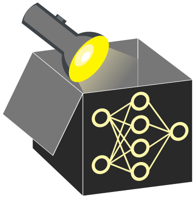
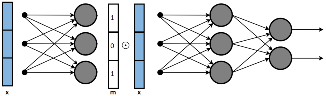
Interpretable Machine Learning



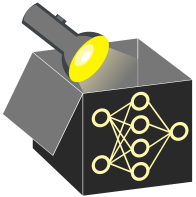
Learning to Explain



Learning goals

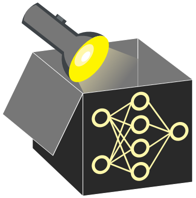
- Optimization problems in hard-masking
- Generative masking
- Sampling-based instance-wise feature selection

INSTANCE-WISE FEATURE SELECTION

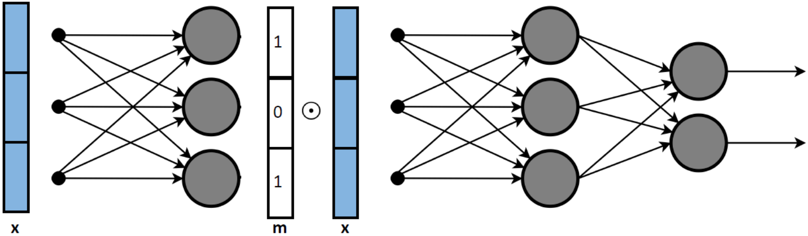


- What happens when we do not have explanation data?
- Need to use the task-specific supervision signal to create explanations
- Key principle: Given an instance, automatically learn to select features during inference from the task-specific supervised signal

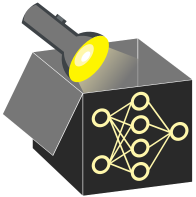
INSTANCE-WISE FEATURE SELECTION



- Key principle: Given an instance, automatically learn to select features during inference
- The selected features can be implemented as a binary mask over the original feature space
- Selector network selects the mask, predictor network predicts using the masked input

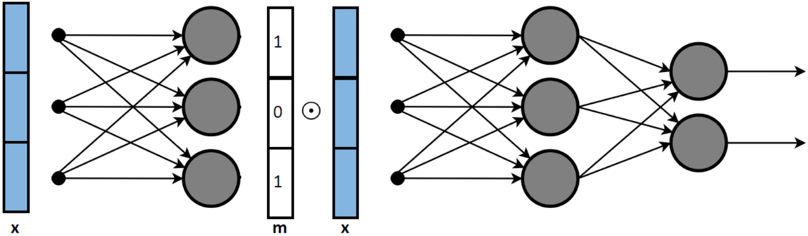


PROBLEMS IN OPTIMISATION

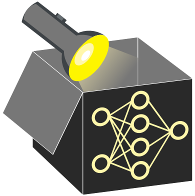


- Selector network selects the mask, predictor network predicts using the masked input
- Binary masking introduces discontinuity in the neural network
- Discontinuity $\rightarrow$ gradient-based optimisation is not possible

- How can we learn the parameters of such a network using gradient-based optimization?



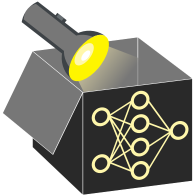
GENERATIVE MASKS



- Masks are generated from a probability distribution
- Instance-wise feature selection as finding the expectation of the predictor function distributed according to human interpretability

$$\mathcal{F}(\boldsymbol{\theta}) := \int p(\mathbf{m}; \mathbf{x}, \boldsymbol{\theta}) f(\mathbf{m} \odot \mathbf{x}; \phi) \mathrm{d}\mathbf{x} = \mathbb{E}_{p(\mathbf{m}; \mathbf{x}, \boldsymbol{\theta})}[f(\mathbf{m} \odot \mathbf{x}; \phi)]$$

INSTANCE-WISE FEATURE SELECTION

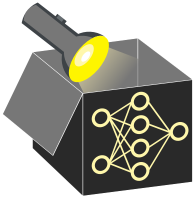


- The distribution over explanations is parameterized by a neural network
- The predictor network is also parameterized by a neural network

$$\mathcal{F}(\boldsymbol{\theta}) := \int p(\mathbf{m}; \boldsymbol{\theta}) f(\mathbf{m} \odot \mathbf{x}; \phi) \mathrm{d}\mathbf{x} = \mathbb{E}_{p(\mathbf{m}; \boldsymbol{\theta})}[f(\mathbf{m} \odot \mathbf{x}; \phi)]$$

- Predictor network accepts a masked input

MONTE CARLO SAMPLING



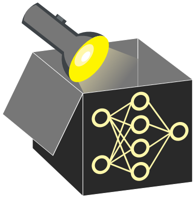
$$\mathcal{F}(\boldsymbol{\theta}) := \int p(\mathbf{m}; \mathbf{x}, \boldsymbol{\theta}) f(\mathbf{m} \odot \mathbf{x}; \phi) \mathrm{d}\mathbf{x} = \mathbb{E}_{p(\mathbf{m};\mathbf{x},\boldsymbol{\theta})}[f(\mathbf{m} \odot \mathbf{x}; \phi)]$$

- **Trick:** $\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) = \frac{\nabla_{\boldsymbol{\theta}} p(\mathbf{x};\boldsymbol{\theta})}{p(\mathbf{x};\boldsymbol{\theta})}$

⇝ In a simplified notation (ignoring **m**), we get the following:

$$\begin{aligned}\boldsymbol{\eta} := \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})}[f(\mathbf{x}; \phi)] \\ &= \nabla_{\boldsymbol{\theta}} \int p(\mathbf{x}; \boldsymbol{\theta}) f(\mathbf{x}) d\mathbf{x} = \int f(\mathbf{x}) \nabla_{\boldsymbol{\theta}} p(\mathbf{x}; \boldsymbol{\theta}) d\mathbf{x} \\ &= \int p(\mathbf{x}; \boldsymbol{\theta}) f(\mathbf{x}) \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) d\mathbf{x} \\ &= \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})} \left[f(\mathbf{x}) \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) \right]\end{aligned}$$

MONTE CARLO ESTIMATOR

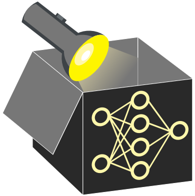


$$\mathcal{F}(\boldsymbol{\theta}) := \int p(\mathbf{m}; \mathbf{x}, \boldsymbol{\theta}) f(\mathbf{m} \odot \mathbf{x}; \phi) \mathrm{d}\mathbf{x} = \mathbb{E}_{p(\mathbf{m};\mathbf{x},\boldsymbol{\theta})}[f(\mathbf{m} \odot \mathbf{x}; \phi)]$$

$$\boldsymbol{\eta} := \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})}[f(\mathbf{x}; \phi)] = \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})}\left[f(\mathbf{x}) \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta})\right]$$
$$= \frac{1}{N} \sum_{n=1}^{N} f\left(\hat{\mathbf{x}}^{(n)}\right) \nabla_{\boldsymbol{\theta}} \log p\left(\hat{\mathbf{x}}^{(n)}; \boldsymbol{\theta}\right); \quad \hat{\mathbf{x}}^{(n)} \sim p(\mathbf{x}; \boldsymbol{\theta})$$

- Sample N masks from the probability distribution p
- Compute the weighted avg. of the samples where:
  - weight = derivative of the log prob. of the sample mask
- update the parameters of the selector network using this weighted average

REDUCING VARIANCE

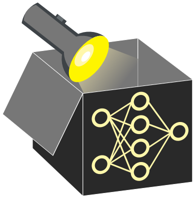


$$\mathcal{F}(\boldsymbol{\theta}) := \int p(\mathbf{m}; \mathbf{x}, \boldsymbol{\theta}) f(\mathbf{m} \odot \mathbf{x}; \phi) \mathrm{d}\mathbf{x} = \mathbb{E}_{p(\mathbf{m}; \mathbf{x}, \boldsymbol{\theta})}[f(\mathbf{m} \odot \mathbf{x}; \phi)]$$

- Monte Carlo estimators suffer from the problem of high variance
- Solution: introduce a constant baseline value β

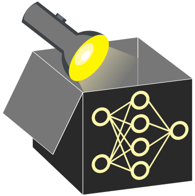
$$\eta = \mathbb{E}_{p(x;\theta)}[(f(x) - \beta)\nabla_\theta \log p(x; \theta)]$$

CONCLUSION



- Prefer simple models for better interpretability
- Regularisation for enforcing sparsity in the parameter space
- Feature selection for enforcing sparsity in the feature space
- Instance-wise feature selection selects different features based on different instances
- Selector and predictor architecture for instance-wise feature selection
- Optimisation using without explanation data requires tricks like Monte-Carlo sampling with gradients

REFERENCES



- “Learning to Explain: An Information-Theoretic Perspective on Model Interpretation” — J. Chen, Song, M.J. Wainwright, M. I. Jordan. ICML 2018.
  - http://proceedings.mlr.press/v80/chen18j/chen18j.pdf

- “Explain and Predict, and then Predict again" — Z Zhang, K Rudra, A Anand. WSDM 2020.
  - https://arxiv.org/pdf/2101.04109.pdf

- “INVASE: Instance-wise Variable Selection using Neural Networks” J. Yoon, J. Jordon, M. Schaar. ICLR 2019.
  - https://openreview.net/pdf?id=BJg_roAcK7