Design Optimization Through Machine Learning Algorithm

ABSTRACT
Within the context of continuous technology transformations, the way scientists and designers process data is changing dramatically from simplification and explicit defined rules to searching and retrieving. Ideally, such a trending method can eliminate issues including deviation and ambiguity with the help of hypothetically unlimited computational power. To process data in this manner, artificial intelligence is necessary and needs to be integrated into the design process. An experiment of a design process that consists of a generative model, a data library, and a machine learning system (GAN) is introduced to demonstrate its effectiveness. The methodology is further evaluated by comparing its output with its input targets, which proves the possibility of employing machine learning systems to aggressively process data and automate the design process. Further improvement of such methodology, including judging criteria and possible applications, and the sensibility of the machine is also discussed at the end.

KEY WORDS
Machine Learning, Automation, Variables, Data Processing, Sensibility, Generative Design
INTRODUCTION
With technological transformations taking place every day, as well as the onset of machines to help aid in the process of design, artificial intelligence, automation, and machine learning are gradually becoming inevitable apparatuses in almost every discipline, including architecture. As Mario Carpo states, instead of compressing data into simple rules and patterns, scientists and designers are trying to straightforwardly handle and utilize all the discrete data by using the almost unlimited power of computation (Carpo 2014). This trending mindset provokes us to delve into understanding how machine learning algorithms could be integrated into digital design, utilizing the ability of the machine to learn and process huge amounts of data in a small amount of time. An automated process of designing a folly is proposed to demonstrate and evaluate the goal of integrating machine learning algorithm into the design process.

BACKGROUND
Throughout the short history of artificial intelligence, it has always been easier for a machine to help in problem solving than to create a design from scratch. The deductive nature of traditional algorithms and their application makes it extremely hard to mimic the seemingly random process of human creative thinking. However, computational artist Parag Mital proposed a provoking approach to such issue (Mital 2013a). Instead of training a machine to create a design from scratch, he trains the machine to start from subtracting existing video clips as elements that he then uses to create a whole new animation in real time. The traditional process of creative design is thus decomposed into several phases during which the machine is only trained to perform the specific phases at which it is good. In his case, Mital trained the machine to develop its own way to select video clips from a source pool and then compose them in a creative manner with no absolute (Figure 2). Following this logic, the question for us becomes which phases during a creative design process are the ones that need a machine with learning abilities the most and how machine learning could be integrated into design processes.

In most cases of digital design, the phases involving repetitive tasks and intensive labor are done by scripting. Such phases usually have predetermined goals, and the script simply realizes them by following human logic using computational power. The script essentially consists of two different kinds of components. The first component is ‘functions,’ which are usually Grasshopper components or Python definitions depending on which scripting platform is employed. The other component is ‘input variables,’ which often appear as number sliders or floats, also depending on which scripting platform is employed. Although these two phases relieve designers from repetitive labors, the phases themselves are still human-created and sometimes extremely time-consuming.

Considering the fact that writing functions involves highly complex skills and thinking processes, training a machine to write functions independently is hard to achieve at this moment. However, the phase of adjusting input variables is relatively simple and repetitive. After developing an entire script, designers still need to look at the generated configurations, then manually adjust every variable accordingly and repeat this process until the design meets their goals. In scenarios that have more than 10 variables to generate a configuration, manually adjusting every variable is extremely time-consuming, making it practically impossible to find an ideal combination. Greg Lynn calls it a ‘happy accident’ that designers can only wait and pray for a result to be satisfying (Lynn 2008). Therefore, the most technically essential part of this experiment is to train a machine to ‘understand’ a designer’s preference and adjust every variable value based on its understanding.

METHODS
1. Generative Model Scripting
To start for the work discussed here, a Generative Model Script is created to output configurations automatically. The basic logic is telling the Rhino program to pick several 3D elements from the already-built source pool and compose them into a whole. Twelve elements are manually created to include as many varieties as possible. Some
of these elements have architectural references—such as traditional Chinese details, Mies Van Der Rohe’s cross column, and Frank Gehry’s sculptural geometries—and some have references outside of architecture, including typical manga and futuristic shapes (Figure 3).

There are also 12 variables that control the output configurations, including Size which affects the size of every and each element. Dance that affects the degree of random movement of every and each control point. Index that determines which element from the source pool is being selected to compose the configuration, etc. (Figure 4). The above elements, variables, and a set of morphology functions together perform as a generative model that is able to catalyze 3D configurations.

2. Library Creating and Labeling
To generate enough number of configurations and corresponding data for machine learning, a mechanism is inserted to automatically generate unbiased random variables for the script that will grab them and generate corresponding configurations. Every and each set of 12 variables is recorded in an Excel file, as every configuration generated is taken as a screenshot in render mode automatically.

In order to train the machine to understand a designer’s preferences, all the output configurations from the library are then labeled by each designer based on certain criteria. The labels are then recorded into the Excel file aligned with corresponding variables. In this experiment, 8,000 configurations and 96,000 variables are created and recorded simultaneously (Figure 5).

3. Machine Learning Operation
To set up a system that can produce configurations to match the designer’s preference, a machine learning system based on a Tensorflow platform is introduced in the analysis process. Generative Adversarial Networks (GANs) are the key strategy applied in the machine learning system to generate preferred variables in this experiment.

3.1 Introduction of GAN
GAN is a type of Generative Model. Its function is to grab the training data from the library and learn the probability distribution from that data (Goodfellow 2014). The system consists of two basic modules. One is Generator Network, which is used for generating variables based on certain logic—in this experiment each and every set of 12 variables can construct a corresponding configuration. Another module is the Discriminator Network, which is used to identify the configurations trained by the Generator Network or human designers.

The goal of the Generator Network is to try and generate as much data as possible that is not able to be identified by the Discriminator Network; the goal of the Discriminator Network is to try and identify the data that is not similar to the input data. The key working strategy of GANs is to allow the two modules to compete, iterate, and ultimately optimize outputs. The final output of the GAN system can be as close as possible to a designer’s preference.

3.2 Generative Adversarial Networks (GAN) Operation
The machine learning process in this experiment can be divided into three stages. The first stage is setting up an initial Discriminator. Initial configurations generated by the Generator are manually labeled according to certain criteria. The labeled data are then imported into the GAN system to get an initial Discriminator. The second stage is optimizing the Generator. The weight of different parameters is adjusted due to the identifying information which is
produced by the initial Discriminator. Thus more data can be generated that is more likely to ‘cheat’ the Discriminator. The third stage improves the accuracy of the Discriminator based on data newly generated by the Generator. Stages 2 and 3 will iterate with each other repeatedly, and finally, the system can train a Generator which generates data that meets designer-labelled criteria (Figure 6).

3.3 Setting up the Initial Discriminator
The Generative Model Script based on Grasshopper is the initial Generator. It generates unbiased random variables, and every 12 variables construct a configuration. In this experiment, the initial Generator generates 2000 groups of random variables which coincide with 2000 configurations. These configurations are regarded as the Library. The initial library is manually labeled as good or bad (“good” is labeled as “1” while “bad” is labeled as “0”). The configurations labeled as “1” account for around 20 percent of the total Library and the “good” configurations are defined as Target. The manual labeling process is based on the rendering images of those configurations in that the designer can only label the data through visual impressions rather than abstract data. When it comes to the subsequent machine learning operations, configurations are analyzed based on their 12 variables, and those operations are purely numeric.

To simplify the whole experiment process, the machine learning process can be regarded as a black box, and the whole process is shown below (Figure 7):

- Generate unbiased random configurations from Generative Model Script
- Manually label the configurations based on certain criteria by a human designer
- Input the labeled data into the Black Box
- The Black Box is trained by the labeled data and iterates itself, and finally outputs configurations which match the designer’s preference.

RESULT & EVALUATION
To further evaluate the effectivity of the system, terms with architectural meanings are selected as both labeling and evaluation criteria to be tested.
On one hand, simple and straightforward criteria—including Big, Thin, Tall, Blue, Red, all of which with clear definitions that could be either perfectly transferred to numbers or verbally articulated without ambiguity—are employed to test the system’s ability of understanding simple notions and achieving straightforward goals.

According to the distribution maps, the trained machine has a clear understanding of each term and possesses the ability to create configurations that precisely meet the criteria (Figure 8). On the other hand, sophisticated criteria, including Delicate, Monumental, Dynamic, Solid, etc., which to a certain extent are abstract and could not be perfectly articulated or transferred to a simple number, are employed to test the system’s ability of understanding sophisticated notions and achieve open-ended goals.

Using sophisticated criteria, in this case, brings up two essential benefits. Firstly, these criteria are judged based on multiple attributes and dimensions that can hardly be processed by any traditional algorithm, which could better test the machine’s advantages. For instance, a traditional algorithm could easily ‘understand’ the notion of Big by detecting the range of the variable Size, which perhaps is bigger than a certain value (say 0.5), but could not return any result from the notion Monumental, because in that notion a variable is not set as a specific range correlated with all the other variables. The GAN system, on the contrary, can handle the job without any obstacle by treating the variables multi-dimensionally. Secondly, such criteria could better test the method’s advantage of treating human vision and computer vision correspondingly. Sophisticated criteria are difficult to be verbally articulated or defined but could be well judged by visual representations. For instance, it is hard for a human being to quantify the notion of ‘Monumentality’ into computer-readable variables, but it is easy for one to judge whether a certain 3D configuration conveys the sense of ‘Monumentality’ or not by simply looking at its image while at the same time let the
machine read its corresponding data, leaving each player of the game focused only on what it is good at (Figure 9).

Comparison between each target and output configuration shows three interesting facts. Firstly, the output configurations are visually similar to the target ones as is expected—the higher the prediction value is, the stronger the similarity becomes. Secondly, within a certain range of prediction value, the machine could output surprising and criteria-meeting configurations with fewer similarities but architecturally more provoking moments with the targets. Lastly, the outputs, based on the same criteria, could vary dramatically among different individuals who label the data, which shows the machine’s ability to understand the architectural notions with designers’ own subjectivities and sensibilities (Figure 10).

DISCUSSION
Judging from the results mentioned above, the proposed methodology processes data in a new aggressive way, and by doing so, eliminates the repetitive labor of manual adjustments and selections of a traditional generative algorithm. Designers do not need to wait and pray for their preferred results anymore; they can simply skim through an automatically generated library and then let the machine generate a pool of customized outcomes that meets their goals.

Such a system could be employed in design collaboration between humans with varying design ideologies. People from different geographies or cultures of the world are able to come together, design in a cooperative alliance where the system facilitates the interaction of designers in the most amicable form, and help generate a common design solution by making designers contribute judgments equally during the supervising process without requiring a dominant designer who makes every final decision.

As a design decrypter, the system could be an executant that helps understand the sensibility of an untrained human, such as a client, who may lack the architectural jargon or vocabulary to fully present his or her thoughts. The system could act similarly to what a Google translator does, but instead of converting spoken languages, it would interpret a client philosophy into a format that is easily legible to the designer.

Another utility of this system could be seen in novel geometric generation, which eliminates the necessity for a preexisting precedence to start the design process, as well as unexpected repetition, unintentional plagiarism, and situations where designers fall into their own mind sets and narrow down their scopes.
However, with that said, it is noticed that the methodology could be further improved in several aspects.

Under current labeling process, designers might find gray areas where a certain configuration neither meets the criteria sufficiently nor fails to meet the criteria entirely. Improvements could be made by adding a typical rating mechanism which provides designers with a scale, for example, from 0 to 5. Such a rating mechanism would ideally eliminate the gray areas and make the target data more precisely reflect the designers’ preference.

Moreover, the experiment brings up a question that needs to be further discussed. It is apparent from the results that subjectivity and sensibility vary between each designer. However, is it possible to merge two supervised learning data from two different designers into a shared one? Answers to such question would open up new possibility of merging two different aesthetic styles or sensibilities together into one uncanny and interesting result.

ACKNOWLEDGEMENTS
This paper is based on the thesis work of Henan Liu, Longtai Liao, and Akshay Srivastava at Taubman College, University of Michigan. Thanks to the instruction of Matias del Campo and Sandra Manninger and the technical support from Alexandra Carlson and Feifan Liao.

REFERENCES


IMAGE CREDITS
Figure 2: Visual Synthesis Demonstration (Mital 2013b).
Figure 11: Render background is downloaded from https://wall.alphacoders.com. Author: Unknown, Date: 4/18/2019. All other drawings and images by the authors.

11 Post-Production of the four output configurations with the highest prediction values
12 Physical model of the selected output configurations

Henan Liu received his M.Arch degree from the University of Michigan, and B.Arch degree from Shenyang Jianzhu University. He currently works as an architectural designer at Populous in Kansas City. www.henanliu.com.

Longtai Liao is an architectural designer who received his Master of Architecture degree from University of Michigan in 2019. Currently, he is a designer at Steinberg Hart, San Francisco. He has worked for Stanley Saitowitz and Mark Cavagnero. His design work engages the building environment at novel methods of digital design. www.liaolongtai.com.

Akshay Srivastava received his M.Arch degree from the University of Michigan, and B.Arch degree from Guru Gobind Singh Indraprastha University, India. He currently works as an architectural designer at Solomon Cordwell Buenz in Chicago.