Predicting tactical solutions to operational planning problems under imperfect information

E. Larsen, S. Lachapelle, Y. Bengio,
E. Frejinger, S. Lacoste-Julien, A. Lodi

ArXiv:1807.11876v3

45th Conference on the Mathematics of Operations Research

Lunteren, The Netherlands, January 14, 2020

In brief:

Combine machine learning and discrete optimization to solve a problem that we could not solve with any existing methodology.

Challenges:

Very restricted computing time budget.
Imperfect information.

CONTEXT

Planning horizon and increasing level of information

LEVEL OF DETAIL OF SOLUTION

	Long term « strategic »	**Medium term « tactical »**	**Short term « operational »**
			Fully detailed solution - implementable
		Description of solution - level of detail that is relevant to the tactical decision problem	
	Value of the solution		

CONTEXT

Planning horizon and increasing level of information

	Medium term « tactical »	**Short term « operational »**
	Compute description of solution to operational problem under imperfect information	Operational problem of interest: Compute solution under perfect information
seconds to minutes		*Reasonable computing time - within the time budget for the operational problem*
milli-seconds	*Much shorter than the time it takes to solve the full problem under perfect information*	

COMPUTING TIME BUDGET

CONTEXT

Planning horizon and increasing level of information

Medium term
« tactical »

Compute description of solution to operational problem under imperfect information

Short term
« operational »

Operational problem of interest: Compute solution under perfect information

High-precision solution
Reasonable computing time

Solve deterministic optimization problem
mathematical programming

High-level solution
Very short computing time

Stochastic programming

Machine learning
predict the tactical solution descriptions

SOME NOTATION

Planning horizon and increasing level of information

	Medium term « tactical »		Short term « operational »	
Problem instance	Imperfect information	$\mathbf{x}_a$	Perfect information	$\mathbf{x} = [\mathbf{x}_a, \mathbf{x}_u]$
Solution		$\widehat{\mathbf{y}}^*(\mathbf{x}_\mathrm{a})$	Deterministic problem	$\mathbf{y}^*(\mathbf{x}) = \arg \min_{\mathbf{y} \in Y(\mathbf{x})} C(\mathbf{x}, \mathbf{y})$

Tactical solution description $\bar{\mathbf{y}}^* = g(\mathbf{y}^*(\mathbf{x}))$

APPLICATION - LOAD PLANNING

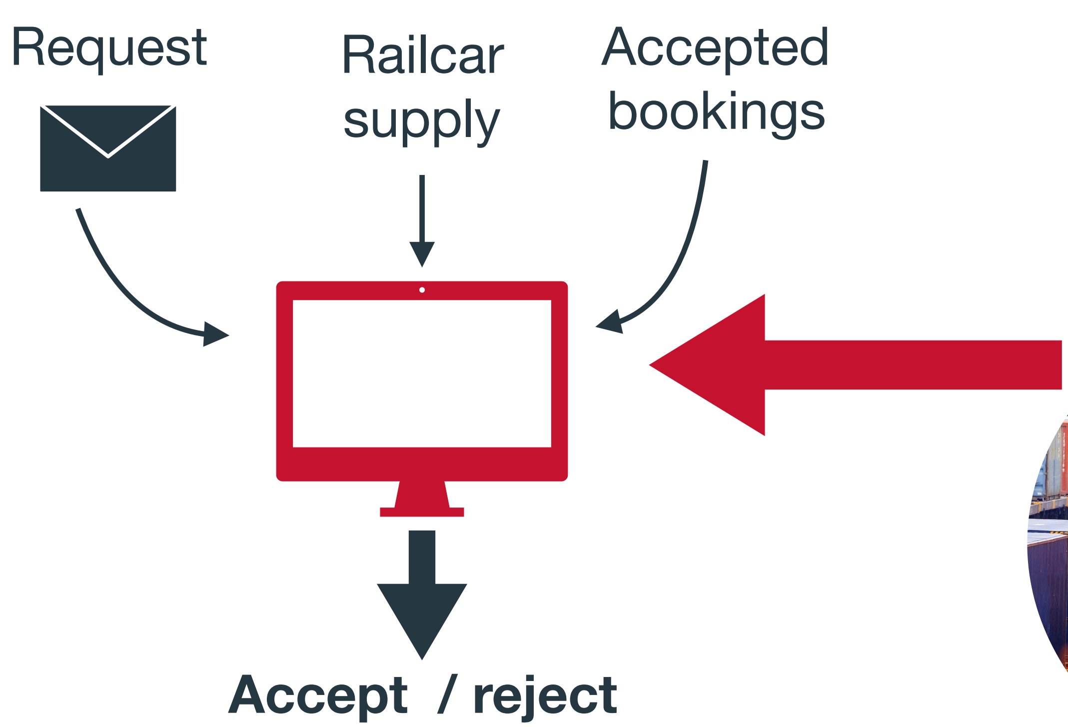
Planning horizon and increasing level of information

Medium term
« tactical »

Capacity management, e.g., bookings

Short term
« operational »

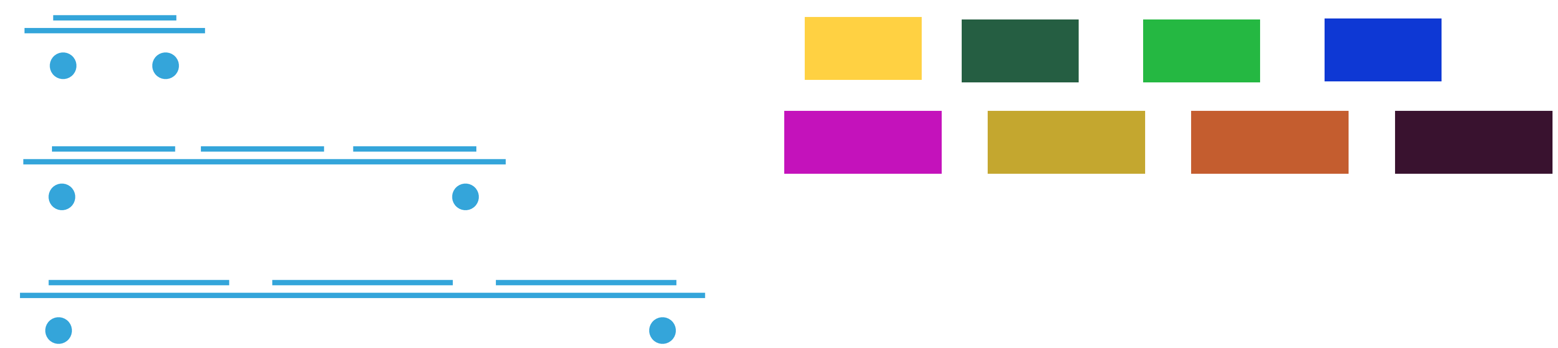
Load planning for double-stack trains



APPLICATION - LOAD PLANNING

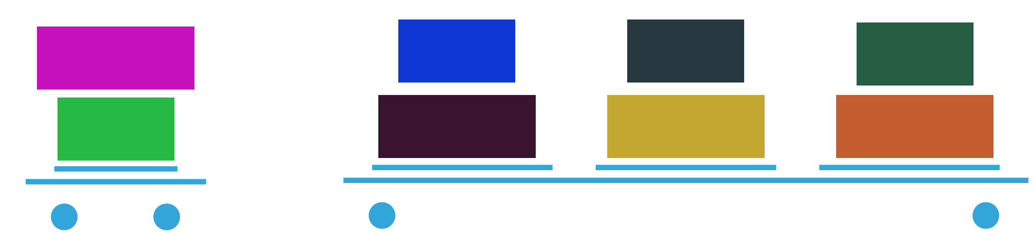
Problem instance

$$\mathbf{x} = [\mathbf{x}_a, \mathbf{x}_u]$$



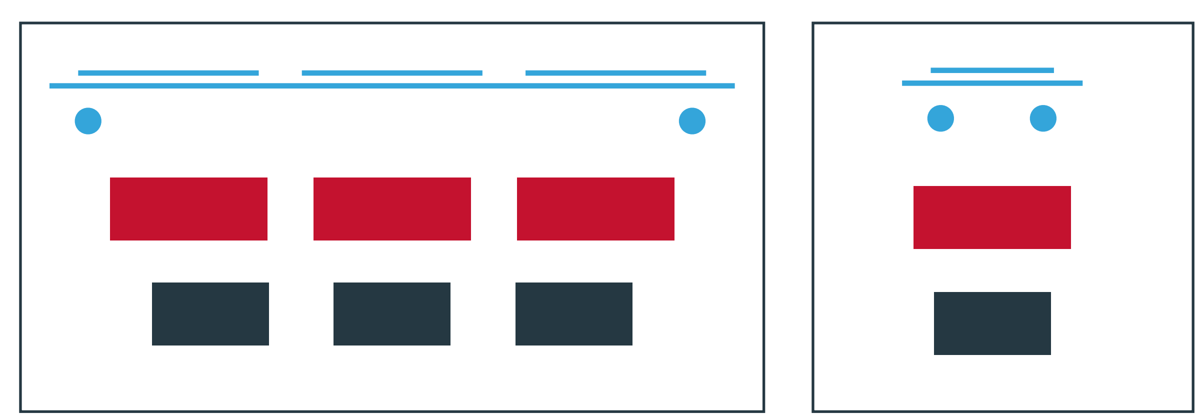
Operational solution

$$\mathbf{y}^*(\mathbf{x}) = \arg \min_{\mathbf{y} \in Y(\mathbf{x})} C(\mathbf{x}, \mathbf{y})$$



Tactical solution

$$\bar{\mathbf{y}}^* = g(\mathbf{y}^*(\mathbf{x}))$$



APPLICATION - LOAD PLANNING

- Containers have different characteristics, for example:
  - Size
  - Weight
- The loading (operational problem) of the containers onto railcars crucially depends on weight
- Weight is unknown at the tactical level

IDEA IN BRIEF

- We know how to solve the deterministic problem - let's use that!
  - Generate a lot of data and pretend that we have perfect information - solve the discrete optimization problem with an existing solver
- Let machine learning take care of the uncertain part: hide the information that is not available at prediction time - find best possible prediction of $\bar{\mathbf{y}}^*$

$$\widehat{\mathbf{y}}^*(\mathbf{x}_\mathrm{a}) \equiv f(\mathbf{x}_\mathrm{a}; \boldsymbol{\theta})$$

State-of-the-art ML model

Parameters

METHODOLOGY

Problem

Two-stage stochastic programming formulation

Optimal prediction conditional on $\mathbf{x}_a$, expectation over distribution of $\mathbf{x}_u$

Optimal solution to deterministic problem for given $\mathbf{x} = [\mathbf{x}_a, \mathbf{x}_u]$

Data

Problem instances and solutions (perfect information)

Machine learning training, validation, test data

Training & performance

Train and validate model

Assess predictive performance, e.g.

METHODOLOGY

Problem

$$\bar{\mathbf{y}}^*(\mathbf{x}_\mathrm{a}) :\equiv \arg \inf_{\bar{\mathbf{y}}(\mathbf{x}_\mathrm{a}) \in \bar{\mathcal{Y}}(\mathbf{x}_\mathrm{a})} \Phi_{\mathbf{x}_\mathrm{u}} \{ \|\bar{\mathbf{y}}(\mathbf{x}_\mathrm{a}) - g(\mathbf{y}^*(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u}))\| \mid \mathbf{x}_\mathrm{a} \}$$

$$\mathbf{y}^*(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u}) :\equiv \arg \inf_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u})} C(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u}, \mathbf{y})$$

Data

Problem instances and solutions (perfect information)

Machine learning training, validation, test data

Training & performance

Train and validate model

Assess predictive performance, e.g.

METHODOLOGY

Problem

$$\bar{\mathbf{y}}^*(\mathbf{x}_\mathrm{a}) :\equiv \arg \inf_{\bar{\mathbf{y}}(\mathbf{x}_\mathrm{a}) \in \bar{\mathcal{Y}}(\mathbf{x}_\mathrm{a})} \Phi_{\mathbf{x}_\mathrm{u}} \{ \boxed{\| \bar{\mathbf{y}}(\mathbf{x}_\mathrm{a}) - g(\mathbf{y}^*(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u})) \|} \mid \mathbf{x}_\mathrm{a} \}$$

$$\boxed{\mathbf{y}^*(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u})} :\equiv \arg \inf_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u})} C(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u}, \mathbf{y})$$

Data

Problem instances and solutions (perfect information)

$$(\ \mathbf{x}^{(i)}\ ,\ \mathbf{y}^{*(i)}\ )\quad i = 1, \ldots, m$$

Machine learning training, validation, test data

$$(\ \mathbf{x}_a^{(i)}\ ,\ \bar{\mathbf{y}}^{*(i)}\ )\quad i = 1, \ldots, m$$

Training & performance

Train and validate model

Assess predictive performance, e.g.

METHODOLOGY

Problem

$$\bar{\mathbf{y}}^*(\mathbf{x}_\mathrm{a}) :\equiv \arg \inf_{\bar{\mathbf{y}}(\mathbf{x}_\mathrm{a}) \in \bar{\mathcal{Y}}(\mathbf{x}_\mathrm{a})} \Phi_{\mathbf{x}_\mathrm{u}} \{ \boxed{\|\bar{\mathbf{y}}(\mathbf{x}_\mathrm{a}) - g(\mathbf{y}^*(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u}))\|} \mid \mathbf{x}_\mathrm{a} \}$$

$$\boxed{\mathbf{y}^*(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u})} :\equiv \arg \inf_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u})} C(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{u}, \mathbf{y})$$

Data

Problem instances and solutions (perfect information)

$$(\mathbf{x}^{(i)}, \mathbf{y}^{*(i)}) \quad i = 1, \ldots, m$$

Machine learning training, validation, test data

$$(\tilde{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}^{*(i)}) \quad i = 1, \ldots, m$$

Training & performance

Train and validate model

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{m'} \sum_{i=1}^{m'} L\left(f(\mathbf{x}_a; \boldsymbol{\theta}), \bar{\mathbf{y}}^{*(i)}\right)$$

Assess predictive performance, e.g.

$$\mathbf{MAE_{test}} = \frac{1}{n} \sum_{i=1}^{n} \left| f(\mathbf{x}_a^{(i)}; \hat{\boldsymbol{\theta}}) - \bar{\mathbf{y}}^{*(i)} \right|$$

METHODOLOGY

- **Data**
  - Historically observed instances and their solutions
    - Purpose: « mimic » behaviour in such data
  - Our approach: generate data by sampling problem instances and computing the corresponding solutions using existing optimization model and solver
    - Purpose: generalization over the domain of $\mathbf{x}$
- The **input structure** is governed by the information available at prediction time
- The **output structure** is governed by the choice of solution description and can be of fixed or variable size
- **Model architecture** depends on input and output structures and on constraints linking the two

RELATED LITERATURE

- Closest to our work are those based on supervised learning but they focus on deterministic problems
  - Fischetti and Fraccaro (2017) predict optimal objective function value
  - Vinyals et al. (2015) define pointer networks to solve a class of discrete optimization problems, constraints are imposed by changing the NMT model architecture
- Nair et al. (2017) propose a reinforcement learning algorithm combined with ILP solver for a two-stage binary stochastic program (unconstrained binary decisions)

DATA GENERATION

- Random sampling of container/railcar types and container weights

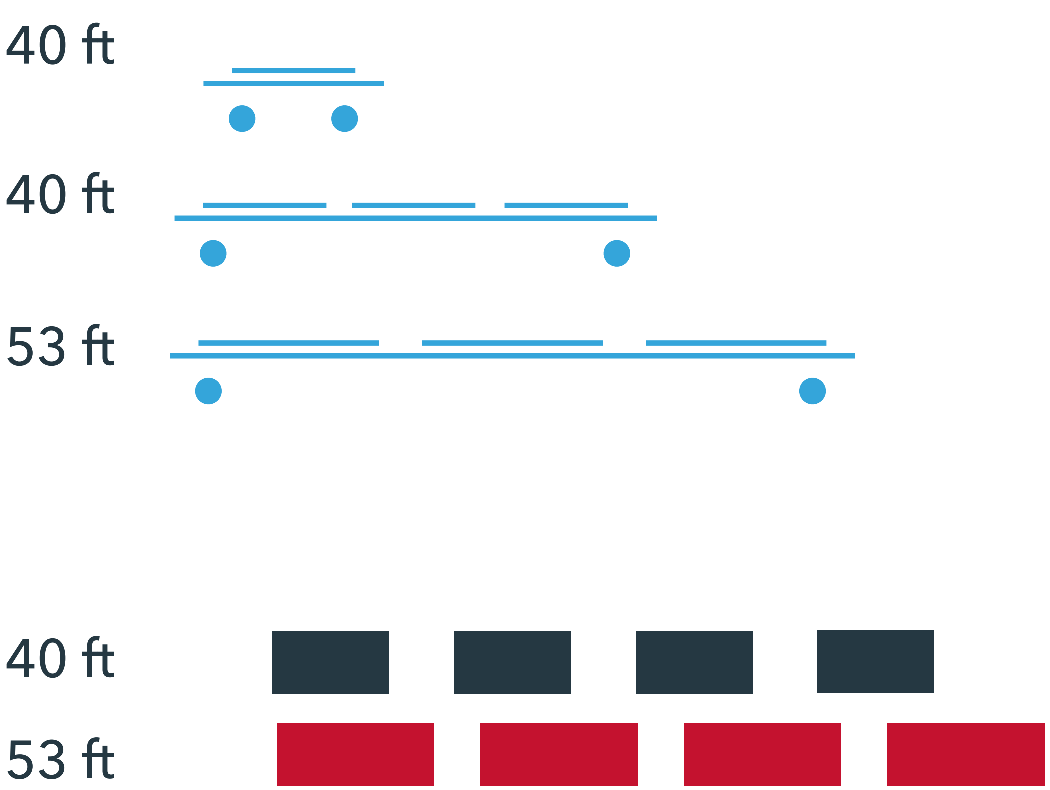
Class	Description	# of containers	# of platforms
A	Simple ILP	[1,150]	[1,50]
B	More containers than A (excess demand)	[151,300]	[1,50]
C	More platforms than A (excess supply)	[1,150]	[51,100]
D	Larger and harder	[151,300]	[51,100]

INPUT-OUTPUT

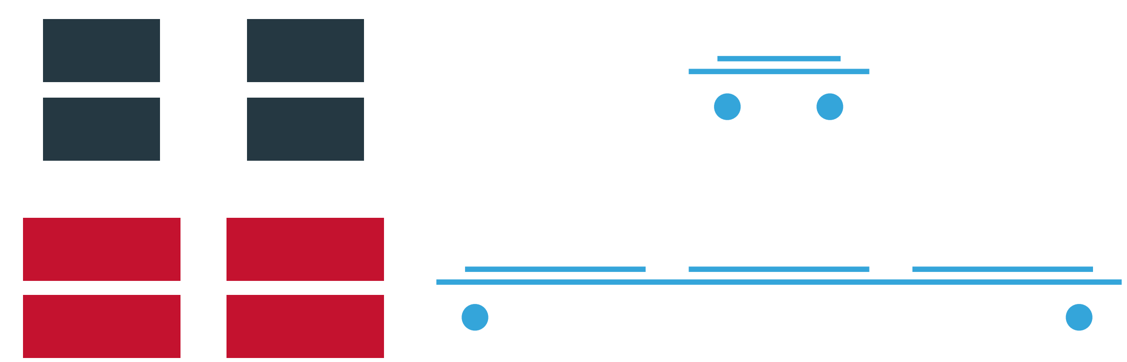
Input: problem instance

2 container types: 40 and 53 ft

10 railcar types: 10 most numerous in the North American fleet



Output: tactical solution



MULTILAYER PERCEPTRON / FEED-FORWARD NETWORK

TACTICAL: MULTILAYER PERCEPTRON

Input

Fixed-size vector

$\mathbf{x}_a$

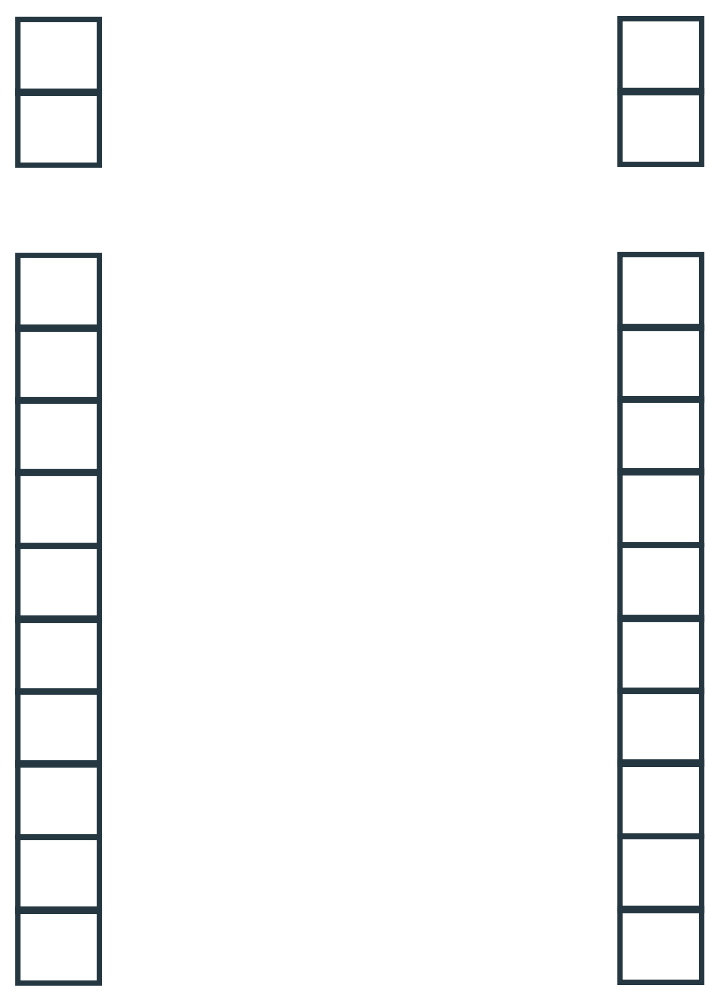
Output

Fixed-size vector

$\bar{\mathbf{y}}^*$

Nb of available containers of each type

Nb of of available railcars of each type



Nb of containers of each type in the solution

Nb of railcars of each type in the solution

TACTICAL: MULTILAYER PERCEPTRON

- Multilayer perceptron (MLP): approximately 7 layers and 500 rectified linear units (ReLU) per layer (hyper parameters)
- Classification / Regression (linear units in output layer and rounding to the nearest integer)
- **Training** and **validation**
  - Minimization of neg. likelihood function / sum of absolute errors
  - Mini-batch stochastic gradient descent and learning rate adaptation by the adaptive moment estimation (Adam) method
  - Regularization: early stopping
  - Random search for hyper parameter selection
- Mean Absolute Error (MAE) over slots and containers

TACTICAL: MULTILAYER PERCEPTRON

- Average performance of the MLP model is very good
  - **MAE** of only **2.1 containers/slots** for classes A, B and C (up to 100 platforms and 300 containers) with very small standard deviation (0.01)
- MLP results are considerably **better than benchmarks**
- The marginal value of using 100 times more observations is fairly small: modest increase in MAE from 0.985 to 1.304 on class A instances)
- **Prediction times are negligible**, milliseconds or less and with very little variation

TACTICAL: MULTILAYER PERCEPTRON

- The models trained and validated on simpler instances (A, B and C) generalize well to harder instances (D)
  - MAE of 2.85 (training on class A)
  - MAE of 0.32 (training on classes A, B and C)
  - Still, significant variability across models with different hyper parameters when only trained on class A (MAE varies between 0.74 and 9.05)
- Numerical analysis of **feasibility**: there exists a feasible operational solution for a given predicted tactical solution in 96.6% of the instances (the share is much lower for the benchmarks)

TACTICAL: MULTILAYER PERCEPTRON

What if we solve a sample average approximation (SAA) of the two stage stochastic program?

- Class A instances
- The average absolute error of the SAA solution is similar to that of the ML algorithm: 0.82 compared to 0.985
- The computing times for SAA vary between 1 second to 4 minutes with an average of 1 minute

Conclusion and perspectives

Novel combinations of **machine learning** and **operations research** methodologies have potential to solve hard decision-making problems under imperfect information.

We presented such a **methodology** that allows to predict solutions to a decision-making problem in very **short computing time**.

A lot of **research** left to be done and numerous **applications** to explore.

Thank you!

andrea.lodi@polymtl.ca