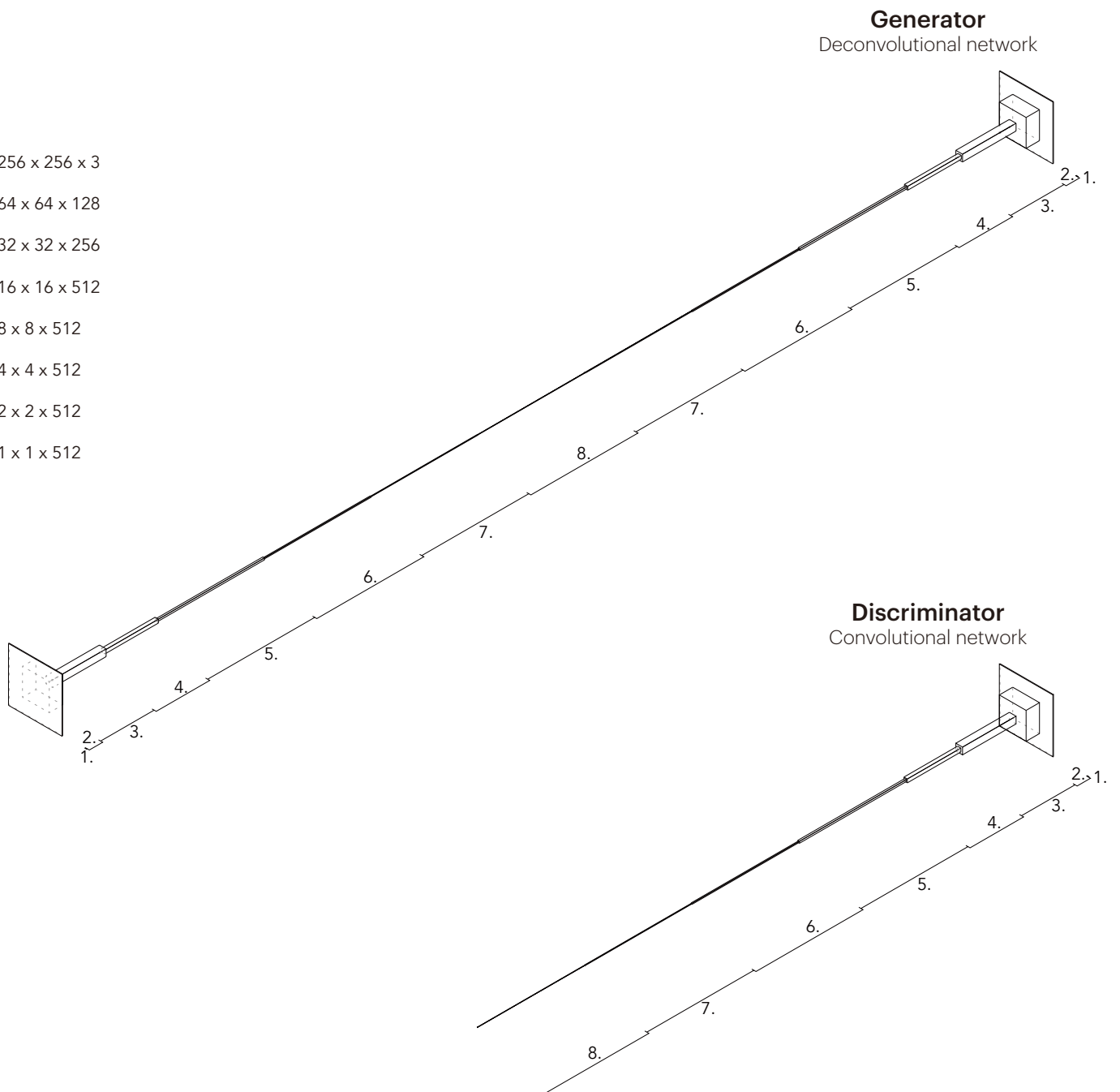


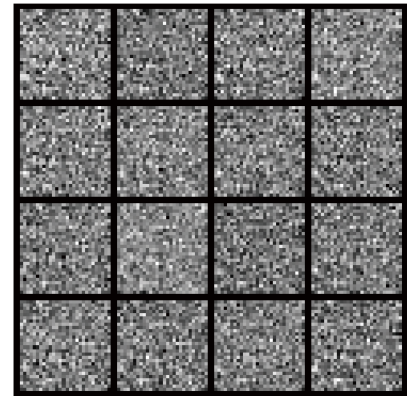
Figure 1
14

if the output could be from the original dataset or if it was generated by the generator. The generator then listens to this and tries to improve its generation to fool the discriminator that the generated output from the network is real.

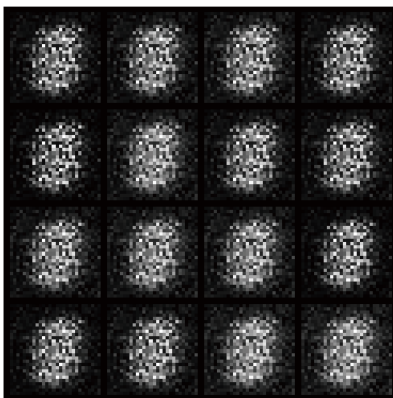
So in short the system works with a zero sum game where the generator tries to fool the discriminator by making a good enough

output. In the end this process often produces output not directly identifiable as computer generated by either computer or human.

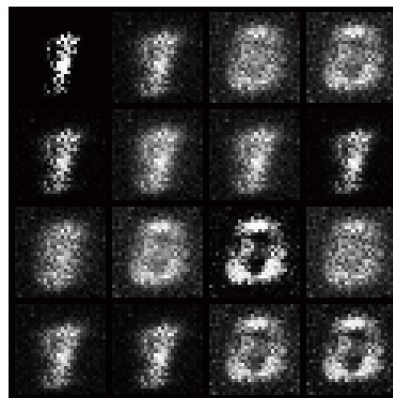
Epoch1



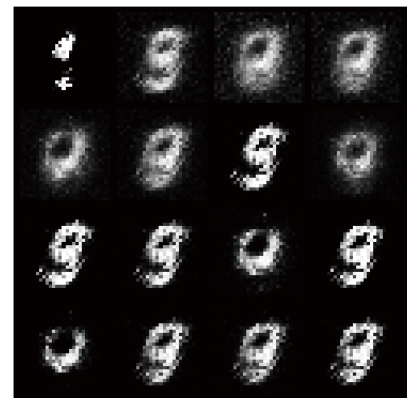
Epoch 3



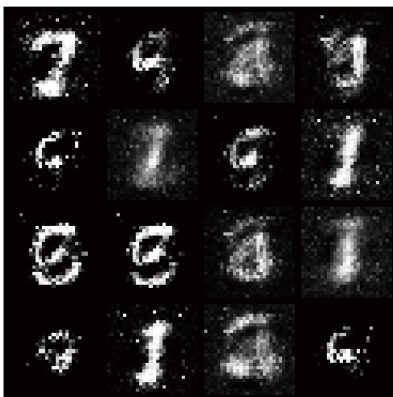
Epoch 6



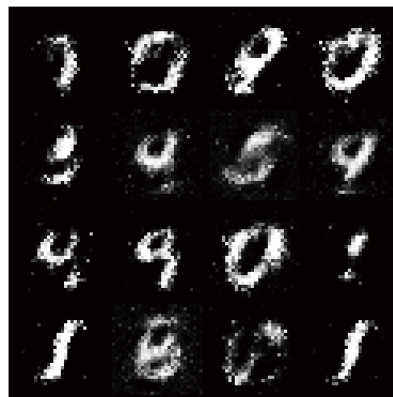
Epoch 9



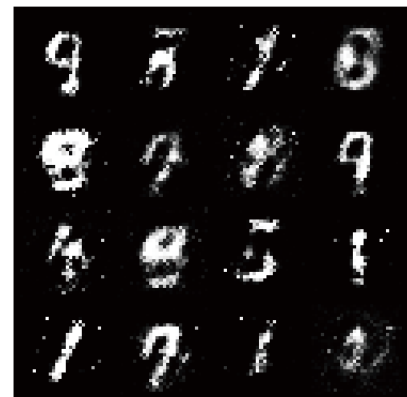
Epoch 12



Epoch 15



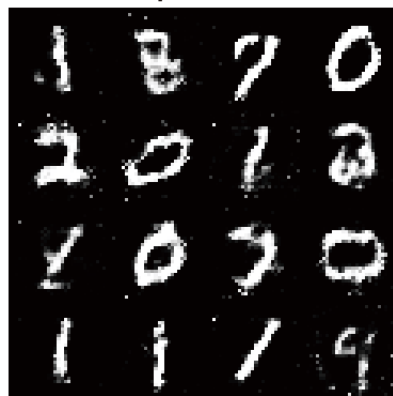
Epoch 18



Epoch 30



Epoch 40



Epoch 200

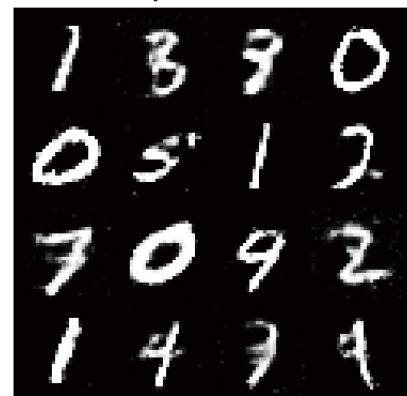


Figure 2

Theory

ConditionalGAN (cGAN)

When exploring the concepts of a GAN it's easy to see the limitations of unsupervised learning, unsupervised meaning that no human input except the definition of the dataset is given to the network. Giving the network free reins of the generation where it tries to replicate what is given in the dataset.

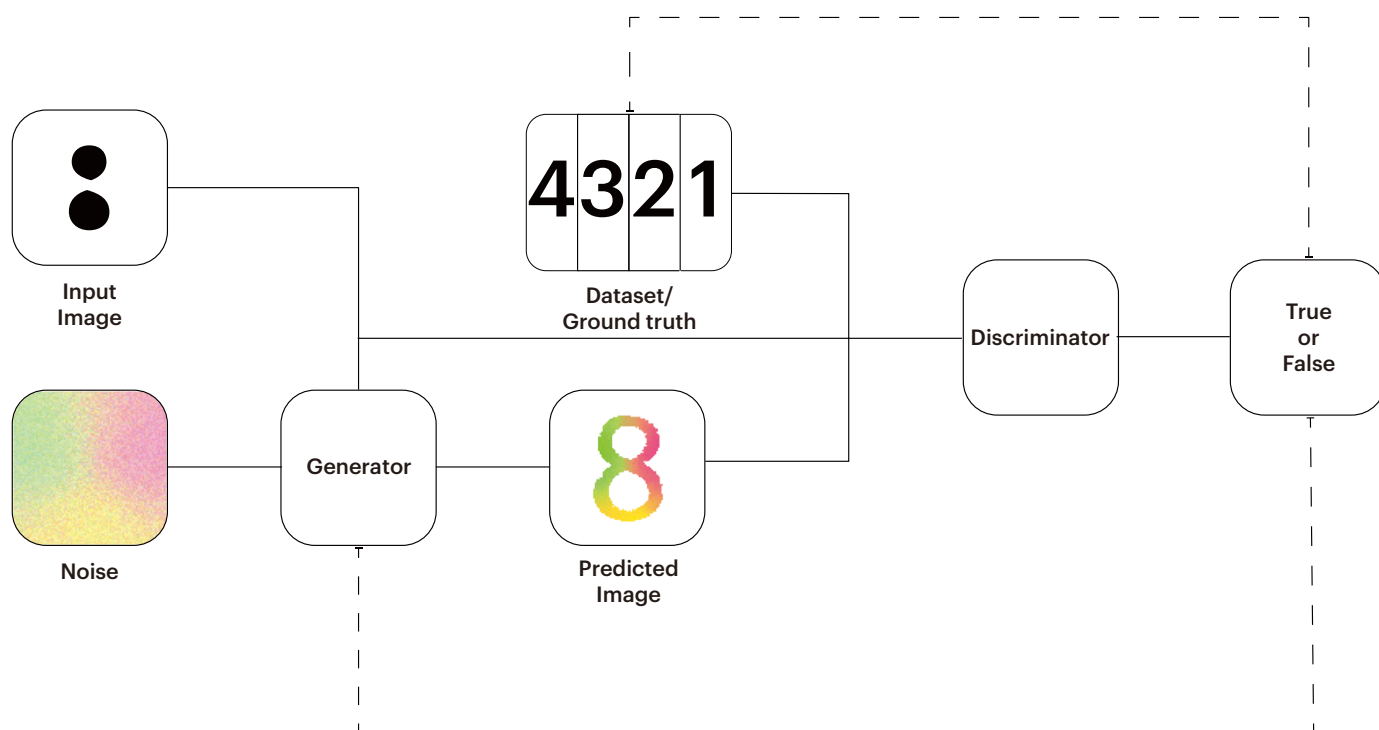
Unsupervised learning is very good for labeling and gener-

ating new ideas based on data but lacks flexibility when trying to adapt the output to the real world. A logical approach is then to move towards Conditional generative adversarial networks (cGANs). in its core It's the same thing but that instead of just providing a dataset that it iterates through and starts to segment into different labels we give it a sort of cheat-sheet to start with where labels are given towards

elements that should be identified these cheat-sheets are called input image and provide the base of generation and discrimination.

This can be described through a quote from the creators of pix-2pix, one of the most commonly used cGANs. "Just as a concept may be expressed in either English or French, a scene may be rendered as an RGB image, a gradient field, an edge map, a

Conditional generative adversarial net (cGAN)



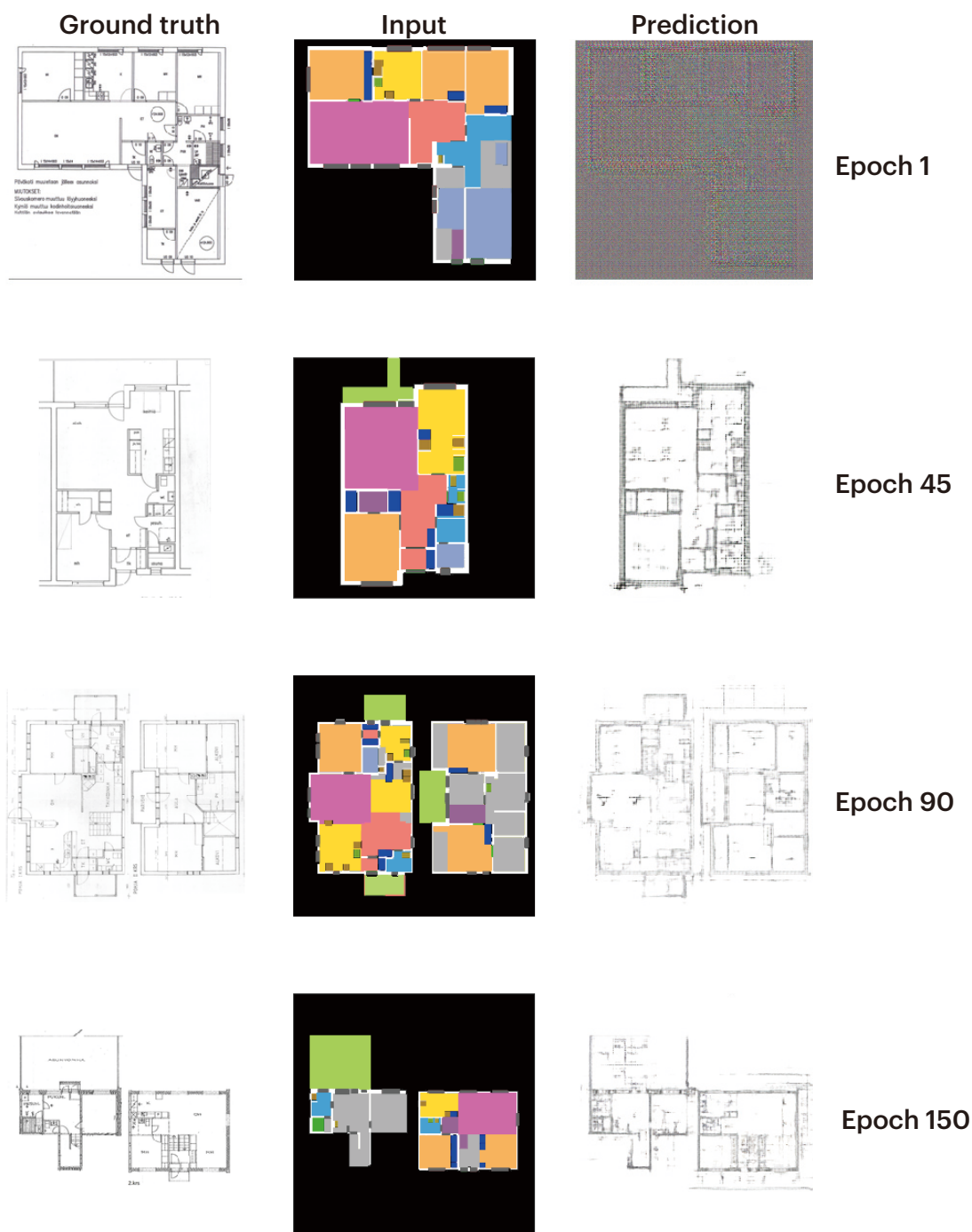
semantic label map, etc. (Isola, P et al. 2017)“ What this is saying is that instead of just providing the image a map of labels is given to the network that it also has to take into consideration when generating the output.

So instead of having the generator starting from scratch it now has a series of rules it has to follow. This can be seen in figure 2 where entrances are labeled in

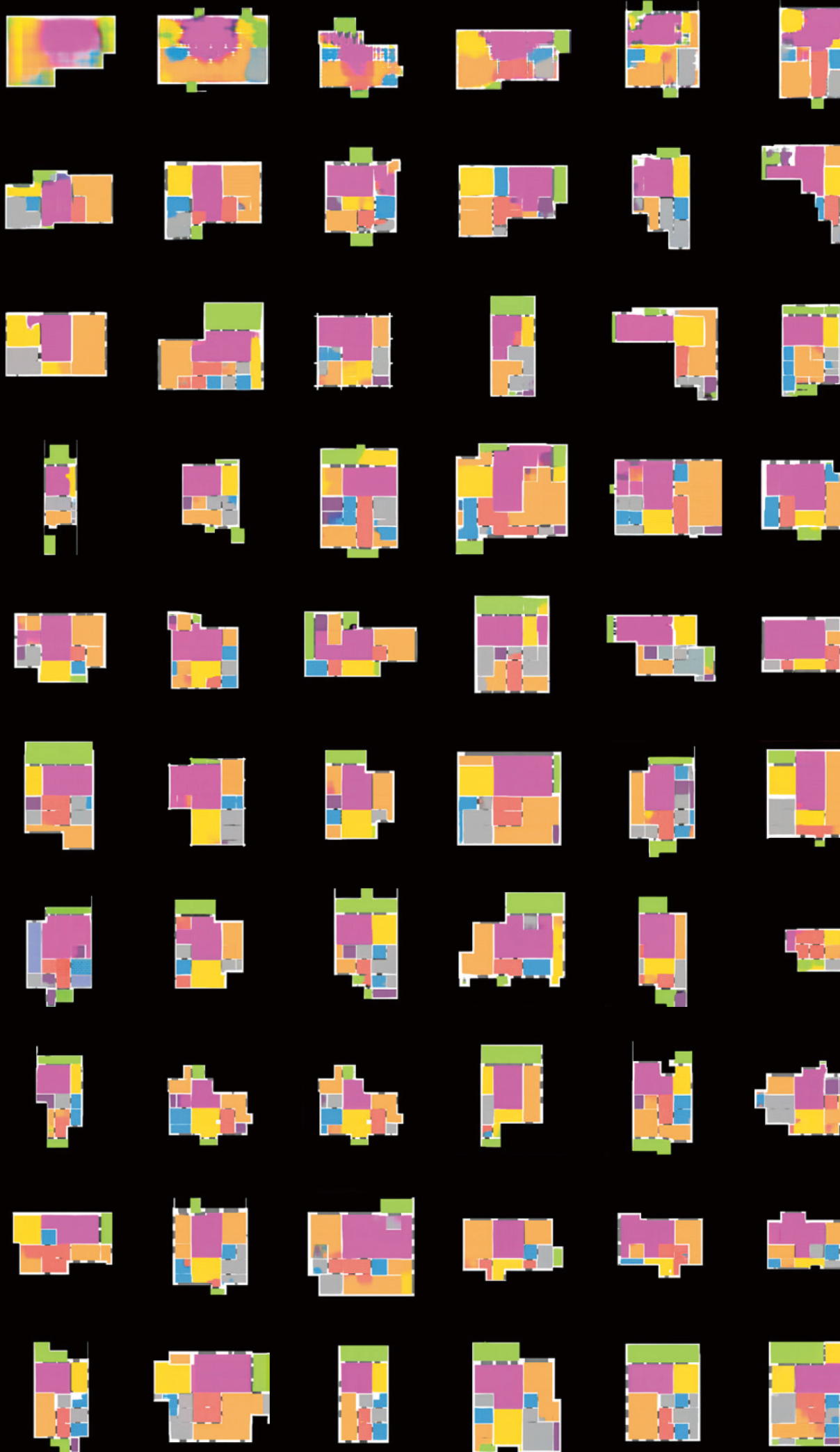
the red color on the floor-plans and bedrooms in the orange hue.

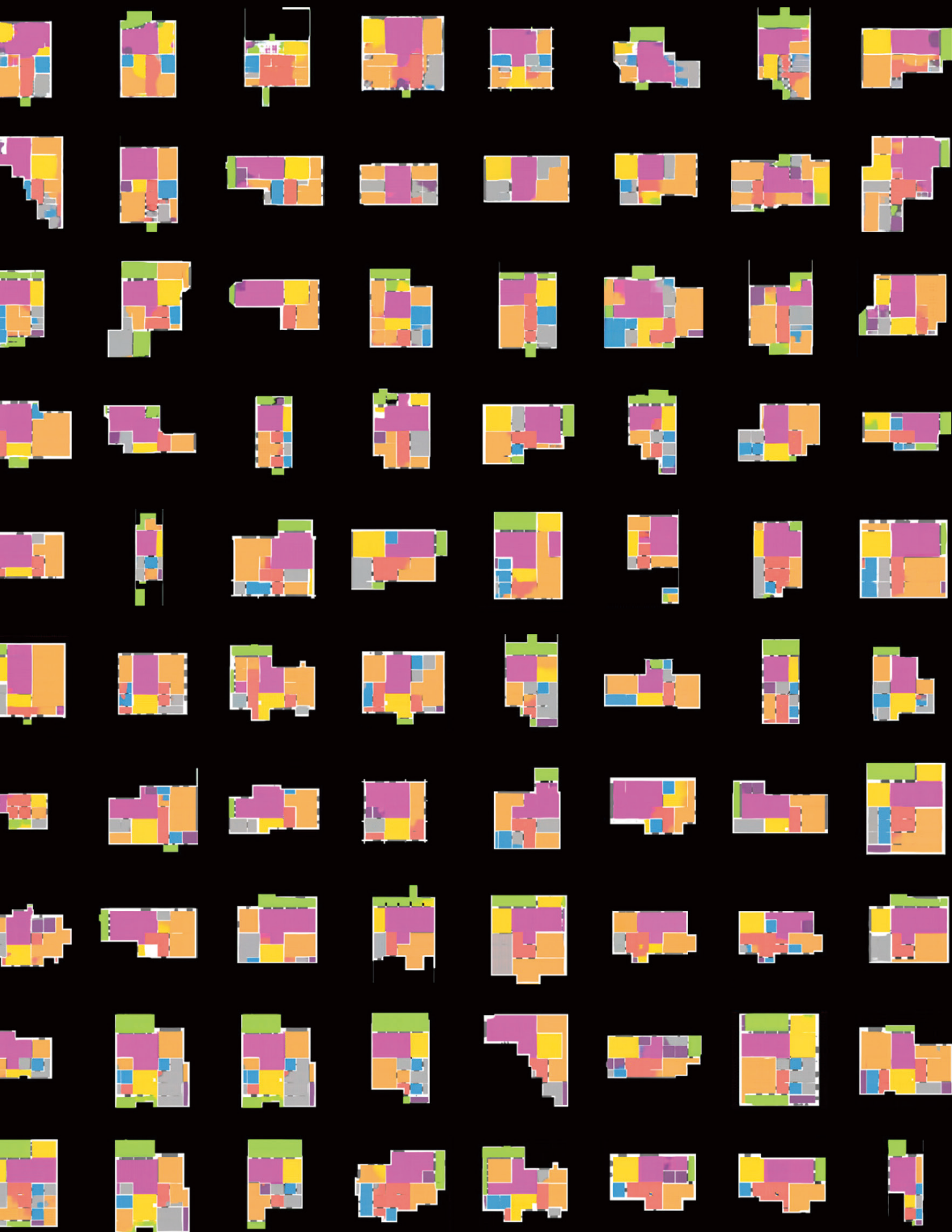
Exactly as the Generative adversarial network explained earlier the two parts of the network tries to outsmart the other and the generator will learn how to produce a floor-plan that the discriminator will think is real. This provides a large control over the output while still forcing the network to provide new novel itera-

tions of them. This technique is very flexible and can provide generative output on multiple sources of information based on the pairs of inputs and ground truths.



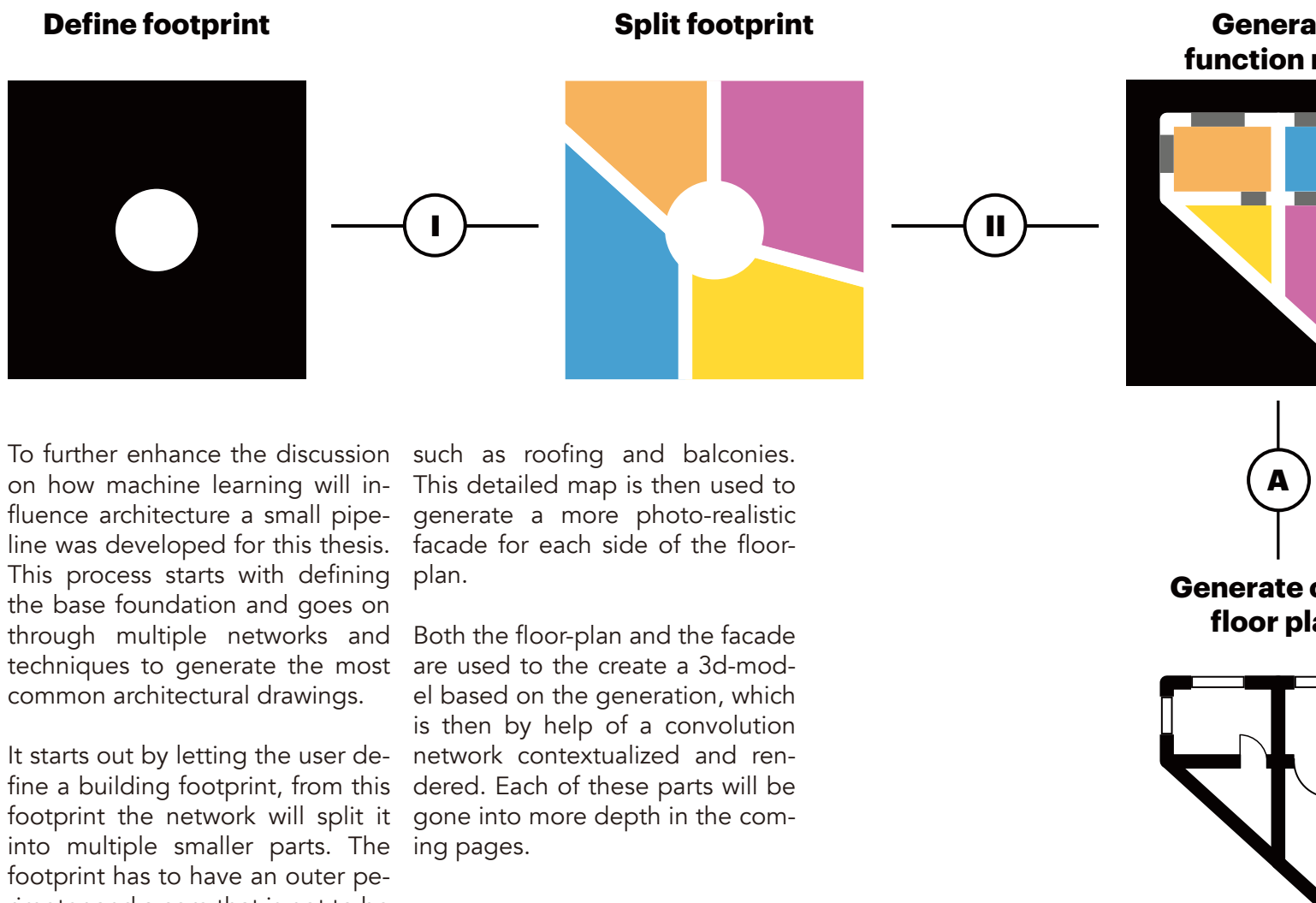
Design tools





Design tools

Pipeline



To further enhance the discussion on how machine learning will influence architecture a small pipeline was developed for this thesis. This process starts with defining the base foundation and goes on through multiple networks and techniques to generate the most common architectural drawings.

It starts out by letting the user define a building footprint, from this footprint the network will split it into multiple smaller parts. The footprint has to have an outer perimeter and a core that is not to be split defined by the user.

After the network has split the footprint into either a set amount of parts defined by the user or a randomized amount it takes each part and generates a function map in it. The function map works as the main organizational tool for the network as it defines not only room functions but also edges and relations between the functions. From this function map a floor-plan in the style which architects are more used to seeing is generated.

From these two drawings a facade label map is created, this defines not only placement of windows but height of the building as well. This base facade is then detailed by the network adding elements

such as roofing and balconies. This detailed map is then used to generate a more photo-realistic facade for each side of the floor-plan.

Both the floor-plan and the facade are used to create a 3d-model based on the generation, which is then by help of a convolution network contextualized and rendered. Each of these parts will be gone into more depth in the coming pages.

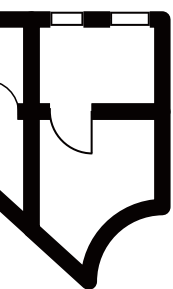
Figure 1

Generate
segmentation map

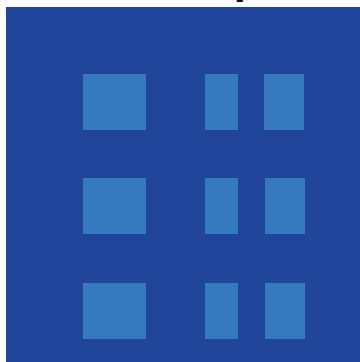


A

Generate clean
floor plan



Generate facade
label maps



A

Detail facade

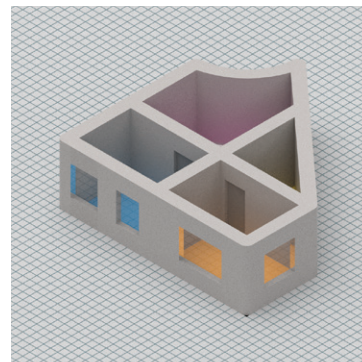


B

Generate realistic
facade

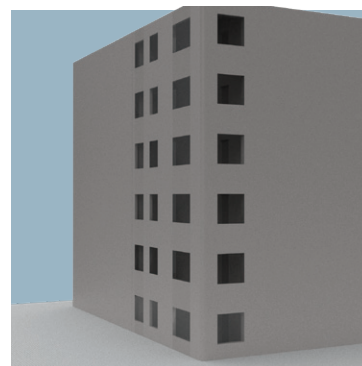


Convert floorplan to
3d-model



A

Sculpt into volume



B

Contextualize &
render



Design tools

Footprint2mask

As a first step in the design process proposed in this thesis is footprint2mask, this tool converts a footprint that is defined as a masked area into multiple masks that can be used by the rest of the network. Some requirements are set for the footprint, a closed border needs to be defined and a core in also needs to be present in the masked area.

This process is not based on any neural networks but instead completely based on image analysis scripts. The amount of split parts is either defined by user input or by random from the script. The script starts all the lines to split on based from the defined core of the footprint and then works it way out to the edges of the area. This can be manipulated by adding predefined areas made by a user or network, so that the script

has some parameters that it has to follow.

When the footprint is broken up into parts it now separates each split part into individual masks as can be seen in figure 1. These masks are then later serialized into a black and white mask so that they are easier for later scripts to process.

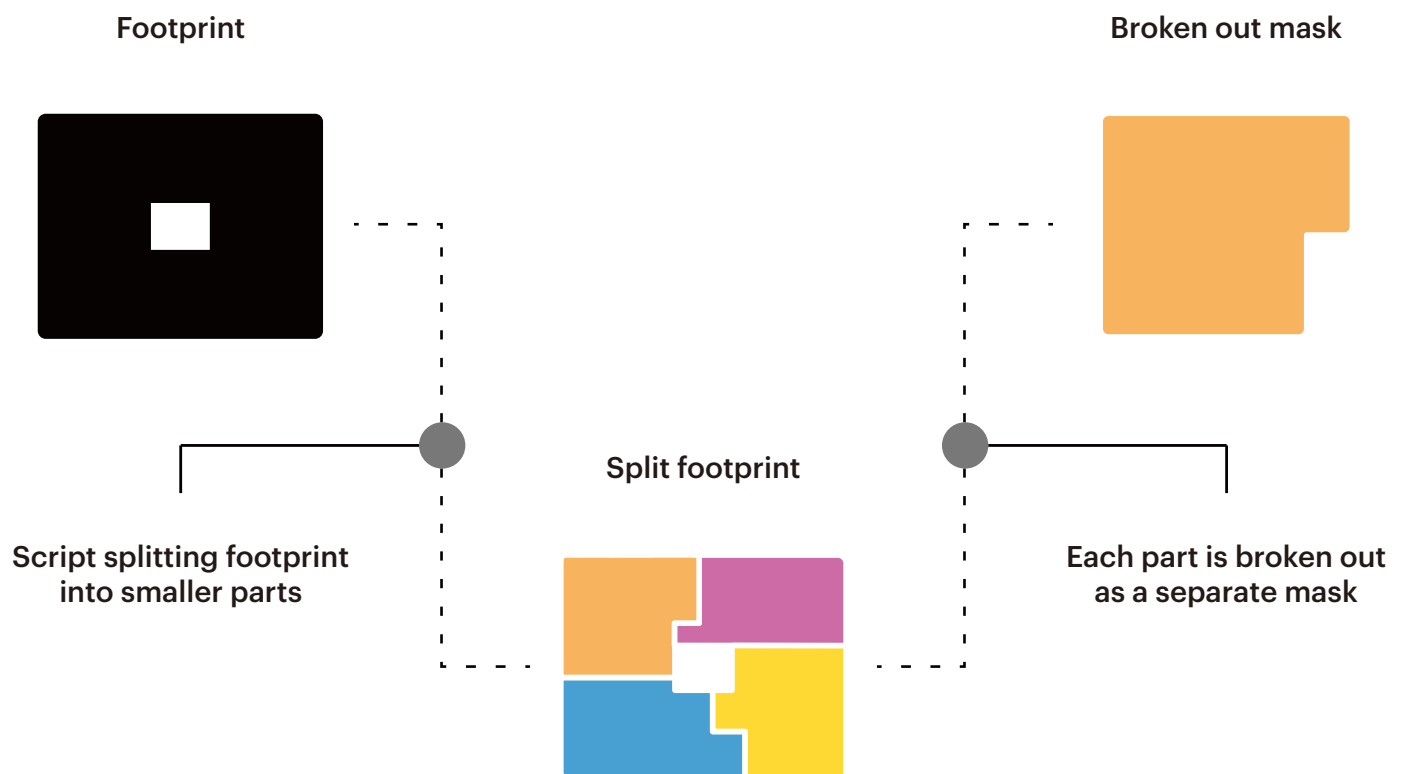


Figure 1

An important part to note is that this process is not based on neural networks but on defined scripts that make the separation based on parameters from the user and the script. When developing this process a first approach was to shoehorn in neural networks in all parts, this process could easily be done by a network but provided the simplicity of the task one is not needed.

When deciding on what processes machine learning could be applied a decision on how complex the tasks are and how precise the output from the system had to be. This issue was discussed in interviews with Anders Neregård (Sweco) and with André Agi (Link).

If an architectural design process is simplified into a linear process where it starts as an idea and ends in a completed and built project. There is an increasing set of parameters the further along the process you travel, more regulations and requirements are given when move along that line. A big decision for this thesis has then been where in the architectural process to position my tools.

A rule based script as used in these examples would fit with a high parameter filled approach so that would position the project in a later stage of the design process. Since machine learning work best with analyzing relations and underlying systems there is a slight clash in where to position it.

Even though a tool for later stages in an design process would make sense for a similar process a decision was made to position myself early in the process looking at how to develop tools that position themselves in the early more creative stages.

input

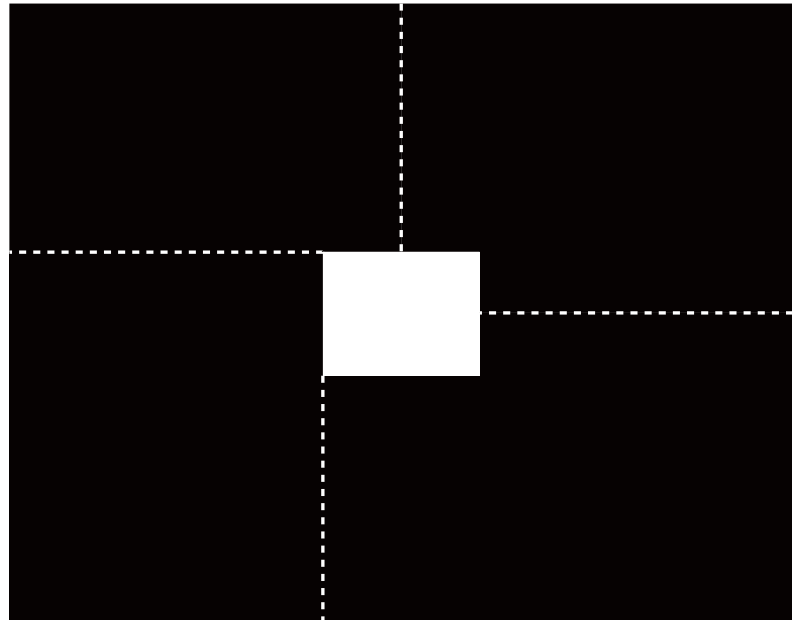


Figure 2

Output



Figure 3

Design tools

Mask2plan

Mask2plan works through multiple networks to produce a drawing from a shape defined by the user. Figure 1 shows this process, the first step is defining the mask followed by a network trained on generating function maps from masks. When the function map is generated another network takes over, this network is trained on the relationship between a function map and a plan drawing.

Resulting in a plan drawing with defined room functions in a fashion we are used to read it. These three steps work well together and show how multiple networks can generate material for the other and thus creating a multi-dimensional network. This is common practice within creative networks as it often deals with complex systems and patterns but when broken down can be easily managed by a network.

As this serves as the main organizational tool for the over-arching design process this system can also be broken out and act as a quick generative tool. The mask does not need to be defined by the network as it is in this design process but could be defined by user input. This process provides a good and clear output that both humans and networks can analyze and interpret.

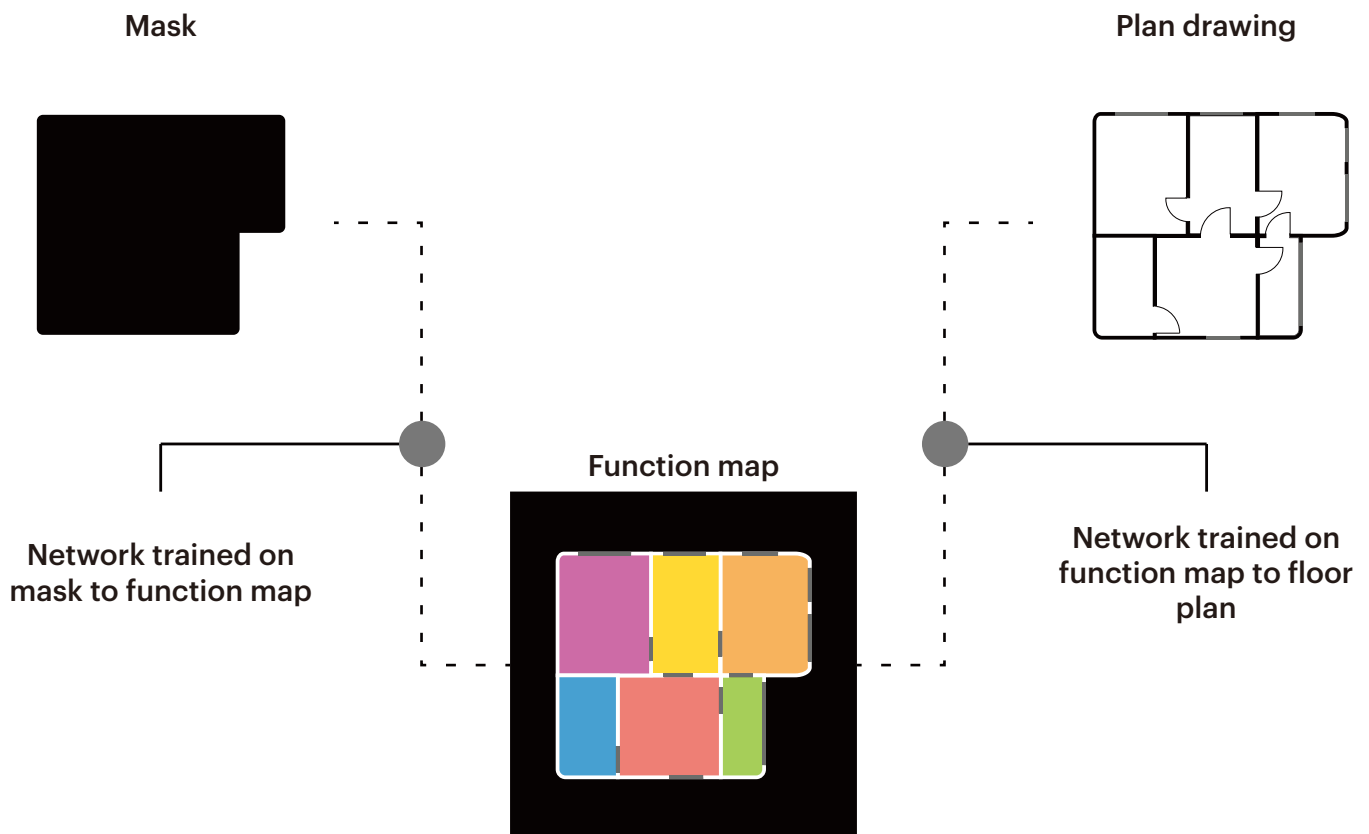


Figure 1

Each function-map is generated based on the mask which contain as little information as needed. Since the network is trained on the same dataset it's easy for it to start repeating patterns as there are no specific details that distinguish the masks from each-other more than shape and size.

An example of this can be seen in figure 3 where in the top right plan and bottom right plan a small grid of rectangular rooms can be seen which are very similar in size and fashion. This is one of the main problems with machine learning as a whole as they think they find the optimal solution to a problem and then try to apply this solution to all applicable parts of the process.

This issue was discussed in the interview with Ola Delsson (White Architecture) will automation lead to more standardization by tools that repeat a pattern learned by the system. Many studies have been made on how networks can be optimized not to get stuck repeating the same patterns (Khandelwal, 2019).

This is a inherent problem in the technology but work is being made on how to solve these issue. A practical approach would be to add more parameters to the input for example window placement, by adding more details the network will have to take into regards this information and shape the predictions on that, still the same information will lead to the same prediction. Another path to take in this is to add randomness to the system this can be done on either the inputs given to the network or adding randomization on the base level in the optimization of the network.

input



Figure 2

Output

Function map

Plan drawing

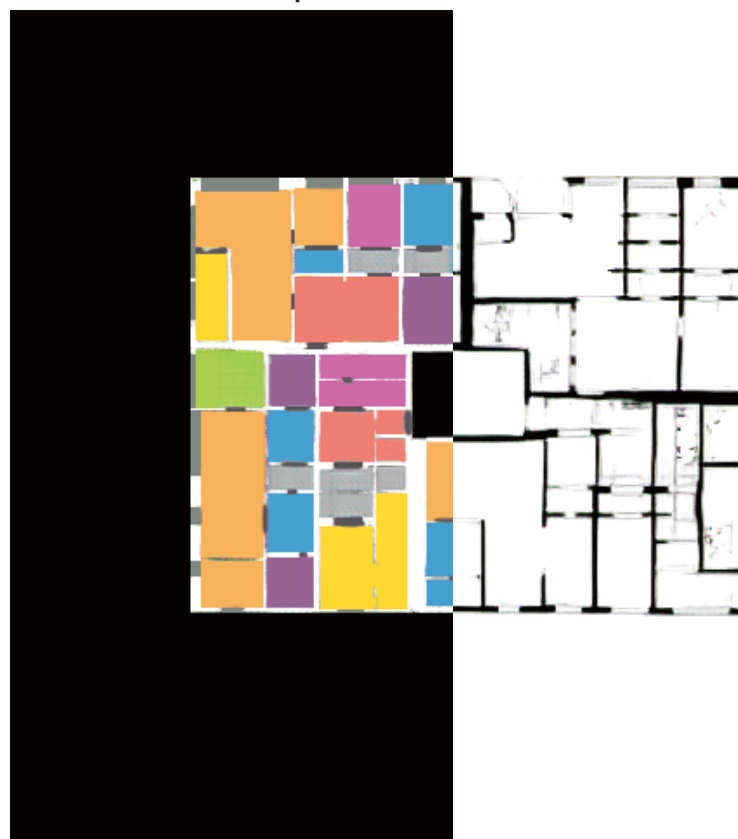


Figure 3

Design tools

Plan2facade

Plan2facade starts off with either a clean plan-drawing or a function map from the drawing that is provided it separates it into four facades each facing different directions. This process is not done by a neural network but instead done by automated image analysis that then locates the windows on the separated parts of the plan and draws a facade from that. Each floor-plan gives the process the amount of windows each

facade has and the placement of them, the process also draws the body of building for the network to add the details to the facade.

By training a network on how details such as balconies, roofs and window shutters are placed on facades this network can now be used to add detailing to the simple facades generated from the floor-plan.

Similar to mask2plan the last step is to generate a new version of this that we are more used to see, a quasi photo-realistic image of the facade generated from the detailed label map. As to the height and shape of the windows it's all based on relations between the amount of floors that are either defined by the user or randomized by the script.

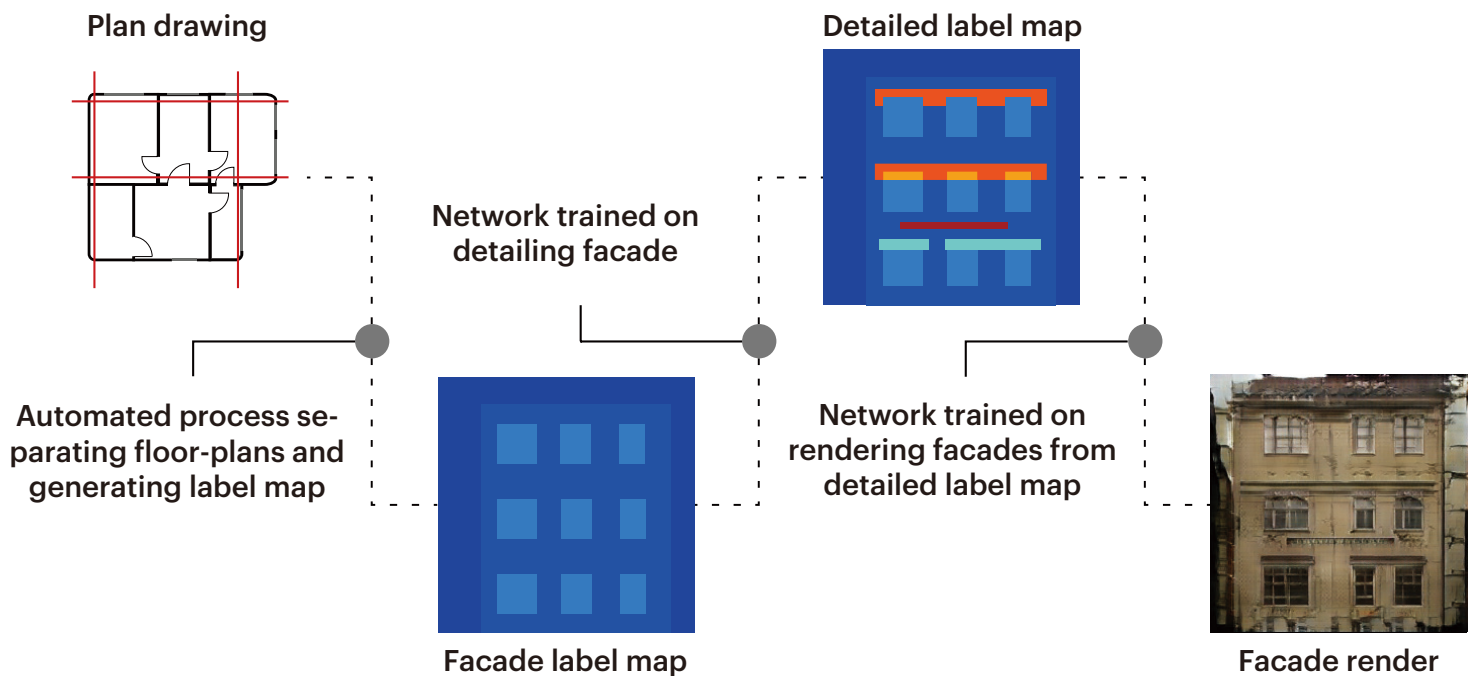


Figure 1

When working with non abstract prediction as seen in figure 5 provides a series of new challenges. For example backgrounds that are non labeled as they are in the input are a problem for neural networks as they are not sure how to process and predict on a undefined parameter.

From this problem rises a bigger overarching issue of context in the generation from neural networks. The network can be trained on facades related to the context the building is to be placed in, this process is not adapting to the context but simply mimicking it.

This issue was discussed in the interview with Sander Schuur (Belatchew Architects) and Anders Neregård (Sweco). While this is part of the underlying system on how neural networks work it can be seen from different perspectives. A creative process often consists of gathering inspiration either by choice or in the sublime (Botella 2018). The network works in the same way, the data set is the inspiration on the predictions it will make and the input can control how that prediction is formed.

So a solution would then be to obfuscate the dataset with new data that can inspire the network to try something new. Forcing it to make a non perfect prediction, this allows the user to manipulate and inspire the network. This type of system fit nicely into the already existing creative processes used in architecture of generating alternatives and then choosing which track to continue with.

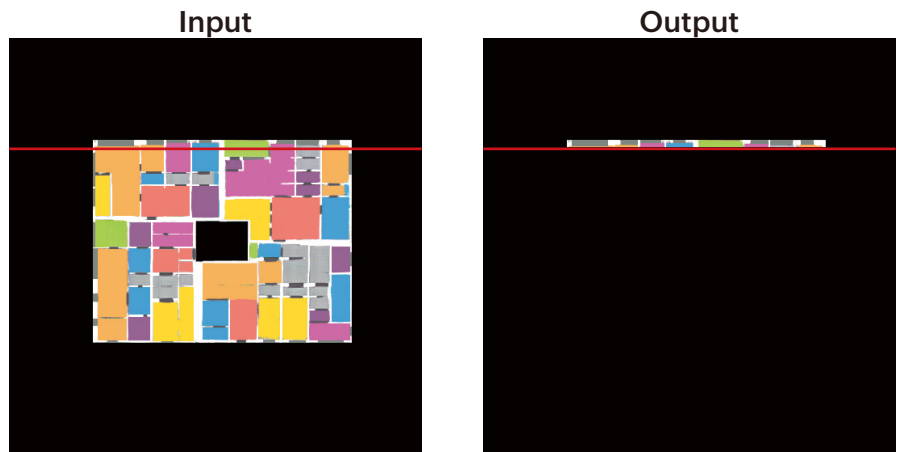


Figure 2

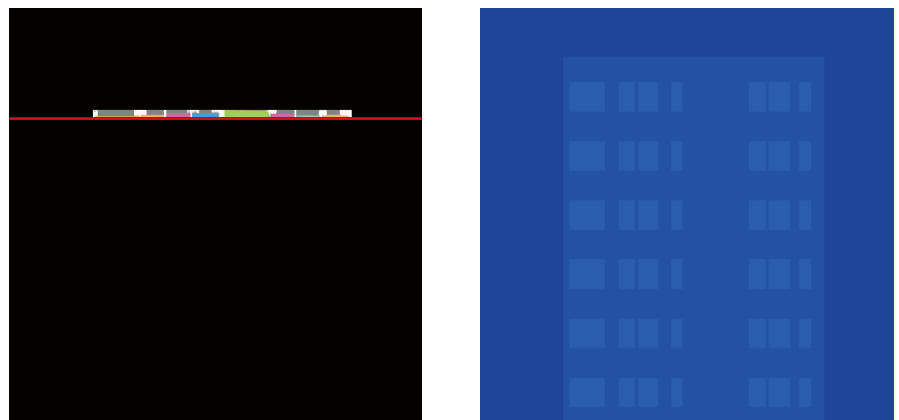


Figure 3

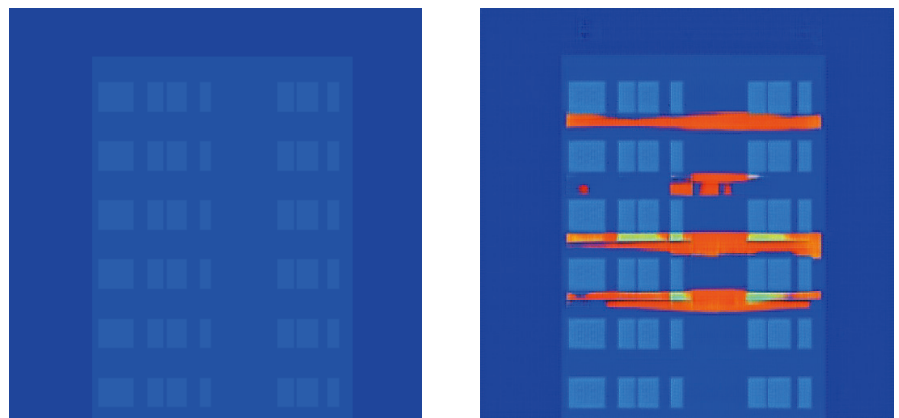


Figure 4

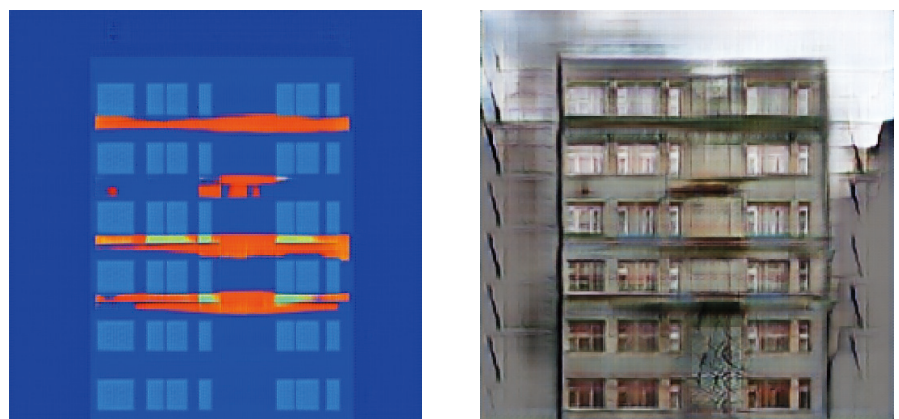


Figure 5

Design tools

Facade2volume

Facade2volume only incorporates a single step that uses neural networks but is instead mostly grasshopper based. As a first step a grasshopper script analyzes the function map generated by the network to create a 3d model of it, this script looks through each pixels and decides what layer to put it in and then extrudes it into a model.

After the 3d model has been

extruded the generated facade is linked to the model both to adapt the models windows to fit with the facade but to create the right height for the building. It does this by image analysis and then stacks the appropriate amount of floors on top of each other to create the right height.

As a final step in this process is to render the model using a convolutional-network (CNN) in a similar

fashion that was outlined in the theory section about style transfer and CNNs. By setting the volume as the content image and applying either a picture of a site from internet or using a rendering of the site the network can generate a image of the volume in a style fitting the site.

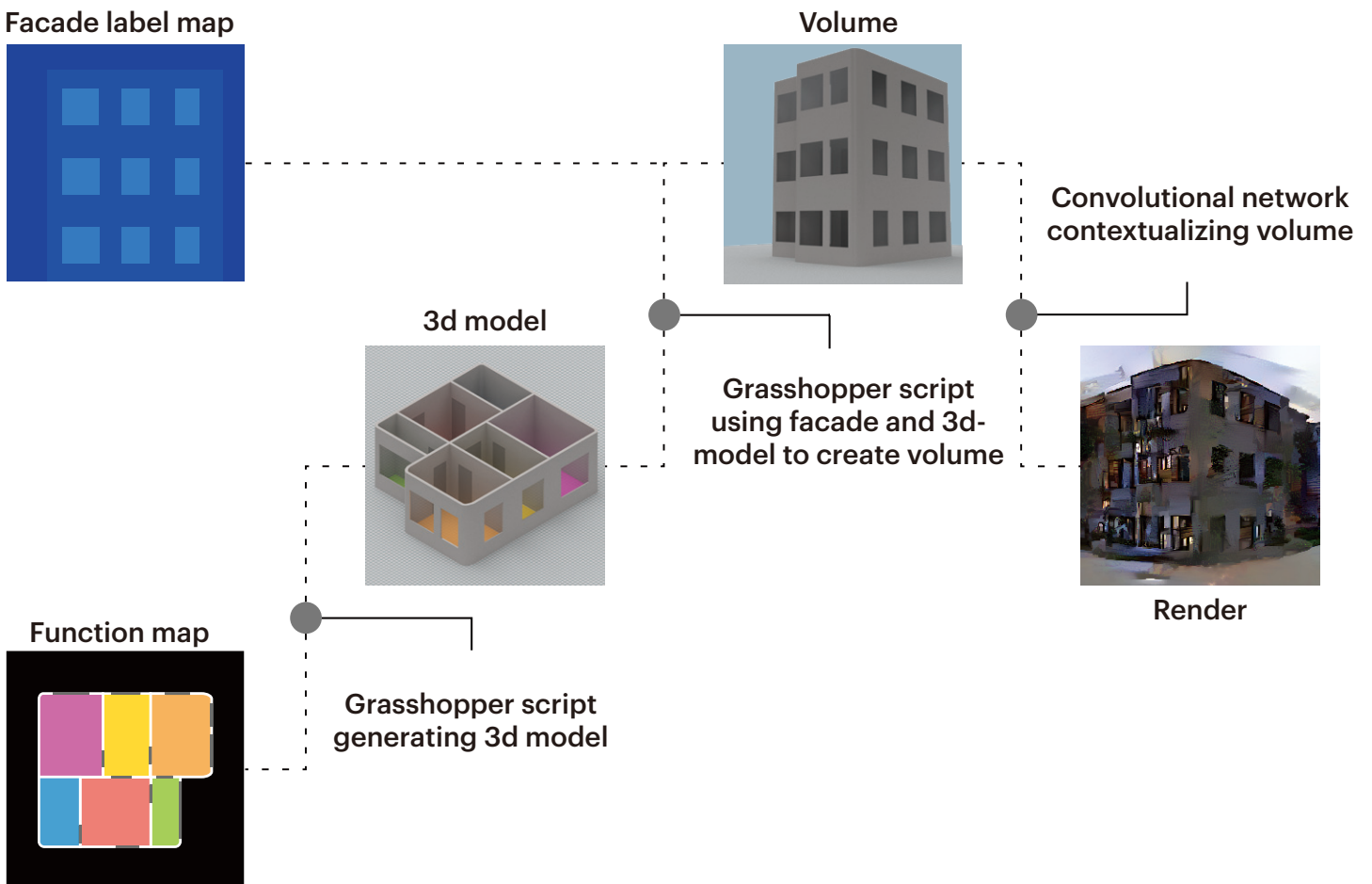


Figure 1

As the model is shaped from a base in the function map it lacks some crucial information when planning a building in three dimensions. Mainly there is a problem of circulation in between floors. This could be solved by adding another layer and another network which could be trained to populate the plan drawings with connections between floors.

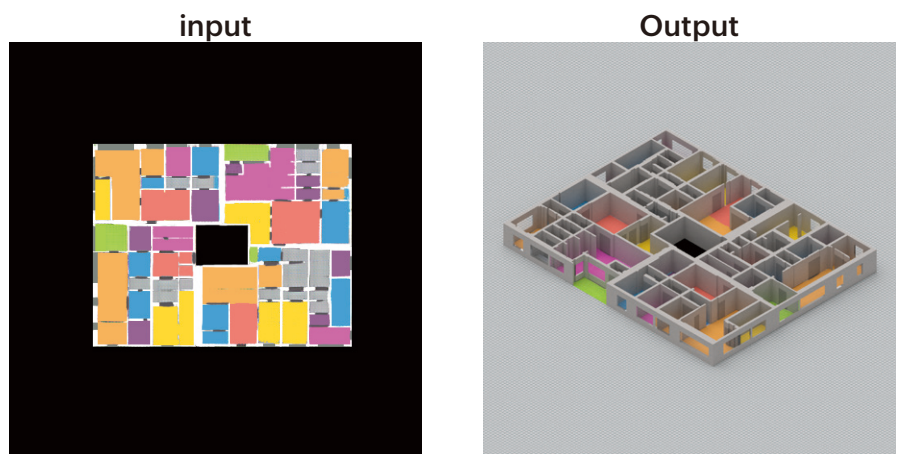


Figure 2

As discussed in the interview with Allison Petty (White architecture) context and building culture is a hard but very important parameter to integrate into generation. As these predictions are based on Finnish drawings they would fit in a Finnish building context at-least in space organization but when expanding the scope to three dimensions that contextual link is weakened.

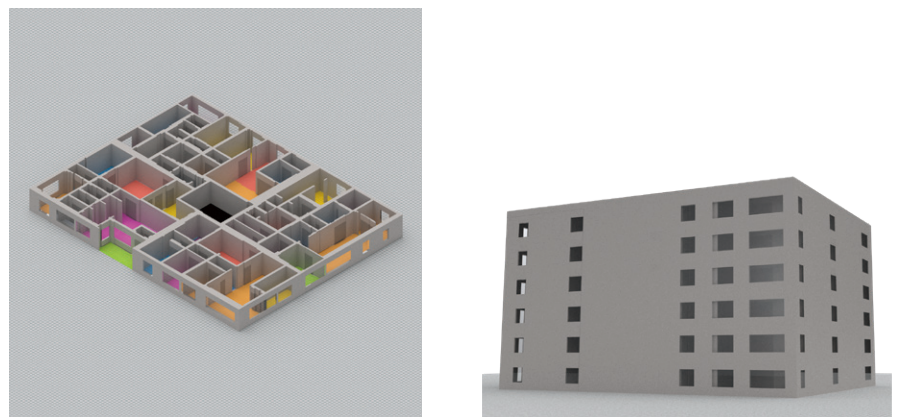


Figure 3

This might be one of the biggest problems that face machine learning in architecture, as it looks today many of the actors (Horn, 2020) develop systems that fit into a single context. But with technology based on open source material and big data sets become more available this might shift into a more global and general style.



Figure 4

Projects might get shaped by metrics and system that not applicable to the context they are placed in both culturally and ecologically. What will be important is the transparency of the data that is shaping these systems so that architects can critically reflect on the data sets implications on the processes it is applied to.



Figure 5

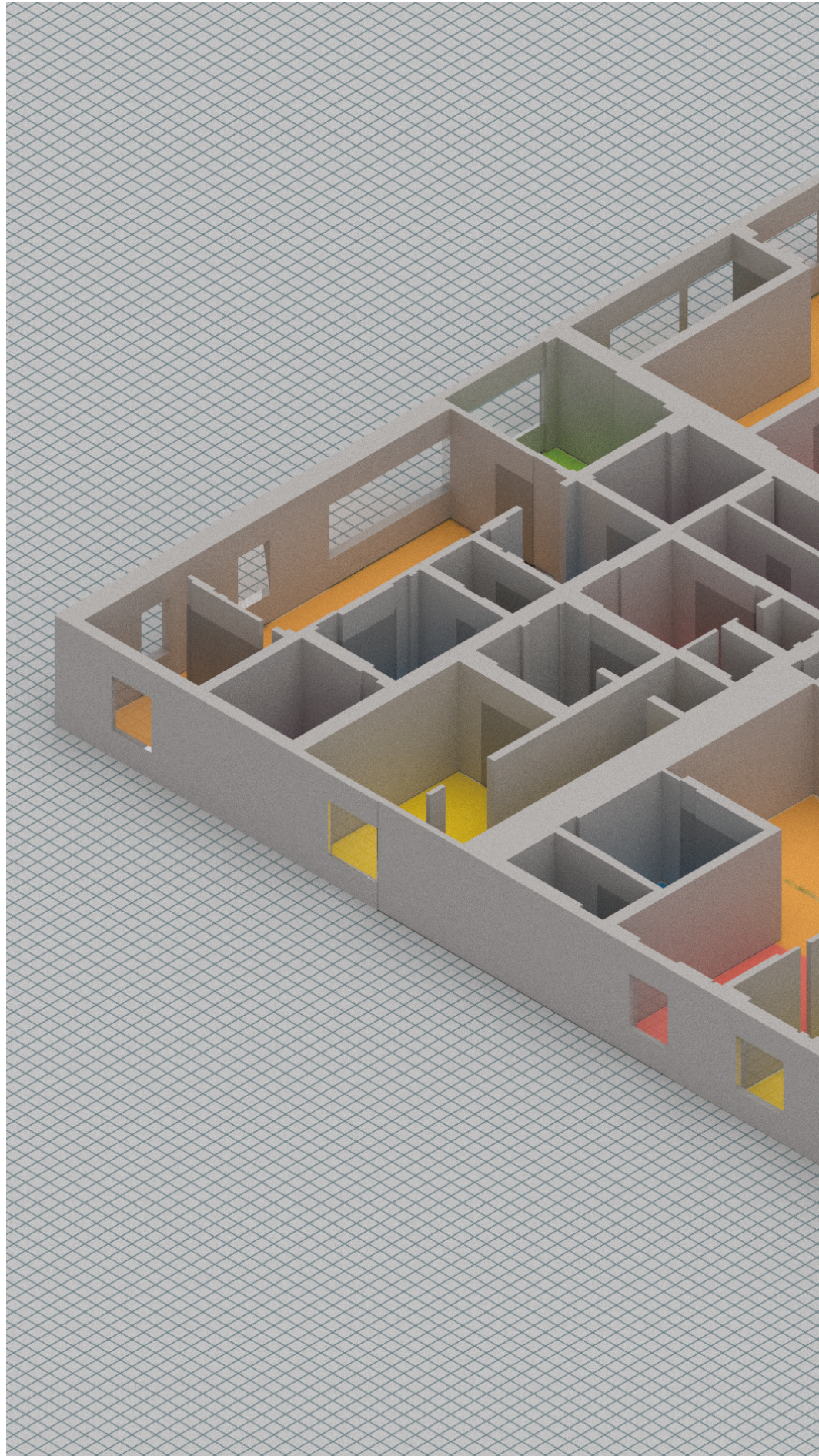
Design Tools

mask2plan

As the process ends we can see how starting from a two dimensional raster image with minimal information we end up with a three dimensional vector volume. This signifies how the networks incrementally can add or subtract information to a drawing by training on specific task, when all these networks are put together a output far beyond what was given at the start can be seen.

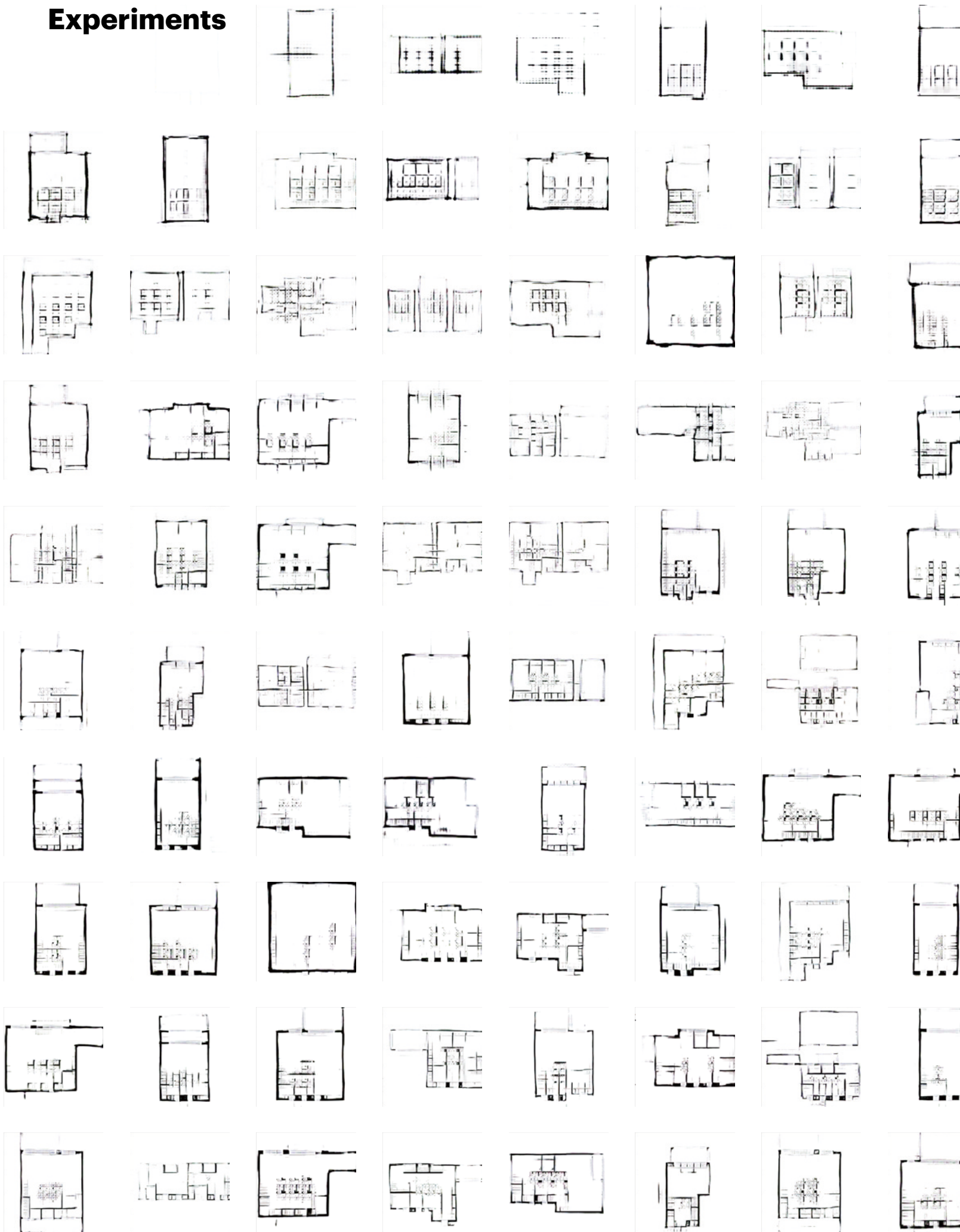
What was outlined in this chapter was just an example of an machine learning pipeline that was develop to showcase potential integrations and practical usage. While a tool for use in architectural practice would be more intricate and detailed in each step, this process shows how multiple networks can work together to produce results based on the other networks output.

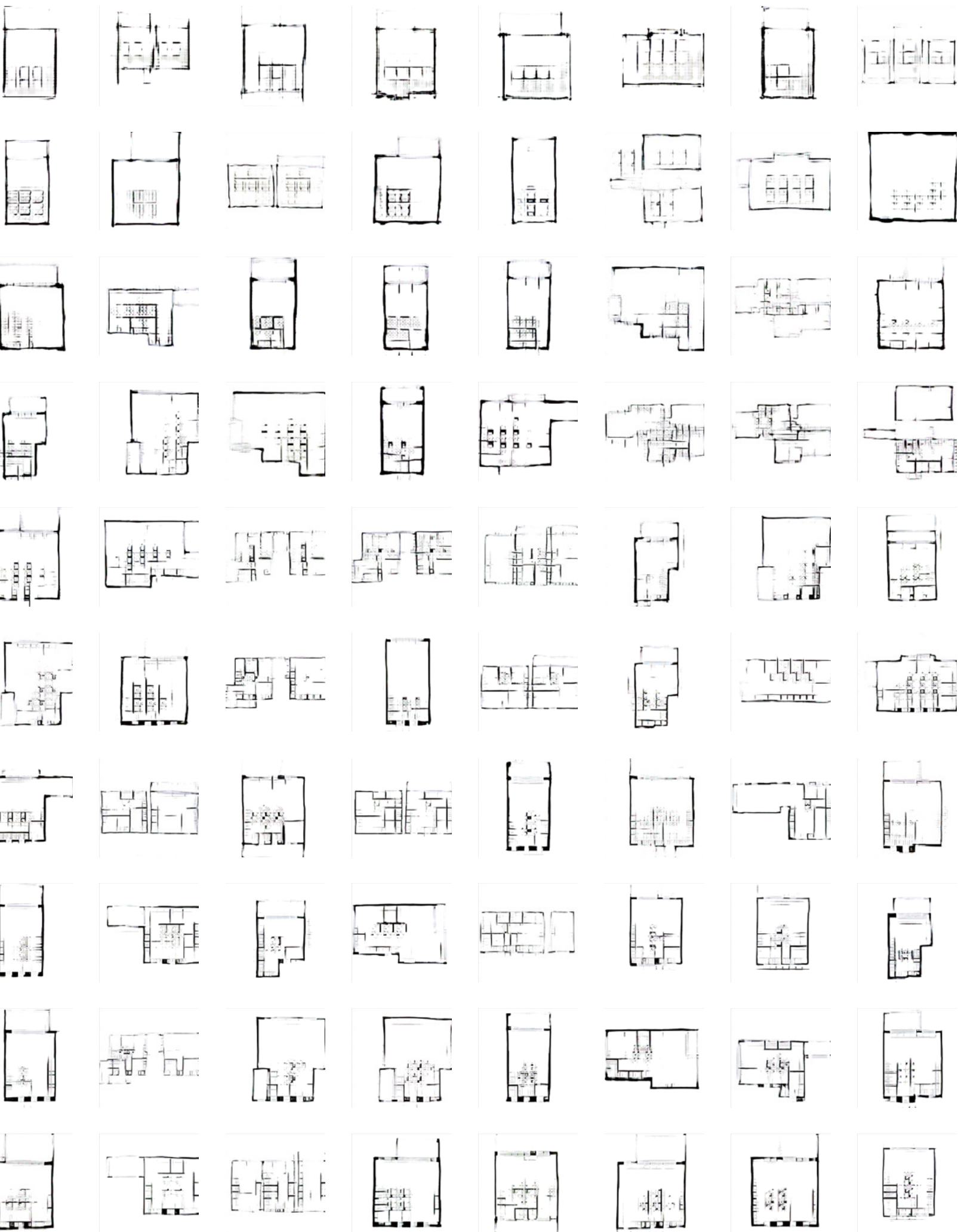
Most parts are fully automated and any number of similar output could be gotten from the pipeline outlined in this chapter granted some would end up with very similar result. While the end result is an building volume, the goal of this thesis was to develop the process that ended up generating this.





Experiments





Experiments

Dataset generation

When starting this project with the goal of generating floor plans a datasets to provide this information had to be built. The first test was using the process underlined in Cubicasa5k (Kalervo, A., Ylioinas et al. 2019.) it's used to process room boundaries and usage and convert this to a label map I call function map. This information will be used to train the cGAN in later steps.

As a first draft of this is the simple division provided in the first example below. Here we define common rooms such as bedroom living room, kitchen etc. It also defines outdoor spaces such as verandas and balconies.

In the next iteration the process is adding labels in more detail, separate labels for windows, doors and also fixtures such as toilets and closets are added. The last

experiment might look like a step backwards where we remove all detail and just provide an outline of the plan, but when working with machine learning sometimes less information provides better results. It is also much easier to test the results when the information is more abstracted.

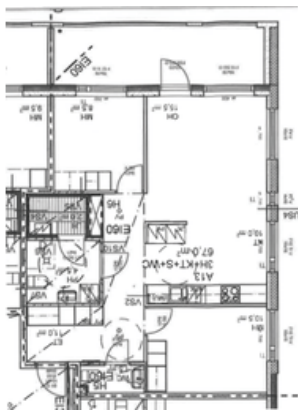


Figure 1

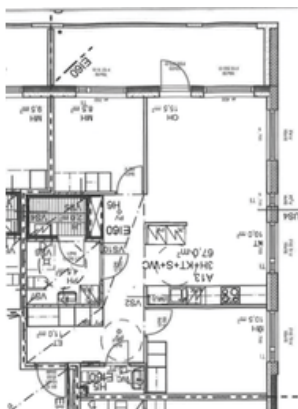


Figure 2

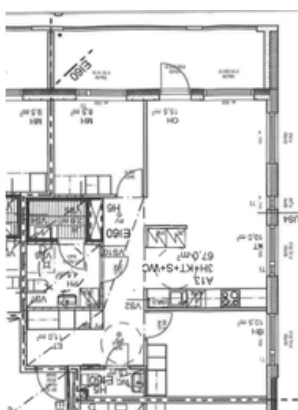


Figure 3

- Background
- Walls
- Outdoor
- Kitchen
- Living Room
- Bed Room
- Bath Room
- Entry
- Storage
- Garage
- Undefined
- Doors & Windows
- Chimney
- Bathtub
- Fire place
- Sauna bench
- Sink
- Toilet
- Electrical appliances
- Closet

Trying to adapt the dataset even more, a process called Deepfloorplan (Zhiliang Zeng et al. 2019) was included. This process provides larger flexibility and does not work with built in limitations that the earlier process was more prone to.

The earlier method worked solely with sharp 90 degree corners so rounded objects was removed when ran through the process.

While this new system does not provide the same amount of detail such as furniture and separate icons for windows and doors the information is less detailed but often more precise.

In figure 4 we see results made from original plan drawings. Figure 5 shows an original plan drawing with the boundary broken out from the process, while figure 6 depicts the results from an draw-

ing that has been run through a cleaning script. The results vary but the cleaned plans create a more definite result in most of the cases.

Both systems have their strengths because they are trained in different areas, the latter system will be used to a larger extent since it is more flexible and provides cleaner results.

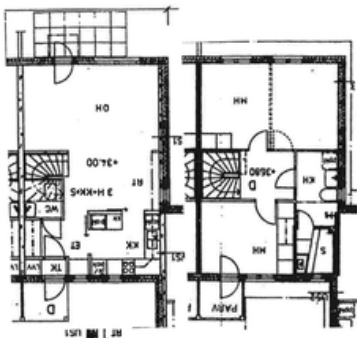


Figure 4

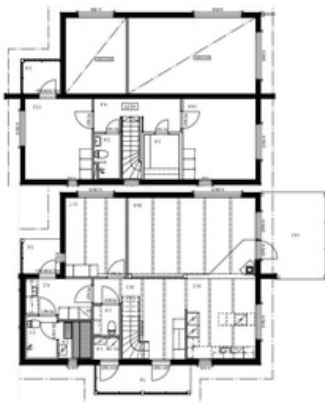


Figure 5

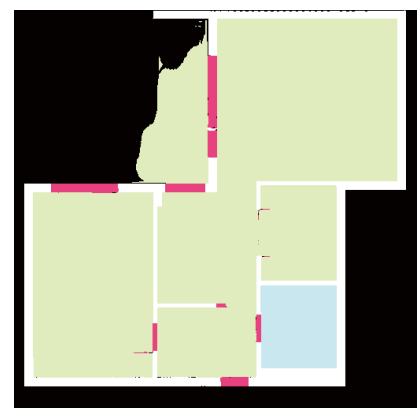
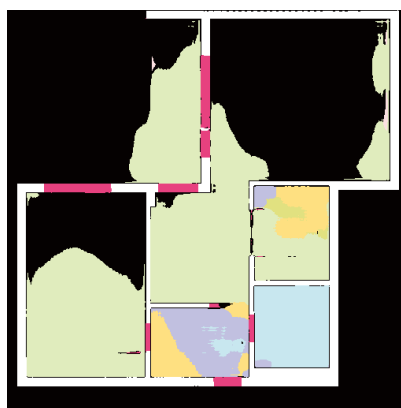
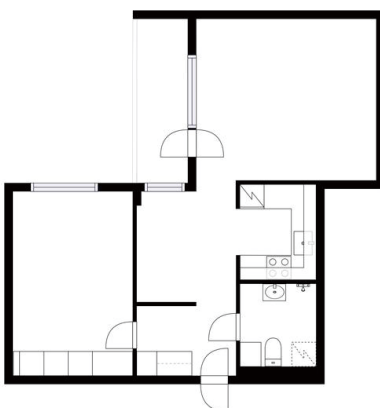


Figure 6

Experiments

Floorplan training

As the dataset and method has been defined for our process training can begin. A training process is where the network learns how these labels created in the dataset relate to each other and from it try to generate new output. A training process is divided into two main parts: generations and epochs. A generation consists of the network going through all paired images in the dataset and from it produce a predicted

image. When one generation has passed it starts with the next epoch.

Below we can see an example of this made on the cubicasa5k dataset. There is a ground truth of an original floor plan, an input image that consists of the room based functions that were segmented in the dataset generation and a prediction generated by the network.

When the network is going through the epochs it learns how to generate a prediction based upon the input that is as similar to the ground truth as possible. A number of epochs is given to the network and when it reaches the final epoch the network stops. The number of epochs is not when the network decides that no more progress can be made but simply a stop point set by the user.

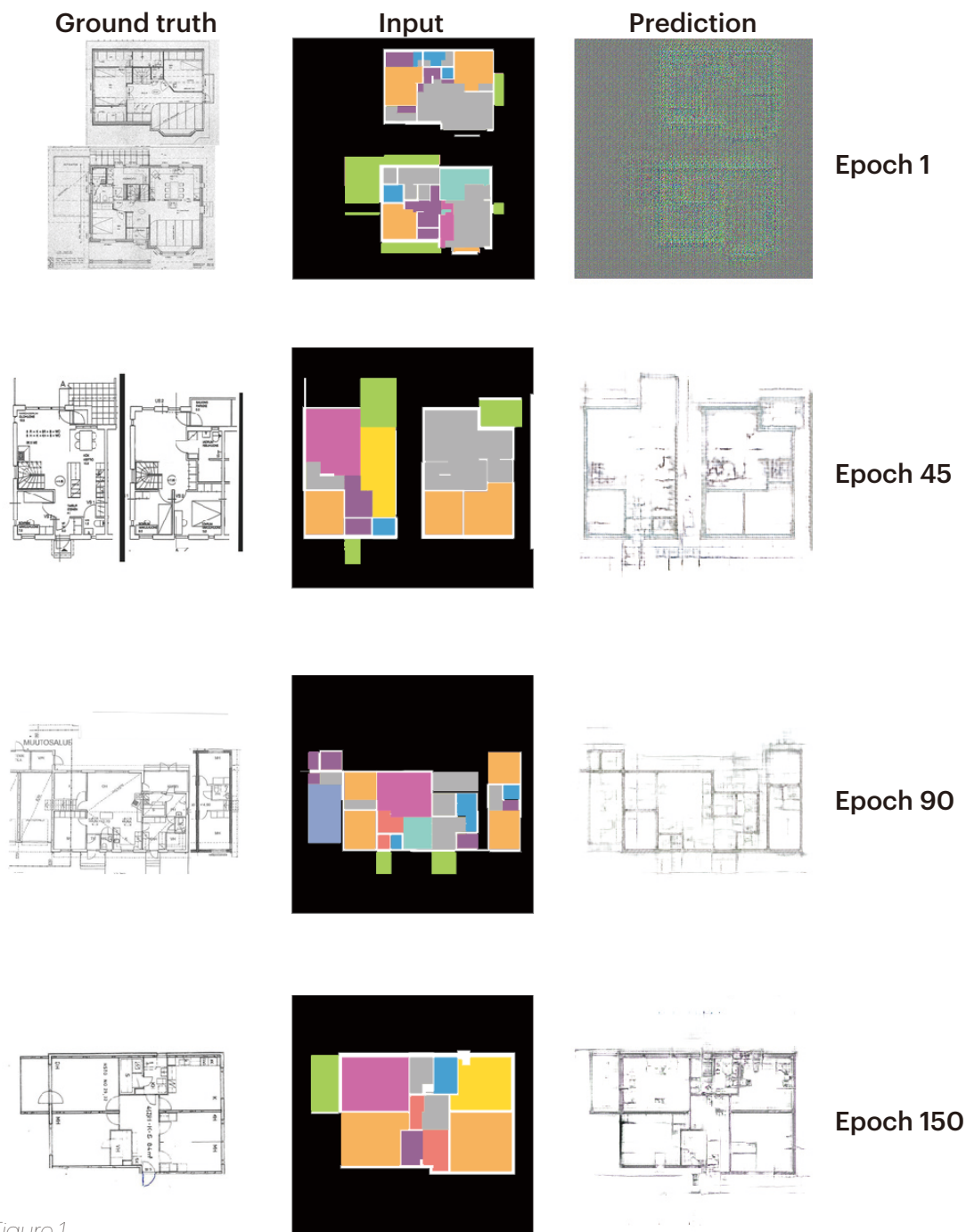


Figure 1
36

As can be seen in figure 1 this training process is done in the same way but with the dataset that contains more information, also regarding interior and exterior detailing. In the same way as in figure 1 the generator of the network will only see the input image while the discriminator only sees the prediction and the ground truth. This is why the network is called adversarial.

When adding the more detailed information the plan drawings start to form better and more coherent predictions. Doors and windows are added where they are placed on the original floor plans and installations are drawn in a definitive manner.

As for the predictions or the output the results look better and have more information. But when comparing the amount of infor-

mation given to the network and the information in the output to the more simple one in figure 1. It could be argued that the tests in figure 2 performed worse in regards to creativity. It mimics but will not add anything new to the drawings.

The training process here took around 4 hours to complete since the quality of the images is quite low (256 x 256 px).



Figure 2

Experiments

Floorplan training

The final dataset we generated from the cubicasa5k data was the most simple mask of the outlines. While the approach has the least amount of information in the input image it gives the network the highest freedom. Instead of giving the network where each room and function is a large area devoid of information is given.

An approach that forces the network to try and fill in the masked

area with a prediction that will be read as a plan drawing. This method produces many incoherent predictions where a quick glance over the output immediately highlights the faults. But the results do differ quite a lot from the ground-truth, showing the potential in this approach as it's able to produce new drawings.

While these masks are automatically generated the goal of this

training is to produce a tool that is able to generate new drawings based on human input. That is where this method really shines, to be able to produce a drawing based on as little as possible information made from a input image created manually. In figure 2 we can see the generations made from masks as tests that were trained on this method.

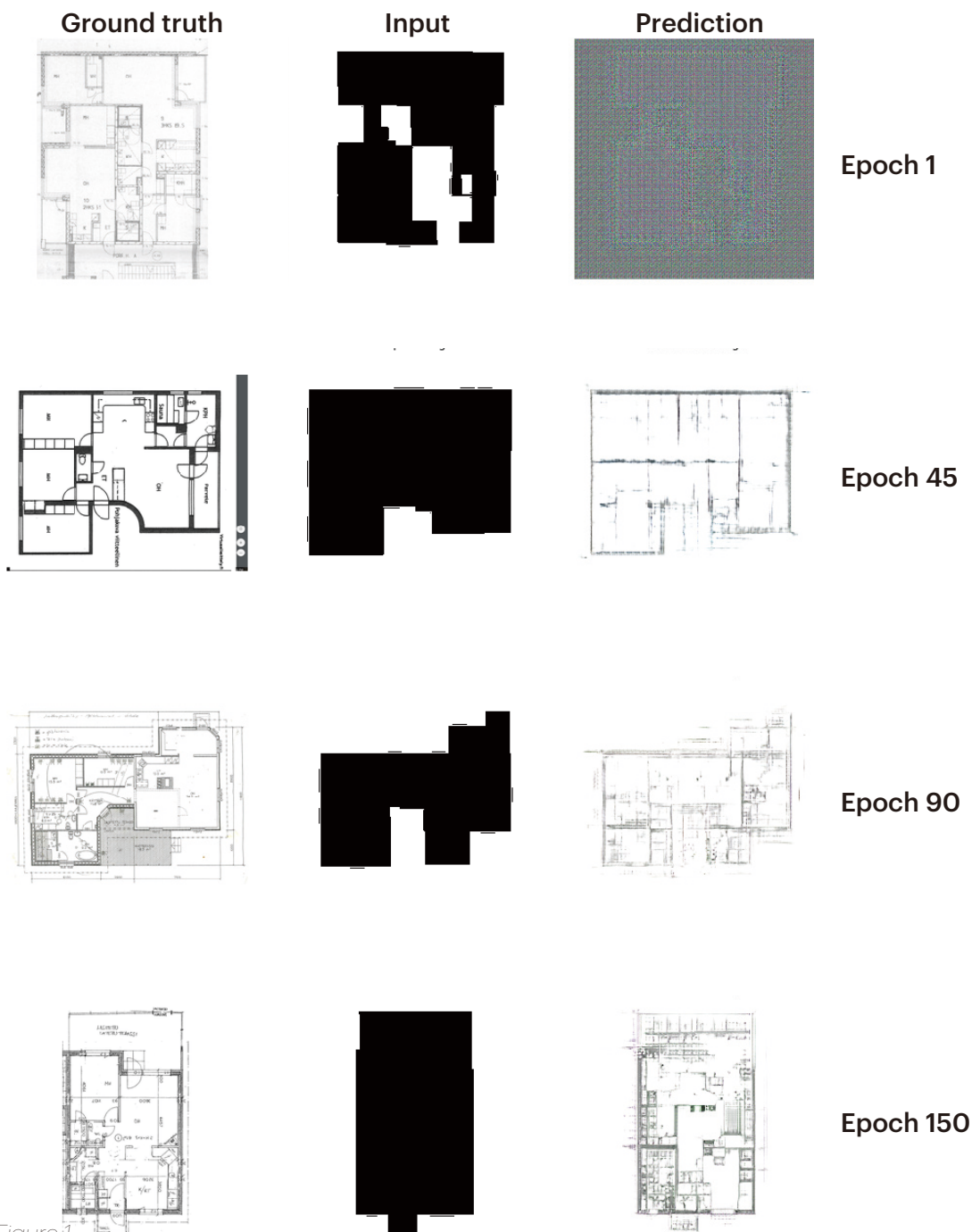


Figure 1
38

When the training is complete one can simply use the trained model to generate new images based on the logic that the network was trained on.

When the network was trained on the relationship between floor plan and mask we can now give the network either of those and it will try to fill in the other part. Here we can see the network produce plan-drawings when the mask is supplied.

These generations made here might not be the easiest to read or interpret since the quality is still only 256x256 but it shows how this technique can generate new and novel plans.

While these are great as a first test the output on these needs to be made more clear and easier to understand for it to be viable as a inspirational tool.

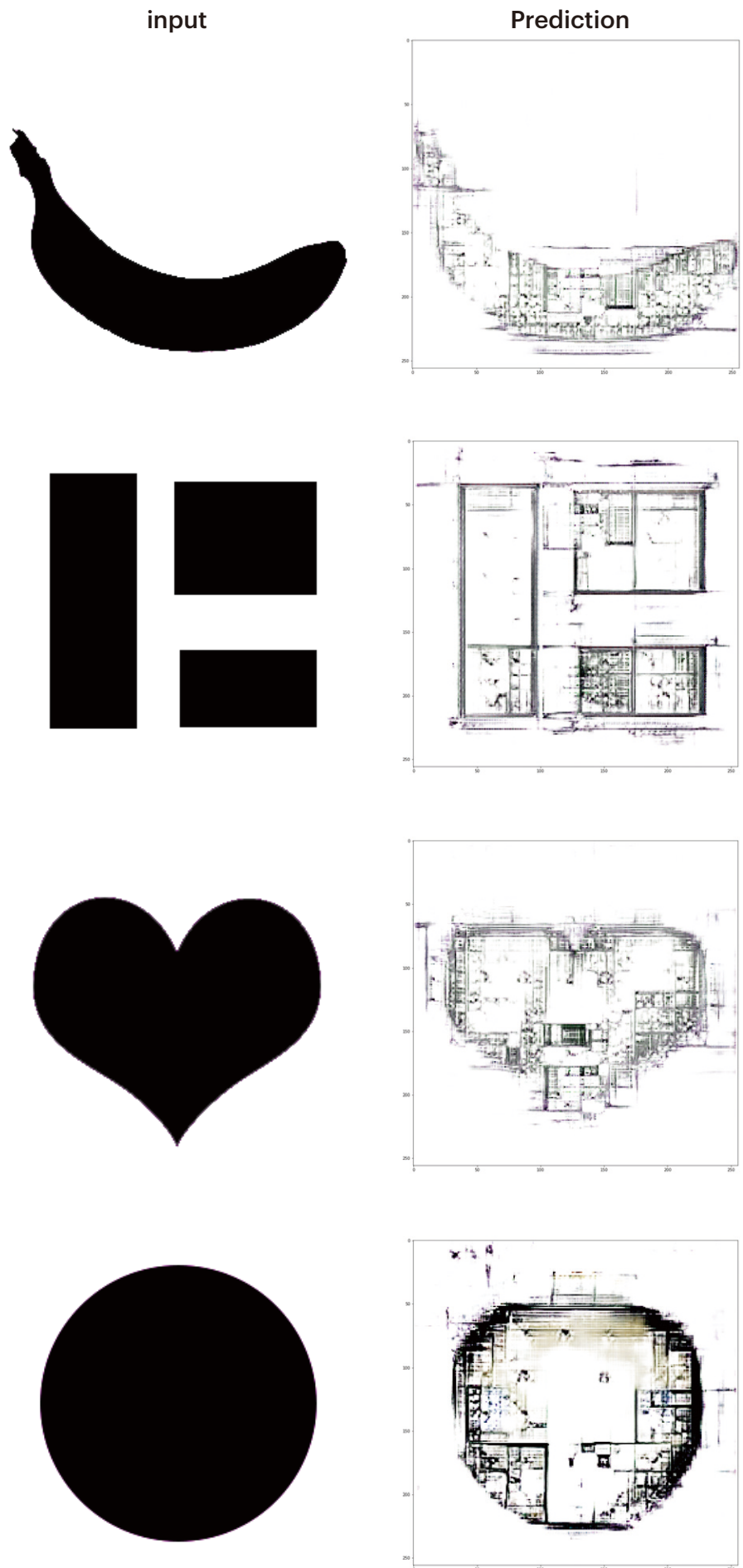


Figure 2

Experiments

Floorplan training

To try a more flexible approach a secondary set of dataset images was created, made from the deepfloorplan method. These provide less detail and less precise results but in exchange they provide larger flexibility in their input. Able to produce function maps on almost any sort of forms. While training the network to understand these drawings one can see the lack of precision really inhibiting the predictions. While

some of the input images have holes in them and they create small islands where the network is not quite sure what is present as can be seen in the input image of Epoch 1.

As for the training process it is quite straightforward but the results themselves look fine even though the input images lack precision. What also can be seen is the lack of interior detailing but

inclusion of only walls and doors provide good results in the floor plans generated. Interior detailing is still planned out but not replicating the ground-truth.

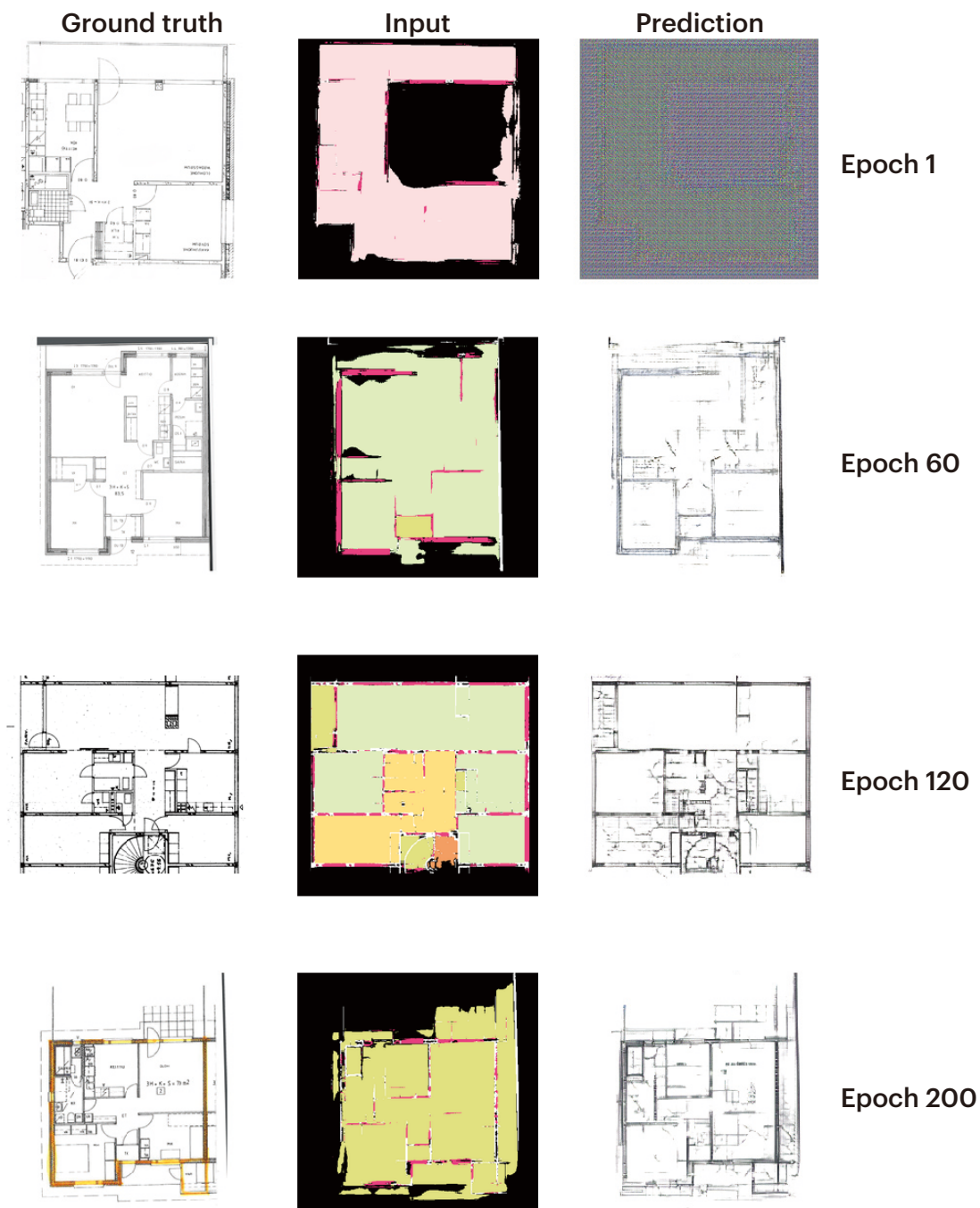


Figure 1
40

All earlier training has been done on a 256 x 256 pixels dataset providing very small and blurry images; this one was run on a 512 x 512 dataset. Except that the image size was larger it ran on the exact same dataset as figure 1.

dataset. While this training took 14 hours to complete, while this is not an obscene long process it slows down the iteration speed quite a lot. The images produced are more crisp and are easier to read.

the predictions in any meaningful way.

One obvious difference in these two sets was the training time, a 256 x 256 dataset takes between 4-6 hours to complete depending on the amount of images in the

Otherwise the results from this training is very much similar to figure 1 in all regards showing that scaling the resolution of the dataset images doesn't change



Figure 2

Experiments

Sketch

Sketches are one of the main tools an architect has to work with. While a sketch can contain a large amount of information it often is a minimalistic approach of a architectural drawing containing the absolute necessities for it's purpose. Inspired by the successful training of masks to floor-plans a dataset of quick outlines in relation to a floor-plan was produced.

Each floor-plan has a hand-drawn

outline produced that is then used for the training of a network able to read a quick sketch and then generate a plan-drawing based upon the sketch.

As for the method it works exactly the same as earlier trainings. Two images are given to the network in relationship to each other: a ground-truth of a floor-plan and a input image representing the outlines of the floor-plan.

This dataset consists of 500 images paired in this way. As a result the training time went down and more epochs could be added, the training on this model was made through 250 epochs and done in less than three hours.

input

Groundtruth

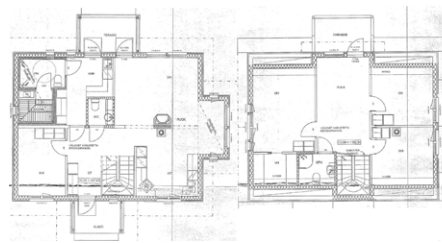
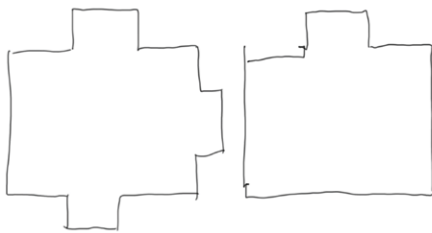


Figure 1

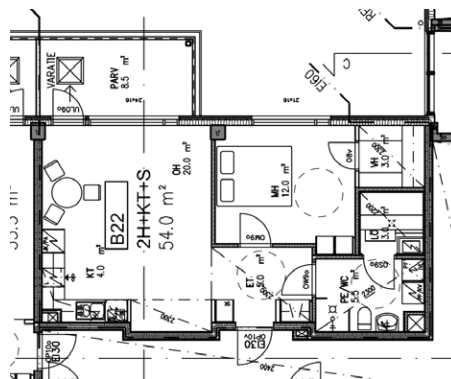
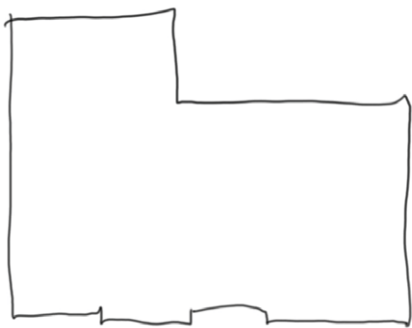


Figure 2

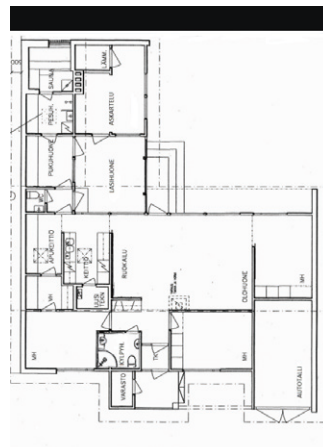
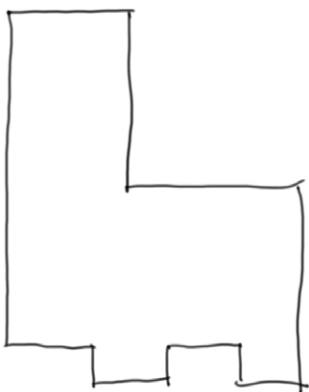


Figure 3

After the training is completed we can feed the network new input outlines never seen before and it will try and solve a floor-plan for these outlines.

While the results are not perfect it shows potential of how tools like these could really speed up early stages of architectural projects.

When generating these new predictions patterns of the network start to show. In the middle of each predictions (right where the beak of the duck ends) the same type of line is repeated as well as in the lower part another pattern repeats itself.

These patterns are one of the greatest problems with using creative networks in machine learning. The network will eventually think it solved something and will keep repeating the same patterns over and over as they are for the network the correct way to generate it.

input

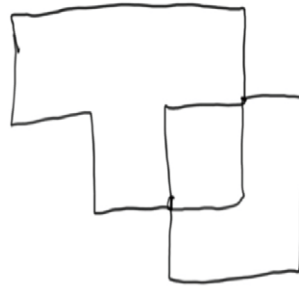


Figure 4

Prediction

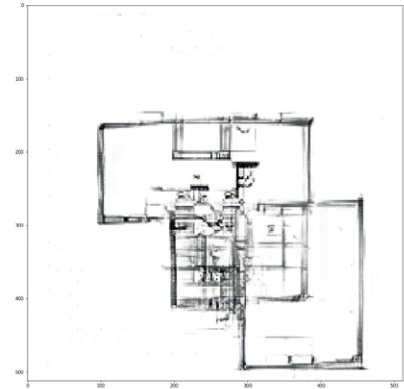


Figure 5

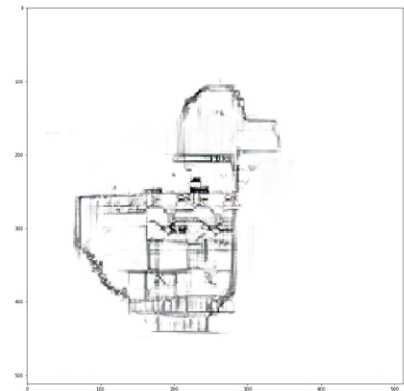


Figure 6

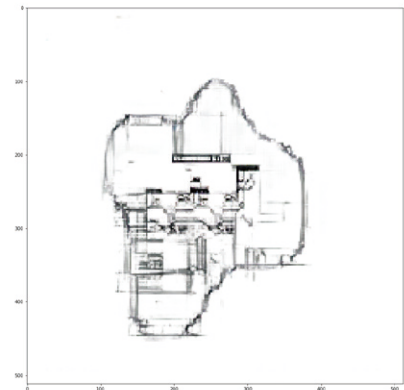
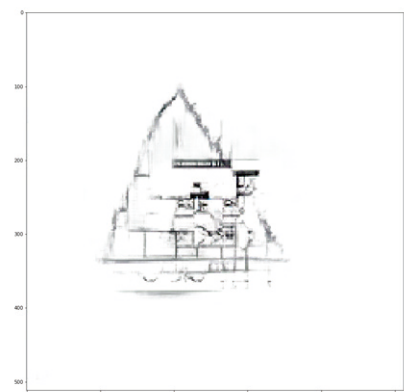


Figure 7



Experiments

Sketch

While the sketch to plan process provided okay results and lowered the threshold between the network and the user some changes could be made. A new dataset was produced where new input images as outlines was defined and new ground truths as cleaned floor-plans.

To be able to make these floor-plans a new process was created of cleaning and removing excess

information from vectorized plan drawings. These cleaned drawings provide a good base for making more detailed and reducing the noise from the floor-plans used before.

These two iterations of basically the same system shows how important defining and finding a good dataset can be to machine learning. While both use outlines as an input and a floor-plan as the

ground-truth they produce vastly different outcomes. Something to note is that the dataset doesn't consist of the same 500 pairs of input to ground-truth.

input

Groundtruth

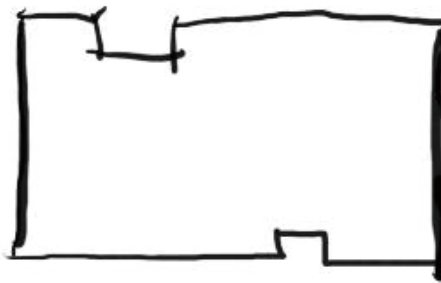


Figure 1



Figure 2



Figure 3

When this model is given new input outlines to try and predict we can see that they move towards being more clean and easier to read. While they still are hard to read it has vastly improved.

As can be seen in figure 4 the network has trouble with rounded shapes as the provided dataset of architecture seldom consist of round forms so it simplifies it to straight lines and corners, while this might not be intended by the user the networks bias towards the architecture it's fed by the user shows very strongly in both figure 4 and figure 7.

Since the training process is quite fast on these examples the quality is turned up. While these predictions are ran on a 512 x 512 network they could be generated on a 1024 x 1024 quite easily while still having a smooth process, this is a function of both having a smaller dataset and more easily interpreted input.

input



Figure 4

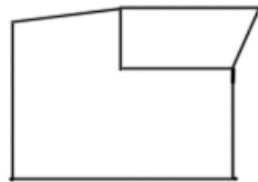


Figure 5

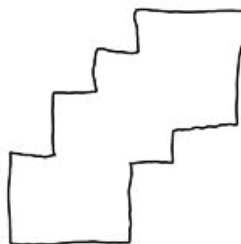


Figure 6

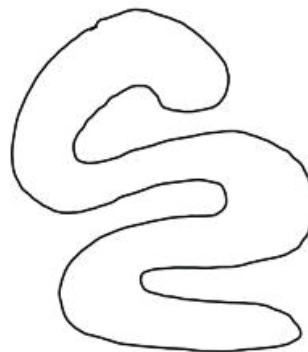
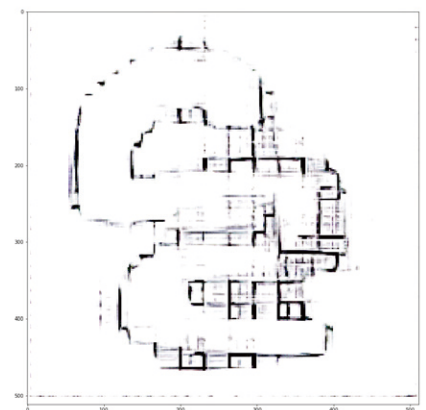
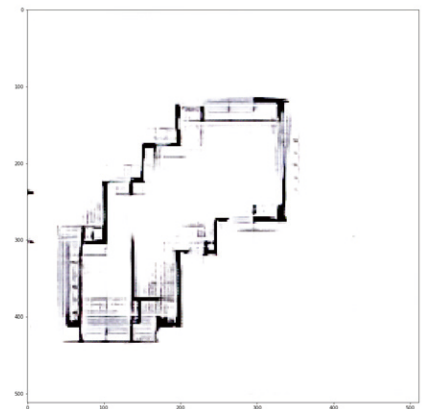
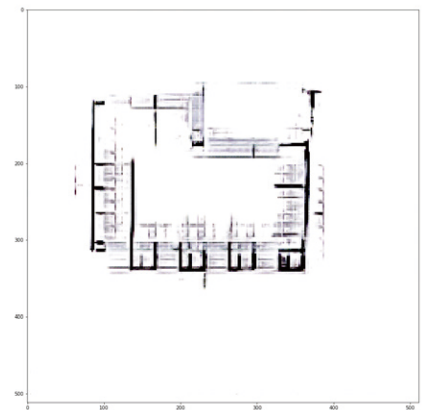
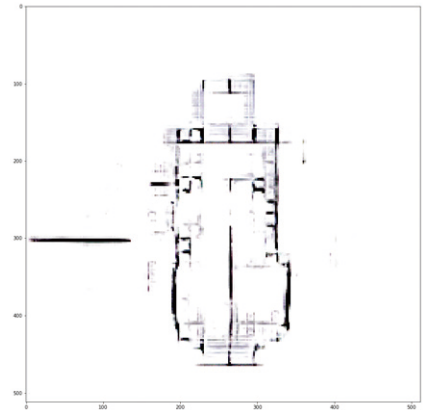


Figure 7

Prediction



Experiments

Masks

While sketches provided a easy way to create user generated input the precision of the sketches could easily become quite low. To be able to provide more precise and more relevant output and input solid masks that mark the area that is of interest was a next logical step.

Instead of directly working with the clean plans work started with coupling the masks with function

maps instead and a second network trained on the relationship between function map and cleanly drawn floor-plan will be trained.

This creates a very focused network that is only trained on understanding shapes and the relationship between them and the functions that inhabit the floor-plans. To these function maps a lot of information was removed and only the minimum necessary

information of room functions and windows and doors was kept.

These masks can easily be interpreted by the user and relationships of these functions can easily be read both by neural networks and by humans. Colors seen in figure 1-3 are chosen based upon cartography research (Brewer, C. A) for ease of reading.

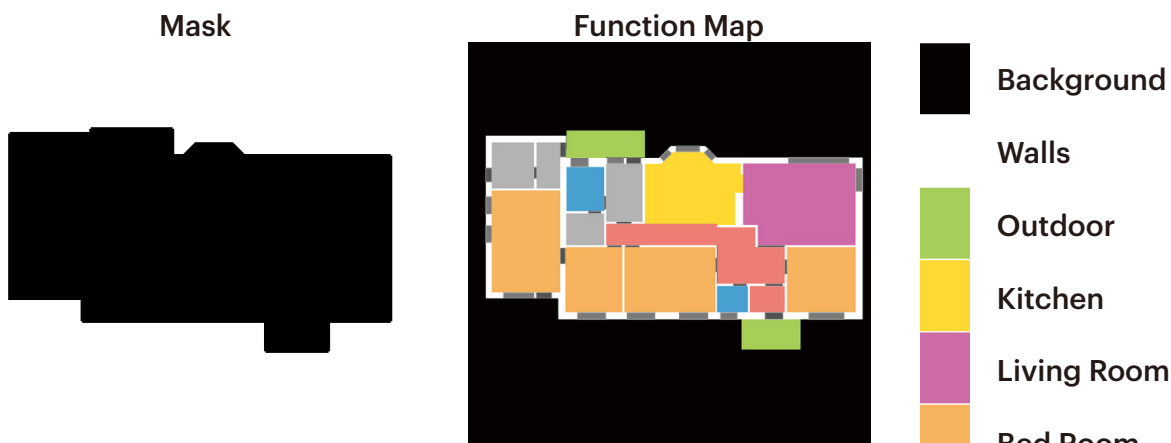


Figure 1



Figure 2



Figure 3

Just as there can be created a relationship between a mask consisting of a very abstracted version of the floor-plan and function map a network can be trained on understanding the relationship between function maps and floor-plans.

This part of the process is not meant to have any human input and meant to simply convert the function map generated by the

network into a more easily read floor plan as we are used to see and read them. While this process adds little new information and even obfuscates some information on rooms function it abstracts it in a format that is more commonly used by architects.

The form that these floor-plans are generated in are by choice with as little information as possible removing notes on mea-

surements, room functions and furnishing to provide a floor-plan that is as simple as possible for the network to understand.

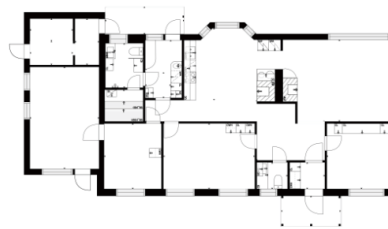
This process works as a proof of concept on how a network could transform creative predictions into more concrete and ready to build solutions.

Function Map



Figure 1

Floor plan




-  Background
-  Walls
-  Outdoor
-  Kitchen
-  Living Room
-  Bed Room
-  Bath Room
-  Entry
-  Storage
-  Garage
-  Undefined
-  Doors & Windows



Figure 2

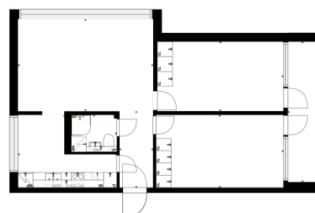


Figure 3



Experiments

Masks

When starting to train these networks the first one is on the relationship of masks to function maps. During this training a network is trained to generate a function map based on a masked outer boundary. This training used the dataset generated for this tool and consists 5000 pairs of masks and function maps with each image having a size of 512 x 512 px.

A higher than normal epoch count was used as the results need to be as good and defined as possible. When increasing amount of epochs and the quality of the dataset images a training time of 36 hours was reached. Increasing the overall quality of the training with higher training times will be beneficial when moving on to generating the floor-plans. Some issues arose with masks spanning to the edge of the

image and the network having difficulties with identifying the borders of the masks. All images was shrunk to 95% as a work around.

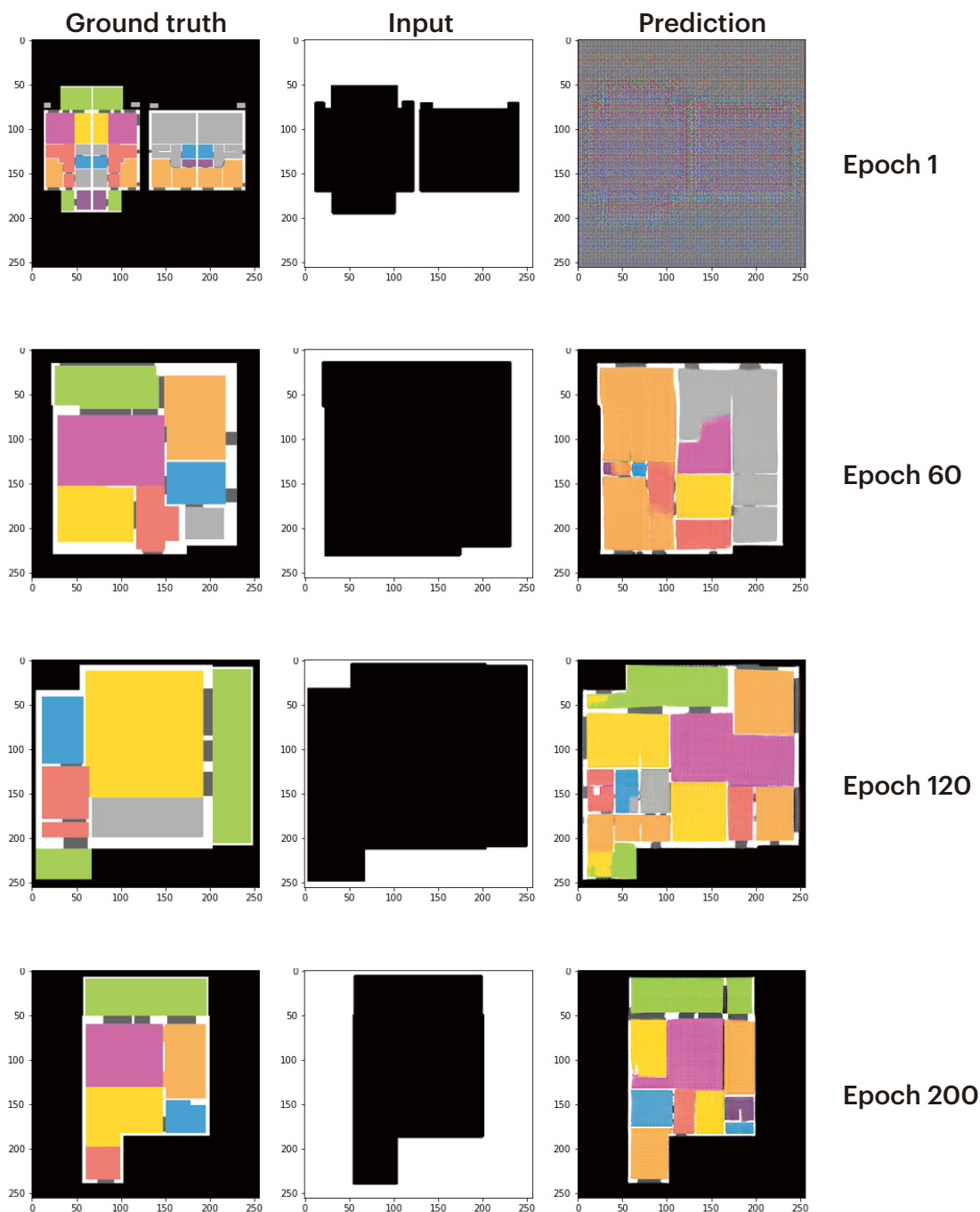


Figure 4
48

The same way that the earlier example of the sketches the network can provide a prediction based on the relationships it has learned in the training process ran on the correlation of masks and function maps while taking into account the mask given as input. While a sketch is quicker to produce the precision of the masks provides a better prediction in most cases.

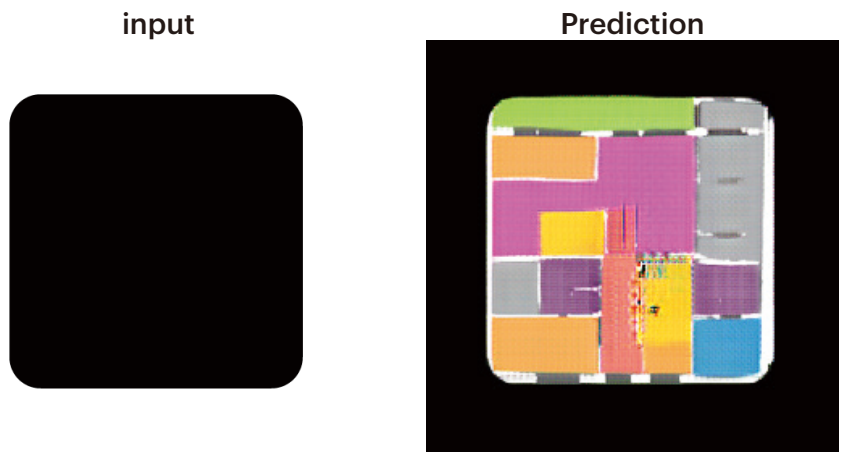


Figure 4

These predictions inherits the standards and relationships that are used in the architectural drawings that are fed to the system. Since the dataset consists of mainly Finnish floor-plans we can see how to data provided shapes the generation, for example entrances and bath rooms are placed in connection to each-other which is the norm in Finnish architecture.

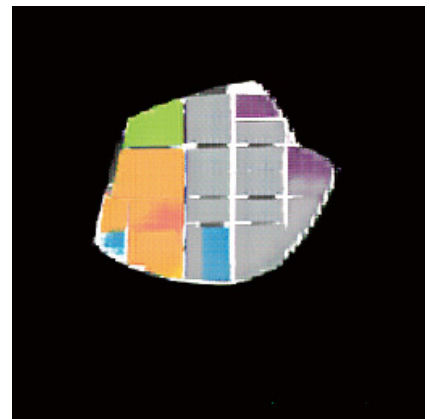
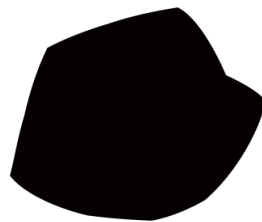


Figure 5



Figure 6

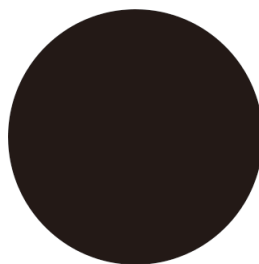


Figure 7

Experiments

Masks

Just as previous training processes the network is coupled with an input and a ground truth, here it consists of the cleaned floor-plans and the function maps. It still uses a 512 x 512px output which provides decent quality output.

In contrast to the network trained on the relation between masks and function map was ran on a high epoch count of 200 this training can be lowered. Mostly

due to that the quality of the predictions was increasing quite fast and the inherent loss of the optimizer informed the training that it was not needed to go further.

Optimizers and loss is a system in place for the network to avoid getting stuck in a repetitive pattern (Diederik P, 2014). Imagine that the training process can be expressed as a three dimensional surface where the most optimal

solution is the lowest point. An optimizer tries to navigate this surface to find both the locally lowest point and the overall lowest point. So when the loss is low it means that the network has found a solid spot that is low on the surface this might happen early by chance but when it's found a good spot the training process can be completed.

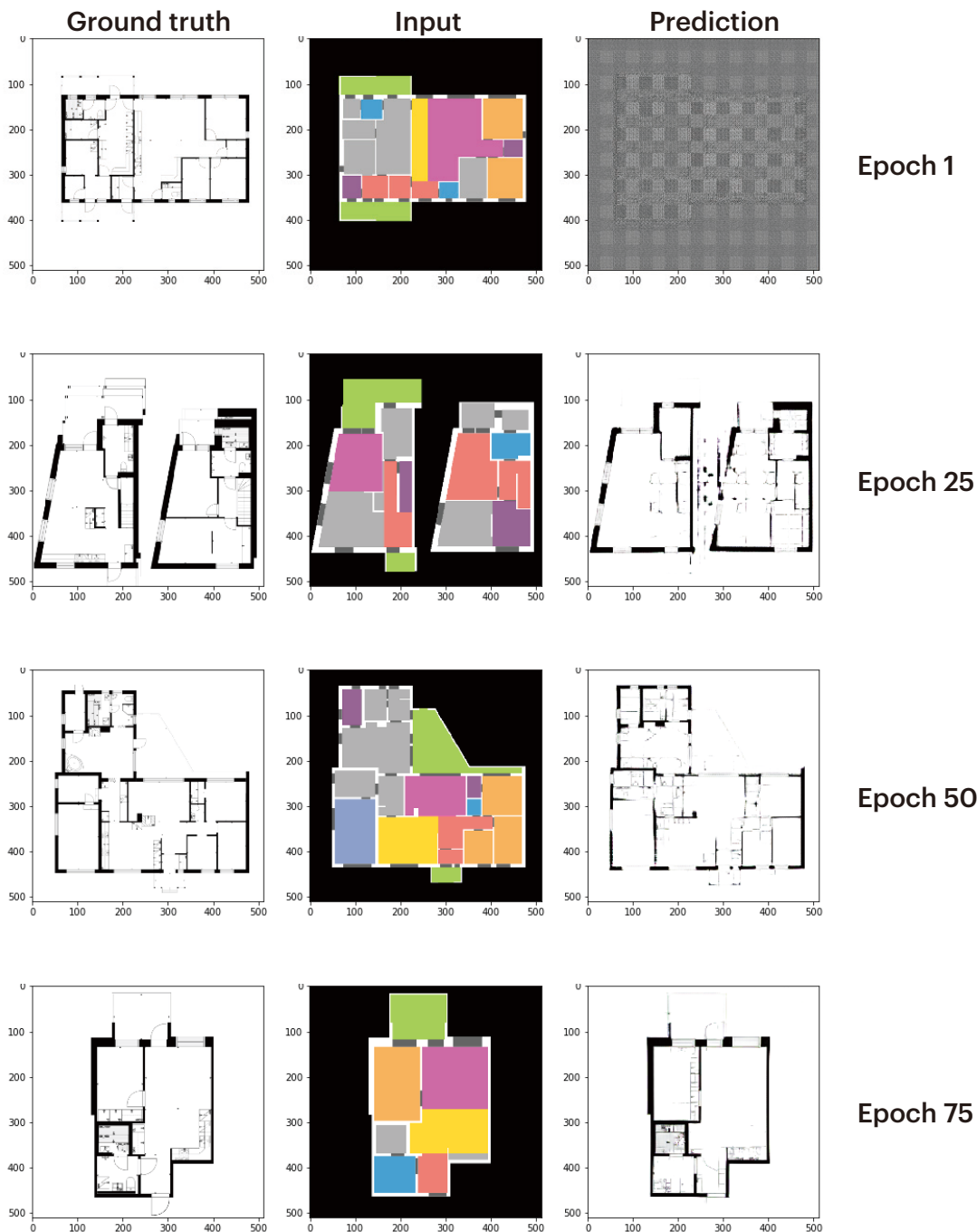


Figure 1
50

The same way as the earlier examples of generation a input is given to the network and it will try to generate a new image based on the relationship it's trained on. The difference here is that the input is not generated by the user but it's a prediction by the network on a user input that was supplied earlier.

The networks are thus working in co-junction on producing a more reliable and more precise floor plan. One can quite easily see the potential of multi dimensional networks trained on very specific and tailored tasks working in co-junction to assist user in a multitude of tasks.

These predictions are not complete and fully functional drawings but act as a base for further analysis by both network and humans.

input



Prediction



Figure 2

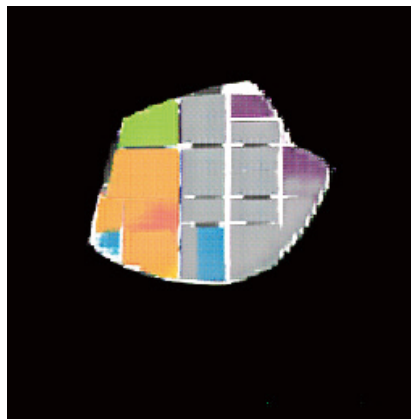


Figure 3



Figure 4

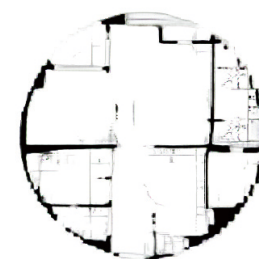
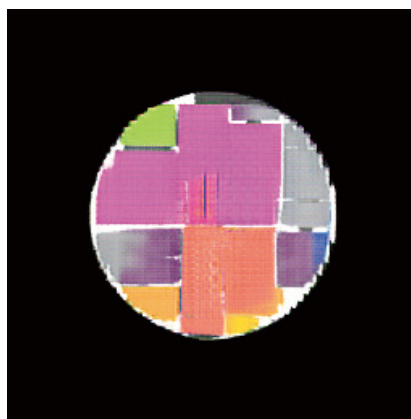


Figure 5

Experiments

Facades

The method of image to image translations outlined in the (Isola, P et al. 2017) pix2pix paper can be applied to a multitude of systems. In the paper they show an example of facade generation made from the CMP facade dataset (Tyleček & Šára, 2013). This was slightly modified to better fit with the floor-plan dataset created for this paper.

It works though exactly the same

principles as the floor-plans but now applied to another set of relations. Each detail on the facade is given a color and the network is trained on reading these and understanding this relationship. When training on this relationship it learns how to render the facades in a realistic fashion.

Something to note about this dataset is that it consists mostly of classicist facades as can be seen

in figure 4, it thus has a bias towards that specific style. Some modification to alleviate this was made by adding more modern facades as well but it still tends to generate classicist facades both in training and in generation.

Facade labels

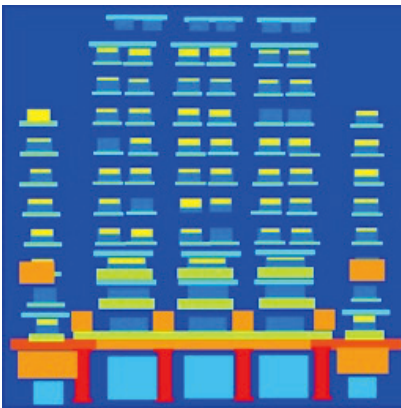


Figure 1

Ground truth

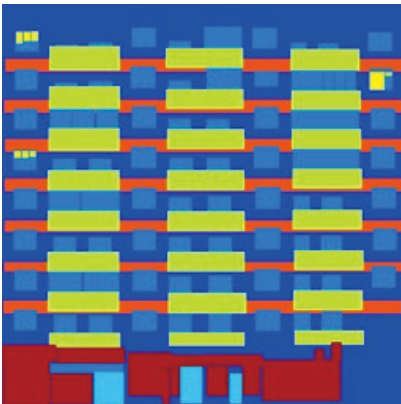


Figure 2

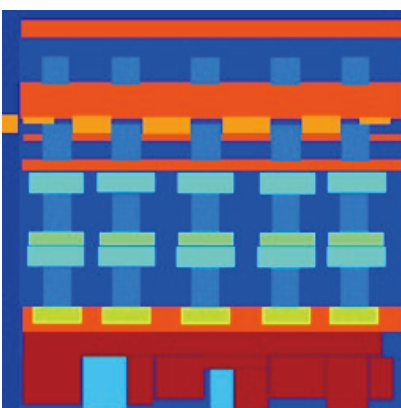


Figure 3



Training the network on the relationship between facade labels and the ground truth works exactly the same as earlier training examples. As the network goes through the epochs of training it begins to form a better understanding of the correlation between the two modes of representing the facade.

One thing to note in this process is that instead of using an input

abstraction of the ground-truth to train on generation of another abstraction as the function map or a regular floor-plan is instead this network is trained on generation of a realistic facade image as would be seen in real life.

As the network is trained on this dataset a sort of internal style of the dataset is created and stored in the model that is created from the training. Similar techniques

are being used in many fields to provide a strong argument for making informed decisions. This could easily be adapted to do very in-depth site analysis to provide strong data-driven design proposals.



Figure 4

Experiments

Facades

As the masks consisted of multi-dimensional networks working in tandem to produce better drawings the facade generation network was linked to the floor-plans as well. As can be seen in figure 1 the facade is split into four parts as to be able to analyze it so that facades can be drawn. Each floor-plan is automatically divided into parts based on the most extreme x and y coordinates of each floor-plan. Then these are

indented a small distance to then broken out to check each part if there are any windows and where they are placed in the image.

After the windows have been broken out a facade is generated as shown in figure 2 and 3 where based on the floor-plan analyzed a facade for side is created. The facades contain as much information as it can get from the generated function maps which are

width of facade and placement of windows. Then a floor count is either randomly generated or chosen by the user in which the facade is then labeled. The height of each facade detail is thus not based on any real life measurements but on relation between width and floor count.

Floor plan



Figure 1

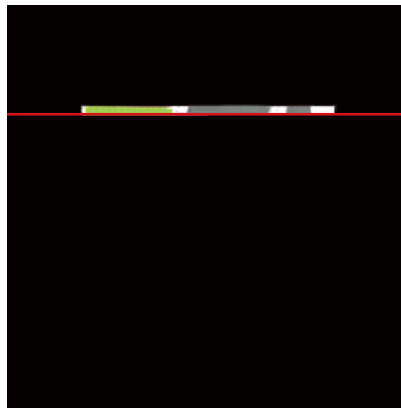


Figure 2



Figure 3



After the process of generating each facade a process of detailing these is made, this is done by a network trained on the relationship between a facade with only the width and height of it's body and window placement. The three parts of this dataset is shown in figure 4.

A network is then able to predict and generate a more detailed facade label map. This detailed

facade label map now contains information on for example roof placement, entrance placement and balconies on the facade.

When this more detailed version of the facade generated from the floor-plan is made it can be ran through a network trained on the relationship between detailed facade label maps and real life photos of facades.

Thus the network will first translate an abstracted floor-plan based on functions to a facade, then generate a abstracted facade with windows and body. From this it will still work with abstractions and detail it to later be able to read that facade and try to create a life-like photo of the facade. All three steps taken on the facade generation is shown in figure 5 and 6.

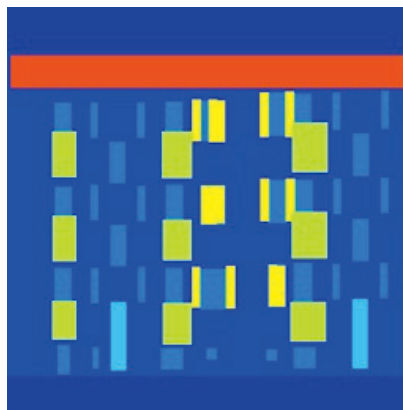


Figure 4

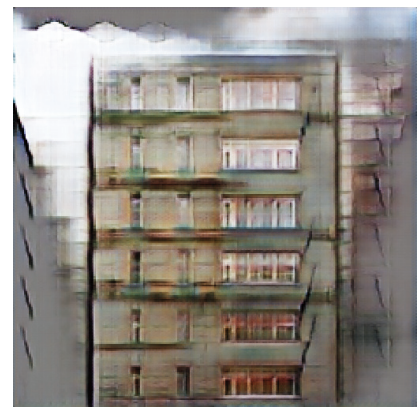
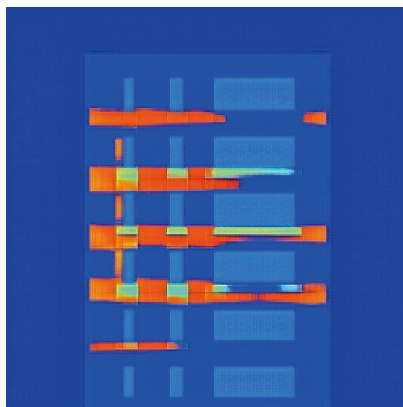


Figure 5

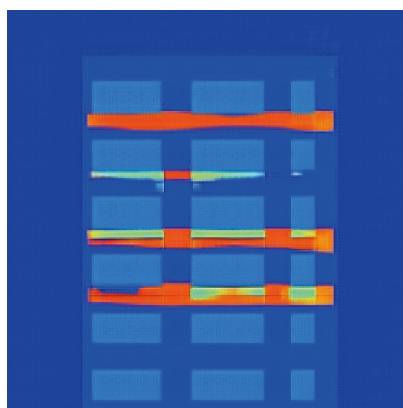
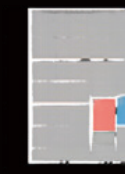
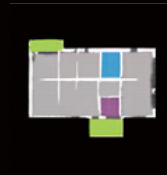
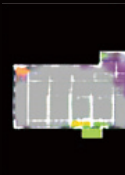
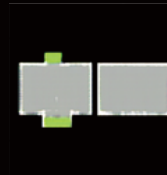
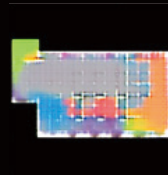
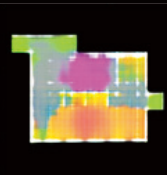
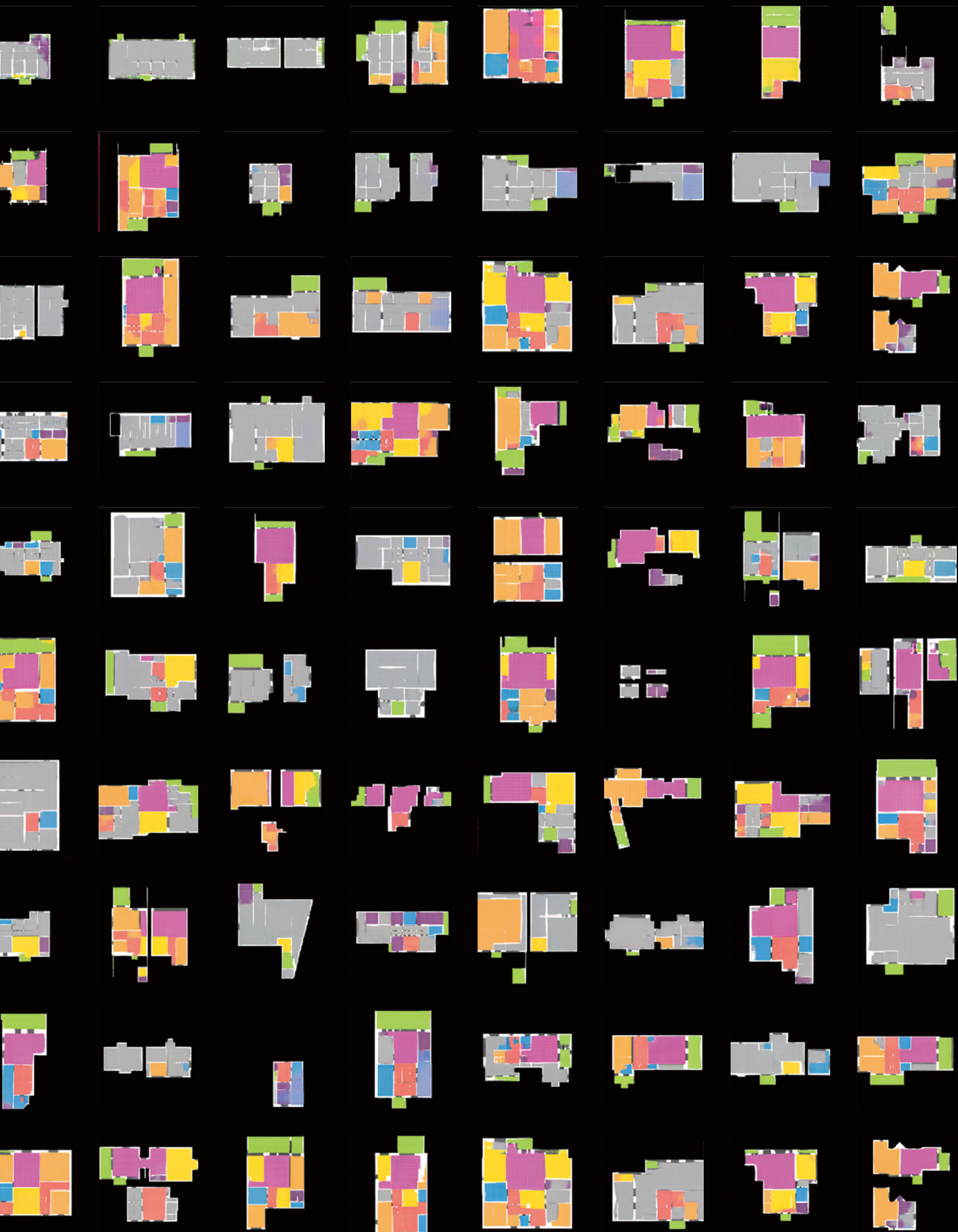


Figure 6

Interviews





Interviews

Anders Neregård - Sweco

Anders Neregård works as chief of digitalization on Sweco architects. Our main topic of discussion was where in an architectural process machine learning should and would be applied.

If an architectural process is simplified as a straight process starting from an idea of a building ending up in a physically built project as shown in figure 1. In the early stages of an architectural process the parameters are fluid and open while the further you travel along this axis of time the more solid and concrete the parameters become. This is especially true in Swedish architecture where the building regulations are very solid and demanding.

Anders and Sweco are currently working on project positioned on both of the extremes on this scale. But views that tools that could position themselves early in the process would be able to aid in the creation of more interesting projects. And that a tool that could generate creative alternatives to a given problem would slot nicely into an architectural design process.

Tools that come later in the process are often the focus of development as they could automate a lot of repetitive tasks that architects currently do and thus save time and money for the builders. While this may be in the main interests of the building companies this may not be true for architects.

Regarding the future of the architectural practice Anders believes there is no direct threat to architects by digitalization of architecture. These tools can be seen as another co-worker in an already big team of competences that architectural projects consists of.

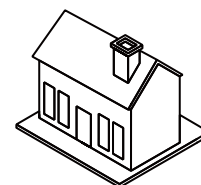
Architects have a strong domain knowledge of organization and structuring of projects.

This knowledge is not something that could easily be replaced by machine learning and would almost be required for it to be applied to an architectural project. What is important though is that architects will own and lead the development of the tools.

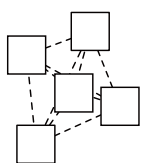
Early design process



Late design process

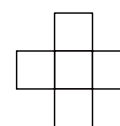
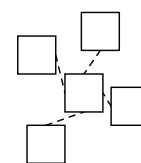


Creative networks



Un-structured

Strict networks



Structured

Figure 1

André Agi - Link

André Agi works as lead Computational Design & Architect on Link arkitektur. Our interview consisted of conversations about machine learning systems and techniques as well as discussions on how they might affect architectural practice.

One of our main topics of discussion was on the position on raster and vector drawings for use with neural networks. André saw potential in the use of machine learning in sorting and presenting vector based drawings by giving each drawing a quantitative value that the network then could learn to sort and process. By contrary this thesis revolves around generation of raster drawings by training the network on relations between often the same quantitative values.

While working in vectors gives a stronger precision to the networks and allows for more direct translations to a 3d environment as most architects work in. A vector drawing is harder to process and thus makes the networks more complex to design and data-set generation becomes more abstruse. Vector based generation or sorting by a network also gives the project a immediate connection to scale, which for direct application to an existing project is key.

A try to mitigate this geometrical disadvantage of working with rasters gave the process, is to link multiple raster based images together to be able to stream line a process to an 3d environment see figure 2. While this still doesn't provide the project with scale it works by relations between the different parts generated through the project.

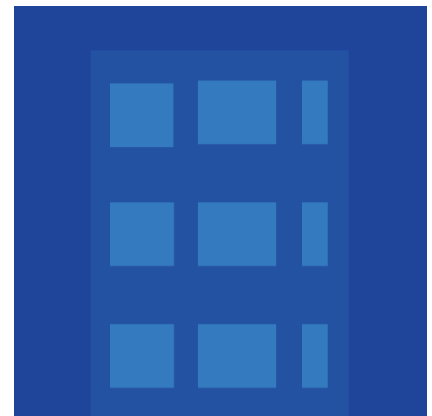
When discussing how architectural practice might be affected by machine learning. André viewed that specialized knowledge would still be high in demand while architects working on later stages in projects could be replaced by automated tools. To make a serialized drawing from an already made bare bones sketch is something a network can be easily be trained on, similar tools are currently being developed.

Raster plan-drawing



+

Raster facade based on relations of plan-drawing



=

Vectorised 3d model

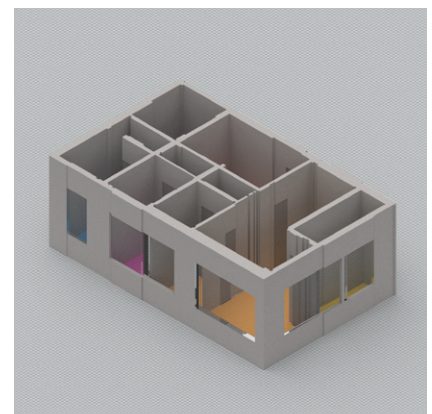


Figure 2

Interviews

Ellen Simonsson & Jacob Flårback - Liljewall

Ellen Simonsson and Jacob Flårback works as architects at Liljewall in the same databased design group. Our discussion was wide and spread but focused on how machine learning can be adapted as a tool and integrated into already existing design structures.

Starting out our discussion jumped into that of raster and vector drawings and how these differentiate the process of generation. Ellen thought that when working with raster drawings the tools might not know scale but they still work with flows and connections. The exact measurements might not be informed but the placement and relations between functions are.

Jacob worked with similar processes where by automated analysis of floor-plans drawings get scores on how functions relate to each other. A similar approach could be applied to the generation from the networks where an analysis on function connections and flows could be done see figure 1.

When looking forward towards the future and how machine learning might shape the architectural practice. Both Ellen and Jacob expressed a large optimism and skepticism towards the current trends. Both agree that digitalization and automation is the future of architecture but the implications of such processes might not lead to improved architecture.

Jacob saw it as mundane and repetitive tasks are replaced by automated systems the profession might move towards being more artistic as workload is shifted from production to design. Such a shift and also with current improve-

ments towards manufacturing complex geometry could move us towards more freedom of expression. While Ellen also saw the potential in this shift a worry about soft values such as social sustainability was hard to quantify and could be lost or shoehorned in to automated processes. Overall it was agreed upon that architects needs to own and develop these tools and not external parties.

Flow analysis

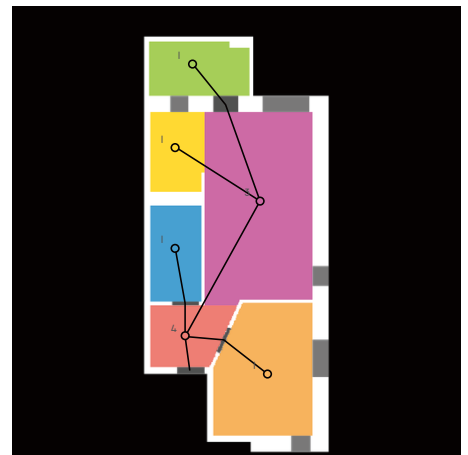


Figure 1

Connection analysis

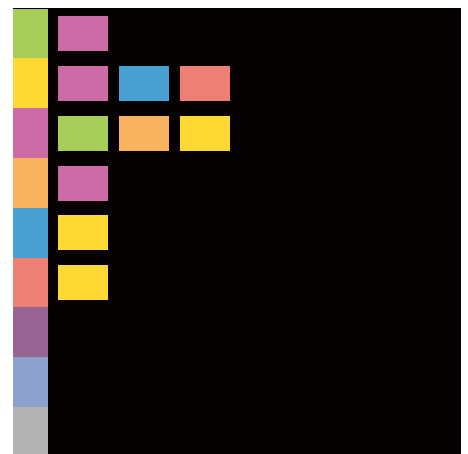


Figure 2

Ola Dellson - White

Ola Dellson work as an architect at White architecture with a focus on digital manufacturing. Our discussions was focused around how automation might lead to standardization.

Swedish building industry is centered around a few strong actors, these few actors produce the majority of the buildings built in Sweden. All of these actors have their own factory where they produce building elements used in construction, they are also the ones leading the development in automation and machine learning in the Nordic region.

Ola argues that in regards to a few big actors having a close to monopoly on the construction industry the tools for automation will be shaped around what their factories are able and have been producing. This issue might lead to an increase in non contextualized and standardized architecture rising in Sweden. If a neural network would produce to complex drawings that doesn't fit into the molds used today it won't be able to be built.

Context is one of the main parameters that differentiate architecture from other manufacturing industries. A rise in non contextual and standardized architecture Ola sees as a mayor problem in the development happening across in particular Sweden.

This is in short an issue with companies building in bias into their tools that they provide to architects. As a technology it isn't flawed just as tools used by architects today but when applied to the real world it can be shaped to deliver uniform and highly directed results.

All this then translates into a direct threat towards architects as automated tools that can generate drawings for an already existing infrastructure of production can easily lead to architects losing jobs. A shift in focus on developing the tools and means of production for architects in the future is then argued to being a necessity.

3d volumes based on input with slight differentiations

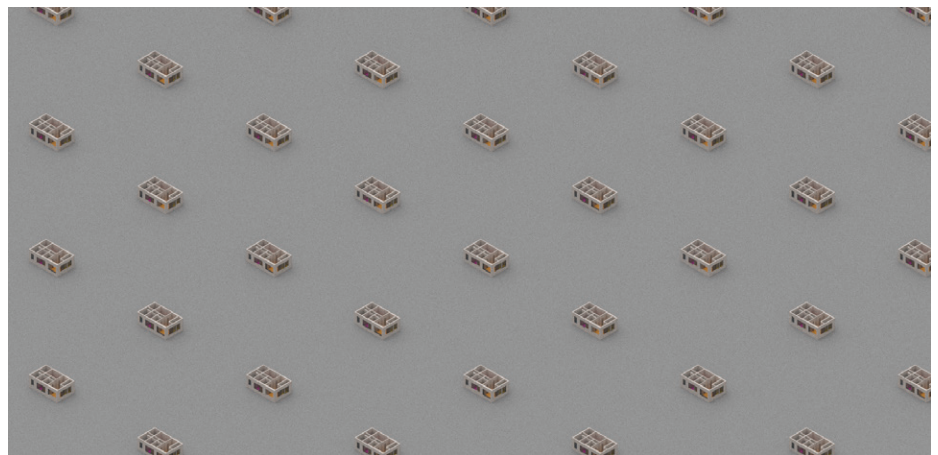


Figure 3

Optimizer counteracting repetitive predictions

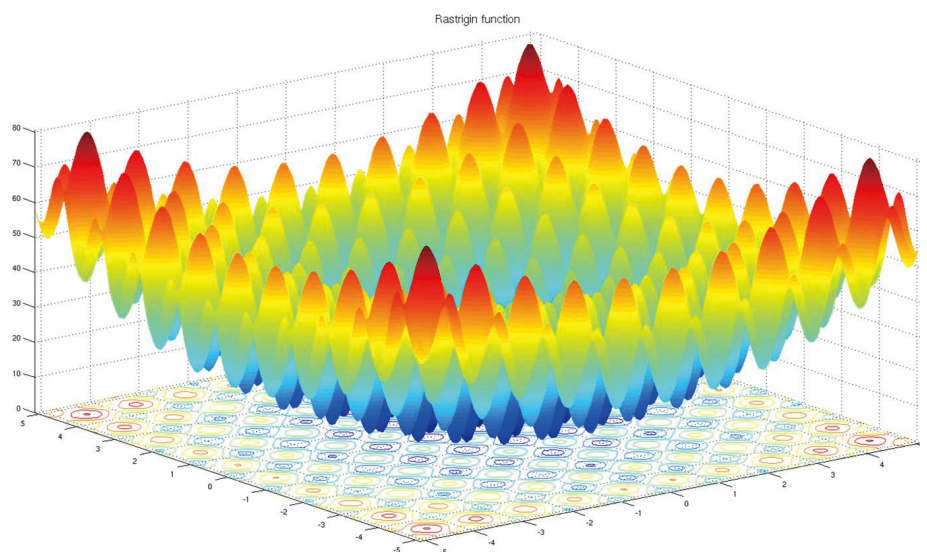


Figure 4

Interviews

Sander Schuur - Belatchew

Sander Schuur works as an architect and creative director on Belatchew architects. The discussion revolved around creativity and soft parameters in architecture.

Sander sees a direct connection between how the development of machine learning correlates to the development that was made in parametric tools in the early 2000s. Starting out as a tool to ease workflows that was relevant at the but through creative usage of the tool a new type of complex and parameter based architecture emerged.

Through a profession striving for creative approaches and application of tools machine learning could become something that architects use a integral part of their everyday workflow. Sander strongly believes that even though the tools at a start may not be meant for creative generation architects will adapt and "hack" the tool to be able to accommodate for much more than was intended at a start.

While creative networks within machine learning is an emerging practice not much work has been done on how these networks can be applied to design processes. While a neural network is based on logic these logics are trained and learned by the network similar to how an artist gathers inspiration and experience from life and practice.

Even though Sander acknowledge that great art is as much or even more based on cultural context than individual skills of the artist, he sees a potential in the amount and speed of generation a network provides.

While a sense of an artist back-

ground can be sensed in a project, a network could be trained in a similar approach by not limiting it to replicating a style or logic. But applying parameters and vectors of soft data such as local culture or global trends, the networks could become more than the sum of its parts. All through the interview a picture of a bright and optimistic future was painted with potential to be creative of new styles and creativity.

Intended workflow

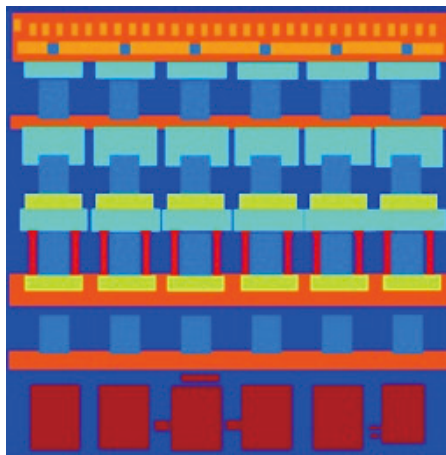


Figure 1



Unintended workflow

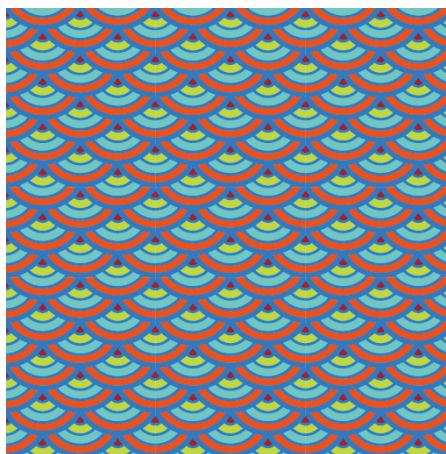
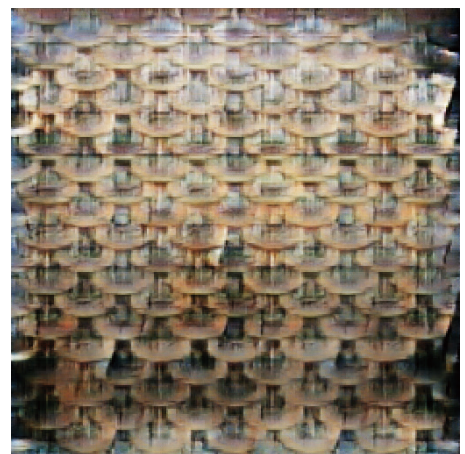


Figure 2



Alison Petty - White

Alison Petty works as BIM-manager on White with a focus on managing and developing workflows she is originally from America and has deep insight in American building praxis. In the interview we discussed regulations and cultural context.

As our discussions on the subject of automation started we quickly began discussing Swedish building regulations, Alison saw them as very limiting framework and any tool developed for a Swedish market would have to be centered around this standard. She saw this as a problem as many of the biggest developers are situated abroad and would then follow their own standards and culture.

With a very practical view of regulations Alison argued that not only would the local regulations have to be integrated into any tool but also cultural standards. As this thesis is mainly built around a Finnish context as would any tool have to be able to adapt to a changing local context. As the shift of building standards and local contexts vary widely over the world as would a tool have to be able to do the same.

If this would not be integrated we could see a true international style where a tool could shape architecture on a global scale. As Alphabet (googles parent company) develop their smart city solutions (Whitney, 2020) these movement have already started.

Context is a hard issue to tackle for machine learning as there are so many moving parts, while machine learning are good at solving high complexity problems it has problems with adapting to new environments.

A belief that automated tools will be an integral part of an architectural practice in the future a worry for the practice was expressed. Especially for Swedish architects that work mainly as consultants with hourly rates when processes can be done more quickly and effectively by automation the time saved might not be allocated to other architectural qualities but instead for making a leaner budget.

Base 3d model

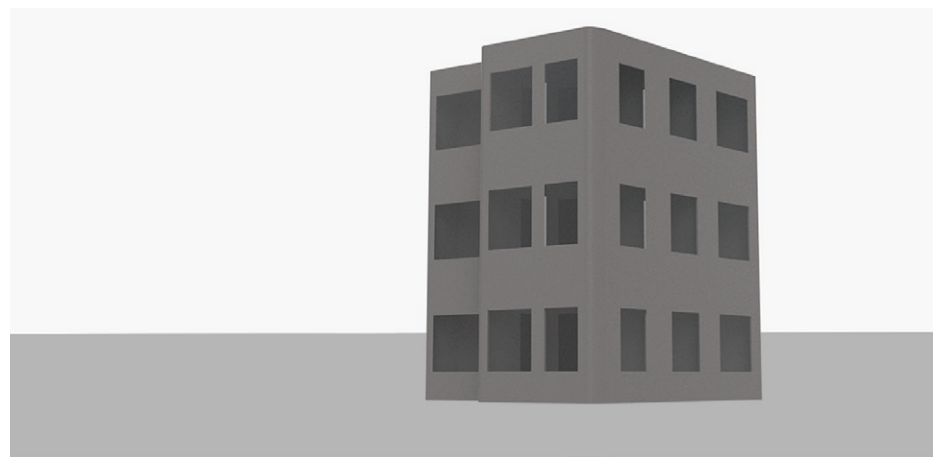


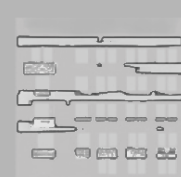
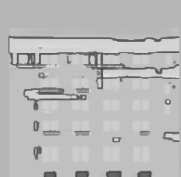
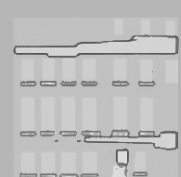
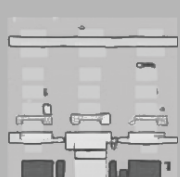
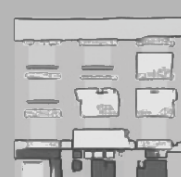
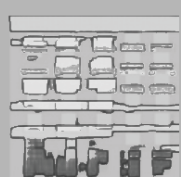
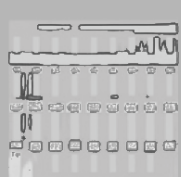
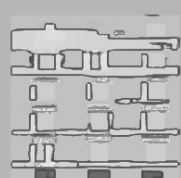
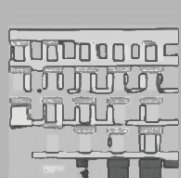
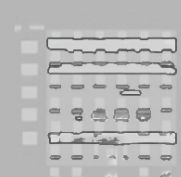
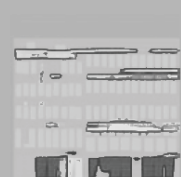
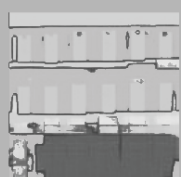
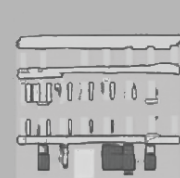
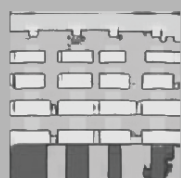
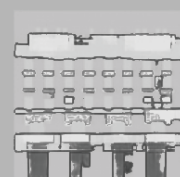
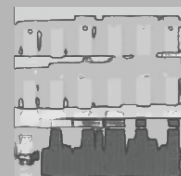
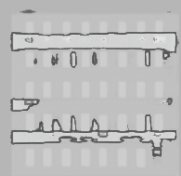
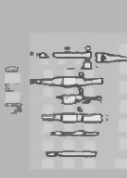
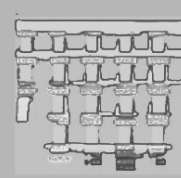
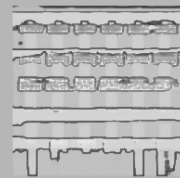
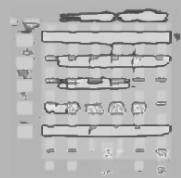
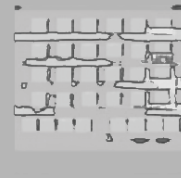
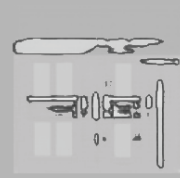
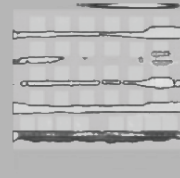
Figure 3

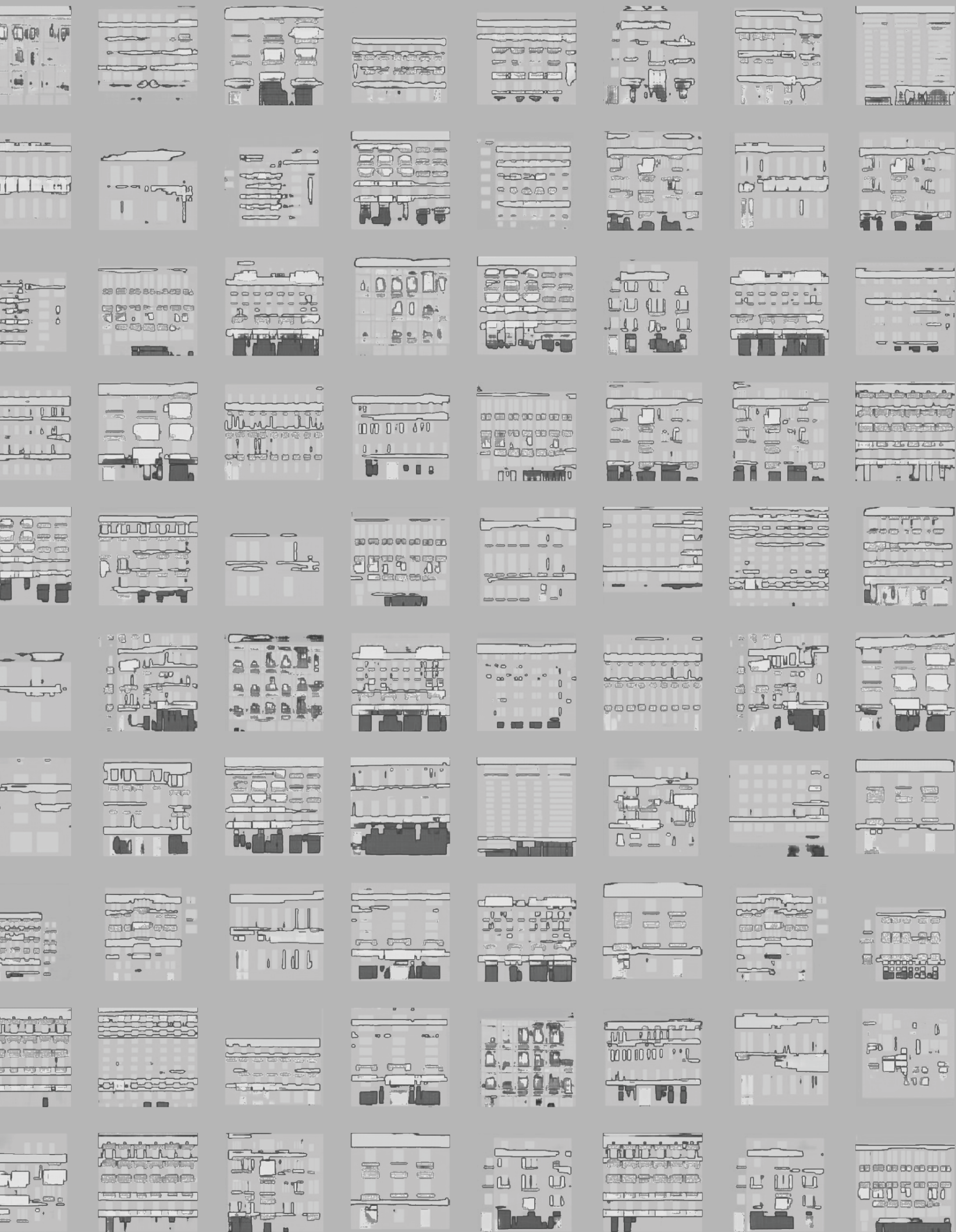
3d model placed into context by CNN



Figure 3

Conclusion





Conclusion

As machine learning becomes a more common tool in digital environments it will affect architecture as well. Most of an architect's work is already in a digital space as it has become the new praxis of development. While machine learning and neural networks is old a concept the ease of application and integration by increased computing power have revolutionized data-based technologies.

While this will provide new tools and opportunities for architects this will also mean a demand for more adept technical knowledge in architects. One can see a trend of architects embracing more digital tools and machine learning

will probably just become another tool to put into the architectural tool-belt.

While automation is already incorporated into the architectural design process through processes such as day-light analysis and wind studies, machine learning will be able to enhance this beyond the boundaries these tools work within today.

Through this thesis an exploration on how machine learning can be integrated into an already existing architectural design process has been made. Starting from a basic sketch to renderings of a three dimensional building.

Machine learning has been used as a base throughout the whole process mainly as a space organizational tool other applications such as context analysis and 3d modeling has been explored. While the outputs might be lower in quality than what would be expected of such a process it still shows the potential of machine learning in an architectural framework.

As with all new technology and methods this shift will come with benefits as well as limitations. These identified risks and potentials will be brought up in here.

Some aspects that either showed potential or was found from studies made in the development of this thesis are highlighted below.

Context analysis was a subject that emerged late in the process but showed potential to further develop. Either by doing facade analysis and look at dimensional- and expression-relations or by doing a more free approach of adapting a proposal to fit into a defined context. Both of these processes could easily be sorted and given parameters to create a data-driven context analysis of a site.

It would have been interesting to

develop these methods but as machine learning requires big data-sets and there was no time to do the prerequisite work that was needed to achieve this. I think that with increasing amounts of open data sources these methods could become corner stones in architectural analysis.

Another points of interest was in giving networks a higher detail of information. This thesis has mainly worked with low information inputs to then further detail these through neural networks. Further studies on how networks can complete and improve on detailed drawings would be an interesting approach as it would

be closer to an actual implementation of the technology. An example of this could be reading drawings in multiple floors to add ducts and connection points between floors, modifying and adapting structural supports and automated analysis of flows and connections in floor-plans.

Vector drawings is also a part that would have been interesting to expand on, this might very well be how architects first come into contact with machine learning as the main bulk of architectural practice is done in vector formats. Working directly in vector would also ease the transition from 2d to 3

Conclusion

A series of potential risks that might come from integrating machine learning into the architectural process has been identified as well.

How automation might lead to standardization of architecture was a subject discussed heavily in the interviews conducted. Machine learning as a technology has a structure that promotes the creation of standardized solutions. When a solution to a problem is found the network will try to replicate that solution indefinitely as long as its not specified to avoid repetition. This inherent gravity towards finding standardized solutions could then be manip-

ulated by developers to push products or solutions.

As an issue this needs to be discussed and addressed by the architectural community, as there are ways to have a network generate novel and new solutions as long as it's built in to the system. I strongly believe that creative applications of neural networks will push the development towards more open ended generation.

Another issue would be that neural networks often work in more than three dimensions adding data that can be hard to interpret for the users. Just as when BIM was first introduced and a 2d line

automatically had an extra dimension added to it and unexpected results could then appear for the user the same might happen with machine learning.

As almost all professions have digital tools more incorporated into them a demand in technical knowledge will rise. This might lead to architects that are not used to digital environments getting pushed out of the market, this is already happening today but as an architects tools get more complex so will the demand in knowledge. Higher complexity of tools might also lead to a lack of control for architects.

In conclusion, I believe that machine learning will profoundly change how architects work by providing new tools that can be integrated into existing design methods and practices such as space organization and highly parameter driven analysis. These systems will probably be the first ones applied to automated tasks currently done by architects but with time develop into more complex tools.

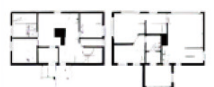
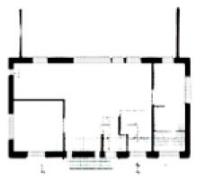
A similarity to how parametric tools can be seen in the potential development of machine learning. As parametric tools was starting out as a system to enhance productivity but through rigorous

development and advances in technology it has enabled architects to produce more complex architecture.

At it's core machine learning is only a new technological advance that will change our profession just as previous advances have. Ramifications for the practice will of course be huge, many architects will have to change their workflows or be outmaneuvered by automation and practices working with highly repetitive tasks might disappear all together. But in the broader perspective this is just a new tool for architects that will be incorporated into existing and new processes.

Therefore I think it's very important that architects engage in the development now while the technology is still in it's early stages. Not to stop the development but to be able to shape it into a tool that can inspire creative and novel solutions while also maintaining the current level of transparency that exists in machine learning. While as a subject it might seem daunting but I urge everyone to try and explore this subject either through reading or by trying it out.

All code done in this project is open source and can be found on <https://github.com/FabianSyber/DeepArchitecture>.



Codebase

Description	Link
Main branch consisting of opencv tools and pix2pix code	https://github.com/FabianSyber/DeepArchitecture
Forked version of CubiCasa5k both for adapting to jupyter but also changes in processing	https://github.com/FabianSyber/CubiCasa5k
Forked version of CubiCasa5k changes made to easier loop floor plan analysis	https://github.com/FabianSyber/DeepFloorplan
Neural network pix2pix based training code, this is the main code for training.	https://github.com/FabianSyber/DeepArchitecture/Pix2PixTrainer.ipynb
Neural network pix2pix based generation code, this is the main code for generation.	https://github.com/FabianSyber/DeepArchitecture/Pix2PixGenerator.ipynb
Opencv based processing script that analyze floor plans and creates facade labels from them.	https://github.com/FabianSyber/DeepArchitecture/facadeLabelGenerator.ipynb
Opencv based script that masks out windows and bodies in the premade facade dataset.	https://github.com/FabianSyber/DeepArchitecture/facadeMasking.ipynb

Description

Link

Opencv based processing that create outlines from masked floor plans.

<https://github.com/FabianSyber/DeepArchitecture/Outline.ipynb>

Opencv based processing that masks out areas of interest in floor plans.

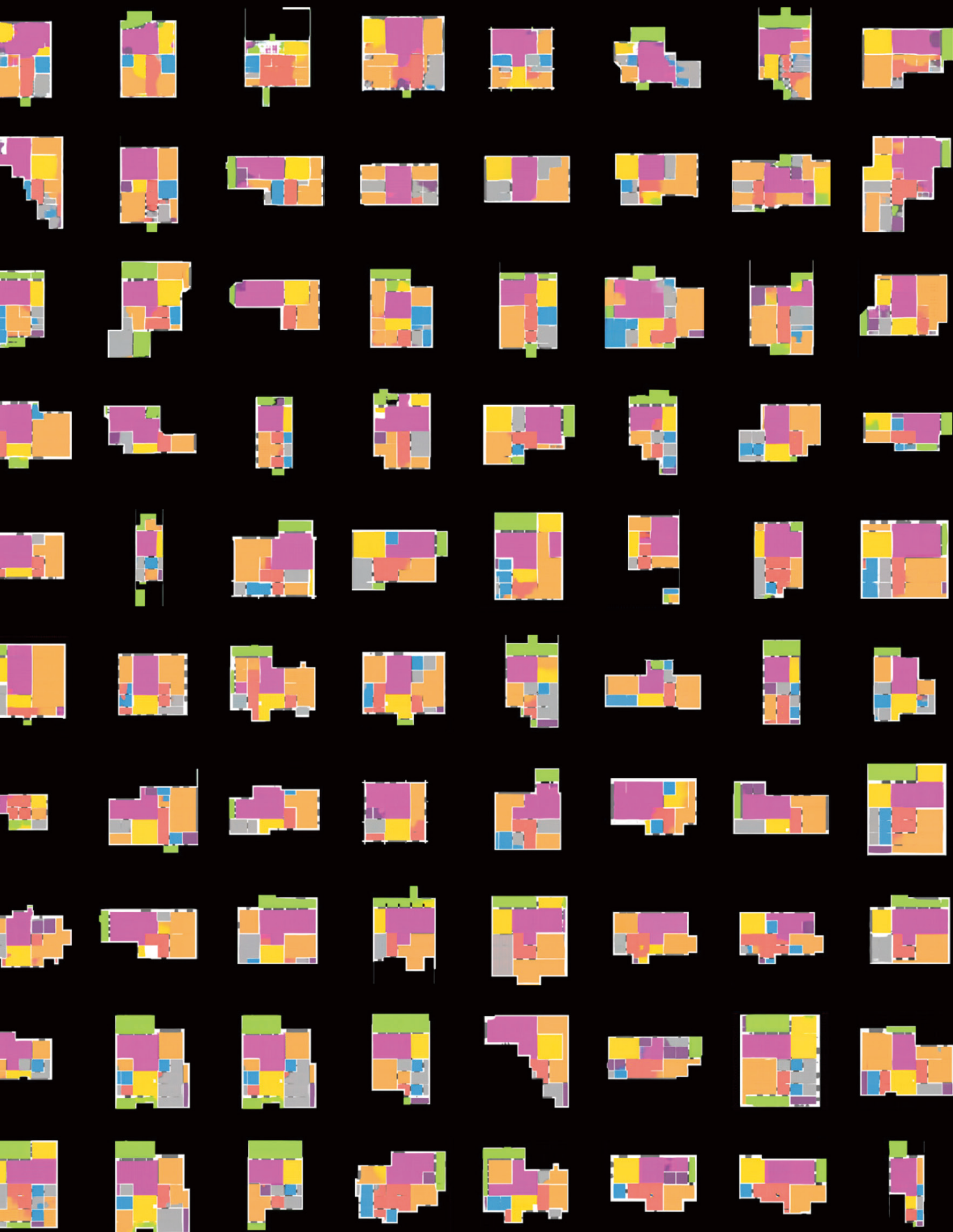
<https://github.com/FabianSyber/DeepArchitecture/MaskGenerationCubicasa.ipynb>

Sorting script for use with CubiCasa5k style drawings, analyzes amount of floors and undefined areas,

<https://github.com/FabianSyber/DeepArchitecture/FloorplanSorting.ipynb>

Modifies SVG files provided by CubiCasa5k to remove unnecessary information.

<https://github.com/FabianSyber/DeepArchitecture/svgModifier.ipynb>



Bibliography

Sebastian Raschka, Vahid Mirjalili (2019). Python Machine Learning. Birmingham: Packt Publishing Ltd.

Benjamin Planche, Eliot Andres (2019). Hands-On Computer Vision with Tensorflow 2. Birmingham: Packt Publishim Ltd.

David Watkin (1986), A History of Western Architecture. London: Laurence King Publishing

Giuseppe Bonaccorso (2018). Mastering Machine Learning Algorithms. Birmingham: Packt Publishim Ltd.

Carmo, M. (2017). The second digital turn: Design beyond intelligence. Cambridge, MA: The MIT Press.

Gatys, L., Ecker, A., & Bethge, M. (2016). A Neural Algorithm of Artistic Style. *Journal of Vision*, 16(12), 326. doi: 10.1167/16.12.326

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio (2014). Generative Adversarial Networks. NIPS Proceedings. Retrieved from <https://arxiv.org/abs/1406.2661>

Zhiliang Zeng, Xianzhi Li, Ying Kin Yu, Chi-Wing Fu (2019). Deep Floor Plan Recognition Using a Multi-Task Network with Room-Boundary-Guided Attention. *International Conference on Computer Vision*. Retrieved from <https://arxiv.org/abs/1908.11025>

Kalervo, A., Ylioinas, J., Häikiö, M., Karhu, A., & Kannala, J. (2019). Cu-biCasa5K: A Dataset and an Improved Multi-task Model for Floorplan Image Analysis. *Image Analysis Lecture Notes in Computer Science*, 28–40. doi: 10.1007/978-3-030-20205-7_3

Brewer, C. A., Hatchard, G. W., & Harrower, M. A. (2003). Color-Brewer in Print: A Catalog of Color Schemes for Maps. *Cartography and Geographic Information Science*, 30(1), 5–32. doi: 10.1559/152304003100010929

Tyleček, R., & Šára, R. (2013). Spatial Pattern Templates for Recognition of Objects with Regular Structure. *Lecture Notes in Computer Science Pattern Recognition*, 364-374. doi:10.1007/978-3-642-40602-7_39

Botella, Marion & Zenasni, Franck & Lubart, Todd. (2018). What Are the Stages of the Creative Process? What Visual Art Students Are Saying. *Frontiers in Psychology*. 9. 10.3389/fpsyg.2018.02266.

Horn, J-T. (2020, April 7). Analyzing environmental noise in Spacemaker. Retrieved from <https://blog.spacemaker.ai/analysing-environmental-noise-in-spacemaker-b2d1d89b6c49>

Khandelwal, R. (2019, February 3). Overview of different Optimizers for neural networks. Retrieved from <https://medium.com/datadriveninvestor/overview-of-different-optimizers-for-neural-networks-e0ed119440c3>

Whitney, V. (2020, January 3). A first step toward the future of neighborhood design. Retrieved from <https://www.sidewalklabs.com/blog/introducing-pmx-model-tall-timber-buildings-cities/>

Diederik P. Kingma, Jimmy Ba (2014). Adam: A Method for Stochastic Optimization. 3rd International Conference for Learning Representations. Retrieved from <https://arxiv.org/abs/1412.6980>

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-Image Translation with Conditional Adversarial Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi: 10.1109/cvpr.2017.632

