Inclusion of Symbolic Domain-Knowledge into Deep Neural Networks

(PhD Thesis Presentation)

Tirtharaj Dash

Dept. of CS & IS and APPCAIR
BITS Pilani, Goa Campus

July 2022

Supervisors

Supervisor	:	Ashwin Srinivasan Senior Professor Department of CS & IS and APPCAIR BITS Pilani, Goa Campus
Co-supervisor	:	Sukanta Mondal Associate Professor Department of Biological Sciences BITS Pilani, Goa Campus

Motivation

- The incorporation of domain-knowledge is the first of the 3 Grand Challenges in developing AI systems:

  "ML and AI are generally domain-agnostic.... Off-the-shelf practice treats [each of these] datasets in the same manner and ignores domain knowledge...

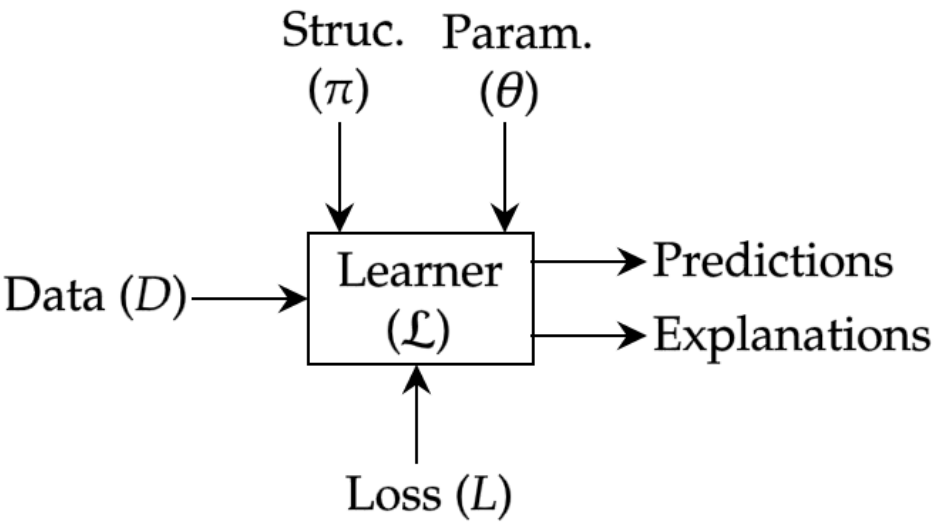
  Improving our ability to systematically incorporate diverse forms of domain knowledge can impact every aspect of AI ..."

(AI for Science Report, 2020)

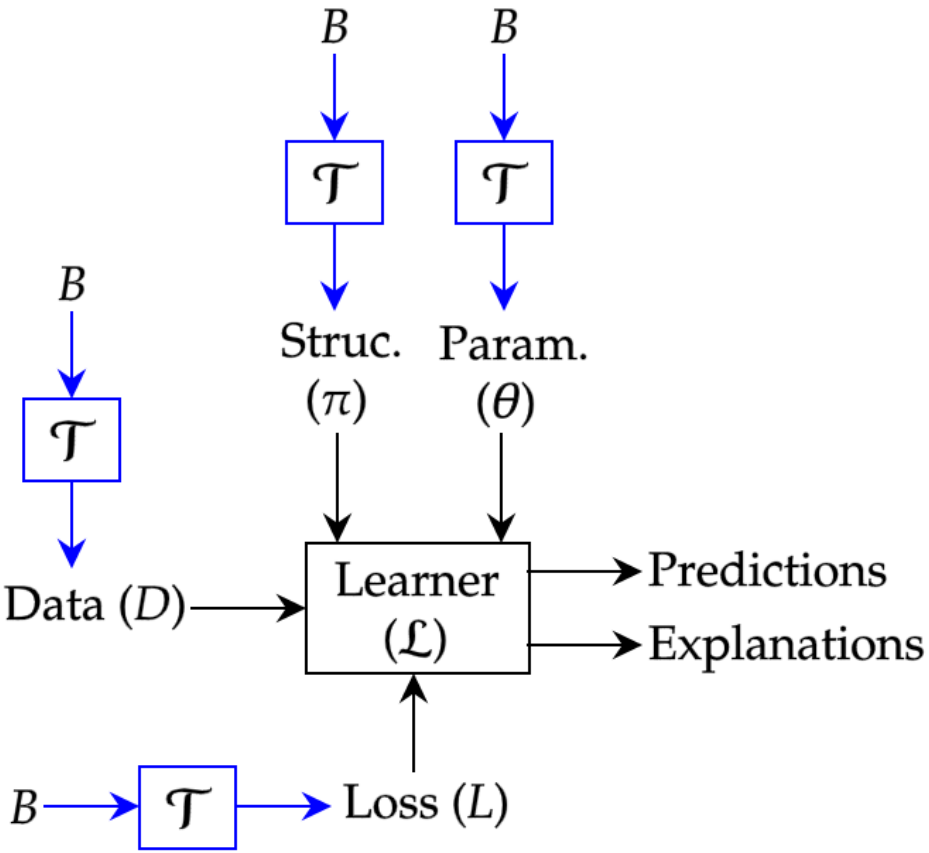
Stevens *et al* (2020): AI for Science, *Argonne National Lab, USA.*

Machine Learning (ML)

A machine that "learns":



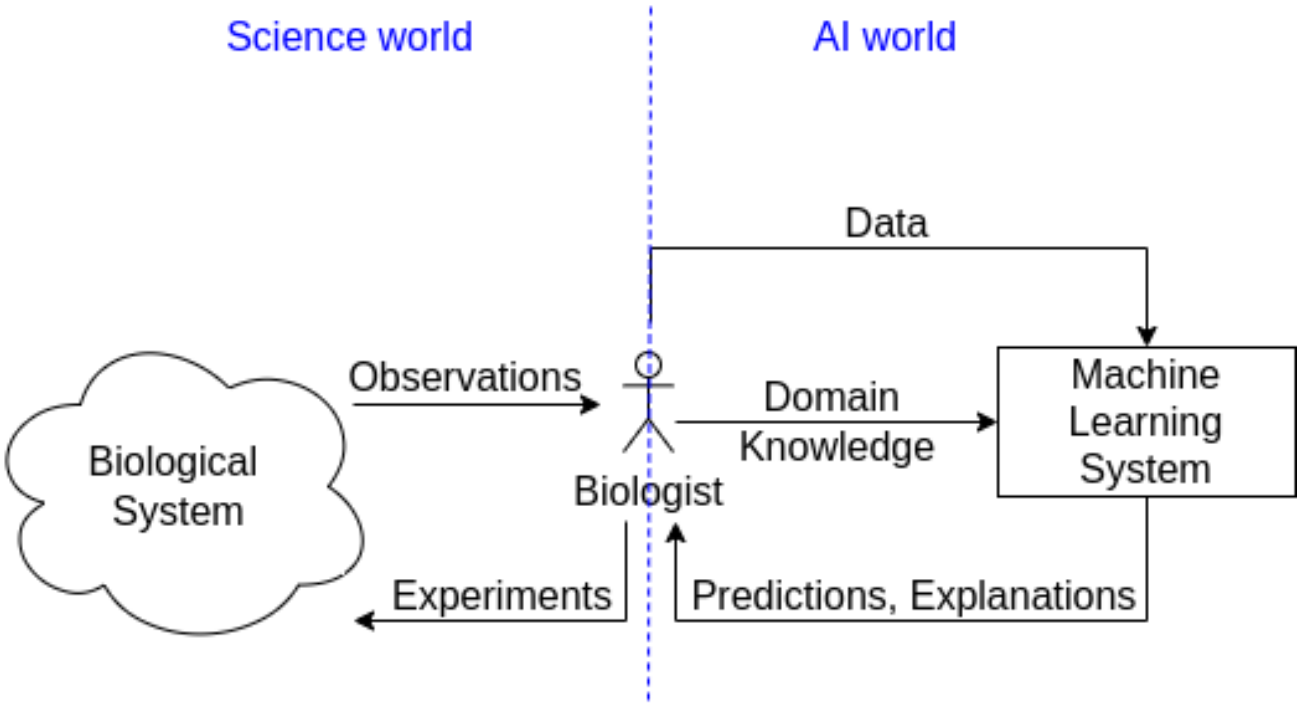
ML with Domain-Knowledge



B: Background knowledge, $\mathcal{T}$: Transducer

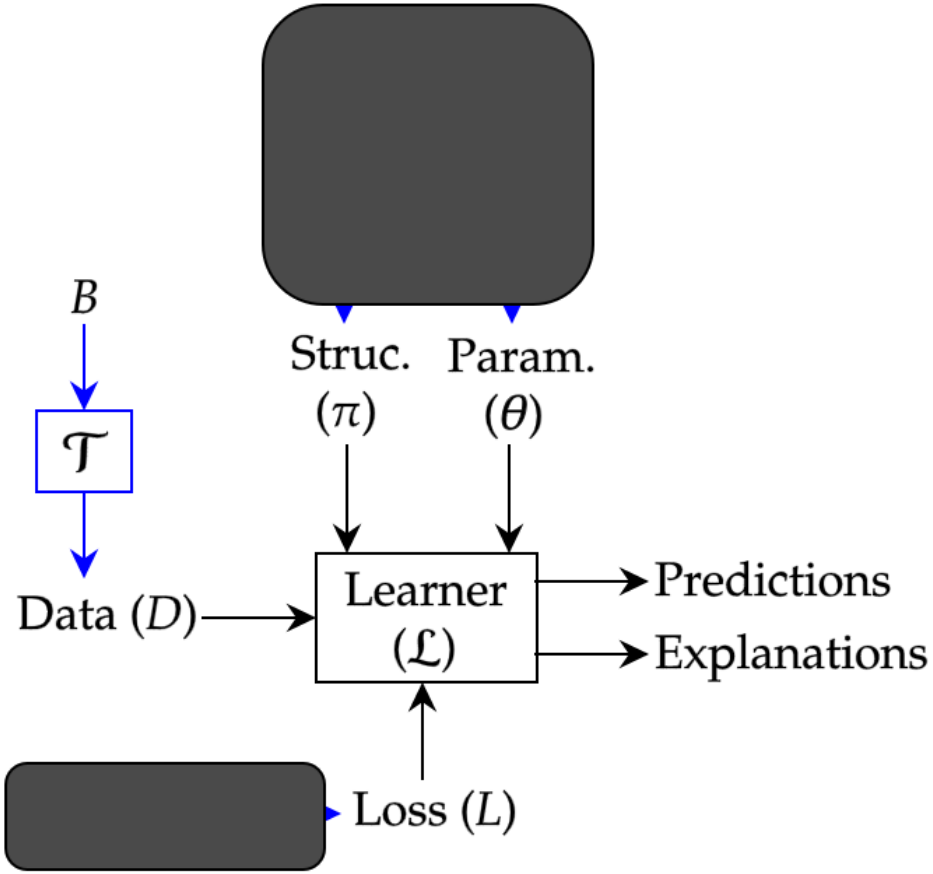
ML with Domain-Knowledge

But from where does this B come?



(B comes from domain-experts; here, a biologist.)

This Thesis


B
$\mathcal{T}$
Data (D)
Struc. (π)
Param. (θ)
Learner ($\mathcal{L}$)
Predictions
Explanations
Loss (L)

Primary Hypothesis

Inclusion of domain-knowledge* can significantly improve the performance of a deep neural network.

*by transforming the data representation

Results:

- This thesis provides conceptual contributions for inclusion of domain-knowledge into some kinds of deep neural networks.
- The experiments conducted in this thesis provide empirical evidence to support the primary hypothesis.

Contributions

Conceptual:

1. Stochastic selection of relational features as inputs for MLPs
2. Simplified inclusion of relational information into GNNs
3. Complete inclusion of relational information into GNNs

Implementational: Resulting in neural-symbolic techniques

1. Deep Relational Machines (DRMs)
2. Vertex-Enriched Graph Neural Networks (VEGNNs)
3. Bottom-Graph Neural Networks (BotGNNs)
4. A modular system for sequence generation that uses a BotGNN as a component

MLP: Multilayer Perceptron

GNN: Graph Neural Network

Contributions

Applications:

1. Investigation of our NeSy techniques on:
   a. Large-scale carcinogenicity problems: Throughout this thesis, we compare DNNs with and without domain-knowledge that provides support for our primary hypothesis.
   b. Lead-discovery problem relevant to drug design: We show a human-in-the-loop application in drug design.

Contributions

Publications from this Thesis:

1. T. Dash, S. Chitlangia, A. Ahuja, A. Srinivasan, "A review of some techniques for inclusion of domain-knowledge into deep neural networks", *Nature Scientific Reports*, 2022. [URL]
2. T. Dash, A. Srinivasan, L. Vig, A. Roy, "Using domain-knowledge to assist lead discovery in early-stage drug design", *International Conference on Inductive Logic Programming*, 2021. [URL]
3. T. Dash, A. Srinivasan, A. Baskar, "Inclusion of domain-knowledge into GNNs using mode-directed inverse entailment", *Machine Learning*, 2021. [URL]
4. T. Dash, A. Srinivasan, L. Vig, "Incorporating symbolic domain knowledge into graph neural networks", *Machine Learning*, 2021. [URL]
5. T. Dash, A. Srinivasan, R.S. Joshi, A. Baskar, "Discrete stochastic search and its application to feature-selection for deep relational machines", *International Conference on Artificial Neural Networks*, 2019. [URL]
6. T. Dash, A. Srinivasan, L. Vig, O.I. Orhobor, R.D. King, "Large-scale assessment of deep relational machines", *International Conference on Inductive Logic Programming*, 2018. [URL] (*Winner of the Best Student Paper Award)

Contributions

Some other publications during this PhD:

1. G. Chhablani et al., "Superpixel-based Knowledge Infusion in Deep Neural Networks for Image Classification", *ACMSE*, 2022. [URL] (*Best Short Paper)
2. A. Sonwane et al., "Solving Visual Analogies Using Neural Algorithmic Reasoning", *AAAI Student Abstract and Poster Program*, 2022. [URL]
3. I. Olier et al., "Transformational machine learning: Learning how to learn from many related scientific problems", *PNAS*, 2021. [URL]
4. S. Chitlangia et al., "Using Program Synthesis and Inductive Logic Programming to solve Bongard Problems", *AAIP@IJCLR*, 2021. [URL]
5. H. Shah et al., "Empirical Study of Data-Free Iterative Knowledge Distillation", *ICANN*, 2021. [URL]
6. S. Krishnan et al., "A Case Study of Transfer of Lesion-Knowledge", *MIL3ID@MICCAI*, 2020. [URL]
7. K. Mahajan et al., "CovidDiagnosis: Deep Diagnosis of Covid-19 Patients using Chest X-rays", *TIA@MICCAI*, 2020. [URL]

Contributions

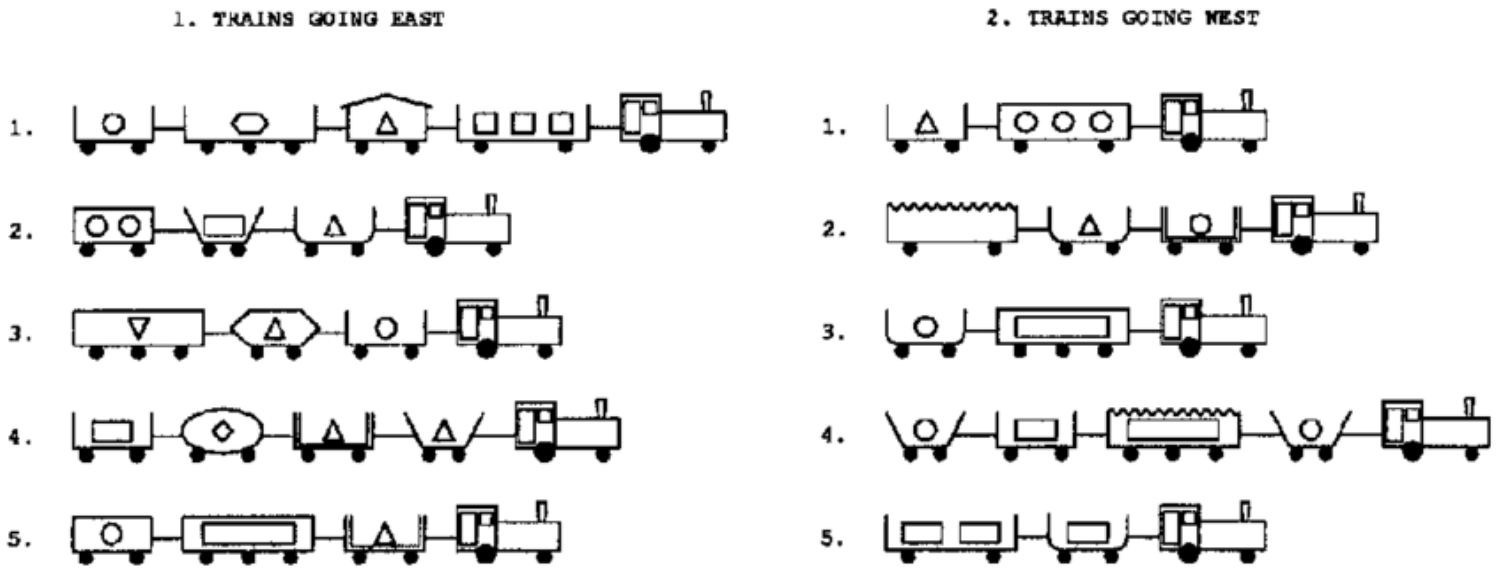
8. S. Yalburgi et al., "An Empirical Study of Iterative Knowledge Distillation for Neural Network Compression", *ESANN*, 2020. [URL]
9. T. Dash et al., "Adversarial neural networks for playing hide-and-search board game Scotland Yard", *Neural. Comput. Appl.*, 2018. [URL]
10. A. Saboo et al., "GASOM: Genetic Algorithm Assisted Architecture Learning in Self Organizing Maps", *ICONIP*, 2017. [URL]
11. P.P. Pai et al., "Sequence-based discrimination of protein-RNA interacting residues using a probabilistic approach", *J. Theor. Biol.*, 2017. [URL]

Inclusion of Domain-Knowledge using Propositionalisation

Implementation: Deep Relational Machines (DRMs)

DRMs

Let's start with Michalski's trains problem:



Learning task: Construct a classifier that distinguishes between eastbound and westbound trains.

Michalski (1980): Pattern recognition as rule-guided inductive inference, *IEEE PAMI*.

DRMs

The following properties are known about the trains:

- *has_car*/2: Which cars are appended to a train
- *short*/1: Whether the cars are short
- *long*/1: Whether the cars are long
- *closed*/1: whether the cars are closed
- *open*/1: whether the cars are open
- *jagged*/1: whether the cars are jagged
- ...

This is domain-knowledge for the trains problem.

DRMs

How do we incorporate domain-knowledge into MLPs?

- Using relational features: Inputs for MLP

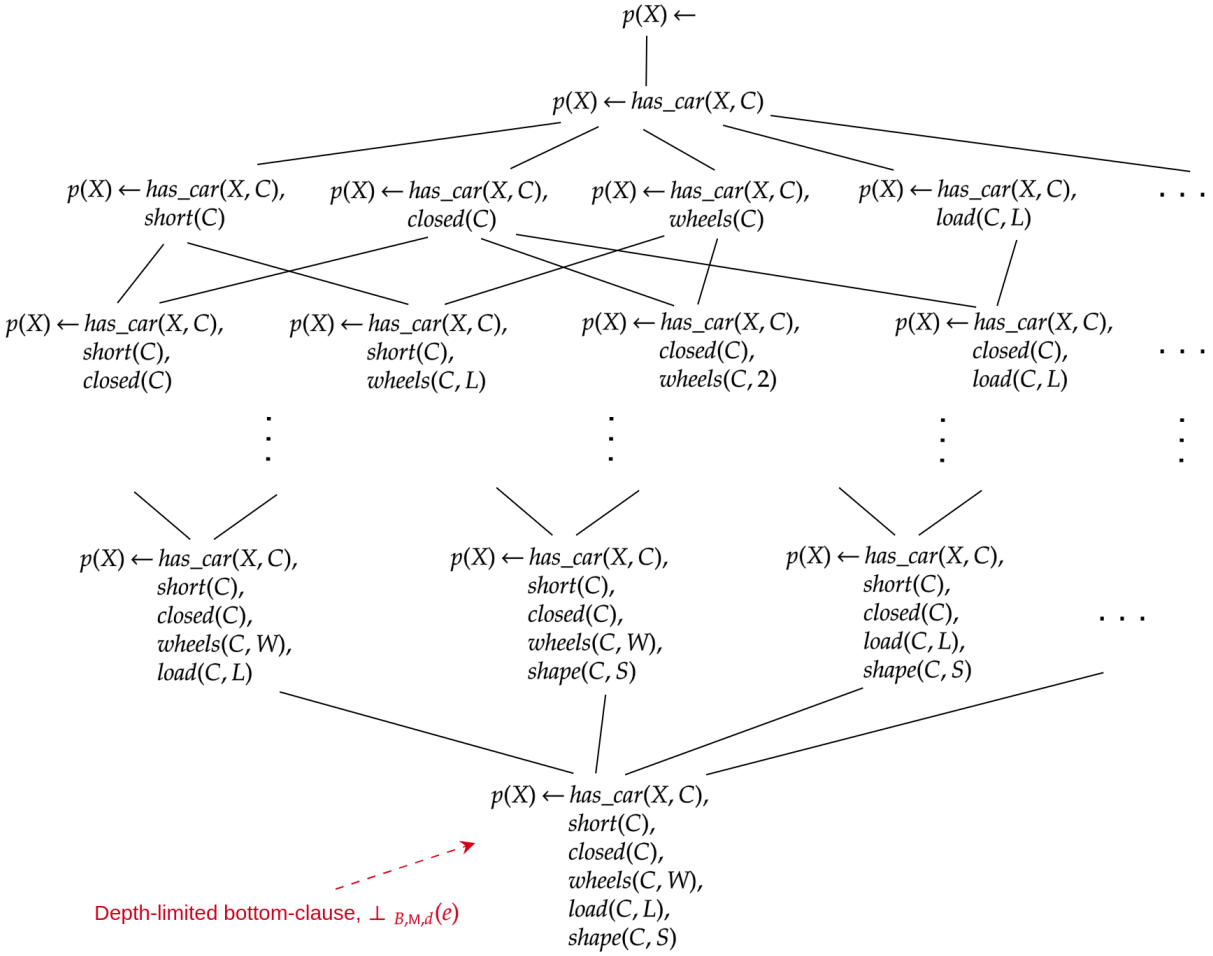
DRMs

Relational features for trains:

$C_1 : (p(X) \leftarrow has\_car(X, Y))$
$C_2 : (p(X) \leftarrow has\_car(X, Y), short(Y))$
$C_3 : (p(X) \leftarrow has\_car(X, Y), closed(Y))$
$C_4 : (p(X) \leftarrow has\_car(X, Y), short(Y), closed(Y))$
$C_5 : (p(X) \leftarrow has\_car(X, Y_1), short(Y_1), has\_car(X, Y_2), closed(Y_2))$

DRMs

But, from where do we get these relational features?



(Subsumption lattice of relational features)

DRMs

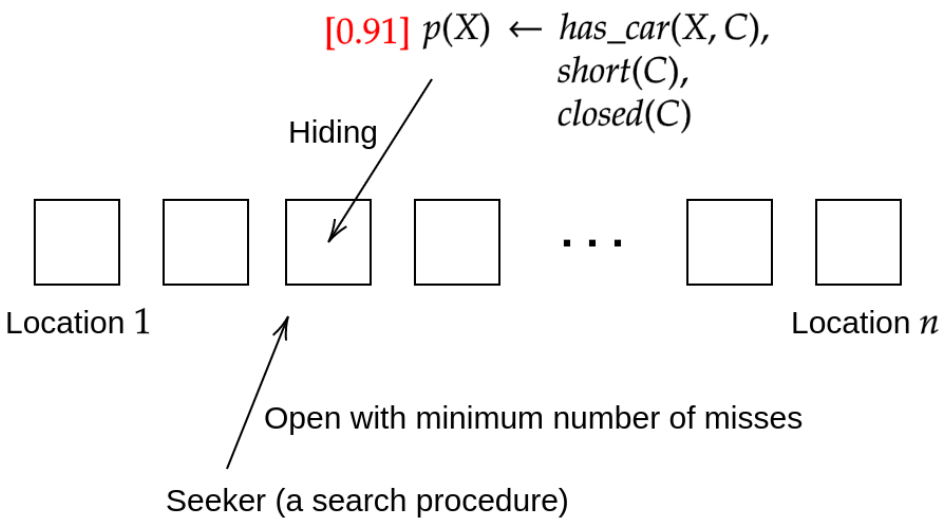
The feature space is very large.

- How to select a (good) set of features?
- How to select a single good feature?

Proposal: Mapping to a hide-and-seek game setting.

DRMs

Hide-and-seek game: A distributional setting



Formalisation:

- Hider distribution known (uniform and non-uniform)
- Hider distribution unknown
  - Non-uniform (Real-world is not adversarial.): Hide-and-Seek Sampling

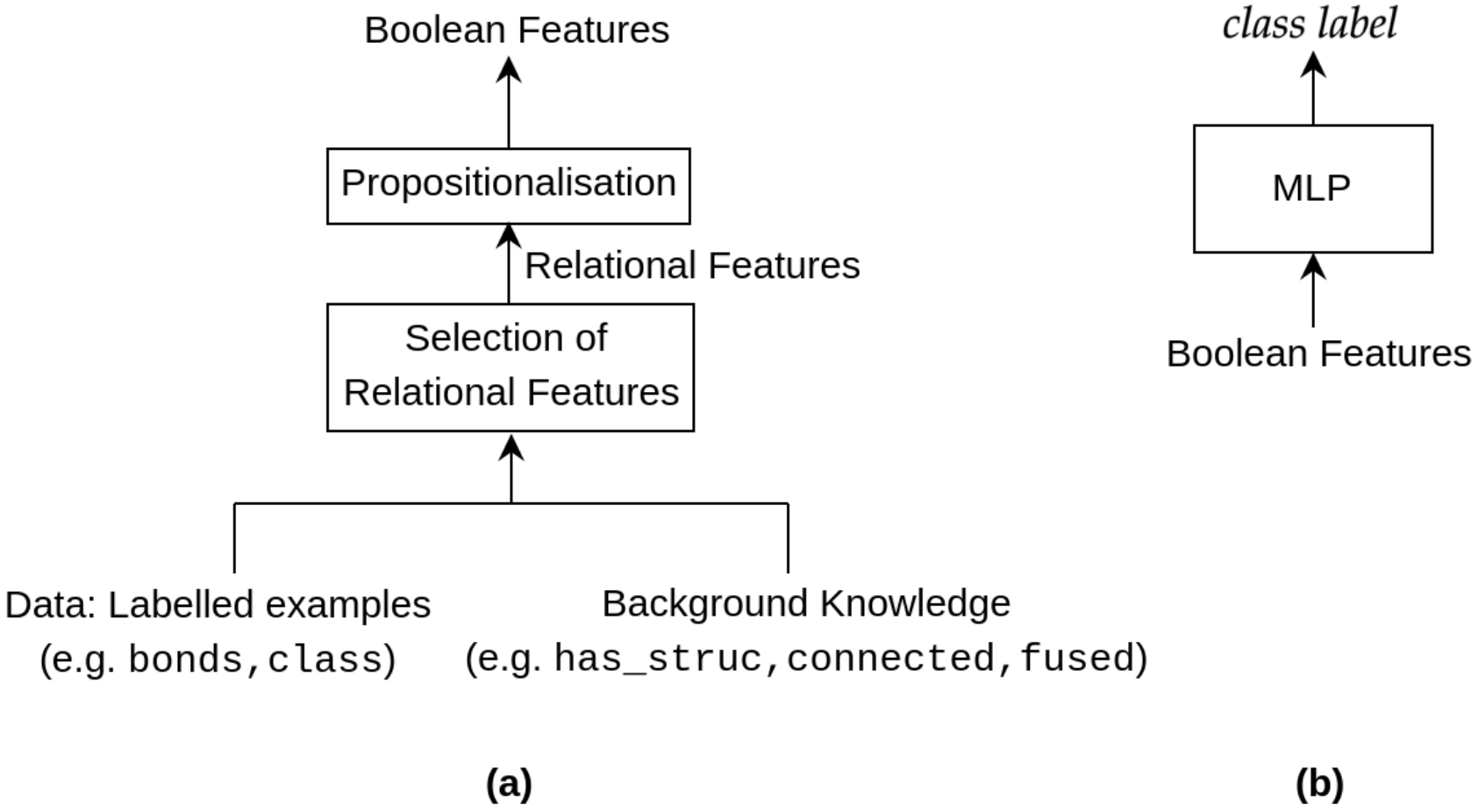
DRMs

Propositionalisation: Maps a relational feature to a Boolean value.

Example	$f(C_1, x)$	$f(C_2, x)$	$f(C_3, x)$	$f(C_4, x)$	$f(C_5, x)$	class
x_1	1	1	1	1	0	eastbound
x_2	1	1	1	1	1	eastbound
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_N	1	1	1	0	0	westbound

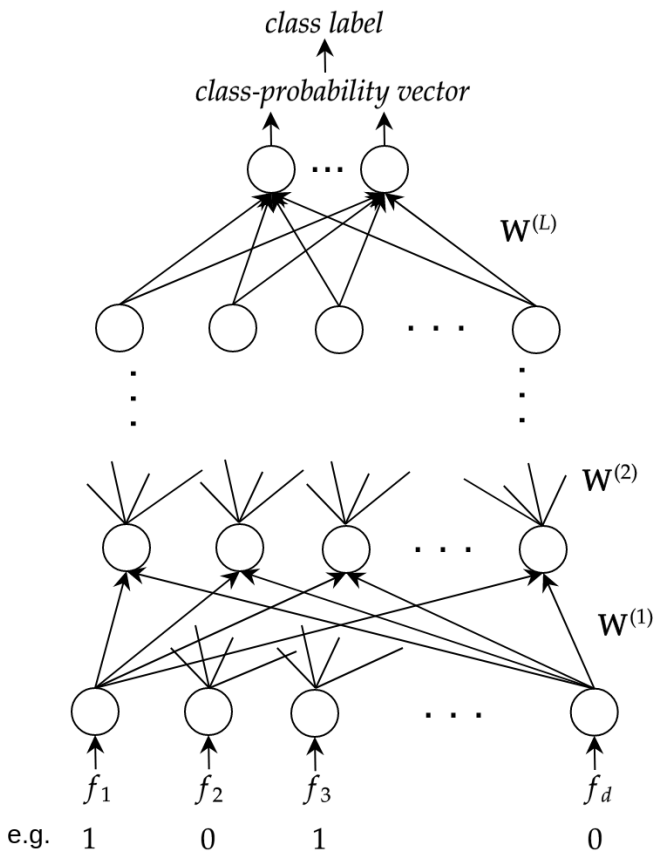
DRMs

Construction of a DRM using Boolean features:



DRMs

A DRM network:



Each f_i is a relational feature.

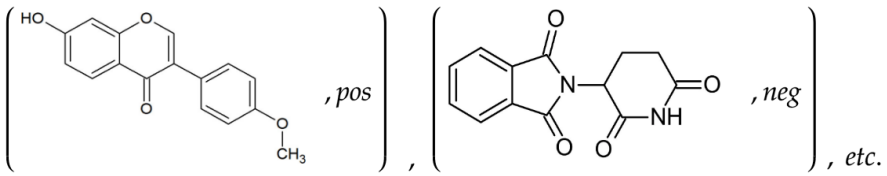
Empirical Evaluation of DRMs

Datasets. NCI-50 datasets (chemical compounds and their activities)

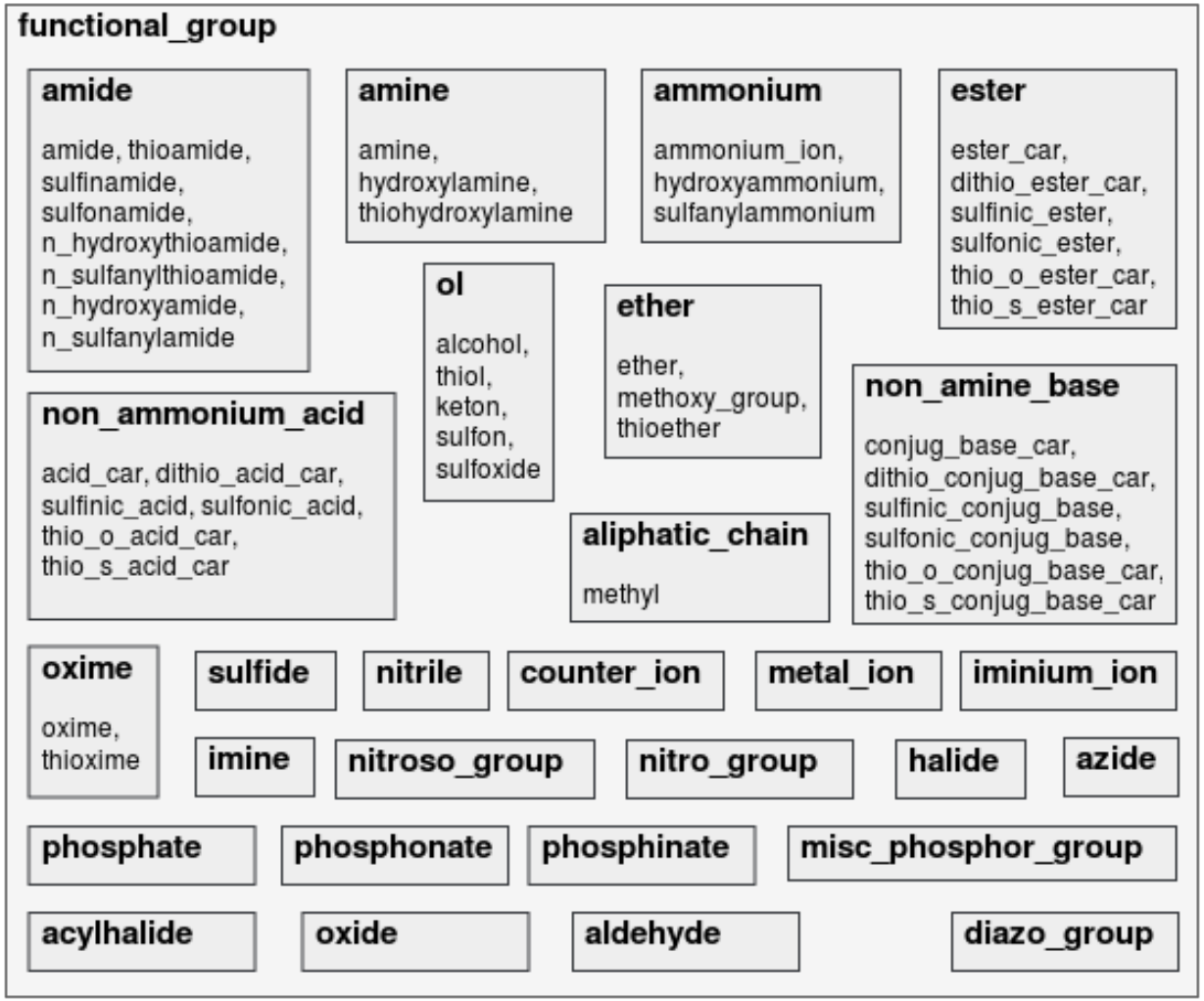
- Number of datasets: 73 (approx. $220,000$ instances)
- A summary:

Avg. # of instances	Avg. # of atoms per instance	Avg. # of bonds per instance	% of positives
3032	24	51	0.4–0.9

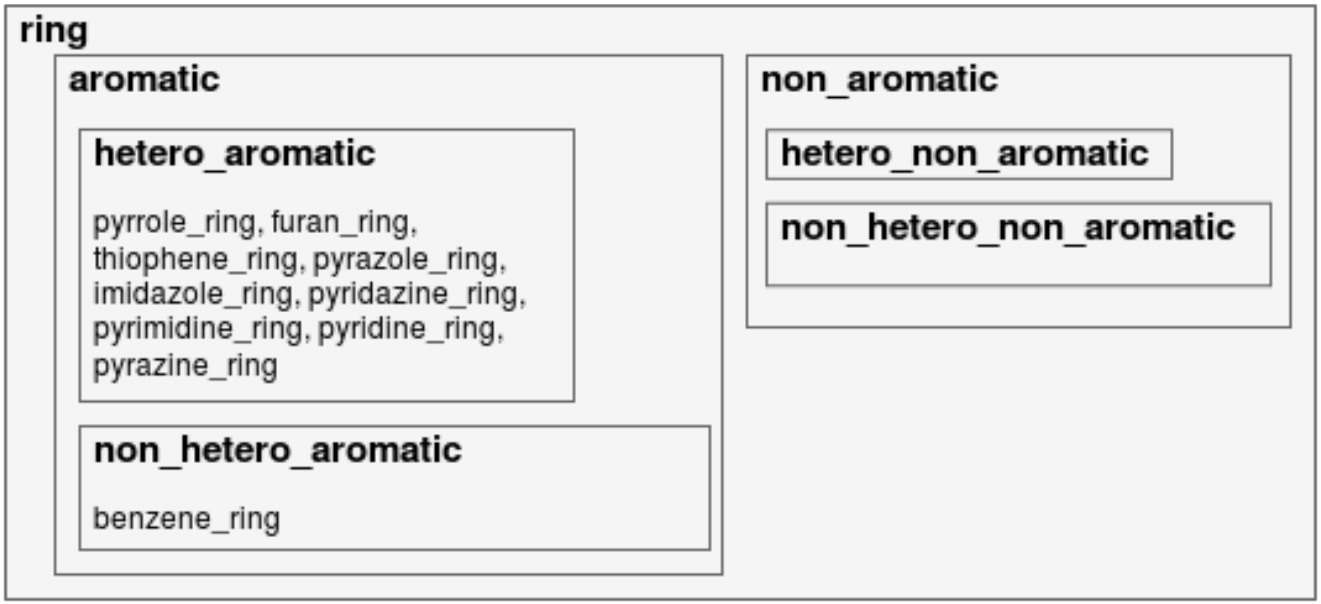
- Each compound has an associated anti-cancer activity (positive or negative).



Background Knowledge. Facts and definitions of chemical structures



Empirical Evaluation of DRMs

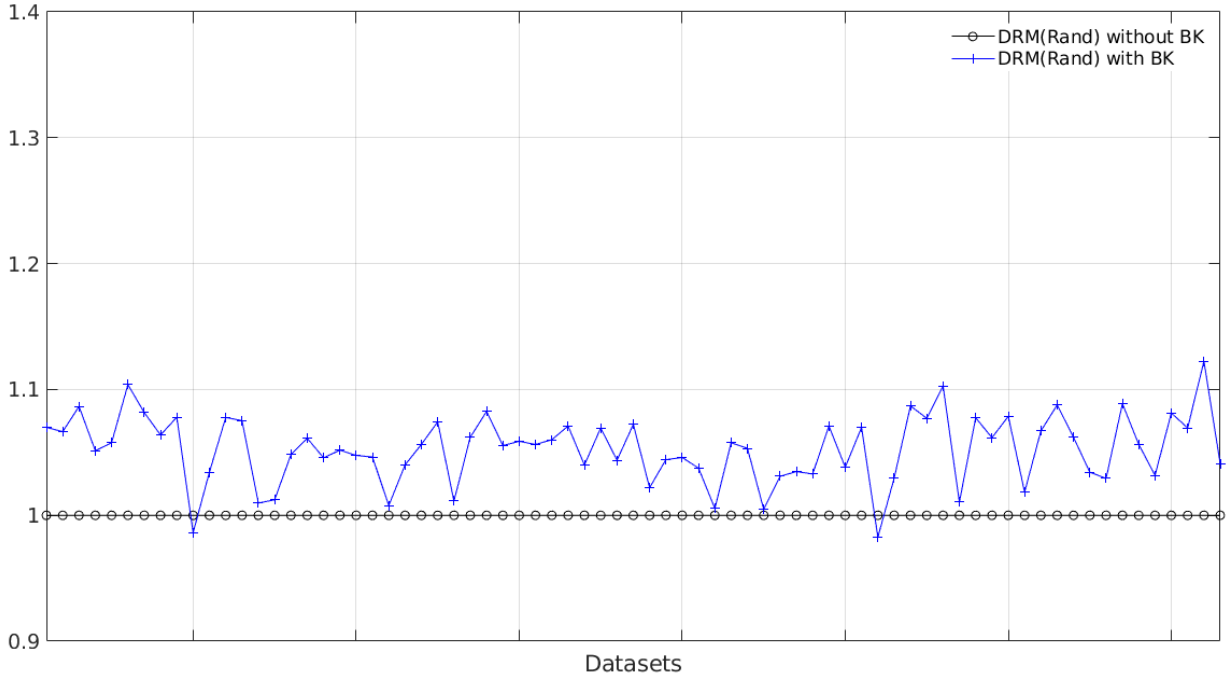


In overall, we have about 100 domain-relations.

Empirical Evaluation of DRMs

Results.

- DRM with domain-relations is substantially better than a DRM without domain-relations.



Gain in predictive performance (Y-axis).

Higher/Lower/Equal: 71/2/0 ($p < 0.001$).

Empirical Evaluation of DRMs

- Number of relational features affects DRMs' performance.

# of Features	Higher/Lower/Equal (p-value)
50	43/18/14 (< 0.01)
100	50/14/9 (< 0.01)
250	48/21/4 (< 0.01)
500	51/21/1 (< 0.01)
1000	44/25/4 (< 0.01)
2500	50/21/2 (< 0.01)
3800	39/22/1 (0.22)

Comparing DRM (Hide-and-Seek) against DRM (Simple random).

Empirical Evaluation of DRMs

Additional results.

- DRMs (hide-and-seek) are significantly better than known approaches to neuro-symbolic modelling:
  - LRNNs
  - BCP-based MLPs

Sourek *et al* (2018): Lifted relational neural networks: Efficient learning of latent relational structures, *JAIR*.

Franca *et al* (2014): Fast relational learning using bottom clause propositionalization with artificial neural networks, *MLJ*.

Empirical Evaluation of DRMs

Limitations.

- DRMs require logically expressive features.
- DRMs cannot achieve relational composition of features.
- Construction of DRMs is cost heavy.

# of 'good' features	$\alpha = 0.99$	$\alpha = 0.95$	$\alpha = 0.90$
1000	43709	28434	21855

Other costs, not shown here, are: Test for subsumption equivalence, evaluation of features for their utility, and propositionalisation.

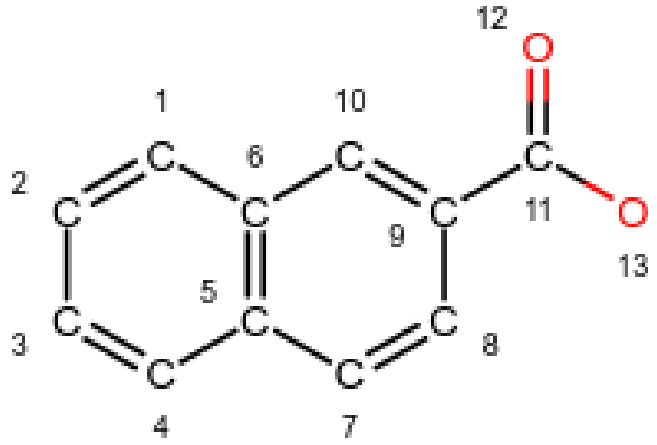
- Structural information of a data instance is lost due to propositionalisation.

Simplified Inclusion of Relational Information using Vertex-Enrichment

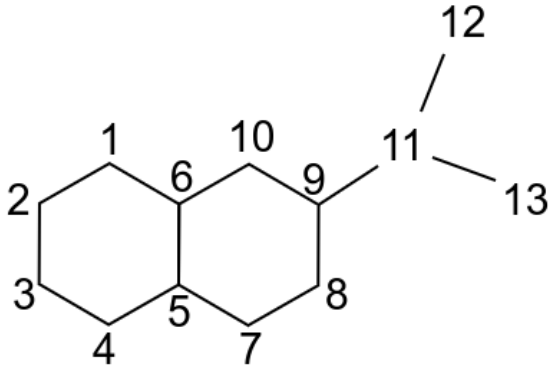
Implementation: Vertex-Enriched Graph Neural Networks (VEGNNs)

VEGNNs

An example of a relational data instance:

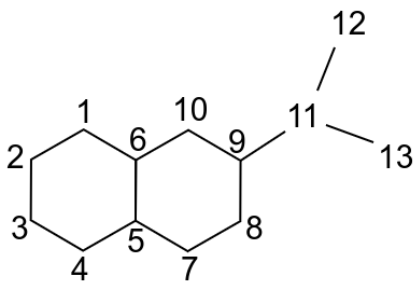


A corresponding molecular graph representation of this molecule is:



VEGNNs

The molecular graph can be represented as a bunch of relational facts (in Prolog, for example).

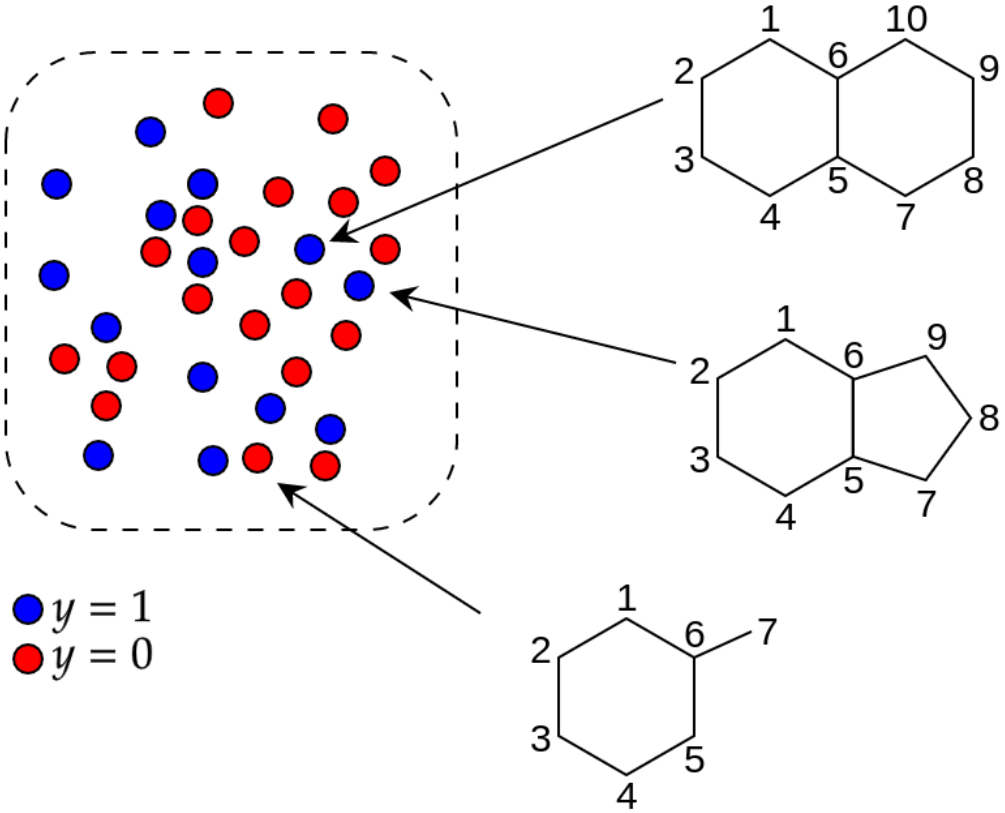


(Re-drawing here for clarity)

```
atom(m1,1,c).
atom(m1,2,c).
...,
atom(m1,13,o).
bond(m1,1,2,double).
bond(m1,2,3,single).
...,
bond(m1,11,13,single).
```

VEGNNs

Graph Neural Networks (GNNs) can operate on graphs.



Graph space: Each instance (● or ●) is a graph.

VEGNNs

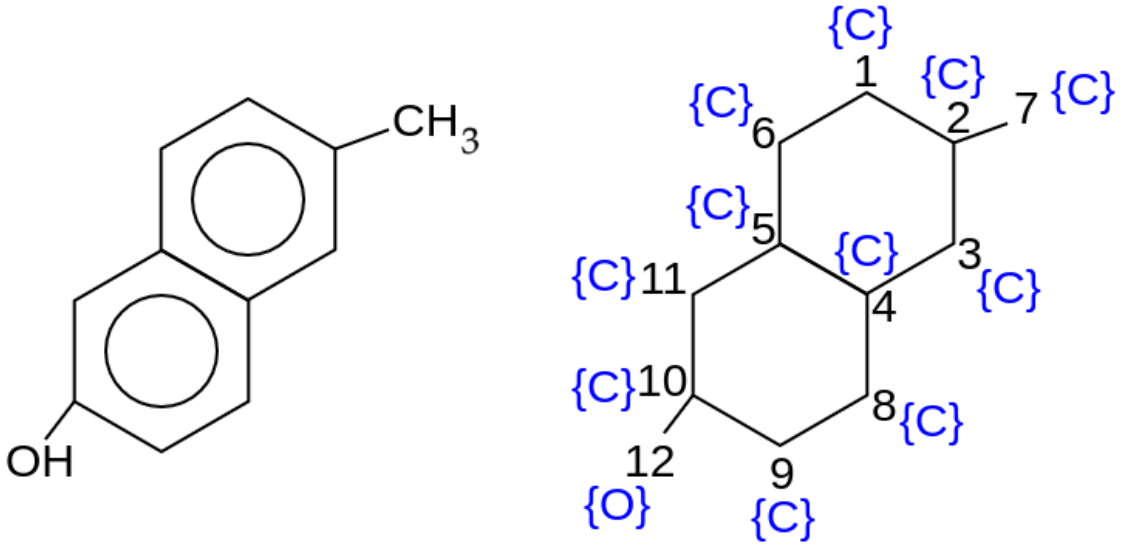
Vertex-Enrichment.

- The symbolic domain-knowledge consists of a set of relations.
- Each relation is treated as a hyperedge.
- Inferring these hyperedges in a graph as present (TRUE) or absent (FALSE).
- Enriches the vertex labellings of the graph with these hyperedge information.

For a graph $G = (V, E)$, a hyperedge h is a non-empty subset of V.

VEGNNs

- Let's consider a molecule m and its corresponding labelled molecular graph:



(a) (b)

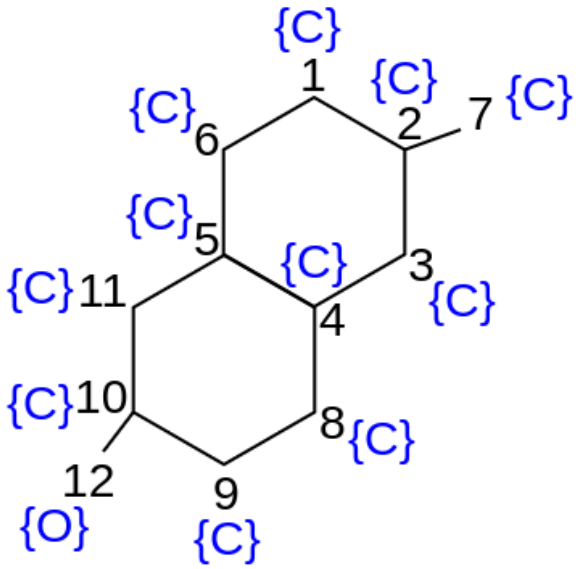
- This is a relational data instance.

VEGNNs

- Let's assume that our domain-knowledge consists of the definitions for the following relations:
  - Benzene ring
  - Methyl group
  - Alcohol group
  - Connected structures
  - Fused ring

VEGNNs

- Inferring the domain relations for m:
  - R_1 (Benzene ring): $\{1, 2, 3, 4, 5, 6\}$
  - R_2 (Benzene ring): $\{4, 5, 8, 9, 10, 11\}$
  - R_3 (Methyl group): $\{7\}$
  - R_4 (Alcohol group): $\{12\}$
  - R_5 (Connected str.): $\{2, 7, 10, 12\}$
  - R_6 (Fused ring): $\{4, 5\}$
- Notice that each relation is a hyperedge.



(Redrawn for clarity)

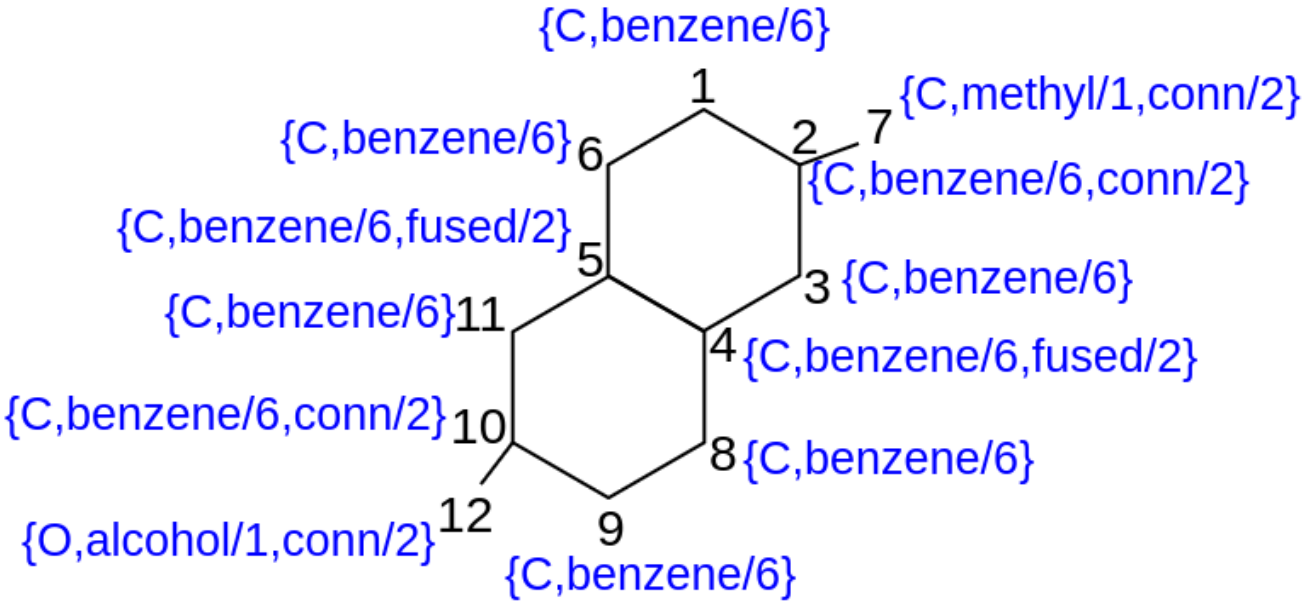
VEGNNs

- Vertex-enrichment with domain relations:

  Vertex v_1 is a member of R_1;
  Vertex v_2 is a member of R_1 and R_5;
  Vertex v_3 is a member of R_1;
  Vertex v_4 is a member of R_1, R_2 and R_6;
  ...
  Vertex v_{12} is a member of R_4, R_5

VEGNNs

- Enriching the vertices with the domain-relations:

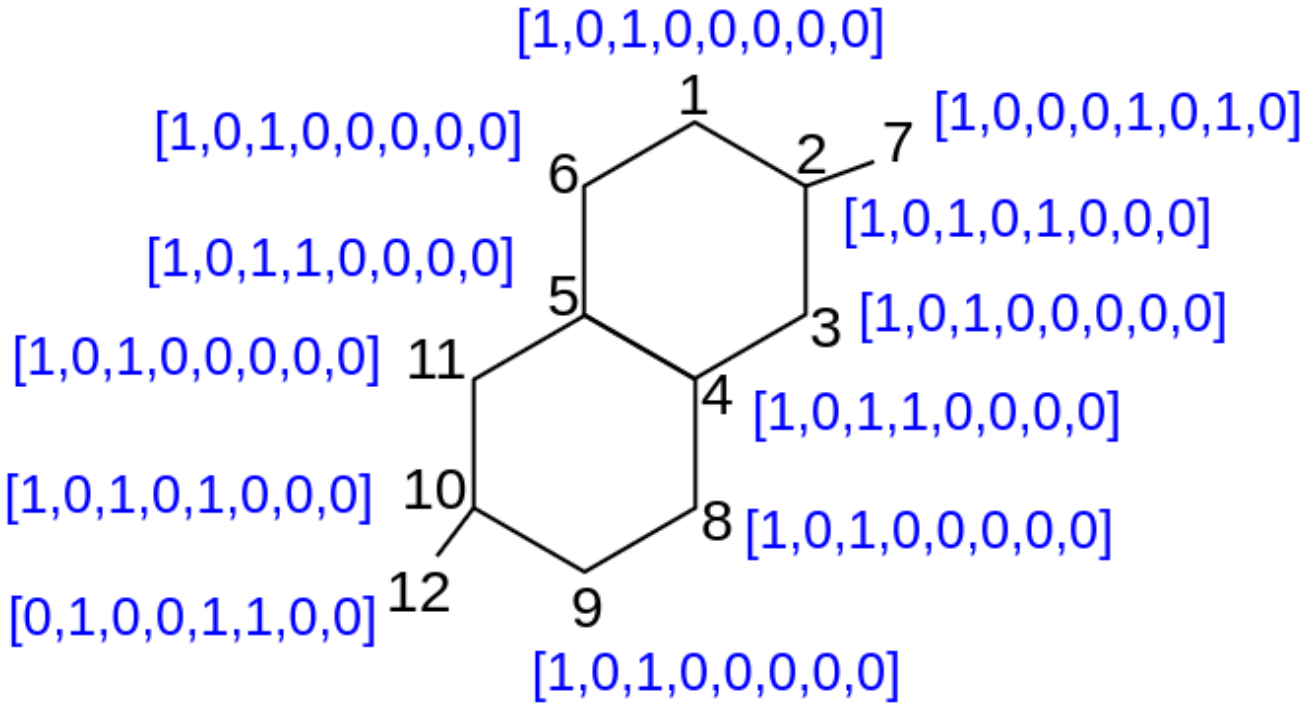


(Vertex-enriched graph)

- But, a GNN cannot operate on this graph.

VEGNNs

- A GNN can operate on this graph:



(Vectorised Vertex-enriched graph)

- VEGNN: A GNN constructed using vectorised vertex-enriched graphs.

Empirical Evaluation of VEGNNs

Data and Background Knowledge. Same as in our study on DRMs.

Variants of GNNs. 5 different GNNs

GNN variant: The graph convolution and pooling operator adopted for implementation.

Empirical Evaluation of VEGNNs

Results.

- GNNs with domain-knowledge (VEGNNs) are better than GNNs without domain-knowledge.

GNN Variant	Accuracy (*VEGNN* vs. *GNN*) Higher/Lower/Equal (p-value)
GNN_1	48/14/11 (< 0.001)
GNN_2	48/19/6 (0.005)
GNN_3	53/11/9 (< 0.001)
GNN_4	54/12/7 (< 0.001)
GNN_5	43/19/11 (0.002)

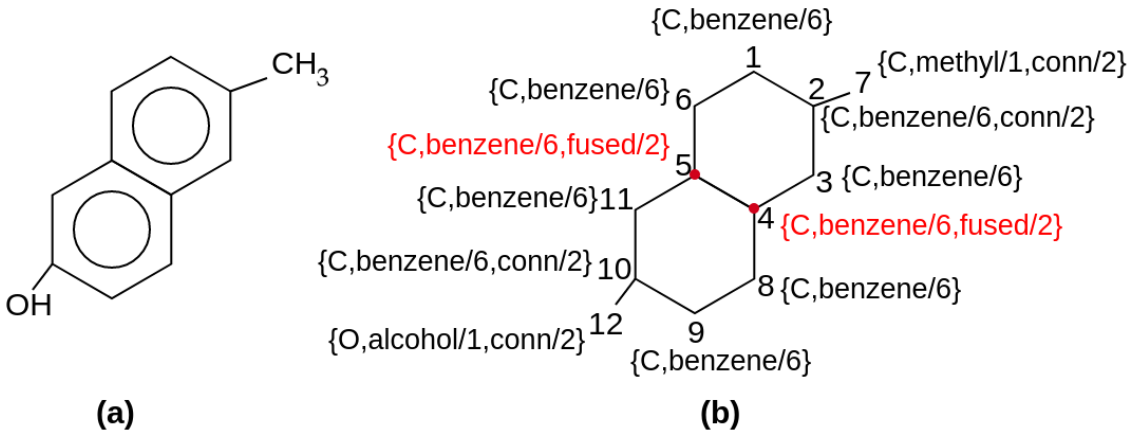
Empirical Evaluation of VEGNNs

Additional Results.

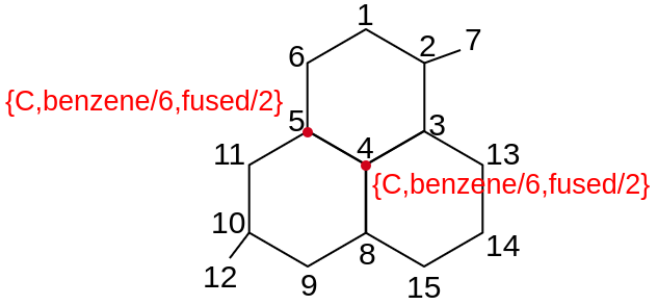
- VEGNNs are comparable to DRMs constructed with small number of relational features.
- VEGNNs are better than BCP-based MLPs that require up to 50000 features.

Limitations.

- Vertex-enrichment simplifies domain-knowledge.



Similarly,



Complete Inclusion of Relational Information using Inverse Entailment

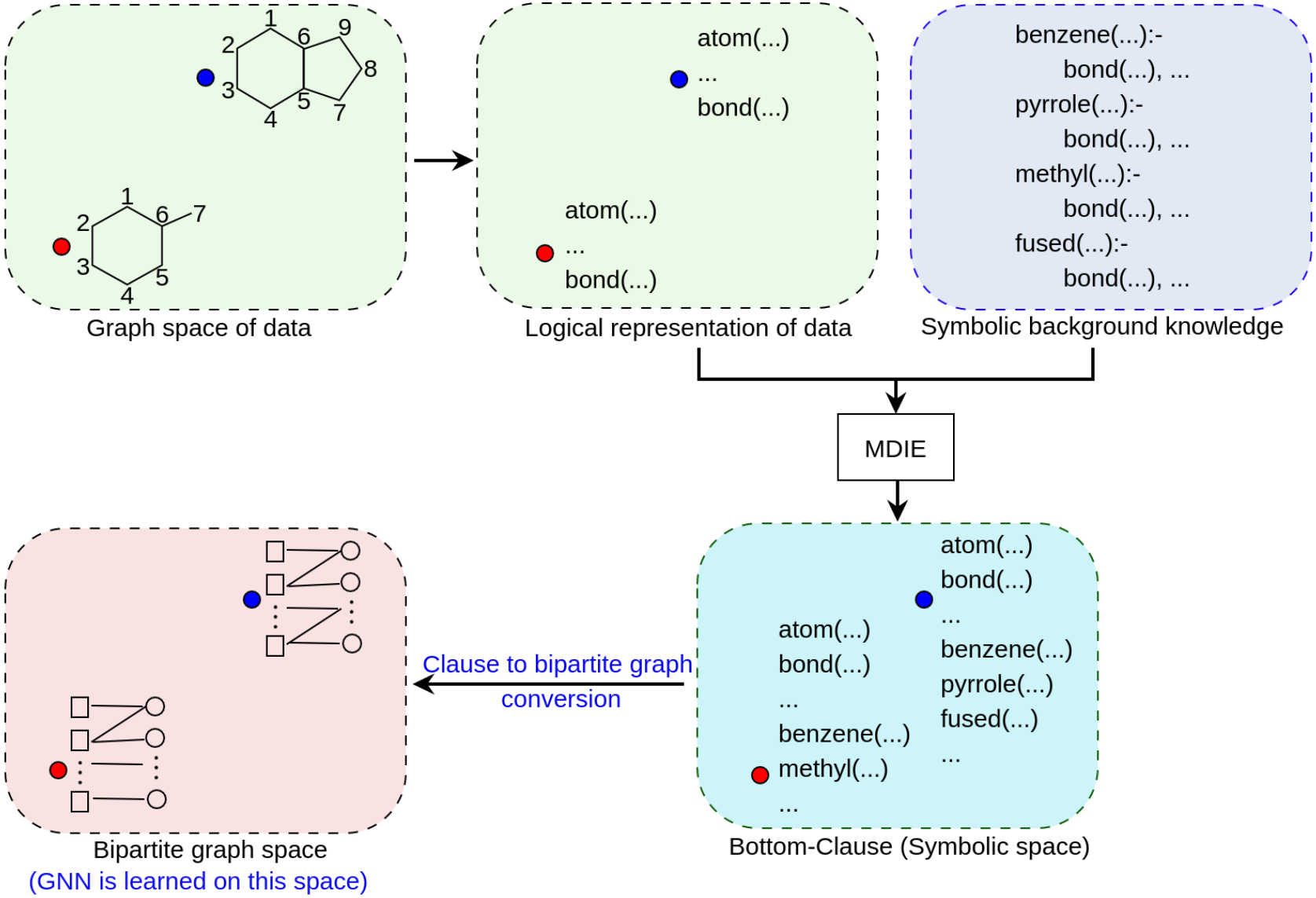
Implementation: Bottom-Graph Neural Networks (BotGNNs)

BotGNNs

1. Given a relational data instance e, background knowledge B, and a mode language $\mathcal{L}_{\mathrm{M},d}$ (d is depth-limit), MDIE identifies a most-specific logical formula $\bot_{B,\mathrm{M},d}(e)$ that contains all the relational information in B that is related to e.
2. We propose a method to transform $\bot_{B,\mathrm{M},d}(e)$ into a "bottom-graph" (a bi-partitite graph).
3. A standard GNN can then be learned using bottom-graphs.

MDIE: Muggleton (1995): Inverse entailment and progol, *New Gener. Comput.*

BotGNNs


1
2
3
4
5
6
7
8
9
Graph space of data
atom(...)
...
bond(...)
atom(...)
...
bond(...)
Logical representation of data
benzene(...):-
bond(...), ...
pyrrole(...):-
bond(...), ...
methyl(...):-
bond(...), ...
fused(...):-
bond(...), ...
Symbolic background knowledge
MDIE
atom(...)
bond(...)
...
benzene(...)
pyrrole(...)
fused(...)
...
atom(...)
bond(...)
...
benzene(...)
methyl(...)
...
Bottom-Clause (Symbolic space)
Clause to bipartite graph conversion
Bipartite graph space
(GNN is learned on this space)

BotGNNs

Step 1 and 2:

B:

$parent(X, Y) \leftarrow father(X, Y)$
$parent(X, Y) \leftarrow mother(X, Y)$

$mother(jane, alice) \leftarrow$

e:

$gparent(henry, john) \leftarrow$
$father(henry, jane),$
$mother(jane, john)$

M:

μ_1 = `modeh(gparent(+person,-person))`
μ_2 = `modeb(father(+person,-person))`
μ_3 = `modeb(mother(+person,-person))`
μ_4 = `modeb(parent(+person,-person))`

Let $d = 2$

BotGNNs

Before the inclusion of domain-predicates, e:

$$gparent(henry, john) \leftarrow father(henry, jane), mother(jane, john),$$

After the inclusion of domain-predicates using MDIE, $\bot_{B,\mathrm{M},2}(e)$:

$$\begin{aligned} gparent(henry, john) \leftarrow\ & father(henry, jane), mother(jane, john), \\ & mother(jane, alice), parent(henry, jane), \\ & parent(jane, john), parent(jane, alice) \end{aligned}$$

Meaning: e is enriched with $mother/2$ and $parent/2$ relations from B.

Step 3: Construction of bottom-graph

- Requires the idea of matching modes and matching types.
  - Literals in $\perp_{\mathrm{M},d}$ are matched against the modes
  - Ground terms in the literals are matched against the types

BotGNNs

$$gparent(henry, john) \leftarrow father(henry, jane), mother(jane, john), mother(jane, alice), parent(henry, jane), parent(jane, john), parent(jane, alice)$$

Literal (λ)	Mode (μ)
$\lambda_1 = gparent(henry, john)$	$\mu_1 = modeh(gparent(+person, -person))$
$\lambda_2 = father(henry, jane)$	$\mu_2 = modeb(father(+person, -person))$
$\lambda_3 = mother(jane, john)$	$\mu_3 = modeb(mother(+person, -person))$
$\lambda_4 = mother(jane, alice)$	$\mu_3 = modeb(mother(+person, -person))$
$\lambda_5 = parent(henry, jane)$	$\mu_4 = modeb(parent(+person, -person))$
$\lambda_6 = parent(jane, john)$	$\mu_4 = modeb(parent(+person, -person))$
$\lambda_7 = parent(jane, alice)$	$\mu_4 = modeb(parent(+person, -person))$

(literals and matching modes)

BotGNNs

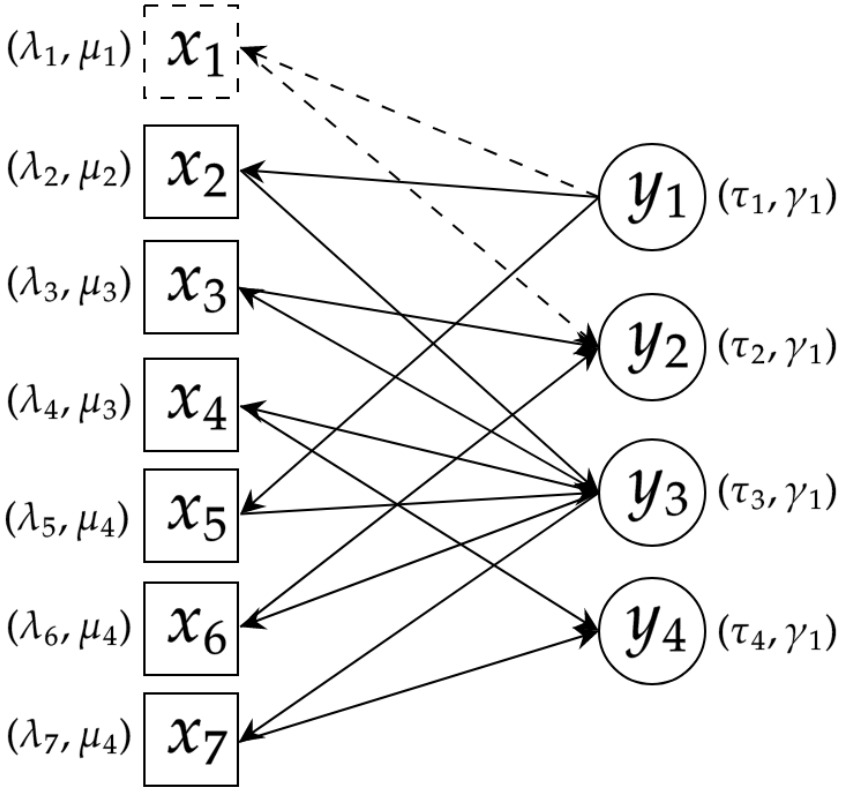
Similarly,

Term (τ)	Type (γ)
$\tau_1 = \textit{henry}$	$\gamma_1 = \textit{person}$
$\tau_2 = \textit{john}$	$\gamma_1 = \textit{person}$
$\tau_3 = \textit{jane}$	$\gamma_1 = \textit{person}$
$\tau_4 = \textit{alice}$	$\gamma_1 = \textit{person}$

(ground terms and matching types)

BotGNNs

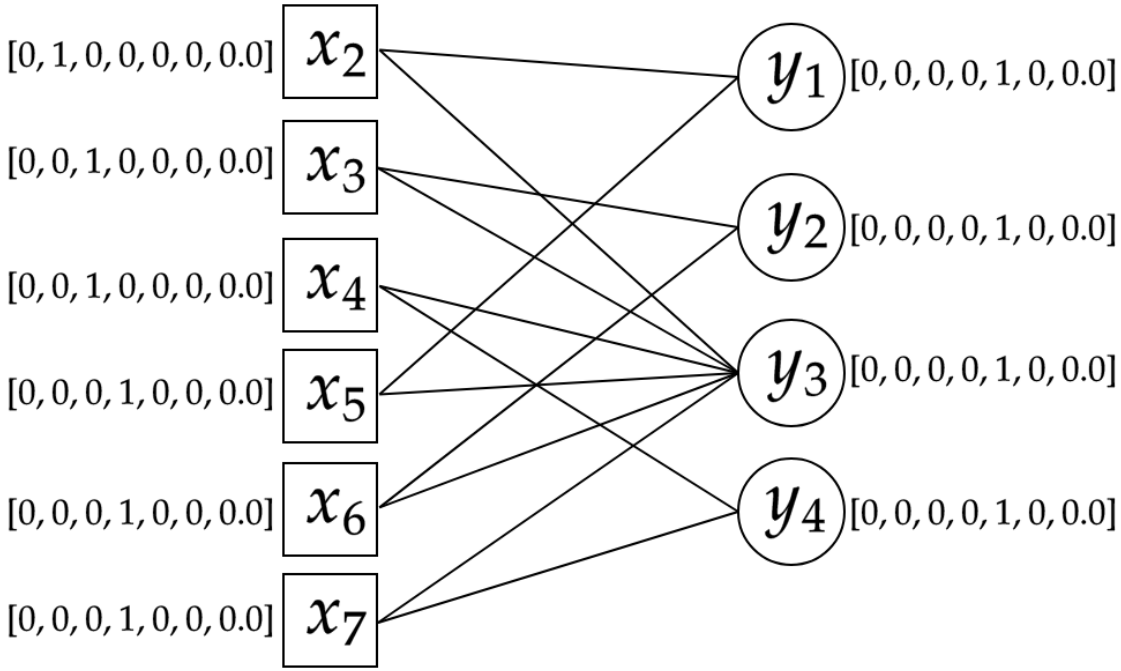
The bottom-graph constructed from $\perp_{B,\mathrm{M},2}(e)$:



But, a GNN cannot work on this graph.

BotGNNs

A GNN can work on this graph (BotGNN graph):



Empirical Evaluation of BotGNNs

Data and Background Knowledge. Same as before.

- A summary of the bottom-graph datasets:

Avg # of instances	Avg. of $\|X\|$	Avg. of $\|Y\|$	Avg. of $\|E\|$
3032	81	42	937

Variants of GNNs. 5 (same as in VEGNNs).

Empirical Evaluation of BotGNNs

Results.

- GNNs with domain-knowledge (BotGNNs) are better than GNNs without domain-knowledge.

GNN Variant	Accuracy (*BotGNN* vs. *GNN*) Higher/Lower/Equal (p-value)
1	59/5/9 (< 0.001)
2	59/8/6 (< 0.001)
3	61/2/10 (< 0.001)
4	63/1/9 (< 0.001)
5	60/4/9 (< 0.001)

Empirical Evaluation of BotGNNs

Additional Results.

- BotGNNs are superior to VEGNNs.
- BotGNNs are better than DRMs.
- BotGNNs are better than BCP-based MLPs.
- BotGNNs are comparable to ILP.

BotGNN as a System Component in a Human-in-the-Loop Setting

Implementation: A modular system for molecule generation

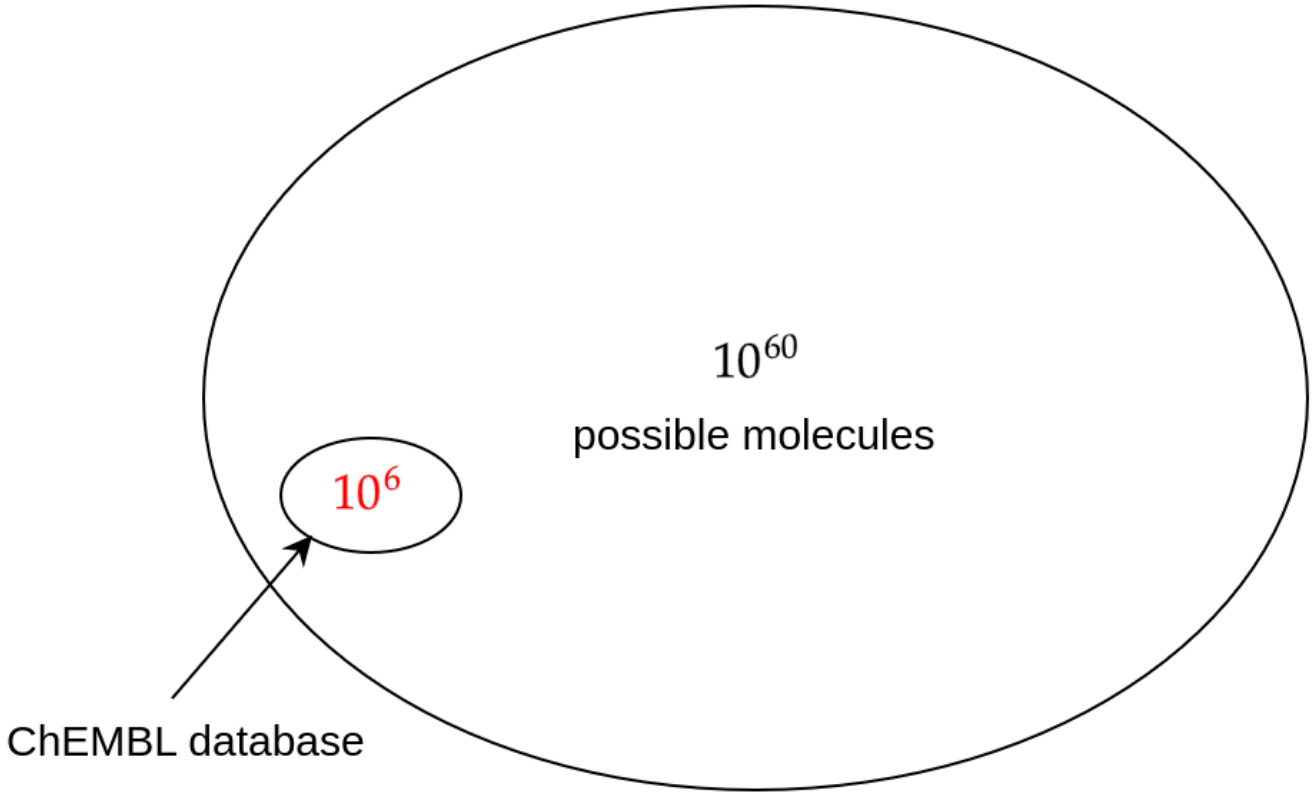
A modular system for molecule generation

The Problem.

- To generate new small molecules which could act as inhibitors of a biological target (JAK2 protein).
- There is limited prior information on the target-specific inhibitors.
- We want to investigate whether domain-knowledge can assist in generating such molecules.

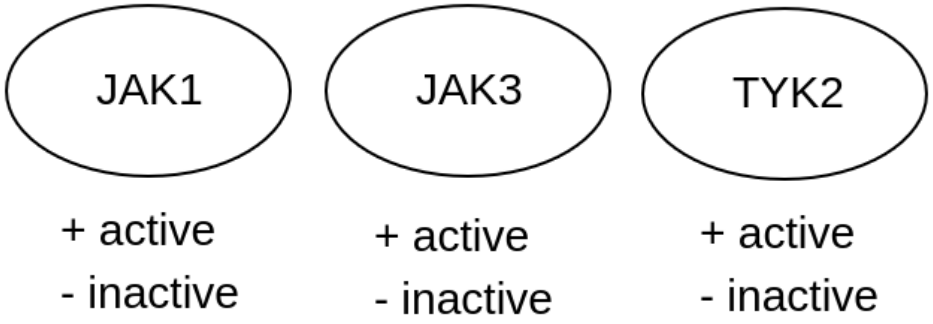
A modular system for molecule generation

Searching the space of molecules:



A modular system for molecule generation

What is available to us?

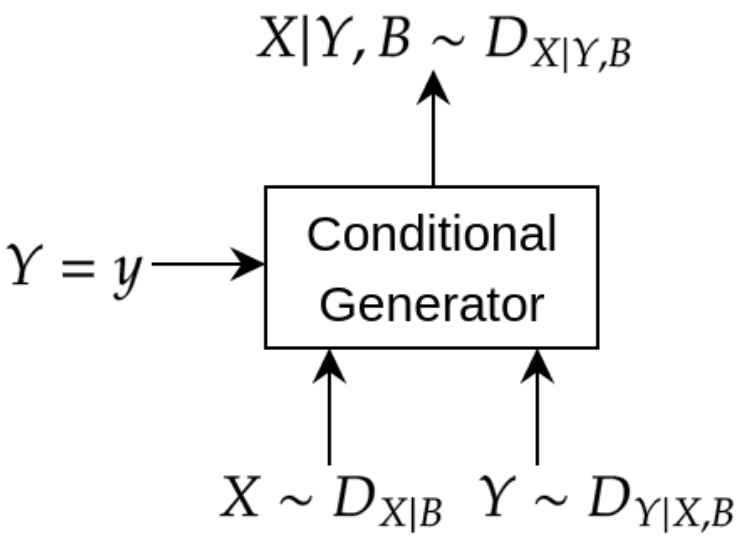


(JAK-homologous proteins)

A modular system for molecule generation

The Idea.

- Molecules and their activities are instances of r.v. X and Y (resp.)
- We want to draw instances from the conditional distribution $D_{X|Y,B}$

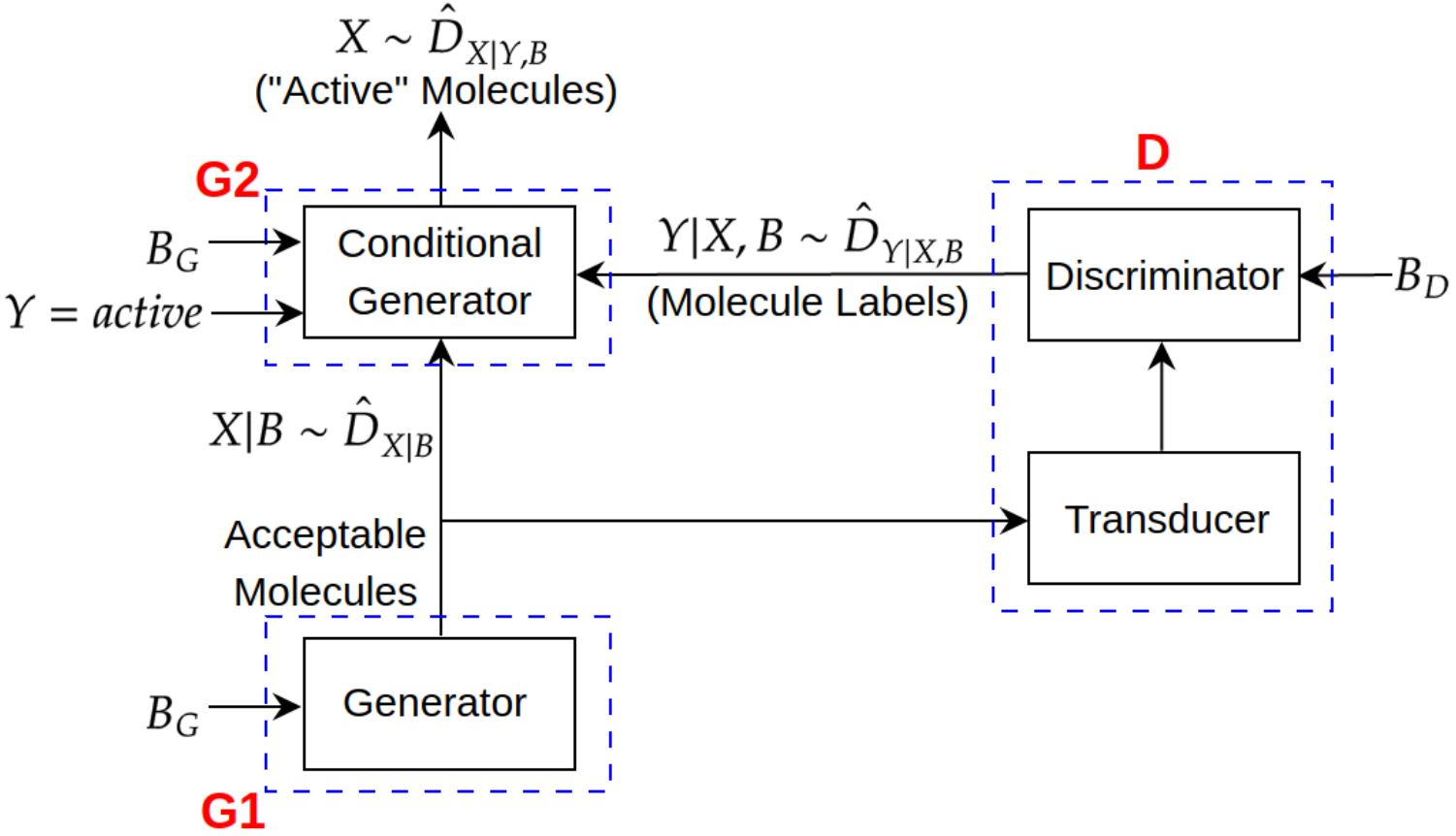


(Conditional generation of data)

Goal: To approximate these distributions using DNNs and a BotGNN

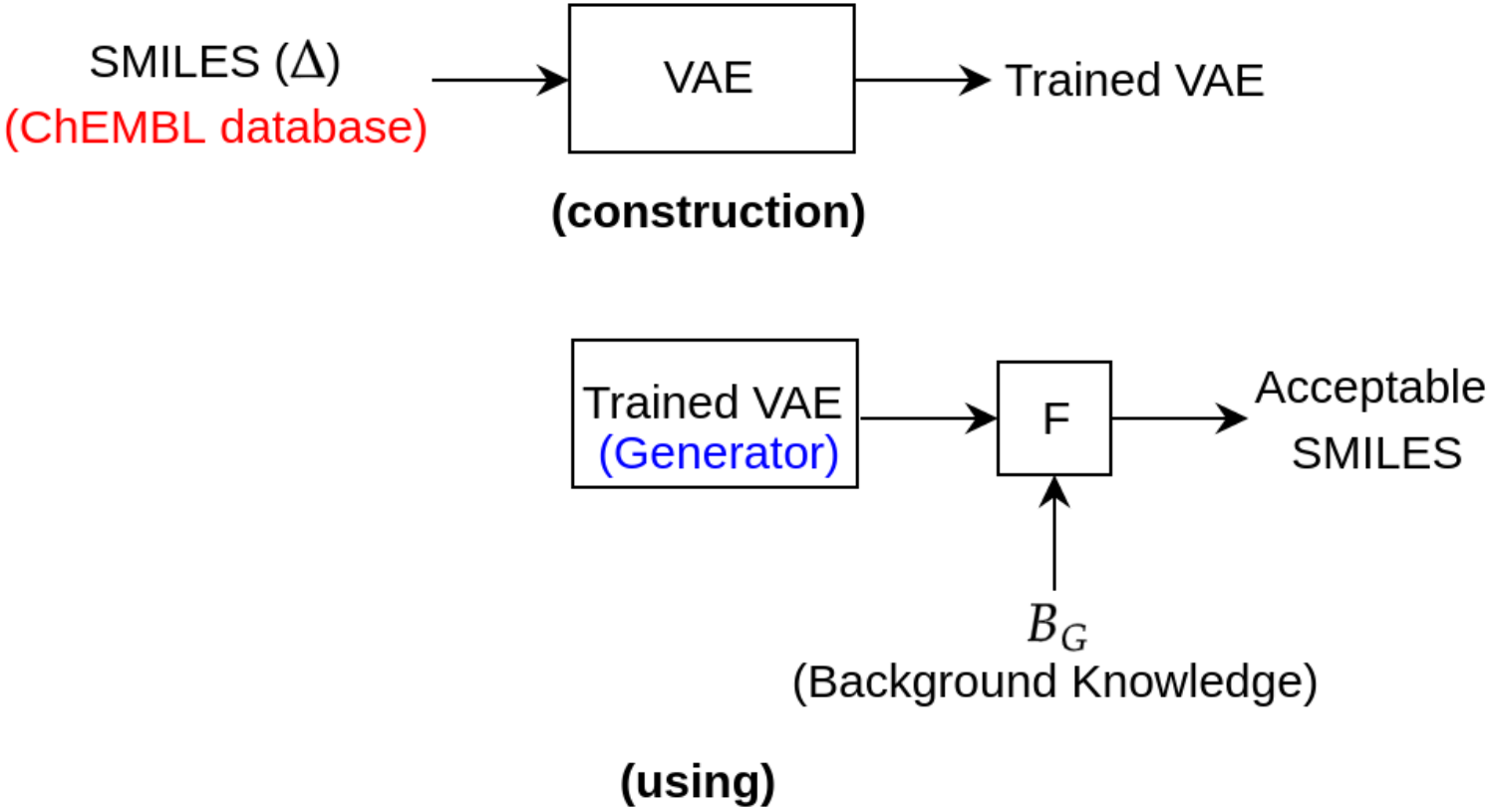
A modular system for molecule generation

System Design. Approximating the distributions



A modular system for molecule generation

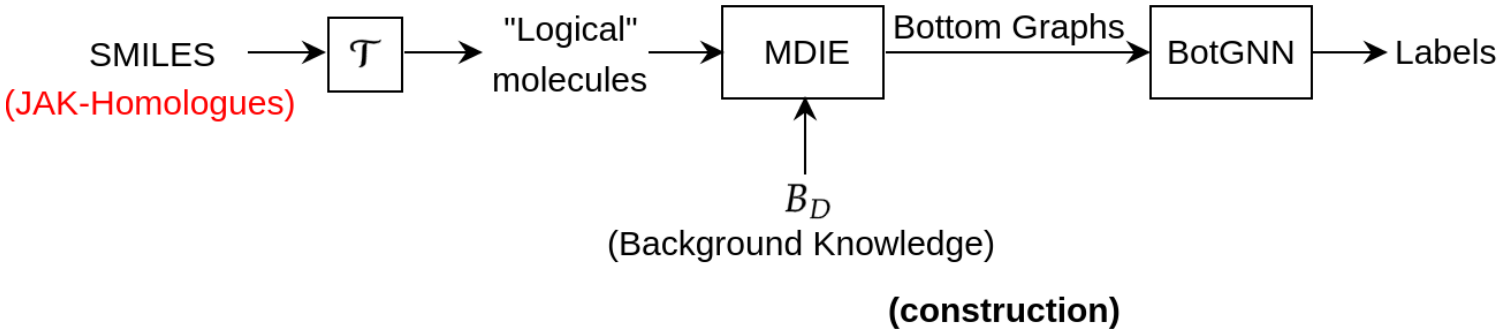
Generator $G1$: Generating acceptable molecules

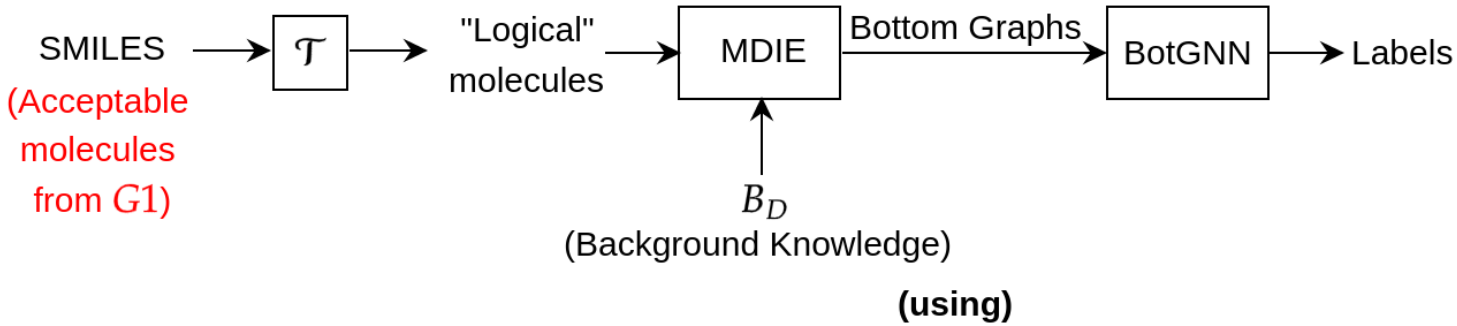


F: Filter

A modular system for molecule generation

Discriminator D: Obtaining labels for acceptable molecules

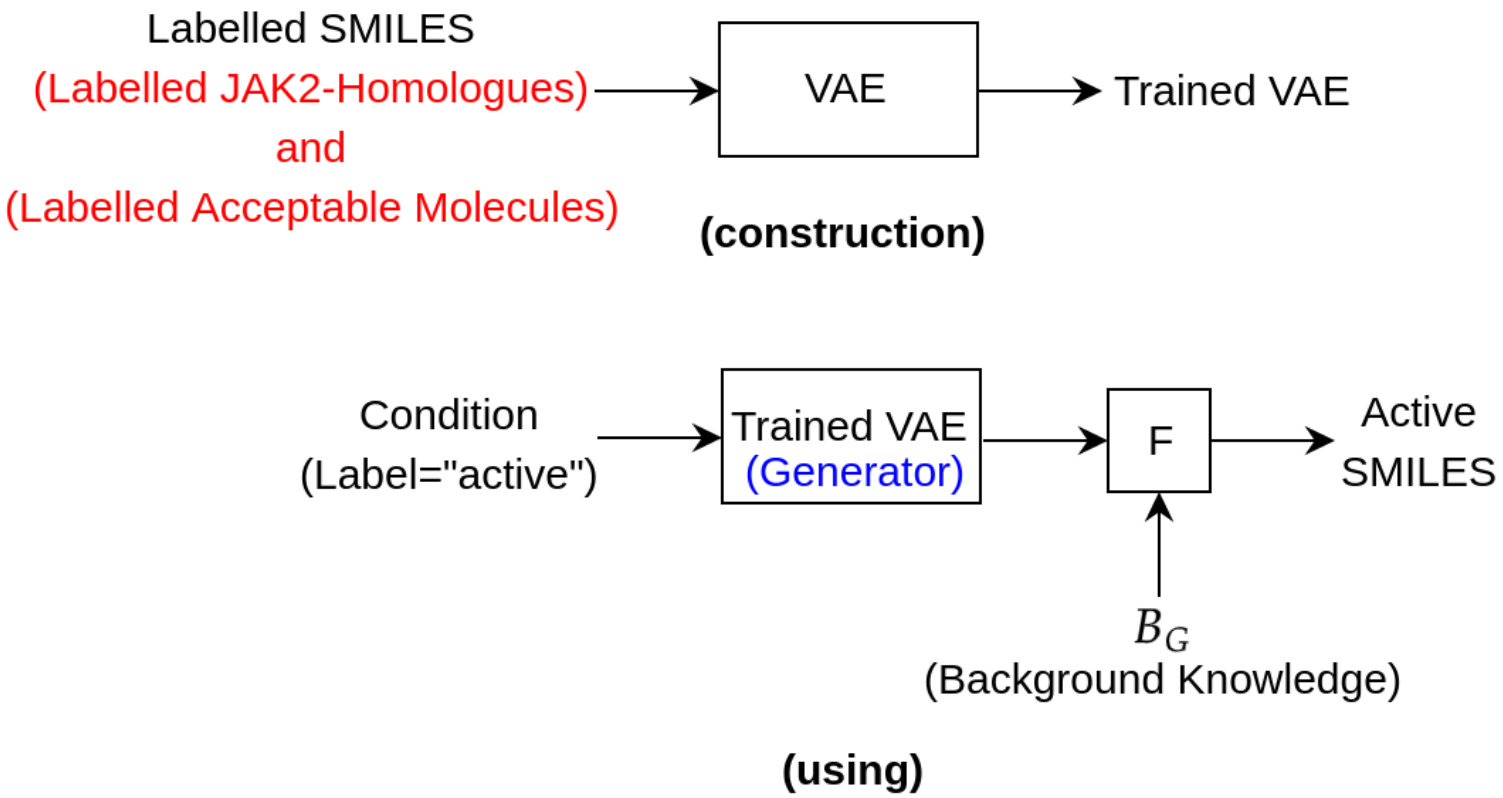




$\mathcal{T}$: a tranducer program

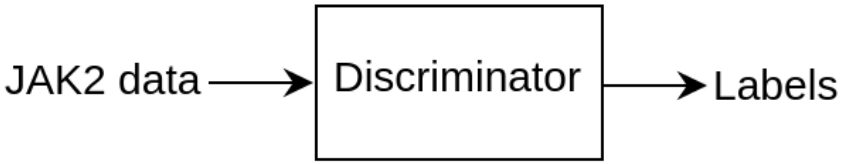
A modular system for molecule generation

Generator G_2: Generating active molecules



A modular system for molecule generation

Proxy model to evaluate the molecules sampled from $G2$:



Discriminator is a Chemprop model.

Stokes et al. (2020): A deep learning approach to antibiotic discovery, *Cell*.

Empirical Evaluation of the System

Data.

- Δ: CHEMBL database (1.9M unlabelled SMILES)
- JAK2-Homologues: 4300 labelled SMILES
- JAK2 data: 4100 labelled SMILES (for the proxy model)

Background Knowledge.

- B_G: Constraints on bulk-molecular properties from the literature
- B_D: Functional groups, rings, fused and connected structures

Generators.

- G_1 and G_2: LSTM-based Variational Autoencoder (LSTM-VAE)
- D: BotGNN

Empirical Evaluation of the System

Evaluation: Quantitative

As compared to the state-of-the-art approach:

- Our system generates significantly higher proportion of *active* molecules that are *active* for JAK2 inhibition.
- Our system generates significantly higher proportion of molecules that are *similar* to JAK2 inhibitors

Krishnan et al. (2021): Accelerating de novo drug design against novel proteins using deep learning, *J. Chem. Inf. Model.*

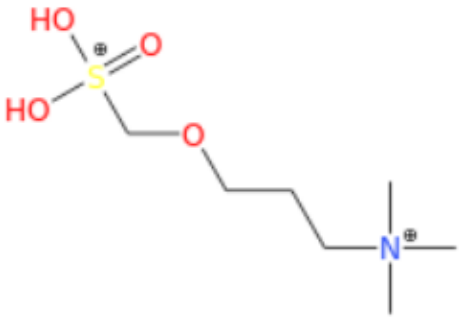
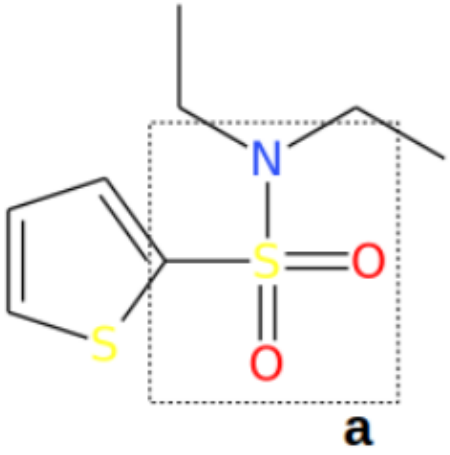
Empirical Evaluation of the System

Evaluation: By an expert (computational chemist)

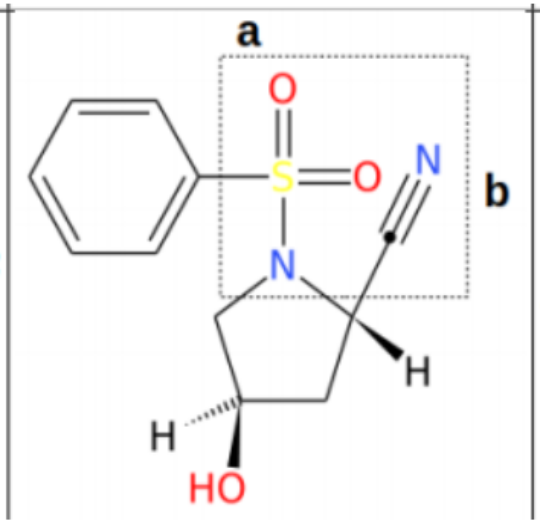
- 10 generates molecules where evaluated:
  - 5 similar to JAK2 inhibitors
  - 5 dissimilar to JAK2 inhibitors
- The expert picked 3 molecules, dissimilar to JAK2 inhibitors that were novel and worth investigating further.

Empirical Evaluation of the System

- Dissimilar and highly active molecules:

ID	Structure	Descriptors	Assessment
551		$Act = 9.12$ $Sim = 0.15$	This molecule has very low similarity to known JAK2 inhibitors. Also none of the groups specific to JAK2 could be identified by the substructure search. Discard this molecule.
1548		$Act = 9.04$ $Sim = 0.22$	This molecule has very low similarity to known JAK2 inhibitors. Also none of the groups specific to JAK2 could be identified by the substructure search. However, the sulfonamide group commonly found in JAK family inhibitors was found to be present (highlighted)

Empirical Evaluation of the System

1562		$Act = 9.49$ $Sim = 0.32$	Despite low similarity to existing JAK2 inhibitors, 1562 had one JAK2-selective subgroup and a group common to JAK inhibitors, indicating potential to act as JAK family inhibitor, but the selectivity to JAK2 cannot be confirmed. Possibly interesting new scaffold (highlighted) and worth pursuing further.

Fan mail:

I just saw your preprint ... ***the last molecule seems indeed quite promising.***

From: a researcher at a prominent research lab in Europe

Concluding Remarks

Conclusion

- This thesis provides techniques for inclusion of symbolic domain-knowledge into deep neural networks.
- The main contributions of this thesis are:
  - Non-uniform sampling of relational features for a DRM
  - Simplified inclusion of domain-relations in a VEGNN
  - Complete inclusion of domain-relations in a BotGNN
  - A modular system for molecule generation that uses a BotGNN

Conclusion

- The large-scale empirical evaluation of all our proposed techniques validates our primary hypothesis:

  Inclusion of domain-knowledge by changing the data representation can significantly improve the performance of a deep neural network.

Repository: https://github.com/tirtharajdash/NeSy

Conclusion

- Possible future directions:
  - Domain-knowledge could be sentences in a natural language.
  - Extended studies that go beyond prediction.

January 2017 – April 2022.

Thank You.

(All my teachers, colleagues, friends, family, . . . , and 786_0.)

Appendix

DRMs

Additional Results:

- DRMs with hide-and-seek are significantly better than known approaches to neuro-symbolic modelling.

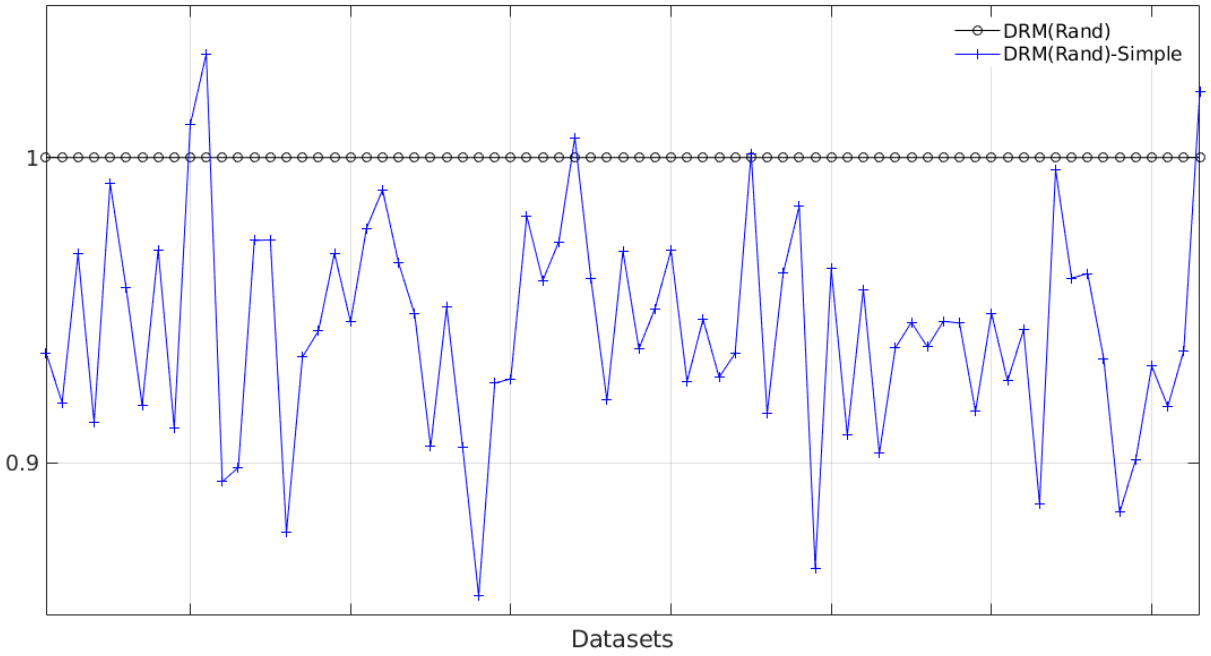
DRM (Hide-and-Seek) # of features	Accuracy (DRM vs. other methods) Higher/Lower/Equal (p-value)	
	LRNN	BCP+MLP
3800	68/5/0 (< 0.001)	69/2/2 (< 0.001)

Comparing DRM against LRNN [SAZ+18] and BCP-based MLP [FZG14].

DRMs

Limitations:

- DRMs require logically expressive features.



- Further, for any language with sufficient expressive power, it is intractable to provide all features within the language.

DRMs

- DRMs cannot achieve relational composition of features (i.e., relational join).

  Example: A neuron taking two features:

  $C_1 : (P(X) \leftarrow (has\_car(X, Y), short(Y)))$, and
  $C_2 : (P(X) \leftarrow (has\_car(X, Y), closed(Y)))$

  cannot produce
  $C : (P(X) \leftarrow (has\_car(X, Y), short(Y), closed(Y)))$

  but, it will produce an approximation to
  $C' : (P(X) \leftarrow (has\_car(X, Y), has\_car(X, Z), short(Y), closed(Z)))$

  i.e., $C'' : \sigma(w_1 C_1 + w_2 C_2 + w_0)$

DRMs

- Construction of DRMs is cost heavy.

# of Features	$\alpha = 0.99$		$\alpha = 0.95$		$\alpha = 0.90$	
	$p = 0.1$	$p = 0.5$	$p = 0.1$	$p = 0.5$	$p = 0.1$	$p = 0.5$
1000	43709	6644	28434	4322	21855	3322
2000	87418	13288	56867	8644	43709	6644
3000	131127	19932	85300	12966	65564	9966
4000	174835	26576	113733	17288	87418	13288

Other costs, not shown here, are: Test for subsumption equivalence, evaluation of features for their utility, and propositionalisation.

VEGNNs

- VEGNNs are comparable to DRMs constructed with small number of relational features.

GNN Variant	Accuracy (*VEGNN* vs. *DRM*) Higher/Lower/Equal (p-value)		
	$\lvert\mathcal{R}'\rvert = 50$	$\lvert\mathcal{R}'\rvert = 100$	$\lvert\mathcal{R}'\rvert = 250$
GNN_1	59/13/1 (< 0.001)	50/22/1 (< 0.001)	21/52/0 (< 0.001)
GNN_2	49/23/1 (< 0.01)	39/33/1 (0.81)	19/54/0 (< 0.001)
GNN_3	54/18/1 (< 0.001)	44/28/1 (0.05)	14/59/0 (< 0.001)
GNN_4	59/13/1 (< 0.001)	52/20/1 (< 0.001)	23/50/0 (< 0.001)
GNN_5	53/19/1 (< 0.001)	42/30/1 (0.06)	17/56/0 (< 0.001)

VEGNNs

- VEGNNs are better than BCP-based MLPs that require up to 50000 features.

GNN Variant	Accuracy (*VEGNN* vs. BCP+MLP) Higher/Lower/Equal (p-value)
GNN_1	51/21/1 (< 0.001)
GNN_2	46/26/1 (0.08)
GNN_3	48/24/1 (0.003)
GNN_4	54/18/1 (< 0.001)
GNN_5	47/25/1 (0.005)

BotGNNs

- BotGNNs are superior to VEGNNs:

GNN Variant	Accuracy (*BotGNN* vs. *VEGNN*) Higher/Lower/Equal (p-value)
1	54/11/8 (< 0.001)
2	61/9/3 (< 0.001)
3	54/10/9 (< 0.001)
4	55/11/7 (< 0.001)
5	52/9/12 (< 0.001)

BotGNNs

- BotGNNs are better than DRMs.

GNN Variant	Accuracy (*BotGNN* vs. *DRM*) Higher/Lower/Equal (p-value)			
	$NumFeats = 50$	$\cdots$	$NumFeats = 500$	$NumFeats = 1000$
GNN_1	64/8/1 (< 0.001)	$\cdots$	46/27/0 (0.15)	39/34/0 (0.98)
GNN_2	63/9/1 (< 0.001)	$\cdots$	31/42/0 (0.17)	29/44/0 (0.05)
GNN_3	65/7/1 (< 0.001)	$\cdots$	42/31/0 (0.66)	37/36/0 (0.46)
GNN_4	65/7/1 (< 0.001)	$\cdots$	43/30/0 (0.18)	40/33/0 (0.72)
GNN_5	67/5/1 (< 0.001)	$\cdots$	44/29/0 (0.26)	36/37/0 (0.83)

BotGNNs

- BotGNNs are better than BCP-based MLPs.

GNN Variant	Accuracy (*BotGNN* vs. BCP+MLP) Higher/Lower/Equal (p-value)
1	58/10/5 (< 0.001)
2	58/11/4 (< 0.001)
3	61/6/6 (< 0.001)
4	62/6/5 (< 0.001)
5	60/6/7 (< 0.001)

BotGNNs

- BotGNNs are not a replacement for ILP; after all, it relies on MDIE.

GNN Variant	Accuracy (*BotGNN* vs. ILP) Higher/Lower/Equal (p-value)
1	62/7/4 (< 0.001)
2	60/9/4 (< 0.001)
3	61/7/5 (< 0.001)
4	62/6/5 (< 0.001)
5	62/4/7 (< 0.001)

Dataset	ILP	BotGNN
DssTox	0.73	0.76
Mutag	0.88	0.89
Canc	0.58	0.64
Amine	0.80	0.84
Choline	0.77	0.72
Scop	0.67	0.65
Toxic	0.87	0.85

The ILP results in the table on the right are from [SKB03].

BotGNNs

Limitations.

The size of the corresponding clause-graph is bounded by
$$(r|\mathrm{M}|j^{+}j^{-})^{dj^{+}}(1+j^{+}+j^{-}).$$

- j^{+}: An upper-bound on the number of $+$ arguments in *modeb* declarations in M and the number of $-$,$\#$ arguments in *modeh* declarations in M
- j^{-}: An upper-bound on the number of $-$,$\#$ arguments in *modeb* declarations in M and the number of $+$ arguments in *modeh* declarations in M
- r: Recall number

Based on: S. Muggleton, "Inverse entailment and Progol", *New Gener. Comput.*, 1995.

BotGNN as a System Component

Evaluation: Quantitative

- $|M|$: number of molecules drawn from the generators
- $|M'|$: number of acceptable found molecules (in M), that is, those satisfying constraints on molecular properties
- Act: proportion of M' that are predicted to be active (by the proxy model)
- Sim: proportion of molecules in M' that are similar to active JAK2 inhibitors (Tanimoto coefficient > 0.75)

BotGNN as a System Component

- Inclusion of domain-knowledge significantly improves conditional generation of molecules:

Qty.	$B_D = B_1$	$B_D = B_0$	*Random*
$\|M\|$	5000	5000	5000
$\|M'\|$	2058	2160	2877
Act	0.47 (0.01)	0.43 (0.01)	0.34 (0.01)
Sim	0.14 (0.01)	0.11 (0.01)	0.00 (0.00)

DeepRL [KBBR21]
10000
–
–
0.03 (0.001)

  - The quantities in bracket are standard deviation values.
  - $B_D = B_0$: Discriminator has no access to symbolic domain relations
  - $B_D = B_1$: Discriminator has access to symbolic domain relations

References

Manoel VM França, Gerson Zaverucha, and Artur S d'Avila Garcez.
Fast relational learning using bottom clause propositionalization with artificial neural networks.
Machine learning, 94(1):81–104, 2014.

Sowmya Ramaswamy Krishnan, Navneet Bung, Gopalakrishnan Bulusu, and Arijit Roy.
Accelerating de novo drug design against novel proteins using deep learning.
Journal of Chemical Information and Modeling, 61(2):621–630, 2021.

Gustav Sourek, Vojtech Aschenbrenner, Filip Zelezny, Steven Schockaert, and Ondrej Kuzelka.
Lifted relational neural networks: Efficient learning of latent relational structures.
Journal of Artificial Intelligence Research, 62:69–100, 2018.

References

Ashwin Srinivasan, Ross D King, and Michael E Bain.
An empirical study of the use of relevance information in inductive logic programming.
Journal of Machine Learning Research, 4(Jul):369–383, 2003.