A Strut and Tie Neural Network Surrogate for Failure-State Prediction of Concrete D-Regions Designed by the Compatible Stress Field Method

Sandesh Lamsal[1] | Rubi Bhandari[2]
[1]Department of Civil and Architectural Engineering, University of Miami, Coral Gables, FL 33146, USA | [2]Department of Civil and Environmental Engineering, Florida International University, Miami, FL 33174, USA |

Correspondence: Sandesh Lamsal, Department of Civil and Architectural Engineering, University of Miami, Coral Gables, FL 33146, USA. sandeshlamsal@miami.edu

Keywords: Compatible Stress Field Method | strut and tie model | discontinuity regions | neural-network surrogate | machine learning | structural concrete | failure load

Abstract

The Compatible Stress Field Method (CSFM) designs the discontinuity regions (D-regions) of structural concrete, such as deep beams, hammerhead piers and pile caps, by combining a lower-bound stress field with kinematic compatibility and realistic constitutive behaviour, but its nonlinear solution procedure is too slow for the repeated evaluations that design-space exploration, reinforcement optimisation and reliability analysis demand. This study presents a neural-network surrogate that, in a single forward pass, maps a normalised description of a D-region design to its failure load factor and to the strut and tie (STM) member forces at the failure state, so the surrogate returns an interpretable and nearly statically admissible force state rather than an opaque capacity number. Training data come from sweeping an existing reference solver over a Latin Hypercube design of experiments across four D-region archetypes: deep beam, hammerhead pier, multi-column bent and pile cap. Trained per archetype, the surrogate predicts the failure load factor with a coefficient of determination of 0.96–0.99 and a mean absolute percentage error of 3–6 %, ahead of a panel of seven tabular baselines, and reproduces the reference member forces to within about 6 % in relative error (coefficient of determination above 0.99), with a nodal-equilibrium residual of about 1.5 % of the applied load. A bagged deep ensemble with split-conformal calibration attaches a prediction interval with a finite-sample coverage guarantee to each output, and a domain-of-validity flag built on the ensemble spread catches the majority of out-of-domain designs, on which accuracy degrades as expected. The reference solver is itself checked against an experimental pier-cap benchmark, completing a two-tier validation, though that tier rests on a single five-specimen series and inherits a systematic conservative bias. Against the reference solver the surrogate is between one and nearly four orders of magnitude faster, depending on the archetype and on batching, which makes the design-space exploration, reinforcement optimisation and reliability analysis of D-regions tractable.

1 | Introduction

Reinforced-concrete structures contain regions in which the Bernoulli hypothesis of plane sections does not hold: the vicinity of concentrated loads and supports, frame corners, corbels, deep beams, openings, pile caps and hammerhead piers. In these *discontinuity regions* (D-regions) the strain distribution is markedly nonlinear, and design must follow the actual flow of forces rather than sectional analysis. Strut and tie modelling (STM), placed on a consistent footing for design practice by Schlaich et al. [1], idealises this flow as a pin-jointed truss of concrete struts and reinforcement ties and applies the lower-bound theorem of plasticity to deliver a safe design.

Classical STM is powerful but has well-known limitations: the truss topology is not unique and relies on engineering judgement, serviceability (deformations and crack widths) is not addressed, and the verification of nodal zones is often conservative and laborious. The Compatible Stress Field Method (CSFM), developed by Kaufmann et al. [2] and now embedded in commercial design software, removes these limitations by combining a continuous (or discretised) stress field with kinematic compatibility and finite, realistic material laws: a parabola-rectangle law with compression softening for concrete [3] and a bilinear law with tension stiffening for reinforcement. The CSFM yields not only a safe strength verification but also the load–deformation response, the failure load and the governing failure mode. Figure 1 illustrates the four D-region archetypes considered in this study.

The price of this richness is computational cost. A CSFM analysis is an iterative nonlinear procedure: the reference load set is scaled by an increasing load factor, and at each increment the constitutive nonlinearity, chiefly the strain-dependent softening of cracked concrete, is resolved by inner iterations. A single design evaluation therefore involves many tens of linear solves. This is acceptable for the verification of one design, but it becomes the bottleneck whenever *many* designs must be evaluated: in parametric studies, reinforcement-layout optimisation, or probabilistic reliability assessment, where the same nonlinear analysis is repeated thousands of times.

Machine-learning surrogates are an obvious remedy, and data-driven models of structural response have proliferated. A surrogate that, once trained, returns the failure load of a D-region in a single forward pass would make these many-evaluation workflows tractable, provided it is accurate enough to be trusted and is validated against the solver it replaces.

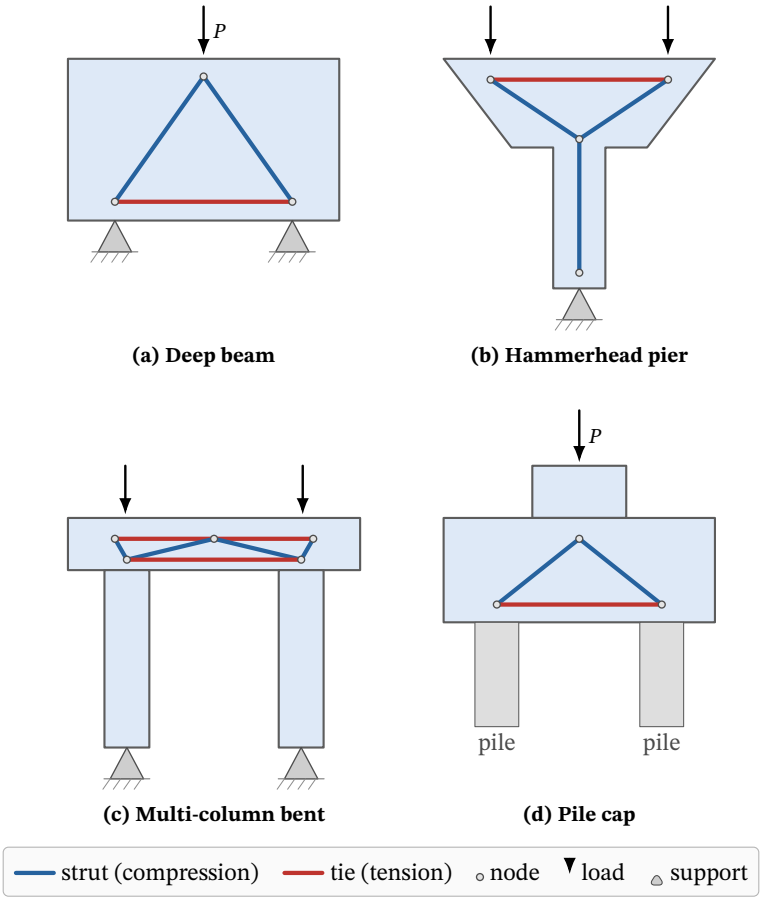

FIGURE 1 | The four discontinuity-region (D-region) archetypes covered by the dataset, each shown with its idealised strut and tie model: compression struts (blue) and tension ties (red) meeting at nodes. (a) A deep beam carries a top load to two supports through a direct strut and tie. (b) A hammerhead pier spreads two bearing loads through a tension tie across the cap and inclined struts into the column. (c) A multi-column bent forms a truss in the cap over two columns. (d) A pile cap spreads a column load through struts to the piles, closed by a bottom tie. Each is analysed by the Compatible Stress Field Method to generate ground-truth failure load factors.

This study presents a neural-network surrogate of a reference solver, a discrete strut and tie solver that applies the CSFM constitutive laws on the truss graph. The surrogate maps a D-region's design parameters to its failure load factor and to the strut and tie member forces at the failure state, so it returns the internal force state rather than the strength alone. In the terms of structural-concrete practice the surrogate addresses an assessment rather than a design task: the geometry and reinforcement of a D-region are given, and the quantity of interest is its load-bearing capacity, not a minimal-material layout[4]. It is developed and evaluated across four D-region archetypes, a deep beam, a hammerhead pier, a multi-column bent and a pile cap, against the reference solver and a set of standard tabular baselines. Each prediction carries a calibrated uncertainty interval, together with a domain-of-validity flag for designs that fall outside the training ranges. The approach is validated in two tiers: the surrogate against the reference solver, and the solver against an experimental pier-cap benchmark from the literature. The data-generation pipeline, the generated dataset, the trained models and the evaluation code are released (see the Data and code availability statement).

The surrogate replaces the reference solver only where many evaluations are needed, and its accuracy ceiling is that of the solver, which is checked against experiments in Section 3. This study is the first in a planned series on machine-learning methods for the CSFM; the companion studies develop an inverse stress-field reconstruction from strain measurements [5] and an arc-length-parametrised neural solver for the full equilibrium path [6].

The surrogate just outlined sits on three bodies of work: the stress-field methods that supply its physics, the physics-informed networks it deliberately sets aside, and the data-driven models it extends. The strut and tie method for the design of D-regions was placed on a consistent footing by Schlaich et al. [1], who unified B-region and D-region design under the lower-bound theorem of plasticity. The plasticity basis of truss and stress-field design was developed by Marti [7]. Stress-field methods generalise the discrete truss to continuous fields of compression and tension [8], while the discrete model itself continues to be refined for D-regions such as deep beams and corbels [9]. The CSFM, formalised by Kaufmann et al. [2], augments the stress field with kinematic compatibility and finite, realistic constitutive laws, including the compression softening of cracked concrete. It delivers the load–deformation response, the failure load and the crack widths within one code-oriented framework, and it underpins the reference solver used here to generate the training data. The compression softening on which the method depends traces back to the modified compression-field theory of Vecchio and Collins [10], which established that the compressive strength of concrete falls as transverse cracking increases, later simplified for routine design by Bentz et al. [11] and embedded in code-oriented frameworks such as the *fib* Model Code [12]; the CSFM carries that effect into a stress-field setting. The method occupies a middle ground between the discrete strut and tie model, which a designer lays out by hand, and a full nonlinear finite-element analysis: it keeps the transparency of the former while supplying the deformation capacity and failure load of the latter. That intermediate character makes it a suitable reference solver for a surrogate, since each analysis is informative yet individually inexpensive.

Bringing machine learning to such mechanics problems can take more than one form. One route embeds the physics directly in the network. Physics-informed neural networks (PINNs), popularised by Raissi et al. [13], embed the residual of a governing differential equation into the loss of a neural network, so that the network is trained to satisfy the physics with little or no labelled data. The broader programme is surveyed by Karniadakis et al. [14], the wider scientific-machine-learning landscape by Cuomo et al. [15], and software libraries such as DeepXDE support its use [16]. A complementary strand, the deep energy method of Samaniego et al. [17], replaces the strong-form residual by the system's potential energy. Such formulations have since been carried into solid mechanics, including continuum micromechanics [18]. Training such networks is not always straightforward. The residual loss can be stiff and ill-conditioned, and Wang et al. [19] document gradient-flow pathologies that stall convergence when a physics term competes with data terms. These difficulties are most pronounced where the governing relations are non-smooth, as the piecewise constitutive laws of the CSFM are. Most of this literature also targets continuous domains and uses automatic differentiation to evaluate PDE residuals, whereas a D-region analysed by the CSFM is naturally posed on a discrete strut and tie graph. The present study therefore does not use a PINN; it trains a purely supervised surrogate of the reference solver, and treats the solver itself as the carrier of the physics.

The other route, and the one taken here, learns from data alone. Data-driven surrogates have been applied widely to structural-response prediction, capacity estimation and design optimisation, built variously on response surfaces, Gaussian-process regression, tree ensembles and neural networks [20]; the breadth of these applications across structural design, assessment and health monitoring is captured in several recent reviews [21–24]. In the specific context of D-regions, machine learning has mostly been used either to predict a single scalar capacity, such as the shear strength of a deep beam [25,26], or to assist the selection of a strut and tie topology [27,28]. Both lines have limits that motivate the present work. The capacity predictors regress an empirical strength from a database of physical tests. They therefore inherit the scatter and the coverage gaps of that database, return a bare scalar with no internal force state, and seldom report a calibrated measure of confidence; the uncertainty-aware model of La et al. [26] is a recent exception on the last point, but still for a scalar capacity. The topology generators produce a strut and tie layout rather than a failure load, and a separate nonlinear analysis is still required to size and verify it. Even when the objective is the capacity itself, as in the reverse-engineering method of Yu and Kaufmann [4], which holds the reinforcement of an existing member fixed and maximises its load-bearing capacity, the strut and tie layout is recovered by a per-structure nonconvex optimisation with an engineer in the loop, so a fresh solve is needed for each design rather than a direct evaluation. That method also sets compression softening aside and reports load-bearing capacities not yet validated against experiments, both of which the present surrogate inherits from its compression-softening-aware reference solver and the experimental check of Section 3. A complementary line learns directly on the structural graph, as the graph-network simulators developed for meshes and particle systems do [29–31] and as a companion study applies to tensile-membrane form-finding [32], at the cost of a heavier architecture. Across these approaches, what remains open is a surrogate of a compatibility-based method, one whose labels already embed deformation capacity and compression softening rather than an empirical strength, that predicts the member-force state and reports a

calibrated uncertainty. Table 1 sets the present study against the representative data-driven approaches. This study develops such a surrogate and validates it in two tiers: against the solver it emulates and, through that solver, against physical experiments.

TABLE 1 | Positioning of this study against representative data-driven approaches to reinforced-concrete D-region design. "Forces" is whether the internal strut and tie member forces are produced; "UQ" is whether a calibrated uncertainty is reported.

Approach	Output	Trained on	Forces	UQ
Shear-strength ML [25]	scalar capacity	test database	no	no
UQ shear ML [26]	scalar capacity	test database	no	yes
STM generation [27,28]	truss topology	numerical opt.	layout	no
STM assessment [4]	STM + capacity	per-structure opt.	yes	no
Classical CSFM [2]	full response	— (solver)	yes	no
This study	failure load + forces	reference solver	yes	yes

A surrogate intended for reliability analysis must report not only a prediction but a measure of how far that prediction can be trusted. Deep ensembles, introduced by Lakshminarayanan et al. [33], estimate this by training several networks and reading the spread of their predictions as an epistemic-uncertainty signal; they need no change to the network and are a standard baseline for this purpose, alongside alternatives such as Monte-Carlo dropout [34] and the broader family of methods surveyed by Abdar et al. [35]. Their raw spread is not guaranteed to be calibrated, however. Conformal prediction [36,37] supplies the missing guarantee. From a held-out calibration sample it produces intervals whose coverage holds in finite samples, with no distributional assumption, and it extends naturally to regression through conformalised quantile regression [38]. The present study combines a deep ensemble with conformal calibration, using the ensemble spread to shape the interval and the conformal step to certify it, in Section 4.7.

The remainder of this study is organised as follows. Section 2 develops the surrogate. Section 3 validates the reference solver against physical experiments. Sections 4 and 5 report and discuss the results. Section 6 states the limitations and Section 7 concludes.

2 | Methodology

Building the surrogate that fills this gap begins with the solver it emulates and the data that solver produces. The surrogate is a fixed-size network tied to one strut and tie topology, so one network is trained per archetype. The mapping from design parameters to the failure state, the force normalisation, the uncertainty calibration and the data-generation pipeline are common to the archetypes; only the trained weights are specific to each. A single model spanning archetypes would require a graph-based architecture that admits a variable topology, which is outside the scope of this study.

2.1 | A discrete strut and tie solver with the CSFM constitutive laws

The reference solver that generates the training data is a discrete strut and tie solver: it applies the CSFM constitutive laws of Kaufmann et al. [2] on a truss graph, but represents each member by a fixed cross-sectional area. It is a computationally inexpensive approximation of the continuum CSFM, which resolves the transverse spreading of a strut that the fixed-area member cannot; the two are kept distinct throughout, and the accuracy cost of the discrete idealisation is quantified in Section 3. The solver is summarised here.

A D-region is represented by a space truss. Let $\mathcal{N}$ be the set of nodes, each with position $\mathbf{x}_i \in \mathbb{R}^3$ and prescribed support conditions, and $\mathcal{M}$ the set of members, each connecting two nodes and carrying an axial force only (tension positive). A reference load set applies nodal forces $\mathbf{P}_i$, scaled by a scalar load factor λ. Each member is assigned a concrete cross-sectional area $A_{c,m}$ and a reinforcement area $A_{s,m}$, so that it may act as a strut or as a tie depending on the sign of its strain. Figure 2 shows the strut and tie graph of a deep-beam D-region.

Kinematics. Let $\mathbf{u}_i \in \mathbb{R}^3$ be the displacement of node i. For a member m connecting nodes a and b, with unit axial vector $\mathbf{c}_m$ and length L_m, the axial elongation and average strain are

$$e_m = (\mathbf{u}_b - \mathbf{u}_a) \cdot \mathbf{c}_m, \qquad \varepsilon_m = e_m / L_m. \tag{1}$$

Constitutive laws. Concrete in compression follows a parabola-rectangle law [3], with the peak strain ε_{c2}, ultimate strain ε_{cu2} and exponent n taken as functions of the characteristic strength f_{ck}. The effective compressive strength of *cracked*

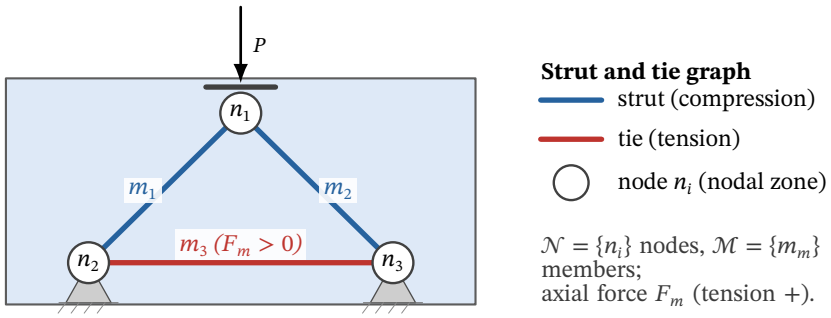

FIGURE 2 | The strut and tie graph of a deep-beam D-region under a central load P: the basic statically determinate model of two compression struts (blue), a tension tie (red) and three nodes. The load spreads through the inclined struts to the supports, and the bottom tie carries the horizontal thrust, each member m carrying an axial force F_m (tension positive). The Compatible Stress Field Method is posed on this discrete graph of nodes $\mathcal{N}$ and members $\mathcal{M}$, and the surrogate predicts the failure load factor and the member forces of the graph.

concrete is reduced by a compression-softening factor k_{c2} that decreases with the principal (transverse) tensile strain ε_1, and by a brittleness factor η_{fc}. In the discrete truss ε_1 is not available pointwise; it is estimated from the strains of the reinforcement crossing the strut and the effective reinforcement ratio, consistent with the host implementation. Reinforcing steel follows an idealised bilinear law with a yield plateau at f_y and a hardening branch to (ε_u, f_t). The two laws are plotted in Figure 3.

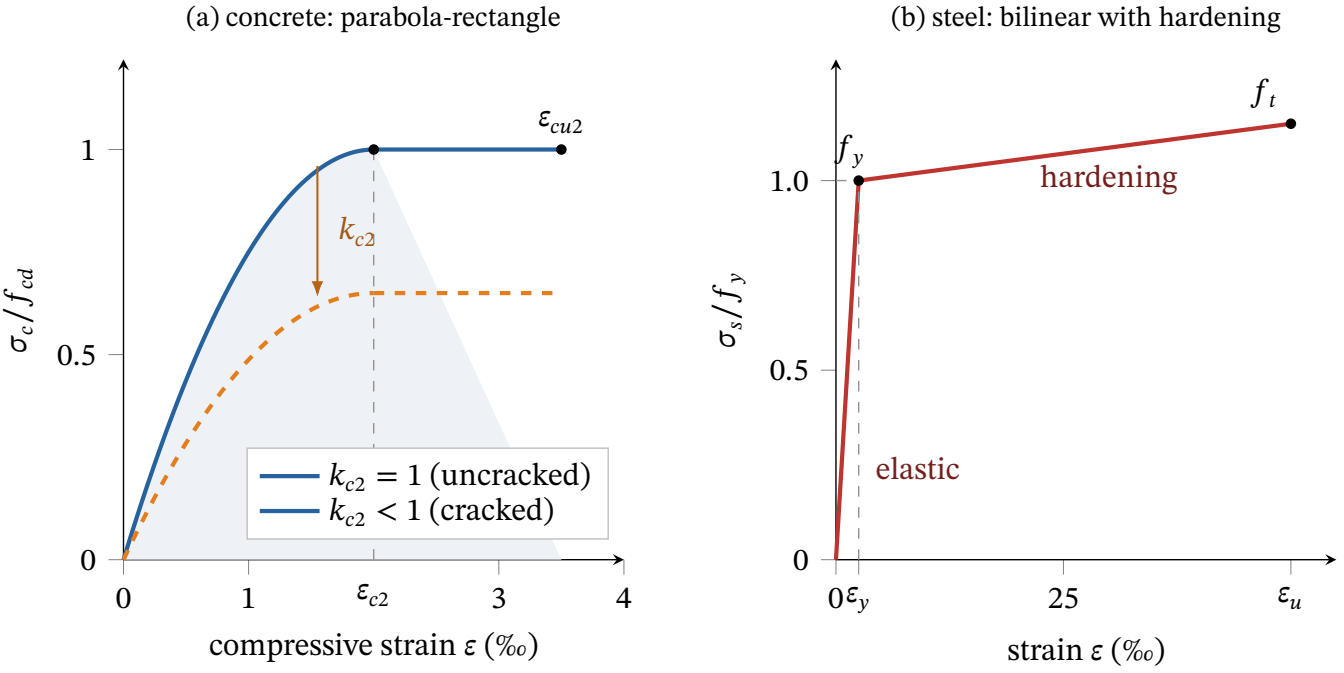

FIGURE 3 | Constitutive laws of the Compatible Stress Field Method, as evaluated by the reference solver. (a) The parabola-rectangle law for concrete in compression (blue), with peak strain ε_{c2}, ultimate strain ε_{cu2} and exponent taken as functions of f_{ck}; for cracked concrete the effective strength is lowered by the compression-softening factor $k_{c2} < 1$ (orange, dashed). (b) The idealised bilinear law for reinforcing steel, elastic to yield at ε_y then hardening to f_t at ε_u. Axes are normalised; the strain scale differs between the panels.

Member forces and the failure criterion. Given the strain ε_m, the member force is

$$F_m = \begin{cases} -A_{c,m}\,\sigma_c(|\varepsilon_m|, \varepsilon_1), & \varepsilon_m < 0 \quad \text{(strut action)}, \\ +A_{s,m}\,\sigma_s(\varepsilon_m), & \varepsilon_m \geq 0 \quad \text{(tie action)}, \end{cases} \tag{2}$$

with σ_c and σ_s the concrete and steel laws above. A member utilisation U_m is the ratio of its force to its capacity, taken as the effective concrete compressive resistance for struts and the yield force for ties. The failure load factor λ_f is the smallest load factor at which the maximum member utilisation reaches unity.

2.2 | Dataset generation

With the solver and its failure criterion fixed, the labels are produced by sweeping it over a designed sample of inputs. Four D-region archetypes are considered: a deep beam, a hammerhead pier, a multi-column bent and a pile cap. For each archetype the free parameters are the principal geometric dimensions, the concrete grade f_{ck}, the steel grade f_y, the provided reinforcement (area and bar diameter) and the reference load. Designs are drawn by Latin Hypercube Sampling[39] over the joint design space, which gives even space-filling coverage with far fewer samples than a full grid. The bounds of that design space are listed in Table 2 and are chosen to span ordinary design practice for each D-region type. Member dimensions cover small to large members, concrete grades range from about C20 to C70, reinforcement grades range from mild to high-yield steel, and bar sizes and counts stay within constructible limits. Samples that the solver reports as geometrically unstable or otherwise invalid are discarded and resampled. Each accepted design is analysed by the reference solver; the recorded quantities are the strut and tie geometry and connectivity, the member forces at the static solution, the failure load factor, the failure mode and the load–displacement curve. The failure load factor is the principal label; the member forces support the optional supervised anchor term. Designs are split into training, validation and test sets *by design*, not by load factor, so that no test design is seen during training. A separate extrapolation set, generated outside these ranges, probes generalisation and is examined in Section 4.9. The pipeline exports the dataset as JSON consumed by the training code, which decouples data generation from learning. Figure 4 summarises the workflow.

For each archetype, 750 valid designs are generated. The fixed-size network requires a single strut and tie topology per archetype. The minority of sampled geometries that flip a truss diagonal form a secondary topology and are set aside, which in practice affects only the hammerhead. The flip is not random across the design space: it occurs for the longer caps that cantilever well beyond a narrow column, every set-aside design having a cap-length-to-column-width ratio above about six (the dominant topology spans ratios of 2.6–6.0, the secondary one 6.0–9.8), so the trained hammerhead model covers the more compact dominant geometry. The retained designs are split 70/15/15 by design. Table 3 reports the resulting counts: each network is trained on 378 to 525 designs and tested on 81 to 113. The per-archetype accuracy of Section 4 is reported on the subset of test designs that fail within the analysed load range, which is smaller again; the full test split, not that subset, is the basis for generalisation.

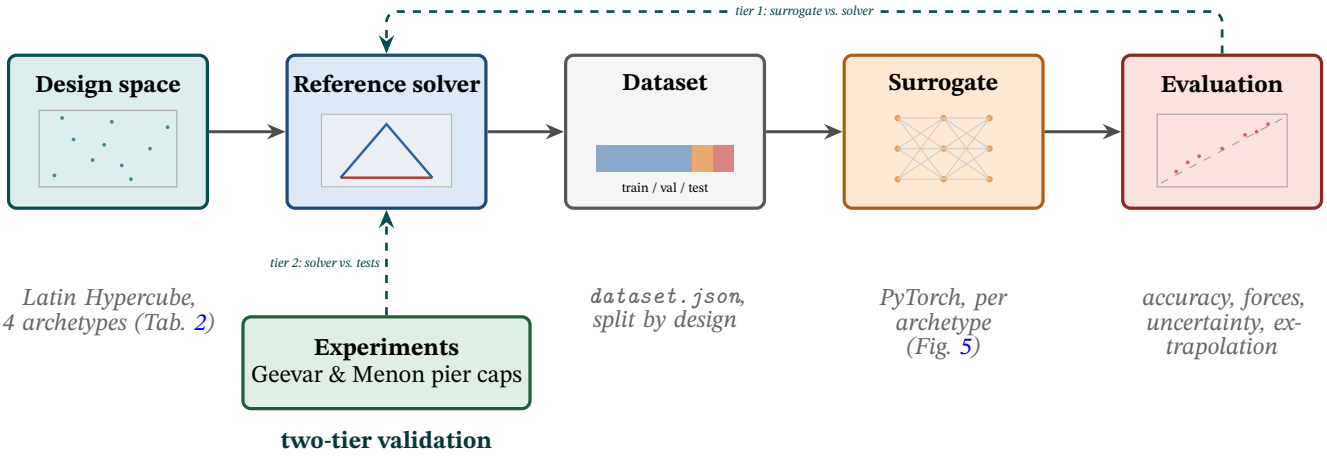

FIGURE 4 | Data-generation, training and validation pipeline. The reference solver, a discrete strut and tie solver with the CSFM constitutive laws, is swept over a Latin Hypercube design of experiments across four D-region archetypes, labelling each design with its failure load factor, member forces, failure mode and load–displacement curve. The dataset is split by design and a surrogate is trained per archetype in PyTorch, then evaluated against the solver. The validation has two tiers: the surrogate is checked against the reference solver (tier 1) and the solver against the Geevar and Menon pier-cap experiments (tier 2).

TABLE 2 | Design-space bounds sampled by Latin Hypercube Sampling. Lengths in mm, loads in kN, strengths in MPa. The concrete grade (f_{ck} = 21–69 MPa) and steel grade (f_y = 280–690 MPa) are common to all archetypes. Bar diameters are snapped to standard sizes (20, 25, 28, 32, 36 mm).

Archetype	Parameter	Range
Deep beam	span	2000–6000
	height	1400–3600
	web thickness	300–700
	support width	300–600
	point load P	1000–6000
	bottom-bar diameter	20–36
	bottom-bar count	4–10
Hammerhead	cap length	6000–12000
	cap depth (centre)	1600–3000
	cap depth (tip)	800–(centre−400)
	cap width	1200–2400
	column width	1200–2400
	bearing load (each)	600–2000
	top-bar diameter	20–36
	top-bar count	8–16
Multi-column bent	cap depth	1000–2200
	cap width	900–1600
	column width	700–1300
	overhang	800–2400
	girder load (each)	400–1400
	top-bar diameter	20–36
	top-bar count	6–12
Pile cap	cap length	2400–4200
	cap width	2400–4200
	cap depth	900–1800
	column width	500–900
	column load	3000–10000
	bottom-bar diameter	20–36
	bottom-bar spacing	120–250

TABLE 3 | Per-archetype dataset sizes. *Generated* is the number of valid Latin-hypercube designs analysed by the reference solver; *retained* is the subset sharing the dominant strut and tie topology, split by design into training, validation and test sets.

Archetype	Generated	Retained	Train	Val.	Test
Deep beam	750	750	525	112	113
Hammerhead	750	540	378	81	81
Multi-column bent	750	750	525	112	113
Pile cap	750	750	525	112	113

2.3 | Network and direct failure-state prediction

These labelled designs define a regression problem: a map from the design parameters to the failure state. Let a D-region design be described by a normalised parameter vector $\boldsymbol{\theta}$ collecting geometry, material properties ($f_{ck}, f_y, E_s, \ldots$), the provided reinforcement and the load pattern. The surrogate is a multilayer perceptron that maps $\boldsymbol{\theta}$ *directly* to the two quantities of interest:

$$\mathcal{S} : \boldsymbol{\theta} \longmapsto \left(\lambda_f,\ \mathbf{F}^{\mathrm{f}} \in \mathbb{R}^{|\mathcal{M}|}\right), \tag{3}$$

the failure load factor λ_f and the member forces $\mathbf{F}^{\mathrm{f}}$ at the failure state. An earlier formulation that predicted the full nodal displacement field at every load level and recovered λ_f by a search over the load factor proved fragile: the failure load is sensitive to small errors in the predicted field, and the search amplifies them. Predicting the failure state directly removes that indirection. The failure load factor is passed through a softplus so that it is positive. The member forces are predicted in a form normalised by the design's applied-load magnitude $F_0 = \|\mathbf{P}\|$ and recovered in physical units by rescaling; this keeps the regression target of order unity, since member forces scale with the applied load, so that the force head trains (an unnormalised head, with targets of order 10^6 N against a network whose natural output scale is of order unity, instead collapses to zero). The normalised forces are otherwise unconstrained (tension positive). Inputs are standardised to zero mean and unit variance over the training split. The strut and tie topology is fixed within an archetype, so one network is trained per archetype. The architecture is a multilayer perceptron with six hidden layers of width 128 and tanh activations;

with an input dimension of 9–10 design parameters and an output dimension of $1 + |\mathcal{M}|$, this amounts to roughly 85,000–88,000 trainable parameters depending on the archetype. The architecture and training settings are collected in Table 4, and Figure 5 shows the network.

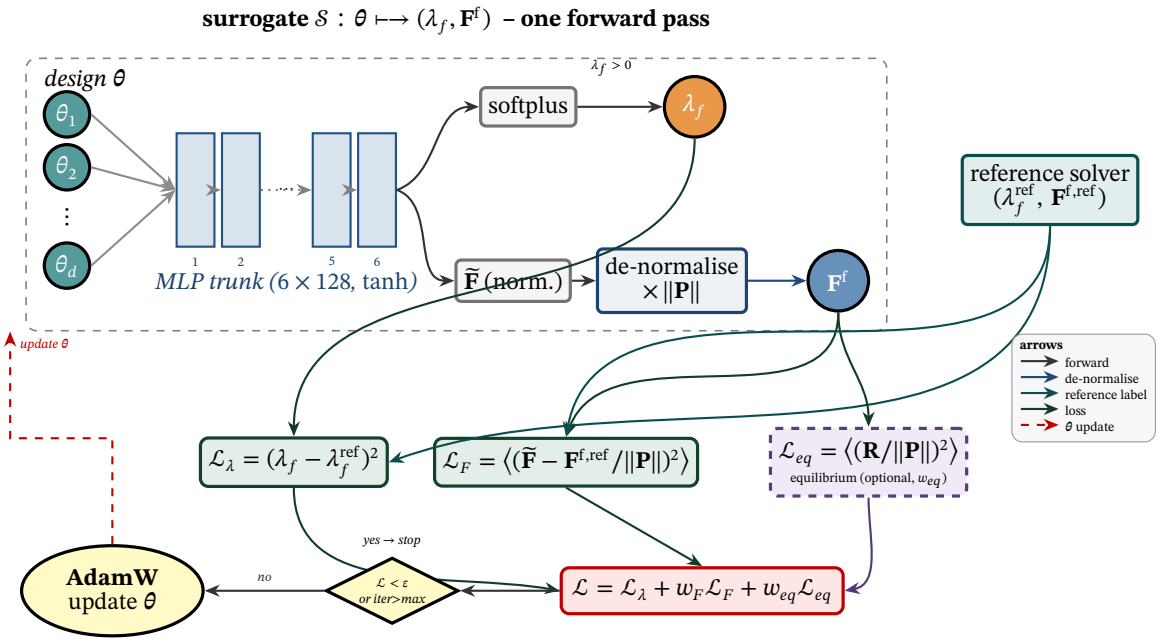

FIGURE 5 | The neural-network surrogate. A multilayer perceptron with a shared trunk (six hidden layers of width 128, tanh) maps the normalised design parameters θ to two heads: a failure-load head, passed through a softplus to give a positive failure load factor λ_f, and a force head that predicts the member forces normalised by the applied-load scale $\|\mathbf{P}\|$, de-normalised to physical forces $\mathbf{F}^{\mathrm{f}}$. Both are trained by supervision against the reference solver through the failure-load loss $\mathcal{L}_\lambda$ and the force loss $\mathcal{L}_F$; an optional nodal-equilibrium regulariser $\mathcal{L}_{eq}$ (weight w_{eq}, zero for the reported models) checks that the predicted forces balance the applied load. The total loss is minimised by AdamW.

2.4 | Loss

What that network learns is set by the loss, which supervises both of its outputs against the reference solver. The primary term is the squared error on the failure load factor,

$$\mathcal{L}_\lambda = \left(\lambda_f - \lambda_f^{\mathrm{ref}}\right)^2, \tag{4}$$

averaged over the design batch. A second term anchors the predicted failure-state member forces to those computed by the reference solver,

$$\mathcal{L}_F = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \left(\frac{F_m^{\mathrm{f}} - F_m^{\mathrm{f,ref}}}{F_0} \right)^2, \tag{5}$$

with $F_0 = \|\mathbf{P}\|$ the design's applied-load magnitude. This scale is set by the load the user applies and so is known at inference without the solver, unlike the per-design peak member force; normalising by it makes the force target of order unity, so that the surrogate actually learns to return a physically interpretable strut and tie force state alongside the failure load factor (Section 4.2). A third, optional term is the discrete nodal-equilibrium residual of the predicted failure state, $\mathcal{L}_{eq}$, which checks that the network's own predicted forces balance its predicted failure load at every free node; it is a physics regulariser weighted by w_{eq} and is zero for the reported models, its effect examined in Section 4.5. The total loss is

$$\mathcal{L} = \mathcal{L}_\lambda + w_F\,\mathcal{L}_F + w_{eq}\,\mathcal{L}_{eq}, \tag{6}$$

with w_F and w_{eq} fixed weights. The reported models use $w_F = 1$ and $w_{eq} = 0$; the sensitivity to both is examined in Section 4.5.

2.5 | Training

Minimising this loss fixes the weights of one network per archetype. The designs are split into training, validation and test sets *by design* (Section 2.2); training minimises Eq. (6) over mini-batches with AdamW[40] under a cosine-annealed learning rate, with gradient-norm clipping. The checkpoint with the lowest failure-load validation loss is retained. All archetypes share the architecture and training settings of Table 4; these were fixed early and not tuned per archetype, so the per-archetype accuracy differences of Section 4 reflect the data rather than hyperparameter search. Because the network outputs λ_f directly, evaluation is a single forward pass that requires no load-factor sweep and no constitutive evaluation; this is the source of the speed-up over the reference solver reported in Section 4.

TABLE 4 | Network architecture and training hyperparameters, shared by all four archetypes.

Setting	Value
Hidden layers	6
Hidden width	128
Activation	tanh
Input dimension	9–10 design parameters
Output dimension	$1 + \lvert\mathcal{M}\rvert$
Trainable parameters	85,000–88,000
Optimiser	AdamW
Learning rate	10^{-3}, cosine-annealed
Weight decay	10^{-5}
Gradient-norm clip	5.0
Batch size	64 designs
Epochs	400
Force-loss weight w_F	1.0
Equilibrium weight w_{eq}	0.0
Train / validation / test	70 / 15 / 15 % by design

3 | Validation of the reference solver

The surrogate of Section 2 is trained to reproduce the reference solver, so its predictions are only as physically meaningful as that solver. This section examines the solver, hereafter the *reference solver*, against physical experiments. The validation has two tiers: the surrogate is verified against the solver (Section 4), and the solver is checked here against measured failure loads. The scope of this second tier should be stated plainly. It is not a broad experimental validation of the CSFM. The method itself has been validated against extensive test programmes spanning many D-region types in the work that established it[2] and the modified compression-field theory it builds on[10,11], and the present study does not repeat that effort. The aim here is narrower: to confirm that the reference solver used to generate the training data reproduces a well-documented, five-specimen series for one representative D-region type, so that a surrogate trained on the solver inherits behaviour that is itself physically grounded. A wider benchmark spanning more geometries, loading conditions and failure modes would strengthen this tier, and is a stated limitation (Section 6). The accuracy claims for the surrogate are correspondingly made against the solver, not against measured strengths.

3.1 | Experimental benchmark

A benchmark that serves this narrow aim must be well documented and govern the strut-crushing mechanism of interest. The series of reinforced-concrete pier-cap tests of Geevar and Menon[41], analysed with the CSFM by Kaufmann et al.[2], meets both. Five specimens (S1–S5) with constant geometry and concentric loading are considered. Each is a stepped D-region comprising a wide cap band, a tapered transition and a narrow stem, loaded through a bearing plate on the stem and reacted by four supports under the cap. All five failed in the tests by concrete crushing of the diagonal strut. Following the validation convention of[2], the mean measured material properties are used with no partial safety factors ($\gamma = 1$). The reference solver already evaluates the failure load at nominal material strength, so this requires only that mean properties be supplied as input.

3.2 | Modelling and results

A geometrically faithful model is essential. When the stepped specimen is idealised as a prismatic block, the diagonal-strut concrete area is over-estimated, the strut never reaches its crushing strength, and the solver wrongly predicts a reinforcement-rupture failure for every specimen. Modelling the true stepped outline, with the strut concrete area taken from the real confined section (the out-of-plane thickness times the loading-plate width), restores the correct mechanism. The solver then predicts concrete crushing for four of the five specimens, consistent with the tests.

Table 5 compares the measured ultimate load P_{exp} with the solver prediction P_{calc} (P is the total applied load). Figure 6 consolidates the two tiers of the validation in one plot, showing the solver, the continuum CSFM analysis of the book and the surrogate of Section 3.4 against the measured loads.

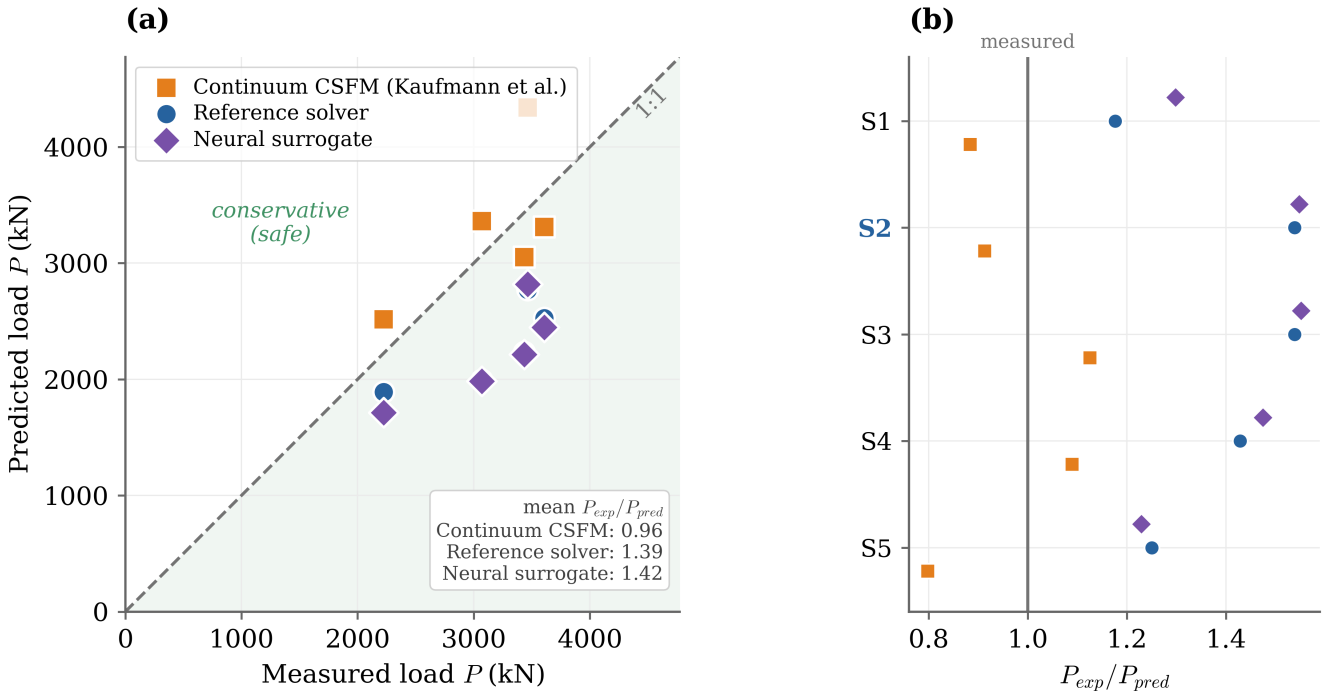

FIGURE 6 | Consolidated two-tier validation against the five Geevar and Menon[41] pier-cap specimens. (a) Predicted against measured ultimate load for the continuum CSFM analysis of Kaufmann et al., the reference solver and the neural surrogate of Section 3.4. The solver and surrogate cluster together in the shaded conservative region below the 1:1 line, so the surrogate tracks the solver (tier 1), while the continuum analysis lies nearer the line; the offset from 1:1 is the conservative bias against experiment (tier 2). (b) The experimental-to-predicted ratio per specimen for the three predictions, with the unity (measured) line; specimen S2, which the solver predicts to fail by reinforcement rupture, is marked. Per-specimen values are listed in Tables 5 and 6.

TABLE 5 | Reference-solver validation against the pier-cap tests of Geevar and Menon, with mean material properties and $\gamma = 1$.

Specimen	f_c (MPa)	P_{exp} (kN)	P_{calc} (kN)	P_{exp}/P_{calc}	failure mode
S1	31.2	2224	1890	1.18	crushing
S2	35.7	3068	1994	1.54	rupture
S3	30.1	3436	2233	1.54	crushing
S4	34.9	3608	2526	1.43	crushing
S5	34.1	3464	2771	1.25	crushing
mean / CoV				1.39 / 0.11	

The solver reproduces the experimental failure loads with low scatter, a coefficient of variation of 0.11 against the 0.13 of the continuum CSFM analysis reported in[2], but with a systematic conservative bias: the mean ratio $P_{exp}/P_{calc} = 1.39$, so the predicted capacity is about 72 % of the measured value.

3.3 | Discussion

The bias follows from representing each strut by a member of fixed cross-sectional area. A real compression strut spreads transversely, forming a "bottle" strut whose effective area exceeds that of the prismatic member used in the discrete model. The fixed-area idealisation therefore under-estimates the strut's effective area, over-estimates the compressive stress carried for a given force, and reaches the crushing limit at a lower applied load than the specimen sustains in test. The discrete solver consequently predicts a capacity below the measured strength, giving the mean conservative ratio $P_{exp}/P_{calc} = 1.39$ of Table 5. The continuum CSFM of Kaufmann et al.[2] resolves this spreading, and a continuum field also redistributes towards more favourable load paths than a fixed truss allows, so it is the less conservative of the two (Figure 6); transverse spreading is the dominant cause, not the only one. Figure 7 contrasts the bottle strut with the prismatic idealisation. The low scatter (CoV 0.11) confirms that the discrete model is mechanically consistent up to this scale factor. A continuum stress-field solver was also evaluated but produced spurious early failures at the re-entrant corners of the stepped outline and is not reported.

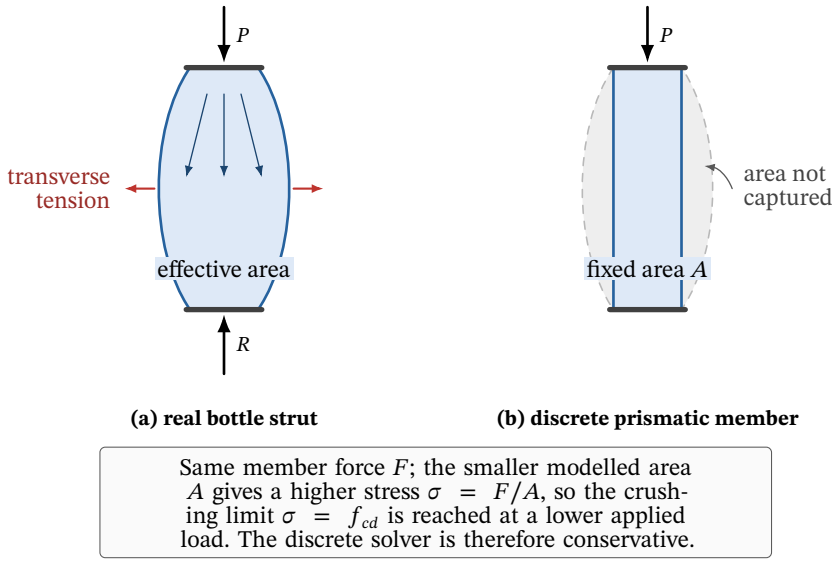

FIGURE 7 | A bottle-shaped compression strut and its fixed-area discrete idealisation. (a) A real compression strut spreads transversely into a "bottle" shape, so its effective area exceeds that of the loaded nodal zone, with transverse tension (red) at the edges of the spread. (b) The discrete reference solver represents the strut by a prismatic member of the fixed area assigned at the nodal zone; the effective area it does not capture is shaded. For the same member force, the smaller modelled area raises the compressive stress $\sigma = F/A$ and brings the crushing limit forward, the source of the conservative bias examined in Section 3.

The one specimen the solver misclassifies, S2, is informative. It is predicted to fail by reinforcement rupture rather than the observed concrete crushing, and Table 5 shows why this is a near miss rather than a gross error: S2 has the highest conservative ratio of the series ($P_{exp}/P_{calc} = 1.54$), so at the predicted load its diagonal strut sits just below the crushing threshold while the tie reaches yield first. The crushing and rupture mechanisms are closely competitive for this specimen, and the fixed-area strut idealisation, which under-estimates the strut capacity as discussed above, is enough to tip the predicted mechanism from one to the other. The same competition can occur in the generated dataset wherever a design sits near the crushing/rupture boundary, and there the surrogate inherits whatever mechanism the solver assigns. This is a limitation of the discrete formulation, not of the surrogate. It is one reason the predicted failure load, not the discrete failure-mode label, is treated as the quantity of record. The analysis of non-failing designs in Section 4.6 shows that the surrogate at least never mistakes a failing design for a surviving one.

Two implications follow. First, the reference solver is a sound basis for the surrogate: it predicts the correct failure mode for four of five specimens and ranks all five with low scatter, so a surrogate trained on it learns physically consistent behaviour. Second, the conservative bias for strut-crushing-governed D-regions is a genuine limitation of the discrete strut and tie formulation. For the design-screening use the surrogate is built for, a mean factor of about 1.4 on the safe side is acceptable, since a fast conservative estimate is well suited to ranking and filtering candidate designs. It is not acceptable as a final capacity, and the bias is addressed by the continuum treatment taken up in subsequent work. Crucially, the surrogate reproduces the solver, including this bias, and does not add error of its own, so the two tiers must be read together: the surrogate predicts the capacity the reference solver would compute, not the measured physical capacity.

3.4 | Direct validation of the surrogate

The two-tier validation establishes the surrogate's experimental accuracy only transitively. As a direct check, a surrogate was trained for the parametric pier-cap geometry of Figure 8 with the method of Section 2, treating it as a further archetype in the data-generation sweep. It reached a coefficient of determination of 0.97 on held-out synthetic designs. The trained surrogate was then evaluated on the five experimental specimens. Table 6 lists the measured ultimate loads with the reference-solver and surrogate predictions, and the consolidated Figure 6 shows the surrogate alongside the solver and the continuum analysis.

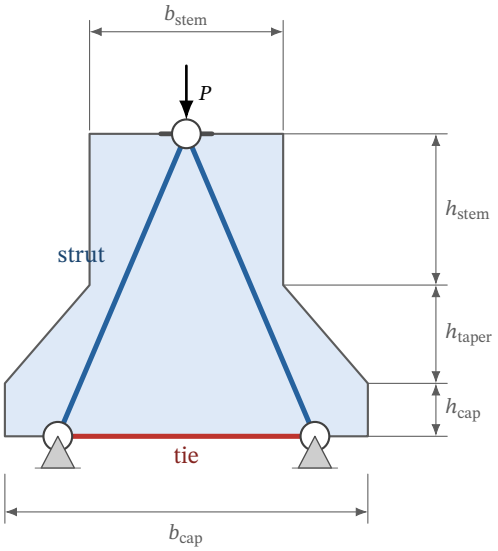

FIGURE 8 | The parametric pier-cap model used for the direct validation of Section 3.4. The stepped D-region, comprising a cap band, a tapered transition and a stem, is described by the cap and stem widths b_{cap} and b_{stem}, the band heights h_{cap}, h_{taper} and h_{stem}, the out-of-plane thickness, the loading-plate width and the support layout. A concentrated load P on the stem spreads through inclined concrete struts to the supports, tied along the base. The Geevar and Menon specimens are one instance of this family.

TABLE 6 | Direct validation of the surrogate against the five pier-cap experiments. Loads are total applied loads in kN.

Specimen	P_{exp}	solver	surrogate	P_{exp}/P_{surr}
S1	2224	1890	1713	1.30
S2	3068	1994	1982	1.55
S3	3436	2233	2215	1.55
S4	3608	2526	2447	1.47
S5	3464	2771	2817	1.23
mean				1.42

The surrogate predictions track the reference solver closely and lie a mean factor of 1.42 below the measured loads, against 1.39 for the solver. The surrogate reproduces the solver, including its conservative bias for strut-crushing-governed D-regions, rather than introducing error of its own. This is the expected behaviour of a surrogate trained to reproduce the solver, and it confirms that the two tiers are consistent. The surrogate's experimental accuracy is, to within a few per cent, that of the reference solver it replaces.

4 | Results

With the reference solver now grounded against experiment, the second tier can be read on its own terms: how faithfully the surrogate reproduces that solver. It is evaluated on the held-out test split (designs the network never saw during training, Section 2.2), with the reference solver as the ground truth. Failure-load accuracy is reported as mean absolute

percentage error (MAPE), root-mean-square error (RMSE) and coefficient of determination (R^2) against the reference solver failure load factor.

4.1 | Failure-load prediction accuracy

Table 7 reports accuracy per archetype on the test split. A separate network is trained for each archetype. The metrics are computed over the designs that fail within the analysed load range; non-failing designs (which do not fail by $\lambda = 3.0$) are excluded from the per-archetype statistics and discussed below.

TABLE 7 | Failure-load-factor prediction accuracy on the held-out test split, per archetype. n is the number of test designs that fail within the analysed load range.

Archetype	MAPE (%)	RMSE	R^2	n
Deep beam	5.6	0.120	0.964	92
Hammerhead	4.3	0.090	0.983	74
Multi-column bent	2.9	0.056	0.994	77
Pile cap	4.9	0.114	0.957	91

The surrogate reproduces the reference solver's failure load factor with a coefficient of determination between 0.96 and 0.99 and a mean absolute percentage error of 3–6 % across all four D-region archetypes. The multi-column bent is predicted most accurately ($R^2 = 0.994$, MAPE 2.9 %) and the pile cap least accurately ($R^2 = 0.957$), but the spread across archetypes is small. Including the non-failing designs, for which the surrogate correctly predicts a load factor at or above the analysed ceiling, raises R^2 to 0.97–1.00, since those designs are well separated from the failing ones. The non-failing designs are examined directly in Section 4.6. Figure 9 plots the predicted against the reference solver failure load factor for every test design. The points cluster tightly on the line of perfect agreement across the whole load-factor range.

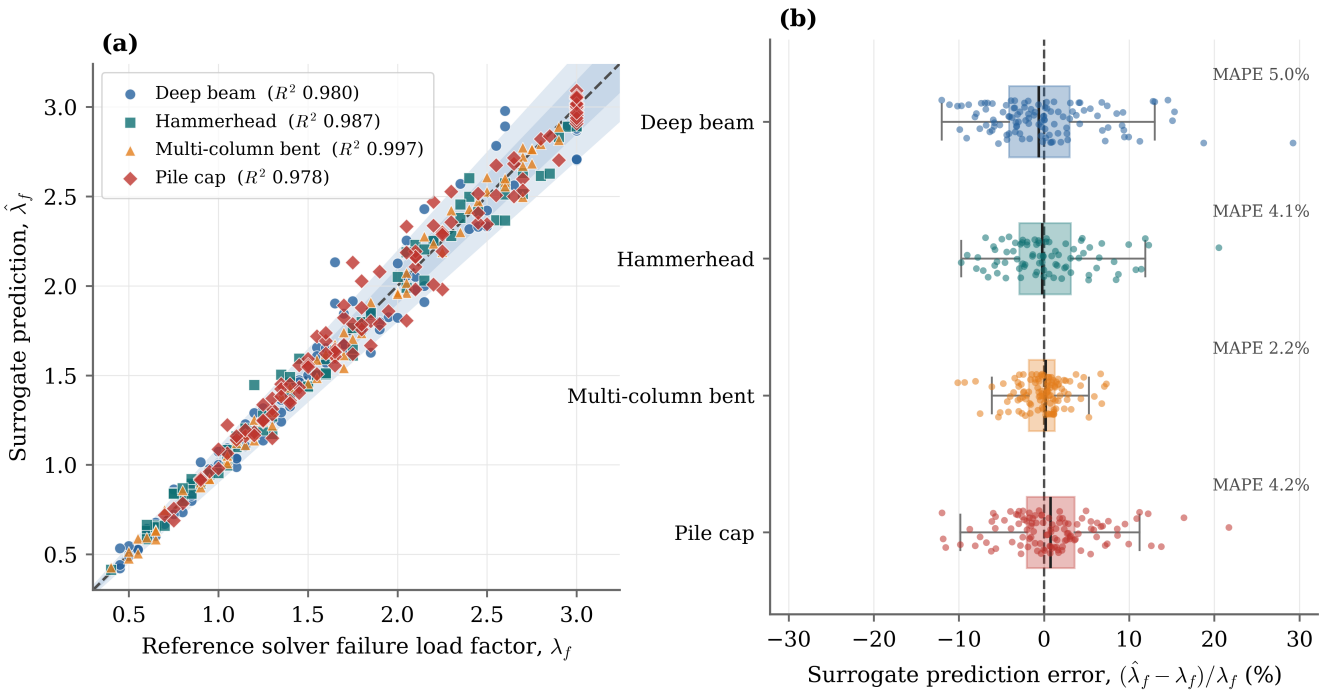

FIGURE 9 | Surrogate accuracy on the held-out test split, for all four archetypes. (a) Predicted failure load factor $\hat{\lambda}_f$ against the reference Compatible Stress Field Method solver λ_f; the dashed line is perfect agreement and the shaded bands mark the ±5% and ±10% tolerances. (b) Distribution of the signed prediction error per archetype, with the box spanning the interquartile range and the mean absolute percentage error annotated. Almost every prediction falls inside the ±10% band. Metrics shown here are computed over all test designs, including the non-failing ones at $\lambda_f = 3.0$; Table 7 reports the stricter statistics restricted to the genuine-failure designs.

4.2 | Member-force accuracy and equilibrium

The failure load factor on its own is only a capacity figure. The surrogate also predicts the strut and tie member forces at the failure state, and the value of that output rests on the forces being both accurate and mechanically consistent. Both are checked on the genuine-failure test designs. Table 8 reports, per archetype, the coefficient of determination of the predicted member forces against the reference CSFM forces (pooled over designs and members), the per-design relative L_2 error $\|\mathbf{F}^{\mathrm{f}} - \mathbf{F}^{\mathrm{f,ref}}\|/\|\mathbf{F}^{\mathrm{f,ref}}\|$, and the discrete nodal-equilibrium residual of the predicted forces, taken as the root-mean-square out-of-balance force over the free degrees of freedom and normalised by the applied-load magnitude.

TABLE 8 | Member-force accuracy and nodal-equilibrium consistency on the genuine-failure test designs. The equilibrium residual is the RMS out-of-balance force at the free nodes, normalised by the applied load; the reference CSFM forces balance to within 10^{-6} by construction.

Archetype	force R^2	rel. L_2 error (%)	equilibrium residual (%)
Deep beam	0.994	6.9	1.8
Hammerhead	0.996	5.9	1.6
Multi-column bent	0.997	5.8	1.2
Pile cap	0.995	5.8	1.8

The predicted member forces reproduce the reference CSFM forces with a coefficient of determination of 0.994–0.997 and a relative L_2 error of about 6 % across the four archetypes. The forces are predicted in a form normalised by the applied-load scale (Section 2.4), so that the regression target is of order unity and the force head trains; they are recovered in physical units by rescaling. The nodal-equilibrium residual of these predicted forces is only 1.2–1.8 % of the applied load, even though equilibrium is *not* imposed during training (the equilibrium weight is zero for the reported models): matching the reference forces is enough to render the predicted force state very nearly statically admissible. The residual can be reduced further by activating the equilibrium regulariser, as the ablation of Section 4.5 shows. The surrogate therefore returns a force state that is not merely a plausible vector but one that tracks the solver and balances the applied load.

4.3 | Computational efficiency

Because the trained network predicts the failure load factor in a single forward pass, inference is far cheaper than the reference solver, but a fair comparison must state the conditions. Table 9 reports wall-clock times measured on a single CPU core of the same machine, including input normalisation. The surrogate is timed both for a single design, warm model (the representative single-design engineering use), and amortised within a large batch (the many-query regime that motivates the surrogate); the one-time cost of loading the trained weights is about 3 ms and is paid once per session. The reference solver is timed per design on the same machine, where its cost depends strongly on the archetype, ranging from sub-millisecond for the planar deep beam and pile cap to about 19 ms for the multi-column bent.

TABLE 9 | Wall-clock timing on one CPU core. The surrogate cost is essentially archetype-independent (the same small network); the reference-solver cost is per design and varies with the archetype. The speed-up is therefore a range rather than a single figure.

Quantity	Time per design
Surrogate, single design (warm, incl. normalisation)	57 µs
Surrogate, batched (amortised)	3.2 µs
Reference solver, deep beam	0.86 ms
Reference solver, pile cap	0.76 ms
Reference solver, hammerhead	4.2 ms
Reference solver, multi-column bent	19.4 ms

The resulting speed-up spans from roughly 15× for single-design use against the fastest solver case (the deep beam) to several thousandfold for batched evaluation against the slowest (the multi-column bent). In the batched many-query regime that the surrogate is built for, the speed-up is between two and nearly four orders of magnitude depending on the archetype, and even the least favourable single-design comparison is more than an order of magnitude. This makes the surrogate suitable for the design-space exploration, optimisation and reliability workflows that motivate this study.

4.4 | Comparison with baselines

Whether a neural network is needed at all is tested against a panel of standard tabular regressors, trained on the same design parameters, the same per-archetype train/test split and the same target: ridge linear regression, a degree-two polynomial response surface, k-nearest neighbours, support-vector regression with a radial-basis kernel, a random forest, a gradient-boosted-tree ensemble[42] and Gaussian-process regression[43]. These span the families most commonly used for structural surrogates, from linear and kernel models to tree ensembles. Table 10 compares their coefficient of determination on the genuine-failure test designs with that of the neural surrogate.

TABLE 10 | Failure-load coefficient of determination R^2 on the genuine-failure test designs, per archetype, for a panel of tabular baselines and the neural surrogate. The mean absolute percentage error follows the same ordering.

Model	Deep beam	Hammerhead	Multi-col. bent	Pile cap
Ridge (linear)	0.758	0.799	0.902	0.643
k-nearest neighbours	0.598	0.686	0.687	0.632
Random forest	0.617	0.636	0.858	0.560
Gradient-boosted trees	0.773	0.838	0.883	0.768
Polynomial (degree 2)	0.863	0.887	0.892	0.839
Support-vector (RBF)	0.934	0.921	0.935	0.903
Gaussian process	0.921	0.927	0.951	0.906
Neural surrogate	**0.964**	**0.983**	**0.994**	**0.957**

The neural surrogate is the most accurate model on every archetype. The gap over the tree ensembles and the simpler models is wide: a coefficient of determination of 0.96–0.99 against 0.56–0.90 for the random forest, gradient-boosted trees, k-nearest neighbours and ridge regression. The two strongest baselines, support-vector regression and Gaussian-process regression, are much closer, reaching 0.90–0.95; the neural surrogate still leads them on all four archetypes, by a smaller and honest margin. The failure load factor of a D-region is a smooth but strongly coupled function of its design parameters: a change in one dimension shifts the inclination of every strut at once. Axis-aligned tree ensembles, which split on one input at a time, approximate such a response poorly, whereas the kernel methods and the multilayer perceptron, which mix the inputs, capture it well. The network's edge over the kernel methods reflects its capacity to represent the smooth high-order interactions directly.

4.5 | Force-loss and equilibrium-weight ablation

Whether the force output costs any failure-load accuracy can be tested by varying the loss weights. The loss combines the primary failure-load term with a force-supervision term of weight w_F and an optional equilibrium regulariser of weight w_{eq} (Section 2.4). Table 11 reports how the accuracy responds to w_F, retraining each archetype at $w_F \in \{0, 0.1, 1, 10\}$ with $w_{eq} = 0$, and summarises the effect of w_{eq}.

TABLE 11 | Effect of the force-loss weight w_F (with $w_{eq} = 0$), reported as the range across the four archetypes. The failure-load accuracy is essentially flat in w_F, while any $w_F > 0$ trains the force head and collapses the equilibrium residual.

w_F	λ_f R^2	force R^2	equilib. residual (%)
0	0.948–0.991	—	37–61
0.1	0.949–0.992	0.993–0.997	1.7–2.8
1	0.956–0.994	0.995–0.998	1.0–1.5
10	0.957–0.995	0.996–0.998	0.7–1.1

The failure-load accuracy is insensitive to the force weight: its coefficient of determination moves by less than 0.01 across two orders of magnitude in w_F, including the limit $w_F = 0$ where the force term is absent. Adding the force term therefore costs nothing in failure-load accuracy. The force head, by contrast, depends on it entirely: with $w_F = 0$ the forces are unconstrained and their equilibrium residual is tens of per cent, whereas any $w_F \geq 0.1$ brings the force coefficient of determination above 0.99 and the residual to a few per cent. The chosen value $w_F = 1$ sits in the middle of this flat, well-behaved region, so the result does not hinge on its precise setting. Raising the equilibrium weight w_{eq} from 0 to 1 at $w_F = 1$

tightens the residual further, for example from 1.0 % to 0.7 % for the multi-column bent, with no measurable change in failure-load accuracy, confirming that the physics term sharpens consistency but is not required for it.

4.6 | Non-failing designs

The accuracy of Section 4.1 is reported on designs that fail within the analysed range; designs that do not fail by the ceiling $\lambda = 3.0$ are termed non-failing, since their true failure load is known only to exceed the ceiling. Rather than set these aside, their handling is examined directly, because a surrogate that silently mistook a non-failing design for a failing one (or the reverse) would be unsafe. Treated as a failure-versus-no-failure decision, with a design predicted to be non-failing when $\hat{\lambda}_f \geq 3.0$, the surrogate classifies the held-out test designs with an accuracy of 0.89–0.95 per archetype, 0.91 pooled over 420 designs. The error structure matters more than the rate: of the 420 designs, none of the genuinely failing designs is misclassified as non-failing (no false negatives), and all 36 misclassifications are non-failing designs predicted to fail. Every error therefore falls on the conservative side, flagging a design as failing when in fact it survives the range, which would at worst refer a safe design back to the solver. The mean one-sided shortfall on the non-failing designs, $\max(0, 3.0 - \hat{\lambda}_f)$, is below 0.06 in load-factor units. Excluding the non-failing designs from the accuracy tables thus removes neither a hidden failure mode nor an unsafe bias.

4.7 | Predictive uncertainty

A point prediction alone is of limited use in the reliability and optimisation settings that motivate the surrogate: those workflows need a sense of how far each prediction can be trusted. Two standard tools are combined to provide this.

The first is a *bagged* deep ensemble[33]. For each archetype, ten networks are trained, each on an independent bootstrap resample of the training split, so that the members genuinely disagree where the training data is sparse rather than merely where their initial weights happen to differ. The ensemble mean $\bar{\lambda}_f$ is taken as the prediction and the across-member standard deviation σ as a raw uncertainty estimate. The ensemble mean matches the accuracy of the single network, with a coefficient of determination of 0.96–0.99 across the four archetypes, and σ is positively correlated with the actual prediction error (pooled Spearman rank correlation $\rho = 0.49$, ranging from 0.26 for the hammerhead to 0.54 for the deep beam), so it does carry information about which predictions are least reliable. This correlation is moderate rather than strong, and σ on its own is not a precise predictor of the error of any single design. It does not need to be. The conformal step below requires only that σ rank design difficulty well enough for the interval width to track it, and a correlation of this strength is sufficient for that, while the coverage guarantee itself holds regardless of how well σ is correlated with the error. That the width does track difficulty is confirmed by binning the test designs into terciles by conformal interval width: the mean absolute error rises monotonically from the narrow to the wide tercile for every archetype, for example from 0.04 to 0.15 in load-factor units for the deep beam.

Read directly, however, σ is *under-dispersed*: the members agree more than their error warrants, so a Gaussian interval $\bar{\lambda}_f \pm z\sigma$ covers far fewer test designs than its nominal level (the lower curve of Figure 10(b)). The second tool corrects this. Split-conformal calibration[36,37] forms the σ-normalised nonconformity score $|\lambda_f - \bar{\lambda}_f|/\sigma$, evaluates it on the held-out validation split, and uses its empirical quantile q to construct the interval $\bar{\lambda}_f \pm q\sigma$. Because the validation designs are exchangeable with the test designs and were not used to fit the ensemble, this interval carries a finite-sample coverage guarantee, while its width still scales with σ and so widens for the designs the ensemble finds hard.

The calibration is effective. At a target coverage of 90 % the conformal multiplier is $q \approx 2.0$–2.8, and the resulting intervals cover 85–95 % of the test designs per archetype, 90 % pooled, in line with the target, with a mean half-width of about 0.09–0.22 in load-factor units. Figure 10(a) shows the ensemble prediction with its 90 % conformal interval, and Figure 10(b) confirms that the conformal coverage tracks the target across the whole range, where the raw ensemble spread does not. Because the reviewer of a reliability analysis will want this per archetype rather than pooled, Figure 11 repeats the reliability diagram for each archetype separately. The conformal interval tracks the target in every case, while the raw ensemble spread is under-dispersed in every case. The surrogate therefore returns not a bare number but a calibrated, design-specific interval, which is the form of uncertainty a reliability analysis can use directly.

4.8 | Data efficiency and robustness

Each network has of order 85,000 parameters and is trained on only 378–525 designs, which raises a legitimate question of overfitting. Two checks address it. First, the learning curves of Figure 12 plot the held-out test error against the fraction of the training set used, each point the mean of three independent random subsamples. The error falls smoothly and its

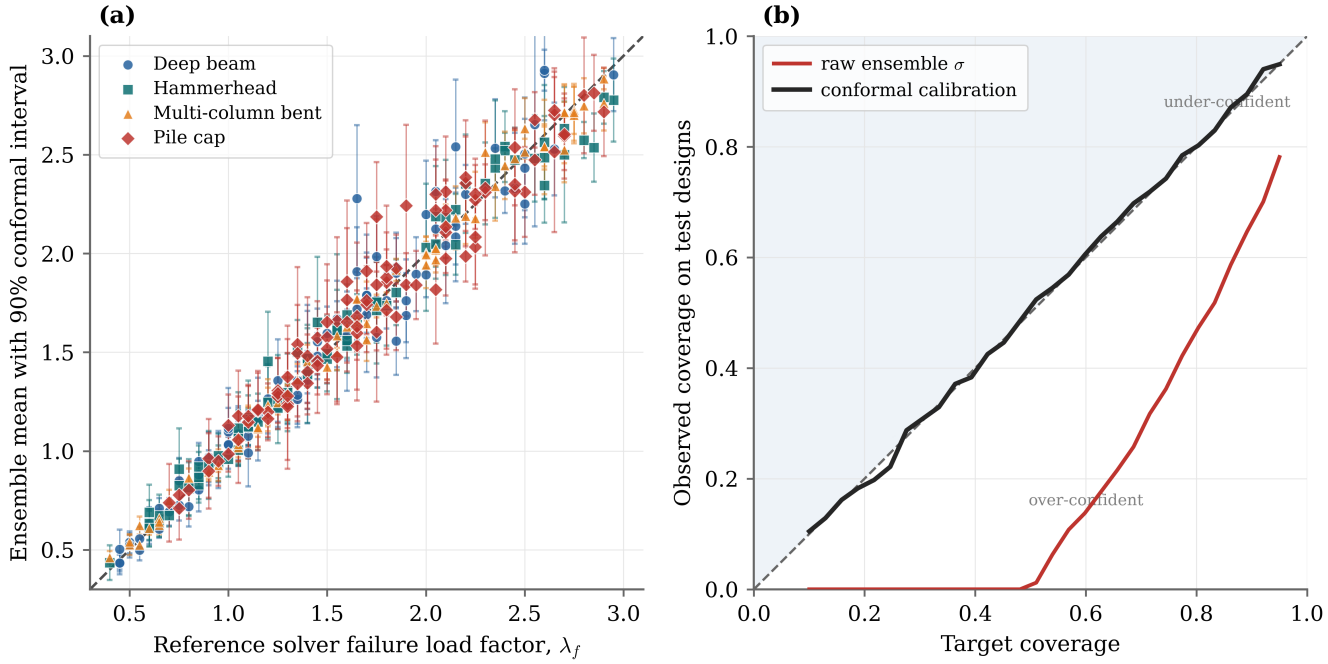

FIGURE 10 | Predictive uncertainty of the surrogate, on the genuine-failure test designs of all four archetypes. (a) Bagged-ensemble mean prediction against the reference Compatible Stress Field Method solver, with bars showing the 90% split-conformal prediction interval; its width adapts to the ensemble spread. (b) Reliability diagram: the raw ensemble standard deviation, read as a Gaussian interval, is under-dispersed and falls below the diagonal, whereas the σ-normalised conformal interval tracks the target coverage, as its finite-sample guarantee requires.

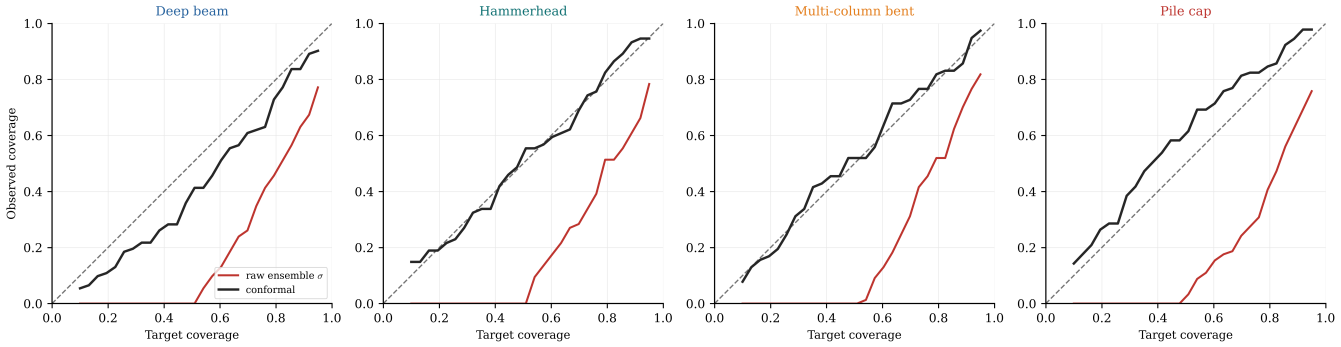

FIGURE 11 | Per-archetype reliability diagrams. For every archetype the raw ensemble standard deviation, read as a Gaussian interval, is under-dispersed and falls below the diagonal, whereas the σ-normalised split-conformal interval tracks the target coverage.

run-to-run scatter shrinks as data is added, with no sign of the train/test divergence that overfitting produces. The curve is flattening at the full training size, indicating the networks are not starved of data there, though the gentle residual slope shows that more data would still help modestly. Second, the headline configuration was retrained on five independent train/validation/test splits. The test coefficient of determination is stable across them, with a standard deviation of at most 0.012 (deep beam 0.981 ± 0.008, hammerhead 0.986 ± 0.003, multi-column bent 0.992 ± 0.001, pile cap 0.952 ± 0.012), so the reported accuracy is a property of the method and the data, not of one fortunate split.

4.9 | Out-of-domain extrapolation

The accuracy above is measured by interpolation within the sampled design space. For design use the more dangerous question is how the surrogate behaves *outside* that space, and whether it knows when it is there. To probe this, additional designs were generated in a shell just outside the training box on every design parameter, at three distances $\delta = 0.1, 0.2, 0.3$ of the parameter range beyond each edge, and labelled by the same reference solver. The trained surrogate was evaluated

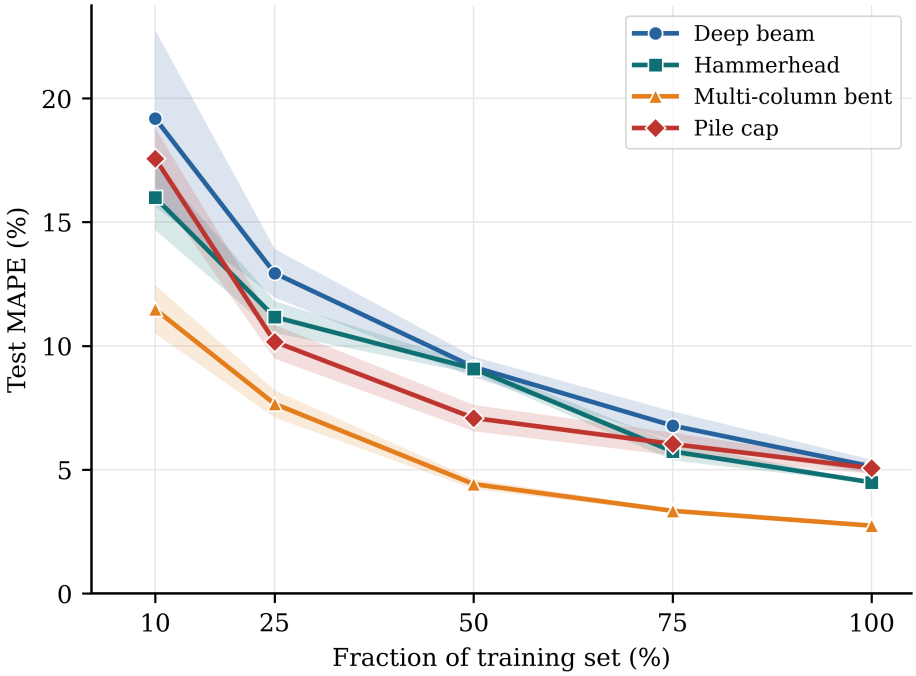

FIGURE 12 | Learning curves: held-out test mean absolute percentage error against the fraction of the training set used, for each archetype (the full set is 378–525 designs, fewest for the hammerhead). Each point is the mean of three independent random training subsamples; the band is one standard deviation. The error flattens well before the full training set, indicating the networks are not starved of data at the sizes used.

on them with its training-set normalisation. Table 12 reports the accuracy and the fraction of designs caught by a domain-of-validity flag, defined as an ensemble spread σ exceeding the 95th percentile of the in-domain test spread.

TABLE 12 | Out-of-domain behaviour. R^2 on genuine-failure designs at increasing distance δ outside the training box, and the percentage of designs caught by the ensemble-spread domain-of-validity flag (calibrated to fire on 5 % of in-domain designs).

	R^2				flagged (%)	
Archetype	in-dom.	δ=0.1	δ=0.2	δ=0.3	in-dom.	δ=0.3
Deep beam	0.964	0.840	0.676	0.359	5	36
Hammerhead	0.983	0.914	0.867	0.700	5	52
Multi-column bent	0.994	0.737	0.625	0.528	5	64
Pile cap	0.957	0.864	0.782	0.581	5	46

The accuracy degrades smoothly and monotonically with distance from the training box, as a purely supervised surrogate must: by $\delta = 0.3$ the coefficient of determination has fallen from above 0.95 to between 0.36 and 0.70. The surrogate should therefore not be trusted outside its training ranges. The value of the uncertainty machinery is that it recognises this: the domain-of-validity flag, set to fire on only 5 % of in-domain designs, catches 36–64 % of the designs at $\delta = 0.3$, because the ensemble spread roughly doubles outside the box. Out-of-domain predictions are thus not silently wrong but flagged as untrustworthy, which is the behaviour a design or optimisation workflow that strays beyond the training domain requires.

5 | Discussion

The surrogate predicts the reference solver's failure load factor to within 3–6 % (coefficient of determination 0.96–0.99) across the four archetypes, ahead of every model in a seven-strong baseline panel, while also returning the failure-state member forces to within about 6 % and at a cost between one and nearly four orders of magnitude below the reference solver. Four aspects of these results bear discussion: why a neural model suits this response, what the force prediction adds, how the uncertainty and domain-of-validity behave together, and what the solver's conservative bias means for use.

The margin over the baseline panel is itself informative. The failure load factor of a D-region is a smooth but strongly coupled function of its geometry and material parameters: a change in one dimension shifts the inclination of every strut at once. A tree ensemble approximates such a function with axis-aligned splits and piecewise-constant cells, which is inefficient when the response turns on combinations of inputs rather than on any single input, and indeed the random forest and gradient-boosted trees trail badly (R^2 down to 0.56). The kernel methods, support-vector and Gaussian-process regression, mix the inputs and do far better (0.90–0.95). The multilayer perceptron, which also mixes the inputs but can represent smooth high-order interactions with fewer effective degrees of freedom, leads them on every archetype, though by an honest and modest margin. The lesson is one of inductive bias rather than tuning: neural surrogates are the appropriate tool wherever the underlying mechanical response is smooth and coupled, but a well-chosen kernel model is a respectable second.

The force prediction is what separates this surrogate from a scalar capacity regressor, and the equilibrium check is what makes it trustworthy. The predicted member forces both track the reference forces (coefficient of determination above 0.99) and balance the applied load at every free node to within about 1.5 %, even though equilibrium is not imposed during training. The network has therefore learned a statically coherent force state, not an arbitrary vector that happens to fit a few numbers. The engineer can read the output as a strut and tie force field, seeing which members carry the load and which ties reach yield, and an optimisation loop can act on the member forces directly rather than on a single capacity number. The ablation shows this richer output costs nothing in failure-load accuracy, which is the usual fear when a second task is added to a network.

The uncertainty machinery supplies the two pieces of trust information an automated workflow needs, one inside the training domain and one at its edge. Inside, the bagged ensemble with conformal calibration attaches a calibrated prediction interval to each output, guaranteed in finite samples and wider for the designs the ensemble finds hard, so a large sweep can be triaged automatically: narrow-interval predictions are trusted and only wide-interval ones referred back to the solver. At the edge, the same ensemble spread flags out-of-domain designs, on which the accuracy is shown to degrade. A sweep that strays outside the training box is then not silently wrong but marked as untrustworthy. The non-failing-design analysis adds a third safeguard: the surrogate never mistakes a failing design for a surviving one, so it can double as a fast failure-screening classifier. None of this needed any change to the network, only a held-out calibration sample and the ensemble spread.

The surrogate inherits the conservative bias of the reference solver, which under-predicts the strength of strut-crushing-governed D-regions by a mean factor of about 1.4 against the pier-cap experiments (Section 3). For the design-screening use that motivates the surrogate this bias falls on the safe side and is acceptable: a fast, conservative estimate is well suited to ranking and filtering candidate designs. It does mean, however, that the surrogate predicts the capacity the reference solver would compute, not the true physical capacity, so the two tiers of validation must be read together and a final design check should still rest on the underlying method. The accuracy is otherwise consistent across archetypes, the pile cap least ($R^2 = 0.957$) and the multi-column bent most ($R^2 = 0.994$). With that scope understood, the surrogate is directly usable in the design-space exploration, reinforcement-optimisation and reliability workflows that motivate this study. Its accuracy ceiling is that of the reference solver validated in Section 3.

6 | Limitations and future work

That ceiling is one of four bounds on the method. First, the surrogate inherits the strut and tie topology supplied by the host solver and does not itself select or optimise that topology. Because the topology is fixed within an archetype, one network is trained per archetype; a single network spanning all archetypes would require a graph-based architecture, such as a graph-network simulator[30,31]. Second, the surrogate's accuracy ceiling is that of the reference solver, which is itself conservatively biased for strut-crushing-governed D-regions (Section 3). Third, the scope is restricted to the four archetypes of Section 2.2. Fourth, the experimental tier of the validation rests on a single five-specimen pier-cap series; it checks the reference solver for one D-region type but is not a broad experimental validation, and a wider benchmark spanning more geometries and failure modes would strengthen it. The conformal intervals of Section 4.7 are a further point to note: their coverage guarantee is *marginal*, holding on average over designs rather than conditionally for every design, so a sharper design-conditional guarantee remains open.

Addressing these limitations, notably the conservative bias of the discrete formulation, the per-archetype scope and the design-conditional sharpening of the uncertainty intervals, is the focus of the studies that follow this one in a planned series, where a continuum treatment of the stress field is taken up in turn through an inverse stress-field reconstruction[5] and an arc-length-parametrised equilibrium-path solver[6].

7 | Conclusions

This study presented a neural-network surrogate for the failure load of concrete D-regions designed by the Compatible Stress Field Method. The network maps a D-region's design parameters directly to its failure load factor and its failure-state strut and tie member forces. Trained per archetype against a reference solver, it predicts the failure load factor with a coefficient of determination of 0.96–0.99 and a mean absolute percentage error of 3–6 % across the four D-region archetypes, ahead of every model in a seven-strong baseline panel. It also reproduces the reference member forces to within about 6 % in relative error, with a coefficient of determination above 0.99 and a nodal-equilibrium residual of about 1.5 % of the applied load, so that its output is an interpretable and nearly statically admissible force state rather than a bare capacity, and this force prediction costs nothing in failure-load accuracy. A bagged deep ensemble with split-conformal calibration attaches to each output a prediction interval with a finite-sample coverage guarantee, while the ensemble spread doubles as a domain-of-validity flag that catches the majority of out-of-domain designs, on which the accuracy degrades as expected. Against the reference solver, whose own cost ranges from sub-millisecond to about 20 ms per design, the surrogate is between one and nearly four orders of magnitude faster, depending on the archetype and on batching.

In practice the surrogate is suited to the inner loop of design-space exploration and reinforcement optimisation, where its member-force output lets the objective act on the force flow and not only on a scalar capacity, and to reliability analysis, where its calibrated intervals quantify the prediction error. Two cautions should accompany that use. The surrogate reproduces the solver, including the solver's conservative bias, so it is a screening tool rather than a final capacity check. It should be applied only within its training ranges, with the domain-of-validity flag enforced. The natural next steps are a continuum treatment of the stress field to remove the conservative bias, a graph-based architecture that spans archetypes in a single model, and a design-conditional sharpening of the uncertainty intervals; these are taken up in the studies that follow this one in a planned series.

NOMENCLATURE

Roman symbols

$A_{c,m}$ concrete cross-sectional area of member m
$A_{s,m}$ reinforcement (steel) cross-sectional area of member m
E_s elastic modulus of reinforcing steel
F_m axial force in member m (tension positive)
F_m^{f} member force at the failure state
F_0 applied-load magnitude, $F_0 = \|\mathbf{P}\|$ (force normaliser)
f_c concrete compressive strength
f_{ck} characteristic (cylinder) concrete compressive strength
f_t ultimate (tensile) strength of reinforcing steel
f_y yield strength of reinforcing steel
k_{c2} compression-softening factor of cracked concrete
L_m length of member m
n exponent of the parabola-rectangle concrete law
P total applied load on the D-region
P_{exp} measured (experimental) ultimate load
P_{calc} reference-solver (calculated) failure load
P_{surr} surrogate-predicted failure load
R^2 coefficient of determination
U_m utilisation of member m (force divided by capacity)
w_F weight of the force-anchor loss term
w_{eq} weight of the equilibrium-residual loss term
z standard-normal multiplier of a Gaussian prediction interval

Greek symbols

γ partial safety factor on material strengths ($\gamma = 1$: mean properties)
δ fractional distance beyond the training-box edge (extrapolation)
ε_m average axial strain of member m

ε_1 principal (transverse) tensile strain

ε_{c2} strain at peak compressive stress of concrete

ε_{cu2} ultimate compressive strain of concrete

ε_u ultimate strain of reinforcing steel

η_{fc} concrete brittleness factor

λ load factor scaling the reference load set

λ_f failure load factor (smallest λ at unit member utilisation)

λ_f^{ref} reference-solver failure load factor

$\hat{\lambda}_f$ surrogate-predicted failure load factor

$\bar{\lambda}_f$ ensemble-mean failure load factor

$\lambda_{\max}$ analysed load-factor ceiling ($\lambda_{\max} = 3.0$)

ρ Spearman rank correlation between σ and the prediction error

σ ensemble across-member standard deviation (raw uncertainty)

σ_c, σ_s concrete and steel constitutive (stress) laws

$\boldsymbol{\theta}$ normalised design-parameter vector (network input)

Sets, vectors and operators

$\mathcal{N}$, $\mathcal{M}$ sets of nodes and of members ($|\mathcal{N}|$, $|\mathcal{M}|$ their sizes)

$\mathcal{S}$ surrogate map $\boldsymbol{\theta} \longmapsto (\lambda_f, \mathbf{F}^{\mathrm{f}})$

$\mathcal{L}$ total training loss

$\mathcal{L}_\lambda$ load-factor loss term

$\mathcal{L}_F$ force-anchor loss term

$\mathcal{L}_{eq}$ nodal-equilibrium-residual loss term

$\mathbf{P}_i$ nodal force of the reference load set at node i

$\mathbf{F}$, $\mathbf{F}^{\mathrm{f}}$ vector of member forces; at the failure state

$\mathbf{u}_i$ displacement vector of node i

$\mathbf{x}_i$ position vector of node i ($\in \mathbb{R}^3$)

$\mathbf{c}_m$ unit axial vector of member m

e_m axial elongation of member m

$\|\cdot\|$ Euclidean (L_2) norm

$\hat{(\cdot)}$, $\bar{(\cdot)}$ surrogate-predicted and ensemble-mean quantity

Subscripts and superscripts

i node index

m member index

c, s concrete; reinforcing steel

$c2$, $cu2$ peak and ultimate points of the concrete law

f failure state

ref reference-solver quantity

0 applied-load (normalising) quantity

Abbreviations

CSFM Compatible Stress Field Method

STM strut and tie model

D-region discontinuity region

MLP multilayer perceptron

MAPE mean absolute percentage error

RMSE root-mean-square error

LHS Latin Hypercube Sampling

MCFT modified compression-field theory

PINN physics-informed neural network

UQ uncertainty quantification

DATA AND CODE AVAILABILITY

The dataset-generation pipeline, the generated dataset, the trained model weights and the evaluation code will be released in a public repository and archived with a permanent identifier on publication.

DECLARATION OF COMPETING INTEREST

The authors declare no competing financial interests or personal relationships that could have influenced the work reported in this study.

FUNDING

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

1. Jorg Schlaich, Kurt Schafer, and Mattias Jennewein. Toward a consistent design of structural concrete. *PCI Journal*, 32(3): 74–150, 1987.
2. Walter Kaufmann, Jaime Mata-Falcón, Manuel Weber, and Tena Galkovski. *Compatible Stress Field Design of Structural Concrete: Principles and Validation*. ETH Zurich and IDEA StatiCa s.r.o., Zurich, 2020. ISBN 978-3-906916-95-8.
3. CEN. *EN 1992-1-1:2004 Eurocode 2: Design of Concrete Structures – Part 1-1: General Rules and Rules for Buildings*. European Committee for Standardization, Brussels, 2004.
4. Karin Yu and Walter Kaufmann. Reverse engineering strut-and-tie models for assessing reinforced concrete structures. *Structural Concrete*, 2026. doi: 10.1002/suco.70637. Early view.
5. Sandesh Lamsal and Rubi Bhandari. Inverse physics-informed neural network for CSFM stress field reconstruction in concrete D-regions. Preprint, engrXiv, 2026.
6. Sandesh Lamsal. A physics-informed neural network with arc-length parametrisation for the equilibrium path of CSFM discontinuity regions. Preprint, engrXiv, 2026.
7. Peter Marti. Basic tools of reinforced concrete beam design. *ACI Journal*, 82(1):46–56, 1985.
8. Aurelio Muttoni, Joseph Schwartz, and Bruno Thurlimann. *Design of Concrete Structures with Stress Fields*. Birkhauser, Basel, 1997.
9. Panatchai Chetchotisak, Jaruek Teerawong, and Sukit Yindeesuk. Modified interactive strut-and-tie modeling of reinforced concrete deep beams and corbels. *Structures*, 45:284–298, 2022. doi: 10.1016/j.istruc.2022.08.116.
10. Frank J. Vecchio and Michael P. Collins. The modified compression-field theory for reinforced concrete elements subjected to shear. *ACI Journal*, 83(2):219–231, 1986.
11. Evan C. Bentz, Frank J. Vecchio, and Michael P. Collins. Simplified modified compression field theory for calculating shear strength of reinforced concrete elements. *ACI Structural Journal*, 103(4):614–624, 2006.
12. fédération internationale du béton (fib). *fib Model Code for Concrete Structures 2010*. Ernst & Sohn, Berlin, 2013.
13. Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
14. George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.
15. Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics-informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.
16. Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021.
17. Esteban Samaniego, Cosmin Anitescu, Somdatta Goswami, Vien Minh Nguyen-Thanh, Hongwei Guo, Khader Hamdia, Xiaoying Zhuang, and Timon Rabczuk. An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. *Computer Methods in Applied Mechanics and Engineering*, 362:112790, 2020.
18. Alexander Henkes, Henning Wessels, and Rolf Mahnken. Physics informed neural networks for continuum micromechanics. *Computer Methods in Applied Mechanics and Engineering*, 393:114790, 2022. doi: 10.1016/j.cma.2022.114790.
19. Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.
20. Lili Xing, Paolo Gardoni, Ge Song, and Ying Zhou. Deep learning-based surrogate capacity models and multi-objective fragility estimates for reinforced concrete frames. *Computer Methods in Applied Mechanics and Engineering*, 440:117928, 2025. doi: 10.1016/j.cma.2025.117928.
21. Hadi Salehi and Rigoberto Burgueño. Emerging artificial intelligence methods in structural engineering. *Engineering Structures*, 171:170–189, 2018.

22. Han Sun, Henry V. Burton, and Honglan Huang. Machine learning applications for building structural design and performance assessment: State-of-the-art review. *Journal of Building Engineering*, 33:101816, 2021.
23. Huu-Tai Thai. Machine learning for structural engineering: A state-of-the-art review. *Structures*, 38:448–491, 2022. doi: 10.1016/j.istruc.2022.02.003.
24. Majdi Flah, Itzel Nunez, Wassim Ben Chaabene, and Moncef L. Nehdi. Machine learning algorithms in civil structural health monitoring: A systematic review. *Archives of Computational Methods in Engineering*, 28:2621–2643, 2021.
25. De-Cheng Feng, Wen-Jie Wang, Sujith Mangalathu, Gang Hu, and Tao Wu. Implementing ensemble learning methods to predict the shear strength of rc deep beams with/without web reinforcements. *Engineering Structures*, 235:111979, 2021. doi: 10.1016/j.engstruct.2021.111979.
26. Hung La, Nguyen-Vu Luat, Kihak Lee, and Tan Nguyen. Uncertainty-aware prediction of shear strength in reinforced concrete deep beams using quantile machine learning and explainable artificial intelligence. *Structures*, 84:110907, 2026. doi: 10.1016/j.istruc.2025.110907.
27. Yi Xia, Matthijs Langelaar, and Max A. N. Hendriks. Optimization-based strut-and-tie model generation for reinforced concrete structures under multiple load conditions. *Engineering Structures*, 266:114501, 2022. doi: 10.1016/j.engstruct.2022.114501.
28. Yi Xia, Matthijs Langelaar, and Max A. N. Hendriks. Optimization-based three-dimensional strut-and-tie model generation for reinforced concrete. *Computer-Aided Civil and Infrastructure Engineering*, 36(5):526–543, 2021. doi: 10.1111/mice.12614.
29. Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.
30. Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 8459–8468, 2020.
31. Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations (ICLR)*, 2021.
32. Sandesh Lamsal. DeepForm: A graph neural network surrogate for real-time tensile membrane form-finding across anticlastic typologies. Preprint, engrXiv, 2026.
33. Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.
34. Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 1050–1059, 2016.
35. Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.
36. Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J. Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018.
37. Anastasios N. Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023.
38. Yaniv Romano, Evan Patterson, and Emmanuel J. Candès. Conformalized quantile regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.
39. Michael D. McKay, Richard J. Beckman, and William J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.
40. Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.
41. Indu Geevar and Devdas Menon. Strength of reinforced concrete pier caps — experimental validation of strut-and-tie method. *ACI Structural Journal*, 116(1):261–272, 2019. doi: 10.14359/51711138.
42. Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.
43. Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.