Training Characteristic Functions with Reinforcement Learning

Stephan Wäldchen, Felix Huber, Sebastian Pokutta

Zuse Institut Berlin

39th International Conference on Machine Learning
July 17 – 23, 2022

How do we interpret black-box Functions?

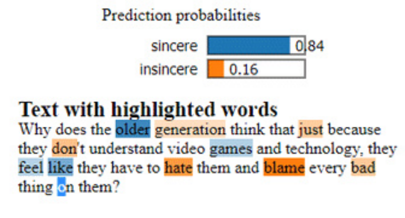
How do we interpret black-box Functions?

- Core idea: Probe black-box function with different inputs

How do we interpret black-box Functions?

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features

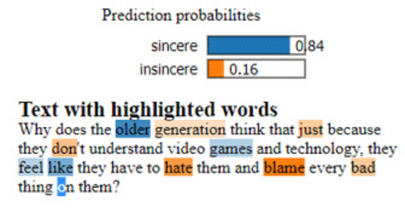


LIME saliency[1]

1) "Explainable ai: A review of machine learning interpretability methods", Linardatos et al. [7]

How do we interpret black-box Functions?

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features
- ML: Saliency $\leftrightarrow$ Coop. Game Theory: Surplus Attribution

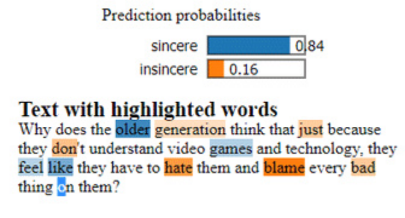


LIME saliency[1]

1) "Explainable ai: A review of machine learning interpretability methods", Linardatos et al. [7]

How do we interpret black-box Functions?

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features
- ML: Saliency $\leftrightarrow$ Coop. Game Theory: Surplus Attribution
- Characteristic function: $\nu \colon 2^{[d]} \to \mathbb{R}$

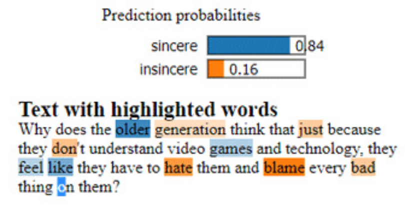


LIME saliency[1]

1) "Explainable ai: A review of machine learning interpretability methods", Linardatos et al. [7]

How do we interpret black-box Functions?

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features
- ML: Saliency $\leftrightarrow$ Coop. Game Theory: Surplus Attribution
- Characteristic function: $\nu\colon 2^{[d]} \to \mathbb{R}$



LIME saliency[1]

Source: https://clearcode.cc/blog/game-theory-attribution/

How do we interpret black-box Functions?

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features
- ML: Saliency $\leftrightarrow$ Coop. Game Theory: Surplus Attribution
- Characteristic function: $\nu\colon 2^{[d]} \to \mathbb{R}$
- Prime Implicant Explanation[2] (mostly for $\nu : 2^{[d]} \to \{0, 1\}$)

$$S^* = \operatorname*{argmin}_{S \subset [d]} |S| \quad \text{s.t.} \quad \nu(S) = \nu([d])$$

Source: https://clearcode.cc/blog/game-theory-attribution/

1) "Explainable ai: A review of machine learning interpretability methods", Linardatos et al. [7] 2) "A symbolic approach to explaining bayesian network classifiers", Shih et al. [13]

How do we interpret black-box Functions?

Source: https://clearcode.cc/blog/game-theory-attribution/

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features
- ML: Saliency $\leftrightarrow$ Coop. Game Theory: Surplus Attribution
- Characteristic function: $\nu \colon 2^{[d]} \to \mathbb{R}$
- Prime Implicant Explanation[2] (mostly for $\nu : 2^{[d]} \to \{0,1\}$)

$$S^* = \operatorname*{argmin}_{S \subset [d]} |S| \quad \text{s.t.} \quad \nu(S) = \nu([d])$$

- Shapley Values[3] (linear, efficient, symmetric, null-player)

$$\phi_i = \frac{1}{d!} \sum_{\pi \in \Pi([d])} \left(\nu(P_i^\pi \cup \{i\}) - \nu(P_i^\pi)\right).$$

1) "Explainable ai: A review of machine learning interpretability methods", Linardatos et al. [7] 2) "A symbolic approach to explaining bayesian network classifiers", Shih et al. [13] 3) "A value for n-person games" Shapley [12]

- Core idea: Probe black-box function with different inputs
- Local interpretation (for specific input $\mathbf{x}$): vary features
- ML: Saliency $\leftrightarrow$ Coop. Game Theory: Surplus Attribution
- Characteristic function: $\nu \colon 2^{[d]} \to \mathbb{R}$
- Prime Implicant Explanation[2] (mostly for $\nu : 2^{[d]} \to \{0,1\}$)

$$S^* = \underset{S \subset [d]}{\operatorname{argmin}} |S| \quad \text{s.t.} \quad \nu(S) = \nu([d])$$

- Shapley Values[3] (linear, efficient, symmetric, null-player)

$$\phi_i = \frac{1}{d!} \sum_{\pi \in \Pi([d])} (\nu(P_i^\pi \cup \{i\}) - \nu(P_i^\pi)).$$

- Problem: We don't have characteristic functions!

Source: https://clearcode.cc/blog/game-theory-attribution/

1) "Explainable ai: A review of machine learning interpretability methods", Linardatos et al. [7] 2) "A symbolic approach to explaining bayesian network classifiers", Shih et al. [13] 3) "A value for n-person games" Shapley [12]

Interpretations Rely on a Model of the Data Manifold

Interpretations Rely on a Model of the Data Manifold

- Approach from Lundberg et al[1]:

$$\nu_{\Phi,\mathbf{x}}(S) = \mathbb{E}_{\mathbf{y}}[\Phi(\mathbf{y}) \,|\, \mathbf{y}_S = \mathbf{x}_S] = \int \Phi(\mathbf{x}) \mathrm{d}\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S].$$

1) "A unified approach to interpreting model predictions", Lundberg et al, [8]

Interpretations Rely on a Model of the Data Manifold

- Approach from Lundberg et al[1]:

$$\nu_{\Phi,\mathbf{x}}(S) = \mathbb{E}_{\mathbf{y}}[\Phi(\mathbf{y}) \mid \mathbf{y}_S = \mathbf{x}_S] = \int \Phi(\mathbf{x}) \mathrm{d}\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S].$$

- Needs good model of $\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S]$!

1) "A unified approach to interpreting model predictions", Lundberg et al, [8]

Interpretations Rely on a Model of the Data Manifold

- Approach from Lundberg et al[1]:

$$\nu_{\Phi,\mathbf{x}}(S) = \mathbb{E}_{\mathbf{y}}[\Phi(\mathbf{y}) \mid \mathbf{y}_S = \mathbf{x}_S] = \int \Phi(\mathbf{x}) \mathrm{d}\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S].$$

- Needs good model of $\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S]$!
- Most methods simply approximate with baseline values (sometimes layer-wise)

1) "A unified approach to interpreting model predictions", Lundberg et al, [8]

Interpretations Rely on a Model of the Data Manifold

- Approach from Lundberg et al[1]:

$$\nu_{\Phi,\mathbf{x}}(S) = \mathbb{E}_{\mathbf{y}}[\Phi(\mathbf{y}) \,|\, \mathbf{y}_S = \mathbf{x}_S] = \int \Phi(\mathbf{x})\mathrm{d}\mathbb{P}[\mathbf{x}_{S^c}|\mathbf{x}_S].$$

- Needs good model of $\mathbb{P}[\mathbf{x}_{S^c}|\mathbf{x}_S]$!
- Most methods simply approximate with baseline values (sometimes layer-wise)
- Change off-manifold behaviour to manipulate: Gradient, Integrated gradients[2,3], LRP[2,4,7], LIME[3,5], DeepShap[3,5], Grad-Cam[7], Shapley-based[6], Counterfactual explanations[8],

1) "A unified approach to interpreting model predictions", Lundberg et al, [8] 2) "Fairwashing explanations with off-manifold detergent", Anders et al. [1] 3) "You Shouldn't Trust Me: Learning Models Which Conceal Unfairness From Multiple Explanation Methods", Dimanov et al. [3] 4) "Explanations can be manipulated and geometry is to blame" Dombrowski et al. [4] 5) "Fooling lime and shap: Adversarial attacks on post hoc explanation methods", Slack et al. [14] 6) "Shapley explainability on the data manifold", Frye et al. [5] 7) "Fooling neural network interpretations via adversarial model manipulation" Heo et al. [6] 8) "Counterfactual Explanations Can Be Manipulated" Slack et al. [15]

Interpretations Rely on a Model of the Data Manifold

- Approach from Lundberg et al[1]:

$$\nu_{\Phi,\mathbf{x}}(S) = \mathbb{E}_{\mathbf{y}}[\Phi(\mathbf{y}) \,|\, \mathbf{y}_S = \mathbf{x}_S] = \int \Phi(\mathbf{x}) d\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S].$$

- Needs good model of $\mathbb{P}[\mathbf{x}_{S^c}|\mathbf{x}_S]$!
- Most methods simply approximate with baseline values (sometimes layer-wise)
- Change off-manifold behaviour to manipulate: Gradient, Integrated gradients[2,3], LRP[2,4,7], LIME[3,5], DeepShap[3,5], Grad-Cam[7], Shapley-based[6], Counterfactual explanations[8],

- Best Performer RDE creates new features![8]

1) "A unified approach to interpreting model predictions", Lundberg et al, [8] 2) "Fairwashing explanations with off-manifold detergent", Anders et al. [1] 3) "You Shouldn't Trust Me: Learning Models Which Conceal Unfairness From Multiple Explanation Methods", Dimanov et al. [3] 4) "Explanations can be manipulated and geometry is to blame" Dombrowski et al. [4] 5) "Fooling lime and shap: Adversarial attacks on post hoc explanation methods", Slack et al. [14] 6) "Shapley explainability on the data manifold", Frye et al. [5] 7) "Fooling neural network interpretations via adversarial model manipulation" Heo et al. [6] 8) "Counterfactual Explanations Can Be Manipulated" Slack et al. [15] 9) "Interpretable Neural Networks with Frank-Wolfe: Sparse Relevance Maps and Relevance Orderings", Macdonald et al. [9]

Interpretations Rely on a Model of the Data Manifold

- Approach from Lundberg et al[1]:

$$\nu_{\Phi,\mathbf{x}}(S) = \mathbb{E}_{\mathbf{y}}[\Phi(\mathbf{y}) \,|\, \mathbf{y}_S = \mathbf{x}_S] = \int \Phi(\mathbf{x}) \mathrm{d}\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S].$$

- Needs good model of $\mathbb{P}[\mathbf{x}_{S^c} | \mathbf{x}_S]$!
- Most methods simply approximate with baseline values (sometimes layer-wise)
- Change off-manifold behaviour to manipulate: Gradient, Integrated gradients[2,3], LRP[2,4,7], LIME[3,5], DeepShap[3,5], Grad-Cam[7], Shapley-based[6], Counterfactual explanations[8],

- Idea: Directly train a characteristic function!

1) "A unified approach to interpreting model predictions", Lundberg et al, [8] 2) "Fairwashing explanations with off-manifold detergent", Anders et al. [1] 3) "You Shouldn't Trust Me: Learning Models Which Conceal Unfairness From Multiple Explanation Methods", Dimanov et al. [3] 4) "Explanations can be manipulated and geometry is to blame" Dombrowski et al. [4] 5) "Fooling lime and shap: Adversarial attacks on post hoc explanation methods", Slack et al. [14] 6) "Shapley explainability on the data manifold", Frye et al. [5] 7) "Fooling neural network interpretations via adversarial model manipulation" Heo et al. [6] 8) "Counterfactual Explanations Can Be Manipulated" Slack et al. [15] 9) "Interpretable Neural Networks with Frank-Wolfe: Sparse Relevance Maps and Relevance Orderings", Macdonald et al. [9]

Setup: Connect Four with hidden colour information

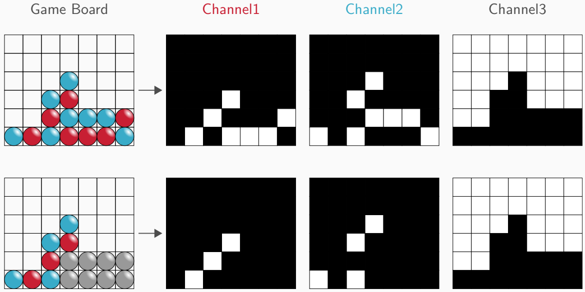
Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional

1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

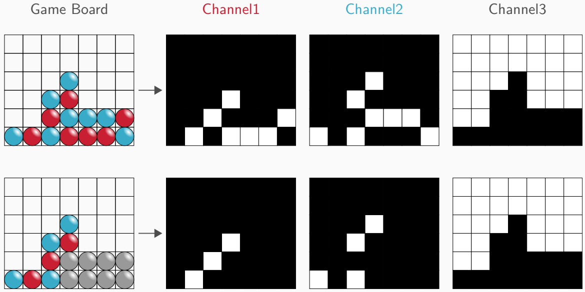
- Abstract Games: complex, yet low-dimensional



1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

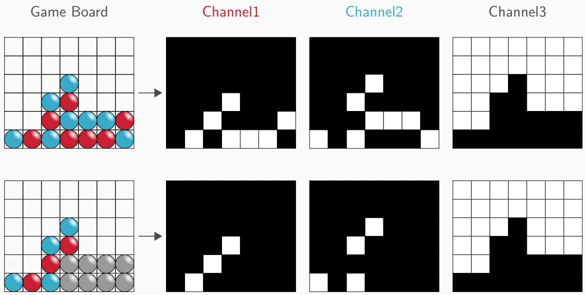
- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$



1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

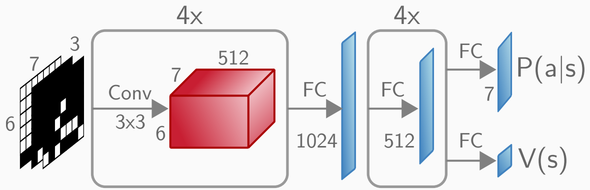
- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random



1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

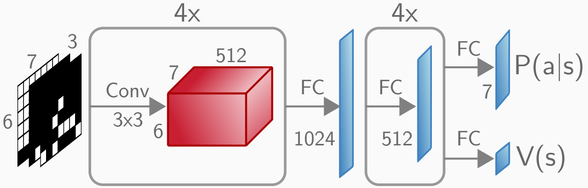
- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random



1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random
- Train agent with Proximal Policy Optimisation (PPO)[1,2]

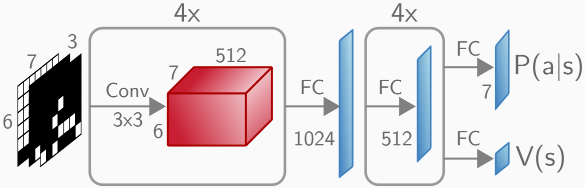


1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random
- Train agent with Proximal Policy Optimisation (PPO)[1,2]

♦ FI: Full Information.



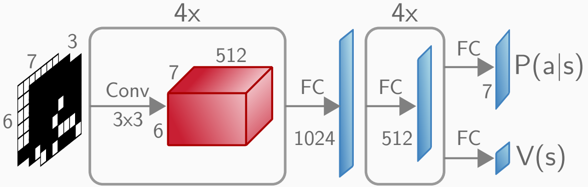
1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random
- Train agent with Proximal Policy Optimisation (PPO)[1,2]

♦ FI: Full Information.

♦ PI-50: with $p_h \sim \mathcal{U}([0, 0.5])$

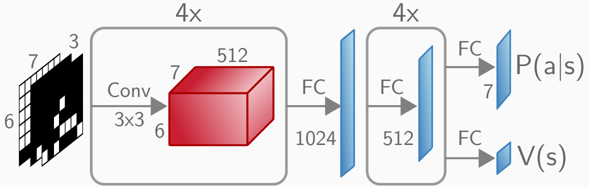


1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random
- Train agent with Proximal Policy Optimisation (PPO)[1,2]

- FI: Full Information.
- PI-50: with $p_h \sim \mathcal{U}([0, 0.5])$
- PI-100: with $p_h \sim \mathcal{U}([0, 1])$

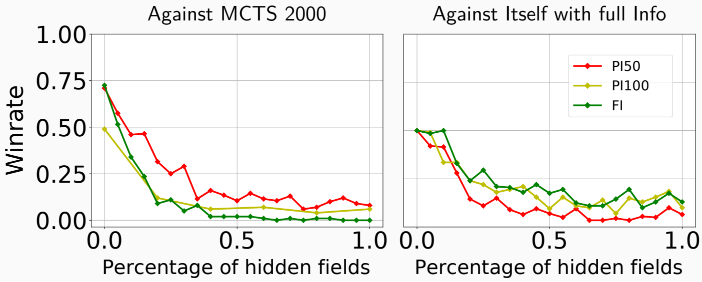


1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random
- Train agent with Proximal Policy Optimisation (PPO)[1,2]

- FI: Full Information.
- PI-50: with $p_h \sim \mathcal{U}([0, 0.5])$
- PI-100: with $p_h \sim \mathcal{U}([0, 1])$

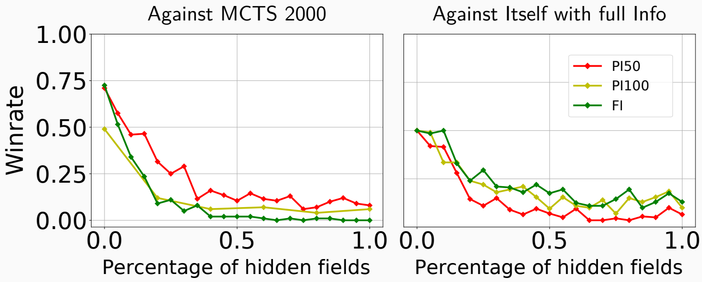


1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Setup: Connect Four with hidden colour information

- Abstract Games: complex, yet low-dimensional
- Every turn t: $p_h \sim \mathcal{U}([0, p_h^{\max}])$
- Hide $\lfloor p_h t \rfloor$ colour features at random
- Train agent with Proximal Policy Optimisation (PPO)[1,2]

- FI: Full Information.
- PI-50: with $p_h \sim \mathcal{U}([0, 0.5])$
- PI-100: with $p_h \sim \mathcal{U}([0, 1])$



1) "Proximal policy optimization algorithms", Schulman et al [11] 2) "Reinforcement Learning for Two-Player Zero-Sum Games", Crespo [2]

Interpretability with Characteristic Functions

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times6\times7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden
- Let furthermore $a^* = \operatorname{argmax}_a P(a\,;\mathbf{x})$

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden
- Let furthermore $a^* = \operatorname{argmax}_a P(a\,;\mathbf{x})$
- We define $\nu^{\text{pol}} : 2^{[t]} \to [0,1]$ and $\nu^{\text{val}} : 2^{[t]} \to [-1,1]$ as

$$\nu^{\text{pol}}(S) = P(a^*; \mathbf{x}^{(S)}) \quad \text{and} \quad \nu^{\text{val}}(S) = V(\mathbf{x}^{(S)}).$$

1) “Deep Neural Network Training with Frank-Wolfe”, Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden
- Let furthermore $a^* = \operatorname{argmax}_a P(a\,;\mathbf{x})$
- We define $\nu^{\text{pol}} : 2^{[t]} \to [0,1]$ and $\nu^{\text{val}} : 2^{[t]} \to [-1,1]$ as

$$\nu^{\text{pol}}(S) = P(a^*; \mathbf{x}^{(S)}) \quad \text{and} \quad \nu^{\text{val}}(S) = V(\mathbf{x}^{(S)}).$$

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden
- Let furthermore $a^* = \operatorname{argmax}_a P(a\,;\mathbf{x})$
- We define $\nu^{\text{pol}} : 2^{[t]} \to [0,1]$ and $\nu^{\text{val}} : 2^{[t]} \to [-1,1]$ as

$$\nu^{\text{pol}}(S) = P(a^*; \mathbf{x}^{(S)}) \quad \text{and} \quad \nu^{\text{val}}(S) = V(\mathbf{x}^{(S)}).$$

- We can approximate the Shapley sum by sampling from $\mathcal{U}(\Pi([t]))$:

$$\phi_i = \frac{1}{t!} \sum_{\pi \in \Pi([t])^{\text{sample}}} \left(\nu(P_i^\pi \cup \{i\}) - \nu(P_i^\pi)\right).$$

1) “Deep Neural Network Training with Frank-Wolfe”, Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden
- Let furthermore $a^* = \operatorname{argmax}_a P(a\,;\mathbf{x})$
- We define $\nu^{\text{pol}} : 2^{[t]} \to [0,1]$ and $\nu^{\text{val}} : 2^{[t]} \to [-1,1]$ as

$$\nu^{\text{pol}}(S) = P(a^*; \mathbf{x}^{(S)}) \quad \text{and} \quad \nu^{\text{val}}(S) = V(\mathbf{x}^{(S)}).$$

- We can approximate the Shapley sum by sampling from $\mathcal{U}(\Pi([t]))$:

$$\phi_i = \frac{1}{t!} \sum_{\pi \in \Pi([t])^{\text{sample}}} \left(\nu(P_i^\pi \cup \{i\}) - \nu(P_i^\pi)\right).$$

- $\mathbb{P}\left[\left|\phi_i - \bar{\phi}_i\right| \leq \epsilon\right] \geq 1 - \delta \to (0.01, 0.01)$-approximation ≈26 500 samples (Hoeffding)

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Interpretability with Characteristic Functions

- Let $t \in [42]$, $\mathbf{x} \in [0,1]^{3\times 6\times 7}$, $S \in [t]$ and let $\mathbf{x}^{(S)}$ be state with colour feature on S^c hidden
- Let furthermore $a^* = \operatorname{argmax}_a P(a\,;\mathbf{x})$
- We define $\nu^{\text{pol}} : 2^{[t]} \to [0,1]$ and $\nu^{\text{val}} : 2^{[t]} \to [-1,1]$ as

$$\nu^{\text{pol}}(S) = P(a^*;\mathbf{x}^{(S)}) \quad \text{and} \quad \nu^{\text{val}}(S) = V(\mathbf{x}^{(S)}).$$

- We can approximate the Shapley sum by sampling from $\mathcal{U}(\Pi([t]))$:

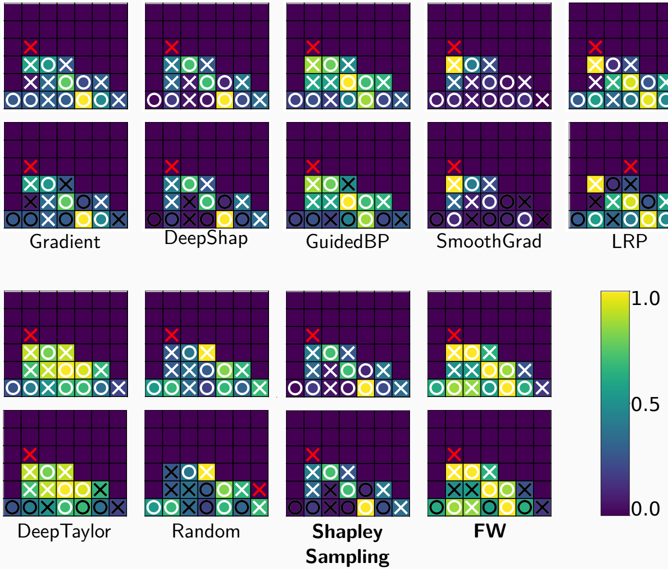
$$\phi_i = \frac{1}{t!} \sum_{\pi \in \Pi([t])^{\text{sample}}} \left(\nu(P_i^\pi \cup \{i\}) - \nu(P_i^\pi)\right).$$

- $\mathbb{P}\left[\left|\phi_i - \bar{\phi}_i\right| \leq \epsilon\right] \geq 1 - \delta \to (0.01, 0.01)$-approximation $\approx$26 500 samples (Hoeffding)
- Calculate PIE with Frank-Wolfe optimiser solving[1] convex relaxation of

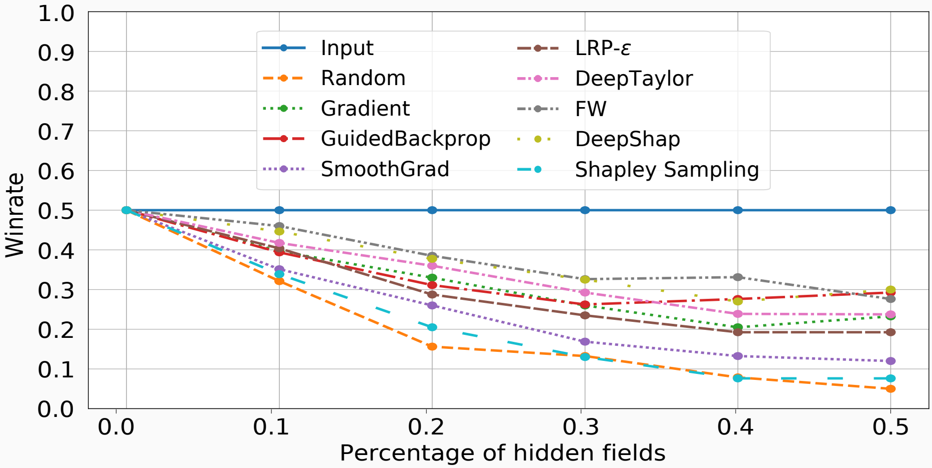
$$S^* = \operatorname*{argmin}_{|S| \leq \lfloor p_h t \rfloor} \left(\nu([t]) - \nu(S)\right)^2.$$

1) "Deep Neural Network Training with Frank-Wolfe", Pokutta et al. [10]

Example Saliencies for different Methods

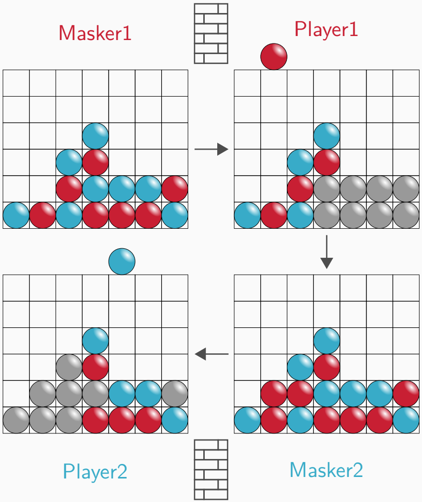


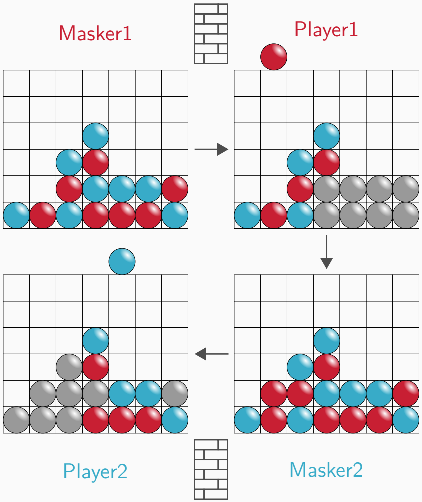
Information-Performance Comparison



Round-Robin Tournament

Round-Robin Tournament





	Input	DeepShap	Guided Backprop	FW	Gradient	LRP-ε	Deep Taylor	Smooth Grad	Random
Input		0.737	0.747	0.706	0.782	0.834	0.774	0.887	0.935
DeepShap	0.263		0.498	0.499	0.573	0.609	0.567	0.742	0.871
Guided Backprop	0.254	0.502		0.496	0.579	0.616	0.616	0.765	0.861
FW	0.294	0.501	0.503		0.587	0.604	0.586	0.742	0.848
Gradient	0.217	0.427	0.421	0.413		0.512	0.502	0.686	0.819
LRP-ε	0.167	0.392	0.385	0.397	0.487		0.472	0.677	0.810
Deep Taylor	0.226	0.433	0.384	0.414	0.497	0.528		0.681	0.805
Smooth Grad	0.113	0.259	0.235	0.258	0.314	0.323	0.319		0.676
Random	0.065	0.130	0.140	0.152	0.181	0.190	0.195	0.324	

Limitations and Outlook

Limitations and Outlook

- So far works only for certain abstract games (Connect Four, Hex, Go, ...)

Limitations and Outlook

- So far works only for certain abstract games (Connect Four, Hex, Go, ...)

$\Rightarrow$ Filter out illegal moves with model-based approaches (see e.g. AlphaGo)

Limitations and Outlook

- So far works only for certain abstract games (Connect Four, Hex, Go, ...)

$\Rightarrow$ Filter out illegal moves with model-based approaches (see e.g. AlphaGo)

- Further extension to real-world tasks (high-dimensional, high redundancy) is challenging

Limitations and Outlook

- So far works only for certain abstract games (Connect Four, Hex, Go, ...)

⇒ Filter out illegal moves with model-based approaches (see e.g. AlphaGo)

- Further extension to real-world tasks (high-dimensional, high redundancy) is challenging

⇒ Our approach is should be used primarily for evaluation of saliency methods

Limitations and Outlook

- ▶ So far works only for certain abstract games (Connect Four, Hex, Go, ...)
- $\Rightarrow$ Filter out illegal moves with model-based approaches (see e.g. AlphaGo)
- ▶ Further extension to real-world tasks (high-dimensional, high redundancy) is challenging
- $\Rightarrow$ Our approach is should be used primarily for evaluation of saliency methods
- ▶ Shapley sampling suffered from unstable policy layer for large hidden information

Limitations and Outlook

- So far works only for certain abstract games (Connect Four, Hex, Go, ...)

⇒ Filter out illegal moves with model-based approaches (see e.g. AlphaGo)

- Further extension to real-world tasks (high-dimensional, high redundancy) is challenging

⇒ Our approach is should be used primarily for evaluation of saliency methods

- Shapley sampling suffered from unstable policy layer for large hidden information

⇒ Train value function instead

Limitations and Outlook

- So far works only for certain abstract games (Connect Four, Hex, Go, ...)

⇒ Filter out illegal moves with model-based approaches (see e.g. AlphaGo)

- Further extension to real-world tasks (high-dimensional, high redundancy) is challenging

⇒ Our approach is should be used primarily for evaluation of saliency methods

- Shapley sampling suffered from unstable policy layer for large hidden information

⇒ Train value function instead

⇒ Q-Learning could be a more stable approach

Conclusion:

Conclusion:

- Interpretability relies on a good model of the data distribution

Conclusion:

- Interpretability relies on a good model of the data distribution
- We can design proxy-task where we know the distribution via abstract games with missing information

Conclusion:

- Interpretability relies on a good model of the data distribution
- We can design proxy-task where we know the distribution via abstract games with missing information
- Use these tasks to evaluate saliency methods without going off-manifold

Conclusion:

- Interpretability relies on a good model of the data distribution
- We can design proxy-task where we know the distribution via abstract games with missing information
- Use these tasks to evaluate saliency methods without going off-manifold

Thank You!

Conclusion:

- Interpretability relies on a good model of the data distribution
- We can design proxy-task where we know the distribution via abstract games with missing information
- Use these tasks to evaluate saliency methods without going off-manifold

Thank You!

Contact: `waeldchen@zib.de`

Paper: *Training Characteristic Functions with Reinforcement Learning: XAI-methods play Connect Four*, S Wäldchen, F Huber, S Pokutta *arXiv preprint* arXiv:2202.11797

C. Anders, P. Pasliev, A.-K. Dombrowski, K.-R. Müller, and P. Kessel.
Fairwashing explanations with off-manifold detergent.
In *International Conference on Machine Learning*, pages 314–323. PMLR, 2020.

J. Crespo.
Reinforcement learning for two-player zero-sum games.
Master's thesis, Tecnico Lisboa, https://fenix.tecnico.ulisboa.pt/downloadFile/1689244997260153/81811-joao-crespo_dissertacao.pdf, 2019.

B. Dimanov, U. Bhatt, M. Jamnik, and A. Weller.
You shouldn't trust me: Learning models which conceal unfairness from multiple explanation methods.
In *SafeAI@ AAAI*, 2020.

A.-K. Dombrowski, M. Alber, C. J. Anders, M. Ackermann, K.-R. Müller, and P. Kessel.
Explanations can be manipulated and geometry is to blame.
arXiv preprint arXiv:1906.07983, 2019.

C. Frye, D. de Mijolla, T. Begley, L. Cowton, M. Stanley, and I. Feige.
Shapley explainability on the data manifold.
arXiv preprint arXiv:2006.01272, 2020.

J. Heo, S. Joo, and T. Moon.
Fooling neural network interpretations via adversarial model manipulation.
Advances in Neural Information Processing Systems, 32:2925–2936, 2019.

P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis.
Explainable ai: A review of machine learning interpretability methods.
Entropy, 23(1):18, 2020.

S. M. Lundberg and S.-I. Lee.
A unified approach to interpreting model predictions.
In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

J. Macdonald, M. Besançon, and S. Pokutta.
Interpretable neural networks with frank-wolfe: Sparse relevance maps and relevance orderings.
arXiv preprint arXiv:2110.08105, 2021.

S. Pokutta, C. Spiegel, and M. Zimmer.
Deep neural network training with frank-wolfe.
arXiv preprint arXiv:2010.07243, 2020.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov.
Proximal policy optimization algorithms.
arXiv preprint arXiv:1707.06347, 2017.

L. S. Shapley.
17. A value for n-person games.
Princeton University Press, 2016.

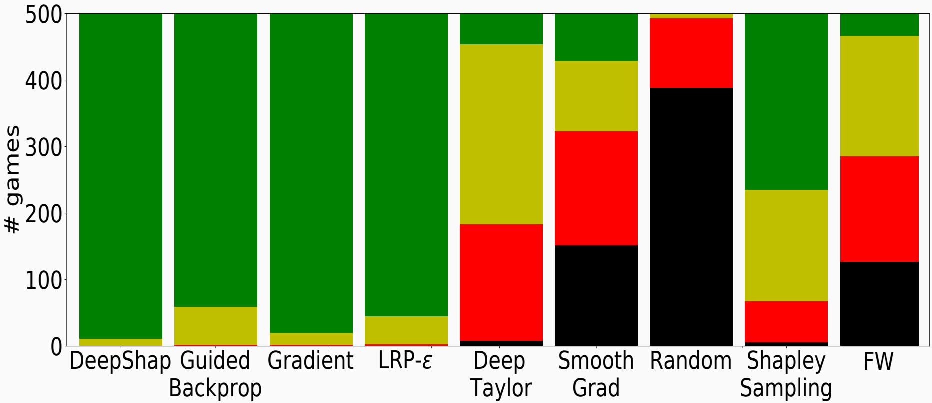
A. Shih, A. Choi, and A. Darwiche.
A symbolic approach to explaining bayesian network classifiers.
arXiv preprint arXiv:1805.03364, 2018.

D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju.
Fooling lime and shap: Adversarial attacks on post hoc explanation methods.
In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.

D. Slack, S. Hilgard, H. Lakkaraju, and S. Singh.
Counterfactual explanations can be manipulated.
arXiv preprint arXiv:2106.02666, 2021.

Appendix

Ground Truth Comparison: Winning Move



Tournament: Standard Deviation and Illegal Move Rate

	Input	DeepShap	Guided Backprop	FW	Gradient	LRP-ε	Deep Taylor	Smooth Grad	Random
Input		0.0188	0.0199	0.0233	0.0225	0.0220	0.0172	0.0152	0.0114
DeepShap	0.0188		0.0173	0.0201	0.0204	0.0144	0.0284	0.0204	0.0196
Guided Backprop	0.0199	0.0173		0.0192	0.0155	0.0202	0.0288	0.0229	0.0097
FW	0.0233	0.0201	0.0192		0.0217	0.0135	0.0263	0.0185	0.0091
Gradient	0.0225	0.0204	0.0155	0.0217		0.0146	0.0173	0.0233	0.0152
LRP-ε	0.0220	0.0144	0.0202	0.0135	0.0146		0.0164	0.0166	0.0142
Deep Taylor	0.0172	0.0284	0.0288	0.0263	0.0173	0.0164		0.0229	0.0159
Smooth Grad	0.0152	0.0204	0.0229	0.0185	0.0233	0.0166	0.0229		0.0164
Random	0.0114	0.0196	0.0097	0.0091	0.0152	0.0142	0.0159	0.0164	

	Input	DeepShap	Guided Backprop	FW	Gradient	LRP-ε	Deep Taylor	Smooth Grad	Random
Input		0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.01
DeepShap	0.02		0.02	0.01	0.02	0.01	0.02	0.01	0.01
Guided Backprop	0.06	0.05		0.04	0.03	0.03	0.03	0.01	0.02
FW	0.06	0.03	0.04		0.02	0.04	0.02	0.01	0.01
Gradient	0.02	0.02	0.02	0.01		0.02	0.02	0.01	0.01
LRP-ε	0.02	0.03	0.01	0.01	0.01		0.01	0.01	0.01
Deep Taylor	0.03	0.03	0.03	0.03	0.02	0.03		0.01	0.01
Smooth Grad	0.01	0.00	0.01	0.00	0.01	0.00	0.01		0.00
Random	0.00	0.01	0.01	0.01	0.01	0.00	0.01	0.01	