Machine Learning for Human Design:
Developing Next Generation Sketch-Based Tools
by

Bryan Ong Wen Xi

B.S. Civil Engineering
University of California, Los Angeles, 2019

SUBMITTED TO THE DEPARTMENT OF CIVIL AND ENVIRONMENTAL ENGINEERING AND THE DEPARTMENT OF ARCHITECTURE IN PARTIAL FUFILLMENT OF THE REQUIREMENTS FOR THE DEGREES OF

MASTER OF SCIENCE IN CIVIL AND ENVIRONMENTAL ENGINEERING AND MASTER OF SCIENCE IN BUILDING TECHNOLOGY

AT THE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2021

Signature of Author .....
Department of Civil and Environmental
Department of Architecture
May 14, 2021

Certified by .....

Caitlin T. Mueller
Associate Professor of Architecture and Civil and Environmental Engineering
Thesis Supervisor

Accepted by .....

Colette L. Heald
Professor of Civil and Environmental Engineering
Chair, Graduate Program Committee

Accepted by .....
Leslie K. Norford
Professor of Building Technology
Chair, Department Committee on Graduate Students

Machine Learning for Human Design:
Developing Next Generation Sketch-Based Tools

by

Bryan Ong Wen Xi

Submitted to the Department of Civil and Environmental Engineering and the Department of Architecture on May 14, 2021 in Partial Fulfillment of the Requirements for the Degrees of Master of Science in Civil and Environmental Engineering and Master of Science in Building Technology

ABSTRACT

Formal computational approaches in the realm of engineering and architecture, such as parametric modelling and optimization, are becoming increasingly powerful, allowing for systematic and rigorous design processes. However, these methods often bring a steep learning curve, require previous expertise, or are unintuitive and unnatural to human design. On the other hand, analog design methods such as hand sketching are commonly used by architects and engineers alike. They constitute quick, easy, and almost primal modes of generating and transferring design concepts, which in turn facilitates the sharing of ideas and feedback. In the advent of increasing computational power and developments in data analysis, deep learning, and other emerging technologies, there is a potential to bridge the gap between these seemingly divergent processes to develop new hybrid approaches to design. Such methods can provide designers with new opportunities to harness the systematic and data-driven power of computation and performance analysis while maintaining a more creative and intuitive design interface. This thesis presents a new method for interpreting human designs in sketch format and predicting their structural performance using recent advances in deep learning. Furthermore, the thesis will also demonstrate how this new technique can be used in design workflows including performance-based guidance and interpolations between concepts.

Thesis Supervisor: Caitlin T. Mueller
Title: Associate Professor of Architecture and Civil and Environmental Engineering

ACKNOWLEDGEMENTS

I want to thank my mentors, friends, family members, colleagues and anyone who have made this research and thesis possible. Professor Caitlin Mueller, who have inspired me even before I started my journey at MIT and continued to do so throughout my academic journey. I would not be able to have such a fulfilling experience at MIT without her guidance and support throughout the years.

I also thank members of the Building Technology faculty – Professors Leslie Norford, John Ochsendorf, Christoph Reinhart, and Leon Glicksman – for providing such a comprehensive and conducive environment for students like myself to engage fully in our intellectual pursuits, as well as a supportive and inclusive community. I want to also thank students in the BT Lab, members of the Digital Structures Group and previous students in the CEE Meng program, including Renaud Danhaive, Mohamed Ismail, Paul Mayencourt, Yijiang Huang, Stella Zhang, Ashley Hartwell, Eamon Whalen, Eduardo Gascon (and Marta Salvat) , Demi Fang, Keith Lee, Nicole Tang, Sarah Mokhtar, Nada Tarkhan, Ramon Weber, Ang Yu Qian, Elizabeth Young, Herbert Nuwagaba, Diego Rivera, Alex Kawar, Anna Sofia Montoya-Olsson, Ayse Heckel, Daniel Seats, Eddy Sambrano, Ernest Ching, Eric Wong, Grace Melcher, Ishani Desai, Rayna Higuchi, Shiyao Sun, and Zoe Lallas. I truly would not have had such an amazing experience if not for these wonderful individuals. Particularly, I want to give a shoutout to Eduardo Gascon who was not only a great friend but also someone who was always encouraging and engaging both in and outside of the lab. I would like to thank Renaud Danhaive specifically, who's own work inspired my research and has given me appropriate guidance along the way. I am also grateful to have worked with Diego Yanexz-Laguna in the prototyping of the web implementation of the research project. I would also want to thank various MIT staff, specifically Stacy Clemons of the Building Technology Program, Cynthia Stewart from the Department of Architecture, and Kiley Clapper from the Civil and Environmental Engineering Department for making my academic journey as seamless as possible.

I want to also recognize my previous supervisors at Skidmore, Owings & Merill, in particular Abel Diaz, Mark Sarkisian, Kevin Chang, Samantha Walker, Abhinav Bindal, and Yanhong Liu, as well as my mentors during my time at UCLA, Jonathan Glassman, Jonathan Stewart, and John Wallace, all of whom kickstarted my interest in applying new and emerging technologies towards solving challenges in architecture and engineering.

I greatly acknowledge JTC Corporation in Singapore for their generous funding during my undergraduate studies at UCLA and graduate studies at MIT, as well as external programs at Harvard and MIT Media Lab.

I would like to like to thank my friends that have supported me in one way or another. To my childhood friend, Hansel Konstantin Tantohari, thank you for the amazing conversations and company throughout the years. To my canoe teammates, Don Loo,

Khoo Pei Koon, and Sean Tan, thank you for being my second family during my time overseas. Thank you to the friends I made in the Singapore Navy, specifically Nicholas Lee, Norman Soh, and Eddie Leow for the fun times we had and will have when I head back home. And a last thank you to the close knit of friends I made during my undergraduate days, Lew Xuan Yu, Paul Song, Jared Rivera, Nanshan Li, Wasiq Al Mamun, Tracy Saw, and Tenn Shaun Lim for your friendships.

My achievements thus far could most definitely not be made possible without the unconditional love and support from both my parents, Mummy and Papa, who have worked tirelessly in their own way to make me the person I am now. I hope I did both of you proud. And also, my siblings, Jie Jie and Tao Tao, who have stuck by me and put up with my shenanigans throughout the years.

And Natasha, who has been with me through the thick and thin for the past three years. Who's better than us beb?

Table of Contents

1. INTRODUCTION

1.1 Research motivation

There are several key factors that determine the effectiveness of early-stage design in architecture and engineering: being able to convey information quickly, using a medium which is clearly interpretable by parties involved, having the space for creative exploration, and attaining feedback for design feasibility. The first two factors point towards sketching – an easy and discernable way to share ideas of early-stage designs. What is lacking is the ability to move past what is produced through the sketch while keeping in mind the restrictions placed by design and performance constraints. The latter two factors point towards procedural and parametric modelling – creating accurate generative models capable of producing new designs. However, this is computationally and time intensive. This paper presents a new approach for structural design that links hand sketching and performance data that builds on recent advances

in machine learning. By leveraging what each established methodology is designed for, in a medium that all designers are familiar with, the objective is to create a data-driven interface that highlights all the benefits.

1.2 Research question and contributions

This research fundamentally asks the question of whether the current processes in early-stage design of structures can be improved, specifically through the leveraging of deep learning. The following contributions are developed to address the question:

1. Sketch data synthesis pipeline to generate synthetic sketch designs of structures with their corresponding performance scores.
2. Development of methodology to predict relative performance of sketch inputs that are structural and architectural in nature to facilitate the design process of the designer.
3. Creation of new designs based on sketch inputs by designer by generative models.
4. Performance-based interpolation techniques to allow designer, architect, or engineer to filter newly generated designs.

1.3 Research scope

This thesis seeks to apply deep learning techniques in the realm of structural engineering and architectural design to help facilitate the exchange and generation of new design ideas in the early conceptual phase of the design process. Titled as *Machine Learning for Human Design*, the thesis centers around the human user as being key to the design process, and the machine as a tool in which to enrich and empower established practices. The ultimate goal of this research is to improve and streamline current design processes for engineers, architects, and designers alike.

1.4 Organization of thesis

The thesis is organized incrementally, with chapters building upon one another towards a potential user interface with applicable features. Chapter 2*, Background and*

Literature Review, looks at related work done in the field that this research is interested in for inspiration and precedence. Chapter 3, *Sketch-Performance Prediction*, introduces the proposed data synthesis pipeline that was used to generate the data for the training of the models. Furthermore, it describes in detail the combination of the two deep learning models used for the problem posed, and their basic applications. Chapter 4, *Sketc- Design Generation*, delves deeper into potential applications of the models proposed and combines features crucial in machine learning as well as structural engineering and architecture. Chapter 5 concludes the thesis and discusses about limitations and potentials for future work.

2. BACKGROUND AND LITERATURE REVIEW

To address the goal of developing a data-driven interface that combines hand sketching and performance for structural design, the fields of computer graphics and algorithmic structural design are both relevant, whereby advances from each discipline can be applied to the development of performance-driven sketch interfaces. In previous work, sketching in the realm of computer graphics has been predominantly focused on the application of some well-formulated techniques and methodologies in machine learning on sketches confined to structural and architectural representations.

2.1 Sketch Data

To develop a machine learning model, a data set containing necessary information is required. In computer vision, there have been many publicly available data sets of designs or drawings, ranging from ImageNet [1] to MNIST [2]. However, these contain data of a specific type which is not relevant to the problem posed in this paper.

Databases for sketches are also available such as Google's Quickdraw! and SketchGraphs [3] which provides over 15 million real-world CAD models, containing CAD drawings, sketch augmented drawings and the procedural pipeline used to generate each one of the models. These databases, though extensive, currently have no performance analysis component attributed to them, which makes them better suited for research specific in computer graphics and not beyond. Hence, there will be a need to create a new data generation pipeline, which is presented in the paper.

2.1.1 Bitmap Images

Sketch data can be represented is various formats. One of the most extensive representation being bitmap images or pixel-based images. The use of bitmap images is also widely used in deep learning generative models, ranging from generative adversarial networks (GANs) [4] to conditional adversarial networks [5]. Large aggregations of bitmap images arguably also led to the advancements of machine learning, specifically through datasets like ImageNet and associated challenges such as the ImageNet Large Scale Visual Recognition Challenge [6].

2.1.2 Sequential Data

Sequential representation of data is becoming increasingly popular, especially in datasets that relates to hand drawings. This ranges from recurrent neural networks recognizing and generating Chinese characters [7], [8] to common objects [9], and even music generation [10]. Rather than relying on a static image as is the case for bitmap data, sequential data considers the order in which the data is structured. This is typically done as positional vectors for the case of hand drawn characters or objects.

2.1.3 CAD models

Recent advances have shifted data representation from the typical 2-dimensional space to 3 dimensional, specifically for the case of computer-aided design (CAD) models. Datasets for geometric deep learning [11] and structural analyses of the models [12] are becoming increasingly available. At the moment, most of the datasets

offered are of individual mechanical parts and rarely of large-scale systems or structures.

2.1.4 Potential Data Re-representation

Current data that might be relevant to this research, as discussed above, usually falls into the following categories: bitmap images. Sequential data, or CAD models. Bitmap images are chosen as a basis of this research due to the currently extent of research done with it and its versatility towards sketches of architectural structures. However, this form of data is far from perfect. The bitmap data representation format contains no direct information pertaining to structural systems. Some work was done to develop a new data representation that can be used for similar problems in the future.

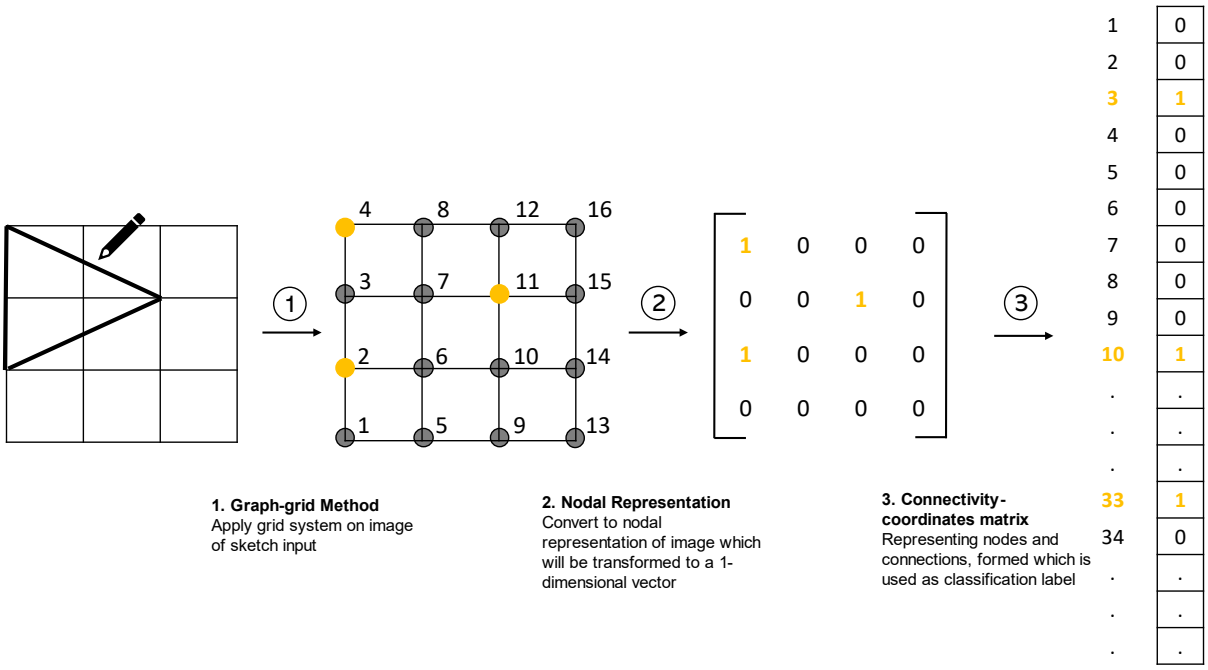


Figure 1: Process of data re-representation from superimposing grid on input image to generating classification labels.

Instead of retaining the pixel-based format, the data is converted to a connectivity-coordinates matrix which represents the connections between nodes presents on the sketched structure. This is done by training a deep learning model to classify input data of 2D shapes with their corresponding matrix label. The matrix data for each

shape is generated by applying a grid system on the image as shown in Figure 1. The nodes of the grid closest to vertices of the shape is noted and the grid is then transformed into a 1-dimensional vector. The vector is used as the basis of the connectivity-coordinates matrix, where non-zero inputs represent vertices of the shapes on the grid and their connections with one another, thereby defining the shape.

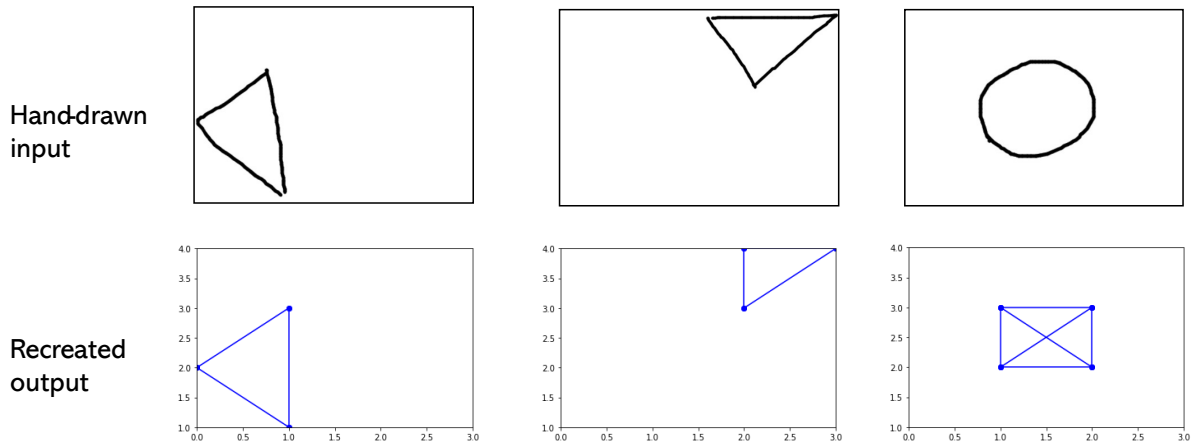


Figure 2: Data re-representation reconstruction results.

As seen from Figure 2, simple hand-drawn inputs could be redefined using the connectivity-coordinates matrix and reconstructed using geometric primitives of points and connecting lines. However, this style of representation, though promising and applicable as an alternative to bitmap images, does not work to the degree of robustness required. Hence, pixel-based images will form the basis of this research.

2.2 Image-based Learning Model

There have been various advancements in machine learning involving predictions based on images. Krizhevsky et al. developed a deep convolutional neural network to effectively classify 2D images from the ImageNet dataset [13]. Moving further to the third dimension, generating accurate models via generative adversarial networks [14] have proved promising. Procedural modelling approaches have also been used to bridge the gap between 2D hand sketches and precise 3D models [15]. With advances in stochastic optimization [16] founded on gradient-based optimizers, machine learning methods can presently also be trained more efficiently to meet this objective.

These methods, though useful in computer graphics, do not include a performance analysis perspective which limits its usefulness for structural and architectural collaboration.

2.3 Surrogate Modelling for Structural Design

On the other side of the spectrum, design analysis and exploration have been comprehensively researched. Data-driven surrogate modelling in early-stage design of structures [17] and the development of computational strategies for creative structural designs via a combination of parametric modelling and interactive optimization [18] are instances where advancements in technology have been applied to the field of structural and architectural design. However, these analytical and design tools are mostly time expensive and require substantial expertise to master. They mostly require a user to develop models with specific features before any useful information can be obtained. These limitations are addressed by the proposed solution presented in this thesis.

2.4 Design Generation

Since the inception of generative models such as GANs and VAEs, deep learning techniques have been widely applied to synthetically generate new designs. Recent advances in the field have also seen its application towards architecture, where new building layouts [19] and substantially better performing models of structures [20] could be generated via learned models. Similar to surrogate modelling, time has to be spent developing models for every new design problem with changing parameters. Though important and relevant during later stages of the design process, it does not serve well given the quick and iterative nature of early stage conceptualization.

2.5 Sketch-based Interfaces

Human sketches have been extensively studied, ranging from applications in sketch recognition [21] to large database creation [22]. Being both versatile and quick to perform, many turn to hand drawings and sketches as a mode of communicating and exploring ideas for conceptual and early-stage design. For the field of deep learning,

Ha and Eck in Google developed Sketch-RNN [9] using the QuickDraw! Dataset [22], essentially building a RNN-VAE which allows users to sketch their designs online and have the model perform a variety of functions, such as the completion of user's sketch inputs, generation of new sketches and the interpolation between sketches. Applications are also found in the realm of structural engineering and architecture. Murugappan et al. developed a sketch interface which provides quick finite element analysis on 2D sketch inputs [23] whereas Keshavarzi et al. used a parameterized algorithm and design constraints integration to parametrize and optimized designs of floor plans from sketch inputs in real time [24]. While these methods present a way to streamline sketching to performance analysis, they leave little room for design generation or creativity by the user, which is a key attraction of using hand drawings and sketches. Whereas the applications that currently exists in computer graphics does not consider performance metrics at all. A hybrid method that aims to apply useful features from both fields will be presented in this thesis.

3. SKETCH-PERFORMANCE PREDICTION

3.1 Methodology

3.1.1 Conceptual Overview

This chapter introduces a method to predict the relative performance of sketch inputs that are structural and architectural in nature to facilitate the design process. There is one specific functionality that will be discussed in this chapter – Given a hand sketch of a design, provide real-time prediction of structural performance.

This is achieved through the application of two concepts from machine learning: variational autoencoders, which project sketch data into compact latent space representations, and surrogate models, which learn relationships between design inputs and performance outputs, shown in Figure 3. This process is challenging to develop for the complete landscape of open-ended sketches; this paper presents a step towards this goal by focusing on applying this method to specific problem types.

From a learning perspective, this means that relevant training data is organized by and developed for specific structural problem types.

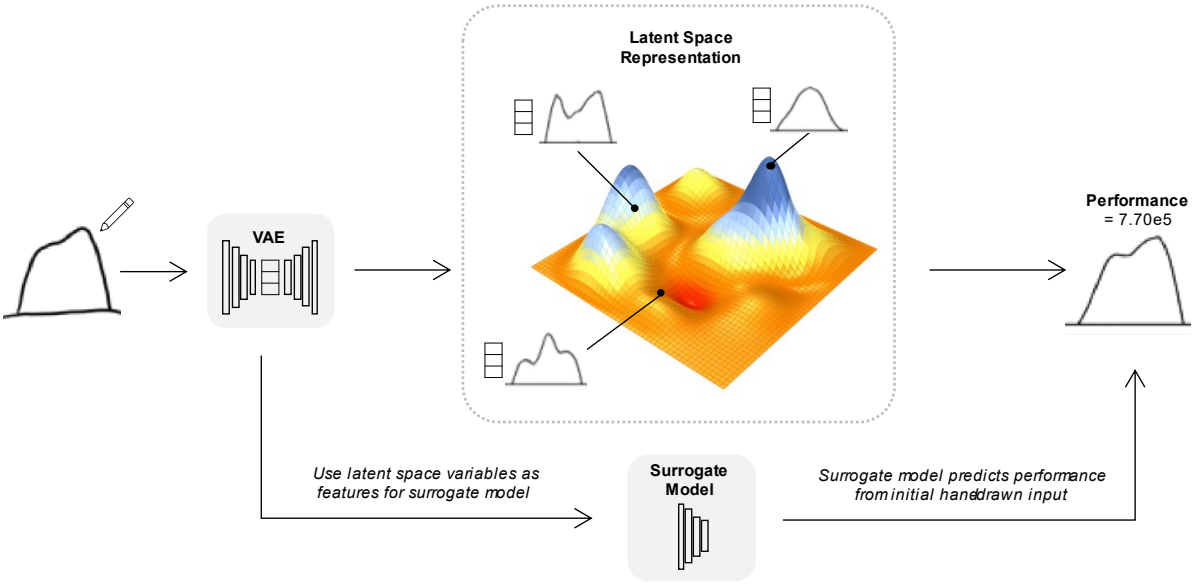


Figure 3: Two-tier surrogate model from hand-drawn input to performance prediction.

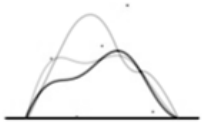
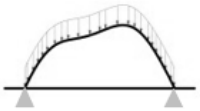
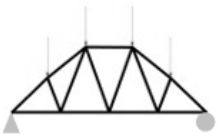
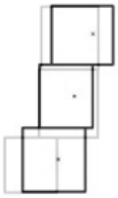
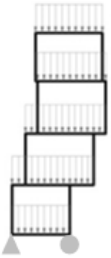
1.1.2 Test Cases

Because natural sketch data from structural problems does not exist at the large scale and in the standardized format needed for machine learning methods, this paper relies on synthetic sketch data generated algorithmically. The synthetic sketch data is designed to mimic human sketches as closely as possible, so that models trained on synthetic data can then be used with real human sketches. Three specific structural model types are used as case studies in this paper: curved frames, spanning trusses, and “stacked box” buildings, summarized in Table 1 below. These structure types are chosen because of their presence in real world structures as shown in Figure 4 and their ability to be digitally modelled.

Figure 4: Real examples of chosen structural types: Heydar Aliyev Cultural Center (left, adapted from [25]), Rey Vitacura (middle, adapted from [26]) and the New Art Museum (right, adapted from [27]).

The methodology is explained through the first model type, curved frames, for clarity. In this model, the curved geometry is both the structural frame and the outer boundary of a building and represents geometrically interesting design options of widely varying structural performance. In the following sections, the process for developing this sketch-based method is given: Data generation (3.2.1), Data processing and augmentation (3.2.2), and Prediction Model (3.3). Finally, Section 3.4 showcases results obtained from the model developed.

Table 1: Test cases with corresponding details.

Case	Variables	Structural Modelling	Performance Metric
Curved frames	 6 variables: vertical locations of control points 1 variable: smoothness	 • Pin-pin support conditions set at either ends of frame • Distributed vertical load of 40kN/m (8kN/m x 5m secondary span) applied along frame. • Frame spans a distance of 30m.	Linear elastic FEA, Performance = (strain energy) x (total length of frame)
Trusses	 6 variables: vertical locations of control points 1 variable: number of subdivisions	 • Pin-roller support conditions set at either ends of truss • Total vertical load of 40kN distributed at each node of upper chord • Truss spans a distance of 10 meters	Linear elastic FEA, Performance = summation over structure of (element axial force) x (element length)
Stacked boxes	 5 variables: locations of control points 2 variables: width of boxes 1 variable: number of stacks	 • Pin-roller support conditions set at either ends of lowest box • Uniform vertical load of 75kN/m applied on floor of each stacked box • Width of box ranges from 7m to 10m, height of box ranges from 5m to 15m, depth of each box fixed at 20m • Additional structural x-bracing modelled for each stacked box to transfer lateral loads	Linear elastic FEA, Performance = (strain energy) / (total volume of stacked boxes)

3.2 Data Synthesis

An important consideration in working computationally with sketches is the data representation format. In previous work, sketches have been represented as sequences of drawing operations (Ha and Eck [9]) or interpreted into CAD primitives (Murugappan et al. [23]). However, the most common way to work with sketches in a learning method is to represent them as rasterized or bitmap images, which are 2D matrices of pixels. This has the advantage of reasonably high-fidelity visual representation and offers the opportunity to work with the most well-studied machine learning technology of the last several years, convolutional neural networks (CNNs), which operate efficiently on image data. Images also work equally well to represent both real sketch data (scanned from hand drawings or digitally created with a stylus)

and synthetic sketch data needed to train models. For these reasons, this paper's research works with image representations of sketches.

3.2.1 Generating Data Sets

The design data needed to train the learning models is generated by sampling parametric models created using Rhino3D and Grasshopper3D. The data generated consists of 2D images of designs of the three model types, curved frames, trusses, and stacked boxes, along with their corresponding performance evaluation metrics. The details of each test case can be seen in Table 1. For each case, a structural model is created automatically and analyzed using Karamba3D, as described in the table. A structural performance metric is associated with each generated design, with a lower value being better.

The open-source Design Space Exploration (DSE) tools suite (Brown et al. [28]), specifically the Sampler and Capture tools, is used to iterate and capture the images and target performance properties of the model.

A dataset of 20,000 bitmap images and their corresponding performance scores are sampled from the proposed data generating pipeline. As seen from Figure 5, the sampled designs have a wide distribution of performance scores, with a lower performance score corresponding to a better performing design with respect to the abovementioned performance metric. Using this method, a myriad of designs can be generated that approximately encompass the variety of designs that might be drawn by a human user.

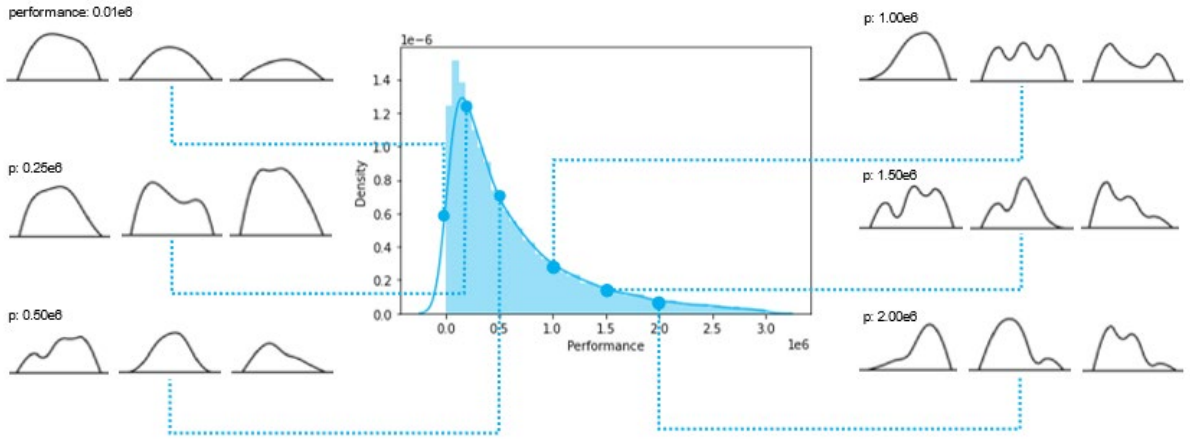


Figure 5: Distribution of structural performance of generated curved frames designs.

3.2.2 Data Processing and Augmentation

The bitmap images obtained are resized to 72x72 pixels to reduce resolution, which increases the efficiency of the model training while retaining the general aspect ratio of the image. Data augmentation methods are applied to mimic human imperfection found in typical hand drawn sketches. Two such methods that were initially explored were Gaussian white noise and elastic deformation of images: salt and pepper noise was added to recreate blank spaces and discontinuity found in hand sketches, and elastic deformation was explored to impersonate the imperfect non-straight lines of human drawings.

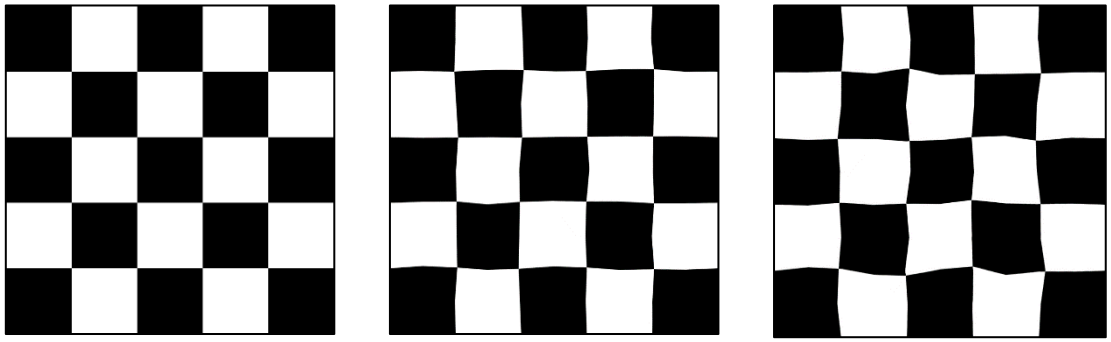


Figure 6. Original checkboard image (left, adapted from [30]), augmented image with grid width and height 5 and magnitude 3 (middle) and augmented image with grid width and height 10 and magnitude 6 (right).

While these methods were promising, they were difficult to replicate effectively for all the data and were more dissimilar to hand sketches than intended. The augmentation technique finally chosen is the application of random, elastic deformations on images using the open source Augmentor Python package (Bloice et al. [29]). Images produced with this method more effectively mimic the nuanced behavior present in the drawings (see Figure 7). The function performs a randomized, elastic gaussian distortion controlled by the grid width and height superimposed onto the image. The size of the grids affects the granularity of the distortions. Larger numbers will produce finer distortions, as shown by the examples presented in figure 6. The magnitude of the distortions can also be changed to give the following results. When applied to the images of the arch frames, the augmented results produce images that close replicates that of hand drawn sketches as seen in figure 7.

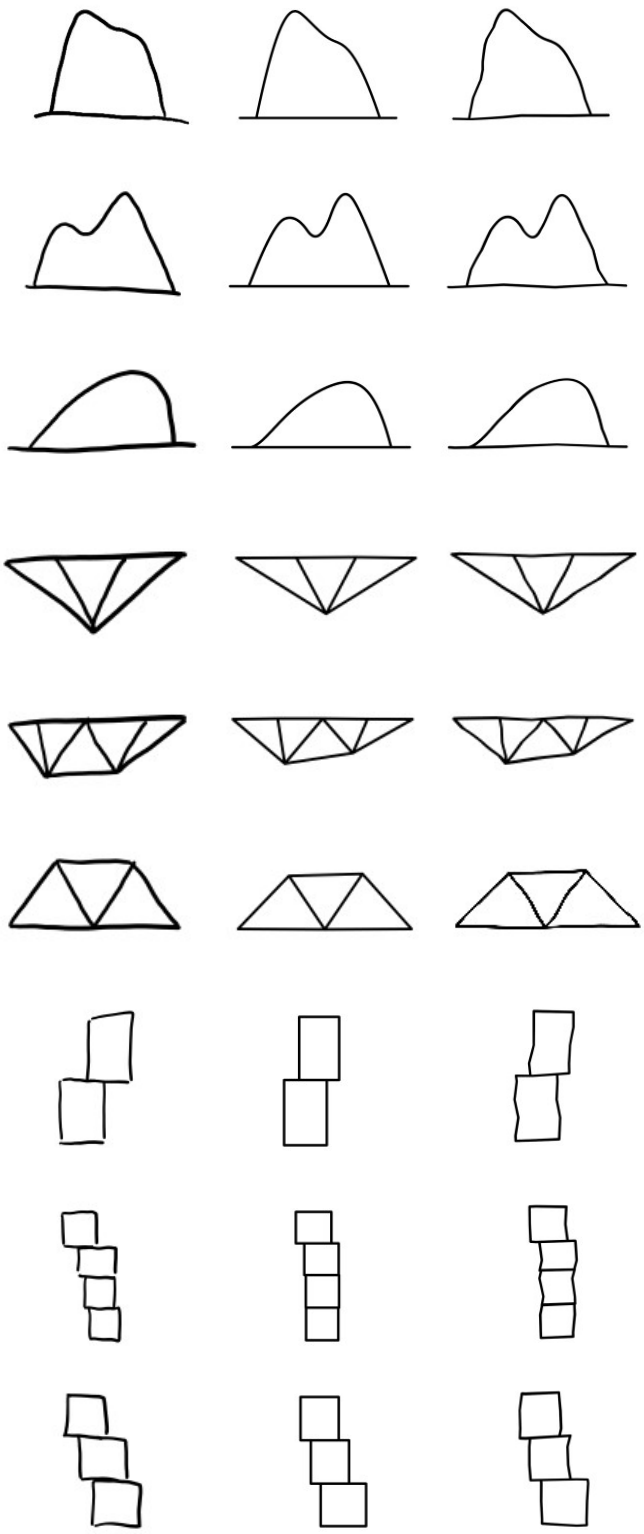


Figure 7: Hand drawn sketch (left), digitally modelled design (middle) and augmented image (right) using Augmentor package [29].

3.3 Prediction Model

The presented method is implemented using open-source platforms to allow for easy prototyping and broad, public application in the future. TensorFlow by Google is used as the open-source artificial intelligence library which acts as the basis of the neural network architecture described later in the paper. The model is scripted in Python on Google Colaboratory, a Jupyter notebook and open-source web application environment which supports the TensorFlow library.

3.3.1 Variational Autoencoder

To begin the construction of the entire model, the variational autoencoder (VAE) must be developed first. Through the process of reconstructing higher dimensional input, the VAE generates lower-dimensional latent vectors which will be used as part of the input for the surrogate model. This is done first by the encoder which compresses data from the input space into the encoded or latent space. The decoder then decompresses the encoded data back into a higher dimensional representation corresponding to that of the initial space. To reduce information loss during the decoding phase, the model is trained to obtain the minimum reconstruction error via an optimization process. However, the lack of regularity in a conventional autoencoder will limit the overall predictive and generative capacity of the model. Hence, explicit regularization will have to be introduced to the model to reduce the possibility of overfitting. This is done by encoding the input as a distribution over the latent space and computing the reconstruction error from a sampled point in that space and backpropagating it through the model. The regularization term expressed as the Kullback–Leibler (KL) divergence, measures differences in probability distributions, regularizes the latent space to obtain one which follows closely to a standard normal distribution. Hence, the overall loss function that the model is trained against will comprise of the reconstruction loss and the KL divergence regularization term.

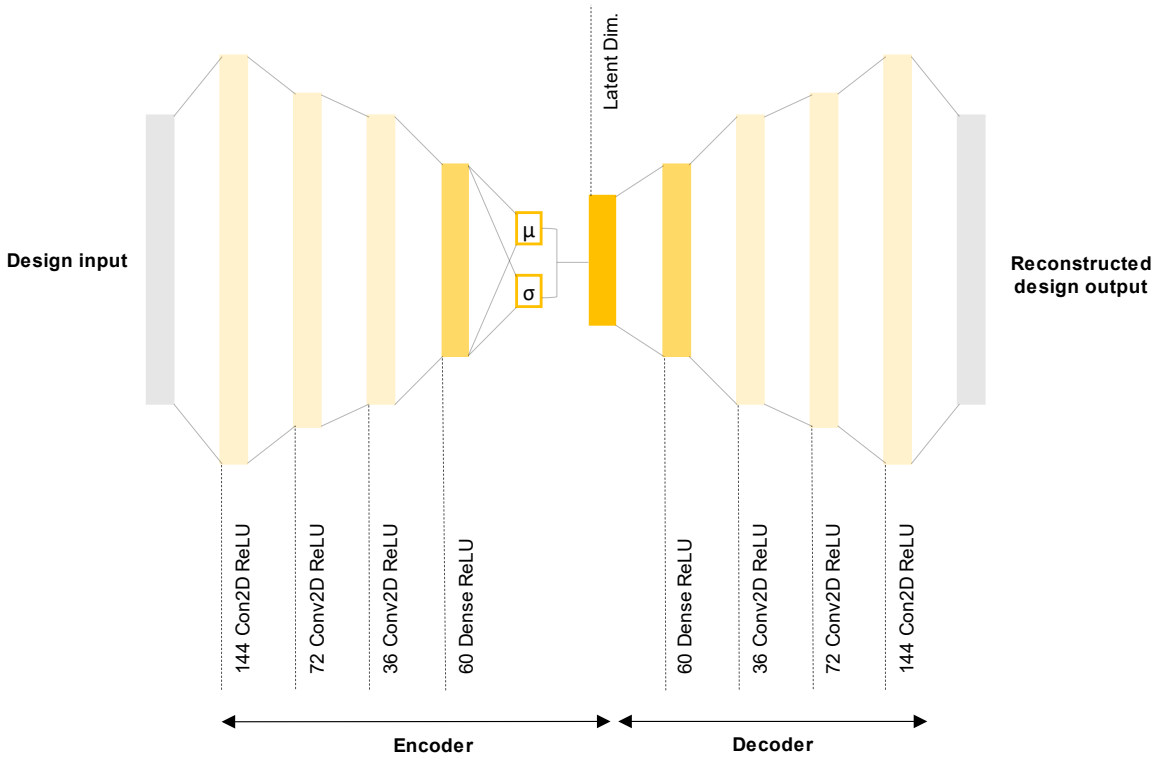


Figure 8: VAE model architecture for curved frames example.

The proposed VAE architecture comprise of four layers in the encoder and four in the decoder. Rectified Linear Units (ReLus) activation functions were utilized to improve training performance of the model. As seen in Figure 8, the encoder is made up of three convolutional layers and one densely connected layer, leading to the bottleneck layer which corresponds to the optimal number of latent dimensions for specific test cases, and the inverse for the decoder. Specifically, the first three convolutional layers are filtered with kernels 16, 64, and 128 respectively, with strides of 2 pixels. The output is then flattened and sent through a fully connected dense layer that have 60 neurons. ReLU activation function is applied to every convolutional and fully connected layer except for the latent layer.

3.3.2 Surrogate Model

The VAE is trained first to obtain the encoder which is used to encoded latent vectors corresponding to each of the training data and their respective performance objective

values. The encoded representation of the input images is trained against the objective scores. The surrogate model is set to solve the presented regression problem, whereby the predicted output would be a real or continuous value. Each set of latent variables associated with an input image will thus have one target performance metric labelled against it (label y_j). A basic neural network model is used where the mean squared error (MSE) loss function, which is the difference between the predicted and true value, is set to be minimized.

$$Mean\ Squared\ Error = \frac{\sum_{j=1}^{n}\left(y_j - y_j^p\right)^2}{n}$$

Equation 1: Mean squared error (MSE) loss function.

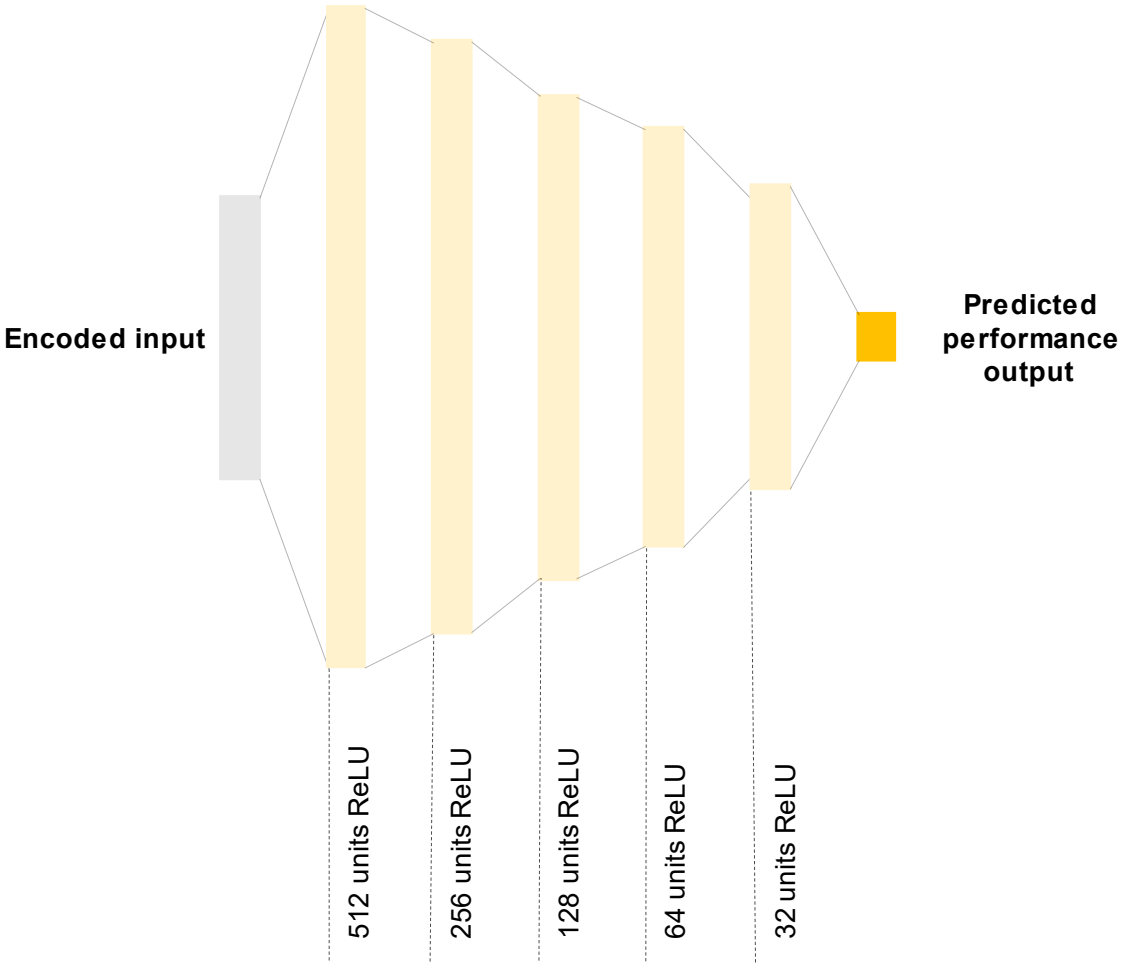


Figure 9: Surrogate model architecture for curved frames example.

The proposed architecture for the surrogate model is made up of four layers as shown in Figure 9: with three fully connected dense layers of 32 kernels followed by one output layer. Again, the ReLU activation function is applied on each layer other than the output layer, where a linear activation function is used instead due to the regression nature of the problem.

3.3.3 Two-tier Surrogate Model

The proposed two-tier surrogate model is an amalgamation of the VAE and surrogate model described earlier. Firstly, the latent vectors obtained during the bottleneck of the VAE is used as the input data, along with the corresponding objective values, for the surrogate model which predicts performance scores. This will allow the first end-to-end function of the two-tier surrogate model, starting with images of sketch inputs which passes through the encoder to produce latent vectors, which will then be used to predict the associated performance scores of the sketches. The second function, which will be further discussed in the later chapter on Performance-driven Sampling (4.3), focuses on design generation from sketch inputs. An additional dimension of performance scores is added to the latent space, which in turn guides the sampling and interpolation results generated via the decoder's image reconstruction.

3.4 Results

This section applies the methodology described above to the curved frame and other case studies for evaluation and demonstration of its potential use.

3.3.1 Model Evaluation

The total loss that the variational autoencoder is trained against is the sum of the reconstruction loss and Kullback–Leibler loss. The total loss of the final train model was determined for each of the input data sampled. A density distribution curve is plotted to show how total loss varies with the type of designs present in the data. As seen in Figure 10, a higher loss corresponds to a fuzzier reconstruction. The complexity and relative realism of designs are largely correlated with the VAE's ability to detect and reconstruct them. Hence, the VAE seems to perform better for more

intuitive and realistic designs of structures, but even the designs with poor total loss values are reconstructed quite well. This means that the VAE latent space is a trustworthy design space for exploring variations on sketch inputs.

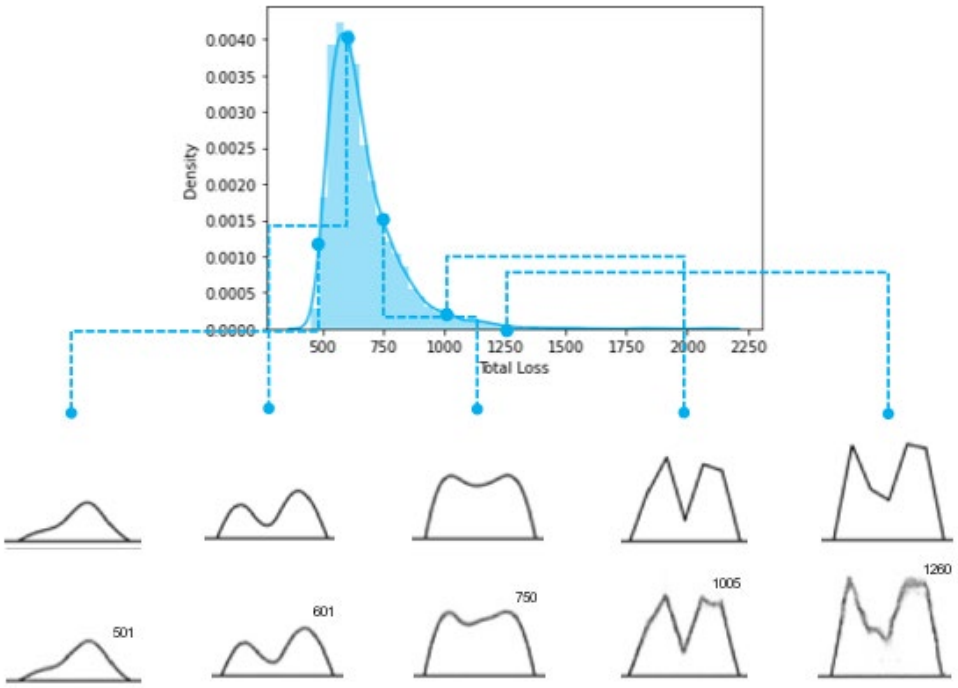


Figure 10: Distribution of total (reconstruction) loss of the trained variational autoencoder across all of the test data (from synthetic sketches). In the called-out designs, the top row shows the original inputs, and the bottom row shows the reconstructions. Even the designs with high reconstruction loss are very recognizable visually compared to the inputs.

Model prediction accuracy is determined by the absolute difference between actual performance values of designs and predicted values normalized by the median performance score of all designs generated. Designs that are more similar to real life structures or sketches of structures made by humans have a lower error value as compared to more complex and unrealistic designs. Hence, it can be argued that the model presented has a high efficacy for realistic designs of structures.

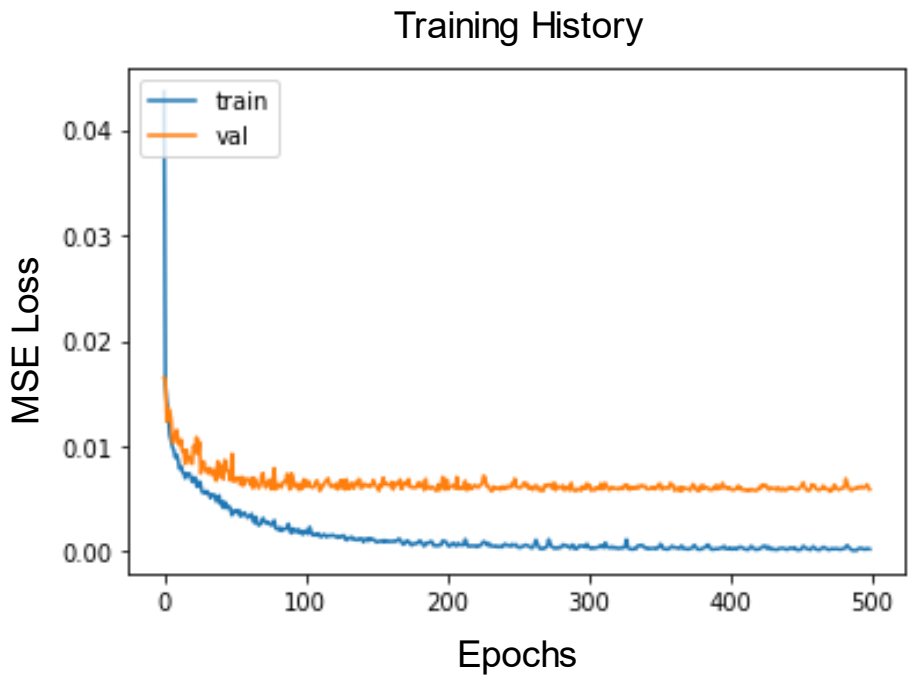


Figure 11: Evolution of training and validation loss for regression model. Final training loss was logged at 2.53e-4 and validation loss at 0.0059.

The deep learning regression model shown in figure 9 was chosen after testing multiple architecture configurations. Hyperparameters were adjusted to obtain a model which resulted in the lowest training and validation loss without overfitting. Specifically, the model was trained with a batch size of 32 for 500 epochs and with a training-validation split of 80-20. The resultant training and validation loss was logged and is shown in Figure 11.

To visualize the accuracy of the performance prediction model, a density plot was obtained which illustrated the absolute prediction error for all the training data (see Figure 12). Here, the absolute prediction error was calculated by finding the absolute difference between the actual performance score of a design and the performance score predicted by the regression model. The resultant mean prediction error for the model was found to be 0.002e6 with a standard deviation of 0.004e6. It should be noted that more structurally realistic and better performing designs have a lower absolute prediction error than the opposite. Hence, it can be argued that the model

performs well in predicting sketched designs by users that corresponds more towards structures or buildings found in real life.

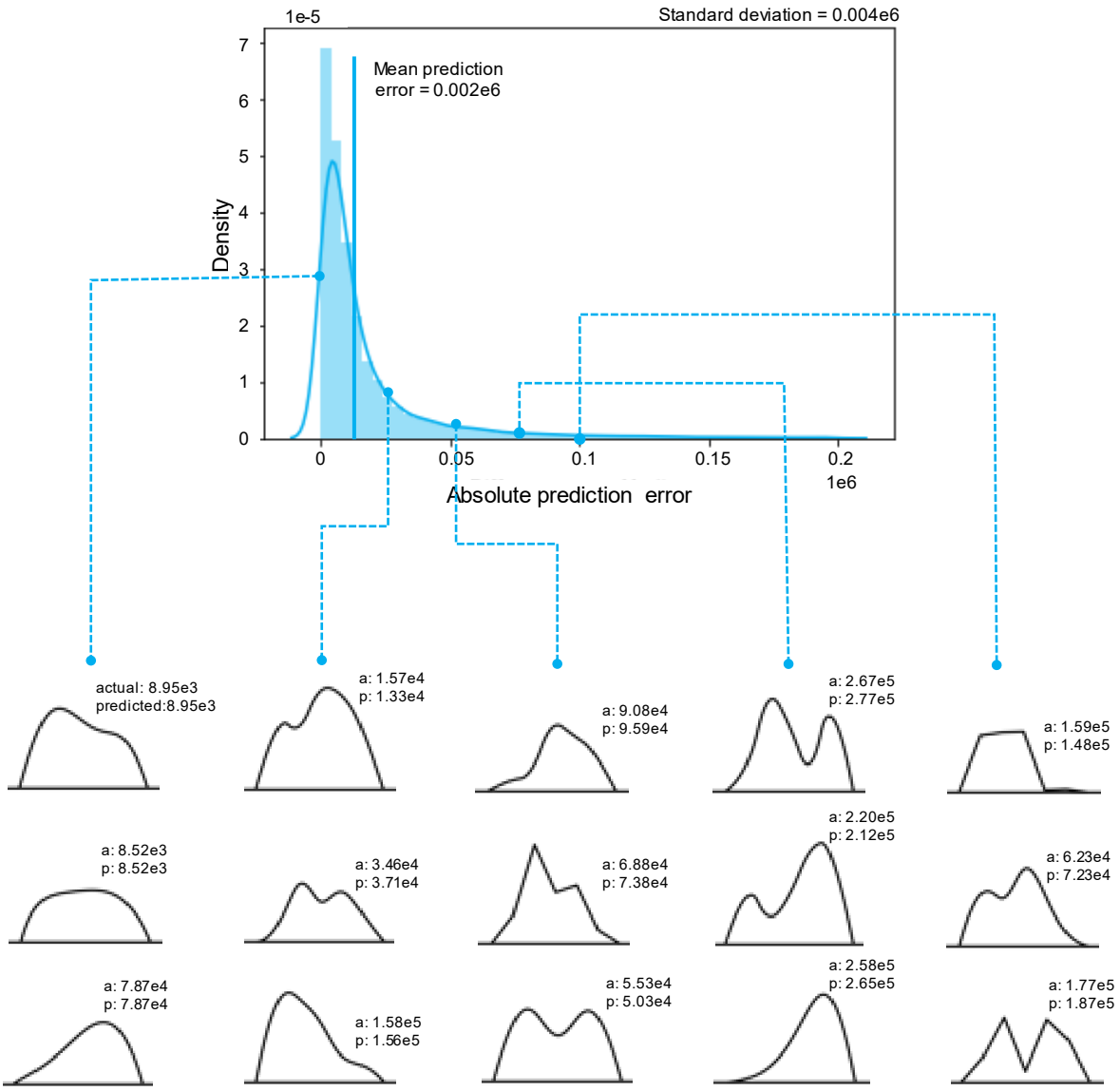


Figure 12: Difference of actual and predicted performance values of most designs less than 0.002e6.

3.5 Conclusion

This chapter proposes a performance prediction model based on a combination of a VAE and a surrogate model. It introduces the fundamental building blocks of the proposed two-tier surrogate model, and the proposed data synthesis pipeline in which

sketch data was generated to train the model. The model is able to predict the performance scores of sketched designs from human users according to data that it was trained on.

In the next chapter, further applications of the two separate tiers of the proposed model will be discussed. Specific test cases will be used to showcase specific results obtained from the model.

4. SKETCH-DESIGN GENERATION

4.1 Methodology

4.1.1 Conceptual Overview

In continuation from the proposed model described in the earlier chapter, this chapter will focus on 2 specific applications that can be obtained from the two-tier model:

1. Given a hand sketch of a design, suggest better-performing alternatives.
2. Given two (or more) design sketches, return a set of new options that visually combine aspects of both (and that perform well).

4.1.2 Test Cases

The same 3 test cases will be utilized in this chapter, mainly the curved frame, truss, and stacked boxes. However, the curved frame will be used as the main example to

describe and explain concepts, algorithms, or other processes that is relevant to the understanding of the proposed applications.

4.2 Design Generation

In performance-driven design processes, rapid performance predictions are quite useful in their own right (Tseranidis et al. [17]), but they can also be used to support design space exploration (Brown et al. [28]). In this paper, the VAE developed to generate encoded data also allows for exploration of the latent design space, as shown in recent previous work for non-sketches (Danhaive and Mueller [15]). Using the latent features found in this space, new designs can be generated that relate and respond to those input by the user.

4.2.1 Data Processing and Collection

The same data generation and processing methods as the one described in chapter 3 is utilized. For the case of hand drawn sketches by human users, which will be used to illustrate the accuracy of the performance predictions and design generation, the sketch data is logged via a digital sketch canvas and an active stylus. In the case of the examples shown later in this chapter, the method of capture was done via Microsoft Whiteboard sketch canvas and the Microsoft Surface Pen.

4.2.2 Latent Space

As explained previously, the VAE is used to generate encoded data in the form of the latent vector which is then utilized as input for the surrogate model. In generating the latent vector, the design latent space of the chosen structure is also be created. This means that the user can move through the newly defined design space to discover potential new designs.

Through the regularized training of the encoder and decoder of the VAE, a normally distributed latent design space is obtained. This space contains features that can be interpreted into physical designs by the corresponding trained decoder. The dimensionality of these features, or latent space vectors, is subjective and based on

the amount needed to achieve a model of high fidelity, and hence dependent on the complexity of the design in question. The extracted latent space vectors provide additional information which, when examined, can be useful to the overall design process. Since the latent space vectors usually contain high-dimensional data, a parallel coordinate plot might be employed to visualize the data of *n*-dimensions. The combined latent features form the latent vector defining a specific design which can then be interpreted by the decoder. By examining the visualized parallel coordinates plots of the data, it is possible to ascertain specific latent space features that contribute more to the performance of a design, or other features that might be of interest to the designer.

4.2.3 Sketch Recreation

After fine-tuning the model by the changing of hyperparameters, the trained model is applied to new, unseen input data to predict the structural performance previously set. Hand-drawn input sketches shown in Figure 13 are presented to the model and designs are reconstructed by the decoder. The predicted performance using the surrogate model for each input is then normalized against the best performing design. Generally, sketches that correspond to designs that are structurally unsound are predicted to have worse (higher) performance scores.

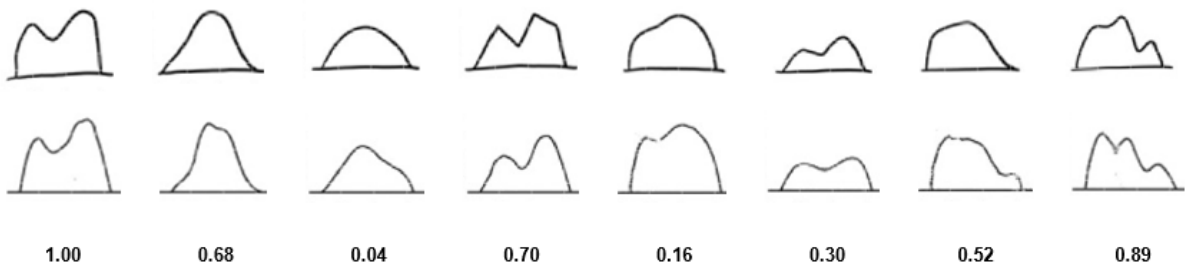


Figure 13: Reconstructions (middle) and normalized predicted performances (bottom) from hand-drawn input designs (top).

4.2.4 Design Suggestions via Interpolation

As shown previously in Figure 3, the training of a variational autoencoder in the first tier of the model allows for the generation of a latent space. In doing so, inputs can be

encoded into latent vectors which corresponds to points in the latent space. Inputs can then be interpolated to view the transformation between one input design and another, resulting in intermediate designs that contains attributes of both designs. This morphing of designs can be seen in Figure 14.

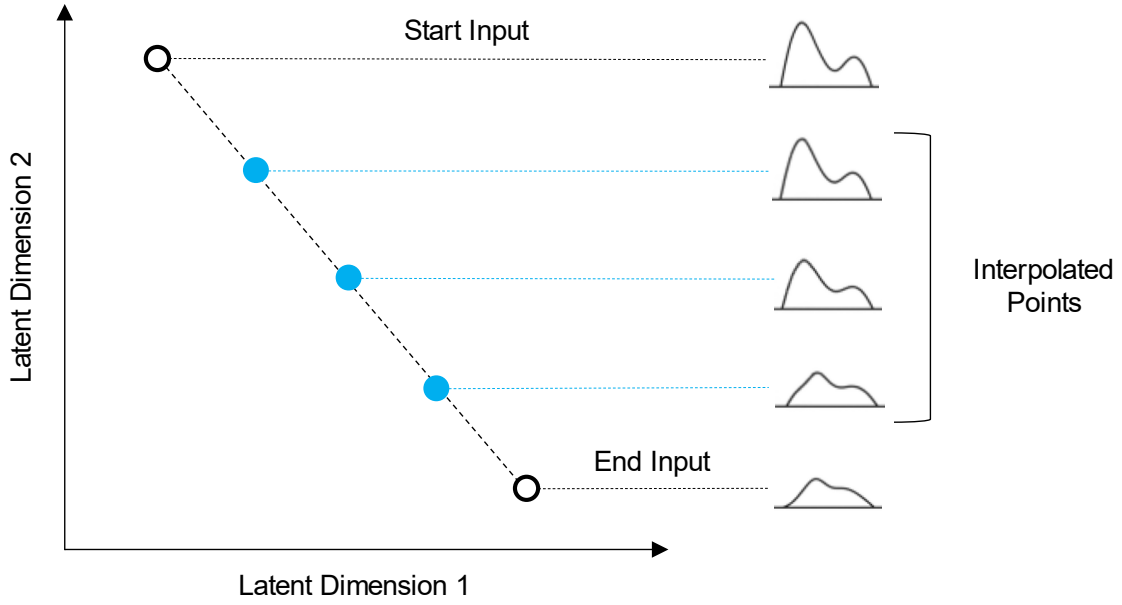


Figure 14: Interpolation with design latent space with 2 latent dimensions.

The interpolation process starts by taking the initial inputs and encoding them into latent vectors using the encoder of the VAE. A new grid space is generated which size corresponds to the number of new designs proposed by the user. For the example shown in Figure 14, the size of the new grid is 5 if we include the two initial design inputs. Given that each latent dimension comprises of a range of real and continuous numbers, a confined range is calculated using the encoded values for the two inputs. Each range per latent dimension is divided into equal segments according to the number of new designs proposed earlier. To put simply, as shown in Figure 14, a line is drawn between the start and end points in the design space and segmented equally. The regularly spaced points are effectively latent vectors in the design space, which will then be decoded into the newly generated interpolated images. These designs contain features of both initial inputs extracted by the VAE, thereby producing a gradually changing morphology from one design to the next.

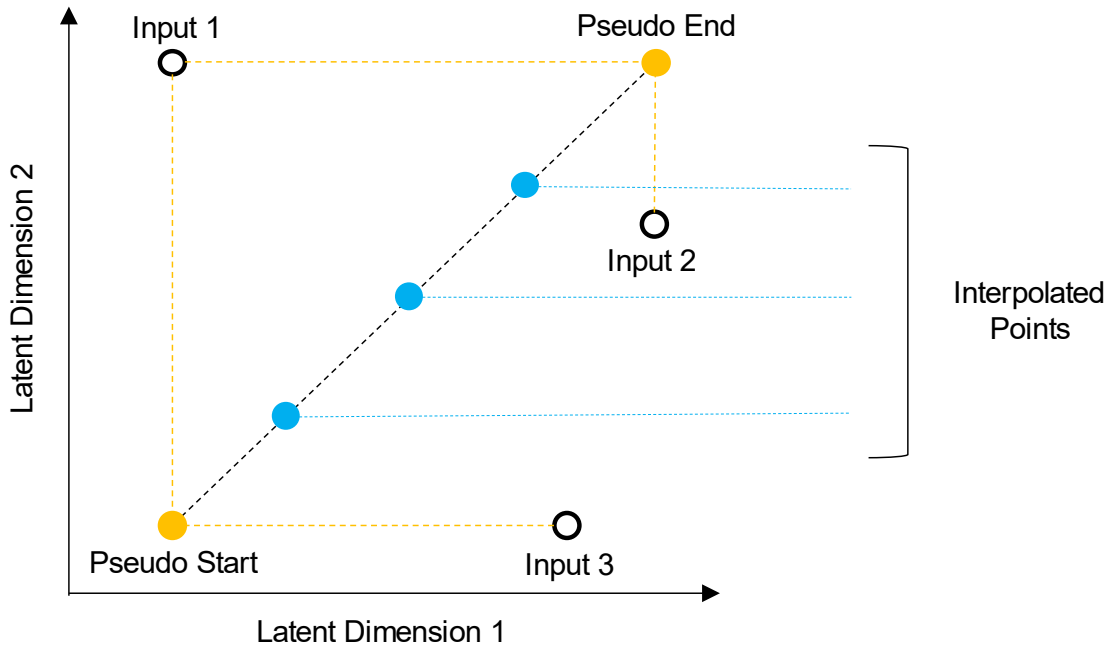


Figure 15: Multi-input interpolation with 3 inputs.

A similar methodology is used for the interpolation of multiple inputs. The difference lies with the start and end points in the design space. Firstly, the maximum and minimum encoded value is found from the input designs for each latent dimension. For the example shown in Figure 15, the maximum encoded value for all of the initial inputs is aggregated into a new pseudo-input whereas the minimum values are developed into the other. The largest range for each latent dimension is chosen over the smallest range due to the increased variety of generated designs similar to that of the inputs. The same average segmentation is applied on the n-dimensional line between the two pseudo-inputs and the interpolated results would contain features present in all the designs initially inputted by the user.

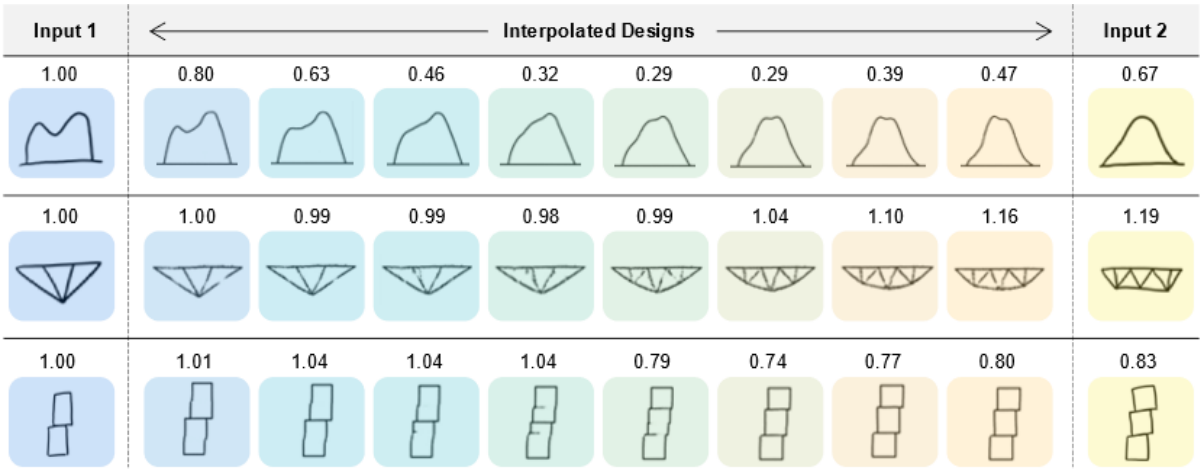


Figure 16: Suggestions via interpolations of 3 selected test cases from hand sketches.

In addition to the curved frame designs, the same process is applied to the other test cases previously mentioned, specifically for trusses and stacked boxes. The number of latent dimensions and hyperparameters are changed according to the requirements for the separate designs. The models with curved frame required 18 latent dimensions, whereas the model with the truss and stacked box required 30. Similar results are obtained for interpolating between input designs, and performance-based design suggestions, as shown in Figure 16.

A different set of designs can be generated if multiple sketches are given as the input for the interpolation process. Shown in Figure 17, an additional design to the original 2 curved frame designs dramatically changes the generated design outputs due to the effect it has on the pseudo start and end points for the interpolation process.

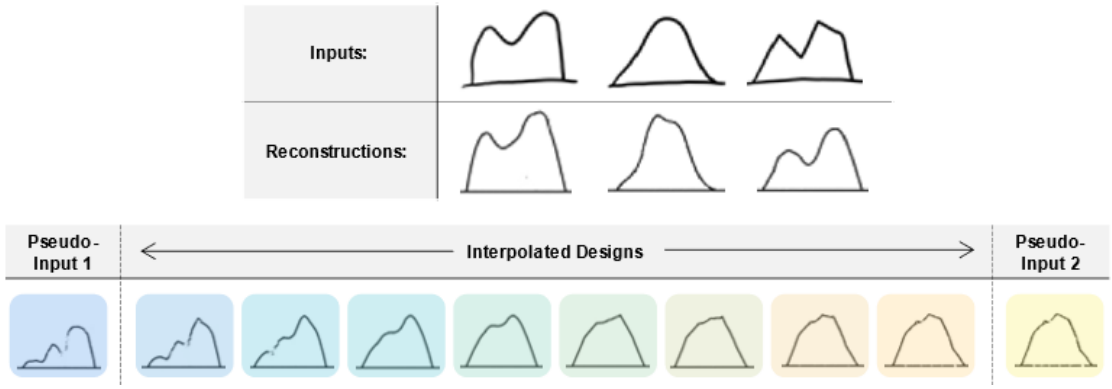


Figure 17: Multi-input interpolation from 3 hand-drawn input sketches.

4.3 Performance-driven Sampling

The surrogate model is designed to predict performance scores given encoded latent variables. This effectively allows an additional dimension to be added to the design latent space comprising of the latent dimensions derived from the VAE (see Figure 18). This new dimension is the performance score corresponding to a specific point in the latent space. The proposed performance-driven sampling leverages on this newly form design space and utilizes the performance score as a metric to guide the generation of new designs, as compared to the conventional interpolation method described earlier which samples based on the averaged internal spacing in the design space.

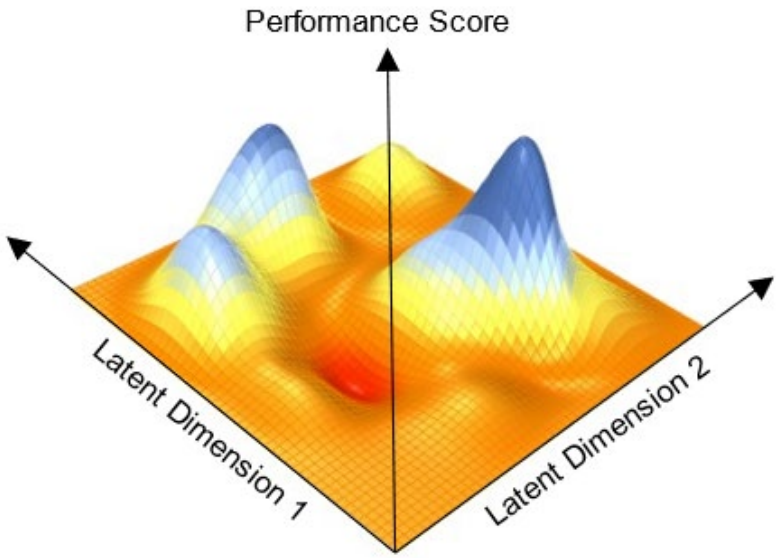


Figure 18: Performance superimposed onto design space.

Generally, performance-guided samples are generated using a n-dimensional cuboid, where *n* refers to the number of latent dimensions given the specific test case in question. Designs are stochastically sampled from this pre-defined cuboid space based on pre-set performance score restrictions. The details of this performance-guided sampling algorithm can be viewed in the pseudo-code present in Algorithm 1 below, whereas other algorithms will be described in further detail in the appendix.

Algorithm 1: Performance-driven Sampling

Input:

- Encoder model
- Decoder model
- Performance prediction regression model
- Input data
- Number of desired output designs, N
- Number of steps per iteration, n_{step}
- Desired percentage change in performance score, Δ
- Sampling radius increment, *rad*

Output:

- Generated design samples
- Corresponding performance scores

Initialization:

- Initialize empty set containing encoded vectors of generated designs of size [N, number of latent dimensions, d]
- Predict performance of input data, p_{input}

for k in [0, …, N-1] **do**
  while (performance score of new design / p_{input}) > (1 – Δ)
    for i in [0, …, n_{step} - 1] **do**
      for q in [0, …, d - 1] **do**
        - Perform random search within vicinity of *rad*
        - Predict performance of new design using regression model, p_{new}

      If (p_{new} / p_{input}) < (1 – Δ)
        - Update empty set with new design
        - Break

    Expand search vicinity by *rad*
  return visualizations of generated designs
  return performance scores of new designs

4.3.1 Sampling Radius

As the latent space sampling is stochastic in nature, the variety of designs sampled would be dependent on the sampling radius. As shown in Figure 19, the range of design morphology increases with sampling radius. The designs generated also tend to move away from the original input in terms of visual feature as the radius in

increased. Similarly, as sampling radius increase, so does the probability of obtaining designs with lower (and higher) performance scores. Hence, there is a need to strike a balance between the occurrence of a better performing design and one that is similar to the one inputted by the user. These factors are taken into account in the proposed expansive-search algorithm which will be elaborated in later sections.

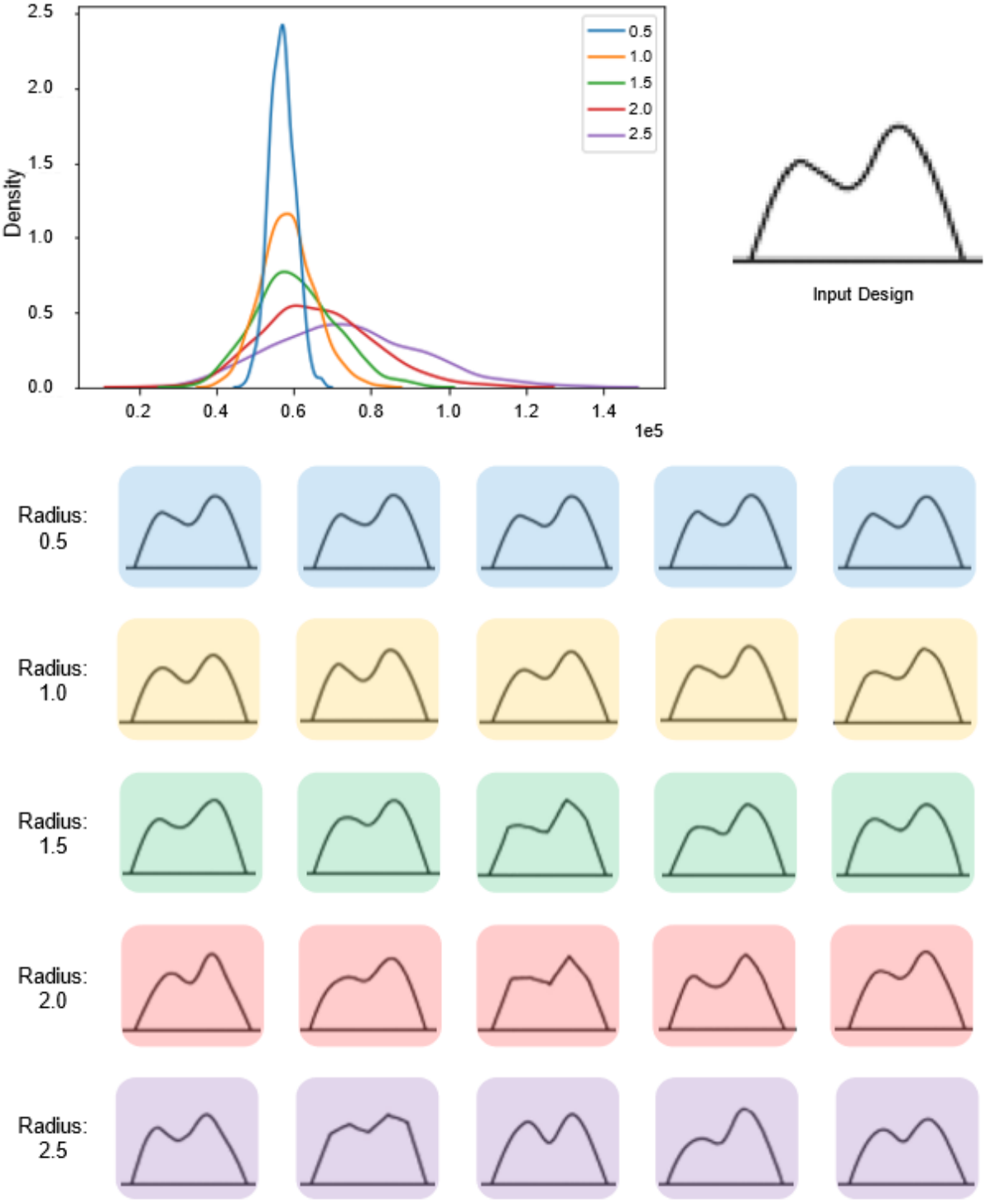


Figure 19: Sampling radius and density relationship. Increasing sampling radius led to a wider variety of possible designs generated.

4.3.2 Single-point design generation

As described previously, through the performance prediction of the surrogate model, the performance score can be added as an additional dimension to the latent space created earlier. In order to generate designs containing similar features to the initial input, as well as designs that are better performing, a performance condition is applied according to the predetermined performance metric. This condition places a minimum improvement bar in the generated designs relative to the design inputted. In Figure 20, improved designs are filtered from latent space in the vicinity of the initial sketch input design.

The proposed expansive-search algorithm takes in sketched designs from the user as inputs. The user is then allowed to declare the minimum improvement in performance scores of designs generated relative to the input design, the number of new designs generated, the design space sampling radius increment value, and the number of iterations per search. The algorithm takes these inputs and performs a random uniform sampling to obtain designs in the vicinity of the input that fulfils the performance score criteria and outputs them along with their performance scores normalized relative to the input design.

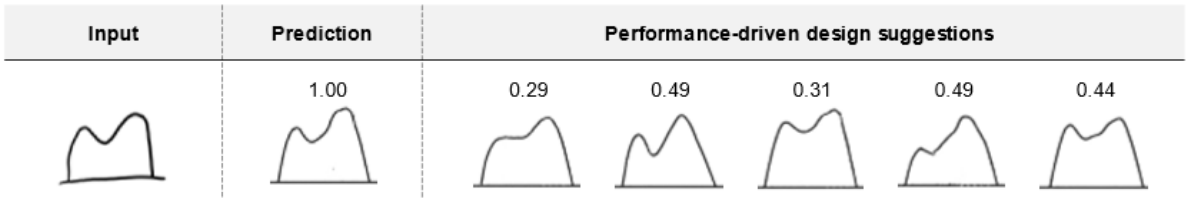


Figure 20: Performance-driven suggestion normalized against predicted performance of hand-drawn input design.

The algorithm is applied to the rest of the test cases as shown in Figure 21. The variety of designs generated relies heavily on the dataset used to train the model as well as the initial input design.

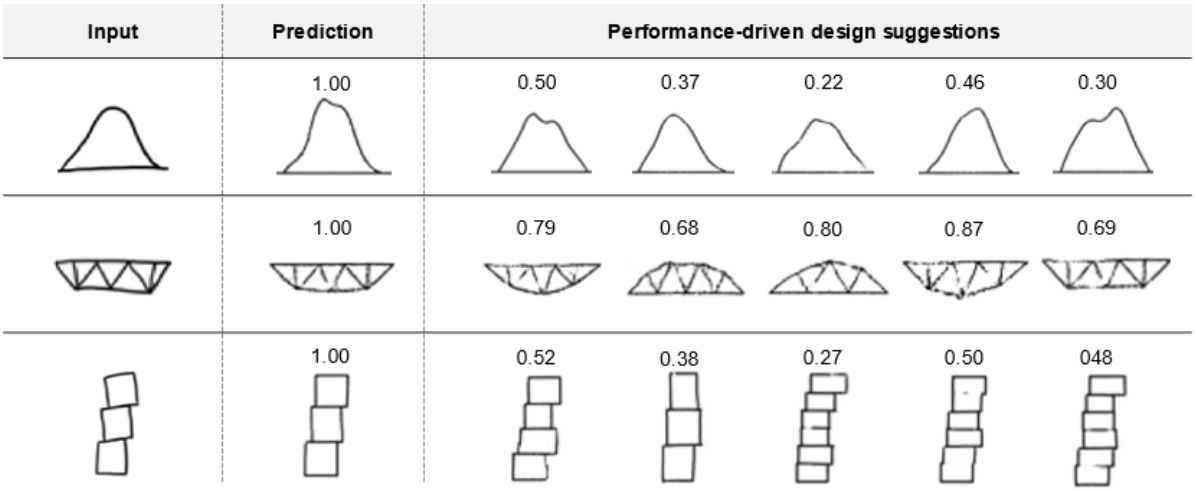


Figure 21: Performance-driven suggestion normalized against predicted performance of hand-drawn input design of three selected test cases.

4.3.3 Performance-based interpolation

Combining the interpolation method described above with the proposed expansive-search algorithm, designs can be generated via performance-driven interpolation. Similar to conventional interpolation, inputs are obtained from the user and encoded into latent vectors by the VAE. The number of interpolated results is declared by the user. The interpolated points determined by the number of final designs stated becomes the initial points for the expansive-search algorithm. The final designs generated using this process will contain features of all designs inputted by the user and contain improvements in performance relative to regular interpolated designs (see Figure 22). The number of design inputs will also not be limited to 2 with the approach described earlier regarding multi-input interpolation.

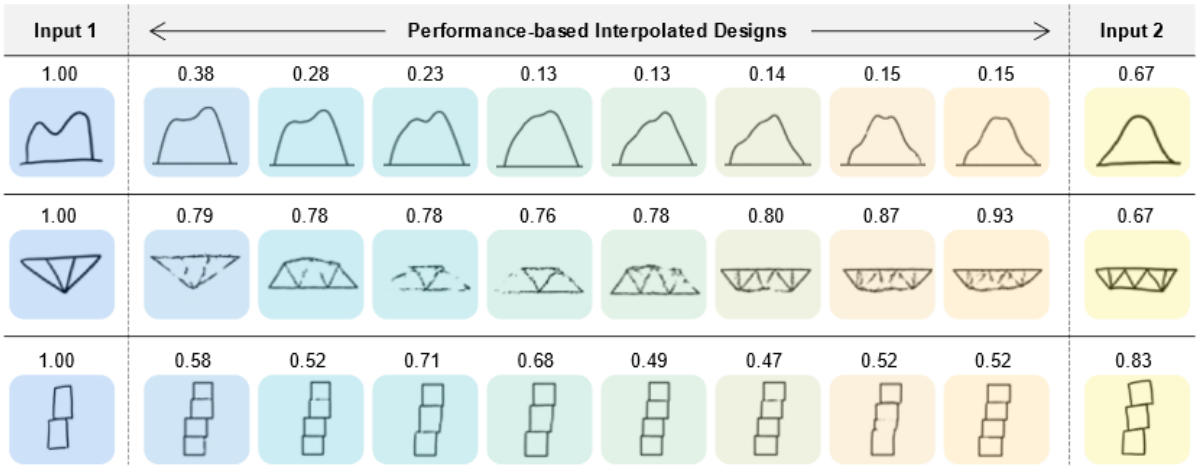
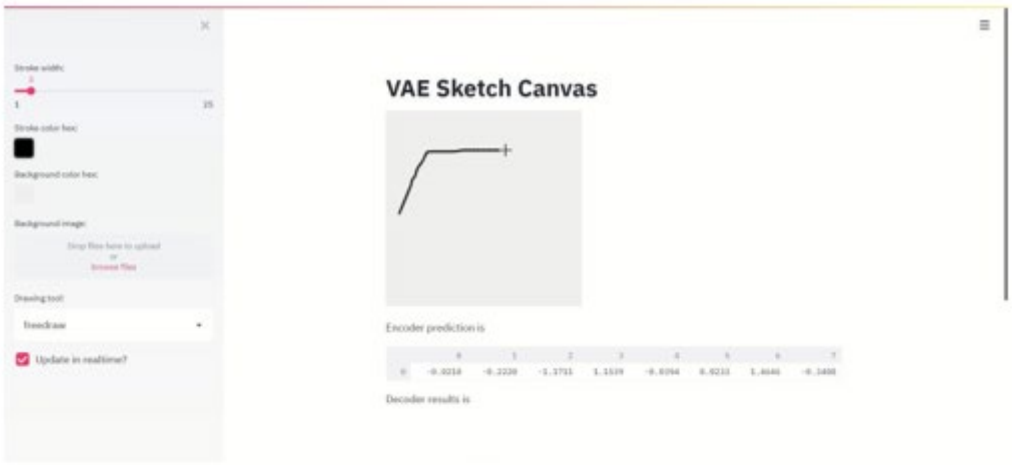


Figure 22: Performance-driven interpolations with corresponding new performance scores from base designs shown in Figure 16 previously.

4.4 GUI Interactions

Since this research focuses on a user's sketch design, the medium in which the user draws the design would be important. Hence, the following features are proposed for the Graphic User Interface (GUI) to provide potential users with an intuitive and comprehensive experience.

4.4.1 Web Implementation



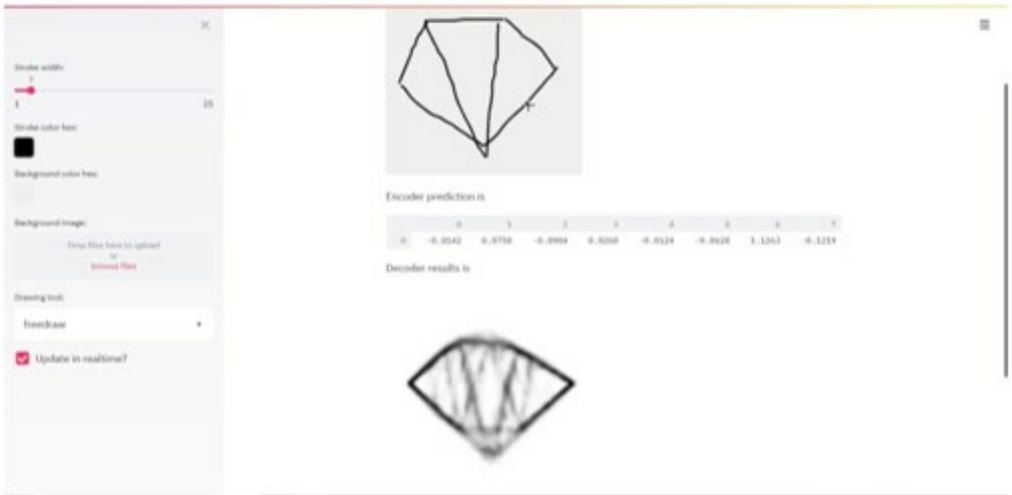


Figure 23: Proof-of-concept web implementation of sketch tool. Allow users to sketch structures in real time to obtain decoder results and reconstructed output.

A simple prototype was developed using Streamlit drawable canvas [31] as the web deployed canvas for sketch input (see Figure 23). A basic VAE model added to the canvas. The prototype allowed for real time encoding of latent vectors to be shown as the user draws on the canvas. The encoded vectors are then decoded to reconstruct the input drawing into the digital representation as identified by the decoder.

4.4.1 Visualization of Data and Design Space

Due to the nature of the designs chosen, the number of dimensions of the latent space usually exceeds that of 3. Hence, a parallel coordinates plot is employed to visualize the latent design space as shown in Figure 24 and 25. Here, the plot is utilized to highlight any common characteristics of the designs that can be attributed to the encoded vectors. For example, the user might be able to see that there are bottlenecks occurring in latent dimensions 1 and 11 in Figure 24 that might be contributing to the better performance in designs, whereas a similar inference cannot be made for poor performing designs in Figure 25. The hope is that an interactive parallel coordinates plot might be implemented into the overall interface so that users will be able to adjust latent variables and learn their attributed features on the fly, thereby informing future designs that might take place.

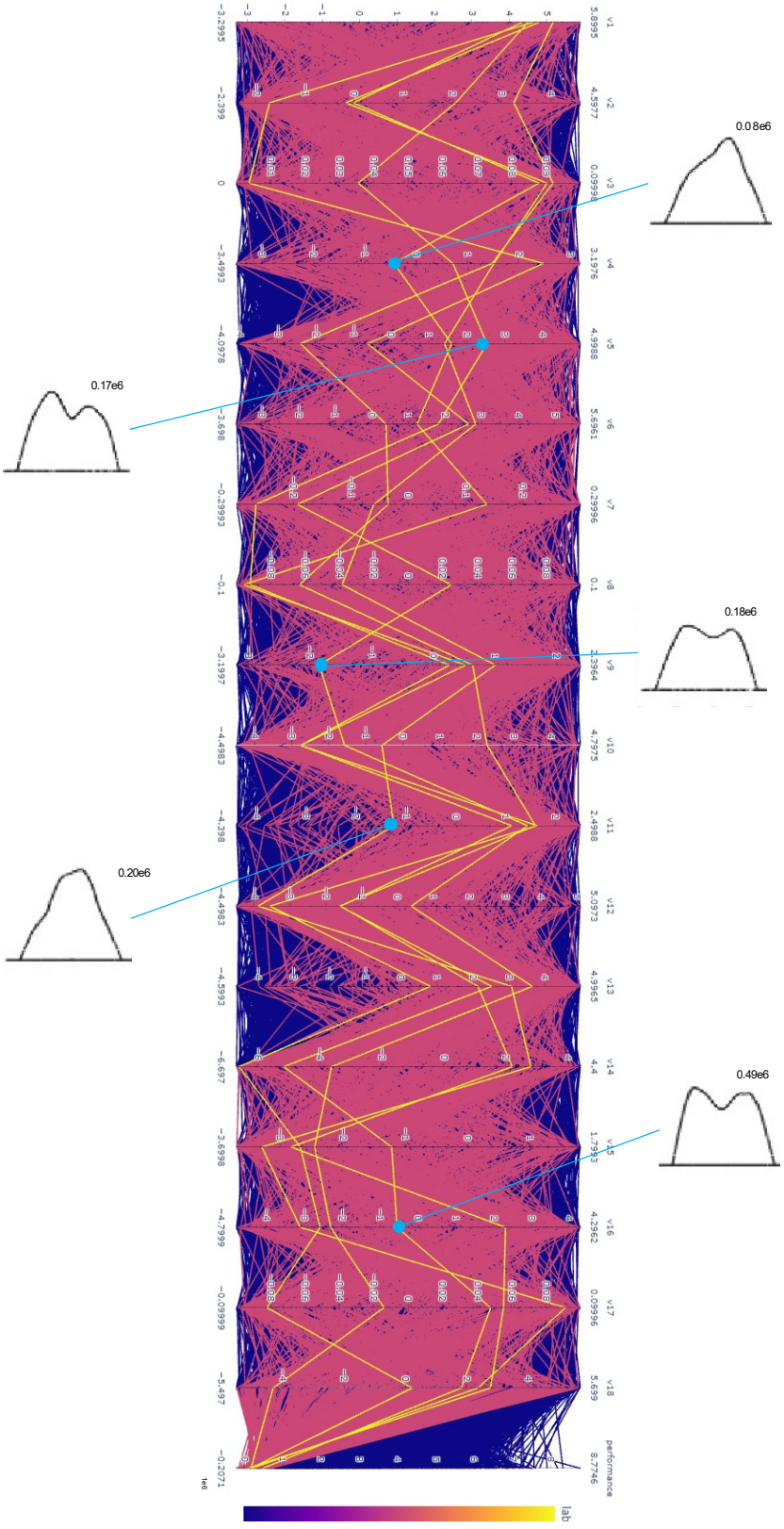


Figure 24: Parallel-coordinates plot of design space showing top 10% performing designs in pink and selected designs with corresponding performance scores in yellow.

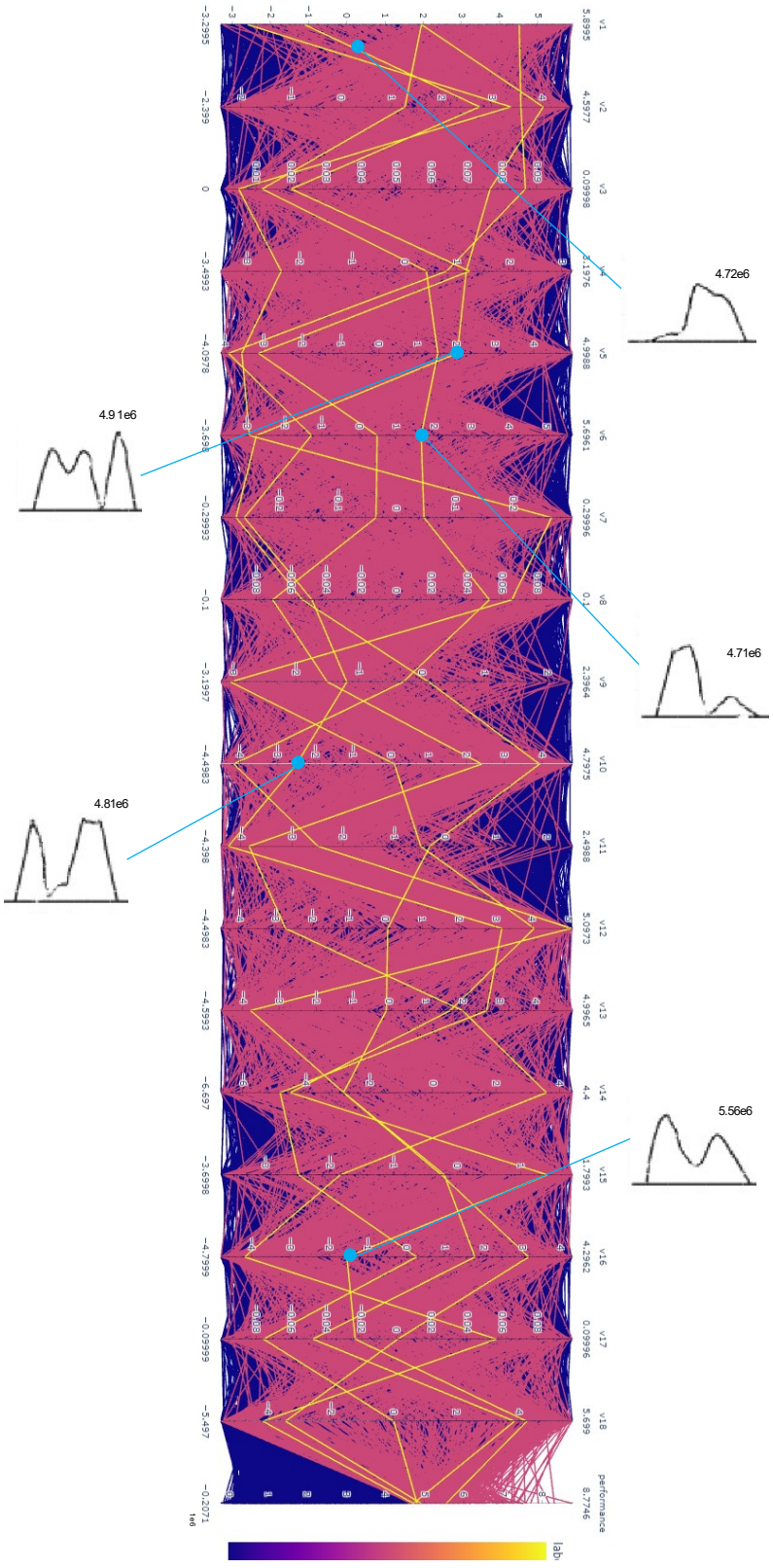


Figure 25: Parallel-coordinates plot of design space showing worst 10% performing designs in pink and selected designs with corresponding performance scores in yellow.

4.5 Results

4.5.1 Evaluation of Model

The training of the VAE model was done by adjusting hyperparameters that obtained the best training loss results, which is the weighted sum of the reconstruction loss and KL loss as described in Section 3.3.1 earlier. Note that as the KL weight approaches zero, the VAE transforms back to a traditional autoencoder. Hence, hyperparameters such as the KL weight, latent dimensions, epoch length, and batch size were all adjusted accordingly. Figure 26 shows the training history of the VAE trained over 1,000 epochs with data from the curved frames. Here, a latent dimension of 18 was used, and the model was trained with a Kl weight of 0.5 and a batch size of 0.5. The resultant VAE model was used to produce the designs found in previous figures above.

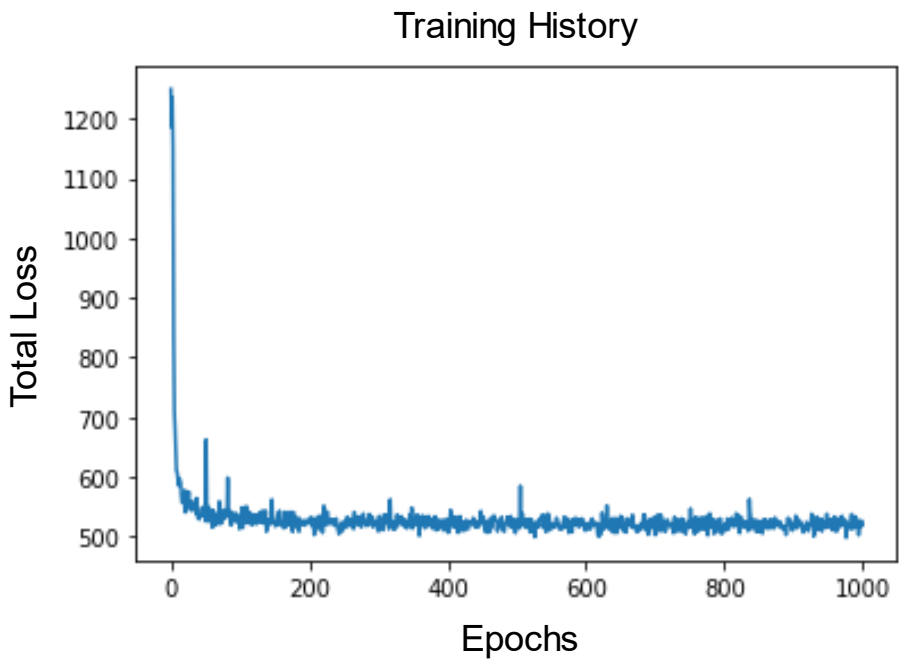


Figure 26: Evolution of total loss (reconstruction loss + weighted KL loss) of VAE model. Final training loss was logged at 519.09 with 515.45 attributed to the reconstruction loss and 3.64 to KL loss.

4.5.2 Abnormal Cases

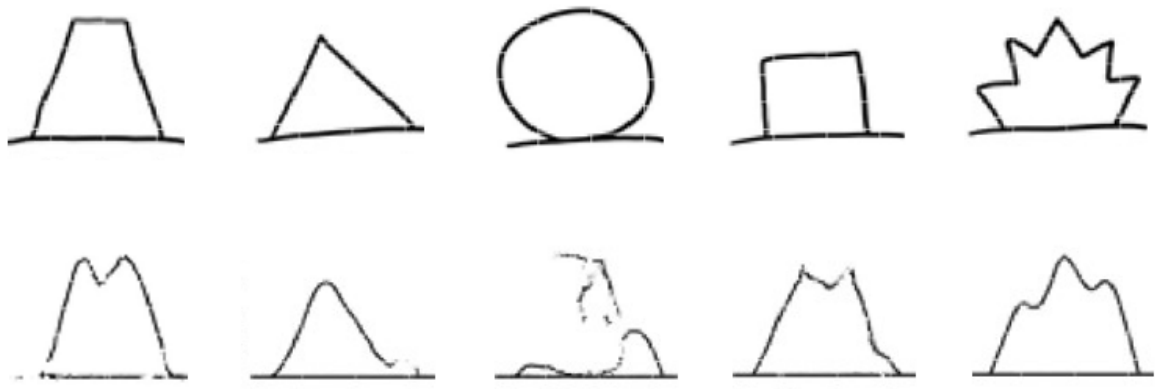


Figure 27: Reconstructions of abnormal designs from hand drawing.

Complex cases dissimilar to those in the training data set are given as input for the model to recreate, as shown in Fig. 27. The reconstructions, and similarly predictions, vary greatly according to the closeness in which the inputs corresponded with the data used. Even so, the model is still able to reconstruct features similar to the initial sketch input for some cases, while failing to detect any similarities for others. This reveals the limits of this approach, and the importance of diverse synthetic sketch data for training the learning models.

4.5.2 Smoothness of Interpolation

Interpolations in Figure 16 and 22 contained designs that lacked perfect smoothness. Particularly for the case of the trusses in Figure 22, some lines of the interpolated designs are seen unconnected to the nodes of the truss. This is due to how the VAE model was trained, specifically on synthetically generated data that aimed to mimic human sketch designs, but still lacks somewhat in identifying all the nuances present in real sketch drawings. However, many of the generated design are still interpretable by a human user, and it can be argued that the design intentions are being conveyed appropriately. Since the interface is targeted towards the early conceptualization stage of the design process, there is merits in keeping the generated designs “imperfect” to reinforce the point that these are design suggestions and not hard orders. This is to

allow space for the designer to consider these new options without overbearing pressure to facilitate the overall design process.

4.6 Conclusions

This chapter proposes potential applications and features derived from the deep learning model described in chapter 3. It suggests possible add-ons that can add value to the early-stage design process. Mainly, the ability to further facilitate collaborations between designers on their designs by taking into account their respective priorities and suggesting potential compromises are enabled via the interpolation methods discussed. These examples shows that deep learning techniques could be used as tools to improve current processes while keeping the human designer at the center of the discussion.

5. CONCLUSION

5.1 Summary of Contributions

Reiterating the overarching question posed in the introduction, the goal that this research sets out to achieve is to improve the current processes in early-stage design of structures through the leveraging of emerging technologies such as increased computational power and deep learning methods. This research has since made several contributions:

1. To address the lack of data needed to solve the problem posed, a sketch data synthesis pipeline was developed to generated designs of structures that mimicked hand drawings with their corresponding performance scores.
2. To highlight performance during the early-stage design process without compromising on the mode of communication, a methodology was proposed

that allowed relative performances of sketch inputs to be predicted and presented to the user.

3. To facilitate collaboration between engineers, architects, and designers alike, a generative model was developed for the creations of new suggested designs based on their respective sketch inputs.
4. The research culminates in proposing new performance-based interpolation techniques that combines attributes of performance analysis and design creation to aid the design ideation process whilst keeping the human designer in the driver seat.

5.2 Potential Impact

This new method of analyzing and generating designs from sketches can have various applications in the fields of engineering, architecture, and design. Being able to analyze and predict relative performance of sketch designs may enable aspects of performance to be included in the early-stage design discussions. This can guide designers towards potentially better performing structures without compromising the initial design intent. Furthermore, the ability to generate new design morphologies based on initial sketch inputs by designers can help facilitate collaboration between different individuals, which in turn fuels the iterative and collaborative nature of the design process. For example, multiple designers such as engineers or architects may have specific designs that they want to put forward. Further designs could be generated using this paper's proposed interpolation method that will not only show a compromise from the initial designs, but also produce new possible varieties for the designers to consider.

5.3 Limitations and Future Work

As with all machine learning models, the predictions and behavior of this paper's method are highly dependent on the training data. Currently, the methodology proposed is limited by the CAD models used to generate the data for training, which comes with their inherent limitations and biases. One goal for the future is to create a public database to record sketches done by designers, architects, and engineers alike

on structures that have been used in the design process. This platform would not only allow individuals to view the variety and complexity of designs that are being created, but also serve as a basis for future research into sketches primarily for the design process. In the meantime, the datasets used in this thesis will be publicly available to allow for more potential research to be done in this field. The details of the database will be described in later papers published. Furthermore, as with the case of computer graphics and machine learning, further methods could be developed to bring performance-informed design generation from sketches to the next stage. Interpolation methods presented here could be expanded to account for other forms of performance, such as energy-based or cost, to suggest compromises between designs input by users to ultimately achieve overall design improvement as per the priorities and needs of the designer. Lastly, as alluded to earlier in the paper, it might be useful for users to directly manipulate the latent variables after starting with an initial design. This can be done so in the form of an interactive parallel-coordinates plot incorporated in an intuitive user interface, in which users can view patterns and effects of changes specific latent variables and generate new designs by exploring the latent design space manually or visually inspecting the suggested designs.

5.4 Concluding Remarks

This paper presents a new method to interpret sketches to predict structural performance and introduces potential applications of the method for design generation. This work hopes to promote and inspire future research and development in the area of design analysis and generation for structural and architectural sketches.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

[2] Y. LECUN, “THE MNIST DATABASE of handwritten digits,” *http://yann.lecun.com/exdb/mnist/*, Accessed: Apr. 01, 2021. [Online]. Available: https://ci.nii.ac.jp/naid/10027939599/

[3] A. Seff, Y. Ovadia, W. Zhou, and R. P. Adams, “SketchGraphs: A Large-Scale Dataset for Modeling Relational Geometry in Computer-Aided Design,” *ArXiv Prepr. ArXiv200708506*, 2020.

[4] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” *ArXiv14062661 Cs Stat*, Jun. 2014, Accessed: May 05, 2021. [Online]. Available: http://arxiv.org/abs/1406.2661

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” *ArXiv161107004 Cs*, Nov. 2018, Accessed: May 05, 2021. [Online]. Available: http://arxiv.org/abs/1611.07004

[6] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015, doi: 10.1007/s11263-015-0816-y.

[7] X.-Y. Zhang, F. Yin, Y.-M. Zhang, C.-L. Liu, and Y. Bengio, “Drawing and Recognizing Chinese Characters with Recurrent Neural Network,” *ArXiv160606539 Cs*, Jun. 2016, Accessed: May 05, 2021. [Online]. Available: http://arxiv.org/abs/1606.06539

[8] D. Ha, “Recurrent Net Dreams Up Fake Chinese Characters in Vector Format with TensorFlow,” Dec. 2015, Accessed: May 05, 2021. [Online]. Available: https://blog.otoro.net/2015/12/28/recurrent-net-dreams-up-fake-chinese-characters-in-vector-format-with-tensorflow/

[9] D. Ha and D. Eck, “A neural representation of sketch drawings,” *ArXiv Prepr. ArXiv170403477*, 2017.

[10] I. Simon, A. Roberts, C. Raffel, J. Engel, C. Hawthorne, and D. Eck, “Learning a Latent Space of Multitrack Measures,” *ArXiv180600195 Cs Eess Stat*, Jun. 2018, Accessed: May 13, 2021. [Online]. Available: http://arxiv.org/abs/1806.00195

[11] S. Koch *et al.*, “ABC: A Big CAD Model Dataset For Geometric Deep Learning,” *ArXiv181206216 Cs*, Apr. 2019, Accessed: May 05, 2021. [Online]. Available: http://arxiv.org/abs/1812.06216

[12] E. Whalen, A. Beyene, and C. Mueller, “SimJEB: Simulated Jet Engine Bracket Dataset,” *ArXiv210503534 Cs*, May 2021, Accessed: May 12, 2021. [Online]. Available: http://arxiv.org/abs/2105.03534

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” p. 9.

[14] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling,” in *Advances in neural information processing systems*, 2016, pp. 82–90.

[15] H. Huang, E. Kalogerakis, E. Yumer, and R. Mech, “Shape synthesis from sketches via procedural models and convolutional networks,” *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 8, pp. 2003–2013, 2016.

[16] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *ArXiv14126980 Cs*, Jan. 2017, Accessed: Apr. 01, 2021. [Online]. Available: http://arxiv.org/abs/1412.6980

[17] S. Tseranidis, N. C. Brown, and C. T. Mueller, “Data-driven approximation algorithms for rapid performance evaluation and optimization of civil structures,” *Autom. Constr.*, vol. 72, pp. 279–293, Dec. 2016, doi: 10.1016/j.autcon.2016.02.002.

[18] R. A. Danhaive and C. T. Mueller, “Combining parametric modeling and interactive optimization for high-performance and creative structural design,” in *Proceedings of IASS Annual Symposia*, 2015, vol. 2015, no. 20, pp. 1–11.

[19] S. Chaillou, “AI + Architecture | Towards a New Approach,” *Harv. Univ.*, Accessed: May 13, 2021. [Online]. Available: https://www.academia.edu/39599650/AI_Architecture_Towards_a_New_Approach

[20] R. Danhaive and C. T. Mueller, “Design subspace learning: Structural design space exploration using performance-conditioned generative modeling,” *Autom. Constr.*, vol. 127, p. 103664, Jul. 2021, doi: 10.1016/j.autcon.2021.103664.

[21] M. Eitz, J. Hays, and M. Alexa, “How do humans sketch objects?,” *ACM Trans. Graph.*, vol. 31, no. 4, p. 44:1-44:10, Jul. 2012, doi: 10.1145/2185520.2185540.

[22] “Quick, Draw!” https://quickdraw.withgoogle.com/ (accessed May 05, 2021).

[23] S. Murugappan and K. Ramani, “Feasy: a sketch-based interface integrating structural analysis in early design,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2009, vol. 48999, pp. 743–752.

[24] M. Keshavarzi, C. Hutson, C.-Y. Cheng, M. Nourbakhsh, M. Bergin, and M. R. Asl, “SketchOpt: Sketch-based Parametric Model Retrieval for Generative Design,” *ArXiv Prepr. ArXiv200900261*, 2020.

[25] “zaha hadid heydar aliyev,” *designboom | architecture & design magazine*, Jul. 15, 2013. https://www.designboom.com/architecture/zaha-hadid-heydar-aliyev-cultural-center-shapes-azerbaijan/ (accessed Apr. 14, 2021).

[26] “10 Projects That Feature Striking Steel Trusses,” *ArchDaily*, Jul. 27, 2017. https://www.archdaily.com/876423/10-projects-that-feature-striking-steel-trusses (accessed Apr. 14, 2021).

[27] “SANAA / Kazuyo Sejima + Ryue Nishizawa, Dean Kaufman · New Art Museum,” *Divisare*. https://divisare.com/projects/289283-sanaa-kazuyo-sejima-ryue-nishizawa-dean-kaufman-new-art-museum (accessed Apr. 14, 2021).

[28] N. C. Brown, V. Jusiega, and C. T. Mueller, “Implementing data-driven parametric building design with a flexible toolbox,” *Autom. Constr.*, vol. 118, p. 103252, Oct. 2020, doi: 10.1016/j.autcon.2020.103252.

[29] M. D. Bloice, C. Stocker, and A. Holzinger, “Augmentor: An Image Augmentation Library for Machine Learning,” *ArXiv170804680 Cs Stat*, Aug. 2017, Accessed: Apr. 12, 2021. [Online]. Available: http://arxiv.org/abs/1708.04680

[30] Indolences, *5×5 black and white checkered pattern*. 2007. Accessed: May 01, 2021. [Online]. Available: https://commons.wikimedia.org/wiki/File:Checkerboard_pattern.svg

[31] F. Andrianasolo, *andfanilo/streamlit-drawable-canvas*. 2021. Accessed: May 13, 2021. [Online]. Available: https://github.com/andfanilo/streamlit-drawable-canvas

Appendices

Algorithm 2: Interpolation

Input:

- Decoder model
- Performance prediction regression model
- Input data (2)
- Number of desired output designs, *N*

Output:

- Generated interpolated designs
- Corresponding performance scores

Initialization:

- Initialize empty set containing encoded vectors of generated designs of size [N, number of latent dimensions, *d*]
- Initialize start and end point according to input data given

for i in [0, …, d-1] **do**

- Populate previous empty set with evenly spaced points divided between the start and end points according to the number N required

return visualizations of interpolated designs

return performance scores of new designs

Algorithm 3: Multi-Input Interpolation

Input:

- Decoder model
- Performance prediction regression model
- Input data
- Number of desired output designs, *N*

Output:

- Generated interpolated designs
- Corresponding performance scores

Initialization:

- Initialize empty set containing encoded vectors of generated designs of size [N, number of latent dimensions, *d*]
- Initialize pseudo start and end point according to input data given by finding minimum and maximum values on every latent dimension

for i in [0, …, d-1] **do**

- Populate previous empty set with evenly spaced points divided between the pseudo start and end points according to the number N required

return visualizations of interpolated designs

return performance scores of new designs

Algorithm 4: Expansive-Search Performance Interpolation

Input:

- Encoder model
- Decoder model
- Performance prediction regression model
- Input data (2)
- Number of desired output designs, N
- Number of steps per iteration, n_{step}
- Desired percentage change in performance score, Δ
- Sampling radius increment, *rad*

Output:

- Generated design samples
- Corresponding performance scores

Initialization:

- Initialize empty set containing encoded vectors of generated designs of size [N, number of latent dimensions, d]
- Initialize start and end point according to input data given

for h in [0, …, d-1] **do**

- Populate previous empty set with evenly spaced points divided between the start and end points according to the number N required

- Predict performance of all new points
- Calculate starting radius rad_{start} by finding difference distance between new points

for k in [0, …, N-1] **do**

while (performance score of new design / p_{input}) > (1 – Δ)

for i in [0, …, n_{step} - 1] **do**

for q in [0, …, d - 1] **do**

- Perform random search within vicinity of rad_{start} for one point
- Predict performance of new design using regression model, p_{new}

If (p_{new} / p_{input}) < (1 – Δ)

- Update empty set with new design
- Break

Expand search vicinity by *rad*

return visualizations of generated designs

return performance scores of new designs

Algorithm 4: Multi-Input Expansive-Search Performance Interpolation

Input:

- Encoder model
- Decoder model
- Performance prediction regression model
- Input data
- Number of desired output designs, N
- Number of steps per iteration, n_{step}
- Desired percentage change in performance score, Δ
- Sampling radius increment, *rad*

Output:

- Generated design samples
- Corresponding performance scores

Initialization:

- Initialize empty set containing encoded vectors of generated designs of size [N, number of latent dimensions, d]
- Initialize pseudo start and end point according to input data given by finding minimum and maximum values on every latent dimension

for h in [0, …, d-1] **do**

- Populate previous empty set with evenly spaced points divided between the pseudo start and end points according to the number N required

- Predict performance of all new points
- Calculate starting radius rad_{start} by finding difference distance between new points

for k in [0, …, N-1] **do**

while (performance score of new design / p_{input}) > (1 – Δ)

for i in [0, …, n_{step} - 1] **do**

for q in [0, …, d - 1] **do**

- Perform random search within vicinity of rad_{start} for one point
- Predict performance of new design using regression model, p_{new}

If (p_{new} / p_{input}) < (1 – Δ)

- Update empty set with new design
- Break

Expand search vicinity by *rad*

return visualizations of generated designs

return performance scores of new designs