EDiT: Interpreting Ensemble Models via Compact Soft Decision Trees

Jaemin Yoo
Seoul National University
Seoul, South Korea

Lee Sael
Ajou University
Suwon, South Korea

ICDM 2019
Outline

- Introduction
- Proposed Method
- Experiments
- Conclusion
Black Box Models

- Most ML models are *black boxes*
  - Learned structures are random and complex
  - Their decisions are not explainable

Image from https://www.investopedia.com/terms/b/blackbox.asp
Interpretable ML

- Research to interpret a model’s decisions
  - Important when each decision is irreversible
- Two types of interpretable models:
  - *Linear models*
  - *Decision trees*
- However, their accuracy is not good
Ensemble Models

- **Ensemble models**
  - Combine the predictions of weak models
  - Produce robust and accurate predictions

- However, they have **low interpretability**
  - Decisions are made by hundreds of learners

Image from https://dsc-spidal.github.io/harp/docs/examples/RF/
Problem Definition

- **Given** a trained ensemble model $M$
- **Train** an *interpretable* classifier $S$
- **Such that**
  - $S$ achieves similar accuracy to $M$
  - $S$ contains fewer parameters than $M$
Outline

- Introduction
- Proposed Method
- Experiments
- Conclusion
Proposed Method

- **Ensemble to Distilled Tree (EDiT)**
  - Given an ensemble model
  - Trains a *compact* soft decision tree
    - Interpretable & more efficient than SDTs

- **EDiT** is based on three main ideas
  - **Idea 1:** Knowledge distillation
  - **Idea 2:** Weight sparsification
  - **Idea 3:** Tree pruning
Preliminary: SDTs

- SDTs are interpretable tree-based models
  - Each internal node is a linear classifier
  - Each leaf node learns a probability distribution

![Diagram of Soft Decision Trees](image)

Image from “Rule-Extraction from Soft Decision Trees” (L. Huang, M. Hsieh, and M. Rajati, BDAI 2019)
Idea 1: Distillation

- **Knowledge distillation**
  - Transfers the knowledge of a teacher to a student
- Replace the labels $y$ in training data $\mathcal{D}$ as

$$ y_i \leftarrow \frac{M(x_i) + y_i}{2} \text{ for each } (x_i, y_i) \text{ in } \mathcal{D} $$

- $x_i$ is a feature vector that corresponds to $y_i$
Idea 2: Sparse Weights (1)

- **Weight sparsification**
  - Improves the efficiency by sparse weights

- Propose three different approaches
  - 1) **L1 regularization**
    - Adds an L1 regularizer to the loss function
  - 2) **Weight masking**
    - Inactivates randomly some of the weights
  - 3) **Weight pruning**
    - Prunes weights whose learned values are small
Idea 2: Sparse Weights (2)

- **Weight masking**: 2 steps
  - Random masking
  - Training

- **Weight pruning**: 3 steps
  - Pre-training
  - Pruning
  - Fine-tuning
Idea 3: Tree Pruning

- **Tree pruning**
  - Removes nodes of small arrival probabilities
  - Enables a large depth to be adopted

- Tree pruning vs. weight pruning
  - Weight pruning removes redundant weights
  - Tree pruning removes redundant tree nodes
Summary

- Result of applying our ideas to an SDT

  - Sparse weights from sparsification
  - Narrow tree structure from tree pruning
Outline

- Introduction
- Proposed Method
- Experiments
- Conclusion
Does EDiT outperform the baselines?

EDiT shows the best balance in all cases

- High accuracy with only a few parameters
Sparsification Methods

- Which is the best sparsification method?
  - **Weight pruning** works generally the best
  - L1 regularization fails even with large $\lambda$

![Graphs showing comparison of different sparsification methods]

Jaemin Yoo (SNU)
Outline

- Introduction
- Proposed Method
- Experiments
- Conclusion
Conclusion

- **Ensemble to Distilled Tree (EDiT)**
  - Our approach to interpret ensemble models
    - **Idea 1:** Knowledge distillation
    - **Idea 2:** Weight sparsification
    - **Idea 3:** Tree pruning
  - **EDiT** gives the most efficient predictions
    - **Accuracy:** $\text{DT} \ll \text{RF} \approx \text{SDT} \approx \text{EDiT}$
    - **Parameters:** $\text{DT} \approx \text{EDiT} \ll \text{SDT} < \text{RF}$
Thank you!

GitHub: https://github.com/leesael/EDiT