ce/papers Proceedings in civil engineering

Ernst & Sohn A Wiley Brand

Full paper

Design Suggestions for I-Shaped Knee Connections with Conditional Variational Autoencoder

Andreas Müller[1] | Andreas Taras[1] | Martin Vild[2]

Correspondence

Dr. Andreas Müller
ETH Zurich D-BAUG Institute of Structural Engineering
Stefano-Franscini-Platz 5
8093 Zürich
Switzerland
Email:
andreas.mueller@ibk.baug.ethz.ch

[1] ETH Zurich D-BAUG Institute of Structural Engineering, Zurich, Switzerland
[2] Brno University of Technology Faculty of Civil Engineering Institute of Metal and Timber Structures, Brno, Czech Republic

Abstract

Conditional variational autoencoder (CVAE) are used throughout the literature in the sense of performance-based generative design. One main advantage is the possibility of an inverse problem formulation, allowing the exploration of design possibilities/variations. This allows for a more dynamic design, including performance and code-based parameters as a preselection criterion, conditioning the overall design solutions. The work in the presented paper takes this general idea of CVAEs and applies it to connection design, e.g., welded knee connections from I-shaped sections. All created data sets used for the training of the models are based on component-based finite element simulations (CBFEM), performed with the software IDEA StatiCa Connection and the built-in Python API. The overall focus of this paper is specifically set on data creation/post-processing, its manipulation and incorporation in a forward and inverse design loop to demonstrate its possibilities in a more dynamic and intuitive design process compared to the classical workflow.

Keywords

CBFEM, GMNIA Simulations, Gusset plate connections, local and global imperfections

1 Introduction

1.1 Motivation

Over the past decade, Machine Learning (ML) algorithms and Artificial Neural Networks (ANNs) have gained traction in civil engineering applications [1]. Today, a wide spectrum of ML techniques provides innovative solutions for problem-solving and design exploration, enabling engineers to generate, interpret/understand, and expand data more efficiently than ever before.

IDEA StatiCa Connection is a well-established software tool that enables the design of practically any steel connection, regardless of topology and loading, using the component-based finite element method (CBFEM). This overcomes the need of modelling the actual cross-sections, intersections and other operations within the software, but still using shell-based finite elements.

This paper presents research conducted as part of an industrial collaboration with IDEA StatiCa, within the framework of the project "Machine Learning-based Design Optimization of Steel Connections" (MADESCO). The project focuses on integrating deep learning techniques into the software to enhance and accelerate the design and verification processes.

MADESCO's core objective is to develop a novel, hybrid design and optimization tool within IDEA StatiCa Connection, where traditional mechanics-based approaches, such as the finite element method (FEM), are augmented with advanced ML methods. These include data-driven deep learning models trained on extensive datasets of synthetic structural simulations. The overarching strategy supports both forward and inverse design tasks, enabling rapid prediction of the behavior and performance of steel connections, within defined template categories, while ensuring their compliance with design requirements. Forward design is used for a direct prediction of the resistance/utilization with respect to the user input, i.e., material properties, profile geometry, and load contribution. Inverse design, on the other hand, is used to propose optimized solutions (connection configurations) based on specific user-defined conditions. Those might be the profile geometry, material parameters, as well as load combinations, which in general fit several possible design solutions.

https://doi.org/10.1002/cepa.70120

1.2 Limitations and Outline

The results presented in this paper stem from preliminary investigations and reflect only a limited portion of the overall progress achieved within the MADESCO project. To ensure focus and manage computational complexity, the following constraints were applied: the study is restricted to welded frame connections composed exclusively of I-sections. Loading conditions are limited to in-plane shear force–moment interactions. Local imperfections and potential buckling effects were not considered in the simulations. The weld seam was assumed to be sufficiently robust and was therefore excluded as a possible parameter. The finite element (FE) model used was left unmodified, with all parameters set to their default values within IDEA StatiCa Connection. Material behavior was represented using a bilinear stress–strain model with minimal strain hardening, employing a reduced elastic modulus of $E/1000$ to avoid convergence issues. All simulations were conducted using steel grade S355 for the considered cross-sections.

The first part of the paper (Section 2) introduces the overall assumptions made to create the FE models. It further explains the step of data development and extraction to create the data sets used herein. Section 3 is used to describe predictions made by deep neural networks (DNN) in a feed forward sense, meaning that features are used to make a forward prediction of the overall resistance of knee joint connections. Section 4 describes the idea of using a CVAE in connection design; it explains the model architecture and shows the first results made by the network.

2 Data Development

2.1 Selected Parameters

Figure 1 shows the selected knee joint connection for the investigations presented herein.

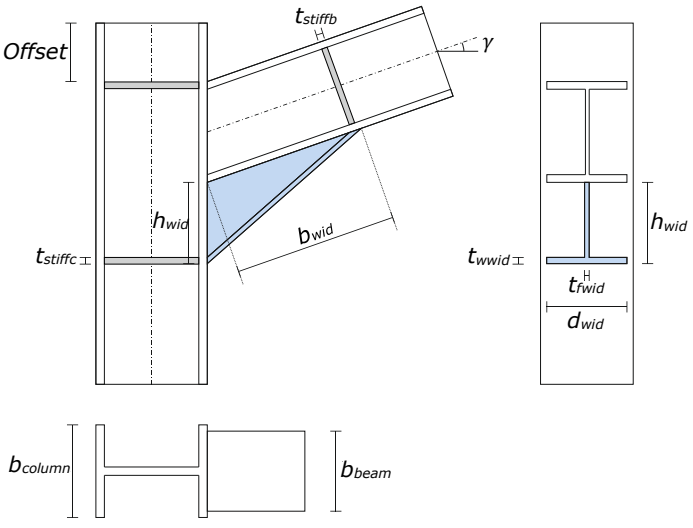

Figure 1 Knee joint connection with chosen parameters

Table 1 gives an overview of all parameters varied for generating the dataset. Their combination is explained below.

The profiles of the columns and beams are combined inside predefined ranges to have similar sizes ($0.8 \cdot b_{column} \leq b_{beam} \leq b_{column}$) and resistances in bending ($0.8 \cdot M_{pl,column} \leq M_{pl,beam} \leq 1.2 \cdot M_{pl,column}$), leading to 196 combinations. The steel grade f_y was assumed to be the same for all components and was not varied between the combinations of beams and columns. For each combination, four randomly chosen values between 0 and 300 mm (always including 0 mm as a hard condition) were chosen for the offset. Two values for the stiffener thickness were chosen randomly for each combination from the values in Table 1. They were placed in the column, either on the upper and lower flange location of the beam (in the intersection) or at the bottom of the widener. The chosen dimensions of the widener are summarized in Table 1. The idea behind randomizing the choice of several parameters leads to a minimization of the overall bias.

Table 1 Specimens properties

Parameter	Range	Amount
Column (Profile x) and Beam (Profile y)	HEA100 – 1000, HEB100 – 1000, IPE80 – IPE600	196
Steel grade f_y [N/mm²]	S235, S275, S355, S450	4
Beam inclination γ [rad]	0°, 15°, 30°, 45°	4
Widener height h_{wid} [mm]	Function of the beam height: $h_{widener} = \frac{1}{2}\ and\ \frac{2}{3}\ h_{beam}$	2
Widener width b_{wid} [mm]	Function of the widener height: $b_{widener} = \left[1, \frac{4}{3}, \frac{5}{3}, 2\right] \cdot h_{widener}$	4
Widener's depth d_{wid} [mm]	Depth: $d_{widener} = b_{beam}$	1
Widener flange-thickness t_{fwid} [mm]	Flange thickness: $t_{f,widener} = t_{f,beam}$	1
Widener web-thickness t_{wwid} [mm]	Web thickness: $t_{w,widener} = t_{w,beam}$	1
Stiffener's thickness in column t_{stiffc} [mm]	10, 12, 15, 18, 20, 22, 25	2
Offset [mm]	0–300	4
Load	V-M Interaction	15

2.2 Dataset generation

The dataset generation was done in an automated way using the IDEA StatiCa API in combination with Python. Therefore, several parametrized templates, see Figure 2, were updated in a loop with the defined parameters from Table 1 and analyzed in IDEA StatiCa. The overall resistance of the connection was saved as the target value, on which the neural network models were trained on in the next step. Due to computational recourses and the time needed for the generation of the full data set, using all parameters from above, the dataset was strongly reduced, by random down-sampling. The total amount of combinations was reduced from 2.822.400 to 874.946.

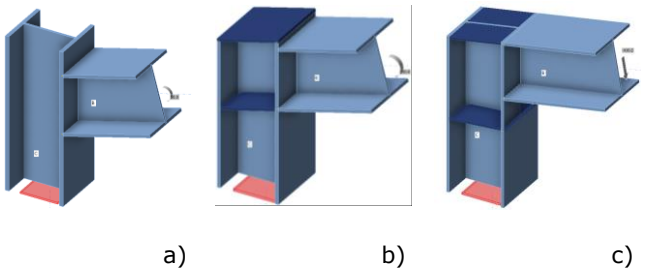

Figure 2 Used connection templates for the dataset development

This reduction was motivated by preliminary investigations, which revealed that the shapes of the V-M interaction "curves" exhibit a very high degree of similarity. Consequently, not every simulated resistance value is necessary for the neural network to achieve accurate predictions during training. An illustrative example is provided in Figure 3, where the thickness of the column stiffener was incrementally increased. The simulation results demonstrate that the modified parameter (stiffener thickness) has a predictable, scalable influence on the structural resistance—namely, increasing the thickness generally leads to higher resistance values. Initial studies indicated that a reduction of up to 30% of the full dataset was sufficient to maintain high prediction accuracy across the entire dataset. Nevertheless, these findings are preliminary and part of ongoing research that will be published in future work.

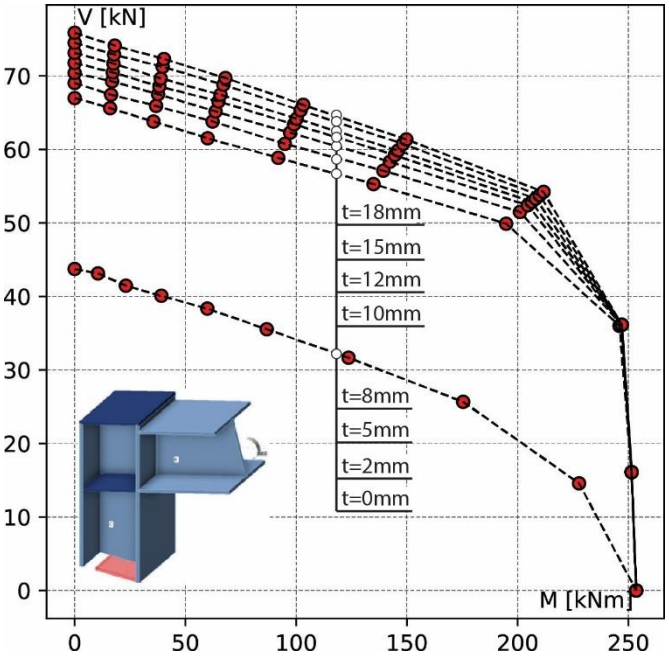

Figure 3 V-M interaction resistances for a knee joint connection made from HAE240 profiles including column stiffener

3 Forward Design

3.1 General Concept

The concept of neural networks dates back to the 1940s, with foundational work by McCulloch and Pitts [4], later expanded by D. Hebb [5] and M. Minsky and S. Papert [6]. Originating from efforts to logically model the transmission of impulses between individual neurons within a larger nervous system, deep learning methods have since evolved and are now widely applied across various domains within architecture, engineering, and construction. This growing popularity is driven by greater access to large datasets, advancements in graphics processing units (GPUs), continuous development of algorithms, and significantly easier access to machine learning through the availability of high-level libraries and APIs—compared to previous decades.

$$y(x) = a \cdot \left(\left(\sum W_n \cdot x_n \right) + b \right) \tag{1}$$

An artificial neuron consists of three main components: weights, a bias, and an activation function, and is typically represented as shown in Eq. (1). The input parameters (features) are multiplied by randomly initialized weights, which are updated over the course of training across multiple epochs. These weights reflect the strength of connections within the network. Parameters that do not significantly affect the prediction are progressively assigned small values during training. The bias is an additional trainable parameter (which can be zero or nonzero) that is added to the sum of the weighted inputs. Mathematically, a neuron is represented as a single vector, which is then passed through an activation function characterized by a predefined threshold. This basic concept can be extended to more complex systems by adding multiple neurons and layers, thereby increasing the number of trainable parameters to better handle non-linear problems.

At the end of each network prediction, a loss function is used to quantify the deviation between the predicted and the true values. For regression problems, two common loss definitions are typically employed: the mean squared error (MSE) and the mean absolute error (MAE), as shown in Equations (2) and (3). Based on the calculated loss, the weights and biases of the network are updated through gradient descent and backpropagation of the computational graph. This involves computing the derivative of each node by applying the chain rule and adjusting the parameters using the gradient descent algorithm, as outlined in Equations (4) and (5).

$$MSE: L = \frac{1}{N}\sum_{i=1}^{N}(\delta_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2}$$

$$MAE: L = \frac{1}{N}\sum_{i=1}^{N}|\delta_i| = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3}$$

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial v} \cdot \frac{\partial v}{\partial x} \tag{4}$$

$$x^{(j+1)} = x^{(j)} - \eta \cdot \nabla f\left(x^{(j)}\right) \tag{5}$$

3.2 Model Architecture and Evaluation

The DNN architecture is strongly based on previous publications [2, 7, 8], where similar problems were investigated predicting the non-linear resistance of cross-sections, as well as the overall pre- and post-buckling behaviour. Table 1 shows the chosen hyperparameters for the DNN model and the training process. The network optimization was performed on the basis of a train and test philosophy, meaning that a specific data amount was used for the training (80%) and an additional independent

amount for the validation process (20%).

Table 1 DNN architecture and hyperparameters

Model Parameter	Selected Properties
Hidden layer 1	54
Hidden layer 2	54
Hidden layer 3	27
Hidden layer 4	18
Activation function	ReLU
Optimizer	Adam
Learning rate	0.0005
Epochs	2000
Batch size	64

All used features (input values) are summarized in Table 2 as follows. For the training process it is necessary to transform or scale the values to eliminate the major problem of multiple features having different magnitudes, ranges and units. The usual way to do so is by data normalization (scaling the magnitudes of features between values of 0 and 1 or -1 and 1, corresponding to the lowest and highest value of the feature) and data standardization (mean value of the feature is set to 0 and the standard deviation to 1; see Equations (6) and (7).

$$\hat{x}^{(i)} = \frac{x^{(i)} - x_{min}^{(i)}}{x_{max}^{(i)} - x_{min}^{(i)}} \tag{6}$$

$$\tilde{x}^{(i)} = \frac{x^{(i)} - \bar{x}^{(i)}}{\sigma} \tag{7}$$

Preliminary investigations showed that standardizing the data led to better performance and prediction results.

Table 2 Used features in the forward prediction

Group name	Features
Profile x	A, I_y, W_{el}, W_{pl}, f_y
Profile y	A, I_y, W_{el}, W_{pl}, f_y
Joint geometry	γ, *Offset*
Widener	h_{wid}, b_{wid}, d_{wid}, $t_{f,wid}$, $t_{w,wid}$
Stiffener (column)	t_{stiff}
Loading	$V_{contribution}$, $M_{contribution}$

The loading, i.e. the V-M interaction had to be normalized in a different manner. From the perspective of the user only the specific load combination, acting on the connection, is known, which might be higher or lower than the actual overall resistance. Therefore, it is more appropriate to calculate the geometric normalized contribution of the acting shear force and moment; see Equation (8). The predicted target can be than formulated as a combination of the simulated shear and moment resistance; see Equation (9). The actual prediction process is formulated in Equation (10), using the input contribution and the plastic profile resistance of the beam.

$$V_{contribution} = \frac{V_{Ek}/V_{pl,y}}{\sqrt{\left(V_{Ek}/V_{pl,y}\right)^2 + \left(M_{Ek}/M_{pl,y}\right)^2}}$$

$$M_{contribution} = \frac{M_{Ek}/M_{pl,y}}{\sqrt{\left(V_{Ek}/V_{pl,y}\right)^2 + \left(M_{Ek}/M_{pl,y}\right)^2}} \tag{8}$$

$$target = \hat{y} = \sqrt{\left(V_{Rk}/V_{pl,y}\right)^2 + \left(M_{Rk}/M_{pl,y}\right)^2} \tag{9}$$

$$\hat{V}_{Rk}^{(i)} = \hat{y}^{(i)} \cdot V_{contribution} \cdot V_{pl,y}$$

$$\hat{M}_{Rk}^{(i)} = \hat{y}^{(i)} \cdot M_{contribution} \cdot M_{pl,y} \tag{10}$$

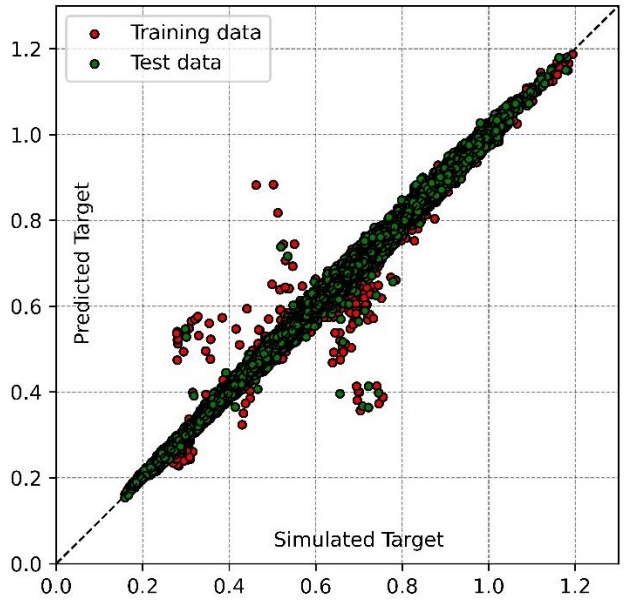

Figure 4 Comparison between DNN model predictions and IDEA StatiCa simulation results

Figure 5 shows the predictions made by the neural network (y-axis) in comparison to the simulation results by IDEA StatiCa (x-axis). First training attempts showed a good alignment and high prediction accuracy in almost all considered cases with an achieved $r^2 > 0.995$ for training, as well as testing data. The majority of back calculated V-M interaction resistances match the simulated results, as illustrated in Figure 6. The red dots symbolize the available resistances used for the training of the DNN model. The grey dots, on the other hand, the predicted results, filling up the spaces between the simulated results. However, there is a higher deviation for some profiles than for others, which follows a certain logic. For particularly small profile sizes, the prediction error appears to increase drastically. This may be due to the fact that there are fewer possible combinations in these areas or that the DNN model is not properly capturing the boundary areas (very small or very big profiles). One additional factor is the shape of the interaction curve, which might differ for smaller profiles from the "mean" interaction curve.

Generative design has emerged as a key application area in recent years, particularly in fields that are highly visible

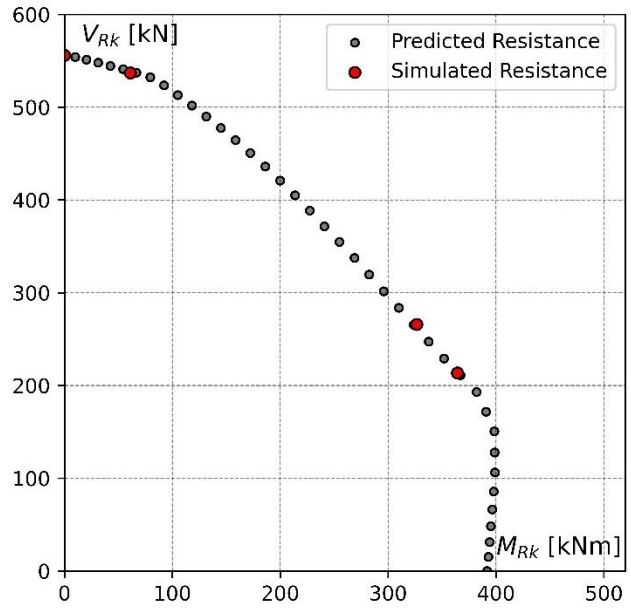

Figure 5 V-M interaction resistances for a randomly selected profile

4 Inverse Design

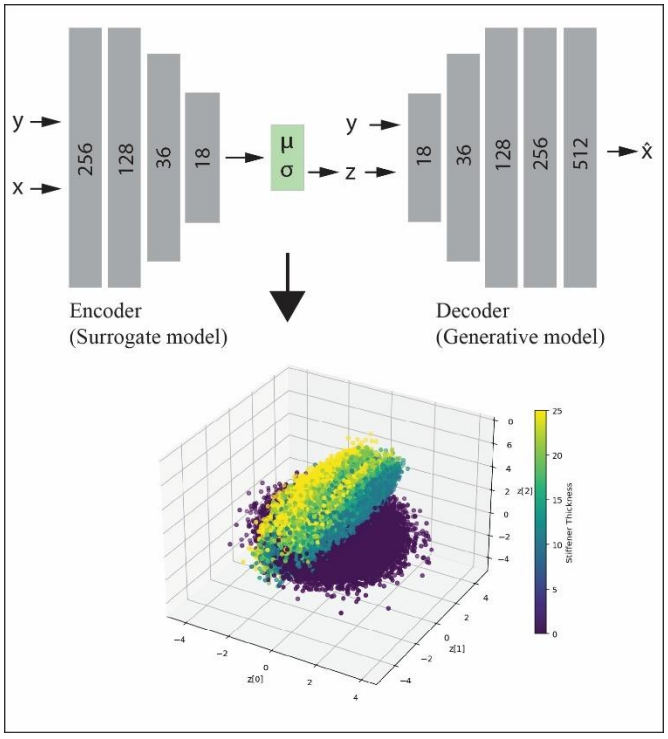

Figure 6 Used CVAE mode and 3D latent space representation

and accessible to the general public, such as computer vision and natural language processing. In contrast, its use in civil engineering remains largely underexplored, primarily due to challenges related to data scarcity, variability, and limited accessibility. However, as outlined in Sections 2 and 3, data-driven simulation is well-suited to the design of steel structures—provided that the underlying finite element (FE) models are thoroughly validated.

While Section 3 focuses on feedforward prediction, this part shows the use of a conditional variational autoencoder (CVAE) to investigate the design space and enable inverse predictions based on specified performance criteria. This approach offers engineers a more intuitive and interactive design tool. Similar concepts have been applied in previous studies—such as in the design of vertical gardens or pedestrian bridges [9, 10]. In this framework, the classical CVAE architecture is adopted as part of a preliminary study into design space exploration, highlighting its potential benefits and advantages over traditional feedforward methods.

The architecture of the selected Conditional Variational Autoencoder (CVAE) is illustrated in the upper part of Figure 6. The architecture of the encoder slightly differs from the one of the DNN model from Section 3. Further, the encoder architecture is inverted, whereby an additional layer was added, and used for the decoder part. The encoder takes the preset conditions y, along with a set of features x, and outputs the mean and standard deviation vectors, as well as the latent vector z. Table 3 lists the associated features and conditions, which follow the typical workflow in IDEA StatiCa Connection. At the stage of connection design, certain parameters—such as profile geometry, material properties, offset, inclination, and applied loads—are already known and treated as conditions y. The features x, in contrast, represent design variables (e.g., stiffener thickness and widener geometry) that are typically adjusted manually by the user.

Table 3 Used features in the forward prediction

Features	Conditions
t_{stiff}, h_{wid}, b_{wid}, d_{wid}, $t_{f,wid}$, $t_{w,wid}$	***Profile x***: h_x, b_x, t_{wx}, t_{fx}, f_y ***Profile y***: h_y, b_y, t_{wy}, t_{fy}, f_y γ, Offset, V_{norm}, M_{norm}

The decoder reconstructs the features using the latent vector z and the conditions y. Each block in both the encoder and decoder consists of a fully connected layer followed by a Rectified Linear Unit (ReLU) activation function. The latent space is defined with 8 dimensions, meaning that μ, σ, and z are each represented by 8-dimensional vectors. The performance of the decoder is heavily influenced by the loss function, which combines the mean squared error (MSE) for reconstruction accuracy and the Kullback-Leibler (KL) divergence to measure how closely the model distribution Q approximates the true distribution P. Balancing these two components during training is critical, as it governs the sampled variability and quality of the decoder's outputs. Once the CVAE model is trained, the decoder can be used to generate samples based on defined conditions from the latent space. A representation is shown in the lower part of Figure 6, using the stiffener thickness to colour the labels. A clear separation between the stiffener thickness t_{stiff} = 0 mm and t_{stiff} > 0 mm can be identified. This means that beside of random sampling from this representation it might be also possible to logically navigate through the space.

A randomly selected knee joint configuration—using an HEB650 profile for both the column and beam, steel grade S450, an offset of 240 mm, γ = 0 and V_{norm}=0.66; M_{norm}=0.26, was used to condition and sample from the decoder. The DNN model from Section 3 then predicted the corresponding V-M interaction resistances for the sampled combinations. Figure 7 illustrates the possible outputs

generated by combining the inverse and forward models. The red dot represents the conditioned load input, which all sampled configurations are intended to match. The green dots show the predicted resistances based on the decoder's sampled features. While the resistances align at the red dot, they exhibit greater variation in pure bending. This variation reflects the sampled diversity of feature combinations, as detailed in Table 4.

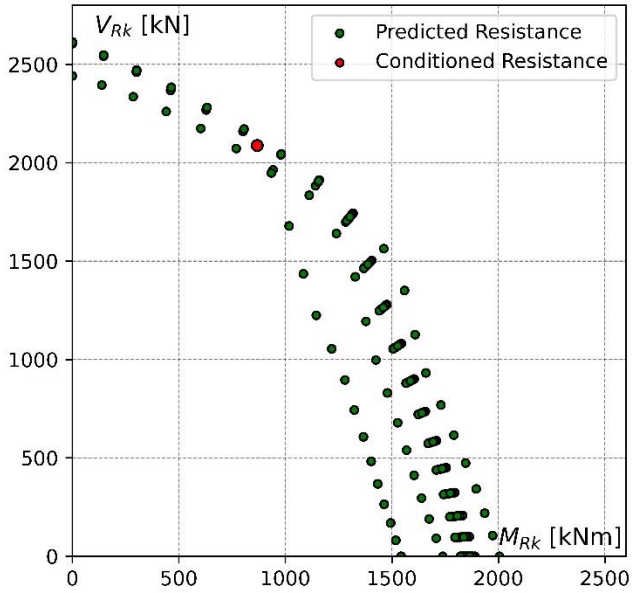

Figure 7 Sampled design possibilities (inverse design) and predicted interaction curves (forward design)

Table 4 Selected (rounded) outputs of the sampled features in [mm]

t_{stiff}	h_{wid}	b_{wid}	d_{wid}	$t_{f,wid}$	$t_{w,wid}$
0	0	0	0	0	0
10	0	0	0	0	0
22	0	0	0	0	0
10	140	125	16	4	3.5

5 Conclusion and Outlook

This paper shows preliminary investigations on a data driven strategy for connection design using deep learning concepts to solve forward and inverse design tasks. Based on generated numerical datasets, using the software IDEA StatiCa Connection, the first part of the paper explains the forward design idea using a fully connected deep neural network. First results show a high accuracy overall, although outliers were identified for smaller profile groups, e.g. HEA 100, HEB 100. This point is subject of current research and is considered in the context of hyperparameter tuning and active learning. The latter is primarily intended to answer questions regarding data set size and distribution within an automated training and data generation workflow.

The second part of this paper explains the inverse design idea using a Conditional Variational Autoencoder (CVAE), presenting a possible streamline from a user-based perspective. First results demonstrate a promising pathway, integrating inverse design to propose already feasible connection types and forward design to quickly predict the associated interaction resistances. This allows the user to explore design alternatives much faster and more intuitively compared to the classical workflow, with the information of the overall connection behaviour.

Future work will fucus on refining the forward and inverse design model architectures and extend the methodology to cover a broader range of connection types. The question of how much data is needed to make reliable and accurate predictions is crucial when it comes to generating data sets with reasonable computational effort. This is where *active learning* presents a promising strategy, where data is generated iteratively during the training process, targeting regions of high error or model uncertainty to maximize learning efficiency.

References

[1] Thai, H.-T (2022). *Machine Learning for Structural Engineering: A State-of-the-Art Review*. Structures 2022, 38, 448–491. https://doi.org/10.1016/j.istruc.2022.02.003.

[2] Müller, A.; Taras, A.; Kraus, M. A. (2022) *Scientific Machine and Deep Learning Investigations of the Local Buckling Behaviour of Hollow Sections.* ce/papers 2022, 5 (4), 1034–1042. https://doi.org/10.1002/cepa.1848.

[3] Haghighat, E.; Juanes, R. (2021) *SciANN: A Keras/TensorFlow Wrapper for Scientific Computations and Physics-Informed Deep Learning Using Artificial Neural Networks.* Comput. Methods Appl. Mech. Eng. 2021, 373, 113552.

[4] McCulloch WS, Pitts W. (1943) *A logical calculus of the ideas immanent in nervous activity*. The bulletin of mathematical biophysics. 5:115–33.

[5] Hebb D. (1949) *The organization of behavior*. New York.

[6] Minsky M, Papert SA. (2017) *Perceptrons, Reissue of the 1988 Expanded Edition with a new foreword by Léon Bottou: An Introduction to Computational Geometry*. MIT press.

[7] Müller A, Taras A. Prediction of the local buckling strength and load-displacement behaviour of SHS and RHS members using Deep Neural Networks (DNN)–Introduction to the Deep Neural Network Direct Stiffness Method (DNN-DSM). Steel Construction. 2022;15(S1):78–90.

[8] Müller A, Taras A. (2023) *Prediction of the deformation and local buckling behavior of structural systems using the deep neural network direct stiffness method (DNN-DSM).* In: Proceedings of the Annual Stability Conference Structural Stability Research Council [Internet]. Charlotte, North Carolina p. 18.

[9] Salamanca L, Apolinarska AA, Pérez-Cruz F, Kohler M. (2022) *Augmented Intelligence for Architectural Design with Conditional Autoencoders: Semiramis Case Study.* In Berlin, Germany. p. 108–21.

[10] Balmer VM, Kuhn SV, Bischof R, Salamanca L, Kaufmann W, Perez-Cruz F, et al. *Design Space Exploration and Explanation via Conditional Variational Autoencoders in Meta-model-based Conceptual Design of Pedestrian Bridges*