Summary

- Reduce model complexity of Bayesian network classifiers
- Transfer techniques from deep learning [1] to Bayesian networks
- Method 1: Model-size-aware TAN structure learning
  - Apply differentiable TAN structure learning from [2]
  - Structure learning using backpropagation
  - New extension: Trade off between accuracy and model size
- Method 2: Quantization-aware Training
  - Quantize log-probabilities (CPTs) to few bits
  - Apply straight-through gradient estimator
- Comparing Bayesian networks and deep neural networks

Bayesian Network (BN) Classifiers

- BNs define a factorization of a joint distribution via a directed acyclic graph $\mathcal{G}$ as
  \[ p(X) = \prod_{i=1}^{n} p(X_i | pa(X_i)) \]
- This work: Discrete features $X_i$
  - Parameters: Conditional probability tables (CPTs) $\theta$
  - Classification: Additional distinct class variable $C$
  - Classify according to $\arg\max \log p(x, c)$
  - Classification requires only $(D+1) \cdot \#(\text{classes})$ additions in log-domain

Naive Bayes vs. tree-augmented naive Bayes (TAN)

- Naive Bayes
  - Few parameters
  - Few operations
  - Accuracy
  - Many possible TAN structures $\Rightarrow$ structure learning
- TAN
  - Few parameters
  - Few operations
  - Accuracy
  - Few operations
  - Model size depends on structure

Model-Size-Aware TAN Structure Learning

- Encode TAN graph $\mathcal{G}$ using one-hot vectors $s = (s_1, \ldots, s_D)$
- CPTs for all possible parents: $\Theta = \{ \theta_1, \ldots, \theta_D \}$
- Treat parameters $\Theta$ and structure $s$ jointly
  \[ \log p(x, C) = \log p_0(C) + \sum_{i=1}^{D} \sum_{j=0}^{1} s_{ij} \log p_{ij}(X_i | X_j, C) \]
- Continuous relaxation of one-hot $s_i$ results in probabilities $\Phi_i$
- Probabilities $\Phi = (\Phi_1, \ldots, \Phi_D)$ define a distribution over TAN graphs $\mathcal{G}$
- Optimize expectation with respect to $\Phi$ and take most probable graph $\mathcal{G}^* = \arg\max \mathbb{E}[L_{\text{SL}}(\Phi, \theta)]$ (proposed in [2])
- $L_{\text{SL}}$ is differentiable with respect to $\Phi \Rightarrow$ optimize with backpropagation
- This work: New term penalizes number of parameters
  \[ L_{\text{SL}}^{\text{MS}}(\Phi, \theta) = L_{\text{SL}}(\Phi, \theta) + \frac{\text{model parameters}}{\text{operations}} \]

Quantization-Aware Training

- BNs: Quantize log-probabilities to negative fixed-point values
  \[ Q_{\text{BN}}(\theta) = \text{clip}(\text{round}(\theta \cdot 2^{B_f}) \cdot 2^{-B_r}, -U, 0) \]
- DNNs: Quantize weights according to [3]
  \[ Q_{\text{DNN}}(w) = Q \left( \frac{\text{clip}(w, -1, 1) + 1}{2} \cdot B \right) \cdot 2 - 1 \]
  \[ Q(v; B) = \frac{1}{2^B - 1} \cdot \text{round}(2^B - 1) \cdot v \]
- The gradient of quantization functions is zero almost everywhere
  - Apply the straight-through gradient estimator [4] during training

At backpropagation: Pretend that the gradient of a quantization function is non-zero

Experiments: Model-Size-Aware TAN Structure Learning

- Effective trade-off between accuracy and model size
- Pareto frontier: Model size vs. accuracy

Experiments: Quantization-Aware Training

- Fixed parameter memory budget
- Naive Bayes vs TAN

Comparing quantized DNNs and quantized BN classifiers

- BNs: $\checkmark$ few operations $\checkmark$ decent accuracy
- CNNs: $\checkmark$ best accuracy $\checkmark$ few parameters $\checkmark$ many operations
- fully connected DNNs: $\checkmark$ flexible trade-offs

References

[2] Roth and Pernkopf, Differentiable TAN Structure Learning for Bayesian Network Classifiers, PGM 2020
[5] Tuchalski et al., Integer Bayesian Network Classifiers, ECML 2014

Acknowledgments: This work was supported by the Austrian Science Fund (FWF) under the project number I2700-R33.