Learning under Budget Generative Adversarial Nets

Nataliya Sokolovska

Sorbonne University
Paris, France

Master 2 BIM
January, 17, 2020

Outline

Learning under Budget

- Motivation for Learning under Budget
- Cascade Classifiers
- A deep reinforcement learning model
- MDP and POMDP
- Some Applications in Medicine

Generative Adversarial Learning

- Generative Adversarial Networks
- Example: a Real Materials Science Application
- Real Data and Discussion

Outline

Learning under Budget

- Motivation for Learning under Budget
- Cascade Classifiers
- A deep reinforcement learning model
- MDP and POMDP
- Some Applications in Medicine

Generative Adversarial Learning

- Generative Adversarial Networks
- Example: a Real Materials Science Application
- Real Data and Discussion

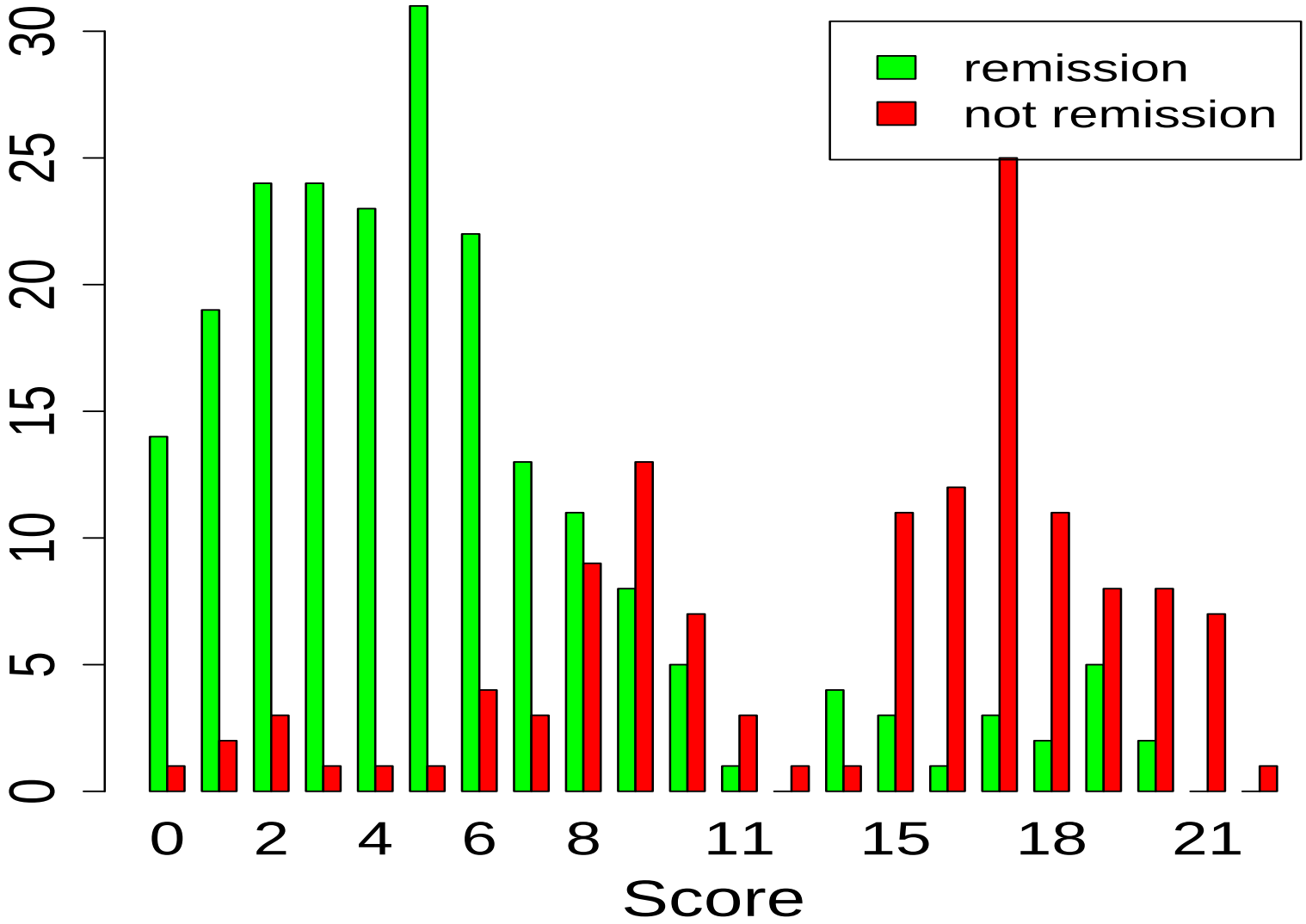
Example: the DiaRem (Diabetes Prediction) Score

Variable	Thresholds	Score
Age	<40	0
	40–49	1
	50 – 59	2
	>60	3
Glycated hemoglobin	<6.5	0
	6.5 – 6.9	2
	7 – 8.9	4
	> 9	6
Insuline	No	0
	Yes	10
Other drugs	No	0
	Yes	3

Classify as Remission if sum of scores < 7
Classify as Non-remission if sum of scores ≥ 7

C. D. Still et al., Preoperative prediction of type 2 diabetes remission after Roux-en-Y gastric bypass surgery: a retrospective cohort study, 2013

DiaRem accuracy on NutriOmics data



Classification under budget constraints

Problem: predict y from x when this is reliable; abstain otherwise.
The cost:

$$\delta = \begin{cases} 1 & \text{, if } \hat{y} = y, \\ \delta & \text{, if rejection,} \\ 0 & \text{, if } \hat{y} \neq y. \end{cases}$$

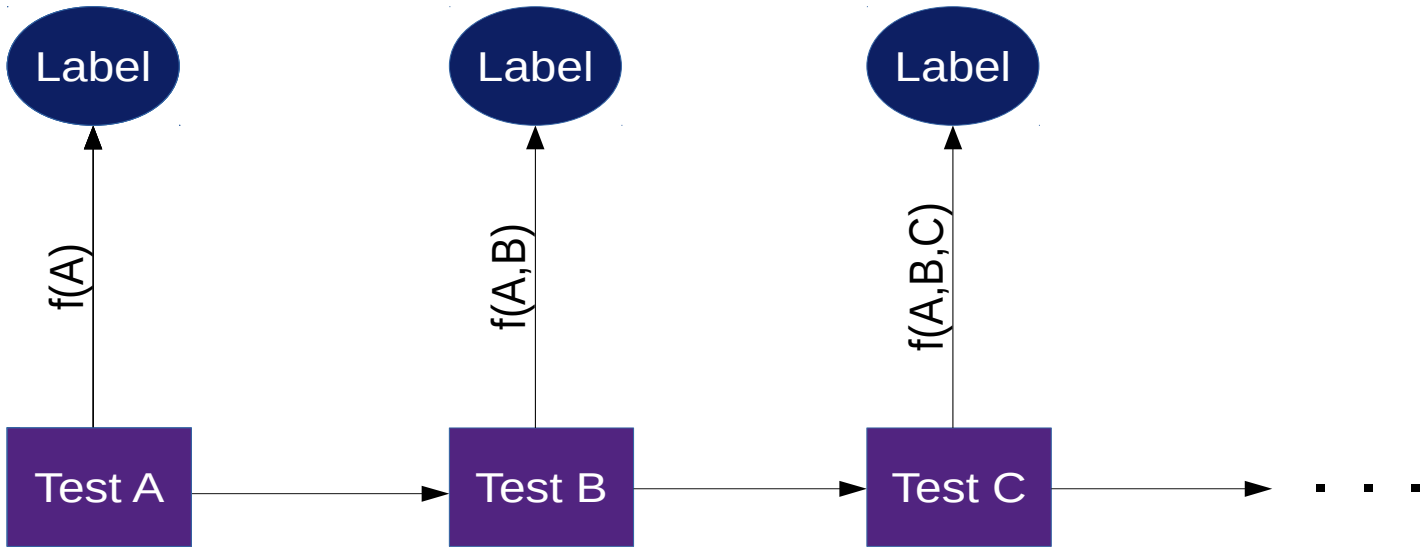
$\delta < 1/2$, $\hat{y}$ is a classifier.

The cost is used to avoid misclassification.

- Boosting with Abstention, C. Cortes, G. DeSalvo and M. Mohri, *Advances in Neural Information Processing Systems*, 2016:

`Many thanks to Matthieu Clertant for sharing his slides`

Homogeneous cascade with abstention



Tradeoff between accuracy and cost: The system incurs a penalty of δ_{k+1} at k-th stage if it rejects to seek more measurements. Order of tests and classifier with abstention are learned purely from the data.

The Costs

In real life application, budget is always limited.

Possible costs:

- time, money,
- side effects of medication,
- complexity (interpretability, signature disease) ...

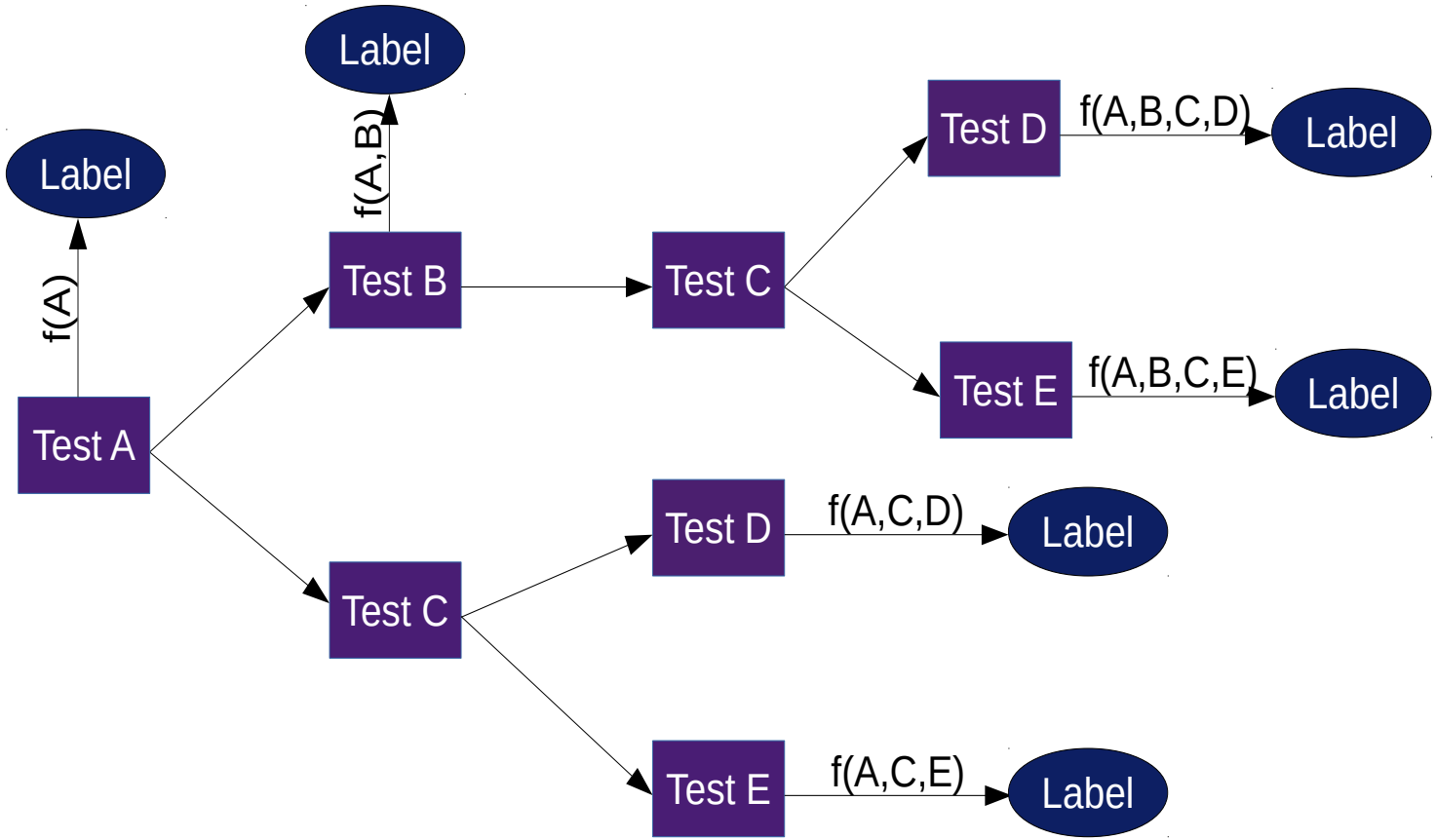
The costs δ_k depends of the tests or the nature of acquired data.

Sum of costs:

$$\Delta = \sum_{k=1}^{K} \delta_k .$$

Δ is a tradeoff parameter between accuracy and cost.

Heterogeneous Cascade with abstention



Tradeoff between accuracy and cost: Dynamic diagnostic protocols are individually tailored acquisitions of patients data with the aim to provide the most accurate diagnostics for the lowest cost.

Goals

The machine learning sytem has to provide:

- individualized treatment of patients,
- exploration of $K!$ homogeneous cascades,
- a tree structure or heterogeneous cascade,
- 2^K classifiers and 2^K rejecters,
- classifiers and rejecters jointly,
- interpretable classifiers,

The algorithm has to learn purely from data.

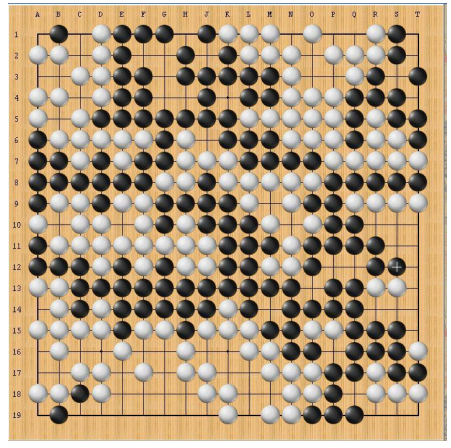
Strategy Games: Human vs Computer

Deep Blue (1997):

- it won against Gary Kasparov
- almost exhaustive exploration
- human knowledge (evaluation function, memory of real games)

Alpha Go (2015):

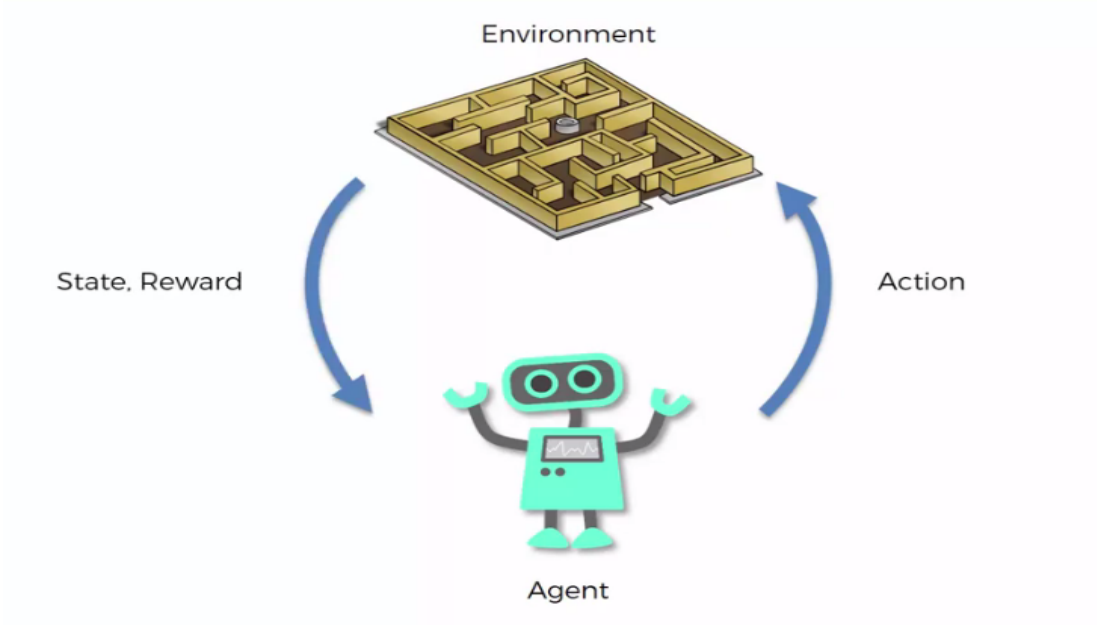
- it won against Lee Sedol
- no exhaustive exploration
- human knowledge (memory of real game)



Alpha Zero (2017): no human knowledge.

These board games are **Markov Decision Processes.**

Markov Decision Process



A Markov Decision Process is a 5-tuple $(S, A, P, C, \mathcal{X} \times \mathcal{Y})$.

- S : set of states,
- A : set of actions,
- P : set of conditional transition probabilities,
- C : cost function,
- $\mathcal{X} \times \mathcal{Y}$: set of observations.

From POMDP to MDP

Partially Observable Markov Decision Process deals with:

- uncertainty related to the effects of actions (cost).
- uncertainty about the current state.

Purely epsistemic MDP deals with the second uncertainty.

Historical state s_t:

- $a_{[t]}$, set of tests done at time t,
- $x_{[t]}$, all the features available at time t.

Purely epistemic MDP+ historical data (state)= simple MDP.

Deep Q Learning

Goal: Determinating a policy π of action.

Global cost C and cost from step t, noted C_t,:

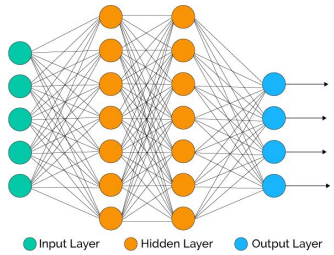
$$C(s_K) = \sum_{k \in [K+1]} c(s_k) \text{ and } C_t(s_K) = \sum_{k \in [t:K+1]} c(s_k)\,.$$

The function Q (also called cost-to-go to a_t) is the expected cumulative cost from time t to the end if we undertake action a_t:

$$Q^{\pi}(s_{t-1}, a_t) = \mathbb{E}_{\pi}\left[C_t(s_K) \mid s_{t-1}, a_t\right].$$

Optimal solution: $\pi^{\star} = \arg\min_{\pi} \mathbb{E}_{\pi}[C(s_K)]$.

The function $Q^{\pi^{\star}}$ is approximated with deep neural network.



Optimal policy

Theorem

Let $\hat{y}$ and $\hat{r}$ be two natural candidates in case of classification and rejection when the distribution D is known:

$$\hat{y}_t = \underset{a_t \in [L]}{\arg\max}\ \mathbb{P}_D(a_t = Y \,|\, s_{t-1}) \ \textit{and}\ \ \hat{r}_t = -\ \underset{a_t \in [-K:-1]}{\arg\min}\ \ Q^\pi(s_{t-1}, a_t)\,.$$

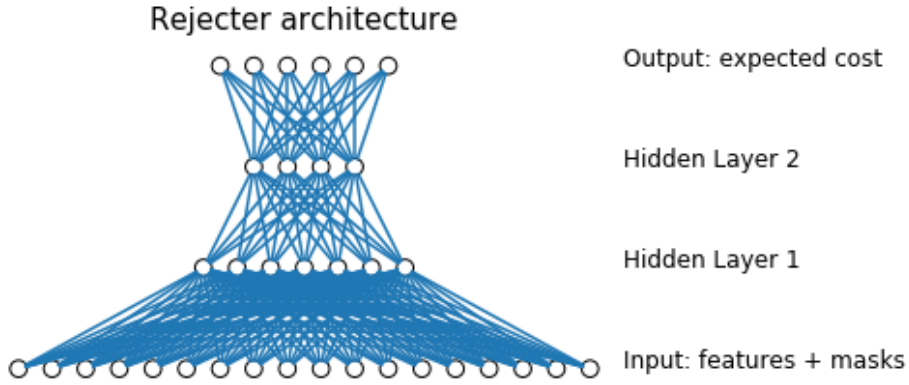
For all $t \in [K]$ (or to the end of the cascade), the optimal solution of the MDP system satisfies:

$$\pi^\star(s_{t-1}) = a_t^\star = \begin{cases} \hat{y}_t & \text{, if } \max\limits_{a_t > 0} \mathbb{P}_D(a_t = Y \,|\, s_{t-1}) \\ & \quad > 1 - \min\limits_{a_t < 0} Q^\pi(s_{t-1}, a_t) \\ \hat{r}_t & \text{, otherwise.} \end{cases}$$

Deep Neural Networks

Environment: dataset of patients features and final diagnostic.

Agent (Part 1):
Rejecter/selector



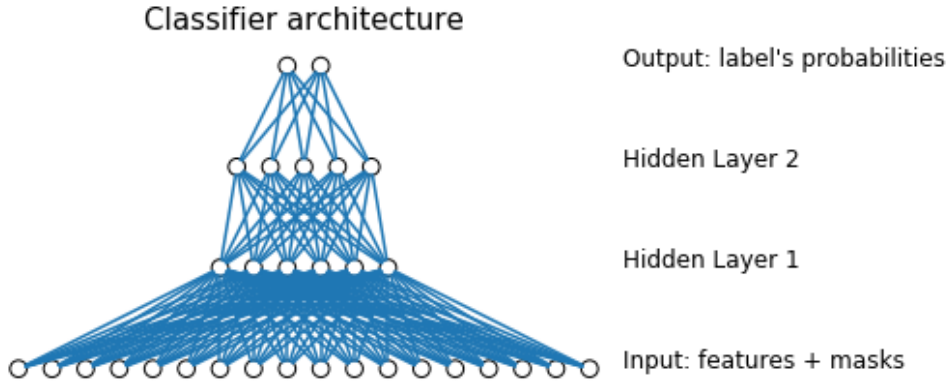
At time t in the cascade:

- inputs are summarized by $s_t = (a_{[t]}; x_{[t]})$ (historical state: tests+ features),
- outputs are the approximated cost-to-go to each available tests (function Q).

Deep Neural Networks

Environment: dataset of patients features and final diagnostic.

Agent (Part 2):
Classifier



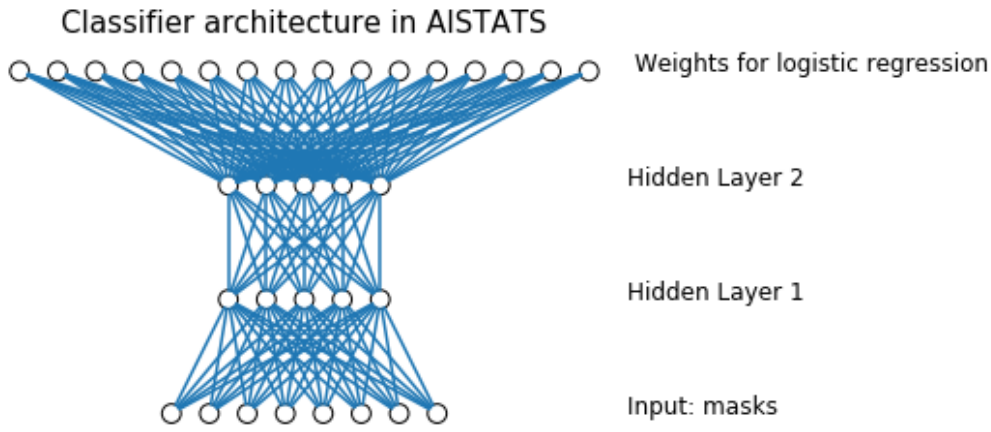
At time t in the cascade:

- inputs are summarized by $s_t = (a_{[t]}; x_{[t]})$ (historical state: tests+ features),
- outputs are the probability of each class (function Q).

Deep Neural Networks

Environment: dataset of patients features and final diagnostic.

Agent (Part 2):
Classifier



At time t in the cascade:

- inputs are summarized by $s_t = (a_{[t]}; x_{[t]})$ (historical state: tests+ features),
- final outputs are the probability of each class (function Q).

Familly of logistic regressions: $f_w(s_t, a) \propto \exp \langle \beta_a(M_{[t]}), M_{[t]} \odot x \rangle$.

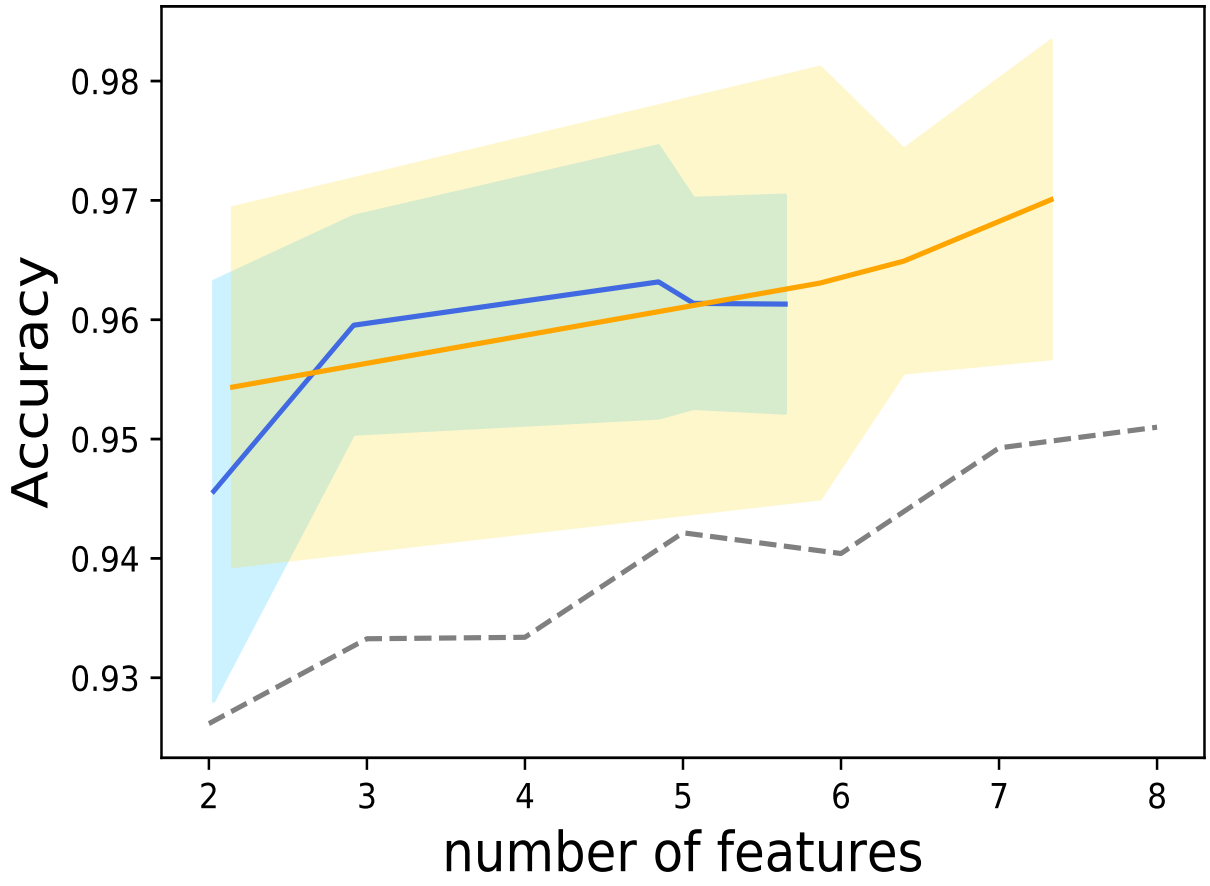
Experimentation setup

Three models:

- ICCA stands for Interpretable Cascade Classifier with Abstention; it uses a classifier network generating a family of softmax regression,
- p-ICCA, same model after a run of pretraining using only the mask for the rejecter network.
- Random forest with feature selection (Importance measure by mean decrease impurity).

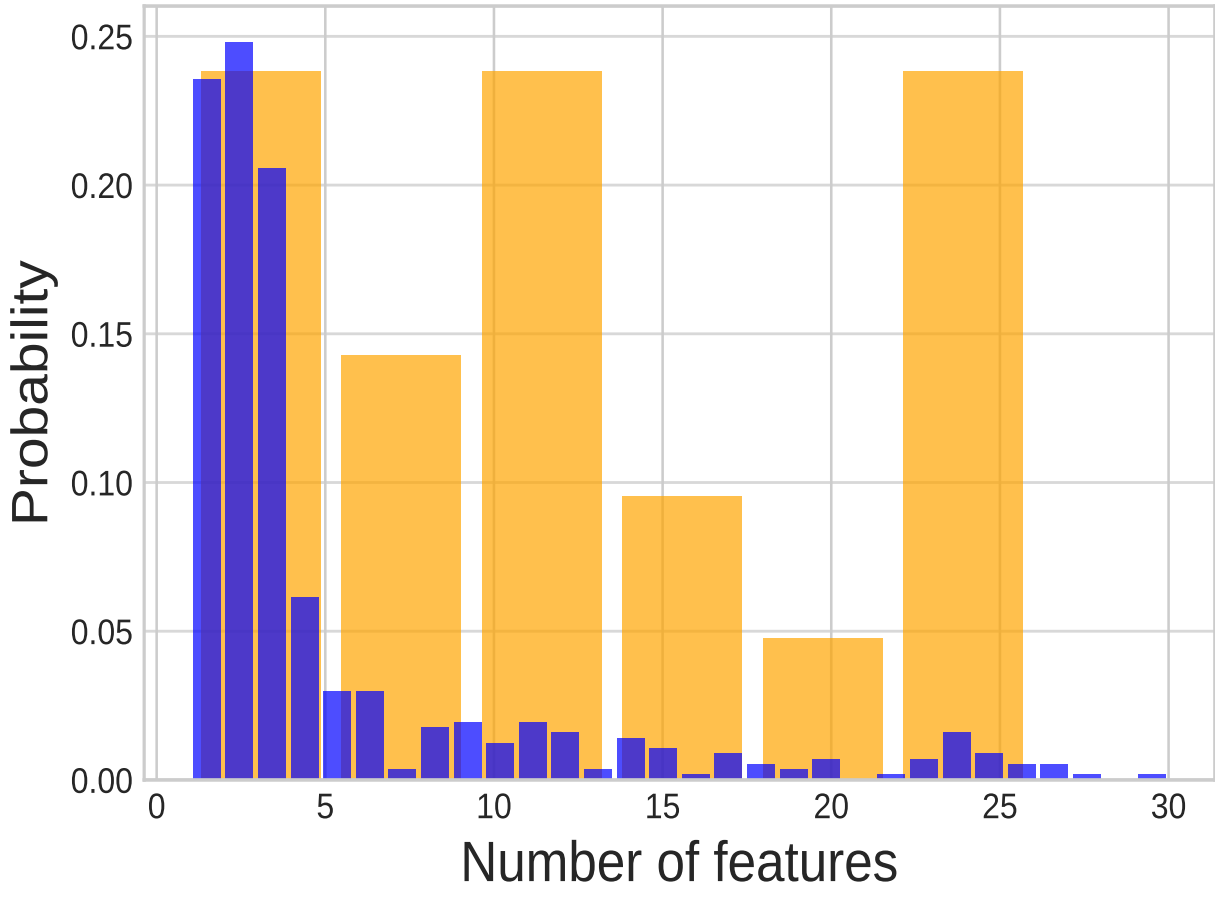
Data: *Breast Cancer Wisconsin Diagnostic (UCI)*. We dispose of 30 parameters describing characteristics of the cell nuclei and medical images for 569 patients. All parameters are continuous. Costs of feature acquisition are the same for all the features.

Data of breast cancer diagnostic



Accuracy and standard deviation for ICCA (in blue), pretrained ICCA(in yellow), and the random forest classifier (dotted line). The features share the same cost: $\delta \in [10^{-4}, 1/3.10^{-2}]$.

Data of breast cancer diagnostic



Histogram of the number of features explored for ICCA. In blue: all observations (size: 569, mean: 4.8); in orange: the misclassified observations (size: 21, mean: 12.3). Overall accuracy: 96,31%.

1 of 9 233,26%

Interpretable Cascade Classifiers with Abstention

Matthieu Clertant
NutriOmics, INSERM
Sorbonne University
Paris, France

Nataliya Sokolovska
NutriOmics, INSERM
Sorbonne University
Paris, France

Yann Chevaleyre
LAMSADE
Dauphine University
Paris, France

Blaise Hanczar
IBISC
Evry University
Evry, France

Abstract

In many prediction tasks such as medical diagnostics, sequential decisions are crucial to provide optimal individual treatment. Budget in real-life applications is always limited, and it can represent any limited resource such as time, money, or side effects of medications. In this contribution, we develop a POMDP-based framework to learn cost-sensitive heterogeneous cascading systems. We provide both the theoretical support for the introduced approach and the intuition behind it.

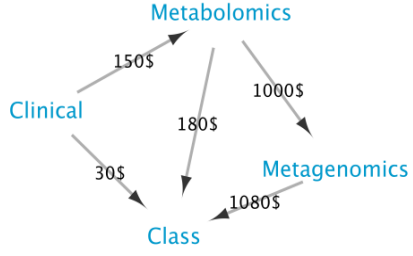


Figure 1: An example of an individualized diagnostic protocol: heterogeneous data sources involved in the health status prediction, and their corresponding costs.

Published in proceedings of AISTATS, 2019.

The State-of-the-art and Open Challenges

More or less achieved:

- methods for learning an (individualized) diagnostic protocol,
- based on theoretical support (MDP, POMDP),
- deep network able to learn a family of parametric models for missing data.

Still a Challenge:

- Missing data alternatives as generative adversarial network and variational autoencoder,
- numerical experiments for special cases (comparison between cascading system, multi-stage system and classifier with embedded feature selection),
- adaptation to genomic, transcriptomic and metabolomic data,
- features variation in time (POMDP)

Outline

Learning under Budget
- Motivation for Learning under Budget
- Cascade Classifiers
- A deep reinforcement learning model
- MDP and POMDP
- Some Applications in Medicine

Generative Adversarial Learning
- Generative Adversarial Networks
- Example: a Real Materials Science Application
- Real Data and Discussion

Generative Learning

- Discriminative learning tries to **classify** data (maps features to classes)
  - Find a boundary between classes
- Generative learning tries to provide features given a label
  - Model data distribution
- To be precise, **generative learning**: given a class, how likely are these or those features?

Generative Adversarial Networks

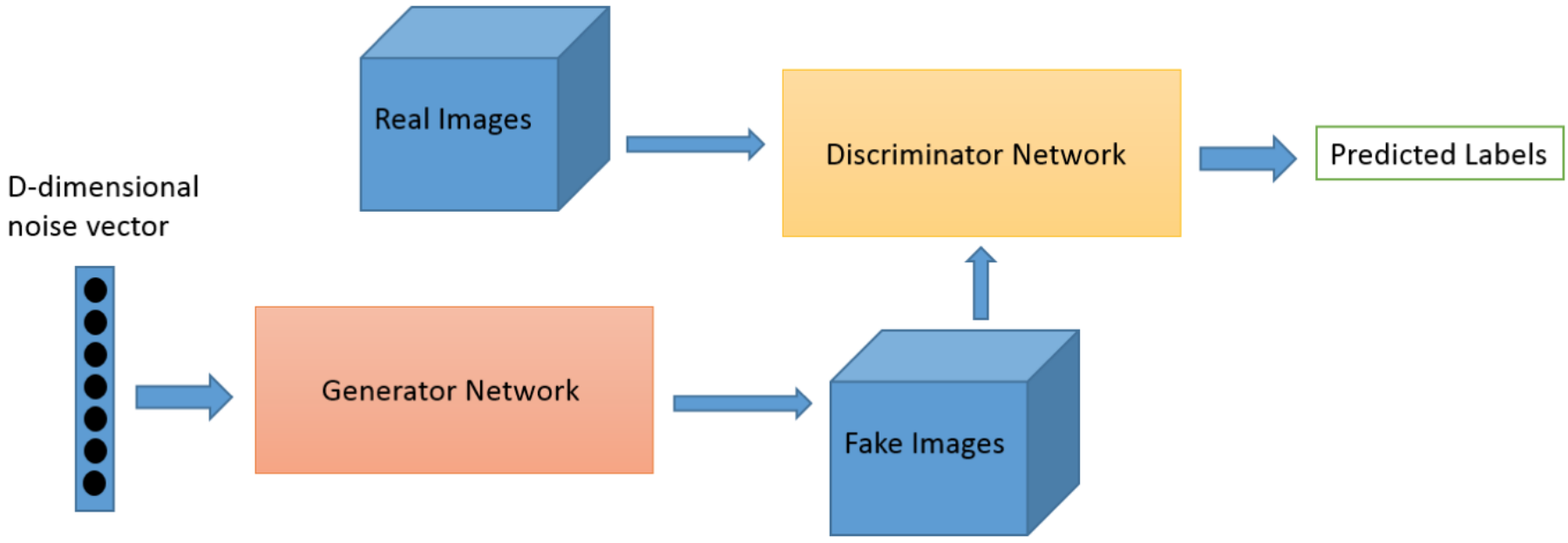
Two components:

1. Generator – generates new data
2. Discriminator – estimate the probability to belong to a class

Steps of the GANs:

1. The generator generates data
2. The generated and observed data are fed into the discriminator
3. The discriminator returns probabilities to belong to the classes (fake or true)

GANs



from https://skymind.ai/wiki/generative-adversarial-network-gan

- Feedback loop: discriminator and the true classes
- Feedback loop: generator – discriminator

GANs

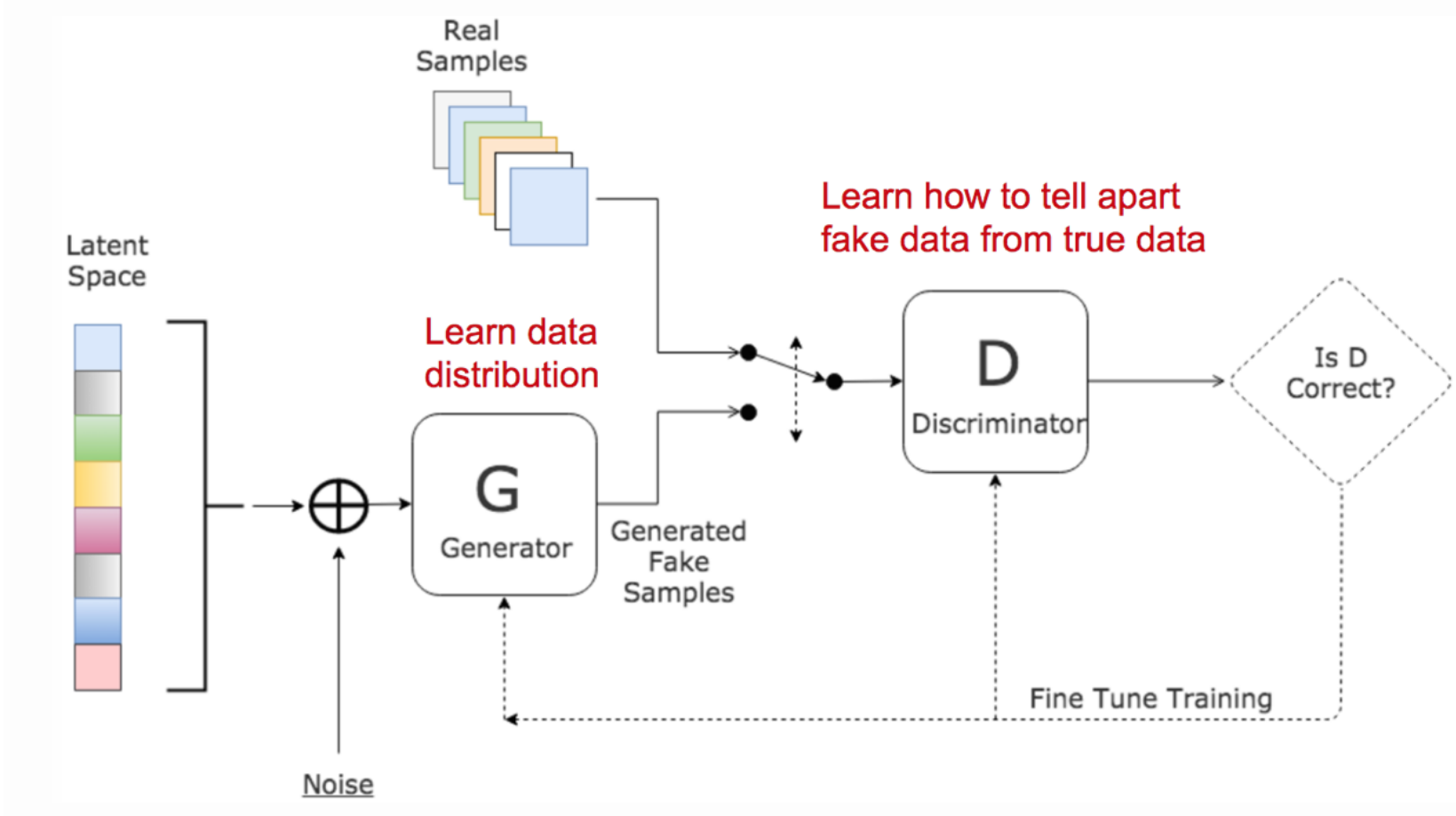
- p_z – Data distribution over noise (uniform)
- p_g – Generator's distribution over (observed) data
- p_r – True distribution

Play a **minimax game**:

$$\min_G \max_D L(D,G) = E_{x\sim p_r}\{\log D(x)\} + E_{z\sim p_z}\{\log(1-D(G(z)))\}$$
$$= E_{x\sim p_r}\{\log D(x)\} + E_{x\sim p_g}\{1-\log D(x)\}$$

- D is expected to maximize $E_{z\sim p_z}\{\log(1-D(G(z)))\}$
- G is trained to fool D, and to minimize $E_{z\sim p_z}\{\log(1-D(G(z)))\}$

GANs



from http://www.kdnuggets.com/2017/01/generative-adversarial-networks-hot-topic-machine-learning.html

GANs: the Learning Algorithm

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

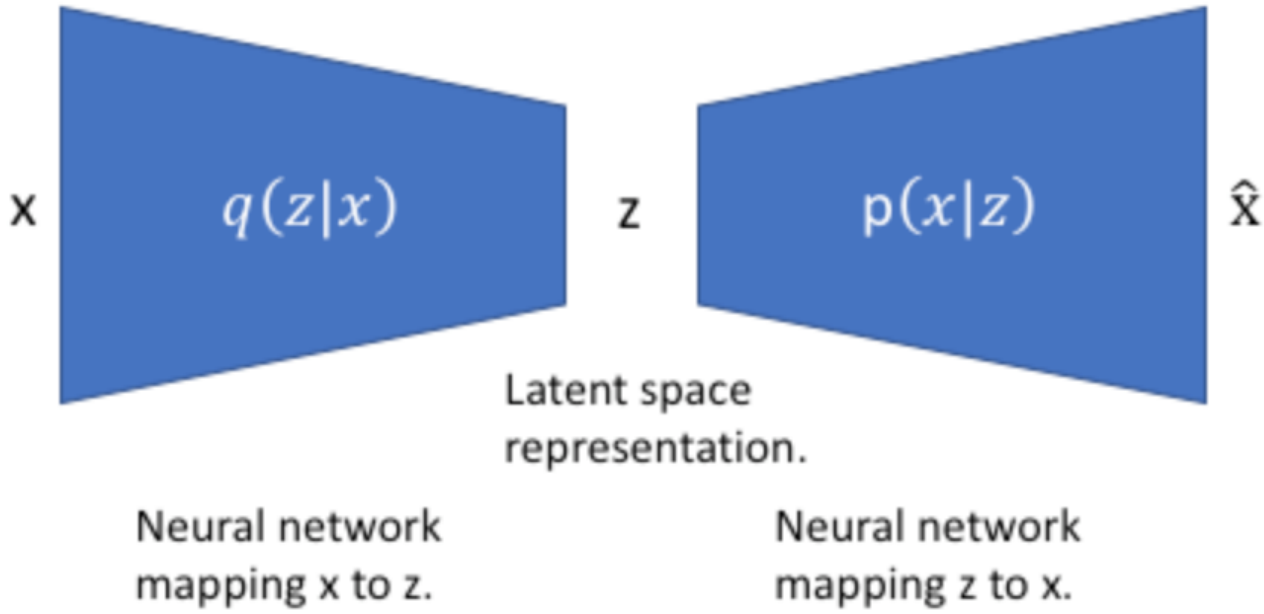
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

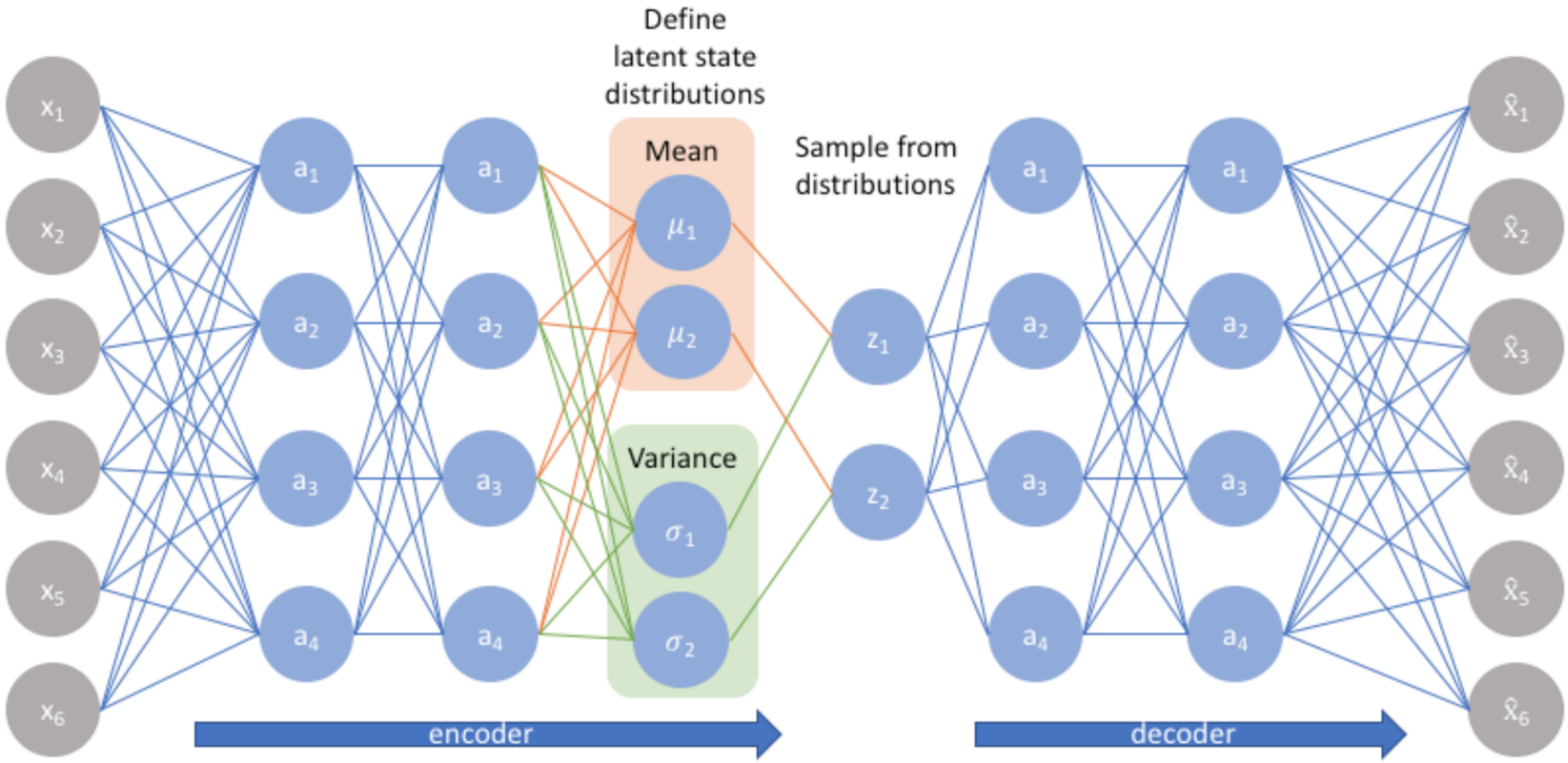
from *Ian J. Goodfellow et al., 2014l*

Autoencoders



from https://www.jeremyjordan.me/variational-autoencoders/

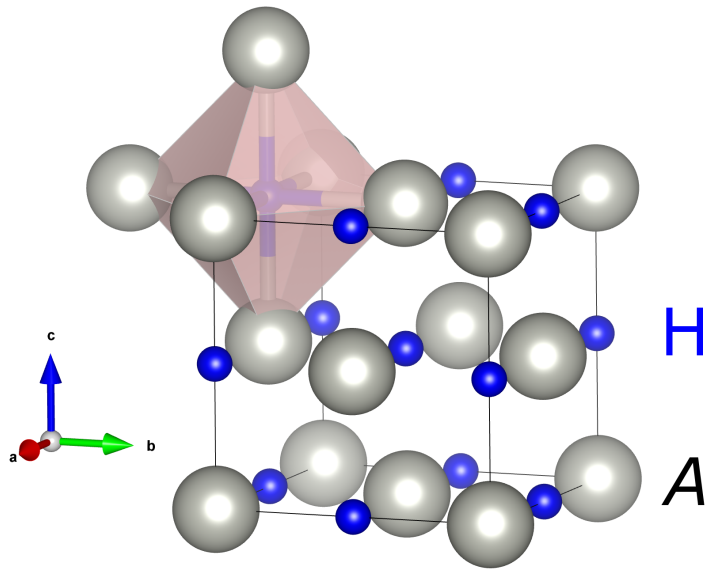
Variational Autoencoders



from https://www.jeremyjordan.me/variational-autoencoders/

Generative Adversarial Learning
Generative Adversarial Networks
Example: a Real Materials Science Application
Real Data and Discussion

Some Materials Science Examples



- Synthesis of new organic and inorganic compounds
- Combinatorial problem for human experts
- “Clean” energy: hydrogen storage

Problem Formulation

Generate: ternary hydride compounds of the form "A (a metal) - H (hydrogen) - B (a metal)".

- The training algorithm observes stable binary compounds containing chemical elements A+H which is a composition of some metal A and the hydrogen H, and B+H which is a mixture of another metal B with the hydrogen.
- A machine learning algorithm has access to observations $\{(x_{AH_i})\}_{i=1}^{N_{AH}}$ and $\{(y_{BH_i})\}_{i=1}^{N_{BH}}$. *Our goal is to generate novel ternary, i.e. more complex, stable data x_{AHB} (or y_{BHA}) based on the properties learned from the observed binary structures.*

GANs and Cross Domain GANs

- DiscoGAN and CycleGAN: cross-domain learning
  - Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. ICCV, 2017
  - Kim, T.; Cha, M.; Kim, H.; Lee, J. K.; and Kim, J. *Learning to Discover Cross-Domain Relations with Generative Adversarial Networks*, ICML, 2017.
- Discover relations between two different domains from unpaired samples, and to find a mapping from one domain to another
- How to generate data with augmented complexity?

GANs and Cross Domain GANs

We consider a function $\mathbf{G}_{ABZ}$ that maps elements from domains A and B to domain Z which includes the co-domains A and B. In an unsupervised learning scenario, $\mathbf{G}_{ABZ}$ can be arbitrarily defined, however, to apply it to real-world applications, some conditions on the relation of interest have to be well-defined.
In an idealistic setting, the equality

$$\mathbf{G}_{ABZ} \circ \mathbf{G}_{ZAB}(x_A, x_B) = (x_A, x_B) \tag{1}$$

is satisfied. However, this constraint is a hard constraint, it is not straightforward to optimize it, and a relaxed soft constraint is preferred. As a soft constraint, we can consider the distance

$$d\left(\mathbf{G}_{ABZ} \circ \mathbf{G}_{ZAB}(x_A, x_B), (x_A, x_B)\right), \tag{2}$$

and minimize it using a metric function such as L_1 or L_2.

$$-\mathbb{E}_{x_A, x_B \sim P_{A,B}} \left[\log \mathbf{D}_Z(\mathbf{G}_{ABZ})(x_A, x_B)\right]. \tag{3}$$

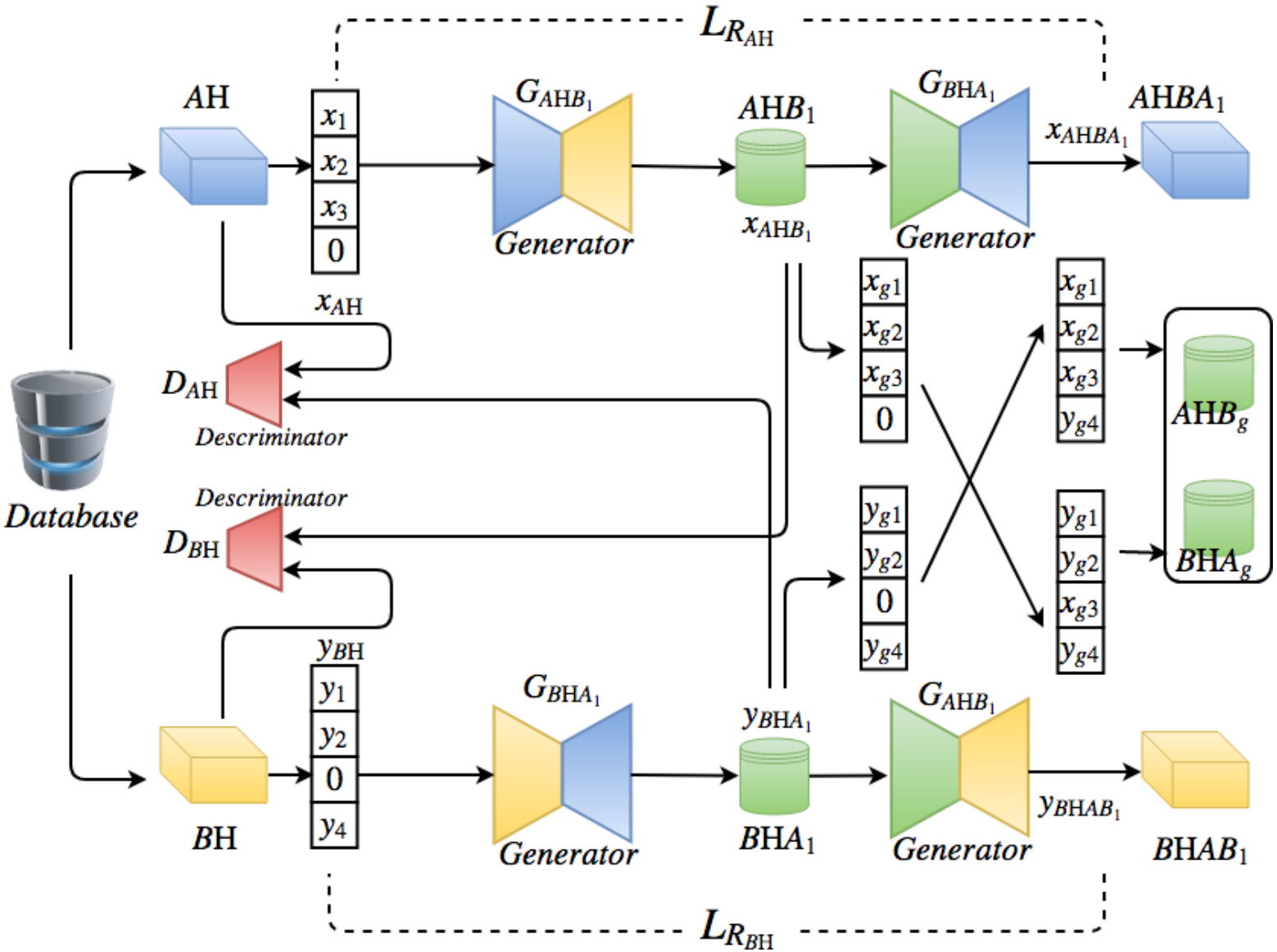
CrystalGAN

Our goal: Generate x_{AHB} (or y_{BHA}) from $\{(x_{AH_i})\}_{i=1}^{N_{AH}}$ and $\{(y_{BH_i})\}_{i=1}^{N_{BH}}$

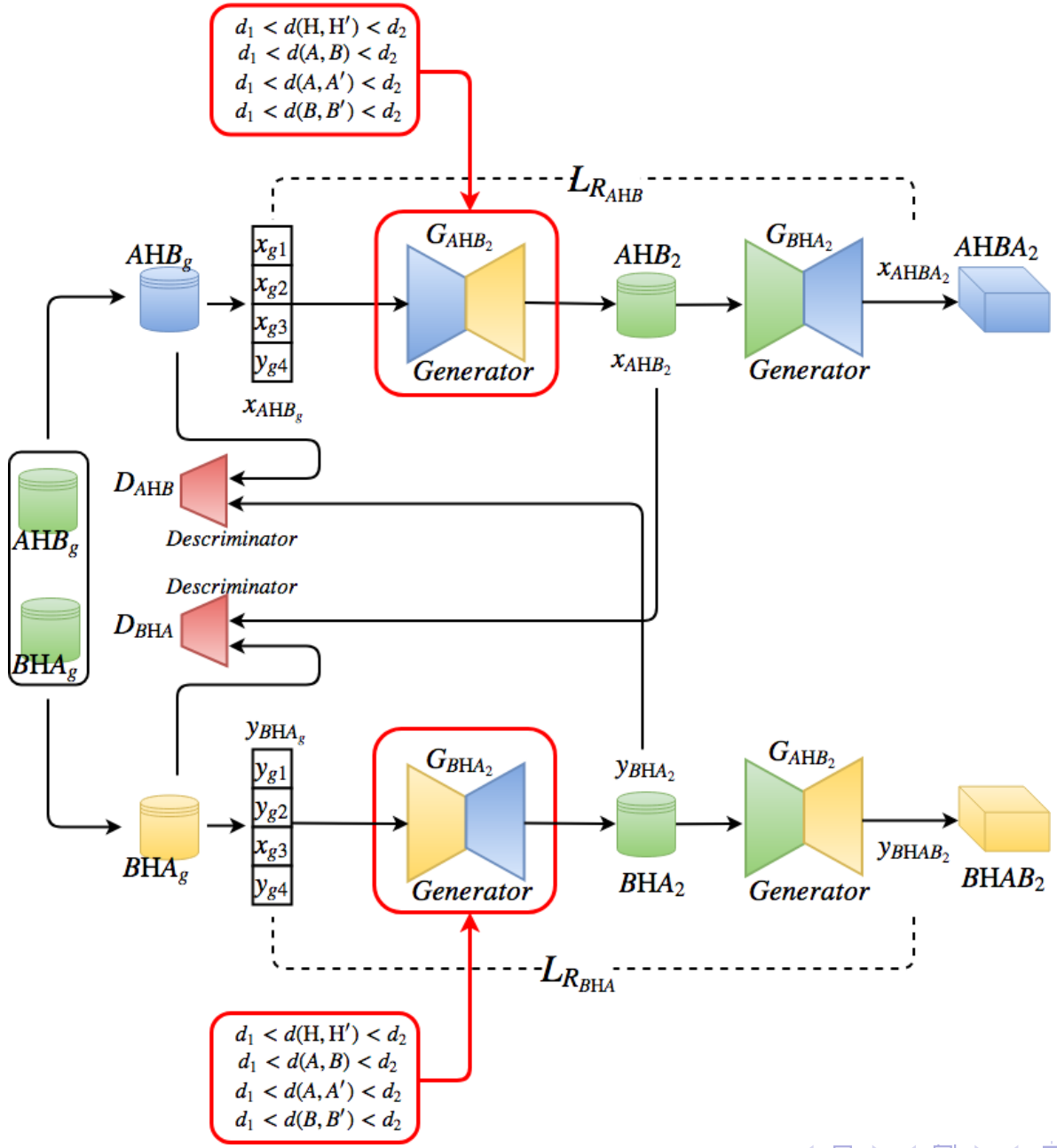
Three steps of **CrystalGAN:**

1. First step GAN which is closely related to the cross-domain GANs, and that generates pseudo-binary samples where the domains are mixed.
2. Feature transfer procedure constructs higher order complexity data from the samples generated at the previous step, and where components from all domains are well-separated.
3. Second step GAN synthesizes, under geometric constraints, novel ternary stable chemical structures.

CrystalGAN: First Step



CrystalGAN: Feature Transfer and Second Step



Outline

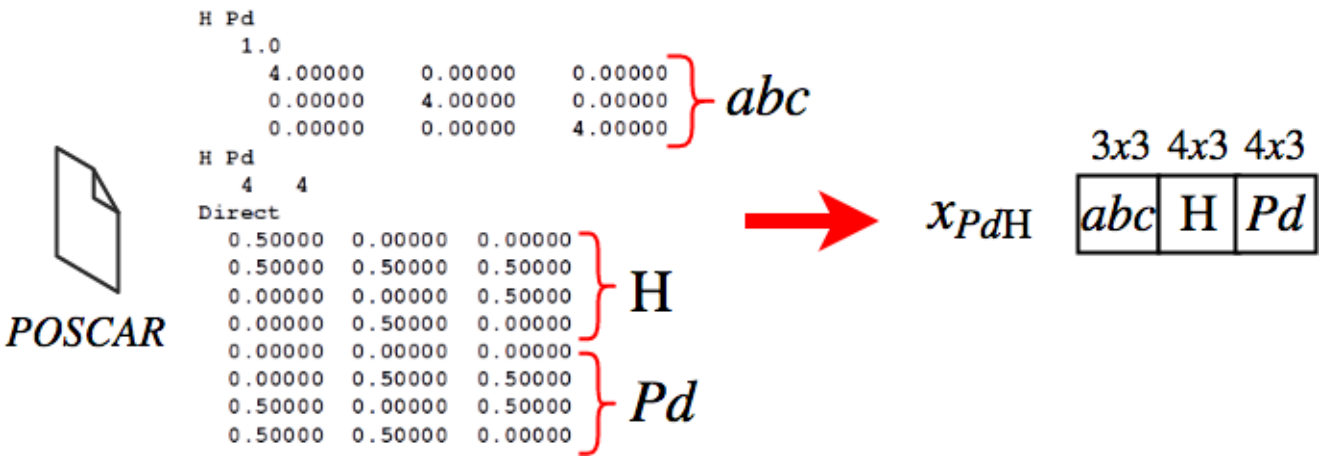
Generative Adversarial Learning
- Generative Adversarial Networks
- Example: a Real Materials Science Application
- Real Data and Discussion

Numerical Experiments: POSCAR files

A Material Science application:
generate stable data for hydrogen storage

Dataset: POSCAR files: 3 matrices:

1. *abc* matrix, corresponding to the three lattice vectors defining the unit cell of the system
2. atomic positions of H atom, and coordinates of metallic atom A (or B).



Results on Crystals Dataset

Number of ternary compositions of good quality (stable) generated by the tested methods

Composition	GAN (standard)	Disco GAN	Crystal GAN without constraints	Crystal GAN with geometric constraints
Pd - Ni - H[1]	0	0	4	9
Mg - Ti - H[2]	0	0	2	8

(learning from about 35 observations from each class)

[1] Palladium – Hydrogen – Nickel

[2] Magnesium - Hydrogen - Titanium

CrystalGAN: Learning to Discover Crystallographic Structures with Generative Adversarial Networks

Asma Nouira[1], **Nataliya Sokolovska**[2], **Jean-Claude Crivello**[1]
[1]University Paris Est, ICMPE (UMR 7182)
CNRS, UPEC, F-94320 Thiais, France
[2]Sorbonne University, INSERM, NutriOmics team, Paris France

Abstract

Our main motivation is to propose an efficient approach to generate novel multi-element stable chemical compounds that can be used in real world applications. This task can be formulated as a combinatorial problem, and it takes many hours of human experts to construct, and to evaluate new data.

the progress in materials science. Machine learning methods, namely, generative models, are reported to be efficient in new data generation (Friedman, Tibshirani, and Hastie, 2009), and nowadays we have access to both, techniques to generate a huge amount of new chemical compounds, and to test the properties of all these candidates.

Published in proceedings of AAAI-MAKE, 2019.

Many thanks

to Matthieu Clertant (learning under budget)
and Asma Nouira (GANs for materials science)
for sharing their slides and figures