

Eberhard Karls Universität Tübingen
Mathematisch-Naturwissenschaftliche Fakultät
Wilhelm-Schickard-Institut für Informatik

Bachelor's Thesis in Cognitive Science

Topological Pattern Recognition in Experimental Configurations for the Detection of Gravitational Waves

Raphael Jontofsohn

27.07.2026

Reviewer

Prof. Dr. Mario Krenn

Artificial Scientist Lab
Wilhelm-Schickard-Institut für Informatik
University of Tübingen

Supervisor

Jonathan Klimesch

Artificial Scientist Lab
Wilhelm-Schickard-Institut für Informatik
University of Tübingen

Name, first name:

Raphael Jontofsohn

Student ID:

6408525

Topological Pattern Recognition in Experimental Configurations for the Detection of
Gravitational Waves

Bachelor's Thesis in Cognitive Science

Eberhard Karls Universität Tübingen

Period: 16.02.2026–27.07.2026

Abstract

Gravitational-wave detectors provide a glimpse into astrophysical events occurring in distant galaxies. Designing these highly sensitive laser interferometers requires exploring a vast search space and has traditionally relied on human expertise. We analyzed 34,239 broadband configurations based on the quasi-universal interferometer, each optimized via gradient descent using the Differometer physics simulator. Each configuration is characterized by three modalities: a discrete topology, continuous physical parameters, and the resulting strain sensitivity curve summarized by a scalar loss. Searching for structural patterns at multiple levels, we found that individual parameters could not explain low loss values. Sensitivity-curve clustering identified eight groups, six of which corresponded to topologies from the re-optimization group. To jointly map all three modalities into a shared 32-dimensional latent space, we developed a multimodal variational autoencoder. The resulting representation contained several topology-related clusters, and the local organization reflected gradients in detector loss. The results indicate that detector performance emerges from coordinated, nonlinear interactions between topology and continuous parameters rather than from simple universal design rules.

Contents

1	Introduction	1
2	Background	2
2.1	Automated Detector Design	2
2.2	Sensitivity Curves and Loss Functions	3
2.3	Principal Component Analysis	4
2.4	Autoencoders	4
2.5	Variational Autoencoders	4
2.6	Clustering Methods and UMAP Visualization	6
2.7	Token-Based and Transformer-Based Topology Encoding	7
3	Dataset and Preprocessing	8
3.1	Overview	8
3.2	Categorization	8
3.3	Scaling	10
4	Methods	11
4.1	Parameter-Level Analyses	11
4.2	Sensitivity-Curve Clustering	11
4.3	VAE on Sensitivity Curves	12
4.4	Multimodal Architecture Family	14
4.5	Patch-Based Sprout-Inspired Multimodal VAE	15
4.6	General Patch-Based Model	17
4.6.1	Configuration Representations	17
4.6.2	Common Patch-Based Processing	17
4.7	Structured Topology-Aware VAE	18
4.7.1	Shared Token Representation	18
4.7.2	Structured Encoder and Reconstruction Objective	18
5	Results	20
5.1	Skewed Parameter Distributions	20
5.2	No Evidence for Universal Important Parameters	21
5.3	Sensitivity-Curve Clustering	21
5.4	VAE Sensitivity Analysis	22
5.5	Multimodal VAE Results	23
5.5.1	Optimizable-Parameter Models	23
5.5.2	Complete Configuration Models	23
6	Discussion and Conclusion	25
	Bibliography	27
A	Additional Results	29

1 Introduction

Gravitational waves (GWs) propagate through the universe as distortions of spacetime. They interact only weakly with matter, allowing them to travel over cosmological distances with little attenuation, making them a valuable source of information for astrophysical events. A GW detector must be sensitive across different frequency bands to observe different kinds of phenomena. The Advanced Laser Interferometer Gravitational-Wave Observatory (aLIGO) enabled the first direct observation of GWs in 2015 [1]. It is based on a dual-recycled Fabry–Pérot Michelson interferometer with 4 km arms.

Motivated by the challenge to design better GW detectors, Krenn et al. [2] introduced a quasi-universal interferometer (UIFO), a highly expressive parameterized optical interferometer. Each UIFO configuration consists of a fixed topology on an $n \times n$ grid, encoding the optical components such as beam splitters. These components are described by different continuous parameters, such as beam splitter reflectivity. The physical behavior of the configuration is then simulated computationally, resulting in a strain sensitivity curve sampled at discrete frequency points. This sensitivity curve is summarized as a single loss value.

To optimize such an optical interferometer, Klimesch [3] developed Differometer, a JAX-based differentiable interferometer simulator. This made it possible to investigate the continuous-parameter search space of GW detectors on a large scale using gradient-based optimization algorithms. The resulting dataset contains 34,239 broadband optimization runs of UIFO designs, most of which are based on a 3×3 grid.

This large dataset introduces several scientific challenges. First, each run starts from a randomly initialized discrete topology, which remains fixed during the subsequent parameter optimization, separating discrete topology search from continuous optimization. Second, across 3×3 UIFO configurations, the median number of optimizable parameters is 192, while the whole dataset contains 271 distinct optimizable parameters for this UIFO size. Their physical interactions are highly nonlinear, and the resulting optimization objective is expected to be non-convex. Additionally, the system appears to be highly overparameterized, having more degrees of freedom than necessary to describe its behavior. Symmetries, interchangeable components, and compensating parameter effects may create strong degeneracy, where different parameter choices produce similar sensitivity curves.

The thesis investigates recurring structures that characterize low-loss broadband UIFO configurations. We first examine how individual parameters and parameter types differ between the full dataset and the 10% of runs with the lowest loss and evaluate how parameter substitutions affect the loss. We then cluster sensitivity curves to test whether similar sensitivity behavior is associated with recurring configuration structures. A variational autoencoder (VAE) is used to analyze sensitivity curves in a low-dimensional latent space. Finally, we compare different topology representations and jointly encode topology, physical parameters, and sensitivity curves in a shared multimodal latent space. The analysis is based on existing optimization results rather than on newly performed optimizations.

2 Background

2.1 Automated Detector Design

The design of GW detectors can be formulated as an optimization problem. Laser light propagates through an arrangement of mirrors, beam splitters, squeezing sources, and other components whose parameters determine the detector's sensitivity. Their nonlinear interactions produce a loss function expected to be non-convex and difficult to optimize.

The UIFO serves as a flexible template for encoding discrete topologies and continuous parameters, as illustrated in Figure 2.1 [2]. It consists of $n \times n$ central cells and $4n$ boundary cells. Each central cell contains either a beam splitter or a four-port Faraday isolator, referred to in the dataset as a directional beam splitter. The central element is enclosed by four mirrors. Light can be injected through boundary cells, while the output signal is measured at a homodyne detection cell. The components are parameterized by continuous quantities such as reflectivity, laser power, squeezing parameters, mirror masses, and propagation lengths.

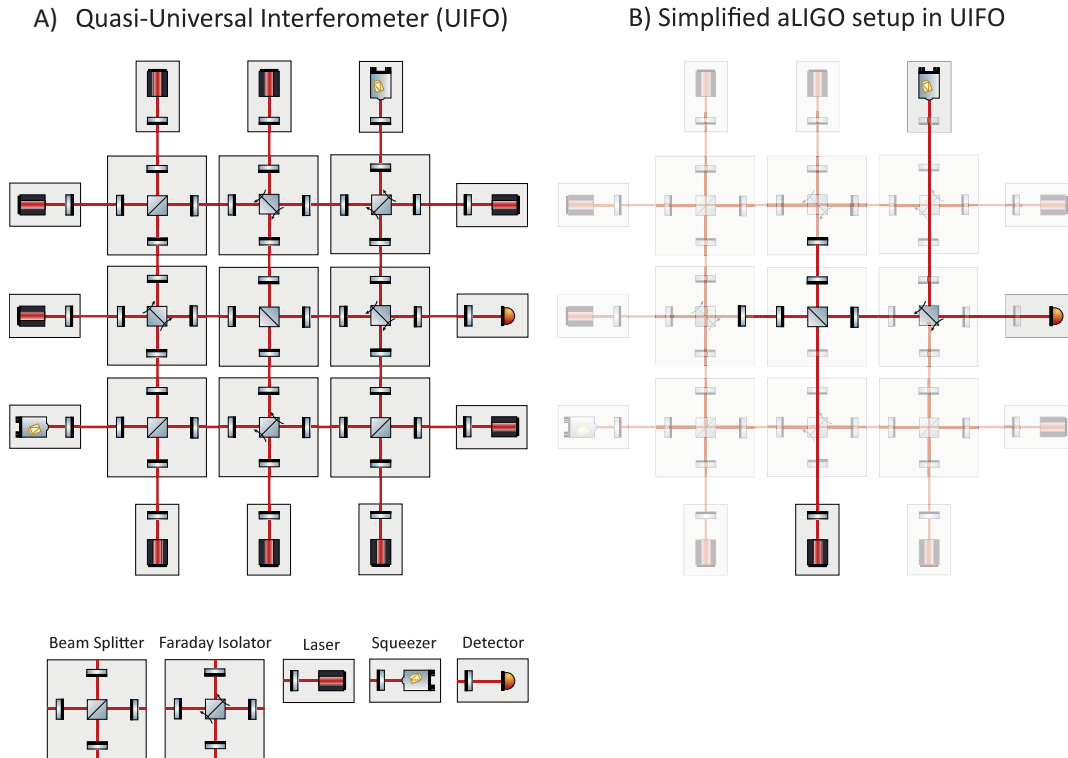


Figure 2.1: UIFO representation. (A) A 3×3 template consisting of nine central cells. Each cell contains either a beam splitter or a Faraday isolator enclosed by four mirrors. Light from lasers or squeezed vacuum sources is injected through boundary cells and the output is measured at a boundary detector. (B) Simplified aLIGO configuration represented within the UIFO template. Transparent components represent unused elements. Reproduced with permission from [3].

Even with only one laser, ten optical elements, and two detectors, the design space contains more than 100 million distinct configurations, motivating automated search methods. To support the search in such a large design space, Differometer was developed as a differentiable frequency-domain interferometer simulator implemented in Python using the JAX framework [3]. It uses just-in-time (JIT) compilation, graphics processing unit (GPU) execution, and automatic differentiation to compute steady-state power distributions and sensitivity curves. Differometer enables the search for new GW detector designs but does not by itself reveal recurring structures within large collections of optimized runs.

2.2 Sensitivity Curves and Loss Functions

Strain sensitivity, the central UIFO performance measure, describes the smallest resolvable GW strain amplitude across frequency. It is represented as an amplitude spectral density in $\frac{1}{\sqrt{\text{Hz}}}$ and sampled at 50 logarithmically spaced frequencies. Lower values indicate better sensitivity.

Each configuration is evaluated using a scalar objective function for optimization. Throughout this thesis, “loss” always refers to this total loss, which combines a strain-sensitivity term with power-violation penalties that discourage configurations with high internal optical power, which could damage components.

The strain-sensitivity component, $\mathcal{L}_s$, is determined by comparing the simulated sensitivity curve $S(f_i)$ to a reference curve $S_{\text{ref}}(f_i)$ at the same frequency points. In this thesis, the reference curve is based on the Voyager detector proposed by LIGO, which is a next-generation detector concept that uses cryogenic silicon technology [4]. The sensitivity term is computed as the mean logarithmic ratio between the simulated and the reference curve

$$\mathcal{L}_s = \frac{1}{N} \sum_{i=1}^N \log_{10} \frac{S(f_i)}{S_{\text{ref}}(f_i)}.$$

Thus, a strain-sensitivity term of zero indicates that a configuration achieves the same average logarithmic sensitivity as the Voyager baseline across all sampled frequencies. Consequently, negative values denote configurations whose sensitivity outperforms this reference overall.

The loss depends on the frequency range over which the sensitivity curve is evaluated. In this thesis, the main focus is the broadband range from 20 Hz to 5 kHz, which covers a wide spectrum of GW signals, ranging from low-frequency signals such as cosmological GW backgrounds to higher-frequency signals such as core-collapse supernovae and post-merger signals from neutron star mergers [2].

Because the loss is computed through a differentiable JAX-based simulation pipeline, it can be optimized with gradient-based methods such as Adam. However, the scalar loss is not an injective representation of the sensitivity curve, as different curve shapes can result in the same loss value. This limits analyses based solely on the scalar performance score.

2.3 Principal Component Analysis

To summarize high-dimensional data, principal component analysis (PCA) finds orthogonal basis vectors that maximize the data variance, keeping only the important principal components [5]. Because PCA is linear, it may fail to capture complex nonlinear structure.

2.4 Autoencoders

Autoencoders (AEs) provide a nonlinear alternative to such linear dimensionality reduction methods [6]. They are artificial neural networks consisting of three parts: an encoder, a latent space, and a decoder. The encoder maps the high-dimensional input x to a latent vector $z = g_\varphi(x)$, where φ denotes its weights. The latent space acts as an information bottleneck before reconstruction [7]. The decoder reconstructs $\hat{x} = h_\theta(z)$, where θ denotes its weights. Both networks are trained by backpropagation to minimize the reconstruction loss $L_{\text{rec}}(\theta, \varphi; x)$ between x and $\hat{x}$. For continuous data, the mean squared error (MSE) is commonly used, formulated as

$$\mathcal{L}_{\text{rec}}(\theta, \varphi; x) = \frac{1}{D} \|x - \hat{x}\|^2, \quad (1)$$

where D denotes the number of dimensions of the data space. While the reconstruction of the original data often works well for standard autoencoders, the learned representation may be sensitive to perturbations. Without regularization of the latent space, the regions between the training samples may be irregular and difficult to interpolate or sample from.

2.5 Variational Autoencoders

Variational autoencoders (VAEs) provide a framework to learn deep latent-variable models and corresponding inference models [8]. While standard AEs map data deterministically, VAEs introduce a probabilistic latent representation. Instead of learning a single code for each input, the encoder predicts a distribution over latent variables, from which latent samples are drawn during training. This encourages a more continuous and structured latent space.

VAEs can reveal latent structure in complex scientific design spaces and have been applied to quantum-optics experiments [9].

The VAE assumes that the observed data x are generated from unobserved continuous latent variables z drawn from a prior distribution $p(z)$. A standard normal distribution, $p(z) = \mathcal{N}(0, I)$, is commonly chosen as the prior.

The posterior $p(z | x)$ describes the latent variables associated with an observation x , whereas the likelihood $p_\theta(x | z)$ describes how an observation is generated from a latent variable.

To infer the latent vector from a given data point, the true posterior distribution

$$p(z|x) = \frac{p(x|z) \cdot p(z)}{p(x)} \quad \text{where} \quad p(x) = \int p(x | z) \cdot p(z) \, dz$$

must be calculated, where $z \in \mathbb{R}^d$ denotes a d -dimensional latent variable. However, computing the marginal likelihood $p(x)$, also called the evidence, requires integrating over all possible latent

variables. For nonlinear generative models, this integral generally has no tractable closed-form solution [8].

To address this problem, the VAE framework employs variational inference [10]. Instead of computing the intractable exact posterior, one implements a probabilistic encoder $q_\varphi(z|x)$ that approximates it. This encoder outputs a probability distribution, defined by a mean μ and variance σ^2 , from which a latent vector z is then sampled.

The optimization objective for the VAE is the evidence lower bound (ELBO). Kingma and Welling [11] show that it can be decomposed into two terms

$$\mathcal{L}_{\text{ELBO}}(x) = \underbrace{\mathbb{E}_{z \sim q_\varphi(z|x)} [\log p_\theta(x|z)]}_{\text{Expected Log-Likelihood}} - \underbrace{D_{\text{KL}}(q_\varphi(z|x) \parallel p(z))}_{\text{KL Divergence}}.$$

The stochastic latent sampling must now be incorporated into the training objective. The first term evaluates how well the model reconstructs the data. The sampled latent vector z is passed to a decoder that parameterizes the likelihood $p_\theta(x|z)$. The decoder network produces a reconstruction $h_\theta(z) = \hat{x}$, which represents the mean of this distribution. Assuming a Gaussian likelihood with fixed variance, maximizing the expected log-likelihood is equivalent, up to a constant factor, to minimizing the mean squared error between the input x and its reconstruction $\hat{x}$ (see Equation 1). Thus, the reconstruction part of the loss function can be formulated as the expected value

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{z \sim q_\varphi(z|x)} [f(z)], \quad \text{where} \quad f(z) = \frac{1}{D} \|x - h_\theta(z)\|^2.$$

Here, D denotes the number of dimensions of the input space. To train the VAE by backpropagation, gradients of $\mathcal{L}$ with respect to the network parameters φ and θ must be computed. For the decoder parameters θ , the gradient calculation remains as in the standard AE case, because the distribution does not depend on θ

$$\nabla_\theta \mathbb{E}_{z \sim q_\varphi(z|x)} [f(z)] = \mathbb{E}_{z \sim q_\varphi(z|x)} [\nabla_\theta f(z)].$$

For the encoder parameters φ , this step is not directly possible because φ also determines the distribution from which z is sampled.

$$\nabla_\varphi \mathbb{E}_{z \sim q_\varphi(z|x)} [f(z)] = \nabla_\varphi \int q_\varphi(z|x) f(z) dz.$$

A score-function estimator such as REINFORCE could be used, but its high variance makes optimization of deep neural networks unstable [12].

To resolve this, Kingma and Welling [11] propose the reparameterization trick, which expresses stochastic sampling as a deterministic transformation of parameter-independent noise. They achieve this by using a deterministic encoder function g_φ to output the mean μ and the variance σ^2 of the distribution $q_\varphi(z|x)$. The randomness is then injected via an independent, standard normally distributed noise variable ε resulting in

$$z = \mu + \sigma \odot \varepsilon \quad \text{with} \quad \varepsilon \sim \mathcal{N}(0, I). \quad (2)$$

This corresponds to sampling a random point from a standard normal distribution, scaling it by the standard deviation of the latent distribution and shifting it by its mean. Because the auxiliary noise variable ε is independent of φ , gradients can be propagated through the deterministic transformation

$$\nabla_{\varphi} \mathbb{E}_{\varepsilon \sim p(\varepsilon)} [f(g_{\varphi}(\varepsilon, x))] = \mathbb{E}_{\varepsilon \sim p(\varepsilon)} \nabla_{\varphi} f(g_{\varphi}(\varepsilon, x)).$$

With the reparameterization trick enabling low-variance gradient estimation for the encoder, the complete network can now be trained with standard stochastic gradient descent [11]. The VAE is trained by minimizing the negative ELBO, which consists of a reconstruction term and a KL-divergence term

$$\mathcal{L}_{\text{VAE}} = \underbrace{\frac{1}{D} \|x - h_{\theta(z)}\|^2}_{\text{Reconstruction Loss}} + \underbrace{D_{\text{KL}}(q_{\varphi}(z|x) \parallel p(z))}_{\text{KL Divergence (Regularization)}}. \quad (3)$$

The beta-VAE extends the conventional VAE objective by introducing a weighting factor β for the KL-divergence term [13]. The resulting objective can be written as

$$L_{\beta\text{-VAE}} = \underbrace{L_{\text{Rec}}}_{\text{Reconstruction}} + \underbrace{\beta \cdot D_{\text{KL}}(q(z|x) \parallel p(z))}_{\text{Regularization}}. \quad (4)$$

The parameter β controls the balance between reconstruction accuracy and latent-space regularization. Values above one impose stronger pressure on the approximate posterior $q_{\varphi}(z|x)$ to match the prior $p(z)$, typically a normal distribution. Values below one allow the latent variables to retain more information about the input, potentially overfitting the data. In the present work, β was treated as a tunable regularization parameter. To prevent regularization before reconstructions had been learned, β was increased linearly from zero to its final value during an initial warm-up period. This training strategy is commonly referred to as KL annealing.

2.6 Clustering Methods and UMAP Visualization

K-means clustering partitions data into a predefined number of clusters k [14]. In practice, the objective is commonly optimized using Lloyd’s algorithm, which iteratively assigns each point to the nearest cluster center and updates each center to the mean of its assigned points [15]. Since the objective minimizes squared distances to these centroids, k-means is best suited for approximately isotropic clusters.

One common way to evaluate different values of k is the silhouette score [16]. For a data point i , the silhouette score is defined as

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}.$$

Here, $a(i)$ is the mean within-cluster distance, while $b(i)$ is the smallest mean distance to another cluster. Scores range from -1 to 1 , with higher values indicating better cluster cohesion and separation. The overall score is the mean across all points.

Gaussian mixture models (GMMs) provide a probabilistic clustering approach [5]. Rather than assigning each data point exclusively to one cluster, GMMs estimate its probability of belonging to each cluster. They model the data as a mixture of Gaussian distributions with different variances and covariances. Posterior cluster probabilities and model-based log-densities can be used to identify uncertain assignments and points that lie in low-density regions of the fitted mixture.

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [17] is a density-based clustering algorithm. It constructs a hierarchy of clusters across different density levels, implemented as a tree. In contrast to k-means and GMMs, HDBSCAN does not require a predefined number of clusters and can label points in sparse regions and other outliers as noise. It also makes fewer assumptions about cluster shape and can therefore identify more complex nonlinear structures.

The dataset is often clustered in high-dimensional representations. Since such structures are difficult to inspect directly, low-dimensional visualization methods are often used alongside clustering. Uniform Manifold Approximation and Projection (UMAP) is a nonlinear dimensionality reduction method that projects high-dimensional data into a low-dimensional embedding while aiming to preserve local neighborhood structure [18]. In this thesis, UMAP is used for visualization purposes rather than as the space in which clustering is performed.

2.7 Token-Based and Transformer-Based Topology Encoding

Encoding physical detector configurations becomes more complicated when varying topologies are included. A UIFO configuration is more than a flat vector of optical parameters. The parameters belong to components, spatial positions on a grid, and connections between optical elements. This makes the representation of the configuration an important architectural choice.

Surrogate Predictions for Reiterative Optimization with Update Training (Sprout) [19] is a neural surrogate pipeline for accelerating interferometer evaluation and optimization. It operates on predefined UIFO layouts with fixed discrete design choices and predicts sensitivity and internal optical powers from continuous parameters.

Rather than processing these parameters as a flat vector, Sprout treats the UIFO grid as an attributed graph. Parameters associated with connected nodes and edges are grouped into overlapping patches that preserve spatial position and connection to neighboring optical components. In the original Sprout implementation, the patch layout is determined by the predefined detector topology rather than being learned across varying topologies.

After patch construction, Sprout adds positional information and applies a trainable Fourier feature map, consisting of a linear layer followed by a sine activation. Such feature expansions can help neural networks represent higher-frequency variations in the input-output relation. The embedded patches are then processed by a transformer encoder [20]. Through self-attention, the model can exchange information between patches based on learned relationships, which is useful because the effect of one optical component can depend on nonlocal interactions with other parts of the interferometer. The encoder ends with a multilayer perceptron (MLP).

Although Sprout was developed for neural surrogate modeling rather than latent-space analysis, it demonstrates how the continuous parameters of a fixed optical configuration can be encoded as interacting spatial patches instead of a flat vector. The dataset considered in this thesis additionally contains varying component choices and topologies. The multimodal architectures introduced in the Methods chapter therefore extend this principle by explicitly representing topology-dependent component, connection, and parameter information.

3 Dataset and Preprocessing

3.1 Overview

The dataset was generated continuously during the thesis, as new optimization runs were still being produced while the analysis was ongoing. This thesis analyzes a broadband subset of the optimization dataset. Unless stated otherwise, the term dataset refers to this analyzed subset. At the stage when the VAE analyses were implemented, the analyzed dataset consisted of 34,239 broadband runs, covering 16,889 different topologies. Only 800 runs were based on 4×4 UIFO grids, while all remaining runs used 3×3 grids.

Each run contains an optimized sensitivity curve, continuous physical parameters, topology information, and metadata such as the optimization algorithm, learning rate, the UIFO grid size, and the loss value. The stored continuous parameter values correspond to unbounded optimizer variables. For analyses that require physical interpretation, these variables were mapped to their allowed physical ranges (see Table 3.1) using sigmoid transformations. The continuous parameters are categorized into seven types according to their physical meaning: mirror and beam-splitter tuning, mirror and beam-splitter mass, mirror and beam-splitter reflectivity, squeezer angle, squeezing amplification, cavity length, and laser power. The different numerical ranges and units of these parameter types motivate the preprocessing steps described below.

Table 3.1: Physical parameter types and value ranges used to interpret optimized configurations.

Parameter Type	Value Range	Unit	Component
Laser power	[0, 200]	W	Laser
Squeezing amplification	[0, 10]	dB	Squeezer
Squeezer angle	[0, 360]	deg	Squeezer
Cavity length	[0.1, 4000]	m	Cavity
Reflectivity	[0, 1]	Dimensionless	Mirror, beam splitter
Tuning	[−360, 360]	deg	Mirror, beam splitter
Mass	[0.01, 200]	kg	Mirror, beam splitter

3.2 Categorization

The dataset can be divided into four groups, as shown in Figure 3.1. The first two groups comprise the base dataset. One group contains topologies that occur only once, and the other contains topologies that were optimized multiple times. The repeated-topology group may contain multiple runs of the same topology due to fixed random seeds or testing of specific physical properties on a common topology. In contrast, the unique-topology group contains a broader range of distinct configurations.

Two additional groups were generated from the base runs. The run that was already optimized in the base group is called the parent run. In the group of re-optimizations, Gaussian-distributed noise $\varepsilon \sim \mathcal{N}(0, 1)$ was added to the parameters of well-performing base configurations, with the topology kept fixed. The perturbed configurations were re-optimized and often achieved lower loss values than their parent runs. Within each topology, the runs differ mainly by small parameter perturbations and subsequent local re-optimization.

For the base groups, simplification candidates were generated by removing components whose parameters were close to neutral values. Examples include mirrors with reflectivity close to 0 or tuning close to 0° , lasers with power close to 0 W, and squeezers without squeezing amplification, corresponding to 0 dB. These simplified configurations were first simulated again without further optimization. Most of them performed worse than their parent configurations. However, configurations that showed a relevant improvement were optimized again, resulting in 424 runs in the simplification group. Among the 411 simplified 3×3 runs, the median number of active parameters was 79, compared with 192 for non-simplified 3×3 runs. Regardless of the UIFO grid size, the simplified runs achieved low loss values.

For the multimodal VAE analyses in Section 4.4, the re-optimization group was capped at five runs per topology because additional runs contributed many closely related configurations but little topological variation. In addition, 1,296 fractional runs were excluded because their varying mirror distances made them not directly comparable with the remaining runs.

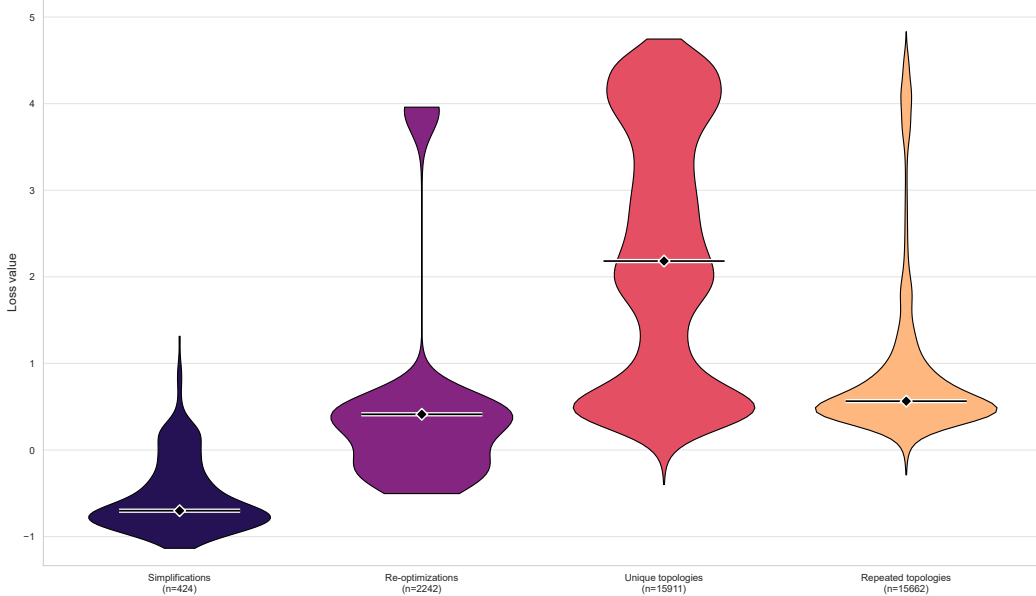


Figure 3.1: Loss distribution across the four dataset groups. The base dataset contains 15,911 unique topologies optimized once and 636 repeated topologies optimized multiple times, resulting in 15,662 repeated-topology runs. Two additional groups were generated from base runs: re-optimizations of well-performing configurations, where parameters were perturbed with noise while topology was kept fixed, and simplifications, where components close to neutral parameter values were removed before re-optimization. The re-optimization group contains 17 unique topologies, while the simplification group contains 235 unique topologies. Each violin shows the loss distribution within one group; black horizontal bars indicate the group median.

3.3 Scaling

Sensitivity curves were converted to $\log_{10}$ -space. We conducted this transformation because their values span several orders of magnitude, approximately from 10^{-26} to $10^{-22} \frac{1}{\sqrt{\text{Hz}}}$. This representation reduces the influence of absolute differences in strain sensitivity and emphasizes relative differences.

The configuration representation contains both continuous physical parameters and class-like topology entries. This section focuses on the transformation of continuous parameters. Topology-related features are described in Section 4.6.1. Scaling is required because the physical parameters have different numerical ranges and units. Without scaling, parameter types with large values could dominate the training objective. Before applying these transformations to the final models, parameter-specific scalings were tested against a simpler linear encoding and retained only for parameter types where they improved reconstruction performance.

The periodic parameters include the squeezer angle and the tuning of mirrors or beam splitters. For these parameters, values such as 0° and 360° represent the same physical configuration but lie at opposite ends of the numerical range. Before applying the trigonometric transformation, angular values were converted from degrees to radians. Each periodic parameter x was then represented by sine and cosine components

$$x_{\sin} = \frac{\sin(x) + 1}{2}$$

$$x_{\cos} = \frac{\cos(x) + 1}{2}.$$

This transformation preserves the periodic structure while mapping both components to the interval $[0, 1]$. Mirror and beam splitter masses are strongly skewed toward large values. Therefore, a logarithmic transformation was applied before the final scaling step, where $x_{\min}$ and $x_{\max}$ denote the physical bounds from Table 3.1

$$x_{\text{scaled}} = \frac{\log_{10}(x) - \log_{10}(x_{\min})}{\log_{10}(x_{\max}) - \log_{10}(x_{\min})}.$$

Finally, all remaining continuous parameters were normalized using min-max scaling

$$x_{\text{scaled}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}.$$

After these transformations, the continuous parameters are represented on comparable numerical scales. This is important for the multimodal VAEs, which reconstruct normalized configuration feature vectors.

4 Methods

4.1 Parameter-Level Analyses

The variance-correlation analysis tested whether detector performance could be explained using a smaller subset of informative parameters. A typical 3×3 configuration contains a median of 192 optimizable parameters, making direct interpretation difficult. The working hypothesis was that parameters that consistently converge to similar values across optimized runs may be important for detector performance.

The analysis evaluated 247 parameters across a preliminary dataset of 3,088 3×3 configurations. The stored optimizer variables were transformed to physical parameter values and then scaled independently for each parameter to the interval $[0, 1]$. For each scaled parameter, we computed the variance across all runs and across the best 10% of runs, defined by the lowest loss values. Spearman correlations between each scaled parameter and the loss were computed to test for a monotonic relationship between individual parameter values and detector performance.

The subsequent parameter-substitution analysis tested whether individual parameter values from high-performing runs could improve a median baseline of the same topology. For this purpose, we used a dataset of 7,258 runs comprising six topologies, each represented by more than 1,000 runs. For each topology, the median-loss run served as the baseline configuration. Across the six topologies, 33 runs with loss values below 0 served as reference configurations and provided the parameter values used for substitution.

We first replaced one parameter at a time in the median baseline while keeping the topology and remaining parameters unchanged. We then replaced all parameters of one of the seven parameter types simultaneously. After each substitution, the modified configuration was re-simulated using Differomator, and the loss change was evaluated. A negative delta loss indicated improvement relative to the baseline, whereas a positive delta loss indicated deterioration.

4.2 Sensitivity-Curve Clustering

Sensitivity-curve clustering examined whether recurring output-level structures could be identified from curves consisting of 50 logarithmically spaced frequency points between 20 Hz and 5 kHz. Sensitivity curves provide a unified representation of all runs, independent of UIFO grid size and the number of parameters or components. The working hypothesis was that characteristic curve shapes could indicate different underlying physical mechanisms. We performed the analysis in two phases. First, three clustering algorithms were compared. We then applied the most promising method to two larger datasets.

The algorithms were initially evaluated separately on six fixed topologies, each containing more than 1,000 runs and together comprising 7,258 runs. We applied PCA to the logarithmically

transformed sensitivity curves. For each topology, we retained the smallest number of principal components explaining at least 99% of the variance. This reduced the 50-dimensional curves to between 7 and 9 dimensions. We excluded runs with loss values above 1 to focus the comparison on at least moderately performing configurations.

We used k-means as a distance-based baseline and candidate values of k were compared using the silhouette score. Gaussian mixture models (GMMs) provided a probabilistic alternative. Because GMMs do not provide an intrinsic noise class, we defined runs as uncertain or noise-like when their maximum posterior probability was below 0.75 or their fitted log-density lay in the lowest quartile. The third method was Python’s implementation of Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [21]. Hyperparameter configurations were evaluated using Density-Based Cluster Validation (DBCV) and the resulting noise fraction.

To visualize the resulting cluster assignments in two dimensions, UMAP was applied to the PCA representation. UMAP was used only for visualization, while clustering was performed in PCA space. Cluster characteristics were also inspected using median sensitivity curves and interquartile ranges.

Based on the method comparison, HDBSCAN was applied to the combined set of 7,258 runs from six topologies. This tested whether recurring sensitivity structures could also be discovered across different topologies. It was then applied to the larger dataset containing 33,769 runs that did not yet include several simplification and re-optimization runs generated later. For this analysis, the sensitivity curves were reduced to seven dimensions, and a loss threshold of 0.25 retained 1,590 runs. This threshold was selected as a compromise between retaining a sufficient sample and focusing on well-performing configurations. The final HDBSCAN configuration used `cluster_selection_epsilon = 0.5`, `min_cluster_size = 10` and `min_samples = 7`.

4.3 VAE on Sensitivity Curves

To investigate low-dimensional structure in the sensitivity curves, we trained beta-variational autoencoders on the 50-point representation spanning 20 Hz to 5 kHz. Only sensitivity curves were provided as model inputs. Topology information, physical parameters, and loss values were used only for data selection and subsequent interpretation. Three analyses were conducted: a proof-of-concept comparison with the structure previously identified by HDBSCAN, a stability analysis of initialization-dependent variation within three fixed topologies, and a diversity analysis across distinct topologies.

The exploratory analysis used the same initial dataset as the preceding HDBSCAN analysis, comprising 33,769 runs, while the two following stability and diversity analyses had access to the complete dataset of 34,239 runs. Analysis-specific selection criteria were then applied, and each resulting dataset was divided into an 85% training split and a 15% validation split. Before training, each sensitivity value x was transformed by taking the $\log_{10}(x + 10^{-35})$. Each frequency dimension was then standardized to zero mean and unit variance using statistics fitted exclusively on the training split and subsequently applied to the validation set.

The encoder and decoder were implemented in PyTorch as approximately symmetric fully connected neural networks with two hidden layers each. Sigmoid Linear Unit (SiLU) activation functions were applied after each hidden layer, following their use in physics-informed neural

networks [22]. The final decoder remained linear and reconstructed the sensitivity curves. The encoder parameterized the diagonal Gaussian approximate posterior $q(z|x)$ through μ and $\log \sigma^2$. Latent samples were generated through the reparameterization trick in Equation 2, while posterior means provided deterministic latent coordinates.

Training was performed using the Adam optimizer with weight decay and gradient clipping according to the beta-VAE loss function in Equation 4, with the MSE in Equation 1 serving as the reconstruction term. Model selection was based on the validation loss after completion of the beta warm-up, and training was stopped early if the validation loss did not improve for 50 consecutive epochs. The analysis-specific selection criteria, model architectures, and training hyperparameters were selected through exploratory comparisons and are summarized in Table 4.1.

Table 4.1: Dataset sizes, model architectures, and training hyperparameters used for the three sensitivity-curve VAE analyses.

Configuration	Exploratory	Stability	Diversity
Selected runs	15,645	2,436	13,567
Loss criterion	≤ 0.7	< 3.0	< 4.0
Hidden layers	128, 64	64, 32	128, 64
Latent dimension	9	2	2
Learning rate	5×10^{-4}	5×10^{-3}	10^{-3}
Batch size	128	64	256
Maximum β	0.05	0.5	0.2
β warm-up	30 epochs	40 epochs	50 epochs
Visualization	UMAP	Direct 2D	Direct 2D

For the exploratory analysis, a threshold of 0.7 was selected as a compromise between retaining a sufficiently large sample and preserving topological diversity, resulting in 15,645 runs. The latent dimensionality was heuristically set to nine, matching the number of principal components required to explain at least 99% of the variance in a preceding PCA analysis.

The stability analysis tested whether runs from fixed topologies formed latent-space structure associated with topology or parameter initialization. The three most frequently represented topologies were selected, and two groups were constructed for each topology: runs initialized from identical parameters and runs initialized from different parameters. A total of 406 runs were sampled without replacement from each of the six groups, resulting in 2,436 runs.

The diversity analysis investigated sensitivity-curve structure across distinct topologies, while reducing the influence of repeatedly sampled designs. Re-optimization and simplification runs were excluded from training and only the lowest-loss run for each topology was retained, resulting in 13,567 runs. After training, the best runs from 235 excluded simplification topologies were encoded using the training-fitted scaler and frozen encoder. Their latent positions were compared with the main dataset, while the overall latent coordinates were evaluated through linear regressions with respect to loss, number of parameters, and maximum internal laser power. For the stability and diversity analyses, a two-dimensional latent space was chosen to permit direct visualization without an additional projection step.

4.4 Multimodal Architecture Family

The previous analyses considered only the sensitivity curves. Although these provide a topology-independent description of detector performance, they do not directly encode the physical configurations that produce them. The multimodal model family therefore jointly represents each optimized interferometer configuration and its corresponding sensitivity curve.

Each optimization run contains a preprocessed and normalized configuration comprising the topology and optimizable parameters, together with a sensitivity curve sampled at 50 frequency points. Since optimizable parameters depend on the topology, configurations differ in the number and type of active parameters. The vocabulary contains 271 parameter dimensions, of which a median of 192 are active per run; inactive dimensions are filled with zeros.

Following the preprocessing and normalization described in Section 3.3, 800 runs using a 4×4 UIFO grid, 1,296 fractional runs, and 2,164 re-optimization runs exceeding the limit of five runs per topology were excluded, leaving 29,979 runs. For an additional analysis designed to reduce the influence of overrepresented topologies, all dataset groups were restricted to the five lowest-loss runs per topology, yielding a reduced dataset of 16,355 runs.

The objective is to learn a compact latent representation linking the interferometer structure to its sensitivity. The three modalities are processed through two encoder branches: one jointly encodes topology and physical parameters, while the other encodes the sensitivity curve. The two branches are fused and compressed into a shared latent space for reconstruction. This bottleneck can then be analyzed for organization among detector configurations.

We implement the bottleneck either as an autoencoder (AE) or a variational autoencoder (VAE). The AE directly produces a deterministic latent vector, whereas the VAE parameterizes an approximate posterior and samples a latent vector using the reparameterization trick in Equation 2. All architecture variants share the sensitivity encoder, fusion module, latent dimensionality, and general decoder structure. They differ primarily in the encoding of the configuration and in the corresponding reconstruction objective.

All variants encode the 50-dimensional sensitivity curve as a 64-dimensional representation using a linear layer followed by a SiLU activation. The configuration encoder produces a 64-dimensional representation. Both embeddings are concatenated and mapped by a fusion MLP to a shared 64-dimensional representation, allowing the two encoder branches to interact.

In the VAE, the fused representation parameterizes a 32-dimensional mean vector μ and log-variance vector $\log \sigma^2$ of a diagonal Gaussian posterior. A 32-dimensional latent sample is then obtained through reparameterization. In the AE, the posterior parameterization is replaced by a direct mapping to a deterministic latent vector of the same dimensionality.

The shared decoder maps the latent vector first to 64 and then to 128 dimensions. Modality-specific output heads subsequently reconstruct the sensitivity curve and the selected configuration representation. Their exact structure differs between the patch-based and structured models.

The training objective combines configuration and sensitivity-curve reconstruction losses. Variational models additionally include KL regularization. The sensitivity reconstruction term is weighted by a factor of 100 to balance its contribution against the configuration loss.

4.5 Patch-Based Sprout-Inspired Multimodal VAE

The first two multimodal models use a patch-based configuration representation inspired by the structured encoding strategy in Sprout. They adapt the idea of spatially grouping optical parameters into structured neural tokens. The input is a flat vector containing the 271-dimensional vocabulary of optimizable parameters. To encode the geometrical structure, the patch-based encoder reorganizes the representation into physically motivated patches.

The fixed 3×3 UIFO grid defines 24 directed physical edges, which form the basis of the patch representation. Each edge-centered patch contains up to 34 parameter values, with shorter patches padded with zeros. Neighboring patches overlap by sharing information associated with their connecting node, thereby retaining local structural relationships.

A sinusoidal feature mapping projects each patch to a 256-dimensional embedding, helping mitigate the spectral bias of neural networks toward low-frequency functions [23]. Learned positional embeddings distinguish the physical edge locations, and a learned CLS token is added to summarize the configuration. The resulting sequence is processed by an eight-layer transformer encoder. The final 256-dimensional CLS-token embedding is used as the global configuration representation and compressed to 64 dimensions by a two-layer MLP.

The 64-dimensional configuration and sensitivity representations are concatenated and mapped by the common fusion module to a shared 64-dimensional representation. In the VAE, this representation parameterizes the mean and log-variance of a 32-dimensional diagonal Gaussian posterior, from which the latent vector is sampled using Equation 2. The shared decoder feeds two output heads. The sensitivity head reconstructs the 50-dimensional curve, while the parameter head reconstructs the complete 271-dimensional configuration vector. Parameter reconstruction uses binary cross entropy, whereas sensitivity reconstruction uses the mean squared error defined in Equation 1. The latter is weighted by a factor of 100

$$\begin{aligned}\mathcal{L}_{\text{VAE}} &= \mathcal{L}_{\text{param}} + 100 \cdot \mathcal{L}_{\text{sens}} + \beta \cdot D_{\text{KL}} \\ &= \text{BCE}(x_{\text{param}}, \hat{x}_{\text{param}}) + 100 \cdot \text{MSE}(x_{\text{sens}}, \hat{x}_{\text{sens}}) + \beta \cdot D_{\text{KL}}.\end{aligned}$$

Here, x_{param} and x_{sens} denote the parameter vector and sensitivity curve, while the hatted quantities denote their reconstructions. The complete architecture is shown in Figure 4.1.

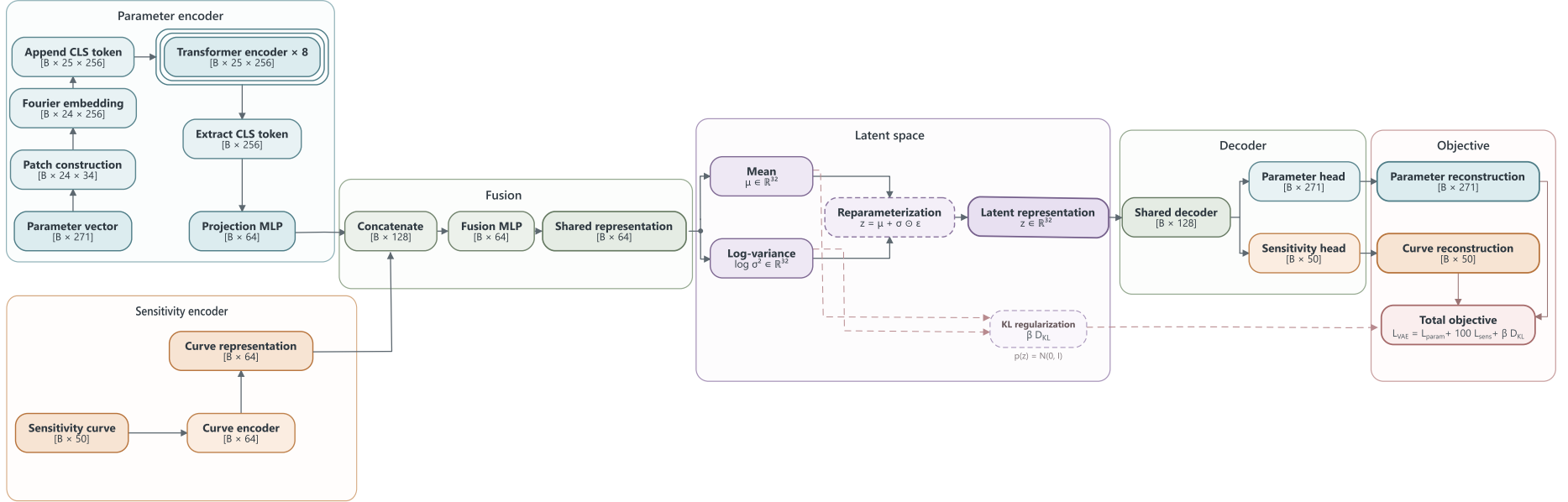


Figure 4.1: Architecture of the patch-based multimodal variational autoencoder. The optimized parameter vector is organized into 24 spatial patches and processed by a transformer encoder, while the sensitivity curve is encoded through a separate branch. Both modality-specific representations are fused and mapped to a shared 32-dimensional variational latent space. A shared decoder reconstructs the parameter vector and sensitivity curve through separate output heads. Dashed connections indicate Kullback–Leibler regularization.

4.6 General Patch-Based Model

The patch-based model used a 271-dimensional vector containing optimizable parameters. It therefore omitted discrete design choices, such as the presence and orientation of components without continuous parameters. To study the effect of configuration encoding, the general patch-based model (GPM) replaces this vector with three topology-aware representations: Flat, Grid, and Aliased. All are generated from the same source files and optimized parameter vectors.

4.6.1 Configuration Representations

The Flat representation concatenates all features into one vector. Discrete components are encoded by binary entries, while continuous parameters use separate feature entries. Its vocabulary contains 557 dimensions: 221 binary node features and 336 continuous parameter features.

The Grid representation assigns each component and parameter to a grid coordinate and local subposition. Component types and their properties are encoded separately, while lengths are represented as edge features. This makes spatial position explicit and distinguishes otherwise similar components at different locations. Its vocabulary contains 510 dimensions: 174 component-type indicators, 312 position-specific properties, and 24 edge lengths.

The Aliased representation retains the spatial structure and component classes of Grid but replaces property-specific names with shared continuous slots. The meaning of each slot depends on the associated component class; for example, mirror slots encode reflectivity, tuning, and mass, whereas laser slots encode power. This reduces the vocabulary to 498 dimensions: 174 class indicators, 300 shared slots, and 24 edge lengths.

4.6.2 Common Patch-Based Processing

For each representation, the GPM parses the vocabulary names and maps the features to 24 edge-centered patches. Features associated with one coordinate are assigned to neighboring patches, while features containing two coordinates are assigned to the corresponding edge. Patches are zero-padded to the maximum size required by each representation, producing tensors with batch dimension B :

- Flat: $B \times 24 \times 73$,
- Grid: $B \times 24 \times 61$,
- Aliased: $B \times 24 \times 61$.

After patch construction, all variants use the same Fourier mapping, positional embeddings, transformer encoder, sensitivity encoder, multimodal fusion module, and VAE bottleneck. Using the same architecture and random seed allows the influence of the configuration representation to be compared without changes in the model design or initialization.

However, patch construction still depends on feature-name parsing and representation-specific assignment rules. The patches also combine node and edge information and mix discrete topology features with continuous parameters, while the decoder treats these feature types uniformly. These limitations motivate the shared node, edge, and global token representation and feature-specific reconstruction heads of the final model.

4.7 Structured Topology-Aware VAE

The final model retains the common multimodal VAE backbone but replaces vocabulary-dependent patch construction with a shared UIFO-specific token representation. Each input representation is mapped to the same ordered sequence of node, edge, and global tokens. The main changes concern the configuration encoding, decoder heads, and reconstruction loss.

4.7.1 Shared Token Representation

The vocabulary of the selected Flat, Grid, or Aliased representation is parsed, and each input dimension is assigned to a token according to its physical role. The sequence contains 235 tokens: 210 node tokens representing 21 grid and boundary nodes with ten component slots each, 24 edge tokens, and one global token.

Node tokens represent local components and parameters, edge tokens describe connections and propagation, and the global token stores configuration-wide features. All vocabularies use the same token ordering, enabling direct comparison between representations. Because token lengths vary, shorter vectors are zero-padded, while a feature mask identifies valid entries.

4.7.2 Structured Encoder and Reconstruction Objective

The token features are projected to 256-dimensional embeddings. Learned token-type and positional embeddings encode each token’s role and location. A learned CLS token increases the sequence length to 236 before being processed by an eight-layer transformer encoder.

Self-attention exchanges information across node, edge, and global tokens. The final 256-dimensional CLS embedding summarizes the configuration and is projected to the 64-dimensional representation used by the multimodal fusion module.

The decoder maps its 128-dimensional representation to three output heads for discrete configuration features, continuous parameters, and the 50-dimensional sensitivity curve. Discrete features use binary cross entropy, whereas continuous values use mean squared error.

A configuration mask restricts reconstruction to valid dimensions. Discrete, continuous, and edge-related features are evaluated as mutually exclusive subsets, while the sensitivity head uses the weighted mean squared error of the multimodal model

$$\begin{aligned}\mathcal{L}_{\text{VAE}} &= \mathcal{L}_{\text{feat}} + 100 \cdot \mathcal{L}_{\text{sens}} + \beta \cdot D_{\text{KL}} \\ \mathcal{L}_{\text{feat}} &= \mathcal{L}_{\text{class}} + \mathcal{L}_{\text{value}} + \mathcal{L}_{\text{edge}} \\ &= \text{BCE}(x_{\text{class}}, \hat{x}_{\text{class}}) + \text{MSE}(x_{\text{value}}, \hat{x}_{\text{value}}) + \text{MSE}(x_{\text{edge}}, \hat{x}_{\text{edge}}), \\ \mathcal{L}_{\text{sens}} &= \text{MSE}(x_{\text{sens}}, \hat{x}_{\text{sens}}).\end{aligned}$$

Here, x_{class} , x_{value} , and x_{edge} denote the mutually exclusive discrete, continuous, and edge-related feature subsets, while hatted quantities denote their reconstructions. The complete architecture is shown in Figure 4.2.

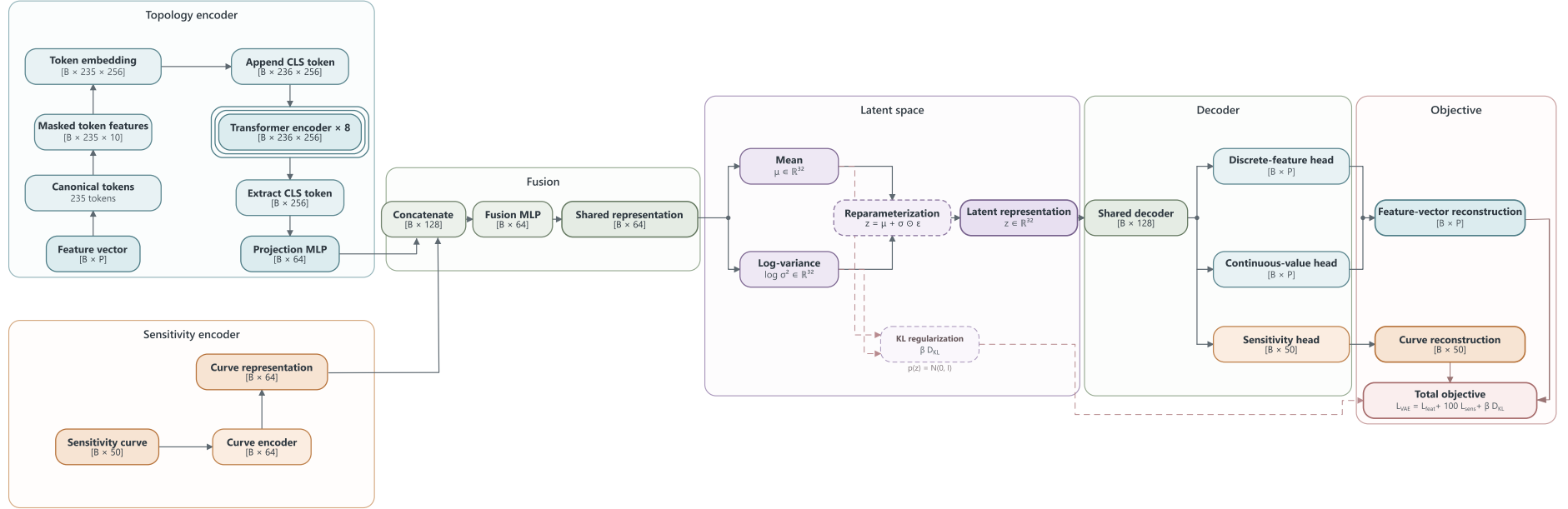


Figure 4.2: Architecture of the topology-aware multimodal variational autoencoder. The selected configuration representation is mapped to a shared sequence of node, edge, and global tokens before transformer-based processing. The resulting topology embedding is fused with a separately encoded sensitivity curve and mapped to a shared 32-dimensional variational latent space. The decoder reconstructs discrete configuration features, continuous parameter values, and the sensitivity curve through separate output heads. Dashed connections indicate Kullback–Leibler regularization.

5 Results

5.1 Skewed Parameter Distributions

All seven parameter types show non-uniform distributions, with several accumulating near physical boundaries rather than concentrating around a single interior value (see Figure 5.1). Tuning parameters and the squeezing angle are measured in degrees and are periodic, so values that differ by 360° can represent equivalent configurations. Reflectivity, laser power, and squeezing amplification show strong accumulation near 0 (see Figure 5.1), indicating that the corresponding optical elements or sources are often close to neutral settings.

This observation motivated the simplification procedure, in which components close to neutral values were removed and the resulting configurations were evaluated. Most simplification attempts on the 31,573 base runs worsened the loss. However, 424 simplified runs remained in the analyzed dataset because they improved after simplification. Through re-optimization they achieved low losses around -1 , corresponding to an average strain sensitivity approximately a factor of 10 lower than the Voyager reference (see Figure 5.1).

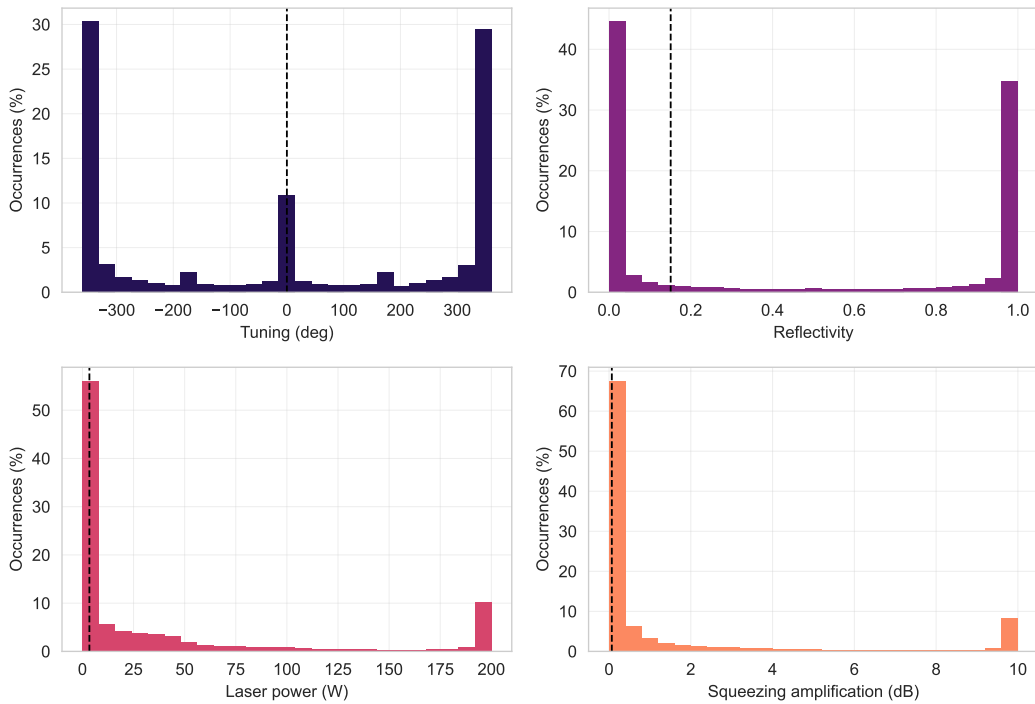


Figure 5.1: Distribution of selected physical parameter values across the analyzed dataset. Histograms show the percentage of parameter occurrences for each selected parameter type. Dashed black lines indicate the median. The selected parameter types illustrate strongly non-uniform distributions, accumulation near neutral values, and periodic structure.

5.2 No Evidence for Universal Important Parameters

All seven parameter types retained large variability across their allowed range. For example, cavity length ranged from 0.1 to 4000 m with a median standard deviation of approximately 1898 m. Comparing all runs with the best 10% of runs did not reveal a general reduction in parameter variance. Spearman correlations between individual scaled parameters and loss were weak, with the strongest absolute correlation below 0.11. Thus, no individual parameter showed a strong monotonic relationship with detector performance.

Substituting a single parameter value from a high-performing run into a median-performing run of the same topology worsened the loss in most cases. When a substitution improved the loss, the effect was small, with a maximum improvement of 0.011. Substituting all parameters of a parameter type either left the loss nearly unchanged or worsened it. Together, these results suggest that high-performing configurations cannot be explained by examining individual parameters or parameter types alone.

5.3 Sensitivity-Curve Clustering

K-means clustering separated the sensitivity curves according to their corresponding loss. A common pattern consisted of three clusters representing comparatively low-, intermediate-, and high-performing runs. GMMs produced an intermediate result, combining separation by overall sensitivity level with some differentiation of frequency-dependent curve shapes. Approximately 25% of the runs met our defined uncertainty or noise-like criterion. HDBSCAN produced the most distinct curve-shape groups, achieving a DCBV score of 0.651 with a noise fraction of 6.0% (see Figure 5.2). The eight identified clusters contained sensitivity curves with similar characteristics (see Figure 5.2). On closer inspection, six of these clusters corresponded to the six re-optimization topologies represented by at least ten runs.

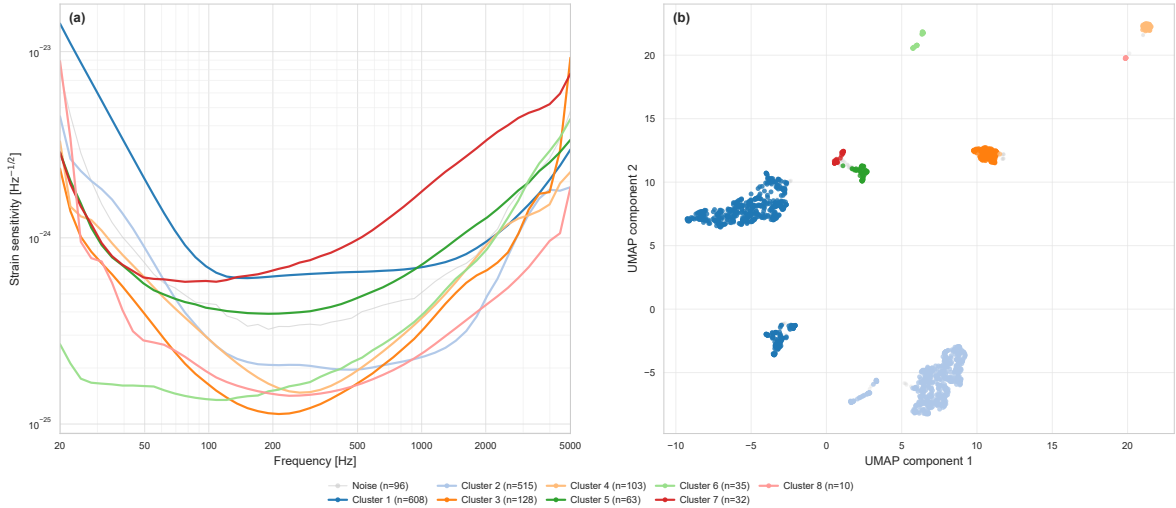


Figure 5.2: HDBSCAN clustering of the large sensitivity-curve dataset. (a) Median sensitivity curves of the identified clusters and the noise group. (b) Two-dimensional UMAP projection of the seven-dimensional PCA representation, colored by HDBSCAN cluster assignment. Clustering was performed in PCA space, while UMAP was used only for visualization. The selected configuration achieved a DBCV score of 0.651 and classified 6.0% of the runs as noise.

5.4 VAE Sensitivity Analysis

The exploratory VAE analysis reproduced the common sensitivity-curve structure of the six re-optimization topologies previously identified by HDBSCAN. However, this similarity did not extend to the non-re-optimized runs examined in the stability analysis. Their latent representations showed substantial overlap and no visually distinct grouping by either topology or parameter-initialization strategy, even when runs shared both the same topology and identical parameter values.

The diversity analysis produced a latent space that was organized by overall loss, as shown in Figure 5.3. A linear regression based on the two latent coordinates explained 93.58% of the variance in loss ($R^2 = 0.9358$). The number of parameters showed no linear association with the latent coordinates ($R^2 = 0.0001$), whereas the maximum internal laser power showed a stronger association ($R^2 = 0.7329$).

The 235 simplification topologies excluded from training were subsequently projected into the learned latent space. Most were mapped to a similar region well outside the main distribution of non-re-optimized runs. Together with their substantially lower loss values, this observation is consistent with the dominant organization of the latent space by detector performance.

Closer inspection showed that runs with the same topology, loss rounded to four decimal places, and similar sensitivity curves can arise from different parameter values; see Figure 5.3.

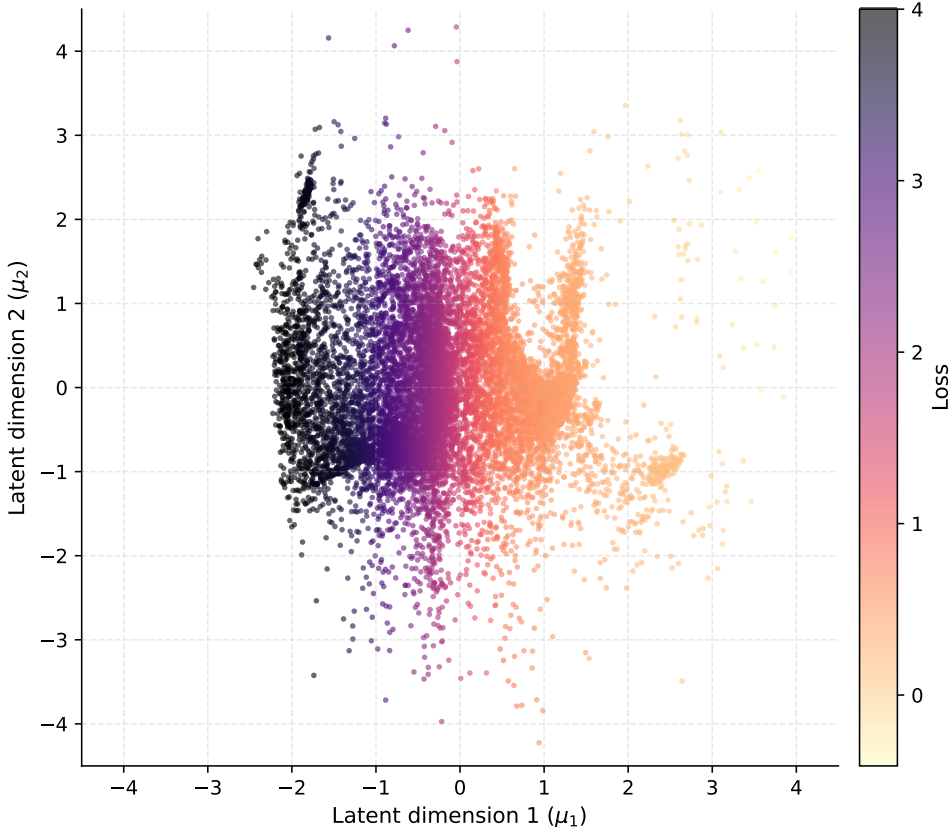


Figure 5.3: Posterior means of the diversity-analysis runs in the two-dimensional latent space, colored by sensitivity loss. For visualization, the axes were restricted to runs within a Euclidean distance of 4.5 from the origin, excluding 12 of 13,567 runs. The continuous color gradient indicates that the learned representation was strongly organized by detector performance.

5.5 Multimodal VAE Results

5.5.1 Optimizable-Parameter Models

For the first patch-based models, the sensitivity reconstruction loss was approximately 340 times smaller than the parameter reconstruction loss, motivating a weight of 100.

The predicted posterior log-variance showed little variation and was concentrated near zero. Since $\log \sigma^2 \approx 0$ implies $\sigma^2 \approx 1$, the posterior variance remained close to that of the standard normal prior and showed little dependence on the input. Together with the low variance of repeated stochastic reconstructions, this suggested that the model made limited use of the stochastic latent variables and was consistent with partial posterior collapse.

To examine the trade-off between latent-space regularization and reconstruction quality, we evaluated several values of β . Increasing β strengthened prior matching but also increased the validation reconstruction loss. The resulting Pareto curve showed a knee near $\beta = 0.005$, whereas $\beta = 0.001$ preserved better reconstruction quality at the cost of weaker regularization.

The VAE–AE comparison revealed a trade-off between reconstruction quality and latent-space regularization. The AE achieved the lower validation loss of 0.3732, mainly due to improved parameter reconstruction. Its latent representation appeared more coherent and showed gradients with respect to physical loss. The VAE reached a higher validation loss of 0.4916 and produced a more fragmented latent representation that did not seem to follow a loss gradient.

5.5.2 Complete Configuration Models

For the GPM, the Aliased representation achieved the lowest validation loss of 0.025307, whereas Flat and Grid exceeded 0.17. However, this value is not comparable to the losses of the final structured model because the two models use different reconstruction objectives. Their latent space visualizations also differed. Flat and Grid formed one large central cluster surrounded by seven to nine smaller groups. The central Aliased cluster separated into two main clusters.

These representation-dependent differences were substantially reduced by the final structured topology-aware model. The validation losses were similar across Aliased (0.1059; see Figure 5.4), Flat (0.1066, see Figure 5.7), and Grid (0.1111, see Figure 5.8). Their latent spaces likewise showed a common structure: one large central cluster with performance gradients, surrounded by several smaller clusters that also separated better- and worse-performing configurations.

A closer analysis linked the peripheral clusters to two groups of configurations. The low-loss peripheral cluster corresponded to simplification runs, as visible in Figure 5.4. Several peripheral groups aligned with the most frequent topologies in the dataset. Five showed a correspondence with individual cluster cores, whereas the sixth showed only partial alignment. Each topology was represented by at least 1,000 runs, as shown in Figure 5.5.

The topology-capped analysis retained the same broad latent-space structure across representations, with one dominant central cluster and a peripheral cluster of simplification runs. Validation losses were 0.1126 for Aliased (see Figure 5.9), 0.1310 for Flat, and 0.1902 for Grid.

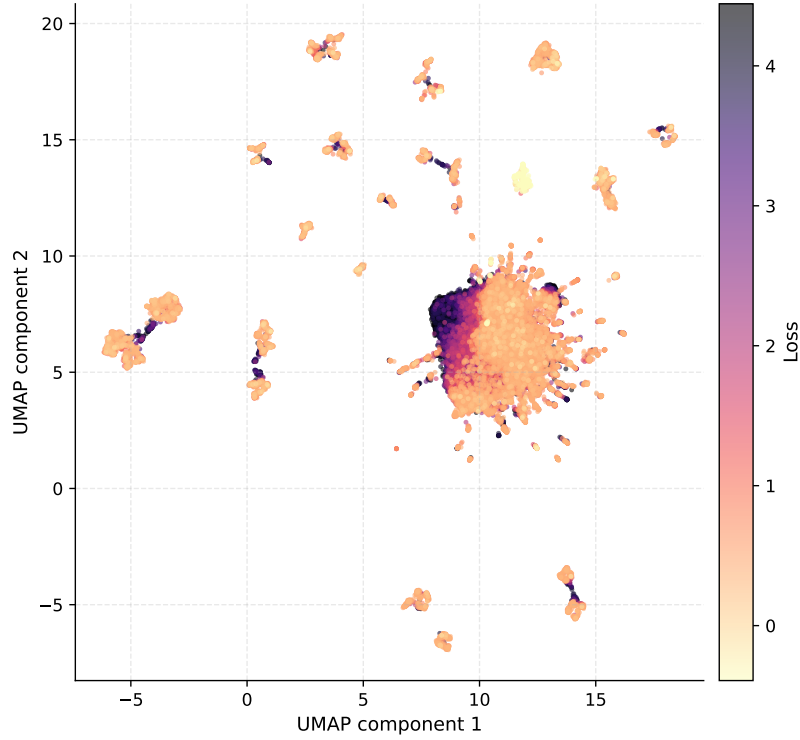


Figure 5.4: UMAP projection of the 32-dimensional latent space learned by the final structured topology-aware multimodal VAE using the Aliased representation. Points are colored by the corresponding physical loss, with lower values indicating better-performing configurations. The embedding forms one dominant cluster and several smaller isolated groups, while local color gradients suggest that the latent representation captures variations in physical performance. The model achieved a validation loss of 0.1059.

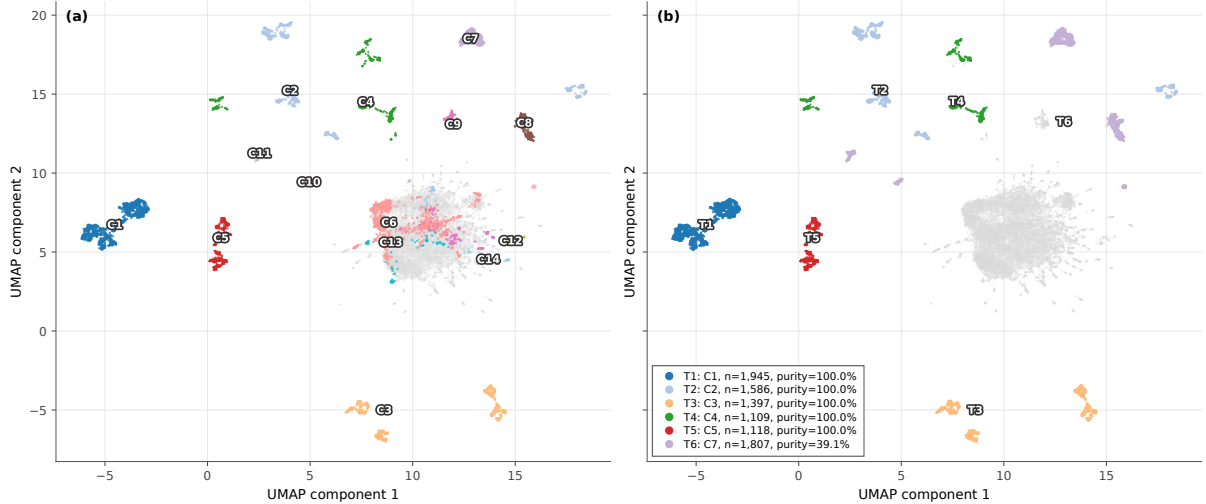


Figure 5.5: (a) HDBSCAN assignments projected into the two-dimensional UMAP space; gray points denote configurations classified as noise. (b) The same latent-space projection, highlighting the six most frequent topologies among non-noise configurations. The spatial overlap between several highlighted topology families and HDBSCAN cluster cores suggests that part of the cluster structure is associated with discrete configuration topology.

6 Discussion and Conclusion

The strongest general pattern was that the best-performing runs were simplified configurations that had subsequently been re-optimized. This suggests that many UIFO designs may be over-parameterized, containing components that are not necessary for strong performance. Reducing the number of parameters may simplify the optimization landscape, making the parameter updates easier to identify. The analysis did not reveal a universal design rule that characterizes high-performing configurations.

One reason is the size and complexity of the search space. A 3×3 UIFO grid can be represented using a vocabulary of 271 optimizable parameter dimensions in addition to several hundred discrete topology features, depending on the chosen representation. These variables interact nonlinearly and the effect of one parameter depends strongly on the remaining configuration. This helps explain why neither individual parameters nor parameter types consistently determined performance. The optimized configurations instead appear to form tightly coordinated systems in which parameters are adapted to one another. Replacing a small group of parameters with those from a better-performing run of the same topology can move the configuration away from its local optimum and worsen the loss.

The dataset also shows degeneracy. Runs with the same topology, similar sensitivity curves, and the same rounded loss value can still be produced by different parameter settings. Sensitivity curves alone therefore provide an incomplete description of the physical configuration. This limits the interpretation of clusters derived only from curve shape and motivated the multimodal models developed in the second part of the thesis, beginning in Section 4.4.

These models jointly encode the sensitivity curve, optimizable parameters, and discrete topology information. Their purpose is to learn a compact representation that preserves relationships between physical structure and detector performance. A central challenge is how to encode heterogeneous configuration features and allow them to interact before compression into a much smaller latent space.

The comparison of Flat, Grid, and Aliased representations showed that the encoding influences reconstruction quality, although this effect became smaller in the final structured model. The Aliased representation generally achieved the lowest validation loss, but no representation consistently revealed a different organization of the latent space. This suggests that the internal model structure and reconstruction objective may be at least as important as the original input vocabulary.

The main methodological contribution of this thesis is not the application of a VAE alone, but the adaptation of multimodal representation learning to optimized UIFO configurations. The final model combines sensitivity information with a structured representation of nodes, edges, global properties, and continuous parameters. This provides a more explicit basis for comparing different configuration encodings and for studying how topology and performance are organized in a shared latent space.

The same modeling flexibility is also a major limitation. The learned representation depends on architectural choices, loss weighting, latent dimensionality, KL regularization, token construc-

tion, and many additional hyperparameters. It was not possible to evaluate every combination empirically, so the reported results inevitably reflect selected design assumptions.

The UMAP projections add another limitation because they preserve local neighborhoods more reliably than global distances. Consequently, latent clusters should be interpreted as exploratory patterns rather than direct evidence of distinct physical mechanisms.

Further limitations arise from the data itself. The analysis focuses primarily on optimized broadband configurations rather than the full optimization trajectories. The dataset also changed during the thesis, and some groups contained many closely related re-optimizations. Although these effects were reduced through filtering and the topology-capped analysis, they may still influence the learned latent structure.

Future work could therefore pursue two complementary directions. First, new ways of simplifying UIFOs could be developed, as these re-optimized configurations seem to be a promising approach for finding well-performing UIFO designs. Second, alternative structured encoders, such as graph neural networks, could represent components and optical connections more directly than the current transformer tokenization. Including optimization trajectories rather than only final configurations may further reveal which structural changes consistently lead to improved designs.

Bibliography

- [1] B. P. Abbott *et al.*, “Observation of Gravitational Waves from a Binary Black Hole Merger,” *Physical Review Letters*, vol. 116, no. 6, Art. no. 061102, 2016, doi: 10.1103/PhysRevLett.116.061102.
- [2] M. Krenn, Y. Drori, and R. X. Adhikari, “Digital Discovery of Interferometric Gravitational Wave Detectors,” *Physical Review X*, vol. 15, Art. no. 021012, 2025, doi: 10.1103/PhysRevX.15.021012.
- [3] J. Klimesch, “Differometer: A Differentiable Interferometer Simulator for the Computational Design of Gravitational Wave Detectors,” Master’s thesis, Max Planck Institute for the Science of Light, 2025. [Online]. Available: <https://github.com/mcren/differometer>
- [4] R. X. Adhikari *et al.*, “A Cryogenic Silicon Interferometer for Gravitational-Wave Detection,” *Classical and Quantum Gravity*, vol. 37, no. 16, p. 165003, 2020, doi: 10.1088/1361-6382/ab9143.
- [5] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.
- [6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016. [Online]. Available: <http://www.deeplearningbook.org/>
- [7] G. E. Hinton and R. R. Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006, doi: 10.1126/science.1127647.
- [8] D. P. Kingma and M. Welling, “An introduction to variational autoencoders,” *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.
- [9] D. Flam-Shepherd, T. C. Wu, X. Gu, A. Cervera-Lierta, M. Krenn, and A. Aspuru-Guzik, “Learning Interpretable Representations of Entanglement in Quantum Optics Experiments Using Deep Generative Models,” *Nature Machine Intelligence*, vol. 4, pp. 544–554, 2022, doi: 10.1038/s42256-022-00493-5.
- [10] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, “Stochastic variational inference,” *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.
- [11] D. P. Kingma and M. Welling, “Auto-Encoding Variational Bayes,” in *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014. [Online]. Available: <https://arxiv.org/abs/1312.6114>
- [12] J. W. Paisley, D. M. Blei, and M. I. Jordan, “Variational Bayesian inference with Stochastic Search,” in *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, Omnipress, 2012, pp. 1367–1374.
- [13] I. Higgins *et al.*, “beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework,” in *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France: OpenReview.net, 2017. [Online]. Available: <https://openreview.net/forum?id=Sy2fzU9gl>

- [14] J. MacQueen, “Some Methods for Classification and Analysis of Multivariate Observations,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, 1967, pp. 281–297.
- [15] S. P. Lloyd, “Least Squares Quantization in PCM,” *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982, doi: 10.1109/TIT.1982.1056489.
- [16] P. J. Rousseeuw, “Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis,” *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987, doi: 10.1016/0377-0427(87)90125-7.
- [17] R. J. G. B. Campello, D. Moulavi, A. Zimek, and J. Sander, “Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 10, no. 1, pp. 1–51, 2015, doi: 10.1145/2733381.
- [18] L. McInnes, J. Healy, and J. Melville, “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction,” *arXiv preprint arXiv:1802.03426*, 2018, doi: 10.48550/arXiv.1802.03426.
- [19] C. Ruiz-Gonzalez *et al.*, “Neural Surrogates for Designing Gravitational Wave Detectors,” *arXiv preprint arXiv:2511.19364*, 2025.
- [20] A. Vaswani *et al.*, “Attention Is All You Need,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, 2017. [Online]. Available: <https://arxiv.org/abs/1706.03762>
- [21] L. McInnes, J. Healy, and S. Astels, “hdbscan: Hierarchical density based clustering,” *The Journal of Open Source Software*, vol. 2, no. 11, p. 205, 2017, doi: 10.21105/joss.00205.
- [22] D. V. Dung, N. D. Song, P. S. Palar, and L. R. Zuhail, “On The Choice of Activation Functions in Physics-Informed Neural Network for Solving Incompressible Fluid Flows,” in *AIAA SCITECH 2023 Forum*, Jan. 2023. doi: 10.2514/6.2023-1803.
- [23] N. Rahaman *et al.*, “On the Spectral Bias of Neural Networks,” in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019. doi: 10.48550/arXiv.1806.08734.

A Additional Results

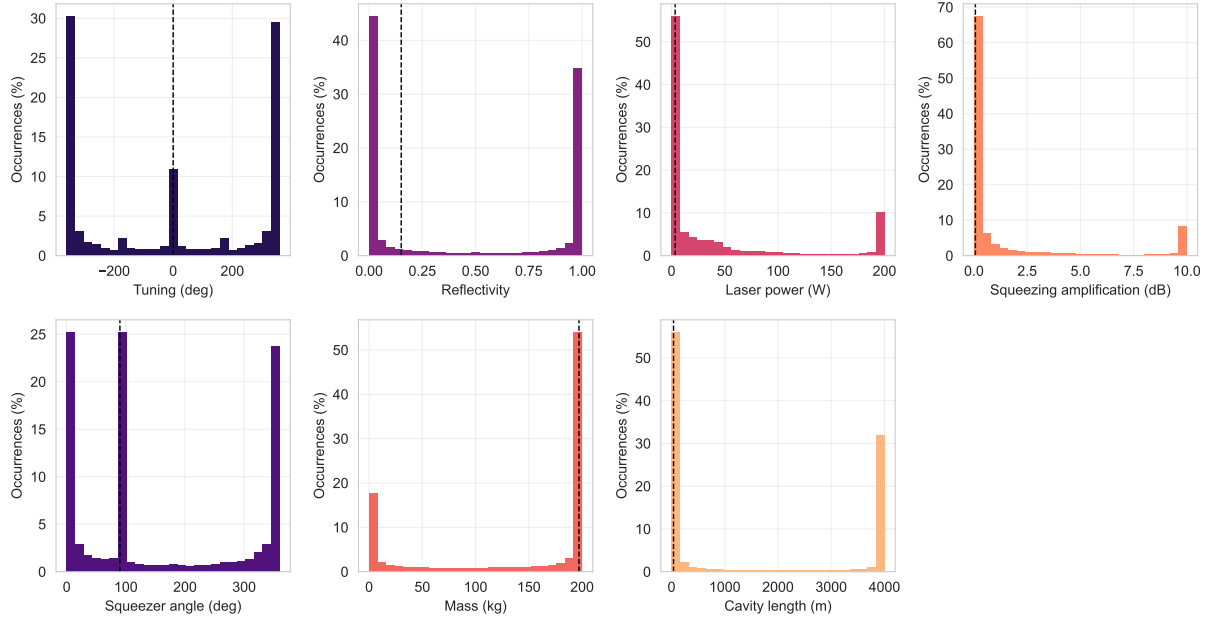


Figure 7.1: Distribution of physical parameter values across all seven parameter types in the analyzed dataset. Histograms show the percentage of parameter occurrences for each parameter type, and dashed black lines indicate the median. The distributions are strongly non-uniform and show parameter-type-specific structure, including boundary accumulation, near-neutral values, and periodic behavior for angle-like parameters.

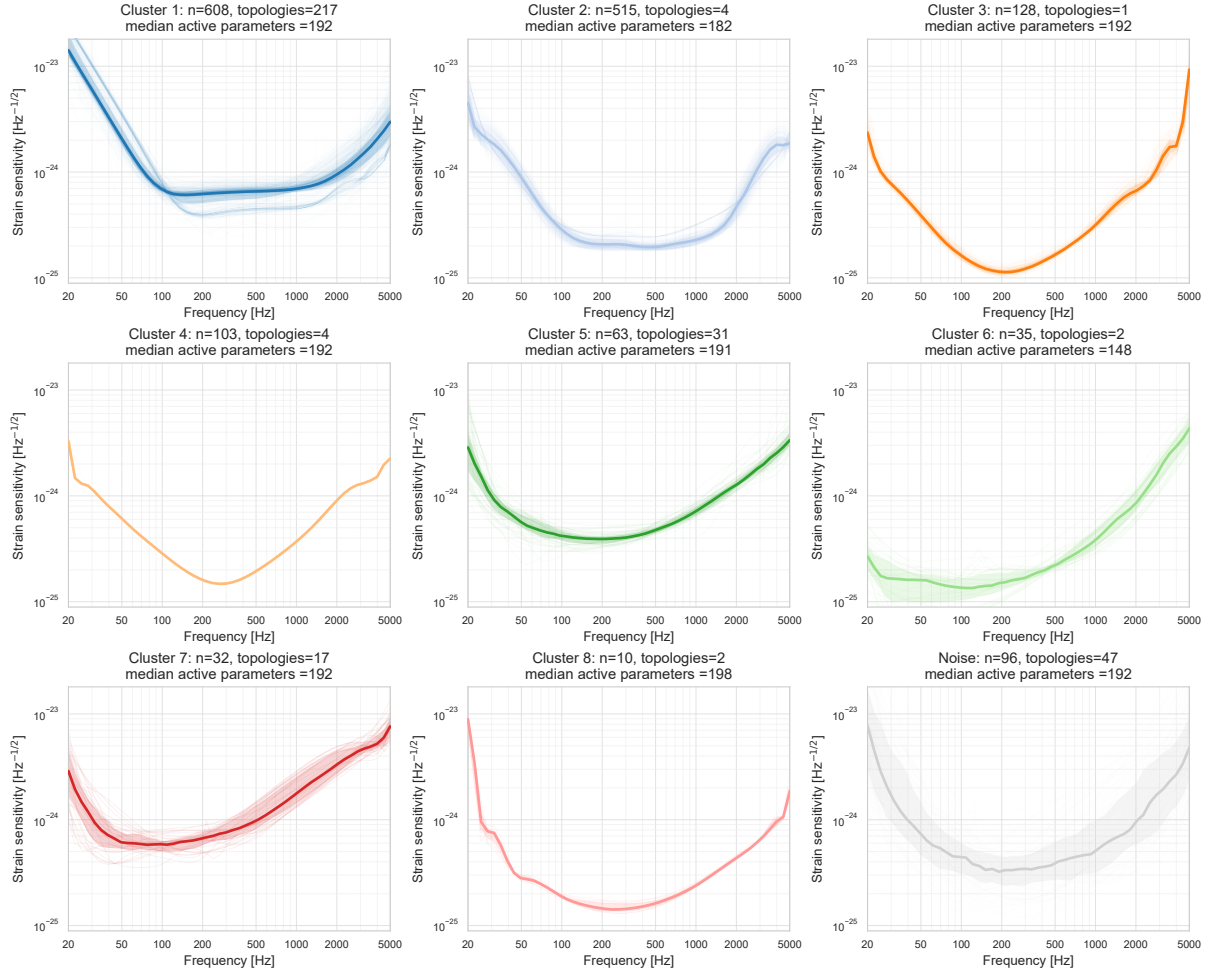


Figure 7.2: Sensitivity curves for each HDBSCAN cluster and the noise group in the final large-dataset analysis. Thin, semi-transparent lines show the individual runs, while the thick line indicates the pointwise median sensitivity curve of each group. Each panel reports the number of runs, the number of represented topologies, and the median number of active parameters. The clusters exhibit comparatively coherent frequency-dependent sensitivity profiles, whereas the noise group contains more heterogeneous curve shapes.

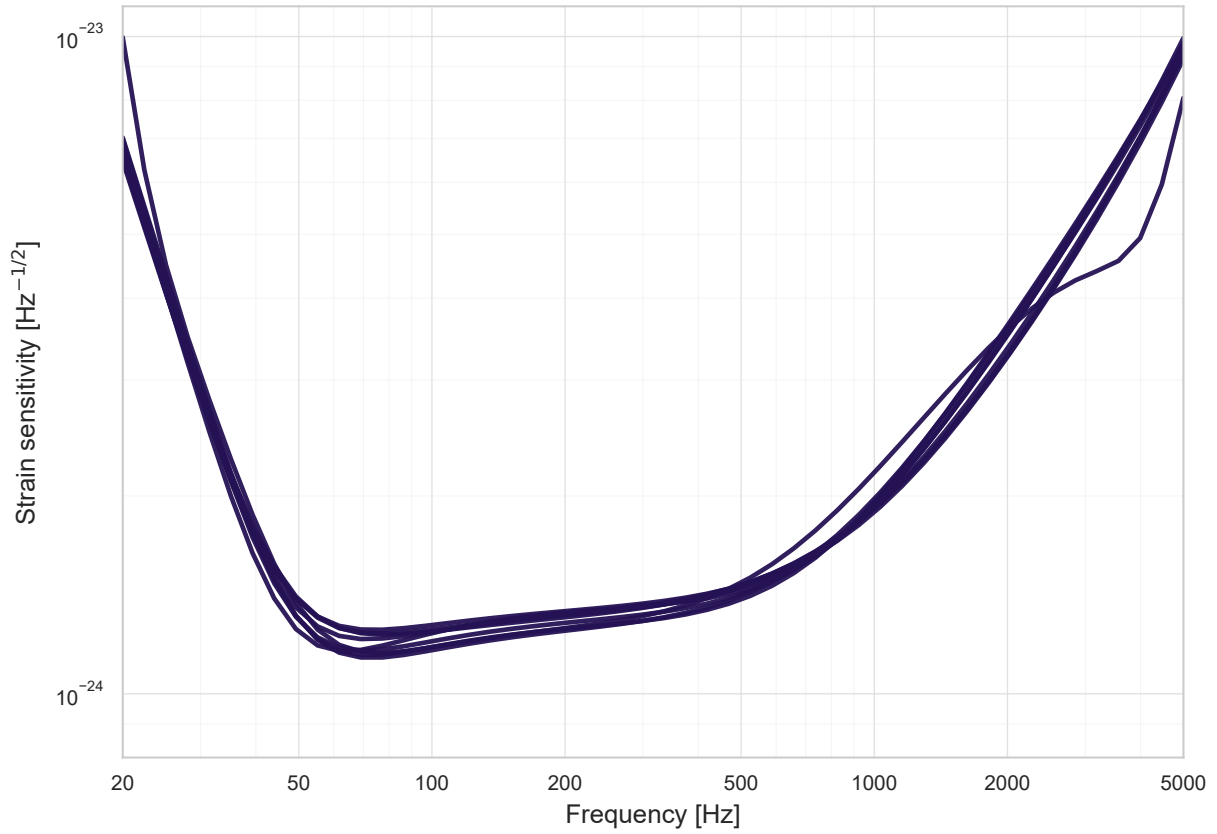


Figure 7.3: Sensitivity curves of nine runs with the same discrete topology and the same set of 194 optimized variables. Despite converging to different parameter vectors, the runs produced similar sensitivity curves and the same rounded loss of 0.4241. Hash-run id pairs for the nine runs are: 02828d4c-19, c6498f13-16, 3f80bbee-0, 093ba988-15, 7d36b4bf-10, 82afcf7c-3, cd142a87-12, 122e15a9-9, and e085b455-7. This illustrates that different parameter settings can yield nearly identical detector performance within the same topology.

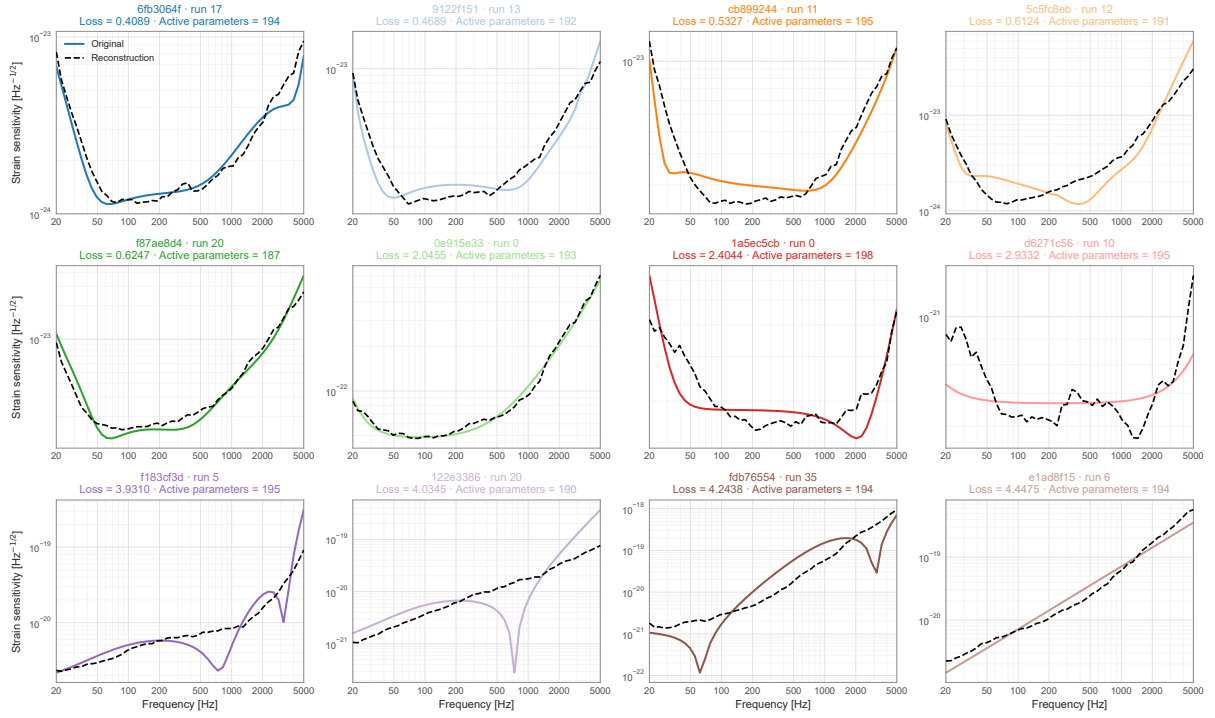


Figure 7.4: Original and reconstructed strain-sensitivity curves for twelve randomly selected samples from the validation split of the VAE trained with the Aliased representation. Each sample corresponds to a distinct topology. Original curves are shown in colors, while deterministic reconstructions obtained from the posterior mean $z = \mu$ are shown as black dashed lines. Samples are arranged row-wise in ascending order of their original optimization loss. Each panel reports the configuration hash, run ID, loss, and complexity and uses an independently scaled vertical axis.

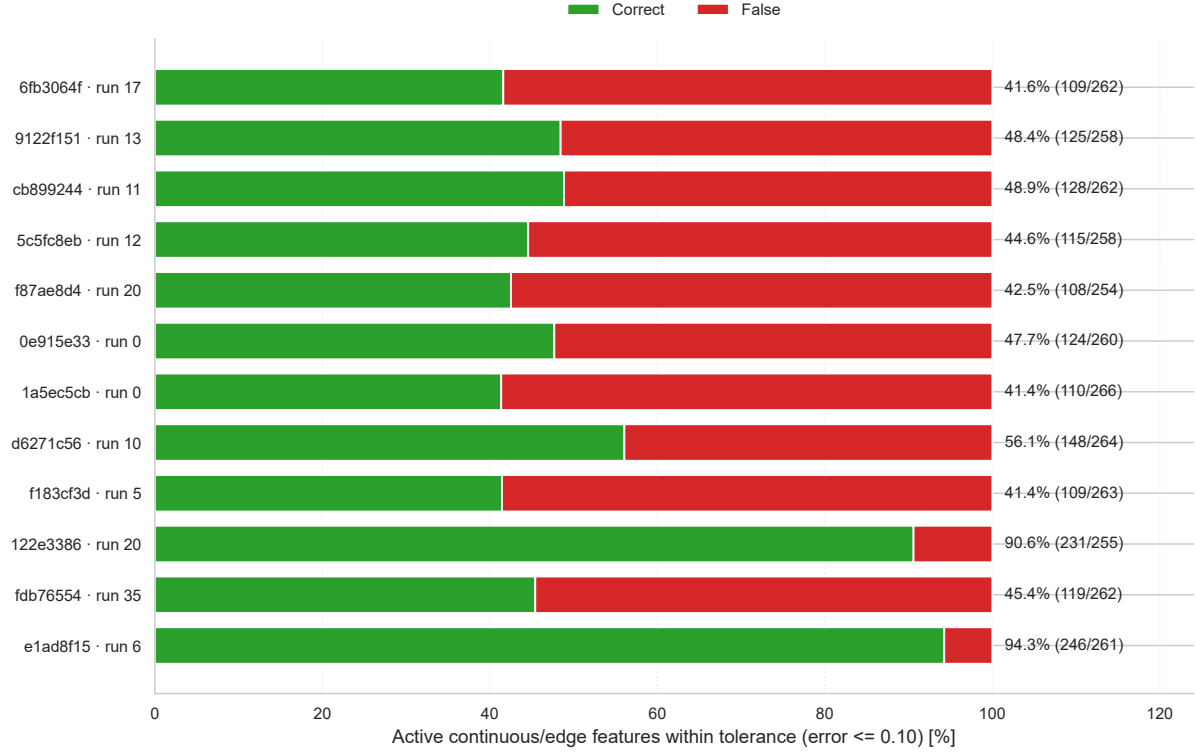


Figure 7.5: Continuous-parameter reconstruction for the same twelve validation samples. The bars show the percentage of active continuous shared-slot and edge features reconstructed within the common normalized tolerance $\tau = 0.10$. Unused shared slots are excluded according to the ground-truth component class, causing the number of evaluated features to vary slightly between configurations. Green and red indicate correctly and incorrectly reconstructed features, respectively. Labels report both the correct percentage and the corresponding number of correctly reconstructed features.

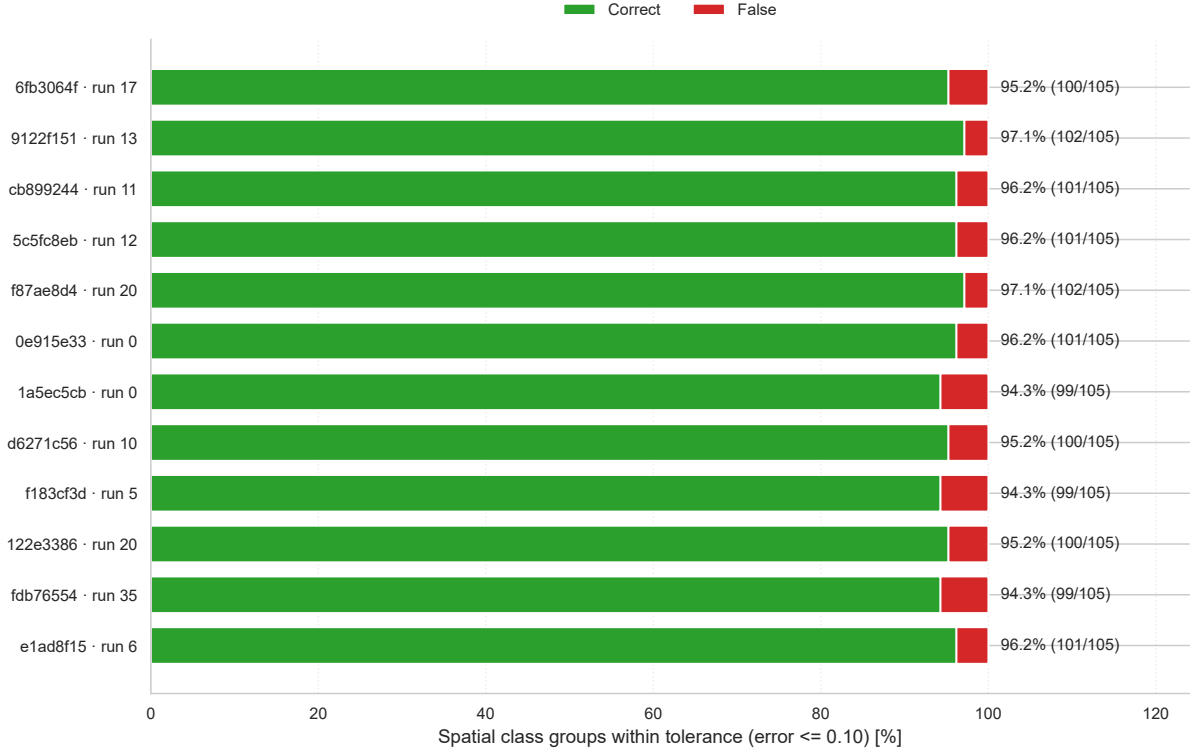


Figure 7.6: Discrete-topology reconstruction for twelve validation samples with distinct exact topologies. The bars show the percentage of spatial position-subposition class groups reconstructed within the common normalized tolerance $\tau = 0.10$. For an occupied group, the error accounts for both missing probability assigned to the ground-truth class and probability incorrectly assigned to competing classes. For an empty group, the error is the largest reconstructed component probability. All 105 class groups, including empty groups, are evaluated. Green and red indicate correctly and incorrectly reconstructed groups, respectively. Labels report both the correct percentage and the corresponding number of correctly reconstructed groups.

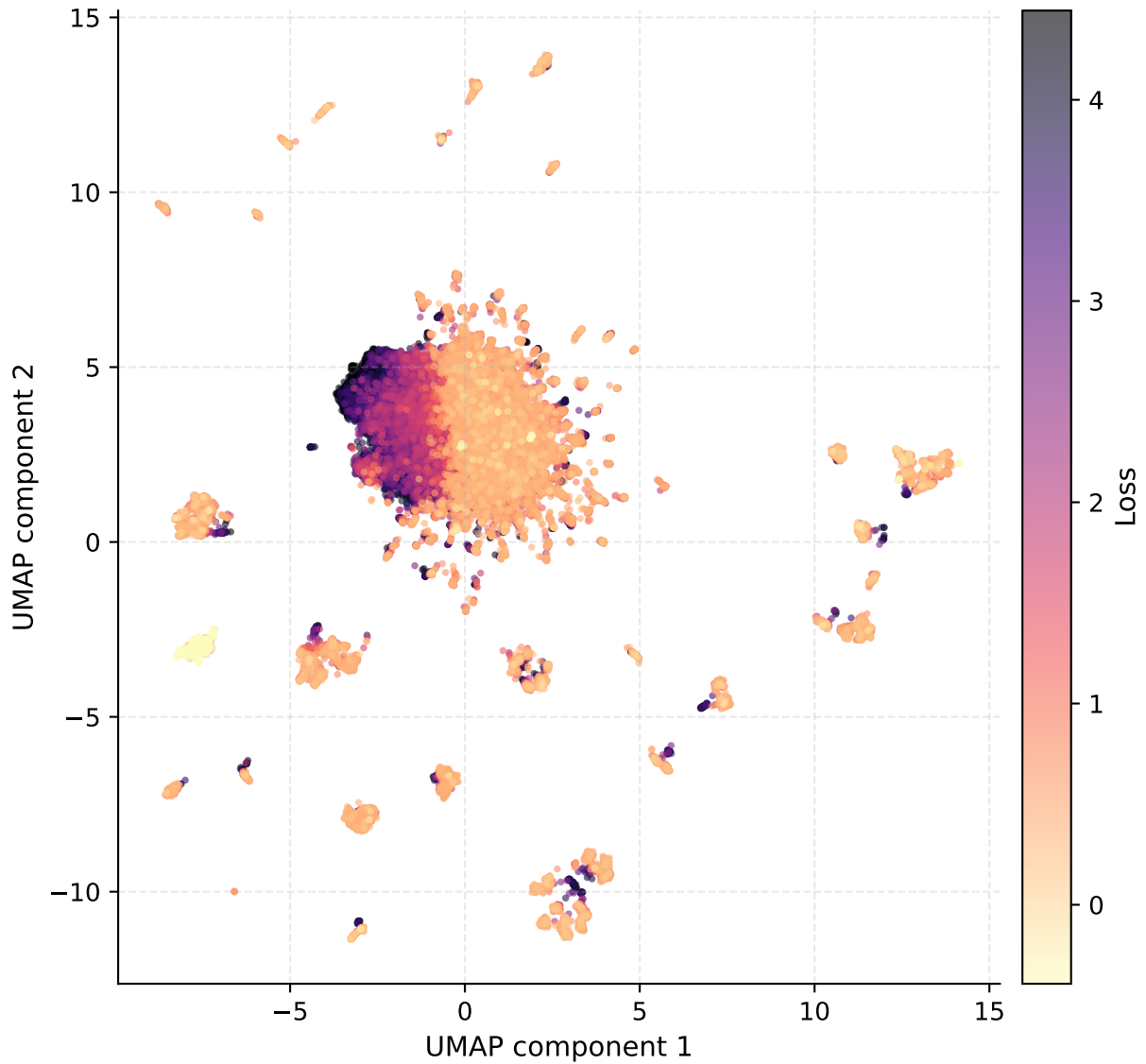


Figure 7.7: UMAP projection of the 32-dimensional latent space learned by the final structured topology-aware multimodal VAE using the Flat representation. Points are colored by the corresponding physical loss, with lower values indicating better-performing configurations. The embedding forms one dominant cluster and several smaller isolated groups, while local color gradients suggest that the latent representation captures variations in physical performance. The model achieved a validation loss of 0.1066.

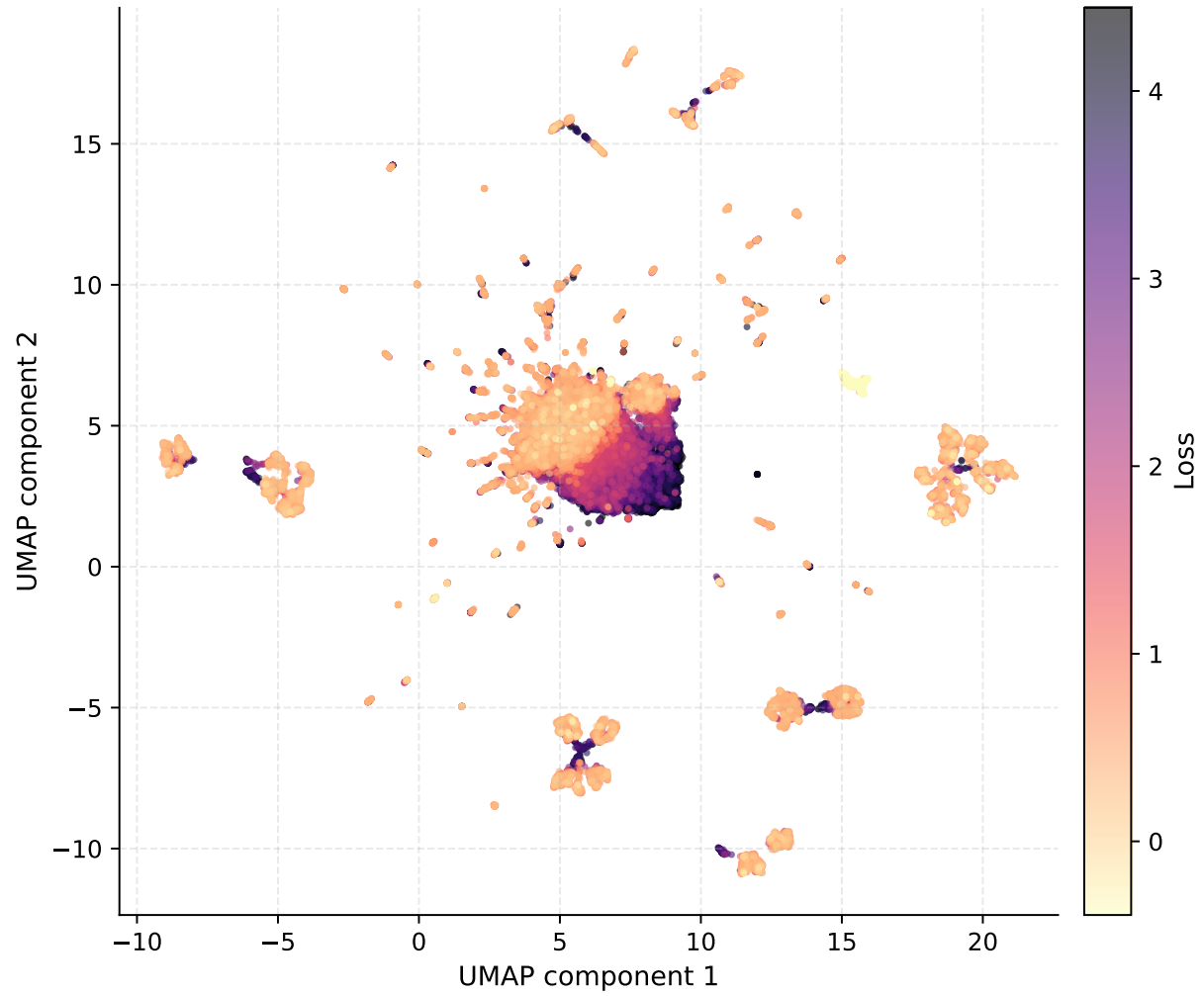


Figure 7.8: UMAP projection of the 32-dimensional latent space learned by the final structured topology-aware multimodal VAE using the Grid representation. Points are colored by the corresponding physical loss, with lower values indicating better-performing configurations. The embedding forms one dominant cluster and several smaller isolated groups, while local color gradients suggest that the latent representation captures variations in physical performance. The model achieved a validation loss of 0.1111.

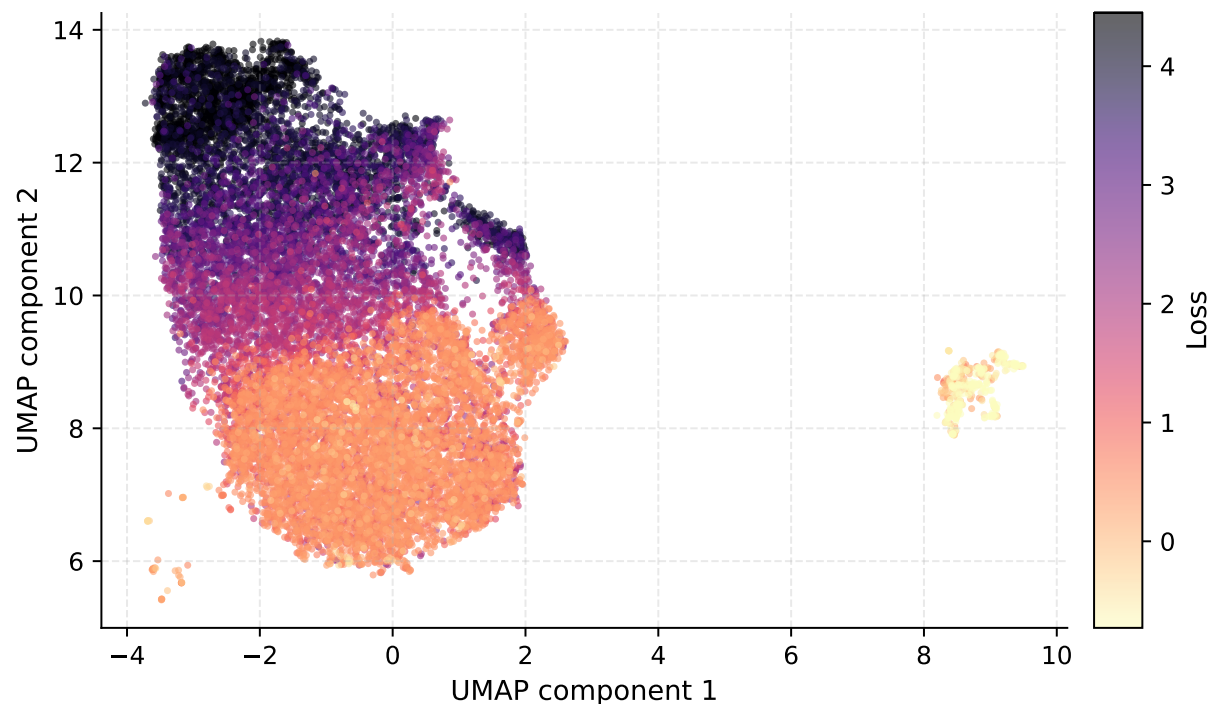


Figure 7.9: UMAP projection of the 32-dimensional latent space learned by the final structured topology-aware multimodal VAE using the Aliased representation after retaining at most the five lowest-loss runs per topology. The filtered dataset contains 16,355 runs and retains the same broad structure as the full analysis, with one dominant central cluster and a separate peripheral cluster associated with simplification runs. Points are colored by physical loss, and the model achieved a validation loss of 0.1126.

Table 7.1: Generative-AI software and model families used during the preparation of this thesis.

Function	Software	Models / versions
Language revision, grammar correction, programming support, code generation for data analysis	ChatGPT	GPT-5.5 Instant, GPT-5.5 Thinking; GPT-5.6 Sol, GPT-5.6 Sol Pro
Language revision, grammar correction, programming support, code generation for data analysis	Google Gemini	Gemini 3 Pro, Gemini 3 Flash, Gemini 3.1 Pro, Gemini 3.5 Flash
Language revision, grammar correction, programming support, code generation for data analysis	OpenAI Codex	GPT-5.4, GPT-5.6 Sol, GPT-5.6 Terra, GPT-5.6 Luna

Erklärung

Laut Beschlüssen der Prüfungsausschüsse Bioinformatik, Informatik, Informatik Lehramt, Kognitionswissenschaft, Machine Learning, Medieninformatik und Medizininformatik der Universität Tübingen vom 05.02.2025 sowie vom 08.07.2026. Gültig für Abschlussarbeiten (B.Sc./M.Sc./B.Ed./M.Ed.) in den zugehörigen Fächern. Bei Studienarbeiten und Hausarbeiten bitte nach Maßgabe des/der jeweiligen Prüfers/Prüferin.

1. Allgemeine Erklärungen

Hiermit erkläre ich:

- Ich habe die vorgelegte Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt.
- Ich habe alle wörtlich oder sinngemäß aus anderen Werken übernommenen Aussagen als solche gekennzeichnet.
- Die Arbeit war weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens.
- Falls ich ein elektronisches Exemplar und eines oder mehrere gedruckte und gebundene Exemplare eingereicht habe (z.B., weil der/die Prüfer/in(nen) dies wünschen): Das elektronisch eingereichte Exemplar stimmt exakt mit dem bzw. den von mir eingereichten gedruckten und gebundenen Exemplar(en) überein.

2. Erklärung bezüglich Veröffentlichungen

Eine Veröffentlichung ist häufig ein Qualitätsmerkmal (z.B. bei Veröffentlichung in Fachzeitschrift, Konferenz, Preprint, etc.). Sie muss aber korrekt angegeben werden. Bitte kreuzen Sie die für Ihre Arbeit zutreffende Variante an:

- ☒ Die Arbeit wurde bisher weder vollständig noch in Teilen veröffentlicht.
- ☐ Die Arbeit wurde in Teilen oder vollständig schon veröffentlicht. Hierfür findet sich im Anhang eine vollständige Tabelle mit bibliographischen Angaben.

3. Nutzung von Methoden der künstlichen Intelligenz (KI, z.B. chatGPT, DeepL, etc.)

Die Nutzung von KI kann sinnvoll sein. Sie muss aber korrekt angegeben werden und kann die Schwerpunkte bei der Bewertung der Arbeit beeinflussen. Bitte kreuzen Sie alle für Ihre Arbeit zutreffenden Varianten an und beachten Sie, dass die Varianten 3.4 - 3.6 eine vorherige Absprache mit dem/der Betreuer/in voraussetzen:

- ☐ 3.1. Keine Nutzung: Ich habe zur Erstellung meiner Arbeit keine KI benutzt.
- ☒ 3.2. Korrektur Rechtschreibung & Grammatik: Ich habe KI für Korrekturen der Rechtschreibung und Grammatik genutzt, ohne dass es dabei zu inhaltlich relevanter Textgeneration oder Übersetzungen kam. Das heißt, ich habe von mir verfasste Texte in derselben Sprache korrigieren lassen. Es handelt sich um rein sprachliche Korrekturen, sodass die von mir ursprünglich intendierte Bedeutung nicht wesentlich verändert oder erweitert wurde. Im Zweifelsfall habe ich mich mit meinem/r Betreuer/in besprochen. Alle genutzten Programme mit Versionsnummer sind im Anhang meiner Arbeit in einer Tabelle aufgelistet.

- ☒ 3.3. Unterstützung bei der Softwareentwicklung: Ich habe KI als Unterstützung beim Schreiben von Code in der Softwareentwicklung genutzt. Es handelt sich hierbei lediglich um Unterstützung und nicht um die automatische Generierung von größeren Programm-Teilen. Im Zweifelsfall habe ich mich mit meinem/r Betreuer/in besprochen. Alle genutzten Programme mit Versionsnummer sind im Anhang meiner Arbeit in einer Tabelle aufgelistet.
- ☐ 3.4. Übersetzung: Ich habe *nach vorheriger Absprache und mit Erlaubnis meines/r Betreuer/in* KI zur Übersetzung von mir in einer anderen Sprache geschriebenen Texte genutzt. Jede derartige Übersetzung ist im laufenden Text gekennzeichnet und der Anhang meiner Arbeit enthält eine Tabelle mit einem vollständigen Nachweis aller übersetzten Textstellen und der verwendeten Programme mit Versionsnummer.
- ☒ 3.5. Code-Generierung: Ich habe *nach vorheriger Absprache und mit Erlaubnis meines/r Betreuer/in* KI zur Erzeugung von Code in der Softwareentwicklung genutzt. Der Anhang meiner Arbeit enthält eine Tabelle mit einem vollständigen Nachweis aller derartigen Nutzungen, der verwendeten Programme mit Versionsnummer und – *falls von dem/der Betreuer/in gewünscht* – der verwendeten Prompts.
- ☐ 3.6. Text-Generierung: Ich habe *nach vorheriger Absprache und mit Erlaubnis meines/r Betreuer/in* KI zur Erzeugung von Text in meiner Arbeit genutzt. Jede derartige Verwendung von KI ist im laufenden Text gekennzeichnet und der Anhang meiner Arbeit enthält eine Tabelle mit einem vollständigen Nachweis aller derartigen Nutzungen, der verwendeten Programme mit Versionsnummer und – *falls von dem/der Betreuer/in gewünscht* – der verwendeten Prompts.

Falls ich in irgendeiner Form KI genutzt haben (siehe oben), dann erkläre ich:

Mir ist bewusst, dass ich die Verantwortung trage, falls es durch die Verwendung von KI zu fehlerhaften Inhalten, zu Verstößen gegen das Datenschutzrecht, Urheberrecht oder zu wissenschaftlichem Fehlverhalten (z.B. Plagiaten) kommt.

4. Abschluss und Unterschrift(en)

Mir ist bekannt, dass ein Verstoß gegen diese Erklärung prüfungsrechtliche Konsequenzen haben und insbesondere dazu führen kann, dass die Prüfungsleistung mit „nicht ausreichend“ bzw. die Studienleistung mit „nicht bestanden“ bewertet wird und bei mehrfachem oder schwerwiegendem Täuschungsversuch eine Exmatrikulation erfolgen bzw. ein Verfahren zur Entziehung eines eventuell verliehenen akademischen Titels eingeleitet werden kann.

Raphael Jontofsohn

Tübingen, 27.07.26

Vorname, Nachname
Student/in

Ort, Datum

Unterschrift

Die Punkte 3.4 - 3.6 erfordern eine Zustimmung des/r Betreuer/in. Sollten Sie einen dieser Punkte angekreuzt haben, dann sollte der/die Betreuer/in bitte hier unterschreiben:

Ich habe der oben genannten Nutzung von KI zur Erstellung der Arbeit zugestimmt.

Mario Krenn

Tübingen, 27.07.2026

Vorname, Nachname
Betreuer/in

Ort, Datum

Unterschrift