Relevance Effect: Exploiting Bayesian Networks to Improve Supervised Learning

Ardavan S. Nobandegani†
ardavan.salehinobandegani@mail.mcgill.ca

Jad Kabbara†
jad@cs.mcgill.ca

Ioannis N. Psaromiligkos
yannis@ece.mcgill.ca

† Co-primary authors

Introduction

We introduce the notion of relevance effect which bears on exploiting BNs to generate realizations of relevant variables to be used for potentially improving the performance of a learning model on a supervised task.

- We explore the use of the relevance effect in Deep Belief Networks (DBNs) with a focus on relational domains.
- Despite being at odds with the non-monotonicity of probability, we attain improvements in learning performance on tasks involving both synthetic and real-world data.
  - ➢ This suggests that relevance effect has the potential to be practiced for improving the performance of supervised learning methods in general.

Relevance Effect

- Consider a classification task involving input variables $\boldsymbol{e}_1, \dots, \boldsymbol{e}_n$ and output variable $\boldsymbol{o}$.
- Variables are part of a domain modeled by some (partially known) BN $\mathcal{B}$.
  - ➢ How to use $\mathcal{B}$ to improve classification performance on output variable $\boldsymbol{o}$?

Rule:

- Mapping that acts on variables $\boldsymbol{e}_1, \dots, \boldsymbol{e}_n$.
- Probabilistic or deterministic.

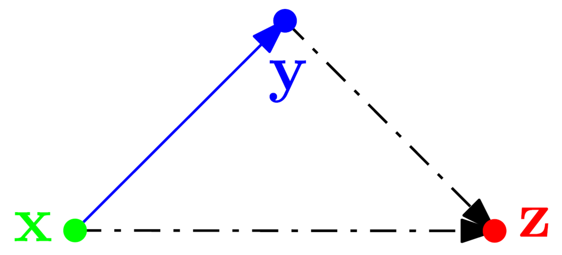
Sampling in the "wrong" way:

- In general, to correctly generate samples of an arbitrary variable $\boldsymbol{s}$ belonging to BN $\mathcal{B}$, one must draw samples from CPD $P(\boldsymbol{s}|\boldsymbol{e}_1, \dots, \boldsymbol{e}_n, \boldsymbol{o})$.
- The wrong way: To purposefully ignore some of the observed variables.
  - ➢ E.g., sample from $P(\boldsymbol{s}|\boldsymbol{e}_1, \dots, \boldsymbol{e}_n)$ or $P(\boldsymbol{s}|\boldsymbol{e}_1)$.

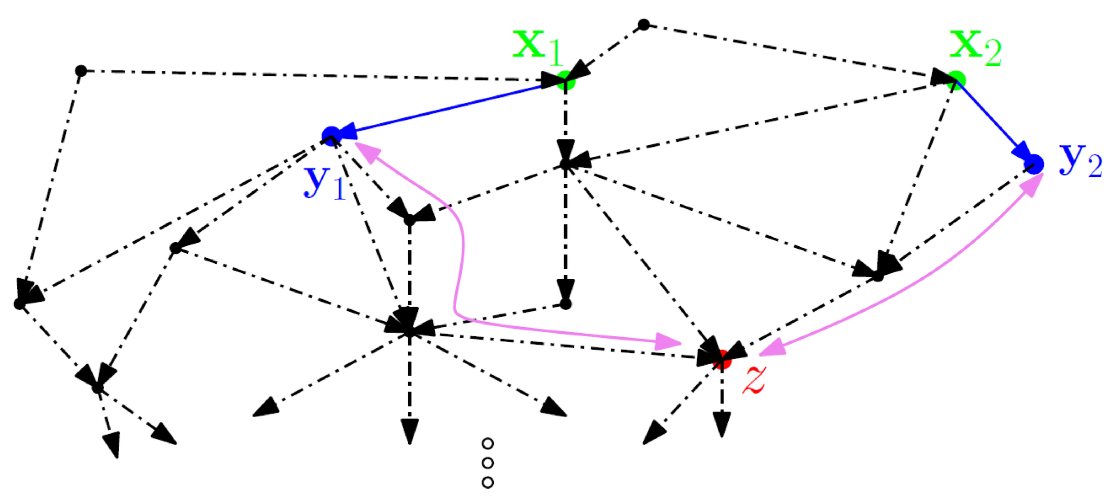
Relevance Effect:

Finding the value of some relevant variable(s), even in the wrong way, and incorporating it into the training process could improve the learning performance.

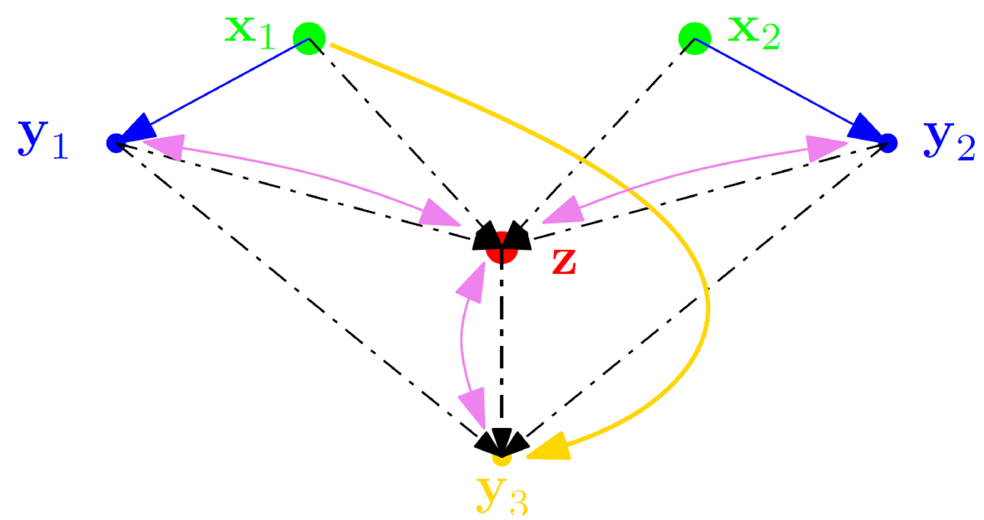
Motivating Examples



- Classification task:
  - Input: $\boldsymbol{x}$, Output: $\boldsymbol{z}$
  - $P(\boldsymbol{y}|\boldsymbol{x})$ known, $P(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})$ unknown
  - $\boldsymbol{x}$ does not d-separate $\boldsymbol{y}$ from $\boldsymbol{z}$.
  - $\boldsymbol{z}$ is observed (training set)
- ➢ Relevance effect → use $P(\boldsymbol{y}|\boldsymbol{x})$ as a rule.

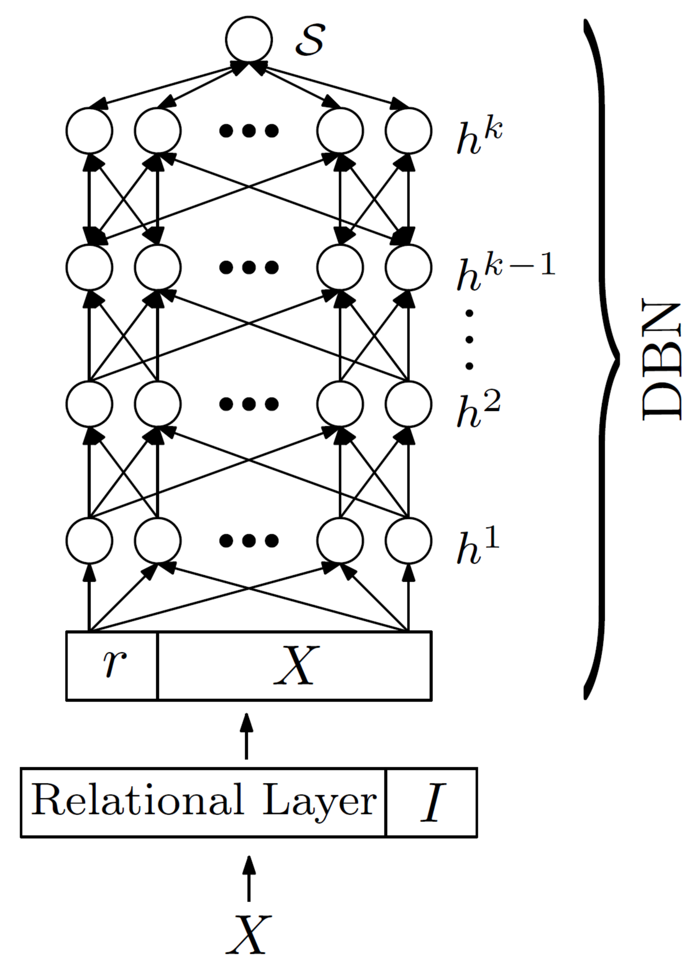


Sparsely-known complex network



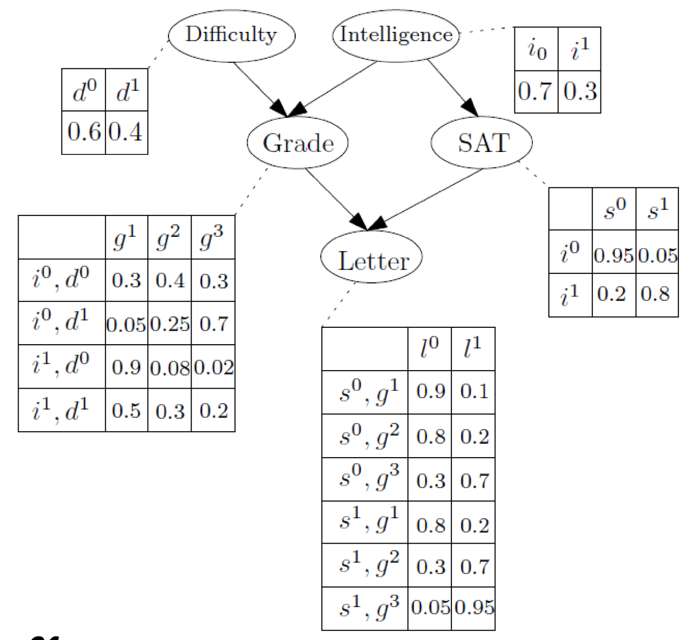
Generalized interpretation of rules

Deep Learning Framework



Experiments

Experiment 1.



- Input: $\boldsymbol{i}, \boldsymbol{d}$
- Output: $\boldsymbol{l}$
- Rule: $P(\boldsymbol{s}|\boldsymbol{i})$
- ✓ 7.53% improvement

Experiment 2.

- BN defined over $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{10}$.

$$\boldsymbol{x}_1 \sim \mathcal{N}(0,1),$$

$$\boldsymbol{x}_2 \sim \mathcal{U}(0,1) \text{ w.p. } p \ \& \ \boldsymbol{x}_2 \sim \mathcal{U}(-1,0) \text{ o/w}$$

$$(\boldsymbol{x}_{i+1}|\boldsymbol{x}_i, \boldsymbol{x}_{i-1}) \sim \begin{cases} \mathcal{U}(0,1) \ if \ \psi(\boldsymbol{x}_{i+1}) > \gamma \\ \mathcal{U}(-1,0) \ if \psi(\boldsymbol{x}_{i+1}) < \gamma \end{cases}$$

$$\psi(\boldsymbol{x}_{i+1}) \coloneqq \frac{1}{2}(\boldsymbol{x}_i + \boldsymbol{x}_{i-1})$$

- Input: $\boldsymbol{x}_4, \boldsymbol{x}_5$
- Output: $\boldsymbol{x}_{10} \gtrless \boldsymbol{0}$
- Rule: $P(\boldsymbol{x}_6|\boldsymbol{x}_4, \boldsymbol{x}_5)$
- ✓ 22% improvement

Experiment 3.

- KEGG Metabolic Relation Network Dataset
- Output: predicting whether enzymes/genes are interacting with more than 3 other neighbors
- Two rules: clustering coefficient and betweenness centrality of enzymes/genes
- Rules obtained using LS-regression (trained on about half of the features , 10% of training set)
- ✓ 10.47% improvement

	RDNN	DBN
Student BN	$\frac{737 \pm 3.51}{2000}$ **(36.85%)**	$\frac{797}{2000}$ (39.85%)
10-Node BN	$\frac{41.33 \pm 1.80}{2000}$ **(2.07%)**	$\frac{53}{2000}$ (2.65%)
Metabolic Network	$\frac{231}{8000}$ **(2.89%)**	$\frac{258}{8000}$ (3.23%)

Performance of RDNN and DBN in terms of number of misclassifications.
The notation is as follows: $\frac{\text{number of misclassifications}}{\text{size of test size}}$ (Percentage).

Remarks

- Using BNs to improve learning performance of Deep Neural Nets.
- A step towards incorporating relational knowledge into Deep Learning paradigm.
- Applicable to all supervised learning models.