SHAPFM: AMORTIZED SHAPLEY ESTIMATION WITH A PRIOR-FITTED MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Shapley values are among the most widely used tools for explainable AI, with applications ranging from feature attribution and data valuation to causal explanation. In these settings, a machine learning problem is represented as a cooperative game over p players, and the goal is to estimate each player's Shapley value. Traditional estimators query each game and estimate Shapley values from scratch. In contrast, recent amortized approaches such as FastSHAP and ExplainerPFN learn across related games to predict Shapley values directly. These methods exploit shared structure, but their predictions are static: once trained, they cannot improve as additional evaluations of the current game become available. We introduce ShapFM, an adaptive amortized estimator that combines the strengths of both approaches. ShapFM is a prior-fitted transformer-based model, pretrained on synthetic ML games to estimate Shapley values given evaluations of a game. Once trained, it yields Shapley estimates for real-world ML games with one forward pass. Like a traditional estimator, its predictions improve as more game evaluations are provided; unlike a traditional estimator, it can leverage structure learned from many related games to obtain accurate estimates from very few queries. We design the architecture specifically around the structure of cooperative games, alternating attention over coalitions and players, incorporating the underlying inputs that define each game, and optionally using evaluations of related games as additional context. Across a broad collection of machine learning games, ShapFM is competitive with state-of-the-art, heavily optimized traditional estimators and can substantially outperform them in the low-query regime.

1 INTRODUCTION

Machine learning models inform decisions in medicine, finance, and public policy, where understanding why a model makes a prediction can be as important as the prediction itself (Doshi-Velez & Kim, 2017; Gilpin et al., 2018). Explainable artificial intelligence (XAI) develops methods for understanding these models (Barredo Arrieta et al., 2020). Shapley values provide a common framework for several such tasks: attributing a prediction to input features (Chen et al., 2023; Fumagalli et al., 2025; Lundberg & Lee, 2017; Štrumbelj & Kononenko, 2014), valuing training data (Jia et al., 2019), explaining causal effects (Jung et al., 2022), quantifying hyperparameter importance (Wever et al., 2026), and more (Rozemberczki et al., 2022).

In the Shapley value framework, a cooperative game $\nu : 2^{\mathcal{P}} \to \mathbb{R}$ assigns a real value to each subset, or coalition, of a set $\mathcal{P}$ of p players. For feature attribution, the players are the input features, and $\nu(S)$ is the prediction when the features in S are available, under a specified treatment of the missing features. The Shapley value of player i averages the change in value when that player joins a coalition (Shapley, 1953):

$$\phi_i(\nu) = \sum_{S \subseteq \mathcal{P} \setminus \{i\}} \frac{\nu(S \cup \{i\}) - \nu(S)}{p\binom{p-1}{|S|}}. \tag{1}$$

The Shapley value is the unique attribution rule satisfying *symmetry*, *dummy*, *linearity*, and *efficiency* (Shapley, 1953). Rozemberczki et al. (2022) review these axioms and their machine learning applications. The Shapley value can be applied to any game, but the choice of ν determines what its attributions mean. For feature attribution, this includes the output being explained and how missing

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

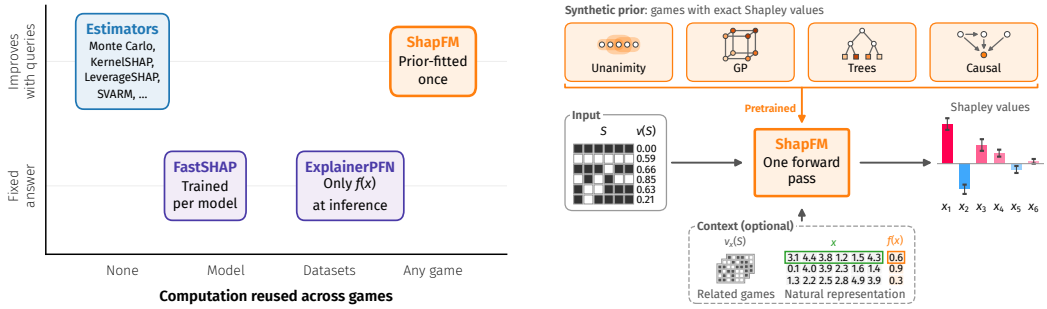

Figure 1: Overview of ShapFM. Left: ShapFM combines *reuse* across games with refinement from additional coalition queries. **Right**: a model pretrained on synthetic games with exact Shapley values maps queried coalitions and optional context to Shapley estimates with uncertainty in one forward pass.

features are handled, with conditional expectations and interventions answering different questions (Covert et al., 2021; Janzing et al., 2020).

For some games, structure makes *exact* Shapley computation tractable (Deng & Papadimitriou, 1994; Grabisch, 2016). Examples include sparse sums of unanimity games (Harsanyi, 1963), Gaussian processes on the hypercube (Rundel et al., 2026), suitable explanation games of tree ensembles (Lundberg et al., 2020; Yu et al., 2022; Zern et al., 2023), and causal models (Witter et al., 2026). Further exact methods have been developed for special cases of feature attribution games (Mohammadi et al., 2025a;b; Muschalik et al., 2025; Štrumbelj & Kononenko, 2014) and data valuation games (Wang et al., 2024; 2023). We exploit the four families above to generate synthetic games with exact Shapley values serving as a pretraining prior for ShapFM (see Figure 1, right).

For a general game, however, exact computation requires up to 2^p evaluations of ν. Even with methods that make coalition evaluations cheaper (Chau et al., 2023; Olsen & Christensen, 2026; Rundel et al., 2024), the practical goal is to estimate Shapley values from a limited budget b of accurate evaluations. Existing estimators broadly follow two approaches, with formal identities in Section A. *Monte Carlo estimators* approximate the Shapley sum or an equivalent reformulation through sampling, often with stratification, antithetic pairing, or sample reuse (Castro et al., 2009; 2017; Kolpaczki et al., 2024; Mitchell et al., 2022). *Surrogate-based estimators* fit an approximation to the game and compute its Shapley values. They use linear or polynomial models (Charnes et al., 1988; Fumagalli et al., 2026a;b; Lundberg & Lee, 2017; Musco & Witter, 2025), tree ensembles (Chen & Guestrin, 2016; Ke et al., 2017; Zern et al., 2023), Gaussian processes (Rundel et al., 2026), or sparse interaction models (Butler et al., 2025), and may add an estimated residual correction (Thies et al., 2026; Witter et al., 2025).

As usually applied, these estimators build an approximation *from scratch* for each game. Monte Carlo averages and fitted surrogates use observations of the current game without learning from previous games, even when the games are closely related. This raises the guiding question of our work:

Can we learn how to estimate Shapley values across games, so that observations of a new game refine an already useful estimate?

Two existing methods amortize Shapley estimation across related games: ExplainerPFN and FastSHAP (see Figure 1, left).

ExplainerPFN (Fonseca & Stoyanovich, 2026) is pretrained to predict feature attributions from tabular inputs and observed model predictions, without querying feature coalitions at inference. These pairs do not generally determine the model's behavior on coalitions, and hence its Shapley values. In our comparison, its recommended correction still leaves error near that of predicting zero (Figure 5).

FastSHAP (Jethani et al., 2022) trains a network to explain different inputs to the same machine learning model. For example, each input defines a game in which a coalition's value is the average prediction when its features are kept and the others are taken from reference inputs. FastSHAP

has been applied to images (Covert et al., 2023), text (Yang et al., 2023), additive models (Enouen & Liu, 2025), neural operators (Stiller et al., 2026), and learning from noisy, unbiased attribution targets (Covert et al., 2024). Once trained, FastSHAP explains a new input in one forward pass, but changing the machine learning model requires training or adaptation, and additional coalition observations do not refine its output.

1.1 Our Contributions

We introduce *ShapFM* (Shapley Prior Fitted Model), a Shapley estimation algorithm learned from a synthetic prior over games (see Section 1.2 for related prior-fitted models). Given observed pairs $(S, \nu(S))$ and the values of the empty and grand coalitions, it predicts the Shapley values in one forward pass. Through pretraining, the model learns to approximate the distribution of Shapley values conditional on these observations.

❶ **Pretraining on exact games.** We combine tractable game families–unanimity, Gaussian processes, tree-ensembles, and causal games–into a synthetic prior spanning varied interaction patterns and explanation tasks. Building on exact methods for structured games, we generate each training game together with its ground-truth Shapley values.

❷ **Combining amortization and estimation.** Our transformer architecture alternates attention over coalitions and players, is equivariant to player order, and satisfies efficiency by construction. It refines its estimates as additional coalition evaluations become available, without retraining. When available, it also uses evaluations of related games as context, for example when explaining the same machine learning model at different inputs. For feature attribution, it can also read the original feature values and model predictions, rather than seeing each feature only as present or absent in a coalition.

❸ **Empirical results.** We evaluate one checkpoint on more than 50 game families, including explanations of tabular, image, and text models, hyperparameter importance, and synthetic games. With game context where available, ShapFM achieves the best mean rank in the low-budget regime (Table 1). This regime was, so far, reserved for amortized explainers such as FastSHAP.

1.2 Related Work on In-context Shapley Estimation

Prior-data fitted networks (PFNs) are conceptually closely related to ShapFM. They generalize across predictive tasks in-context, form the backbone of frontier tabular foundation models, and are target explicands of many XAI methods. ShapFM is also driven by a powerful prior and yields predictions in-context, but uses different architectural components and a different learning task. We discuss this relation in the following.

ShapFM and PFNs. ML models can be trained to learn algorithms. For tasks where the input is a problem instance that can be generated cost-efficiently (e.g., a traveling salesperson problem), models can learn to approximate a solver's solution (Kool et al., 2019; Veličković et al., 2022; Vinyals et al., 2015). More closely related to ours are works that learn to amortize integrating an equation (Lample & Charton, 2020) and to amortize symbolic regression (Biggio et al., 2021), by pretraining large transformers on synthetic equations, and equations and dataset realizations of them, respectively. For tabular ML tasks, we can randomly sample structured causal models to generate synthetic data. Split into features and labels, this yields learning tasks, e.g., a classification problem. Using this data to train a model to output predictions, conditioned on a query point and examples provided in-context (i.e., without any further gradient updates), yields powerful off-the-shelf predictors for real-world tasks. Transformer-based prior-data fitted networks (PFNs) implement this paradigm (Müller et al., 2022) and its most prominent instantiation TabPFN (Hollmann et al., 2023; 2025) performs inference on supervised machine learning tasks in a single forward pass, outperforming traditional per-dataset trained ML models. Furthermore, it has been shown that if trained using cross-entropy PFNs approximate Bayesian inference (Müller et al., 2022). Notably, ShapFM does not perform the same form of in-context learning as TabPFN does (i.e., learning a black-box function from in-context examples) and rather amortizes a mathematical operator to estimate the Shapley value for a single game in-context.

PFNs and Shapley values. Four recent works combine PFNs with Shapley values, two of which use a PFN to evaluate the game. Rundel et al. (2024) explain TabPFN itself and show that its

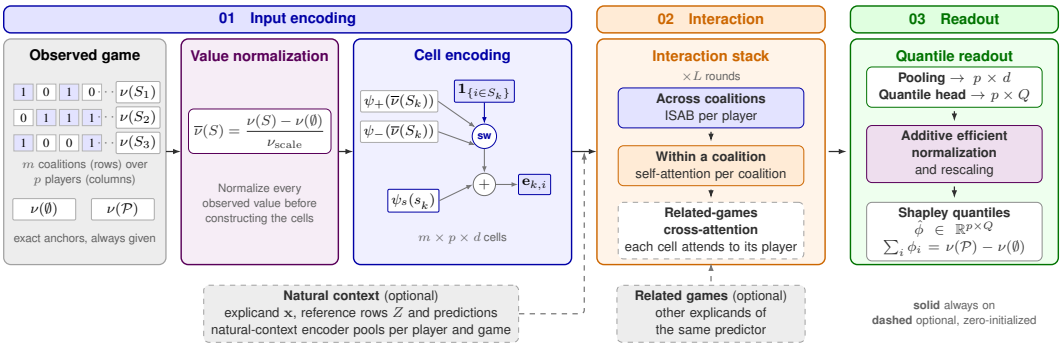

Figure 2: Architecture of ShapFM. The observed game is normalized and encoded as a grid of $m \times p$ cells, L rounds of attention alternate between the coalition and the player axis, and the readout pools each player's column into quantiles that satisfy efficiency by construction. Dashed components are optional and zero-initialized. Section B gives additional architectural details.

in-context learning allows estimating its Shapley values without approximate retraining. Olsen & Christensen (2026) use TabPFN's in-context ability to estimate the conditional expectations that define conditional Shapley values, thus, overcoming costly per-coalition model fitting. Sena & Azevedo (2026) propose ShapPFN, a tabular foundation model whose prediction is additive over features, which yields prediction and its own explanations in one forward pass. ShapPFN is trained on synthetic data from the TabICL prior with a Shapley-kernel-weighted consistency loss on masked inputs. More closely related, ExplainerPFN (Fonseca & Stoyanovich, 2026) predicts attributions from inputs and observed model predictions, conditioned on a reference set of such pairs, without querying feature coalitions at inference. These pairs need not determine the model's coalition values; ExplainerPFN targets a posterior mean attribution under its training prior. ShapFM differs from these works: It replaces the classical estimator, conditions on game-specific observations and works across explicand classes.

2 Amortized Shapley Estimation with a Prior-Fitted Model

ShapFM frames Shapley estimation as learning a mapping from coalition evaluations to Shapley values. A task is a cooperative game ν drawn from a prior over games with exact Shapley values (Section 2.3). We call the game to be explained the *target game*. ShapFM observes m sampled coalitions and their values (evaluations), plus the empty and grand coalition, $D_m = \big(\{(S_k, \nu(S_k))\}_{k=1}^m,\ \nu(\emptyset),\ \nu(\mathcal{P})\big)$. Besides the observations, ShapFM can receive an optional *game context* $\mathcal{C}$, which holds related games of the same family and, for feature attribution, the explicand with reference inputs (Section 2.2). Concretely, ShapFM is a transformer-based neural network

$$\phi_{\theta,\text{ShapFM}} \colon \big(D_m,\ \mathcal{C}\big) \ \mapsto\ \big(\hat{\phi}_i(\tau_q)\big)_{i \in \mathcal{P},\ q=1,\dots,Q}, \tag{2}$$

returning, in one forward pass, Q quantiles of the posterior of every player's Shapley value. Unlike the explainer $\phi_{\theta,\text{FastSHAP}}(\mathbf{x}, y)$ of FastSHAP in (8), which takes an explicand and is fitted to one predictor, ShapFM takes the observations of an arbitrary game. The observations D_m are always provided; game context $\mathcal{C}$ is optional. ShapFM (no context) is the same model with $\mathcal{C} = \emptyset$, seeing exactly what the estimators of Section A see. Hereafter, *context* refers to $\mathcal{C}$.

2.1 Architecture

ShapFM implements (2) in three stages (Figure 2). ShapFM encodes the $m \times p$ observation grid, alternates attention across coalitions and players, then pools each player's cells into quantiles and enforces efficiency. Game context $\mathcal{C}$ enters after encoding and during interaction (Section 2.2).

Encoding of coalitions and values. ShapFM scales each observed value via $\overline{\nu}(S_k) = (\nu(S_k) - \nu(\emptyset))/\nu_{\text{scale}}$ with $\nu_{\text{scale}} = \max_t\big(|\nu(S_t) - \nu(\emptyset)|, |\nu(\mathcal{P}) - \nu(\emptyset)|\big)$ to normalize input. ShapFM's output $\overline{\phi}_i$ is transformed back $\phi_i = \nu_{\text{scale}} \cdot \overline{\phi}_i$ (Section D.1). We set $\nu(\emptyset) = 0$ to simplify notation.

By (5), ϕ_i contrasts coalitions containing player i with those without, weighted by coalition size; ShapFM builds both into each cell embedding. The scaled value is multiplied with a map ψ_+ if $i \in S_k$ (and ψ_- otherwise), with a zero-initialized map ψ_s adding the coalition size $s_k = |S_k|$,

$$\mathbf{e}_{k,i} = \mathbb{1}_{\{i \in S_k\}} \cdot \psi_+(\overline{\nu}(S_k)) + \mathbb{1}_{\{i \notin S_k\}} \cdot \psi_-(\overline{\nu}(S_k)) + \psi_s\big(s_k/p,\ \log(1+s_k)\big). \tag{3}$$

All maps are learned, affine and shared across players and coalitions; neither axis carries a positional encoding (see Section B for further explanation and Table 9 for embedding hyperparameters).

Attention over coalitions and players. The interaction stage alternates between row- and column attention across the cell grid for L rounds. Column attention over $X_i = (\mathbf{e}_{k,i})_{k=1}^{m}$ lets player i compare coalitions containing it with those without, as in (5), but with learned weights. Since the column grows with the budget, we use an induced set attention block (Lee et al., 2019), whose K inducing points summarize the column at a cost linear in m. Row attention exchanges information across players of a coalition as a coalition's value is shared across its members. We start with column attention as initially a row's cells differ only in their membership bit, so the players could exchange nothing but the coalition size.

Alternating the two axes lets information pass between players and coalitions in every round, which improves over a sequential stack in our ablation (Section E.5). The column attention and the readout use the query-aware scaling of TabICLv2 (Qu et al., 2026) for budgets beyond pretraining (Section B).

Readout and exact efficiency. Multi-head attention pooling (Lee et al., 2019) with a learned seed query reduces each column X_i to one vector, which a linear layer maps to Q raw quantiles $\hat{\phi}_i(\tau_q)$ at levels evenly spaced in $(0,1)$; $\hat{\phi}_i^{\text{raw}}$ is their mean. No earlier stage guarantees *efficiency*, so ShapFM applies the additive efficient normalization of (9), shifting every quantile of every player equally,

$$\hat{\phi}_i(\tau_q) \leftarrow \hat{\phi}_i(\tau_q) + \frac{1}{p}\Big(\overline{\nu}(\mathcal{P}) - \sum_{i' \in \mathcal{P}} \hat{\phi}_{i'}^{\text{raw}}\Big). \tag{4}$$

The point estimate $\hat{\phi}_i$ is the mean of its shifted quantiles and therefore satisfies efficiency by construction, also after rescaling. Since (4) spreads the gap equally over the players regardless of where it arises, we additionally penalize the gap before normalization during training (Section D.1).

2.2 Adding Game Context to ShapFM

ShapFM accepts two optional inputs: related games with the same number of players (e.g., other explicands of one predictor) and, for feature attribution, the explicand with reference inputs. Separate zero-initialized, player-equivariant modules process them; without either input, the model reduces to Section 2.1. We call ShapFM with both modules the *context model*; Figure 3 previews its gains, and Section B.1 gives details.

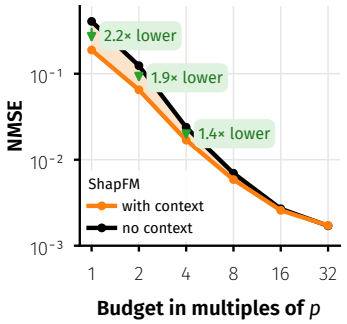

Figure 3: Mean NMSE over 40 games. Context lowers the error most in low-budget regimes (see Figures 4 and 11).

Related games supply their membership matrices and coalition values (without Shapley labels), and pass through network like the target games. After the player attention of each round, every target cell (S_k, i) uses cross-attention on the cells of player i in the related games. As ShapFM treats related games as a set, their order does not affect the prediction. Every game family in the prior provides related games, and during training we vary their number and, with a fixed probability, replace one of them by an unrelated game, to also handle no or misleading related games (Section D.1); Section E.5 shows where this falls short.

Natural context. When the players are features of a data set a predictor f uses, ShapFM can also read the features themselves, which identifies players across related games. An encoder attends over rows and players to turn the explicand and R background rows, each with its prediction, into one vector u_i per player and a game vector u; both are projected onto every cell embedding of player i in the beginning. Only the TabICL TreeSHAP and the Interventional games on causal models families provide this input (see Section 2.3.1).

2.3 Generating a Synthetic Prior for ShapFM

Our prior consists of four families of related games, and every family in our prior returns ϕ exactly, either in closed form or through an exact algorithm. We use only exact targets so ShapFM does not learn estimator errors, inherit their accuracy ceiling, or mistake target noise for uncertainty about the game. Estimated targets would admit more prior families; we leave this trade-off to future work.

We generate training data in groups of related games, following TabICLv2 (Qu et al., 2026). Each batch first draws one band of the budget distribution π_m, which is a mixture over four bands (Table 7), with half of its mass on budgets just above the number of players, where amortization matters most, and a tail up to 512 coalitions. For each group, we draw a number of players p uniformly from $\{6, \dots, 32\}$, a budget $m \sim \pi_m$ within the band, and a prior family $\kappa \sim \pi_{\text{fam}}$ with hyperparameters η_κ. The family draws a structure that the whole group shares, such as the interaction terms of a sum of unanimity games or the fitted predictor of an explanation game, and then draws G games from this structure, each with its own random seed. Thus, the games of a group are related in the sense of Section 2.2; Section C gives the per-family sharing scheme. Across games in a group player i has the same meaning, which the attention over related games relies on. We stack $n_{\text{grp}} = 4$ groups of $G = 8$ related games into a batch of $B = 32$ games (Section C.1).

Depending on the family, a group may receive a dummy player and a pair of symmetric players, each with a fixed probability, so that both axioms appear in the training data. Both are drawn once per group, so all games of a group share their dummy and symmetric players (Section C.1). We redraw a game whose value function is constant or whose grand and empty coalitions have nearly the same value (Section C.1). For each game we draw m training coalitions independently by choosing a coalition size uniformly from $\{1, \dots, p-1\}$ and then a coalition of that size uniformly, so the games of a group observe different coalitions. This sampling law matches the size distribution of leverage score sampling (Musco & Witter, 2025).

2.3.1 Prior Families

Each group draws its family uniformly from four families.[1] Two families, the sum of unanimity games and the Gaussian process on the hypercube, specify an interaction structure, giving control over interaction order, player heterogeneity and the size profile. Two families, TabICL TreeSHAP and the interventional games on causal models, induce a game from a mechanism and inherit whatever structure it produces. The two mechanism families cover the two standard semantics of the value function in feature attribution (Section C) and are the only ones that provide natural context, explaining many rows of one model as is typical in practice.

Sum of unanimity games. (Section C.3) We draw games directly in the Harsanyi basis with value function $\nu(S) = c_\emptyset + \sum_{T \in \mathcal{T}} c_T \, \mathbb{1}_{\{T \subseteq S\}}$ and Shapley values $\phi_i = \sum_{T \ni i} \frac{c_T}{|T|}$. $\mathcal{T} \subseteq 2^{\mathcal{P}}$ holds the non-empty subsets on which the unanimity games live, so a subset with $|T| \geq 2$ contributes a real higher-order interaction. We draw $\mathcal{T}$ in one to three blocks with their own number of subsets, upper bound on $|T|$, and coefficient scale, so training covers different interaction complexity and target magnitude.

Gaussian process on the hypercube. (Section C.4) We place a zero-mean Gaussian process prior on the value function, $\nu \sim \mathcal{GP}(0, k)$, with the Hamming kernel $k(S, S') = \sigma^2 \prod_{i \in \mathcal{P}} \exp\big(-\mathbb{1}_{\{i \in S\} \neq \{i \in S'\}} / \ell_i\big)$ (Rundel et al., 2026): covariance shrinks for every player that the two coalitions do not share, the more so the shorter its lengthscale ℓ_i. All coalition values are then jointly Gaussian, and since the Shapley values are linear in them, the vector of the p Shapley values and the m context coalition values is jointly Gaussian with a covariance that is available in time polynomial in p (Mohammadi et al., 2025a; Rundel et al., 2026). Sampling this vector yields a game with exact Shapley values without enumerating the coalitions, which scales to larger p than an enumeration would allow. A plain sample path, however, has no trend and spreads its variance over many weak interactions, so we add a concave size-scaled mean and Möbius atoms with closed-form Shapley values, keeping the targets exact while making the games more realistic (Section C.4).

[1] As the interventional games on causal models are limited to 16 players, groups with more players draw from the other three.

TabICL TreeSHAP. (Section C.5) Games are local explanations of a fitted predictor. Given a predictor f, an explicand $\mathbf{x}$, and a background sample $\{\mathbf{z}_r\}_{r=1}^R$, we use the interventional game $\nu(S) = \frac{1}{R}\sum_{r=1}^R f\big(\mathbf{x}_S, \mathbf{z}_{r,\mathcal{P}\setminus S}\big)$, where the features in S are taken from $\mathbf{x}$ and the remaining features from the background row.

The predictor is a small tree ensemble fitted on a synthetic tabular dataset from the TabICLv2 prior (Qu et al., 2026), and interventional TreeSHAP in `shapiq` (Muschalik et al., 2024), run with the same background sample, returns the exact Shapley values of this game. The ensemble type, size and depth, the numbers of dataset and background rows, and the task type vary across the prior, and features that the ensemble never splits on are exact dummies.

Interventional games on causal models. (Section C.6) These are Do-Shapley games of Witter et al. (2026). The players are the feature nodes $X_1, \ldots, X_p$ of a random directed acyclic graph with linear Gaussian mechanisms and a single sink Y, and the value of a coalition is the expected outcome after intervening on its members at a fixed explicand $\mathbf{x}$, $\nu(S) = \mathbb{E}\big[Y \mid \mathrm{do}(X_S = \mathbf{x}_S)\big]$, where every feature is a parent of Y, so direct effects mix with mediated ones that intervening on the mediator blocks. Under linear mechanisms every coalition value is exact, so we enumerate all 2^p coalitions and (1) yields ϕ, which limits this family to 16 players. The edge density, the edge weights, and the explicand vary across the prior, and a feature node without edges is an exact dummy.

3 Experiments

We evaluate ShapFM on cooperative games from machine learning with *exact* Shapley values (see Table 10 and Section E.2) and report the NMSE $\|\hat{\phi} - \phi\|^2/\|\phi\|^2$ as the performance metric. The games range from explanations of tabular, image and text models to hyperparameter importance (HPI; Wever et al., 2026) and synthetic games (Tsai et al., 2023), beyond the synthetic prior of ShapFM. We compare ShapFM with state-of-the-art *sample-based estimators* in Section 3.1 and with *amortized explainers* in Section 3.2. ShapFM is trained only on synthetic games, and a single pretrained checkpoint is applied to *every* game and budget. We give ShapFM *natural context* (32 reference rows with their predictions) and seven *related games* as additional context (orange) when a game provides them (Section 2.2). ShapFM (no context) sees only the evaluations of the target game (D_m). For more details on the empirical evaluation and additional results, we refer to Section E.

3.1 Amortizing Shapley Explanations with a Prior Fitted Model

We first compare ShapFM with sample-based estimators, which estimate every game from scratch. These are LeverageSHAP (cyan; Musco & Witter, 2025), permutation sampling (brown; Castro et al., 2009), SVARM (pink; Kolpaczki et al., 2024), OddSHAP (purple; Fumagalli et al., 2026a), RegressionMSR with an XGBoost proxy (blue; Witter et al., 2025) and ShaplEIG (green; Rundel et al., 2026), all as implemented in `shapiq` (Muschalik et al., 2024). Table 1 and Figure 4 report the error per game family in the **low-budget approximation regime**.

ShapFM is most accurate where evaluations are *scarce*. At $p+2$ evaluations, it attains the lowest error on the game families of TabPFN, random forests and XGBoost, 1.5 to 3.6 times lower than the best estimator. Without context, ShapFM still outperforms every estimator on most of the game families where context is not available. ShaplEIG is more accurate on hyperparameter importance. **Context consistently improves the performance of ShapFM** (see also Figures 3 and 11). Averaged over all game families, ShapFM ranks first with context and second without, ahead of every estimator. Several estimators exceed an NMSE of one at this budget, which is worse than predicting zero for every player. ShapFM remains the most accurate method up to $8p$ evaluations (Figures 8 and 9), except for ShaplEIG on hyperparameter importance and SOUM, at more than a thousand times the runtime of ShapFM (Section 3.4). Accurate estimates from a budget just above the number of players are new territory for

Table 1: NMSE at a budget of $p+2$ evaluations, averaged over the n game families of each row (all game families in Tables 11 and 12). Gold, silver and bronze mark the first, second and third best method.

		ShapFM		Traditional estimators					
Game families	n	w/ ctx.	no ctx.	EIG	Lev.	Perm.	RMSR	SVARM	Odd
TabPFN	25	.151	.375	–	.545	.951	.813	1.10	1.79
Random forest	10	.304	.406	–	.536	1.16	.776	1.08	1.81
XGBoost	10	.338	.428	–	.520	.832	.772	.971	1.45
Image clf.	5	–	.530	1.11	1.05	2.72	1.47	2.35	3.60
Sentiment	1	–	.551	.858	1.65	3.76	1.13	1.23	1.97
HPI	4	–	.235	.132	.388	.237	1.14	1.84	3.22
SOUM	4	–	.548	.887	1.31	2.29	1.18	.881	2.34
Mean rank	59	1.09	2.10	2.14	3.41	4.41	4.56	5.44	6.73

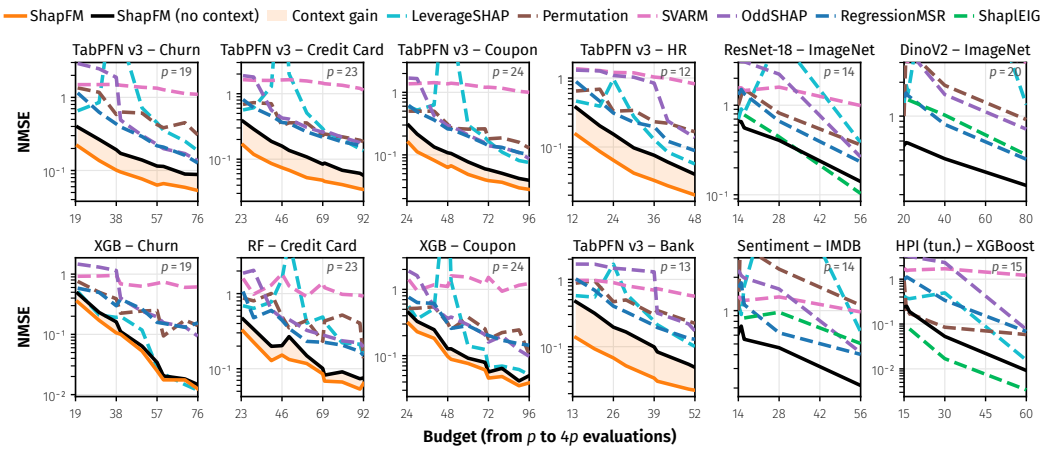

Figure 4: Approximation quality of ShapFM and classical estimators on a collection of different game families. NMSE over the budget, averaged over explained points, for ShapFM with context and without. The shaded area is the gain from context. Larger budgets and more games are provided in Figures 8 and 9.

estimators. So far, only amortized explainers such as FastSHAP (Jethani et al., 2022) reach this regime. Unlike ShapFM, they require a separate training run for every model and setup.

3.2 Comparison with Amortized Explainers

We now compare ShapFM with two amortized explainers. FastSHAP (gray; Jethani et al., 2022) trains one explainer network per model (i.e., game family), and ExplainerPFN (green; Fonseca & Stoyanovich, 2026) adapts TabPFNv2 on synthetic datasets to explain any model from its data and predictions alone. Neither evaluates the game on any coalition when it explains a point. We explain a random forest and an MLP on Adult Census, Bank Marketing and Churn, each on 200 test points (i.e., games) with exact Shapley values. Since FastSHAP learns from sampled coalition values, it can also be trained on the estimates of ShapFM instead, which we call FastSHAP (ShapFM targets, light orange). Section E.4 describes in detail how we train and evaluate all three explainers.

ShapFM matches amortized explainers without per-model training. ShapFM reaches the error of FastSHAP with about $1p$ to $3p$ evaluations per explanation for every model and dataset (Figure 5). With more evaluations, ShapFM keeps improving and becomes 14 to 180 times more accurate at 256 evaluations, whereas the error of FastSHAP is *fixed* once it is trained. FastSHAP must also be retrained for every new model or dataset, and it is unclear how it can be extended beyond local feature attribution to games such as HPI or SOUM. Interestingly, **the estimates of ShapFM are a better training signal than the objective of FastSHAP.** Trained on them, FastSHAP becomes 2.1 to 6.2 times more accurate with fewer evaluations during training, so both approaches combine well when many points of one model must be explained. ExplainerPFN, which never evaluates the model on coalitions, stays close to an NMSE of one, the error of predicting zero for every player, since data and predictions alone do not determine the Shapley values of a specific model. LeverageSHAP overtakes ShapFM only from 512 to 1024 evaluations.

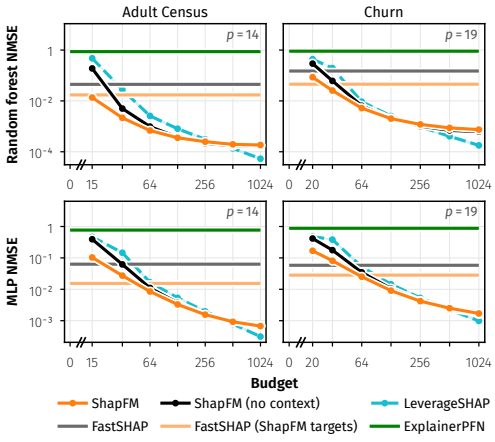

Figure 5: Comparison of ShapFM with FastSHAP and ExplainerPFN (detailed view in Figure 10).

3.3 Ablation of ShapFM Components

We conduct an ablation study to identify which design choices of ShapFM matter. We retrain ShapFM on the synthetic prior without one design choice at a time, each for 10,000 steps at three seeds. We then evaluate every variant on the TabPFN and XGBoost games and compare its error with the unchanged model trained the same way (Figure 6 and Section E.5). **Related games and alternating attention matter most.** Without any context, the error rises up to 2.4 times on TabPFN games, and removing only the related games accounts for most of this rise. The natural context adds little once related games are given. Running all coalition attention before all player attention raises the error up to 2 times on XGBoost games. The cell encoding of (3) helps as well, since a flat encoding that passes the membership as an input feature raises the error up to 1.2 times at larger budgets. Interestingly, context helps where evaluations are scarce, and from $16p$ evaluations on, the model without context is more accurate.

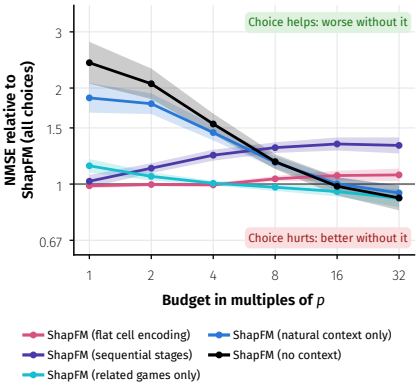

Figure 6: Effect of removing or changing one design choice of ShapFM on 20 TabPFN games (XGBoost games and training curves in Figures 12 and 13).

3.4 Runtime Comparison

Finally, we measure the runtime per explanation of ShapFM and baselines (see Figure 7) on the same datasets used in Section 3.2. ShapFM's inference runtime without context is on par with the fastest estimators. With context, ShapFM remains comparable to traditional estimators with the same budget. ShaplEIG, the strongest estimator in the low-budget regime, is orders of magnitude slower, as it selects coalitions sequentially in an active learning manner, whereas ShapFM and the remaining estimators draw all coalitions simultaneously. FastSHAP is the fastest method at inference, excluding its costly training.

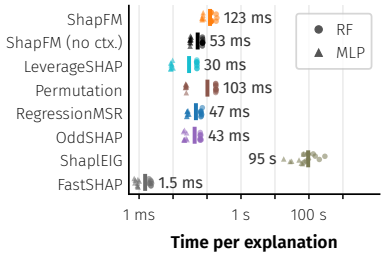

Figure 7: Explanation time (including model calls) at a budget of $8p$.

4 Conclusion

We introduced ShapFM, a prior fitted model that amortizes Shapley value estimation across cooperative games. Pretrained only on synthetic games with exact Shapley values, ShapFM estimates the Shapley values of unseen games in a single forward pass, without task-specific training. It performs best in settings where evaluations are scarce, outperforming state-of-the-art estimators at budgets just above the number of players, and unlike amortized explainers, it keeps improving with more evaluations. Its ability to use additional context, such as related games or the explained model's data, opens many avenues for future work.

5 Limitations and Future Work

ShapFM is pretrained on games with up to 32 players, and its accuracy depends on how well the synthetic prior covers the games it is applied to. At larger budgets, its advantage shrinks and the estimators eventually become more accurate. Our evaluation is restricted to games with exact Shapley values, which limits the range of game sizes and domains we can test. Future work could scale ShapFM to games with many more players, broaden its prior with further game families and forms of context, and study how well its predicted quantiles capture the uncertainty of an estimate.

AI USE STATEMENT

We used large language model assistants, e.g., Claude Code, for the following tasks with required disclosure. They helped implement parts of our code base, including the training and evaluation pipelines and the orchestration of experiments on our compute cluster. They gave feedback on the design of the experiments and suggestions on parts of the method, which the authors evaluated and decided on. They also helped writing scripts to compute summary statistics from the logged results and describe them in the text. Our synthetic training games are sampled by the prior of Section 2.3, which is implemented in our own code with the help of AI tools as described above, and no data set, game or Shapley value was produced by a generative AI model. Tasks related to mathematical proofs, translation and qualitative data analysis are not applicable to this work. Among the tasks with recommended disclosure, we used generative AI tools to create and edit software code, to create figures and tables from the logged results, to draft and edit parts of the paper for readability and to search the literature. We have reviewed all AI-assisted work. The authors reviewed all code, which is covered by unit tests, and every target is checked against exact Shapley values, including the efficiency of every game and the exact solver of `shapiq`. Every figure and table is generated by scripts directly from the logged experiment results, and the authors checked every number in the text against these results. We take full responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work proposes a method to estimate Shapley values for explaining machine learning models and involves no human subjects or personal data. ShapFM is pretrained only on synthetic games. Our evaluation uses public benchmark datasets, some of which contain sensitive attributes, such as Adult Census. We use them only to construct explanation games and draw no conclusions about individuals or groups. Like any estimator, ShapFM can return inaccurate attributions, in particular for games far from its prior (Section 5), and explanations trusted without scrutiny can lend unwarranted credibility to a model. We therefore encourage practitioners to verify important attributions with additional evaluations of the game. By making explanations cheaper, ShapFM may also lower the cost of auditing models.

REPRODUCIBILITY STATEMENT

We aim to make all results of this paper reproducible. Section 2 describes the architecture, the game context and the synthetic prior of ShapFM, and Sections B and C complete it with the context modules, the construction of related games and the hyperparameters of every prior family. Section D specifies the training objective, optimizer and schedule, and Section D.2 documents the hyperparameter search and its search space. Every training batch is drawn from the prior by seeded random number generators and can therefore be regenerated exactly. For the experiments, Section E.1 describes the budgets, the cost of the context, the estimators and the aggregation, Section E.2 describes all cooperative games and how their exact Shapley values are computed, and Section E.4 lists the training settings of the amortized explainers. All estimators run as implemented in `shapiq` (version 1.7.0; Muschalik et al., 2024) with their default settings. We provide the source code of the prior, the training and the evaluation as anonymized supplementary material. The supplementary material also contains the final pretrained checkpoint of ShapFM, so that all experiments can be reproduced without pretraining the model again. We will release both upon publication.

REFERENCES

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020. doi: 10.1016/j.inffus.2019.12.012. URL https://doi.org/10.1016/j.inffus.2019.12.012.

Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo. Neural Symbolic Regression that scales. In Marina Meila and Tong Zhang (eds.), *Pro-*

ceedings of the 38th International Conference on Machine Learning, volume 139 of *Proceedings of Machine Learning Research*, pp. 936–945. PMLR, 2021. URL https://proceedings.mlr.press/v139/biggio21a.html.

Bernd Bischl, Giuseppe Casalicchio, Taniya Das, Matthias Feurer, Sebastian Fischer, Pieter Gijsbers, Subhaditya Mukherjee, Andreas C. Müller, László Németh, Luis Oala, Lennart Purucker, Sahithya Ravi, Jan N. van Rijn, Prabhant Singh, Joaquin Vanschoren, Jos van der Velde, and Marcel Wever. OpenML: Insights from 10 years and more than a thousand papers. *Patterns*, 6: 101317, 2025. doi: 10.1016/j.patter.2025.101317. URL https://doi.org/10.1016/j.patter.2025.101317.

Landon Butler, Abhineet Agarwal, Justin Kang, Yigit Efe Erginbas, Bin Yu, and Kannan Ramchandran. ProxySPEX: Inference-Efficient Interpretability via Sparse Feature Interactions in LLMs. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 72306–72340. Curran Associates, Inc., 2025. doi: 10.52202/085713-2426. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/68dced1638d71d4d21598f79ae91262e-Paper-Conference.pdf.

Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the Shapley value based on sampling. *Computers & Operations Research*, 36:1726–1730, 2009. doi: 10.1016/j.cor.2008.04.004. URL https://doi.org/10.1016/j.cor.2008.04.004.

Javier Castro, Daniel Gómez, Elisenda Molina, and Juan Tejada. Improving polynomial estimation of the Shapley value by stratified random sampling with optimum allocation. *Computers & Operations Research*, 82:180–188, 2017. doi: 10.1016/j.cor.2017.01.019. URL https://doi.org/10.1016/j.cor.2017.01.019.

A. Charnes, B. Golany, M. Keane, and J. Rousseau. *Extremal Principle Solutions of Games in Characteristic Function Form: Core, Chebychev and Shapley Value Generalizations*, pp. 123–133. Advanced Studies in Theoretical and Applied Econometrics. Springer Netherlands, 1988. ISBN 9789400936775. doi: 10.1007/978-94-009-3677-5_7. URL https://doi.org/10.1007/978-94-009-3677-5_7.

Siu Lun Chau, Krikamol Muandet, and Dino Sejdinovic. Explaining the Uncertain: Stochastic Shapley Values for Gaussian Process Models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 50769–50795. Curran Associates, Inc., 2023. doi: 10.52202/075280-2209. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/9f0b1220028dfa2ee82ca0a0e0fc52d1-Paper-Conference.pdf.

Hugh Chen, Joseph D. Janizek, Scott Lundberg, and Su-In Lee. True to the Model or True to the Data? *arXiv:2006.16234v1 [cs.LG]*, 2020. URL http://arxiv.org/abs/2006.16234v1.

Hugh Chen, Ian C. Covert, Scott M. Lundberg, and Su-In Lee. Algorithms to estimate Shapley value feature attributions. *Nature Machine Intelligence*, 5:590–601, 2023. doi: 10.1038/s42256-023-00657-x. URL https://doi.org/10.1038/s42256-023-00657-x.

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*, pp. 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by Removing: A Unified Framework for Model Explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021. URL http://jmlr.org/papers/v22/20-1316.html.

Ian Covert, Chanwoo Kim, Su-In Lee, James Zou, and Tatsunori Hashimoto. Stochastic Amortization: A Unified Approach to Accelerate Feature and Data Attribution. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37,

pp. 4374–4423. Curran Associates, Inc., 2024. doi: 10.52202/079017-0143. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/084252ec5f05f13bf565843c1873686d-Paper-Conference.pdf.

Ian Connick Covert, Chanwoo Kim, and Su-In Lee. Learning to Estimate Shapley Values with Vision Transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=5ktFNz_pJLK.

Xiaotie Deng and Christos H. Papadimitriou. On the Complexity of Cooperative Solution Concepts. *Mathematics of Operations Research*, 19:257–266, 1994. doi: 10.1287/moor.19.2.257. URL https://doi.org/10.1287/moor.19.2.257.

Finale Doshi-Velez and Been Kim. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv:1702.08608v2 [stat.ML]*, 2017. URL http://arxiv.org/abs/1702.08608v2.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

James Enouen and Yan Liu. InstaSHAP: Interpretable Additive Models Explain Shapley Values Instantly. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 22720–22765, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/38fc6254f73450813db3b3e04397a9fc-Paper-Conference.pdf.

Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Desai, David Salinas, and Frank Hutter. TabArena: A Living Benchmark for Machine Learning on Tabular Data. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-0519. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/1697e3fb412da11dc9488249f9e7bbc9-Paper-Datasets_and_Benchmarks_Track.pdf.

Joao Fonseca and Julia Stoyanovich. ExplainerPFN: Towards tabular foundation models for model-free zero-shot feature importance estimations. *arXiv:2601.23068v2 [cs.LG]*, 2026. URL http://arxiv.org/abs/2601.23068v2.

Fabian Fumagalli, Maximilian Muschalik, Eyke Hüllermeier, Barbara Hammer, and Julia Herbinger. Unifying Feature-Based Explanations with Functional ANOVA and Cooperative Game Theory. In Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Emtiyaz Khan (eds.), *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 5140–5148. PMLR, 2025. URL https://proceedings.mlr.press/v258/fumagalli25a.html.

Fabian Fumagalli, Landon Butler, Justin Singh Kang, Kannan Ramchandran, and R. Teal Witter. An Odd Estimator for Shapley Values. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=xILwgiWAUk.

Fabian Fumagalli, R. Teal Witter, and Christopher Musco. PolySHAP: Extending KernelSHAP with Interaction-Informed Polynomial Regression. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 136065–136099, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/dc4db30d695c08219a6b904c39dd605a-Paper-Conference.pdf.

Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining Explanations: An Overview of Interpretability of Machine Learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 80–89. IEEE, 2018. doi: 10.1109/dsaa.2018.00018. URL https://doi.org/10.1109/dsaa.2018.00018.

Michel Grabisch. *Set Functions, Games and Capacities in Decision Making*. Springer International Publishing, 2016. ISBN 9783319306902. doi: 10.1007/978-3-319-30690-2. URL https://doi.org/10.1007/978-3-319-30690-2.

John C. Harsanyi. A Simplified Bargaining Model for the n-Person Cooperative Game. *International Economic Review*, 4:194–220, 1963. doi: 10.2307/2525487. URL https://doi.org/10.2307/2525487.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. IEEE, 2016. doi: 10.1109/cvpr.2016.90. URL https://doi.org/10.1109/cvpr.2016.90.

Tom Heskes, Evi Sijben, Ioan Gabriel Bucur, and Tom Claassen. Causal Shapley Values: Exploiting Causal Knowledge to Explain Individual Predictions of Complex Models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4778–4789. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/32e54441e6382a7fbacbbbaf3c450059-Paper.pdf.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025. doi: 10.1038/s41586-024-08328-6. URL https://doi.org/10.1038/s41586-024-08328-6.

Dominik Janzing, Lenon Minorics, and Patrick Bloebaum. Feature relevance quantification in explainable AI: A causal problem. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2907–2916. PMLR, 2020. URL https://proceedings.mlr.press/v108/janzing20a.html.

Neil Jethani, Mukund Sudarshan, Ian Connick Covert, Su-In Lee, and Rajesh Ranganath. FastSHAP: Real-Time Shapley Value Estimation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Zq2G_VTV53T.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J. Spanos. Towards Efficient Data Valuation Based on the Shapley Value. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 1167–1176. PMLR, 2019. URL https://proceedings.mlr.press/v89/jia19a.html.

Keller Jordan. Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/, 2024. URL https://kellerjordan.github.io/posts/muon/.

Yonghan Jung, Shiva Kasiviswanathan, Jin Tian, Dominik Janzing, Patrick Bloebaum, and Elias Bareinboim. On Measuring Causal Contributions via do-interventions. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10476–10501. PMLR, 2022. URL https://proceedings.mlr.press/v162/jung22a.html.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,

2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

Patrick Kolpaczki, Viktor Bengs, Maximilian Muschalik, and Eyke Hüllermeier. Approximating the Shapley Value without Marginal Contributions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13246–13255. Association for the Advancement of Artificial Intelligence (AAAI), 2024. doi: 10.1609/aaai.v38i12.29225. URL https://doi.org/10.1609/aaai.v38i12.29225.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, Learn to Solve Routing Problems! In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ByxBFsRqYm.

Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Guillaume Lample and François Charton. Deep Learning For Symbolic Mathematics. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1eZYeHFDS.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2:56–67, 2020. doi: 10.1038/s42256-019-0138-9. URL https://doi.org/10.1038/s42256-019-0138-9.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea (eds.), *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015/.

Rory Mitchell, Joshua Cooper, Eibe Frank, and Geoffrey Holmes. Sampling Permutations for Shapley Value Estimation. *Journal of Machine Learning Research*, 23(43):1–46, 2022. URL http://jmlr.org/papers/v23/21-0439.html.

Majid Mohammadi, Siu Lun Chau, and Krikamol Muandet. Amortized Linear-time Exact Shapley Value for Product-Kernel Methods. *arXiv:2505.16516v3 [cs.LG]*, 2025a. URL http://arxiv.org/abs/2505.16516v3.

Majid Mohammadi, Krikamol Muandet, Ilaria Tiddi, Annette Ten Teije, and Siu Lun Chau. Exact Shapley Attributions in Quadratic-time for FANOVA Gaussian Processes. *arXiv:2508.14499v1 [cs.LG]*, 2025b. URL http://arxiv.org/abs/2508.14499v1.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers Can Do Bayesian Inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Maximilian Muschalik, Hubert Baniecki, Fabian Fumagalli, Patrick Kolpaczki, Barbara Hammer, and Eyke Hüllermeier. shapiq: Shapley Interactions for Machine Learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and

C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 130324–130357. Curran Associates, Inc., 2024. doi: 10.52202/079017-4141. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/eb3a9313405e2d4175a5a3cfcd49999b-Paper-Datasets_and_Benchmarks_Track.pdf.

Maximilian Muschalik, Fabian Fumagalli, Paolo Frazzetto, Janine Strotherm, Luca Hermes, Alessandro Sperduti, Eyke Hüllermeier, and Barbara E Hammer. Exact Computation of Any-Order Shapley Interactions for Graph Neural Networks. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 36295–36337, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/59b9582cd35f555ea8415030073e7b22-Paper-Conference.pdf.

Christopher Musco and R. Teal Witter. Provably Accurate Shapley Value Estimation via Leverage Score Sampling. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 64532–64559, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/a224ff18cc99a71751aa2b79118604da-Paper-Conference.pdf.

Lars Henry Berge Olsen and Dennis Christensen. Computing Conditional Shapley Values Using Tabular Foundation Models. *arXiv:2602.09489v1 [cs.AI]*, 2026. URL http://arxiv.org/abs/2602.09489v1.

Miles Olson, Elizabeth Santorella, Louis C. Tiao, Sait Cakmak, Mia Garrard, Samuel Daulton, Zhiyuan Jerry Lin, Sebastian Ament, Bernard Beckerman, Eric Onofrey, Paschal Igusti, Cristian Lara, Benjamin Letham, Cesar Cardoso, Shiyun Sunny Shen, Andy Chenyuan Lin, Matthew Grange, Elena Kashtelyan, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Ax: A Platform for Adaptive Experimentation. In Leman Akoglu, Carola Doerr, Jan N. van Rijn, Roman Garnett, and Jacob R. Gardner (eds.), *Proceedings of the Fourth International Conference on Automated Machine Learning*, volume 293 of *Proceedings of Machine Learning Research*, pp. 21/1–25. PMLR, 2025. URL https://proceedings.mlr.press/v293/olson25a.html.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Florian Pfisterer, Lennart Schneider, Julia Moosbauer, Martin Binder, and Bernd Bischl. YAHPO Gym - An Efficient Multi-Objective Multi-Fidelity Benchmark for Hyperparameter Optimization. In Isabelle Guyon, Marius Lindauer, Mihaela van der Schaar, Frank Hutter, and Roman Garnett (eds.), *Proceedings of the First International Conference on Automated Machine Learning*, volume 188 of *Proceedings of Machine Learning Research*, pp. 3/1–39. PMLR, 2022. URL https://proceedings.mlr.press/v188/pfisterer22a.html.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A Better, Faster, Scalable, and Open Tabular Foundation Model. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SxsyLjIfWB.

Benedek Rozemberczki, Lauren Watson, Péter Bayer, Hao-Tsung Yang, Olivér Kiss, Sebastian Nilsson, and Rik Sarkar. The Shapley Value in Machine Learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pp. 5572–5579. International Joint Conferences on Artificial Intelligence Organization, 2022. doi: 10.24963/ijcai.2022/778. URL https://doi.org/10.24963/ijcai.2022/778.

Luis M. Ruiz, Federico Valenciano, and Jose M. Zarzuelo. The Family of Least Square Values for Transferable Utility Games. *Games and Economic Behavior*, 24:109–130, 1998. doi: 10.1006/game.1997.0622. URL https://doi.org/10.1006/game.1997.0622.

David Rundel, Julius Kobialka, Constantin von Crailsheim, Matthias Feurer, Thomas Nagler, and David Rügamer. Interpretable Machine Learning for TabPFN. In *Explainable Artificial Intelligence*, Communications in Computer and Information Science, pp. 465–476. Springer Nature Switzerland, 2024. ISBN 9783031637971. doi: 10.1007/978-3-031-63797-1_23. URL https://doi.org/10.1007/978-3-031-63797-1_23.

David Rundel, Fabian Fumagalli, Maximilian Muschalik, Bernd Bischl, and Matthias Feurer. ShaplEIG: Bayesian Experimental Design for Shapley Value Estimation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ub9PwBtHqD.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115:211–252, 2015. doi: 10.1007/s11263-015-0816-y. URL https://doi.org/10.1007/s11263-015-0816-y.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv:1910.01108v4 [cs.CL]*, 2019. URL http://arxiv.org/abs/1910.01108v4.

Luan Borges Teodoro Reis Sena and Francisco Galuppo Azevedo. [Short] Real-Time Explanations for Tabular Foundation Models. In *ICLR 2026 Workshop on Foundation Models for Science: Real-World Impact and Science-First Design*, 2026. URL https://openreview.net/forum?id=StSMBSZqxx.

L. S. Shapley. *A Value for n-Person Games*, pp. 307–318. Princeton University Press, 1953. ISBN 9781400881970. doi: 10.1515/9781400881970-018. URL https://doi.org/10.1515/9781400881970-018.

Joshua Stiller, Santo M. A. R. Thies, Felix Czaja, and Eyke Hüllermeier. OperatorSHAP: Fast and Accurate Shapley Value Estimation for Neural Operators. *arXiv:2606.28065v1 [cs.LG]*, 2026. URL http://arxiv.org/abs/2606.28065v1.

Erik Štrumbelj and Igor Kononenko. An Efficient Explanation of Individual Classifications using Game Theory. *Journal of Machine Learning Research*, 11(1):1–18, 2010. URL http://jmlr.org/papers/v11/strumbelj10a.html.

Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41:647–665, 2014. doi: 10.1007/s10115-013-0679-x. URL https://doi.org/10.1007/s10115-013-0679-x.

Santo M. A. R. Thies, Hubert Baniecki, R. Teal Witter, Eyke Hüllermeier, Maximilian Muschalik, and Fabian Fumagalli. Proxy-Based Approximation of Shapley and Banzhaf Interactions. *arXiv:2605.22738v2 [cs.LG]*, 2026. URL http://arxiv.org/abs/2605.22738v2.

Che-Ping Tsai, Chih-Kuan Yeh, and Pradeep Ravikumar. Faith-Shap: The Faithful Shapley Interaction Index. *Journal of Machine Learning Research*, 24(94):1–42, 2023. URL http://jmlr.org/papers/v24/22-0202.html.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. The CLRS Algorithmic Reasoning Benchmark. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22084–22102. PMLR, 2022. URL https://proceedings.mlr.press/v162/velickovic22a.html.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer Networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/29921001f2f04bd3baee84a12e98098f-Paper.pdf.

Jiachen T. Wang, Prateek Mittal, and Ruoxi Jia. Efficient Data Shapley for Weighted Nearest Neighbor Algorithms. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li (eds.), *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 2557–2565. PMLR, 2024. URL https://proceedings.mlr.press/v238/wang24e.html.

Jiachen (Tianhao) Wang, Yuqing Zhu, Yu-Xiang Wang, Ruoxi Jia, and Prateek Mittal. A Privacy-Friendly Approach to Data Valuation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 60429–60467. Curran Associates, Inc., 2023. doi: 10.52202/075280-2640. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/bdd5522a32b3a959a6d81fb6ddc1cb38-Paper-Conference.pdf.

Marcel Wever, Maximilian Muschalik, Fabian Fumagalli, and Marius Lindauer. HyperSHAP: Shapley Values and Interactions for Explaining Hyperparameter Optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 26867–26875. Association for the Advancement of Artificial Intelligence (AAAI), 2026. doi: 10.1609/aaai.v40i32.39898. URL https://doi.org/10.1609/aaai.v40i32.39898.

R. Teal Witter, Yurong Liu, and Christopher Musco. Regression-adjusted Monte Carlo Estimators for Shapley Values and Probabilistic Values. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 21209–21240. Curran Associates, Inc., 2025. doi: 10.52202/085713-0715. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/1e77af93008ee6cd248a31723ce357d8-Paper-Conference.pdf.

R. Teal Witter, Álvaro Parafita, Tomas Garriga, Maximilian Muschalik, Fabian Fumagalli, Axel Brando, and Lucas Rosenblatt. Exactly Computing do-Shapley Values. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Peim0KY6ty.

Chenghao Yang, Fan Yin, He He, Kai-Wei Chang, Xiaofei Ma, and Bing Xiang. Efficient Shapley Values Estimation by Amortization for Text Classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8666–8680. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.483. URL https://doi.org/10.18653/v1/2023.acl-long.483.

Peng Yu, Albert Bifet, Jesse Read, and Chao Xu. Linear TreeShap. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 25818–25828. Curran Associates, Inc., 2022. doi: 10.52202/068431-1872. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a5a3b1ef79520b7cd122d888673a3ebc-Paper-Conference.pdf.

Arber Zela, Julien Niklas Siems, Lucas Zimmer, Jovita Lukasik, Margret Keuper, and Frank Hutter. Surrogate NAS Benchmarks: Going Beyond the Limited Search Spaces of Tabular NAS Benchmarks. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=OnpFa95RVqs.

Artjom Zern, Klaus Broelemann, and Gjergji Kasneci. Interventional SHAP Values and Interaction Values for Piecewise Linear Regression Trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11164–11173. Association for the Advancement of Artificial Intelligence (AAAI), 2023. doi: 10.1609/aaai.v37i9.26322. URL https://doi.org/10.1609/aaai.v37i9.26322.

APPENDIX CONTENTS

A Shapley Estimation Identities **19**

B Method Details **20**

B.1 Game Context Modules 20

C Prior **22**

C.1 Batch Construction and Related Games 22

C.2 Summary of the Priors 23

C.3 Sum of Unanimity Games 24

C.4 Gaussian Process on the Hypercube 25

C.5 TabICL TreeSHAP 26

C.6 Interventional Games on Causal Models 27

D Training **28**

D.1 Pretraining and Inference 28

D.2 Hyperparameter Optimization 28

E Experiments **31**

E.1 Experiment setup 31

E.2 Cooperative Games 32

E.3 Amortizing Shapley Explanations with a Prior Fitted Model 34

E.4 Comparison with Amortized Explainers 39

E.5 Ablation 42

F Notation **44**

A Shapley Estimation Identities

We collect the identities used by the architecture and comparison methods.

Coalition-value form. Separating the two terms in (1) gives the form used by SVARM (Kolpaczki et al., 2024):

$$\phi_i(\nu) = \underbrace{\sum_{S \subseteq \mathcal{P}\setminus\{i\}} \frac{1}{p\binom{p-1}{|S|}}\, \nu(S \cup \{i\})}_{=:\,\phi_i^{+}(\nu)} \; - \; \underbrace{\sum_{S \subseteq \mathcal{P}\setminus\{i\}} \frac{1}{p\binom{p-1}{|S|}}\, \nu(S)}_{=:\,\phi_i^{-}(\nu)}. \tag{5}$$

Weighted least squares. For $p \geq 2$, the Shapley values solve a constrained weighted least-squares problem (Charnes et al., 1988; Lundberg & Lee, 2017):

$$\phi(\nu) = \underset{\beta \in \mathbb{R}^p}{\arg\min}\; \mathbb{E}_{S\sim\mu}\left[\left(\nu(S) - \nu(\emptyset) - \mathbf{1}_S^\top \beta\right)^2\right] \quad \text{s.t.} \quad \mathbf{1}^\top \beta = \nu(\mathcal{P}) - \nu(\emptyset) \tag{6}$$

Here $\mathbf{1}_S \in \{0,1\}^p$ indicates coalition membership, $\mathbf{1}$ is the vector of ones, and the Shapley kernel is

$$\mu(S) \propto \frac{p-1}{\binom{p}{|S|}\,|S|\,(p-|S|)}$$

for $\emptyset \subsetneq S \subsetneq \mathcal{P}$. KernelSHAP estimates the solution from sampled coalition evaluations (Lundberg & Lee, 2017).

Residual correction. For a proxy game $\hat{\nu}$, linearity gives:

$$\phi_i(\nu) = \underbrace{\phi_i(\hat{\nu})}_{\text{proxy}} + \underbrace{\phi_i(\nu - \hat{\nu})}_{\text{residual}}. \tag{7}$$

Estimating the residual term corrects the proxy's attributions, as in RegressionMSR and ProxySHAP (Thies et al., 2026; Witter et al., 2025).

FastSHAP objective. For a fixed machine learning model and feature-removal rule, let $\nu_{\mathbf{x},y}$ be the game explaining output class $y \in \mathcal{Y}$ at input $\mathbf{x} \in \mathcal{X}$. FastSHAP trains an explainer $\phi_{\theta,\mathrm{FastSHAP}} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^p$ with parameters θ by minimizing (Jethani et al., 2022):

$$\mathcal{L}(\theta) := \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\, \mathbb{E}_{y\sim \mathrm{Unif}(\mathcal{Y})}\, \mathbb{E}_{S\sim\mu}\left[\left(\nu_{\mathbf{x},y}(S) - \nu_{\mathbf{x},y}(\emptyset) - \mathbf{1}_S^\top \phi_{\theta,\mathrm{FastSHAP}}(\mathbf{x},y)\right)^2\right]. \tag{8}$$

Here $p(\mathbf{x})$ is the input distribution, $\mathcal{Y}$ is the set of output classes, and μ is the Shapley kernel defined above. During training and inference, additive efficient normalization enforces the sum constraint (Jethani et al., 2022; Ruiz et al., 1998):

$$\phi_{\theta,\mathrm{FastSHAP}}(\mathbf{x},y) \leftarrow \phi_{\theta,\mathrm{FastSHAP}}(\mathbf{x},y) + \frac{1}{p}\Big(\nu_{\mathbf{x},y}(\mathcal{P}) - \nu_{\mathbf{x},y}(\emptyset) - \mathbf{1}^\top \phi_{\theta,\mathrm{FastSHAP}}(\mathbf{x},y)\Big). \tag{9}$$

B METHOD DETAILS

This appendix section provides additional details of our method that extend Section 2.

Cell encoding. The maps $\psi_{\pm}$ of (3) are affine rather than linear, i.e., have a bias term. After the shift by $\nu(\emptyset)$, a non-empty coalition can have the value 0, which a linear map would embed as zero whether or not the player is present, while the biases keep the membership visible. The size term gives each cell its coalition size, which it would otherwise learn only after attention over the players has counted the membership bits. The relative size s_k/p is comparable across games, but the Shapley weights depend on s_k and p separately, so the second feature adds the absolute size. Its logarithm keeps it on the scale of the relative size, maps the empty coalition to zero, and resolves small coalitions, where one player makes the largest difference, more finely than large ones.

Query-aware attention scaling. The coalition axis may present more keys at inference than the largest pretraining budget. The column attention over coalitions and the attention pooling of the readout, which both attend over the coalitions, therefore adopt the query-aware scaling of TabICLv2 (Qu et al., 2026). The player axis, whose key set is fixed by the game itself, is left unscaled.

Hyperparameter selection. Section D.2 describes how we select the hyperparameters of the context model.

B.1 GAME CONTEXT MODULES

Both modules of Section 2.2 are player equivariant and add their output to the cell embeddings through zero-initialized projections. At initialization they therefore leave the cell embeddings unchanged, and a model that receives neither input computes exactly the architecture of Section 2.1.

Natural context. The natural-context module reads the feature values themselves, not membership masks. Its input is the explicand $\mathbf{x}$ with its prediction $f(\mathbf{x})$ and R background rows $\mathbf{z}_1, \ldots, \mathbf{z}_R$ with their predictions $f(\mathbf{z}_r)$, where player i corresponds to feature i of the predictor. Each feature is standardized to zero mean and unit variance over the background rows and the explicand, and the predictions are anchored and scaled by the same map as the coalition values.

Every pair of a row and a player i becomes one token with three scalar entries, the standardized value of feature i in that row, the prediction of the row, and a bit that marks the explicand row. A learned linear layer maps these three entries to a vector of width D, and the rest of the encoding is learned by the attention layers that follow. Self-attention over the rows of each player column is followed by self-attention over the players of each row, and the rows of each player are then pooled into one player vector u_i whose mean over players is the game vector u. A game without natural context receives $u_i = u = 0$, so its cell embeddings stay unchanged. When only the predictions are withheld, the prediction entry of every token is set to zero and the tokens keep the coordinates and the explicand bit, so the module still identifies players across related games but no longer sees the predictor.

Both are projected and added to every cell embedding of the corresponding player,

$$\mathbf{e}_{k,i} \leftarrow \mathbf{e}_{k,i} + W_{\mathrm{p}} u_i + W_{\mathrm{g}} u, \tag{10}$$

before the first interaction round. Rows and players are treated as sets without positional encodings, so the backbone stays player equivariant.

Attention over related games. The related-games module lets the target game attend to its related games. Each related game supplies its own membership matrix and coalition values, embedded by the input encoding of Section 2.1, and no Shapley labels. After the player attention of each round, every target cell (S_k, i) queries all cells of player i in the related games through multi-head cross-attention, with a shared projection of the membership bit added to queries and keys. Related games pass through the same transformer rounds as the target (stage 2 in Figure 2), so at later rounds their cells already carry the information exchanged within their own game. The update enters through a zero-initialized residual projection, so a game without related games receives is not influenced. Whether cross-attention runs in every round or only in the last one is a hyperparameter, and the hyperparameter search of Section D.2 selects every round.

Training with partial inputs. The context model is pretrained on the groups of related games drawn from the prior families (Section 2.3.1), and all games of a group enter the batch, so each of them contributes one summand to (12) and at the same time serves as a related game for the others. To keep the model applicable with *partial* inputs, we transform every group before the model sees it, with the probabilities of the selected configuration in Table 9.

1. Each group draws a subgroup size s that divides G from a fixed distribution and is split into G/s subgroups of consecutive games, within which attention over related games runs. A game in a subgroup of size s sees $s - 1$ related games, and a game in a subgroup of size one sees none. Since every size divides G, the distribution of s is also the distribution of the number of related games a game is trained with.
2. Each game loses its entire natural context with a fixed probability, so neither the coordinates nor the predictions of the explicand and the background rows reach the model, while its coalitions, values and targets stay unchanged.
3. Each group withholds the predictions of its natural context with a fixed probability, so its games keep the coordinates of the explicand and the background rows and the explicand bit, and the prediction entry of every token is zero.
4. With a fixed probability, one related game of a subgroup is replaced by a game with the same number of players from a different group, and the target is unchanged, so the model learns to discount a related game that does not match the target. A replacement requires such a game in the same micro-batch, the part of a batch that one forward pass processes, and because the number of players is drawn per group, replacements are uncommon.

A group is never split across micro-batches, and all of these draws are seeded by the batch and the position of the group within it, so they do not depend on how the batch is split into micro-batches.

Algorithm 1 Generation of one training batch of n_{grp} groups of G related games.

Require: seed and batch index, which initialize the generator
1: draw one band of π_m (Table 7) ▷ batch level
2: **for** each of the n_{grp} groups **do** ▷ group level
3: draw $p \sim \mathcal{U}\{6, \dots, 32\}$, then m within the band subject to $p + 1 \le m \le 2^{p-1} - 1$
4: draw the family κ uniformly among the families that support p players
5: draw the hyperparameters η_κ and the structure the group shares, including its dummy player and symmetric pair
6: **for** each of the G games of the group **do** ▷ game level
7: **repeat**
8: draw a game (ν, ϕ) from the shared structure
9: **until** $\max_i |\phi_i| > 10^{-9}$ and $|\nu(\mathcal{P}) - \nu(\emptyset)| > 10^{-9} \max(1, |\nu(\mathcal{P})|)$
10: draw m coalitions, each with a size uniform in $\{1, \dots, p-1\}$, and evaluate ν on them
11: attach the natural context of the game, or a zero tensor and a flag that marks it as absent
12: **end for**
13: round ϕ to single precision and add the efficiency residual to one unprotected player
14: **end for**
15: stack the groups in order, pad to the largest p and m of the batch, and give every game the identifier of its group

C PRIOR

The two mechanism families of Section 2.3.1 cover the two standard semantics of the value function in feature attribution (Chen et al., 2020). TabICL TreeSHAP intervenes on the inputs of a model: it takes absent features from background rows and therefore ignores the dependence between features (Janzing et al., 2020). The interventional games on causal models instead intervene on the process that generates the data, so setting $X_S = \mathbf{x}_S$ also changes the descendants of S and a feature receives credit for its effect through other features (Heskes et al., 2020).

C.1 BATCH CONSTRUCTION AND RELATED GAMES

Batch construction. Every draw uses one seeded random number generator, so a batch is a deterministic function of the seed and the batch index. Algorithm 1 shows the three nested levels, batch, group, and game, in the order in which the generator is consumed. A family that cannot produce p players is excluded from the family draw and the probabilities are renormalized over the others, so a group with $p > 16$ draws each of the other three families with probability $1/3$. Every game draws its own coalitions, so the model never relies on related games observing the same coalitions. The rejection test, the dummy and symmetric players, and the protected targets are described below.

Rejection of degenerate games. The inner loop of Algorithm 1 rejects a game when all its Shapley values vanish, $\max_i |\phi_i| \le 10^{-9}$, which holds for a constant value function, or when its grand and empty coalitions tie, $|\nu(\mathcal{P}) - \nu(\emptyset)| \le 10^{-9} \max(1, |\nu(\mathcal{P})|)$. The first case leaves the scale ν_{scale} of Section 2.1 zero, and both tolerances lie far below single precision, so the tests remove exact ties up to rounding and keep every game with a small but genuine effect. Ties arise almost only in TabICL TreeSHAP, where a tree ensemble returns one modal prediction for most rows and the background mean can equal it bitwise. The usable-row screen of Section C.5 rejects groups in which this affects most rows before any game is drawn. The other families draw continuous coefficients, so a tie has probability zero. Redraws use the same generator and are capped at 100 per game, after which TabICL TreeSHAP redraws the whole group and the other families abort.

Dummy and symmetric players. A player i is a dummy when $\nu(S \cup \{i\}) = \nu(S)$ for every coalition $S \subseteq \mathcal{P} \setminus \{i\}$, which implies $\phi_i = 0$. Two players i and j are symmetric when $\nu(S \cup \{i\}) = \nu(S \cup \{j\})$ for every coalition $S \subseteq \mathcal{P} \setminus \{i, j\}$, which implies $\phi_i = \phi_j$. Randomly drawn games rarely satisfy either condition exactly, so a family can build them into the structure of its games, and both axioms then hold for the game itself and not only for its Shapley values. With a probability that depends on the family, one player drawn uniformly at random becomes the dummy, and with another such probability, two distinct players drawn uniformly from the remaining players become

the symmetric pair, so the dummy and the pair never overlap. Each family section below describes how the family realizes them, and the sum of unanimity games and the interventional games on causal models are the only families with a symmetric pair.

Protected targets. The Shapley values are computed in double precision and stored in single precision, and the rounding leaves $\sum_i \phi_i$ a few units in the last place away from $\nu(\mathcal{P}) - \nu(\emptyset)$. Spreading this residual over all players would give the dummy a small nonzero value and break the tie of the symmetric pair. We therefore add the residual to the player with the largest $|\phi_i|$ among the players that are neither a dummy nor part of the symmetric pair. The stored targets thus satisfy efficiency up to single precision and the dummy and symmetry axioms exactly.

Batch tensors. The batch consists of coalition masks $\mathbf{S} \in \{0,1\}^{B\times m\times p}$, their values $\nu \in \mathbb{R}^{B\times m}$, the grand and empty coalition $\nu(\mathcal{P}), \nu(\emptyset) \in \mathbb{R}^B$, and the targets $\phi \in \mathbb{R}^{B\times p}$. Games within a group share p and m, so we pad only across groups to the largest p and m in the batch and mask the padded entries.

Invariants of a group. The training procedure relies on four invariants that the generator upholds.

1. All games of a group have the same p and m and therefore the same padding mask.
2. Player i has the same meaning in every game of a group, because every family builds these games on one shared set of players, namely one set of interaction terms, one lengthscale per player, one fitted predictor, or one graph. The attention over related games compares them player by player and depends on this invariant.
3. The games of a group occupy consecutive positions in the batch, so the batch is a sequence of n_{grp} blocks of G games. The split into subgroups and micro-batches of Section B.1 slices the batch by position and therefore relies on this layout.
4. The games of a group count as related only when they share a function or a latent draw, and shared hyperparameters alone do not make games related.

Notation for the tables. The following tables list the distributions of the prior for each family. We write $\mathcal{U}[a,b]$ for the uniform distribution on an interval, $\mathcal{U}\{a,\dots,b\}$ for the uniform distribution on integers, and $\log\mathcal{U}[a,b]$ for the log-uniform distribution. A quantity is drawn per game unless the table says that it is drawn per group.

C.2 Summary of the Priors

Table 2 compares the four families along the properties that matter for pretraining: how the game arises, how many players it admits, where its exact targets come from, which axioms it can exercise, whether it supplies natural context, and what the games of a group share.

Table 2: Properties of the four prior families.

	Sum of unanimity games	Gaussian process on the hypercube	TabICL TreeSHAP	Interventional games on causal models
Game arises from	an interaction structure	an interaction structure	a mechanism	a mechanism
Players p	6 to 32	6 to 32	6 to 32	6 to 16
Exact targets from	the coefficients, in closed form	the joint Gaussian of ϕ and the observed values	interventional TreeSHAP	enumerating all 2^p coalitions
Dummy player	yes	no	yes, a feature the ensemble never splits on	yes, an isolated node
Symmetric pair	yes	no	no	yes, two roots with equal weights
Natural context	no	no	yes, shared by the group	yes, drawn per game
A group shares	$\mathcal{T}$ and the coefficients	lengthscales, atoms, size-mean and one path	dataset, predictor and background sample	graph, edge weights and base explicand
Each game draws	terms with redrawn coalitions and jittered coefficients	its own path, a fraction α shared	its explicand	its explicand
Details	Section C.3	Section C.4	Section C.5	Section C.6

C.3 Sum of Unanimity Games

Interaction terms. In the value function of Section 2.3.1, $\mathcal{T}$ is a finite list of non-empty coalitions $T \subseteq \mathcal{P}$, and each T contributes the unanimity game $\mathbb{1}_{\{T \subseteq S\}}$ with weight c_T, which pays only when all players of T cooperate. We call each pair (T, c_T) an interaction term. A term with $|T| = 1$ is a main effect of one player, and a term with $|T| \geq 2$ is an interaction of order $|T|$. The list $\mathcal{T}$ is drawn in blocks, each with its own number of terms, its own upper bound on $|T|$ and its own coefficient scale. Within a block the size $|T|$ of a term is uniform up to the bound and the members of T are uniform among the players. The Shapley values follow in closed form from the terms, as stated in Section 2.3.1.

Related games. Related games share the list $\mathcal{T}$ and the coefficients $(c_T)_{T \in \mathcal{T}}$ across a group. Each game copies the shared terms and then, independently for every term with a probability drawn once per group, replaces the coalition T of the term by a fresh coalition drawn from the same block, keeping the coefficient of the term. It further replaces every shared coefficient c_T by $c_T(1 + \delta\varepsilon_T)$, where the jitter δ is drawn once per group (Table 3) and $\varepsilon_T \sim \mathcal{N}(0, 1)$ is drawn independently for every term and every game, so each coefficient is perturbed by a relative amount of typical size δ. Finally, it draws its own constant $c_\emptyset$. Related games are therefore perturbations of one game rather than independent draws, and the Shapley values of each game follow in closed form from its own terms, so the targets stay exact.

Dummy and symmetric players. The dummy player is contained in no coalition T, neither in the shared terms nor in those that a game redraws. For the symmetric pair i, j, the membership of j in every coalition T is copied from i, so every term contains both players or neither, and a coalition that this leaves empty has a constant unanimity game, whose coefficient is merged into $c_\emptyset$. Both are drawn once per group, so all games of a group protect the same players.

Distributions. Table 3 lists the distributions. They vary the number, the size $|T|$ and the coefficient scale of the interaction terms, so the games range from a few main effects to many interactions of

order up to six. Coefficient laws with heavy tails produce games in which a few terms dominate, and the jitter, drawn once per group, keeps related games close to each other.

Table 3: Distributions of the sum of unanimity games.

Quantity	Distribution
Number of blocks	$\mathcal{U}\{1, 2, 3\}$
Number of interaction terms per block	$\log \mathcal{U}[2, 12]$
Largest $\lvert T\rvert$ per block	$\log \mathcal{U}[1, 6]$, and each $\lvert T\rvert$ is uniform up to it
Coefficient scale per block	$\log \mathcal{U}[0.3, 2]$
Law of the coefficients (per group)	normal 0.35, Laplace 0.25, Student-t 0.2, Rademacher 0.2
Scale of the constant $c_\emptyset$	$\log \mathcal{U}[0.01, 3]$
Probability of a dummy player (per group)	0.15
Probability of a symmetric pair (per group)	0.2
Probability that a game redraws the coalition T of a term (per group)	$\mathcal{U}[0.1, 0.3]$
Coefficient jitter δ (per group)	$\mathcal{U}[0.05, 0.2]$

C.4 Gaussian Process on the Hypercube

Let $\mathbf{A} \in \mathbb{R}^{p \times 2^p}$ be the Shapley matrix of (1), so that $\phi = \mathbf{A}\nu$ for the vector ν of all coalition values, and let W hold the empty, the grand and the m context coalitions. Under the zero-mean process with kernel matrix K,

$$\begin{pmatrix} \phi \\ \nu(W) \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} \mathbf{A}K\mathbf{A}^\top & \mathbf{A}K(\cdot, W) \\ K(W, \cdot)\mathbf{A}^\top & K(W, W) \end{pmatrix}\right). \tag{11}$$

The product form of the Hamming kernel gives the two blocks involving $\mathbf{A}$ in $\mathcal{O}(p^4)$ and $\mathcal{O}(p^2)$ per coalition, via elementary symmetric polynomials (Mohammadi et al., 2025a; Rundel et al., 2026), and one Cholesky factorization of the $(p + m)$-dimensional covariance in $\mathcal{O}((p + m)^3)$ draws the values of one sample path together with its exact Shapley values.

Given their parameters, the size-mean is deterministic and the atoms are Gaussian, so we draw the lengthscales and the components first, sample the joint Gaussian conditional on them, and add the components to the values and, in closed form, to the Shapley values: the size-mean $A\,(1 - e^{-\gamma|S|/p})$ adds $A\,(1 - e^{-\gamma})/p$ to every player and an atom $c_i\,\mathbb{1}_{\{T_i \subseteq S\}}$ adds $c_i/|T_i|$ to each member of T_i. Table 4 lists the distributions, where the lengthscales are drawn log-normally with a location per game and a spread, the standard deviation of the logarithm, per player, and the amplitude A of the size-mean and the common scale of the atoms are set from a share s such that the component's variance over uniformly random coalitions is $s/(1 - s)$ times that of the process. An atom has a heavy-tailed volume $v_i \sim |\mathrm{Cauchy}(0, 1)|$ and the amplitude $c_i \sim \mathcal{N}(3v_i\varsigma_i,\, v_i^2)$, whose mean-to-noise ratio of 3 pins its sign ς_i, the sign of the game, while keeping the game Gaussian. The family has no dummy players and no symmetric pair, and all distributions were calibrated on a small set of real precomputed training games.

Related games. A group shares the structure of the game, namely the lengthscales, the direction of the size trend, the Möbius atoms and the size-mean, together with one sample path g_s. Game j of the group uses the path $\sqrt{\alpha}\, g_\mathrm{s} + \sqrt{1 - \alpha}\, g_j$, where g_j is a path of its own. Since both paths have the kernel K of the process, every game follows the same distribution as an independent game, and two related games have the cross-covariance αK. With $\alpha = 0$ the games of a group would share only their structure and would not count as related, and our prior therefore draws $\alpha \geq 0.5$.

Cost. The shared path must be sampled jointly on the coalitions of all games of the group, so a group is more expensive to generate than G independent games.

Distributions. Table 4 lists the distributions. Short lengthscales make the values of similar coalitions less correlated, which gives interactions of higher order, and the spread between players makes some players much more influential than others. The atoms and the size-mean add a few dominant

interactions and a trend with coalition size, and their largest shares bound how much of the variance they take from the process.

Table 4: Distributions of the Gaussian process on the hypercube.

Quantity	Distribution
Lengthscales	log-normal with median 0.08, with spread 2.5 of the log-lengthscale between games and 1.3 between the players of a game
Probability that all players share one lengthscale	0.25
Variance of the process	1
Probability that the game increases with coalition size	0.85
Number of Möbius atoms	Poisson with mean $0.8\,p$
Order of an atom	geometric with a parameter from $\mathcal{U}[0.2, 0.75]$, where a larger parameter gives lower orders
Ratio of mean to noise of the atom amplitudes	3
Atoms of even order	none
Largest share of variance of the atoms	0.35
Probability of a size-mean	0.7
Largest share of variance of the size-mean	0.99
Saturation rate γ of the size-mean	log-normal with median 4 and spread 2.2
Shared fraction α (per group)	$\mathcal{U}[0.5, 0.9]$

C.5 TabICL TreeSHAP

Related games. Related games share the dataset, the fitted predictor, and the background sample, and differ only in the explicand, which is a row of the dataset. The rows are visited in random order, and a row is skipped when its explicand or its Shapley values equal those of an earlier game of the group, so no two games of a group share an explicand or a vector of Shapley values. The background sample is part of the game definition, so the targets are exact for the game the model sees rather than for an idealized infinite background.

Natural context. In addition to the coalition values, every game receives natural context consisting of its explicand with its prediction and up to 32 rows of the background sample with their predictions. These rows are drawn once per group, so all games of a group see the same background rows.

Dummy players and rejection. The dummy is an extra feature that the predictor never sees during fitting, so no tree splits on it and its interventional Shapley value is exactly zero. It is added only when the game has at least three players, so that at least two fitted features remain. A constant column of the dataset is an exact dummy for the same reason, and at least one such column occurs in about a quarter of the datasets, so these dummies are protected in the targets as well. The family has no symmetric pair, because a fitted tree ensemble does not treat two features exchangeably. A group is redrawn when its predictor is less accurate than a constant on the held-out rows, or when fewer than 25% of the rows yield a game that is not degenerate.

Distributions. Table 5 lists the distributions. The task, the kind of ensemble, the number of trees and the depth vary the predictor, and the order of an interaction is at most the depth of a tree. The size of the background sample varies how closely the interventional game averages over the data.

Table 5: Distributions of the TabICL TreeSHAP games.

Quantity	Distribution
Dataset generator	graph-based structural causal models of the TabICLv2 prior
Rows of the dataset	$\mathcal{U}\{128, \dots, 384\}$, of which 25% are held out
Rows R of the background sample	$\mathcal{U}\{8, \dots, 48\}$
Task	classification with probability 0.3, in which case the game explains the probability of class 1, and regression otherwise
Tree ensemble	random forest 0.4, extra trees 0.3, gradient boosting 0.3, and only the two forests for classification
Number of trees	$\mathcal{U}\{8, \dots, 48\}$
Tree depth	$\mathcal{U}\{2, \dots, 6\}$
Probability of an extra feature that the predictor never sees	0.15
Rows of natural context (per group)	32, padded and masked when the background sample is smaller

C.6 Interventional Games on Causal Models

Related games. A group shares one linear directed acyclic graph whose sink is the outcome Y, together with its edge weights, a bias, and a base explicand $\mathbf{x}_{\text{base}}$. Each game explains its own explicand $\mathbf{x} = \mathbf{x}_{\text{base}} + \sigma\varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I)$, so σ controls how far the explicands of a group spread.

Game and exact targets. A coalition S intervenes on its nodes, $\mathrm{do}(X_S = \mathbf{x}_S)$, and the remaining nodes follow the linear mechanisms with zero-mean noise. We compute the Shapley values by enumerating all 2^p coalitions, which limits the family to 16 players.

Dummy and symmetric players. All edges of the dummy node have weight zero, so it is an isolated node of the graph and the only feature that is not an ancestor of Y. The symmetric pair consists of two root nodes with the same outgoing edge weights, and their explicand values are equal in every game of the group.

Natural context. The natural context of a game consists of 32 observational rows drawn from the causal model with their outcomes, and, unlike in TabICL TreeSHAP, every game draws its own rows.

Distributions. Table 6 lists the distributions. The edge probability controls how many mediated paths connect the features to Y, and the edge weights control how strongly the effects combine along them. The spread σ sets how different the explicands of related games are.

Table 6: Distributions of the interventional games on causal models.

Quantity	Distribution
Graph (per group)	Erdős–Rényi DAG with edge probability $\mathcal{U}[0.15, 0.45]$, and every feature node except a dummy has an edge to Y
Edge weights (per group)	magnitude $\mathcal{U}[0.3, 1.5]$ with a random sign
Spread σ of the explicands (per group)	$\log \mathcal{U}[0.25, 1]$
Probability of a dummy node without edges (per group)	0.15
Probability of a symmetric pair (per group)	0.05, realized as two root nodes with the same outgoing weights and the same explicand value
Rows of natural context	32 per game

D TRAINING

D.1 PRETRAINING AND INFERENCE

Scaling. For each game, ν_{scale} is the larger of $|\max_t \nu(S_t) - \nu(\emptyset)|$ and $|\nu(\mathcal{P}) - \nu(\emptyset)|$, using only observed values and the two anchors. Before cell encoding, we subtract $\nu(\emptyset)$ from the observed and grand-coalition values and divide by ν_{scale}. Shapley values are unchanged by the constant shift and scale linearly, so the exact training target is $\overline{\phi}_i = \phi_i / \nu_{\text{scale}}$. Both the pinball loss and the efficiency-gap penalty below use these normalized units.

Objective. For each player, ShapFM predicts $Q = 999$ quantiles at the levels $\tau_q \in \{0.001, 0.002, \ldots, 0.999\}$, and we train it with the pinball loss $\rho_\tau(u) = \max\{\tau u, (\tau - 1)u\}$ and a penalty on the efficiency gap. For a micro-batch of B games, the loss is

$$\mathcal{L} = \frac{1}{B} \sum_{j=1}^{B} \left[\frac{1}{p_j} \sum_{i \in \mathcal{P}_j} \sum_{q=1}^{Q} \rho_{\tau_q}\big(\overline{\phi}_i^j - \hat{\phi}_i^j(\tau_q)\big) \; + \; \lambda_{\text{eff}}\, g_j^2 \right], \qquad g_j = \overline{\nu}_j(\mathcal{P}) - \sum_{i \in \mathcal{P}_j} \hat{\phi}_i^{\text{raw},j}, \tag{12}$$

where $\mathcal{P}_j$ denotes the p_j unpadded players of game j, $\hat{\phi}_i^{\text{raw},j}$ is the mean of the raw quantiles before (4), and $\lambda_{\text{eff}} = 0.01$. After (4), the gap would vanish. The gap depends on the quantiles only through their mean, so the penalty shifts each predicted distribution but leaves its spread unchanged.

The objective is unchanged with the optional inputs. We pretrain the context model on the prior of Section 2.3. Every game of a group is both a training game and a related game for the others. To keep the model usable with partial inputs, we split groups, drop the natural context or withhold its predictions, and inject unrelated games, each with a fixed probability (Section B.1).

We optimize the hidden weight matrices with Muon (Jordan, 2024) (momentum 0.95, weight decay 0.01) and all remaining parameters with AdamW. The learning rate follows a linear warmup and a cosine decay to zero, and gradients are clipped at norm 10. Training runs for 30,000 steps with $B = 32$ games per step.

We select the hyperparameters of the context model by Bayesian optimization over 150 proxy trainings of 1,500 steps, searching the backbone and the two context modules jointly (Section D.2). Each trial trains on the same prior and is scored by the normalized mean squared error on real games with related games and natural context. We retrain the three best configurations for 5,000 steps at two seeds, keep the one with the lowest error on a separate grid of real games, and train it once with this schedule. The ablation arms of Section E.5 reuse this configuration and only remove modules.

Inference. At inference, ShapFM receives the observations D_m of an unseen target game and, if available, a game context, and it returns the quantiles of every player's Shapley value in one forward pass without any gradient step. We recompute ν_{scale} from that game's observations and anchor values, normalize the inputs, and enforce efficiency on the predicted quantiles by (4). We then multiply every quantile by ν_{scale} to return to the units of the game; we do not add back $\nu(\emptyset)$, since a constant shift does not affect Shapley values. The point estimate of each player is the mean of its quantiles, and these estimates sum to $\nu(\mathcal{P}) - \nu(\emptyset)$. The quantile spread expresses the uncertainty of the estimate.

D.2 HYPERPARAMETER OPTIMIZATION

We select the hyperparameters of the context model, and every other model of the paper reuses them.

Why the search needs context on both sides. The context modules influence the score of a trial only if the model is trained and scored on games with context. We therefore train each trial on the prior of Section 2.3 and score it on real games with related games and natural context (Section 2.2). We also track the error on games from the prior and on real games without context, but we do not optimize it.

Validation grid. The validation grid consists of interventional games (Section 2.3.1) of random forests and XGBoost models with 32 trees on the 24 datasets of Table 8. For each dataset and model,

Table 7: Budget bands of π_m. Within a band, m is drawn log-uniformly and capped at $(2^p - 2)/2$.

Band	Range of m	Probability
Starvation	$p + 1$ to 48	0.50
Squeeze	16 to 96	0.33
Plentiful	64 to 160	0.15
Tail	257 to 512	0.02

Table 8: The 24 datasets of the hyperparameter search with their numbers of features.

Dataset	Features	Dataset	Features
Fitness Club	6	Wine Quality	12
Health Insurance	6	Good Customer	13
Maternal Health	6	Adult Census	14
Fiat 500	7	Miami Housing	15
Concrete Strength	8	Seismic Bumps	15
Diabetes	8	Online Shoppers	17
Diamonds	9	Credit-G	20
Website Phishing	9	Airline Satisfaction	21
Bank Customer Churn	10	JM1	21
E-commerce Shipping	10	HELOC	23
Give Me Credit	10	Marketing Campaign	25
SDSS17	11	Hazelnut	30

we fit one predictor on the training rows and draw 32 background rows and 16 explicands from held back rows, arranged in two groups of eight, which gives 768 games with exact Shapley values from TreeSHAP. Each game is observed at p+1, 32, 128 and 512 coalitions, capped at $2^p - 2$. The confirmation grid uses three other predictors per dataset and model with six groups each.

Objective. Ax minimizes the NMSE (Section 3) of the point estimate of ShapFM (Section D.1), where we sum the squared errors and the squared Shapley values over all games of the validation grid before dividing. We normalize every game to unit spread, so that every game carries the same weight. We read the objective at the end of each proxy training.

Search space. The search covers 27 hyperparameters jointly (Table 9). Twelve concern the backbone, the learning rate and the warmup. Fifteen concern the two context modules and how context appears during training, namely the probability of dropping the natural context entirely, of dropping only its predictions, and of adding an unrelated game, and the weights for cutting a group of eight games down to one, two or four. Every combination yields a valid model, since every width is a multiple of 16 and every head count is at most 16. The alternating stage order, the number of quantiles, the batch size, the step budget, the optimizer and the loss stay fixed, since changing them would change what the objective measures.

Search procedure. We run 150 trials of 1,500 steps with the Bayesian optimization platform Ax (Olson et al., 2025), the first 40 of them quasi-random. A trial stops early once it is at least 30% through its run and ranks among the worst quarter of the trials so far.

Confirmation. We retrain the three best trials for 5,000 steps at two seeds and score them on the confirmation grid. The selected configuration reaches an NMSE of 0.0282, against 0.0292 and 0.0294 for the other two. The production model and all ablation arms start from it.

Selected configuration. The selected configuration drops the natural context with probability 0.5, never drops only the predictions, and adds an unrelated game with probability 0.4. It cuts a group to two games about as often as it keeps all eight, and rarely to one or four. The model is therefore trained to cope with missing and misleading context, which Section 2.2 relies on. The selected value lies on the boundary of its range for 21 of the 22 numeric and ordered hyperparameters, so wider ranges may improve the model further.

Hyperparameter	Range	Selected
Backbone and optimization		
Cell width d	$\{128, 192, 256, 384\}$	128
Rounds L	2 to 8	8
Heads, coalition attention	$\{4, 8, 16\}$	16
Heads, player attention	$\{4, 8, 16\}$	16
Heads, pooling	$\{4, 8, 16\}$	16
Inducing points K	16 to 256, log	16
Pooling feed-forward ratio	$\{1, 2, 4, 8\}$	1
Query-aware scaling, coalition attention	on, off	on
Query-aware scaling, PMA pooling	on, off	on
Quantile head rank	$\{8, 16, 32, \text{full}\}$	full
Peak learning rate	10^{-4} to $3 \cdot 10^{-3}$, log	$2.09 \cdot 10^{-3}$
Warmup fraction	0.02 to 0.10	0.10
Context modules and context during training		
Encoder width D	$\{32, 64, 128\}$	32
Encoder heads	$\{2, 4, 8\}$	8
Layers over rows	0 to 3	3
Layers over players	0 to 2	0
Layers over rows and players jointly	0 to 2	2
Encoder sees predictions	yes, no	yes
Coordinate normalization	per column, shared	per column
Related-games attention heads	$\{4, 8, 16\}$	16
Related-games attention rounds	all, last	all
Probability of dropping natural context	0 to 0.5	0.5
Probability of dropping predictions only	0 to 0.5	0
Probability of adding an unrelated game	0 to 0.4	0.4
Weight of groups of one game	0.05 to 1, log	0.05
Weight of groups of two games	0.05 to 1, log	1
Weight of groups of four games	0.05 to 1, log	0.05

Table 9: Search space of the hyperparameter optimization and the selected configuration. Groups of eight games have a fixed weight of 1, and the four weights are normalized to the probabilities of the subgroup sizes of Section B.1, which are 0.024, 0.476, 0.024 and 0.476 for subgroups of one, two, four and eight games in the selected configuration.

E EXPERIMENTS

E.1 EXPERIMENT SETUP

This section provides general information about the empirical evaluation conducted in Section 3.

Budget. A budget of b evaluations of the game includes the evaluations of $\nu(\emptyset)$ and $\nu(\mathcal{P})$. ShapFM spends the remaining m evaluations on coalitions drawn by the sampler of `shapiq` (Muschalik et al., 2024), so that $b = m + 2$. The sampler draws a coalition size uniformly and then a coalition of that size uniformly, which is the sampling law of the prior (Section 2.3). Beyond these evaluations, ShapFM costs one forward pass. A budget never exceeds the 2^p coalitions of the game. We evaluate budgets from $p + 1$ to 100 in steps of one and budgets of p to $32p$ in multiples of p.

Cost of the context. Each related game is observed on its own coalitions at the budget of the target game and shares $\nu(\emptyset)$ with it, so a group of eight games costs $1 + 8(b - 1)$ evaluations. When the points of a dataset are explained as a batch, as is common for local explanations, these are the evaluations every point needs anyway, and the related games cost nothing extra. A single explanation pays $7(b-1)$ additional evaluations for them. The natural context needs the predictions on 32 reference rows, which are computed once per predictor.

Estimators. All estimators run as implemented in `shapiq` with their default settings, on the same budget as ShapFM. Some estimators do not support the smallest budgets. OddSHAP requires at least 10 evaluations, and the default initial design of ShaplEIG requires more than $p + 1$ evaluations. We cap every ShaplEIG explanation at 300 seconds and report timed-out calls as missing. Permutation sampling cannot complete a single permutation with p evaluations and then returns zero for every player.

Metrics. We score every estimate $\hat{\phi}$ of a game against its exact Shapley values ϕ with the NMSE $\|\hat{\phi} - \phi\|^2 / \|\phi\|^2$, which divides the squared error by the squared norm of the exact values. It is therefore comparable across games whose values differ by orders of magnitude, and an NMSE of one equals the error of predicting zero for every player.

Aggregation. The error of a game family is the mean NMSE over its games. The medal tables rank the methods within each game family, and the mean rank averages these ranks over the families, where ties share their average rank and missing results are left out. The figures show geometric means over games with 95% confidence bands from the standard error of the log NMSE, so that single games with a very large error, such as low-budget LeverageSHAP estimates, do not dominate. Before taking logarithms, we clip the NMSE at 10^{-12}, since estimates become exact once the budget covers every coalition.

E.2 Cooperative Games

This section describes the cooperative games we used to *evaluate* ShapFM. Note these games described here have not been provided to ShapFM during training and serve only the purpose of evaluating its performance. The synthetic games in the prior are in Section C. Table 10 summarizes the cooperative games of our experiments. They span three tasks, namely local explanations, hyperparameter importance (HPI) and synthetic games. A game family fixes a model with its data, or a hyperparameter optimization setup, and each of its games explains one point, image, text or instance. All games have exact Shapley values.

TabPFN. We explain the default TabPFN v3 classifier or regressor (Hollmann et al., 2025) with one estimator and seed 40 through the logit of class 1 or the prediction. We fit it on 25 tabular datasets capped at 1,300 rows and on ten TabArena datasets capped at 12,500 rows. Twenty of the 25 datasets stem from TabArena (Erickson et al., 2025), and the others are Adult Census, Bike Sharing, California Housing, Forest Fires and Real Estate. Each family explains 100 held-out points. An absent feature is set to $+\infty$ (TabPFN v3's natural way of modelling missingess), and the outputs on all 2^p coalitions yield the exact Shapley values. The related games are those of the seven nearest explained points in standardized feature space, and the natural context consists of 32 training rows with their outputs.

Random forests and XGBoost. We explain random forests and XGBoost (Chen & Guestrin, 2016) on the ten TabArena datasets with the interventional benchmark of `shapiq` (Muschalik et al., 2024). Each game fits the model on its own $80/20$ split and explains the first test row, with the test rows as reference data of the interventional game (Section 2.3.1). Interventional TreeSHAP yields the exact Shapley values (Zern et al., 2023). The related games explain seven further test rows of the same model, and the natural context consists of 32 reference rows with their predictions.

Random forests and MLPs. Sections 3.2 and 3.4 explain a random forest and an MLP on Adult Census, Bank Marketing and Churn through the predicted probability of class 1. The value function is the interventional game with 32 reference rows. Interventional TreeSHAP (Zern et al., 2023) yields the exact Shapley values of the random forests, and enumerating all coalitions yields those of the MLPs (Section E.4).

Image classification. We explain ImageNet classifiers (Russakovsky et al., 2015) through the probability of the class they predict for the full image. The players of ResNet-18 (He et al., 2016) are 14 superpixels, and an absent superpixel takes the mean color of the image. The players of ViT (Dosovitskiy et al., 2021) and DINOv2 (Oquab et al., 2024) are image patches, and an absent patch is replaced by the mask token.

Sentiment analysis. We explain DistilBERT (Sanh et al., 2019) fine-tuned on IMDB reviews (Maas et al., 2011) through its signed sentiment score. The players are the 14 tokens of a movie review, and an absent token is replaced by the mask token.

Hyperparameter importance. HyperSHAP games (Wever et al., 2026) take the hyperparameters of a learner as players. The value of a coalition is the performance after setting its hyperparameters to a tuned configuration (ablation) or after tuning only its hyperparameters (tunability), with all other hyperparameters at their defaults. We use NAS-Bench-301 on CIFAR-10 (Krizhevsky, 2009; Zela et al., 2022) and the XGBoost pipeline of the RBV2 benchmark in YAHPO Gym on OpenML tasks (Bischl et al., 2025; Pfisterer et al., 2022).

Synthetic games. SOUM games (Muschalik et al., 2024; Tsai et al., 2023) sum 30 or 150 unanimity games over coalitions of at most 5 or 15 of the 15 players, with coefficients drawn uniformly from $[-1, 1]$. The prior of ShapFM contains the same kind of game (Section 2.3.1) but draws its terms and coefficients differently.

Table 10: Cooperative games of our experiments. All games have exact Shapley values.

					Context	
Model	**Data**	**Players**	**Families**	**Games**	**Nat.**	**Rel.**
Local explanation (Štrumbelj & Kononenko, 2010)						
TabPFN v3		6 to 24				
↪ $\leq$ 1,300 rows	10 TabArena datasets (Erickson et al., 2025)		10	100	✓	✓
↪ $\leq$ 12,500 rows	10 TabArena datasets (Erickson et al., 2025)		10	100	✓	✓
Random forest, XGBoost	10 TabArena datasets (Erickson et al., 2025)	6 to 24	20	30	✓	✓
Random forest, MLP	3 tabular datasets (Bischl et al., 2025; Erickson et al., 2025)	13 to 19	6	200	✓	✓
ResNet-18 (superpixels)	ImageNet (Russakovsky et al., 2015)	14	1	30		
ViT (patches)	ImageNet (Russakovsky et al., 2015)					
↪ 3×3 patches		9	1	30		
↪ 4×4 patches		16	1	30		
DINOv2 (patches)	ImageNet (Russakovsky et al., 2015)					
↪ 16 patches		16	1	100		
↪ 20 patches		20	1	30		
DistilBERT (tokens)	IMDB (Maas et al., 2011)	14	1	30		
Hyperparameter importance (HPI) (Wever et al., 2026)						
NAS-Bench-301 cell	CIFAR-10 (Krizhevsky, 2009)	16				
↪ Ablation			1	30		
↪ Tunability			1	30		
XGBoost pipeline (RBV2)	30 OpenML tasks (Bischl et al., 2025)	15				
↪ Ablation			1	30		
↪ Tunability			1	30		
Synthetic (Tsai et al., 2023)						
SOUM	30 or 150 terms of size ≤ 5 or ≤ 15	15	4	10		
Total		6 to 24	60	4,210		

E.3 Amortizing Shapley Explanations with a Prior Fitted Model

This section complements Section 3.1. We first detail the context inputs and the construction of the TabPFN and tree games. Figure 8 extends Figure 4 to budgets of up to $32p$ for the TabPFN and tree games, and Figure 9 does the same for the image, sentiment, HPI and SOUM games. Tables 11 and 12 report the error per game family at $2p$ and $4p$ evaluations, complementing Table 1 at $p + 2$.

Related games. The TabPFN games and the tree games of Section E.2 come with seven related games, which explain other points of the same predictor and share $\nu(\emptyset)$ with the target. For the tree games, the target explains the first test row and its related games explain the next seven. For the TabPFN games, they are the seven nearest explained points after dividing each feature by its standard deviation. Each related game is observed on $b - 2$ coalitions of its own and never provides Shapley values.

Natural context. The natural context consists of the explicand with its prediction $\nu(\mathcal{P})$ and 32 reference rows with their predictions, shared by all games of a group. The reference rows are test rows for the tree games and training rows for the TabPFN games. We check that the mean reference prediction matches $\nu(\emptyset)$, so the context has the same scale as the coalition values. Section E.1 gives the cost of both inputs.

Tree games. We build the tree games with the benchmark of `shapiq` 1.7.0, which holds out 20% of the rows as test rows, fits the ensemble with default hyperparameters, and builds the game of the first test row. The related games are interventional games of `shapiq` with the same fitted model and background and a different explicand. We compute all exact values with interventional TreeSHAP of `shapiq` on inputs rounded to single precision, as in the native prediction. The predictions are the probability of class 1 for the random forest and the raw margin for XGBoost.

TabPFN games. The data loader of `shapiq` shuffles each dataset with seed 40, keeps the first rows up to the row cap, and holds out the last 20% as test rows. All 2^p masked copies of an explicand are predicted in one call, since the output of TabPFN depends on the composition of a batch, and the reference predictions are computed in the same way. The empty value $\nu(\emptyset)$ is the prediction on a row with all features missing, and all estimators and related games read the same saved values.

		Budget $2p$							**Budget $4p$**						
		ShapFM	Traditional estimators						ShapFM	Traditional estimators					
Dataset	p	no ctx.	EIG	RMSR	Perm.	SVARM	Odd	Lev.	no ctx.	EIG	RMSR	Lev.	Odd	Perm.	SVARM
ResNet-18	14	.406	.443	.664	.819	1.59	2.22	5.79	.140	.103	.233	.383	.264	.360	.991
ViT-9	9	.317	.656	.681	1.13	1.51	2.46	1.54	.122	.191	.229	.263	1.29	.436	.988
ViT-16	16	.400	.984	.937	1.49	3.22	2.78	34.5	.222	.370	.388	.615	.499	.651	2.56
DinoV2-16	16	.341	.660	.648	1.79	3.17	2.76	40.1	.179	.296	.319	.994	.567	.890	2.33
DinoV2-20	20	.417	1.02	.843	1.88	4.06	1.56	159	.241	.452	.413	1.27	.765	.931	3.03
IMDB	14	.462	.966	.630	2.37	1.33	1.57	21.1	.211	.505	.402	.644	.418	1.12	.973
NB301 abl.	16	.164	.109	.601	.155	2.22	1.35	9.17	.037	.008	.155	.185	.194	.099	1.69
NB301 tun.	16	.143	.100	.658	.116	2.71	1.60	7.64	.035	.008	.222	.138	.287	.065	2.07
RBV2 abl.	15	.048	.020	.202	.098	1.10	1.86	1.18	.007	.001	.035	.018	.026	.059	.717
RBV2 tun.	15	.053	.017	.338	.085	1.71	2.35	.496	.009	.003	.068	.016	.076	.058	1.22
SOUM 30/5	15	.277	.276	.710	.674	.946	1.69	3.52	.135	.071	.176	.181	.413	.380	.575
SOUM 30/15	15	.390	.476	.699	1.89	.634	1.92	54.2	.208	.176	.414	2.01	.724	1.05	.460
SOUM 150/5	15	.420	.329	.385	.684	.892	1.56	1.73	.131	.070	.204	.188	.377	.315	.535
SOUM 150/15	15	.462	.540	.677	1.53	.667	1.63	16.7	.184	.229	.263	1.37	.572	1.00	.495
Mean		.307	.471	.620	1.05	1.84	1.95	25.5	.133	.178	.252	.591	.462	.530	1.33
Mean rank		1.64	1.86	3.14	3.86	5.07	5.71	6.71	1.57	1.64	3.43	4.86	5.00	5.07	6.43

Table 11: NMSE at budgets of $2p$ and $4p$ evaluations (left to right) on general machine learning games: image classification, sentiment analysis, hyperparameter importance and SOUM (mean over the 10 to 100 game seeds of each game). These games were scored without context: ShapFM (no context) and every estimator see only the evaluations of the target game. We adopt the Olympic medal convention: gold, silver and bronze cells signify first, second and third best performance, respectively.

		Budget $2p$							**Budget $4p$**						
		ShapFM		Traditional estimators					ShapFM		Traditional estimators				
Dataset	p	w/ ctx.	no ctx.	RMSR	Perm.	SVARM	Lev.	Odd	w/ ctx.	no ctx.	Lev.	RMSR	Perm.	Odd	SVARM
TabPFN															
QSAR Fish	6	.075	.222	.294	.346	.614	.200	1.25	.015	.038	.028	.048	.170	.731	.266
Houses	8	.107	.319	.475	.836	.767	1.44	1.31	.029	.087	.169	.108	.378	1.10	.435
Amazon	9	.122	.203	.332	.411	.675	.642	1.80	.031	.051	.090	.080	.152	.924	.389
Food	9	.009	.020	.140	.034	.588	.097	.894	.002	.004	.003	.015	.015	.138	.307
Protein	9	.159	.349	.572	.947	.732	2.28	1.75	.079	.133	.263	.188	.465	.628	.449
HR	12	.061	.143	.320	.325	1.20	1.27	1.14	.022	.041	.053	.087	.158	.131	.817
Bank	13	.082	.298	.366	.580	.952	1.80	1.40	.022	.063	.129	.125	.268	.147	.623
Churn	19	.103	.177	.444	.501	1.59	26.3	1.99	.050	.077	.156	.116	.257	.113	1.19
Credit Card	23	.067	.163	.330	.377	1.75	205	.321	.031	.057	.124	.133	.222	.133	1.22
Coupon	24	.038	.072	.222	.184	1.32	29.1	.216	.016	.025	.030	.081	.079	.055	.922
Mean		.082	.197	.350	.454	1.02	26.8	1.21	.030	.057	.105	.098	.216	.410	.662
Mean rank		1.00	2.10	3.60	4.20	5.40	6.00	5.70	1.00	2.20	3.40	3.80	5.50	5.60	6.50
Random forest															
QSAR Fish	6	.062	.110	.163	.110	.761	.207	1.10	.004	.004	.003	.039	.049	.637	.169
Houses	8	.036	.139	.073	.200	.397	.085	.924	.005	.004	.004	.028	.030	.838	.215
Amazon	9	.368	.342	.484	2.33	.829	3.76	1.68	.105	.118	.210	.189	.680	.879	.621
Food	9	.142	.240	.296	.482	.681	1.29	1.38	.028	.027	.043	.099	.153	.656	.379
Protein	9	.227	.272	.405	1.06	1.16	2.24	2.24	.051	.053	.171	.126	.782	.756	.678
HR	12	.103	.117	.243	.286	1.04	.353	1.02	.030	.037	.060	.086	.109	.125	.679
Bank	13	.264	.310	.457	.627	.695	4.91	1.41	.050	.071	.172	.178	.357	.204	.516
Churn	19	.108	.180	.360	1.06	1.00	1.90	1.34	.024	.034	.033	.090	.472	.084	.769
Credit Card	23	.114	.150	.384	.511	1.48	48.4	.375	.055	.068	.235	.162	.463	.199	1.04
Coupon	24	.096	.129	.382	.506	1.72	6.52	.407	.040	.045	.191	.119	.112	.124	1.10
Mean		.152	.199	.325	.718	.976	6.96	1.19	.039	.046	.112	.112	.321	.450	.617
Mean rank		1.10	2.10	3.10	4.50	5.40	6.10	5.70	1.40	2.10	3.20	3.80	5.40	5.80	6.30
XGBoost															
QSAR Fish	6	.106	.142	.190	.127	.677	.221	.971	.005	.005	.004	.048	.058	.511	.139
Houses	8	.084	.159	.170	.169	.338	.106	1.11	.018	.017	.013	.043	.034	.945	.195
Amazon	9	.151	.167	.261	.332	.637	.350	1.29	.022	.022	.023	.074	.129	.712	.380
Food	9	.125	.216	.316	.084	.623	.140	1.33	.014	.015	.012	.082	.037	.855	.347
Protein	9	.364	.405	.483	1.02	.983	1.71	1.68	.141	.134	.142	.262	.822	.828	.796
HR	12	.132	.144	.278	.283	.775	.110	1.04	.014	.013	.025	.070	.088	.115	.515
Bank	13	.151	.243	.306	.231	.675	.356	1.26	.008	.008	.018	.111	.116	.119	.411
Churn	19	.108	.152	.366	.492	.736	.895	1.15	.013	.016	.013	.140	.149	.095	.520
Credit Card	23	.174	.193	.374	.413	1.17	35.2	.384	.081	.087	.076	.168	.232	.175	.821
Coupon	24	.103	.125	.336	.293	1.60	2.22	.354	.035	.039	.056	.115	.116	.101	1.01
Mean		.150	.195	.308	.345	.821	4.13	1.06	.035	.036	.038	.111	.178	.446	.513
Mean rank		1.20	2.60	3.80	3.50	5.70	4.80	6.40	1.80	2.20	2.00	4.40	5.20	6.00	6.40

Table 12: NMSE at budgets of $2p$ and $4p$ evaluations (left to right) on local explanation games of TabPFN fitted on at most 1,300 rows (mean over the 100 explained points of each dataset) and on local explanation games of random forests and XGBoost models (mean over the 30 game seeds of each dataset), on the same datasets. ShapFM receives seven related games and natural context, whereas ShapFM (no context) and every estimator see only the evaluations of the target game. We adopt the Olympic medal convention: gold, silver and bronze cells signify first, second and third best performance, respectively.

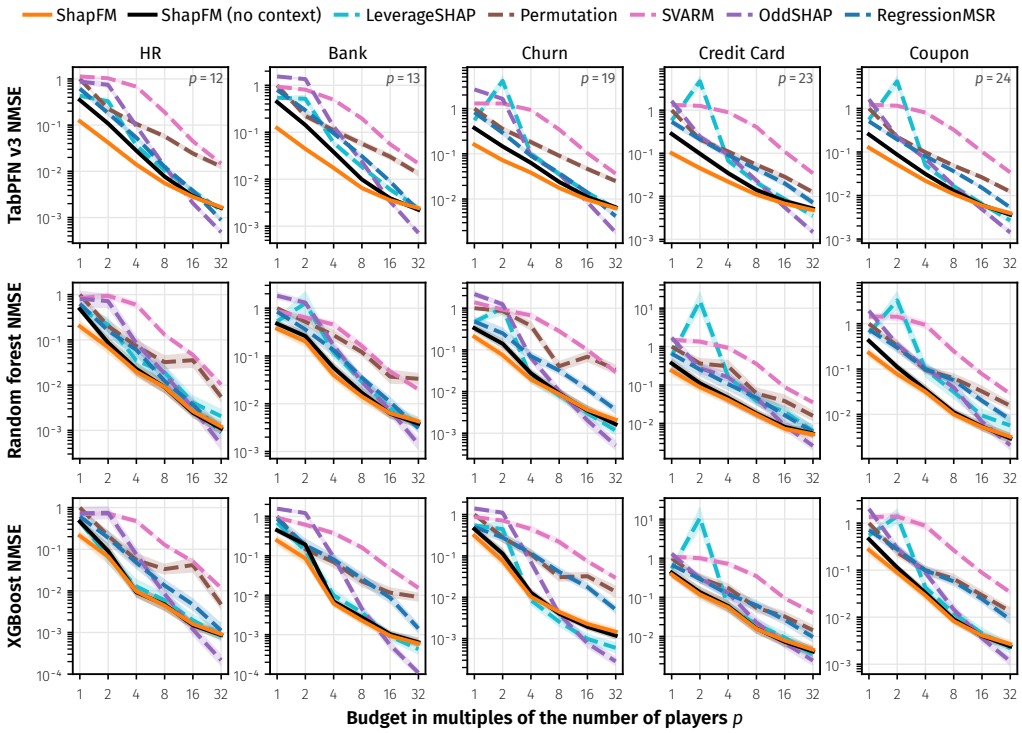

Figure 8: Approximation quality over larger budgets. NMSE over budgets of $1p$ to $32p$ for TabPFN (top, 200 explained points), random forest (middle) and XGBoost (bottom, 30 each), as geometric means with 95% confidence bands.

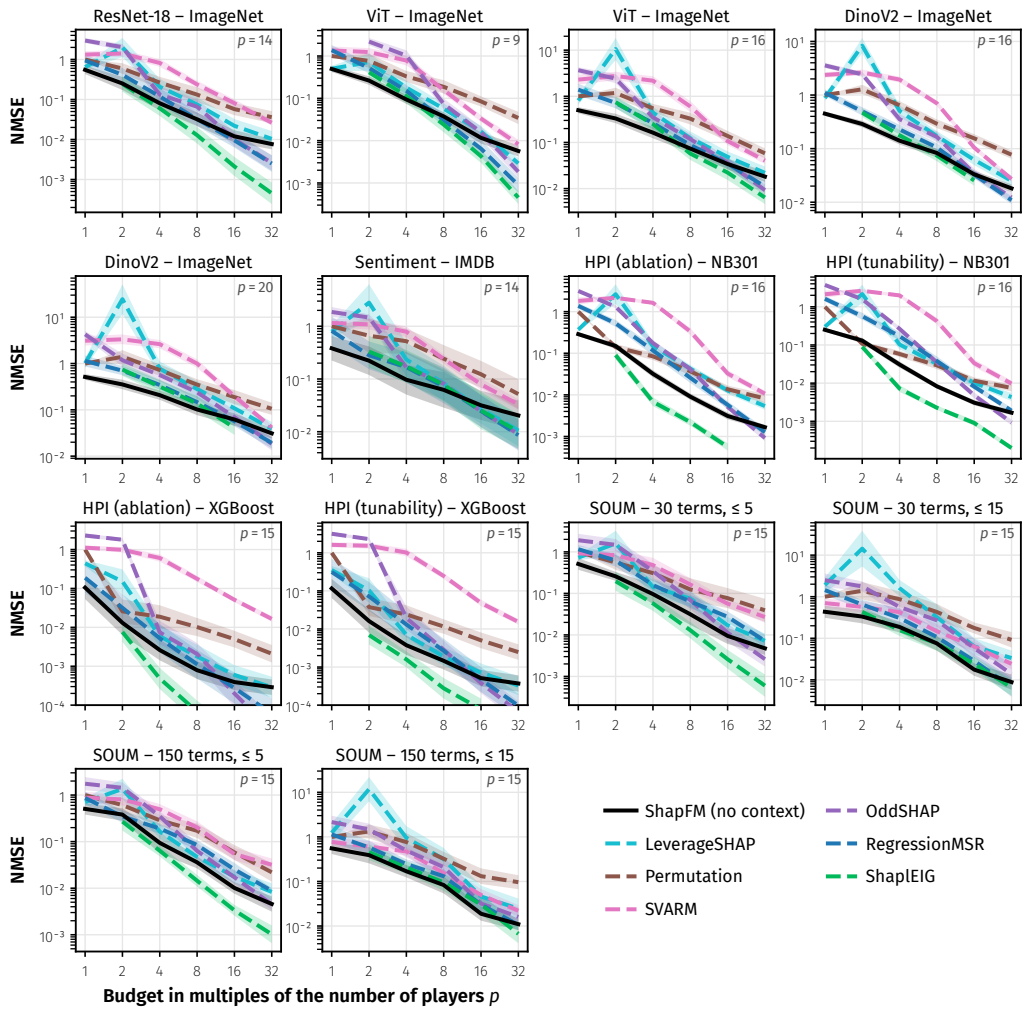

Figure 9: Approximation quality on additional games. NMSE over budgets of $1p$ to $32p$ for image classifiers on ImageNet, sentiment on IMDB, hyperparameter importance (HPI) and synthetic SOUM games, as geometric means with 95% confidence bands. ShapFM receives no context here, and ShaplEIG starts at $2p$.

E.4 Comparison with Amortized Explainers

This section gives additional information for Section 3.2.

Data and predictors. Adult Census is the version shipped by `shapiq` (Muschalik et al., 2024), whose rows we permute with a fixed seed and split $70/15/15$ into training, validation and test rows. Bank Marketing and Churn come from the benchmark registry of `shapiq`, whose test rows form the pool of explicands and whose training rows we split $85/15$ into training and validation rows. The features are not scaled, so the predictors receive the raw columns.

The random forest follows the recipe of the hyperparameter search (Section D.2), with 32 trees of depth at most 6, at least two samples per leaf, all features considered at every split, and bootstrapping. The forest is an average of trees, so interventional TreeSHAP is exact on its predicted probability. The MLP standardizes its inputs and has two hidden layers of 64 units, trained with early stopping for at most 200 epochs, and it reaches 85% validation accuracy on Adult Census. We include it because the prior contains games induced by trees, which could favor ShapFM on the forest alone.

Game and ground truth. For this game, $\nu(\emptyset)$ is the mean prediction on the 32 background rows and $\nu(\mathcal{P}) = f(\mathbf{x})$. LeverageSHAP and the exact computation evaluate the game through the marginal imputer of `shapiq` with all 32 background rows. The trained explainers need a different coalition for every row of a batch, which this imputer does not provide, so they evaluate the same game through a vectorized implementation. The two implementations agree on random coalitions up to 10^{-16}. We draw 200 explicands from the test rows of every dataset. Their exact Shapley values come from interventional TreeSHAP with the background rows as reference data for the forest and from all 2^p coalitions for the MLP, which takes about 7 seconds per explicand on Churn. Efficiency holds for every explicand up to 10^{-16} for the forest and up to 10^{-8} for the MLP, whose predictions are in single precision.

ShapFM and LeverageSHAP. All three variants of ShapFM use the model of Section D.1 with paired sampling and the sampler of Section D.1. Without context, ShapFM receives the $b-2$ coalitions with their values, $\nu(\emptyset)$ and $\nu(\mathcal{P})$. With natural context, it additionally receives the explicand and the 32 background rows with their predictions. With related games, seven other explicands of the same predictor form a group with the target. Each of them is observed on b coalitions of its own, drawn with the same sampler, and comes with its natural context. Since $\nu(\emptyset)$ is shared across the group, one explanation costs $b + 7(b-1)$ evaluations of the game. LeverageSHAP runs as implemented in `shapiq` with its default settings, which include paired sampling. The budgets are $b \in \{p+1, 32, 64, 128, 256, 512, 1024\}$, and each budget is a fresh estimate.

FastSHAP. The explainer is the network of the census example of the authors, with two hidden layers of 128 units and ReLU activations. A frozen layer in front of it standardizes the inputs with the means and standard deviations of the training rows. The layer is needed because raw tabular columns can span several orders of magnitude, five on Adult Census, while the game must keep the raw rows on which the predictor was fitted. FastSHAP originally trains a surrogate model to approximate the removal of features when the game is expensive to evaluate. Here the marginal imputer evaluates f directly on 32 hybrid rows per coalition, so we train the explainer against the very game on which it is scored and train no surrogate. Table 13 lists the training settings, which follow the census example of the authors. At inference, the explainer applies one forward pass and the additive efficient normalization of (9) with $\nu(\mathcal{P}) = f(\mathbf{x})$.

Distilled students. Each student shares the network, the training rows and the normalization of FastSHAP and differs only in its target. For every training and validation row, we draw 128 coalitions, evaluate the game on them, and let ShapFM estimate the Shapley values. The first student is taught by ShapFM without context. The second is taught by ShapFM with natural context and seven related training rows, each observed on 128 coalitions of its own as for ShapFM with related games above. Each student minimizes the mean squared error to the estimates of its teacher, with the additive efficient normalization applied inside the loss. FastSHAP and the students therefore isolate whether the estimates of ShapFM are a better training signal than the weighted least-squares objective of (8). They do not see the same evaluations of the game, since FastSHAP draws 32 new

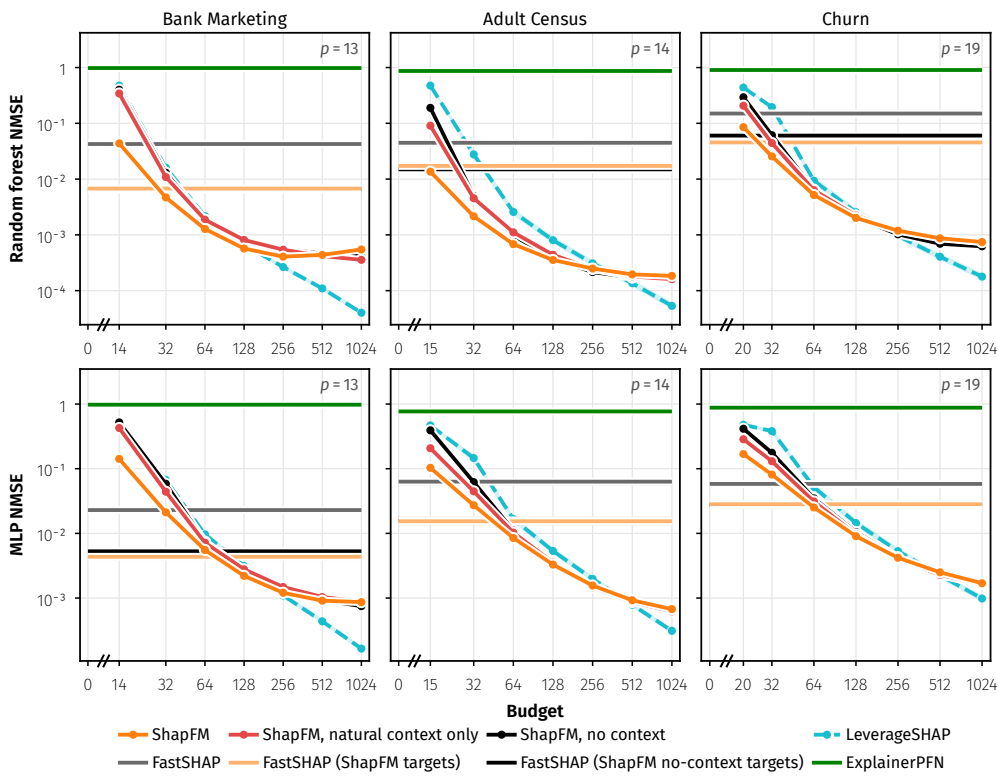

Figure 10: Detailed comparison with FastSHAP and ExplainerPFN. NMSE against exact Shapley values of the interventional game with 32 background rows, for a random forest (top) and an MLP (bottom). Curves are geometric means over 200 test explicands and three seeds, with 95% confidence bands. ShapFM uses natural context and seven related games. FastSHAP (gray) and its variants are trained once per predictor and need no evaluations of the game at inference, so they appear as levels from budget 0. FastSHAP (ShapFM targets) uses FastSHAP's network and training rows but regresses onto ShapFM's estimates at 128 evaluations per row, with context (light orange) or without context (black). ExplainerPFN (green) needs no evaluations either. It comes from a separate timing study with three explicands per case and one seed (Section E.4).

coalitions per row in every epoch and a student sees 128 coalitions per row exactly once, plus those of the related rows for the second student.

ExplainerPFN. ExplainerPFN is fitted on up to 10,000 training rows together with their predictions and predicts one contribution per feature for every explicand from $\mathbf{x}$ and $f(\mathbf{x})$. We score its raw prediction and its recommended correction, which rescales the raw contributions to the moments of Shapley values and then enforces efficiency. The authors publish no fine-tuned checkpoint, and their default loads the weights of the plain TabPFN v2 regressor, so we fine-tune a checkpoint with their recipe (Table 13). The authors describe ExplainerPFN as a proof of concept, so a weak result reflects the method as published rather than our setup.

Table 13: Training settings of the trained explainers of Section 3.2.

	Setting	Value
FastSHAP	Objective	weighted least squares of (8), normalized by (9)
	Rows	4,096 training and 100 validation rows
	Coalitions	32 per row and epoch, paired, 128 per validation row
	Batch size, epochs	32, at most 200 with early stopping after 5 epochs
	Learning rate	$2 \cdot 10^{-4}$, halved on plateau down to 10^{-5}
	Evaluations	$32 \cdot 4{,}096$ coalitions per epoch
Students	Teacher	ShapFM at 128 coalitions per row, paired, either without context or with natural context and seven related rows
	Objective	mean squared error to the teacher, normalized by (9)
	Optimizer	Adam, learning rate 10^{-3}, halved on plateau down to 10^{-5}
	Batch size, epochs	32, at most 200, best validation error
	Evaluations	$4{,}196 \cdot 129$ without and $4{,}196 \cdot 1{,}018$ with context
	Forward passes	4,196 passes of ShapFM per student
ExplainerPFN	Data	512 synthetic datasets, 4 to 32 features, 64 to 1,024 rows
	Targets	interventional Shapley values of an MLP fitted per dataset
	Initialization	TabPFN v2 regressor
	Fine-tuning	10,000 steps, Adam, learning rate 10^{-5}, cosine annealing

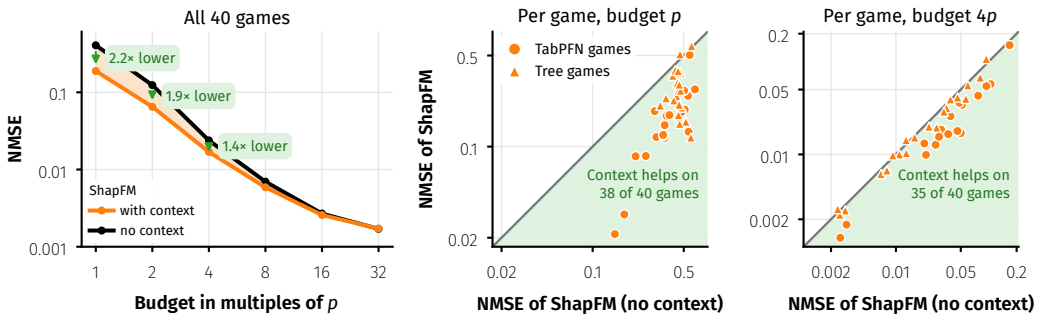

Figure 11: Context improves ShapFM at low budgets. ShapFM with context receives the natural context (the explicand and 32 background rows with their predictions) and seven related games; ShapFM (no context) receives only the observations of the target game. Left: NMSE over the budget, as the geometric mean over 40 games (TabPFN games of 10 datasets at two row caps, TabPFN fitted on at most 1,300 or 12,500 rows of the dataset, and tree games of the same datasets explained under a random forest and XGBoost); arrows give the factor by which context lowers the error. Middle and right: one point per game at budgets p and $4p$; points below the diagonal are games on which context helps. The related games cost additional queries for a single explanation and none when a batch of explicands of the same predictor is explained.

E.5 Ablation

The variants. *Flat cell encoding* replaces the two membership-selected maps $\psi_{\pm}$ of Section 2.1 by one affine map that receives the membership bit as an additional input feature, so a cell is encoded by the same weights whether or not the player is in the coalition. *Sequential stages* keeps the number and the shape of all attention blocks and only reorders them, running every coalition attention block before every player attention block, so the arm has exactly the parameters of the unchanged model. *No natural context* and *no related games* each remove one optional module of Section 2.2 and keep the other. *No context* removes both and leaves a model that sees only the coalition value matrix of the game at hand, trained on the same games as every other arm.

Protocol. All six arms share the selected configuration of Section D.1, the same pre-generated games, the same objective and the same schedule. Each arm is trained for 10,000 steps at three seeds, and all arms start from common initial weights wherever their parameters coincide, so the arms differ only in the ablated decision and in the seed. Every arm is evaluated with the most context its modules admit.

Seed variation. Figures 12 and 13 show the training curves with 95% confidence intervals across three seeds.

Why the two context modules are not comparable. Each family of the prior draws its games in related groups, so related-games attention receives a usable input on every training game. Only two of the four families provide natural context, so the natural-context encoder receives an informative input on roughly half of the training games and on half of the held-out games. The gap between the two ablations within the prior therefore reflects the composition of the prior as much as the value of the modules, and a prior in which every family provided natural context would be needed to separate the two.

Withholding inputs at inference. This experiment uses the deployed 30,000-step ShapFM without retraining. We evaluate the same model on the same TabPFN games while withholding one input at inference and normalize by the same model given everything, so a value above one is a loss. Unlike the arms, this isolates the input at test time from the length and the content of training.

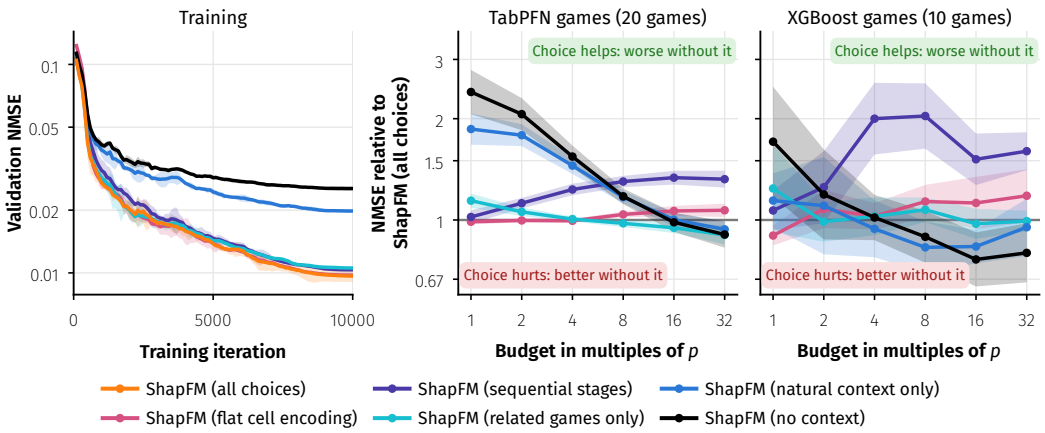

Figure 12: Ablation of ShapFM's design choices. Left: validation NMSE during training. Middle and right: NMSE of each variant relative to ShapFM (all choices) on TabPFN and XGBoost games; above one, the removed or changed choice improves performance. Bands: 95% confidence intervals. Loss curves are in Figure 13. Partly reprinted from Figure 6.

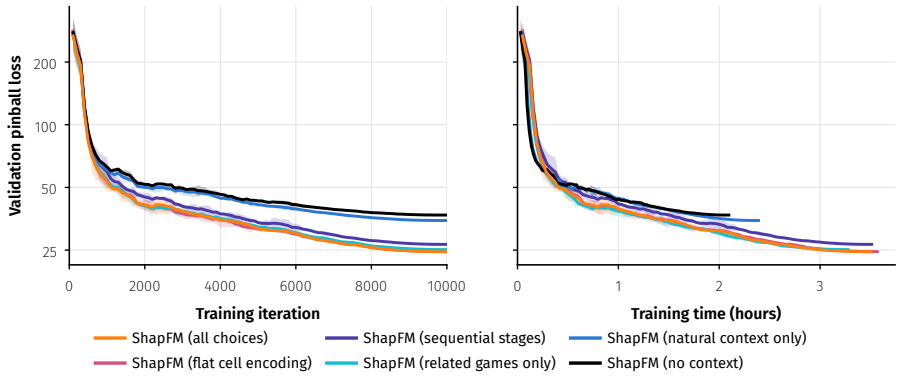

Figure 13: Validation pinball loss of the ablation variants over training iterations (left) and training time (right). Bands: 95% confidence intervals over three seeds.

F NOTATION

Table 14 collects the symbols used in the paper. Two conventions run throughout: an overline marks the exact, invertible normalization of a game, which multiplying by ν_{scale} undoes, whereas a hat marks an estimate, as in the proxy game $\hat{\nu}$ and the prediction $\hat{\phi}$.

Table 14: Notation used throughout the paper.

Symbol	Meaning
Cooperative games and Shapley values	
$\mathcal{P}$	set of players; features, hyperparameters or data points depending on the game
p	number of players, $p = \lvert\mathcal{P}\rvert$
S	a coalition, that is a subset of $\mathcal{P}$
ν	value function of a cooperative game, $\nu : 2^{\mathcal{P}} \to \mathbb{R}$
$\nu(\emptyset)$, $\nu(\mathcal{P})$	value of the empty and of the grand coalition
ϕ_i	exact Shapley value of player i (1)
$\hat{\phi}_i$	Shapley value predicted by ShapFM
$\hat{\nu}$	proxy game fitted to observed coalition values (7)
$\nu_{\mathbf{x},y}$	game induced by explicand $\mathbf{x}$ and class y (8)
b	budget of game evaluations, including $\nu(\emptyset)$ and $\nu(\mathcal{P})$
μ, β	Shapley kernel and regression coefficients of the weighted least squares view (6)
$\mathbf{1}$, $\mathbb{1}_{\{\cdot\}}$	vector of ones and indicator of an event
Interface of ShapFM	
D_m	observations of the target game: m coalitions with their values, plus $\nu(\emptyset)$ and $\nu(\mathcal{P})$
m	number of observed coalitions, $b = m + 2$
S_k, $\nu(S_k)$	the k-th observed coalition and its value
$\mathcal{C}$	optional game context: related games and natural context (Section 2.2)
$\phi_{\theta,\text{ShapFM}}$	the ShapFM network (2)
$\phi_{\theta,\text{FastSHAP}}$	the FastSHAP explainer (8)
θ	network parameters
τ_q, Q	quantile levels and their number
Normalization	
ν_{scale}	normalization scale, the largest absolute shifted value
$\overline{\nu}$, $\overline{\nu}(S_k)$	normalized game and normalized value of coalition S_k
$\overline{\phi}_i$	Shapley value of the normalized game; $\phi_i = \nu_{\text{scale}} \cdot \overline{\phi}_i$
$\hat{\phi}_i^{\text{raw}}$	mean of the raw quantiles, before additive efficient normalization
Architecture	
$\mathbf{e}_{k,i}$	embedding of the cell for coalition S_k and player i (3)
ψ_+, ψ_-	value embeddings used when player i is present and when it is absent
ψ_s	embedding of the coalition size
s_k	size of coalition S_k, $s_k = \lvert S_k\rvert$
X_i	column of player i, its cells over all observed coalitions
u_i, u	per-player and per-game vectors produced by the natural-context encoder
d, D	width of a cell embedding and of the natural-context encoder
L, K	interaction rounds and inducing points of the induced set attention block
Explained model and natural context	
f	the explained predictor
$\mathbf{x}$, y	explicand and explained class
$\mathbf{z}_r$, R	r-th background row and the number of background rows
Z	matrix of the background rows
$\mathcal{X}$, $\mathcal{Y}$	input and label space
Prior	
κ, η	prior family and its hyperparameters

continued on the next page

Table 14, continued

Symbol	Meaning
π_{fam}, π_m	distribution over families and over budgets
G, n_{grp}, B	games per group, groups per batch, and batch size
j	index of a game within a batch
$\mathcal{T}$, T, c_T, $c_\emptyset$	support, one subset, and the coefficients of the Harsanyi basis
k, ℓ_i	Hamming kernel of the Gaussian process family and the lengthscale of player i
α	fraction of the variance that the games of a group share
δ, σ	per-game jitter of the coefficients and of the explicand
$X_1, \ldots, X_p$, Y	feature nodes and sink of the causal family
Training and evaluation	
ρ_τ	pinball loss at level τ
g_j, λ_{eff}	efficiency gap of game j and the weight of its penalty
NMSE	normalized mean squared error, $\|\hat{\phi} - \phi\|^2 / \|\phi\|^2$