Graph-based sparse neural networks for traffic signal optimization

Łukasz Skowronek, Paweł Gora, Marcin Możejko, Arkadiusz Klemenko

Faculty of Mathematics, Informatics and Mechanics
University of Warsaw
&
TensorCell

p.gora@mimuw.edu.pl
http://www.mimuw.edu.pl/~pawelg

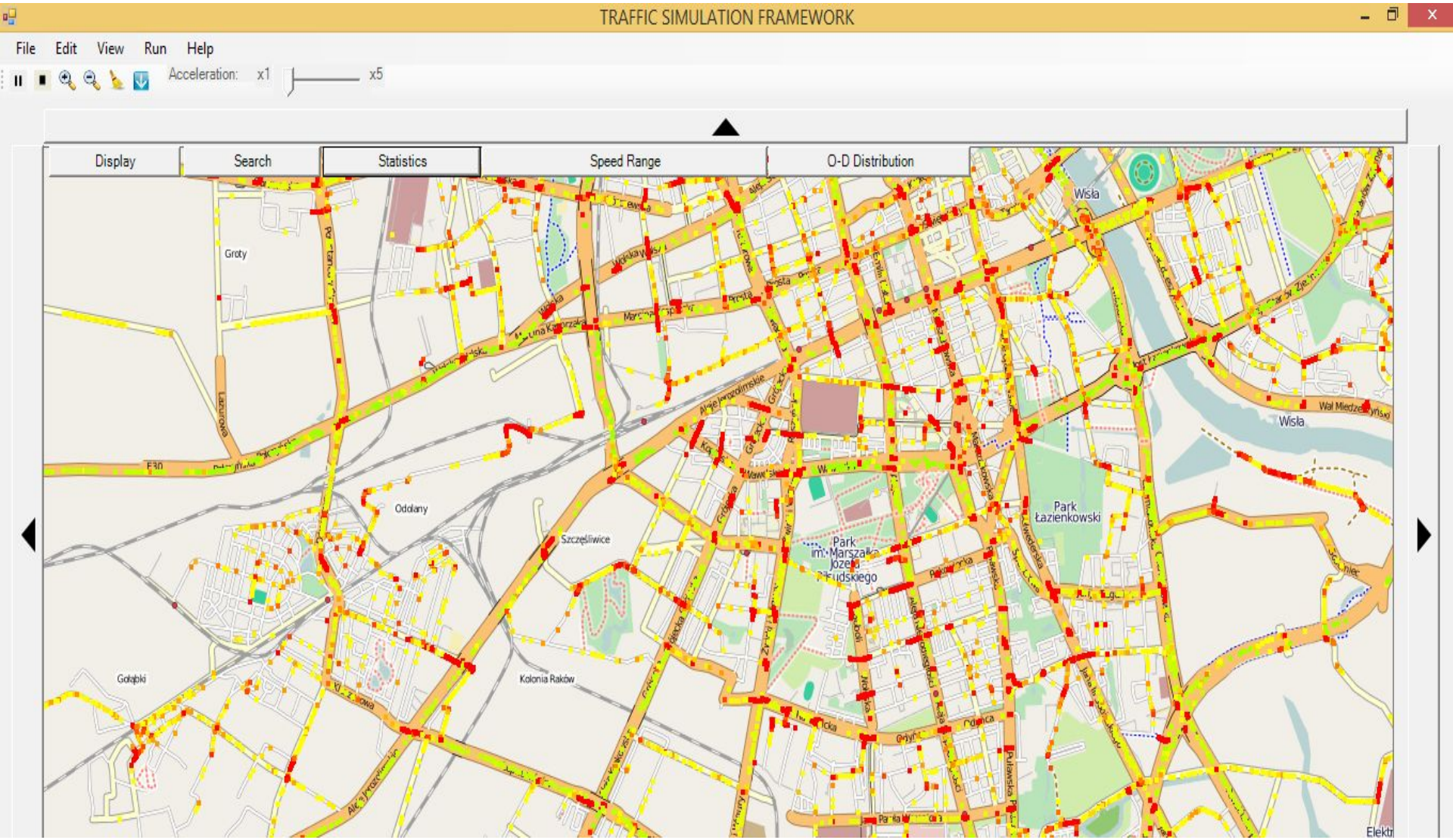
28.09.2021

Problem

- ➔ ~4 bln PLN - annual cost of traffic gridlocks in 7 largest Polish cities (cost of wasted time and fuel)[1)]
- ➔ Air Pollution Cuts Two Years Off The Average World Life Expectancy[2)]
- ➔ Over 1.2 mln people worldwide die each year in car accidents, 20-50 mln are injured[3)]
- ➔ Car accidents cost the U.S. $230.6 billion every year[4)]

1) https://www.ibtta.org/sites/default/files/documents/MAF/Costs-of-Congestion-INRIX-Cebr-Report%20(3).pdf
2) https://futurism.com/air-pollution-two-years-off-average-life-expectancy
3) http://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries
4) https://www.supermoney.com/2018/01/average-cost-car-accident-pay

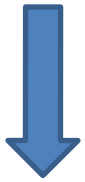
Traffic Simulation Framework



Gora P., "Traffic Simulation Framework - a Cellular Automaton based tool for simulating and investigating real city traffic", in "Recent Advances in Intelligent Information Systems", 2009, pp. 641-653, ISBN: 978-83-60434-59-8. (Screencast: https://www.youtube.com/watch?v=94RatF5SrLw)

Prediction “What-if”

What will happen if we change something in the road network?
What will happen if we change traffic signal settings?



Step toward traffic optimization

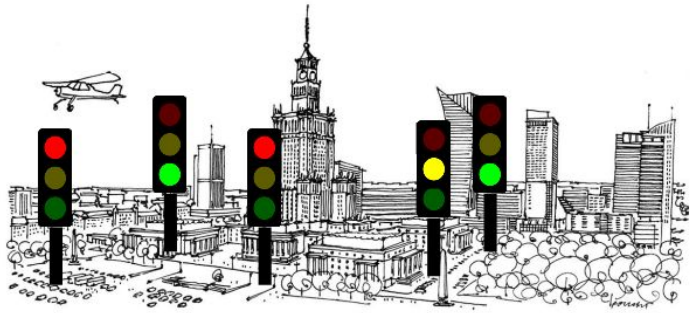
Prediction “What-if”

➔ The most popular and important tools to analyze traffic: **traffic simulations**.

➔ But accurate traffic simulations are time-consuming, especially in a large scale.

➔ In some cases we have to run large number of such simulations, with different settings (e.g., real-time traffic signal control, tolling, finding optimal locations and capacities of parkings / charging stations for electric vehicles).

Nr of games: 10^{700}



Nr of settings: > 120^{800}

(and the problem is proved to be NP-hard!)

More than the number of atoms in the visible Universe!

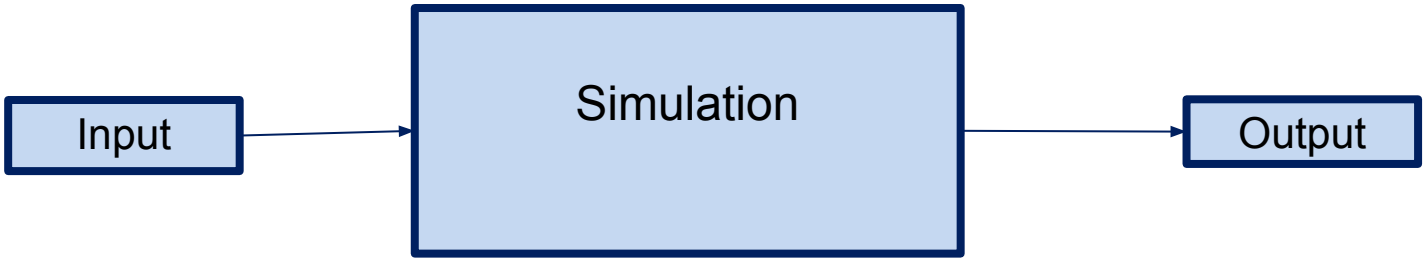
Prediction “What-if”

- ➔ The most popular and important tools to analyze traffic: **traffic simulations**.
- ➔ But accurate traffic simulations are time-consuming, especially in a large scale.
- ➔ In some cases we have to run large number of such simulations, with different settings (e.g., real-time traffic signal control, tolling, finding optimal locations and capacities of parkings / charging stations for electric vehicles).
- ➔ We would like to do it as efficiently as possible.
- ➔ We can distribute computations on GPU or many machines, but it may be expensive and has limitations.

Prediction “What-if”

- ➔ The most popular and important tools to analyze traffic: **traffic simulations**.
- ➔ But accurate traffic simulations are time-consuming, especially in a large scale.
- ➔ In some cases we have to run large number of such simulations, with different settings (e.g., real-time traffic signal control, tolling, finding optimal locations and capacities of parkings / charging stations for electric vehicles).
- ➔ We would like to do it as efficiently as possible.
- ➔ We can distribute computations on GPU or many machines, but it may be expensive and has limitations.
- ➔ In many cases we are not interested in the simulation process, but only in its outcomes. So, maybe we can somehow approximate the outcome based on partial output data?

Problem



F: X -> Y

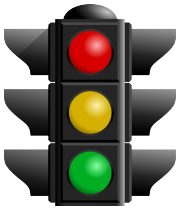
Example: X – traffic signal setting, Y – total delay, total time of waiting etc (real number)
(Y can be also a random variable in case of stochastic models)

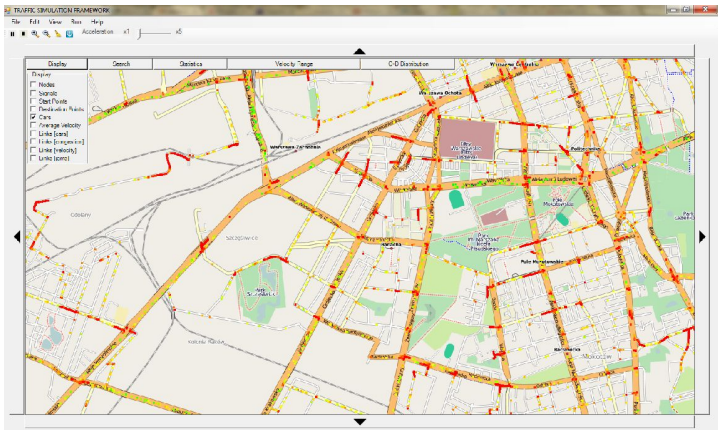
Can we „compute” **F** faster / easier than by running traffic simulations?
Can we find a „metamodel” (surrogate model) approximating outcomes of simulations?

Solution

Using Traffic Simulation Framework we generated set composed of 105336 elements, divided it into training set (85336 elements) and test set (20000 elements). Each run simulated 10 minutes of traffic with 42 000 cars on a realistic map of Warsaw (OSM).

21 x



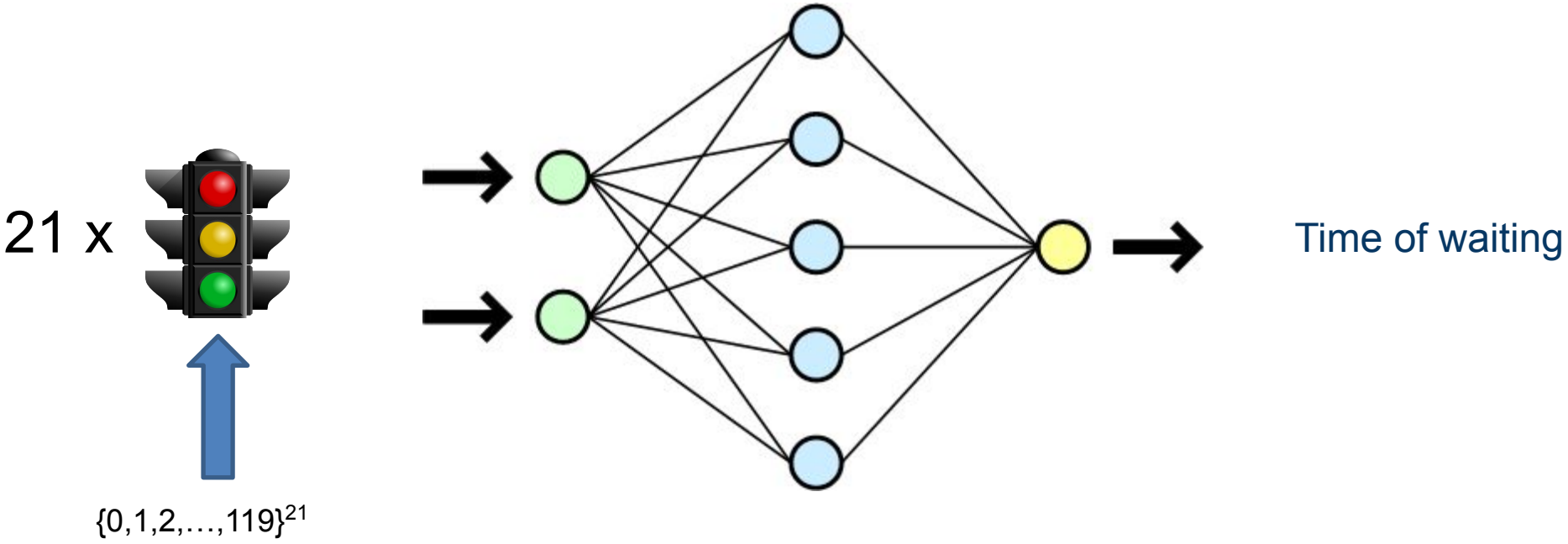


Time of waiting

$\{0,1,2,\ldots,119\}^{21}$

Solution

We focused on approximating the total waiting times on a red signal as a function of signal offset settings (signal offset setting = offsets of 21 traffic signal representatives on a Stara Ochota district in Warsaw).

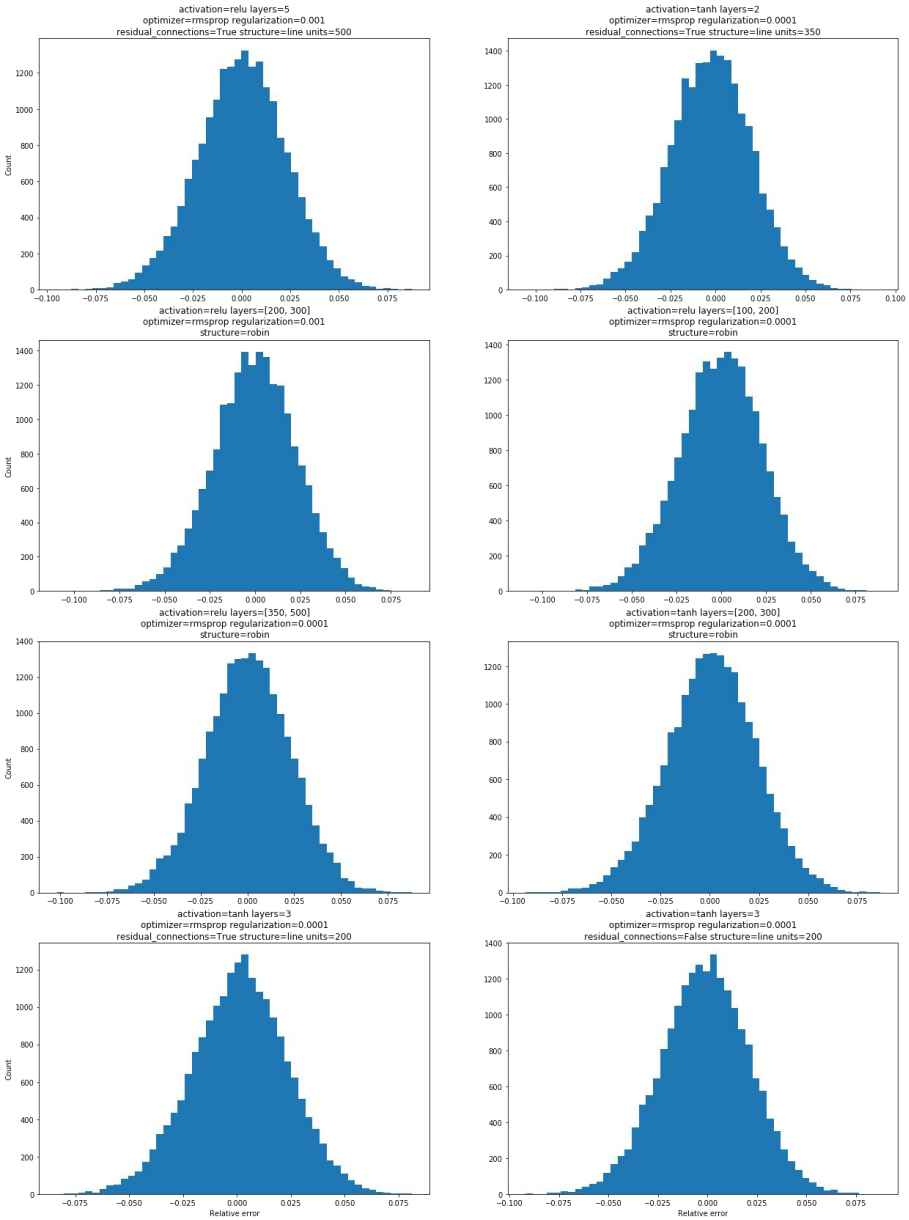


Solution

- We developed a **TensorTraffic** tool for approximating outcomes of traffic simulations using NN and predicting what may happen if we change traffic signal settings.

- Initially, we tested only feed forward neural networks. We found out, that, indeed, **outcomes of traffic simulation can be approximated using NN** with a good accuracy (best mean error on a test set: ~1.62%, maximal error: ~10.95%).

Distribution of error on a test set



Solution

TensorFlow

- We developed a **TensorTraffic** tool for approximating outcomes of traffic simulations using NN and predicting what may happen if we change traffic signal settings.

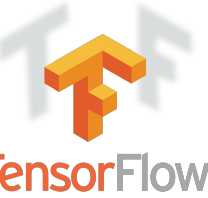
- Initially, we tested only feed forward neural networks. We found out, that, indeed, **outcomes of traffic simulation can be approximated using NN** with a good accuracy (best mean error on a test set: ~1.62%, maximal error: ~10.95%).

- Time of simulating 10 minutes of traffic in a large-scale (e.g., Warsaw) using a microscopic model (TSF) – **(~30 seconds on standard machines)**.

- Time of inferencing neural network: **~0.8 ms** (time of training with GPU: ~10-15 minutes).

- And in the investigated case we don't even need large networks (3-4 layers with a few hundred neurons are sufficient).

Solution



- We developed a **TensorTraffic** tool for approximating outcomes of traffic simulations using NN and predicting what may happen if we change traffic signal settings.
- Initially, we tested only feed forward neural networks. We found out, that, indeed, **outcomes of traffic simulation can be approximated using NN** with a good accuracy (best mean error on a test set: ~1.62%, maximal error: ~10.95%).
- Time of simulating 10 minutes of traffic in a large-scale (e.g., Warsaw) using a microscopic model (TSF) – **(~30 seconds on standard machines)**.
- Time of inferencing neural network: **~0.8 ms** (time of training with GPU: ~10-15 minutes).
- And in the investigated case we don't even need large networks (3-4 layers with a few hundred neurons are sufficient).
- **We also achieved good results using LightGBM** (best avg error: ~1.72%, max error: ~10.83%, inference: ~0.4 ms).

Application

We used both models as surrogate models (metamodels) evaluating traffic signal settings in traffic optimization algorithms.

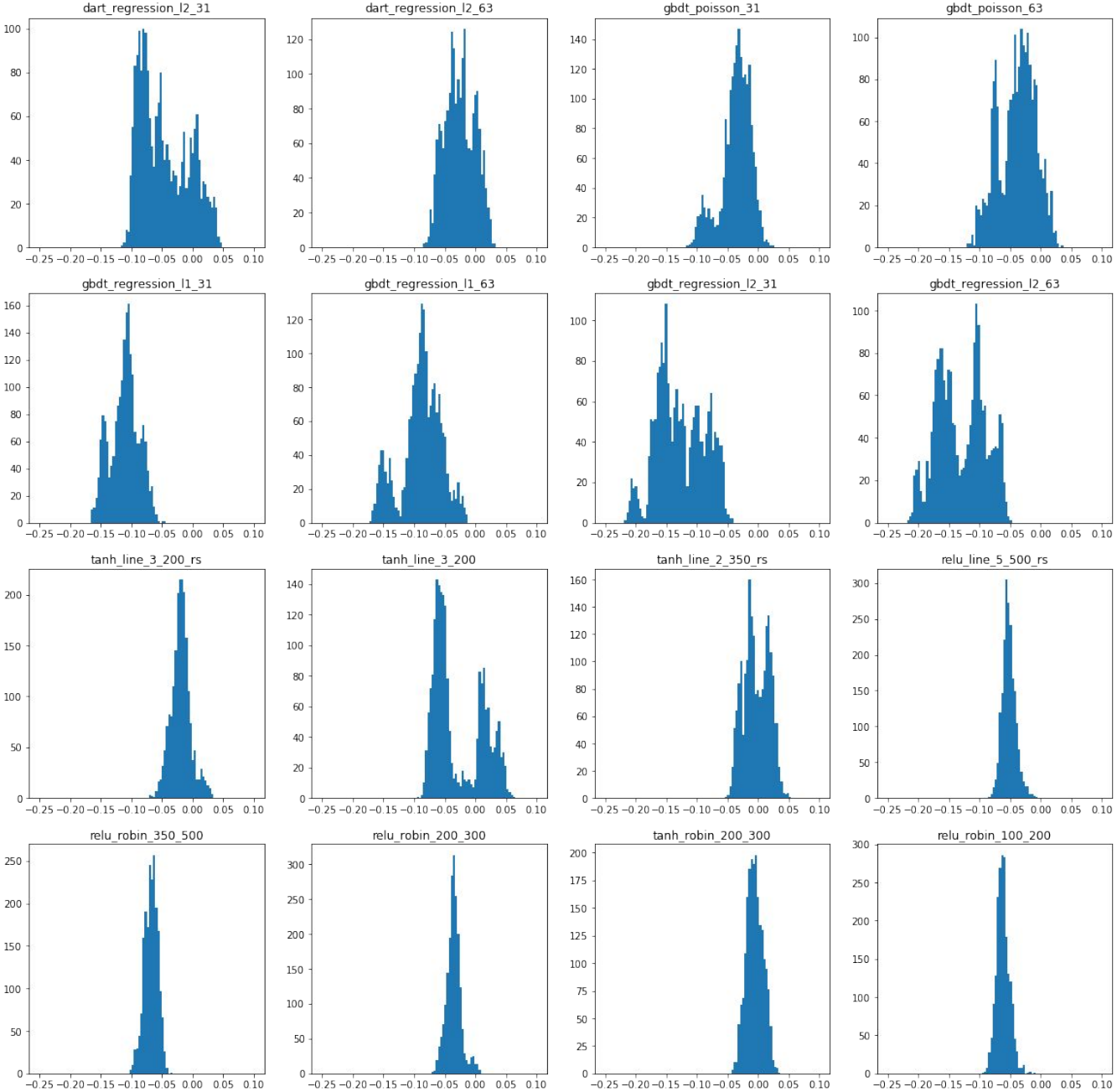
6 algorithms tested:

- Genetic algorithms
- Simulated annealing
- Particle swarm optimization
- Tabu search
- Bayesian optimization (without metamodels)
- Gradient optimization (work in progress)

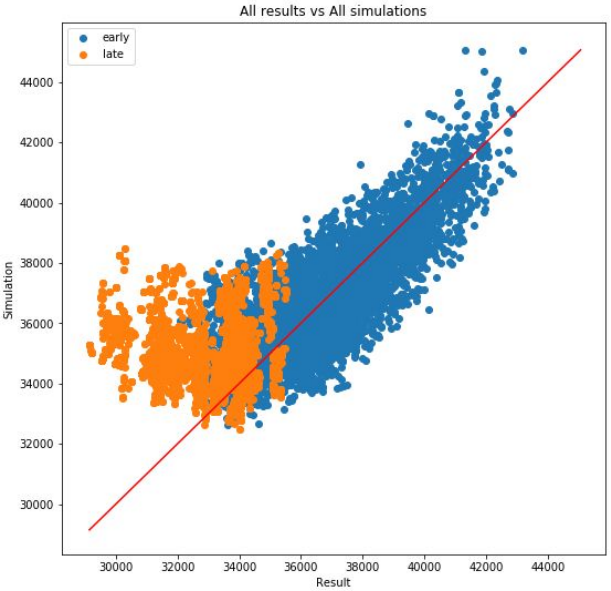
Algorithm	Best result [LightGBM]	Simulation for best result [LightGBM]	Best result [ANN]	Simulation for best result [ANN]	Best result according to simulation
Genetic algorithm	25318	37693	31890	37179	31735
Simulated annealing	31910	33860	32681	35885	33217
Bayesian optimization	N/A	N/A	N/A	N/A	36692
Tabu Search	40647	47384	40873	44864	44747
PSO	41356	43989	41938	44332	41431

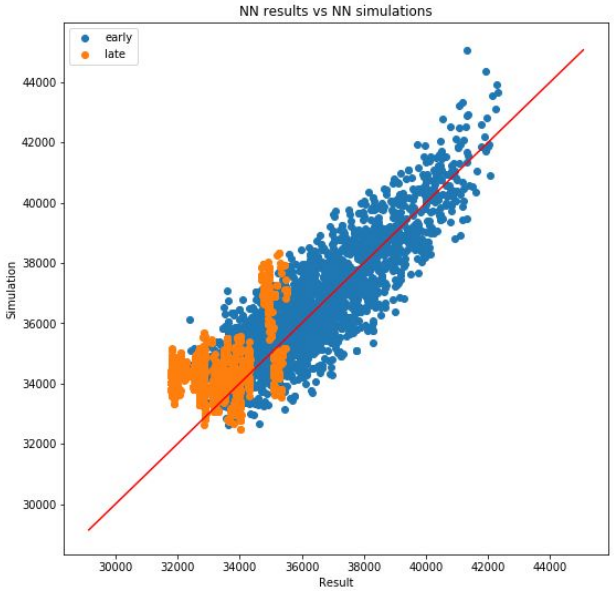
Comparison of best results found by optimization algorithms

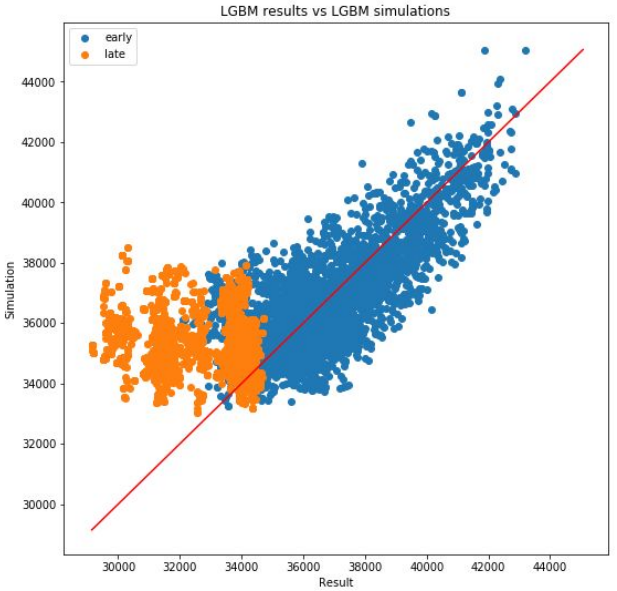
Distribution of error close to local optima



Results of a metamodel vs results of simulation for best settings from genetic algorithms (last 20 points from each run are marked orange)







Graph neural networks

Architecture I:

1. Neurons in the even numbered layers, starting with input layer as layer 0, should be localized at graph vertices (in our case - road crossings).

2. Neurons in the odd numbered layers should be localized at the graph edges (in our case - roads).

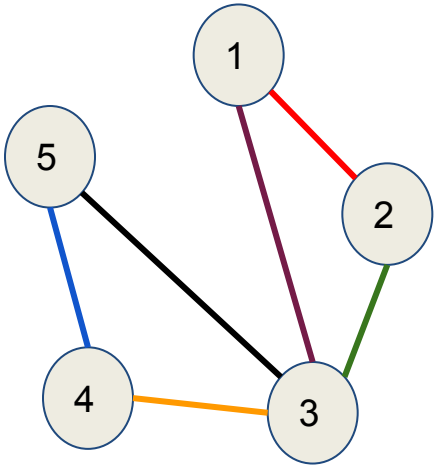
3. An exception should be the final layer with just one neuron.

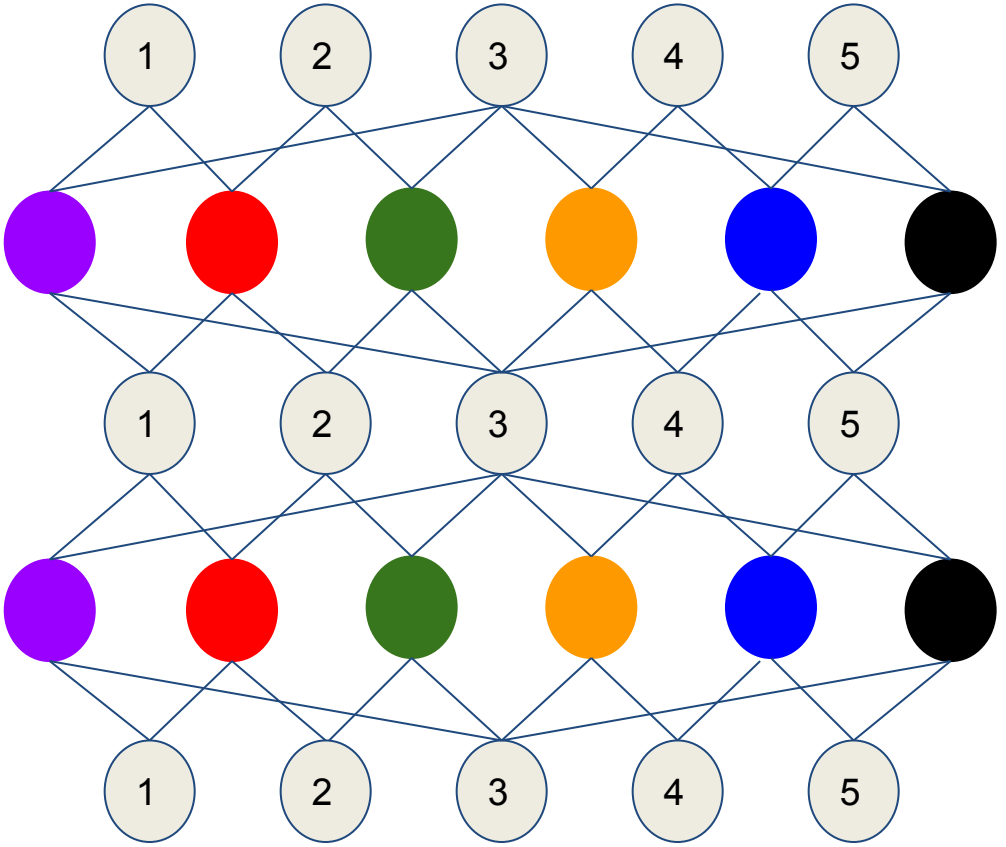
4. Connections from a vertex-localized layer to an edge-localized layer should only be present if a given vertex is an end of a given edge. There will be exactly two such connections for every edge neuron.

5. Connections from an edge-localized layer to a vertex-localized layer should only be present if the edge has the vertex as its end. The number of such connections will be equal to the number of particular vertex neighbors.

Graph neural networks

Architecture I:





Graph neural networks

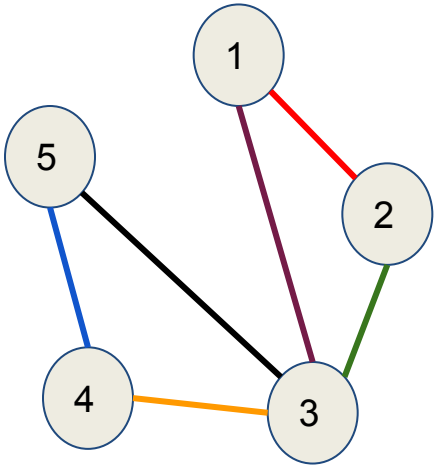
Architecture II:

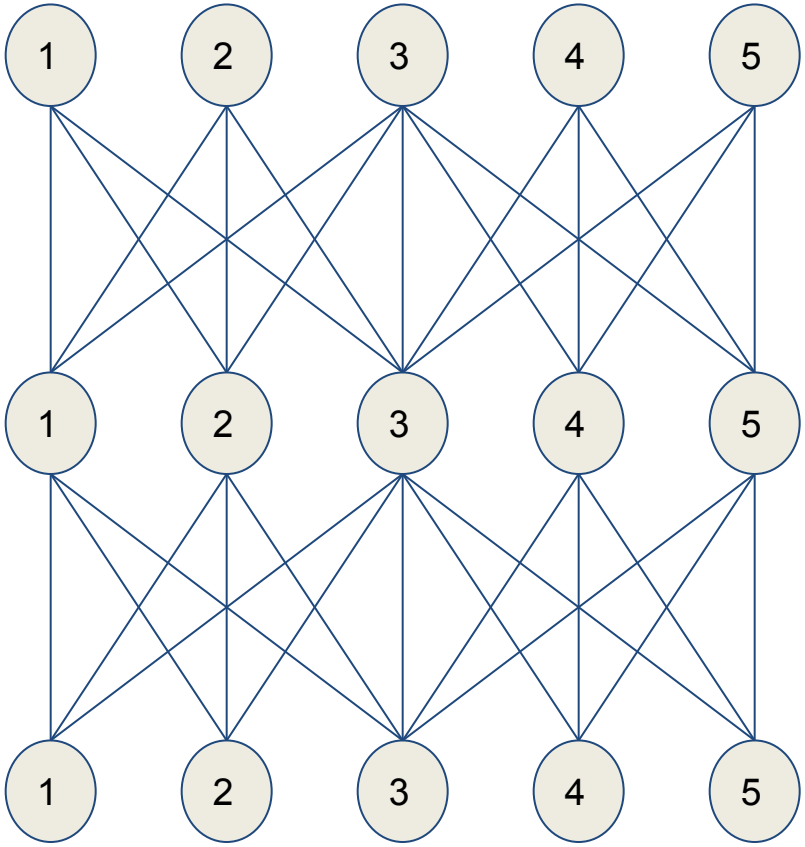
1. Neurons in all layers, with the exception of the output layer, should be localized at graph vertices.

2. Connections from a neuron in one layer to a neuron in the next one should only be present if the corresponding vertices are neighbors in the graph. The number of connections for the vertex node will be equal to the number of the vertex neighbors.

Graph neural networks

Architecture II:





Graph neural networks

Results (Architecture I):

- ➔ reduced avg error of approximation on a test set (before: 1.62%, now: 1.32%),
- ➔ reduced max error of approximation on a test set (before: 10-11%, now: 6.18%),
- ➔ similar minima attained in the simulation (before: 31735, now: 31827),
- ➔ lower approximation error near minima

#Lyr	#Ch	Act	MinSim	ErrTest	ErrSim	ErrSim-ErrTest	<37000.0	<36000.0	<35000.0	<34000.0	<33000.0	<32000.0
3	3	tanh	31827	1.76%	1.90%	0.14%	1.85%	1.66%	1.47%	1.56%	3.49%	6.80%
3	4	tanh	31909	1.65%	1.80%	0.15%	1.63%	1.55%	1.51%	1.86%	2.66%	5.31%
2	4	tanh	31937	1.80%	2.06%	0.26%	1.82%	1.88%	2.26%	2.88%	4.85%	7.37%
5	3	tanh	31957	1.71%	1.80%	0.08%	1.65%	1.50%	1.35%	0.86%	2.30%	4.29%
5	4	tanh	32077	1.49%	2.08%	0.60%	2.03%	2.02%	1.98%	2.62%	4.29%	–
5	5	tanh	32105	1.43%	1.83%	0.40%	1.85%	1.87%	2.10%	2.54%	3.63%	–
5	6	tanh	32120	1.40%	1.85%	0.46%	1.89%	1.77%	1.51%	1.58%	2.53%	–
5	2	tanh	32138	1.91%	2.23%	0.32%	2.24%	2.39%	2.82%	3.72%	5.35%	–
4	3	tanh	32142	1.67%	1.96%	0.29%	1.76%	1.64%	1.41%	1.39%	1.99%	–
2	3	tanh	32246	1.89%	2.36%	0.46%	2.14%	2.03%	2.37%	2.88%	5.17%	–
6	6	tanh	32298	1.32%	1.72%	0.40%	1.68%	1.60%	1.66%	2.07%	2.18%	–
5	4	relu	32301	1.60%	3.24%	1.64%	2.94%	2.47%	2.21%	2.09%	0.91%	–
4	6	tanh	32332	1.40%	1.68%	0.28%	1.64%	1.67%	1.75%	2.35%	3.30%	–
2	6	tanh	32337	1.62%	1.93%	0.30%	1.80%	1.70%	1.38%	1.70%	3.95%	–
3	6	tanh	32360	1.57%	1.63%	0.07%	1.63%	1.58%	1.59%	1.99%	2.71%	–
Average			**32139**	**1.61%**	**2.00%**	**0.39%**	**1.90%**	**1.82%**	**1.82%**	**2.14%**	**3.29%**	**5.94%**

Best 15 rows in terms of the minimum simulator output value obtained by gradient descent. Columns #Lyr, #Ch and Act mean the number of layers, number of channels and activation function, respectively. Column MinSim includes the minimum output value of the simulator. Columns ErrTest, ErrSim and ErrSim-ErrTest contains average errors obtained on a test set, on a gradient descent trajectory and a difference between those average errors, respectively. The next 6 columns contain average errors on subsets of the trajectories obtained by thresholding on the simulator output values.

Graph neural networks

Problem:
It is not fully clear that including information about the topology of a road network brings any value. Perhaps, any similar graph, even not related to the problem at hand, could do equally well.

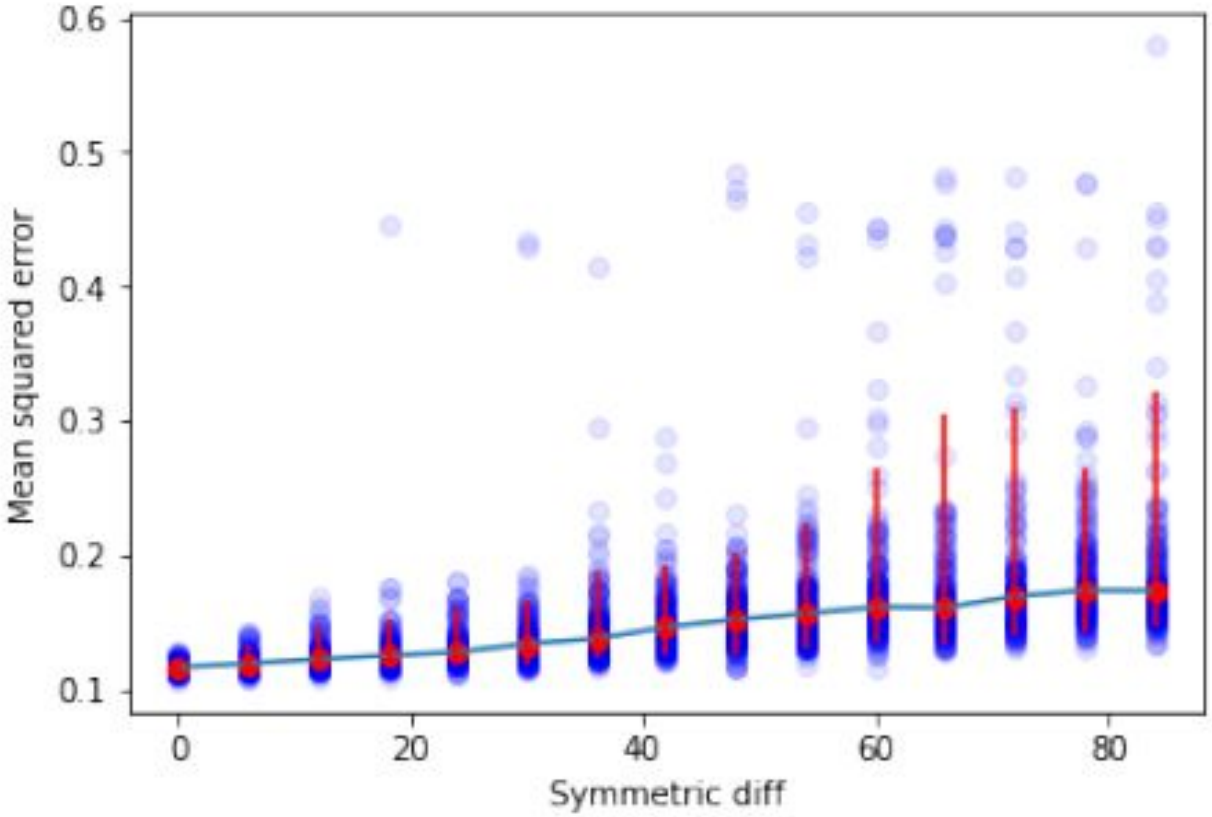
Sanity check:
We fixed the number of layers to 3 and the number of channels to 4 per layer (for architecture of type 1), and built our nets using random graphs with various degrees of similarity to the true problem graph (measure of similarity: symmetric difference between the sets of edges)

- Method 1: random edge insertions and deletions at the same time keeping the desired value of the symmetric difference.
- Method 2: random permutations of the vertex labels while keeping the connection graph structure exactly the same

Graph neural networks

Problem:

It is not fully clear that including information about the topology of a road network brings any value. Perhaps, any similar graph, even not related to the problem at hand, could do equally well.



Thank you for your attention!

Questions?

E-mail: p.gora@mimuw.edu.pl,
tensorcell.research@gmail.com

www: http://www.mimuw.edu.pl/~pawelg
http://www.tensortraffic.com

FB: https://www.facebook.com/TensorCell

"Logic can get you from A to B, imagination will take you everywhere" A. Einstein

*"The sky is **NOT** the limit"*