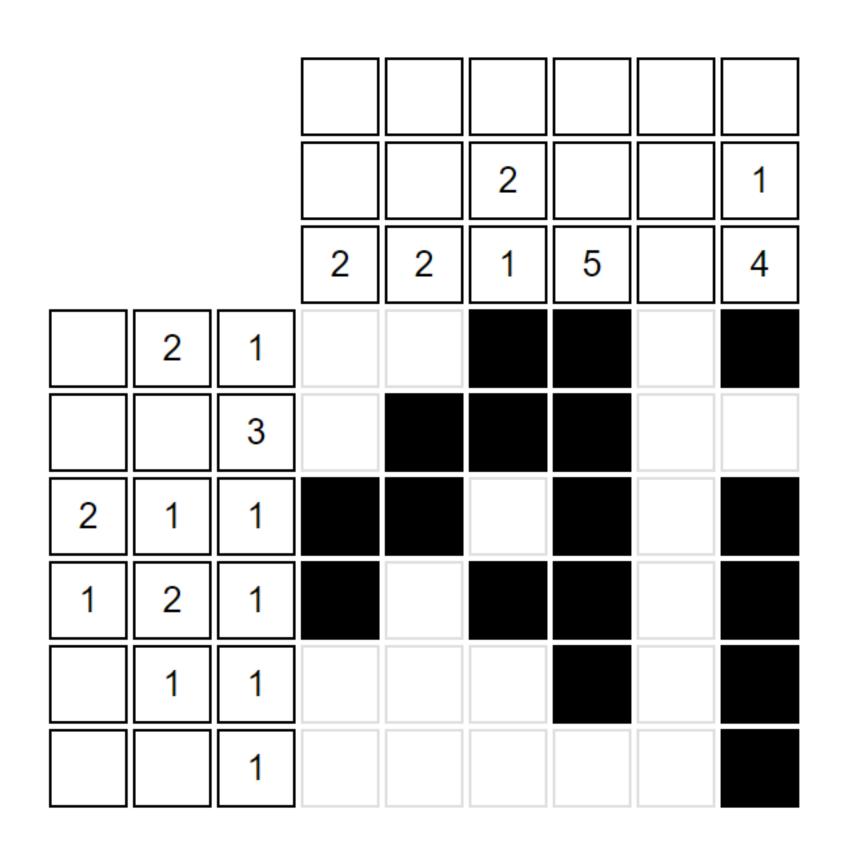
Deep Learning a gyakorlatban Python és LUA alapon Nagy házi feladat

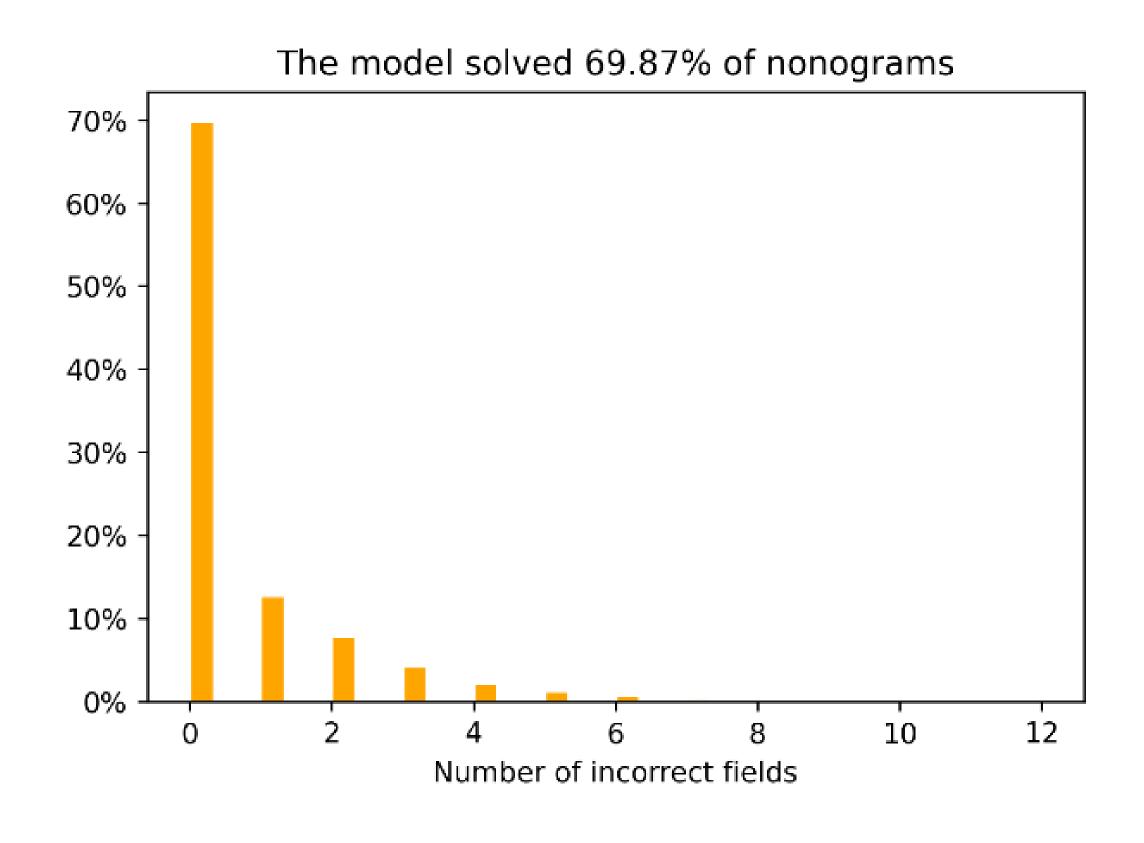
Nonogramok megoldása neurális hálókkal

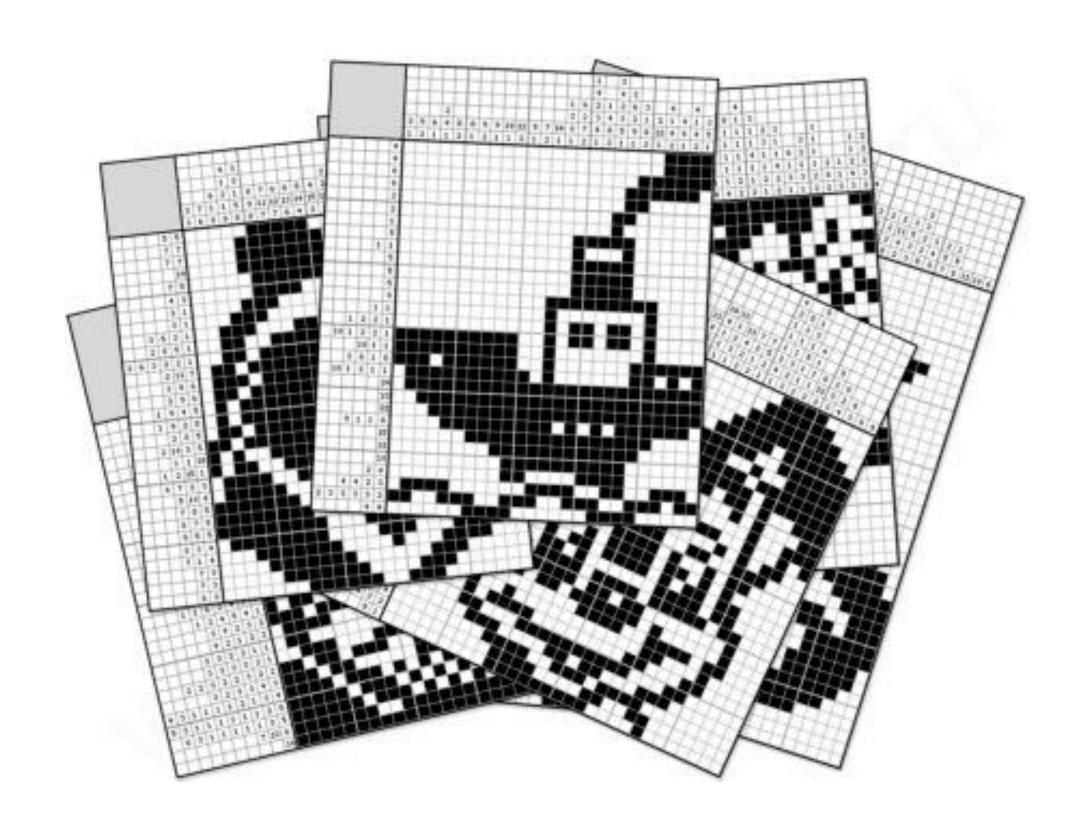
Solving nonograms using neural networks

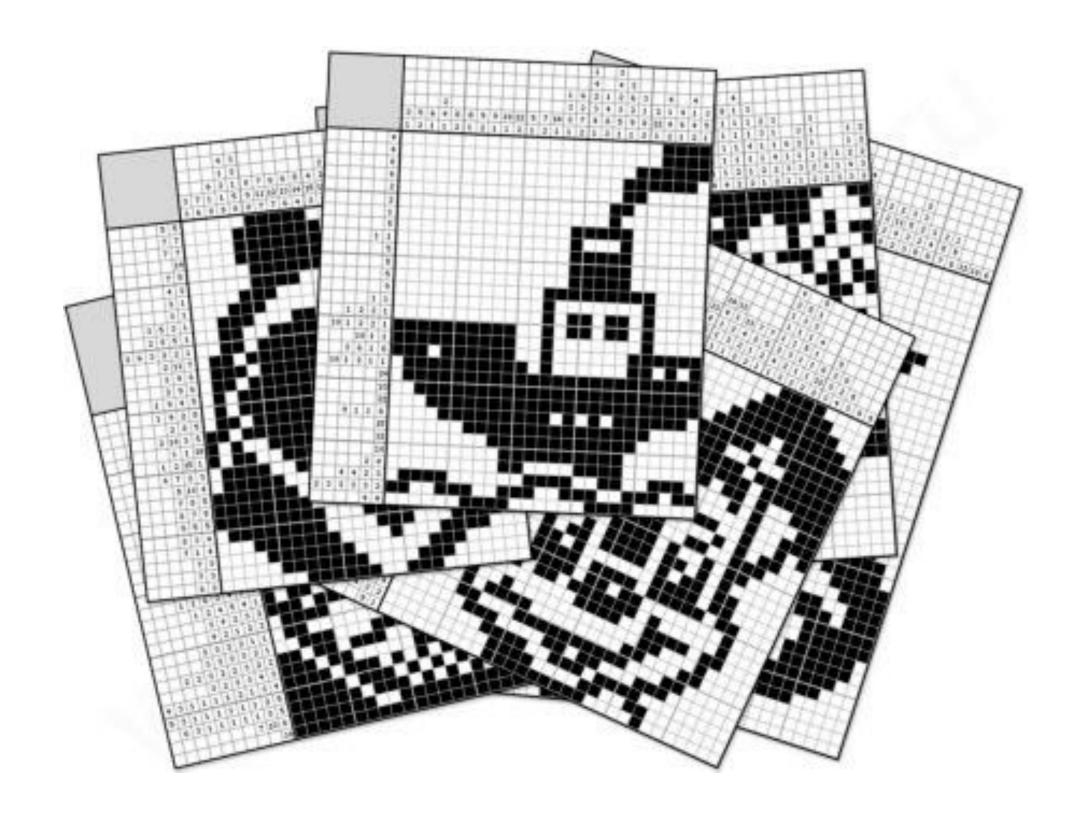


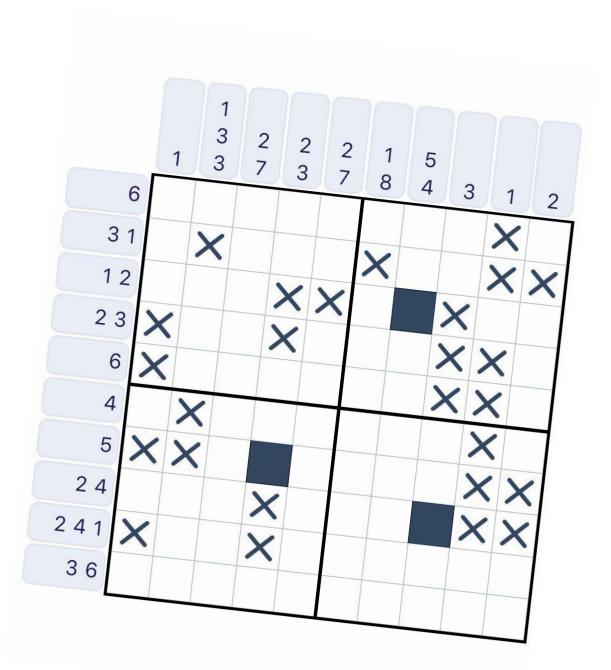
Nonogramok megoldása neurális hálókkal

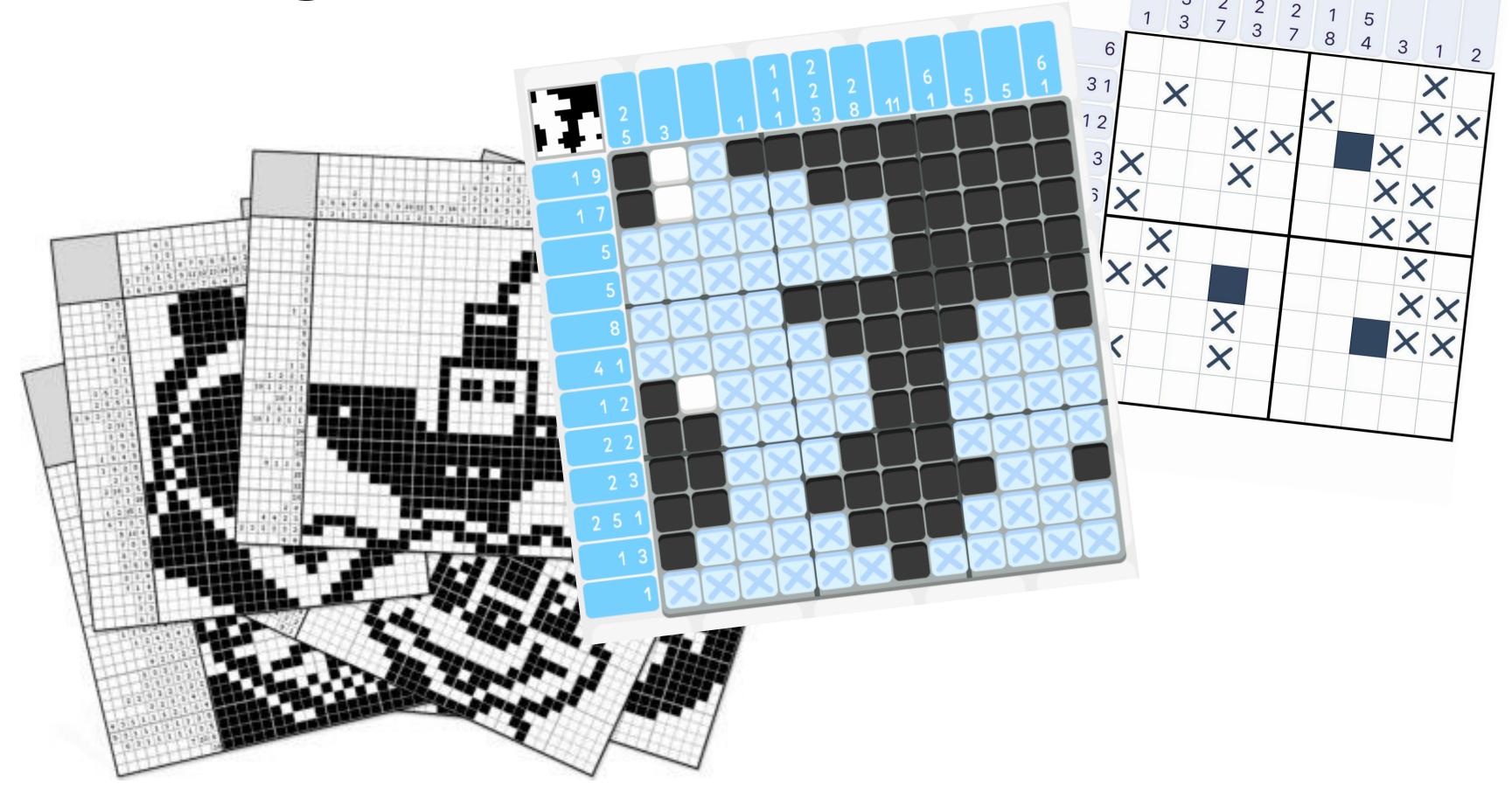
Solving nonograms using neural networks

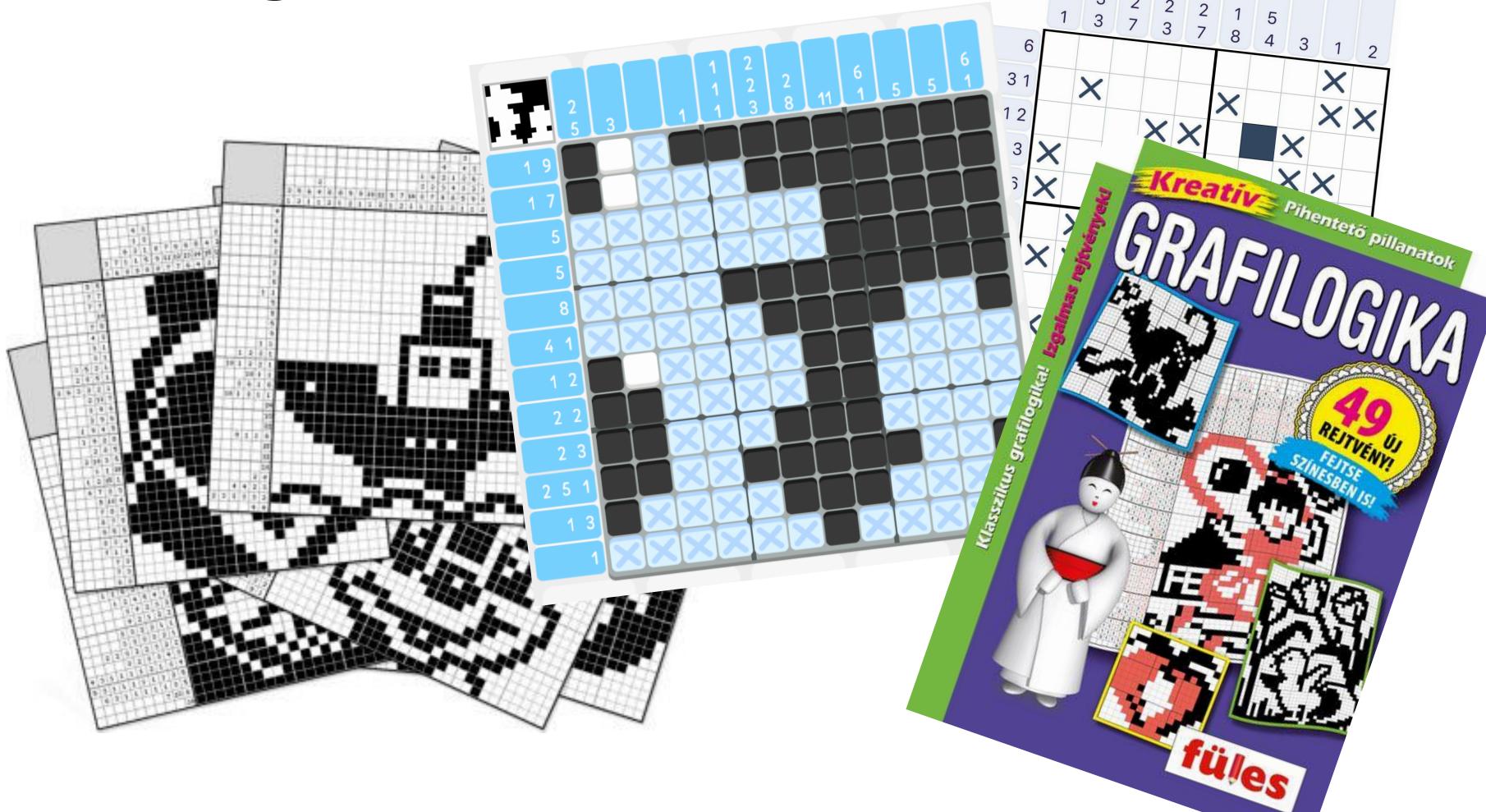






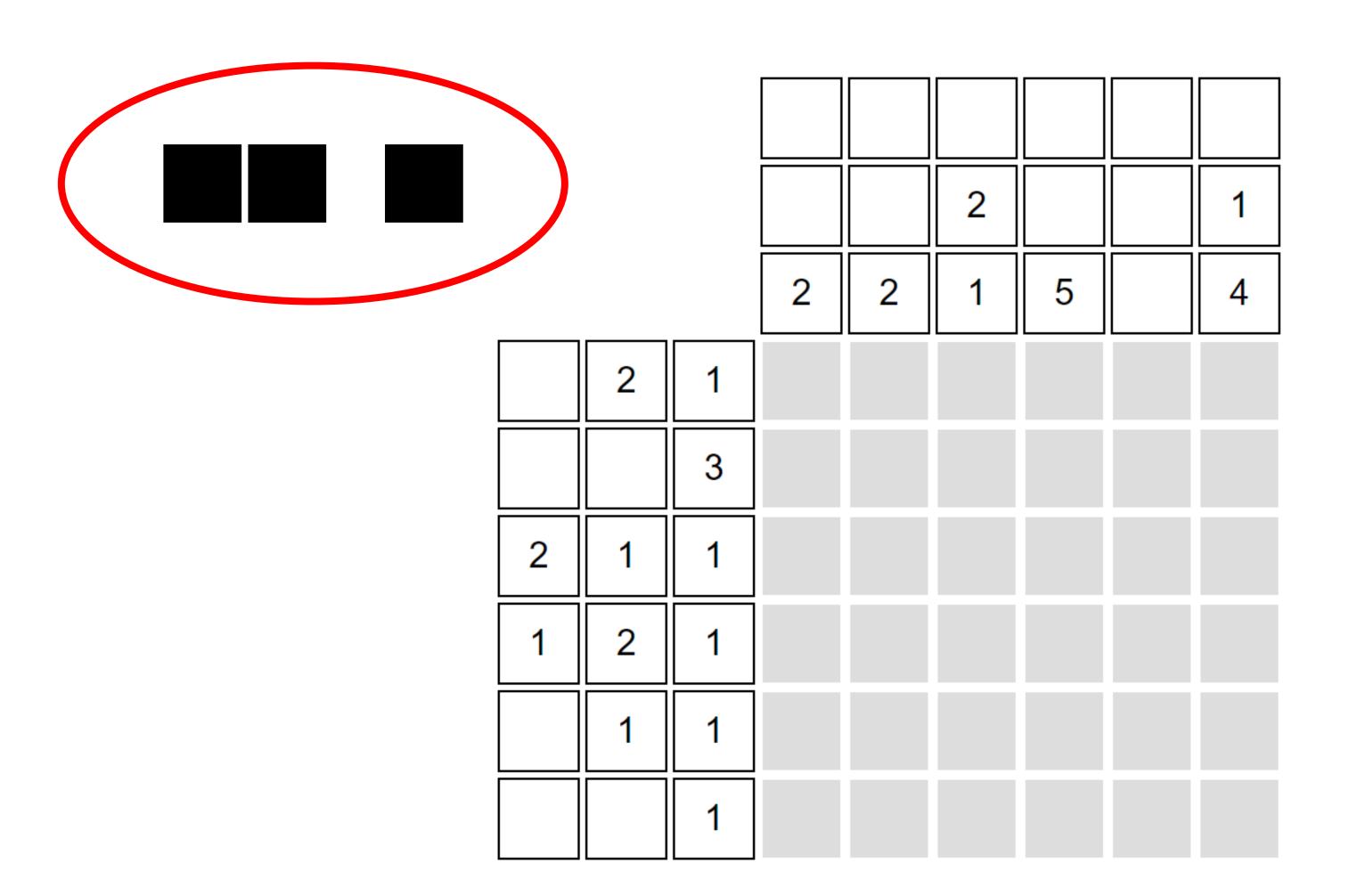


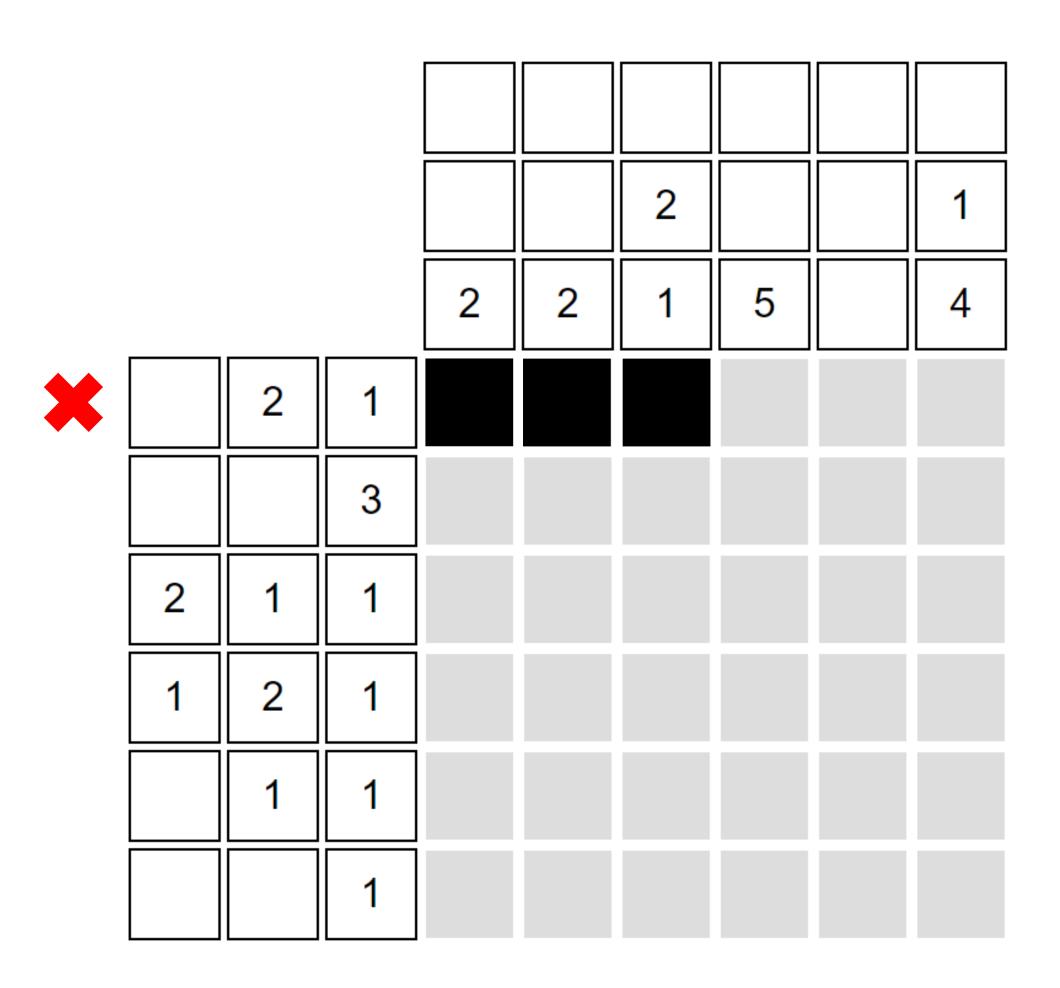


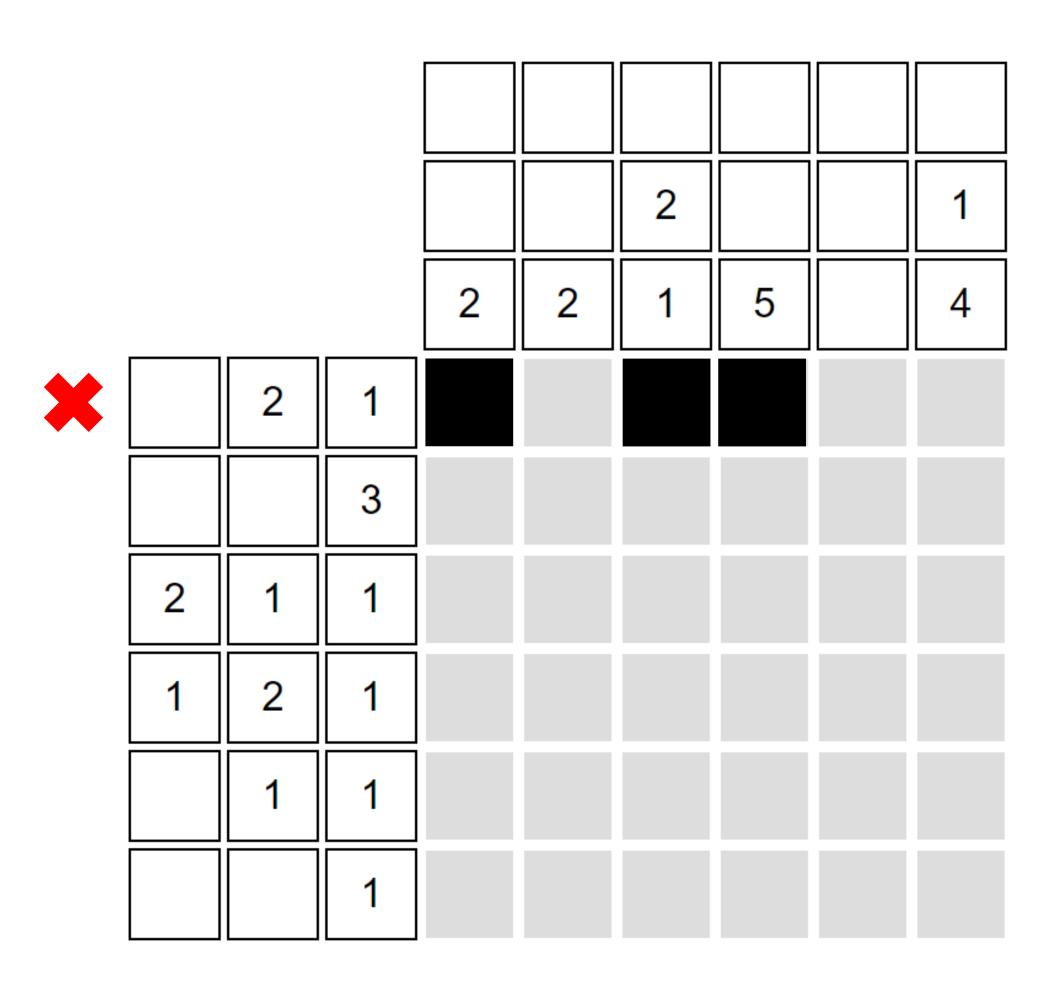


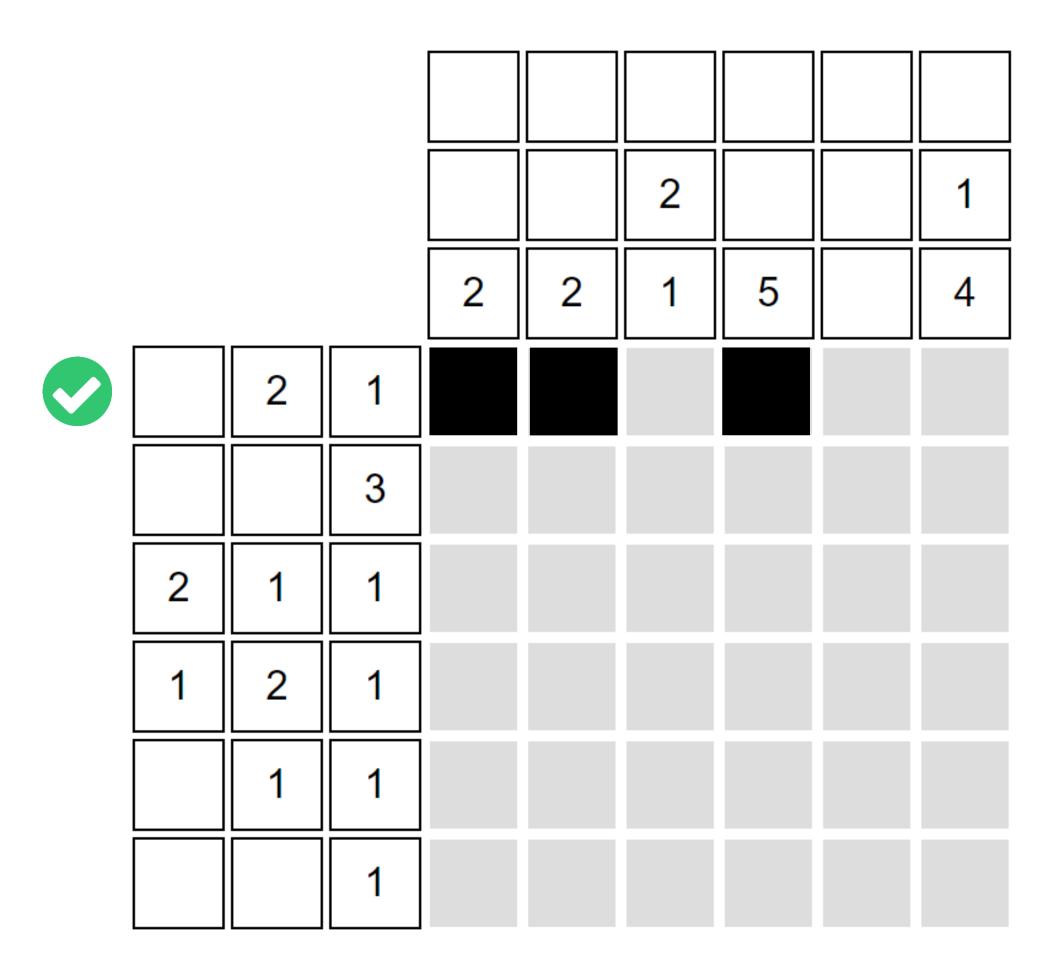
					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

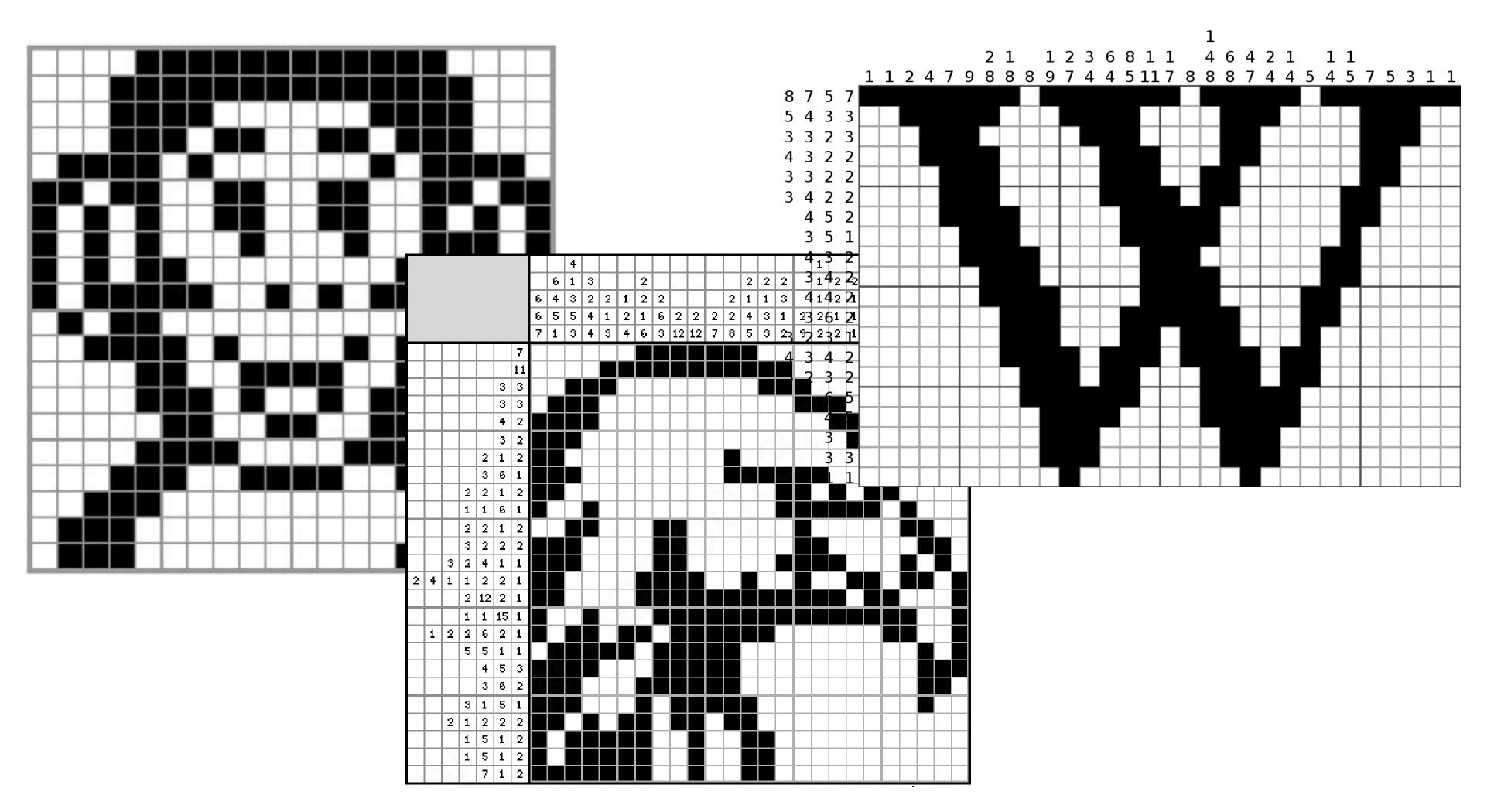
					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

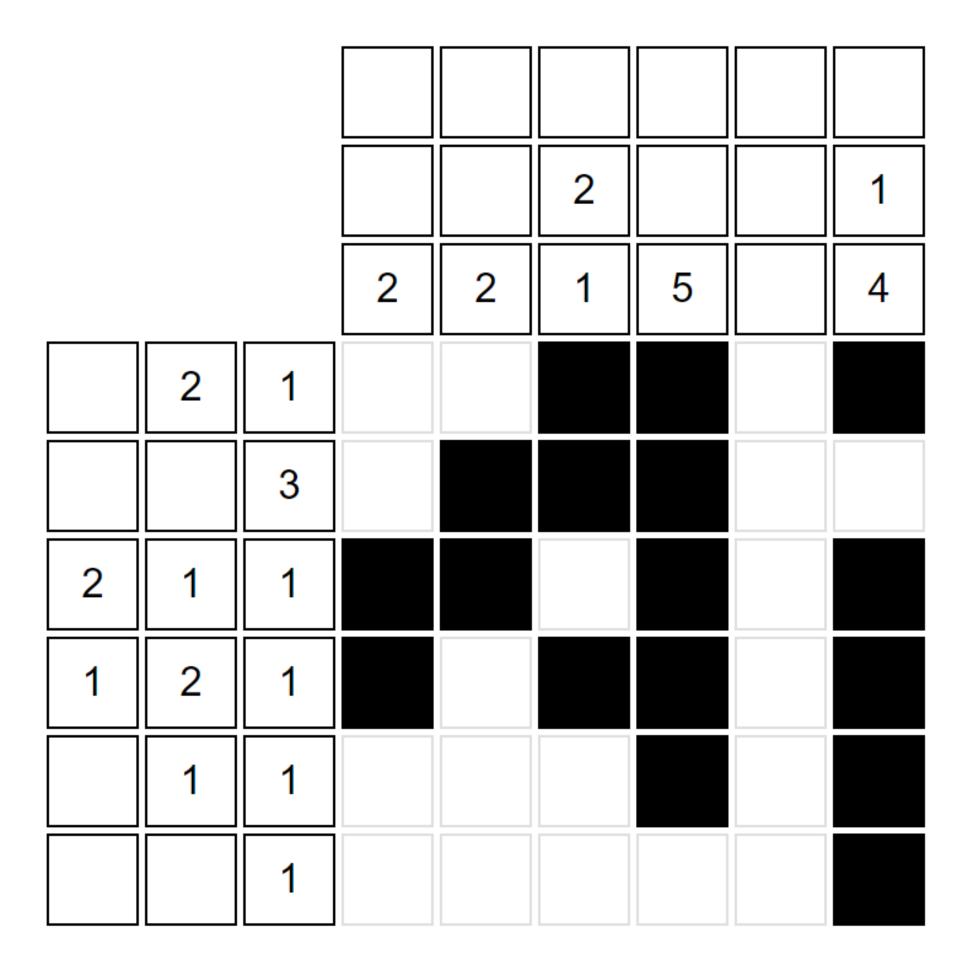












Irodalom

Solving Nonograms by combining relaxations

K.J. Batenburg^{a,*}, W.A. Kosters^b

^aVision Lab, Department of Physics, University of Antwerp, Universiteitsplein 1, B-2610 Wilrijk, Belgium

A comparison of a genetic algorithm and a depth first search algorithm applied to Japanese nonograms

Wouter Wiggers
Faculty of EFCMS University of Twente
w.a.wigg

Improved Automatic Generation of Curved Nonograms

Constructing Simple Nonograms of Varying Difficulty

K. Joost Batenburg^{*,1}, Sjoerd Henstra², Walter A. Kosters², and Willem Jan Palenstijn¹

¹Vision Lab, Department of Physics, University of Antwerp, Belgium

²Leiden Institute of Advanced Computer Science, Leiden

University, The Netherlands

Mees van de Kerkhof

Master Thesis ICA-3822443 Computer Science Utrecht University

bLeiden Institute of Advanced

Komplexitás

Corpus ID: 3040166

NP-completeness Results for NONOGRAM via Parsimonious Reductions

N. Ueda, T. Nagao · Published 1996 · Mathematics

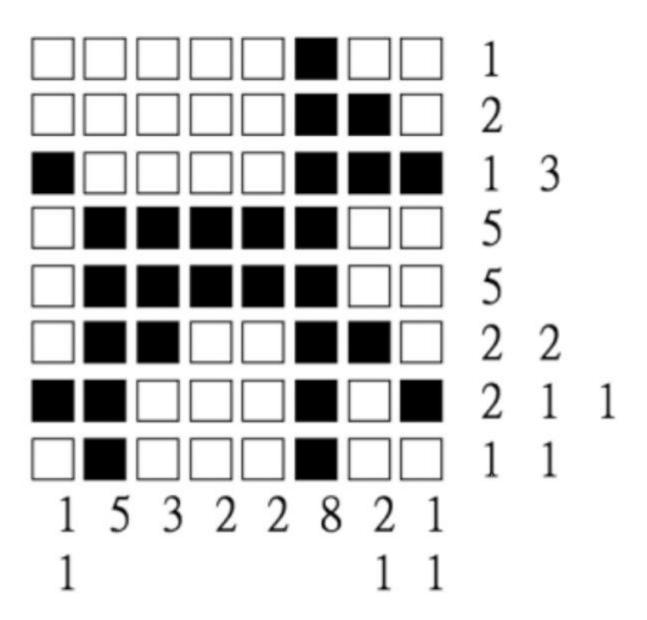
We introduce a new class of NP problems called ANOTHER SOLUTION PROBLEMs. For a given NP problem X, ANOTHER SOLUTION PROBLEM for X (ASP for X) is to ask, for a given instance I for X and its solution, whether there is another solution for I. The di culty of ASP for X may not be immediate from the di culty of X. For example, for some NP-complete problems such as 3SAT or 3DM, it is easy to show that their ASPs are NP-complete; on the other hand, ASP for, e.g., VERTEX COLORING is trivially in P... CONTINUE READING

Save to Library . Create Alert . Cite Launch Research Feed

Algoritmusok

An Efficient Approach to Solving Nonograms

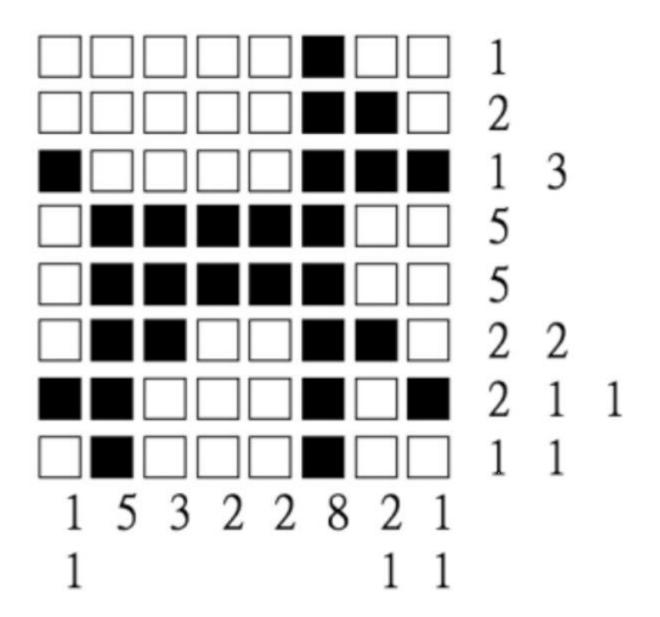
I.-Chen Wu, *Member, IEEE*, Der-Johng Sun, Lung-Ping Chen, Kan-Yueh Chen, Ching-Hua Kuo, Hao-Hua Kang, and Hung-Hsuan Lin



Algoritmusok

An Efficient Approach to Solving Nonograms

I.-Chen Wu, *Member, IEEE*, Der-Johng Sun, Lung-Ping Chen, Kan-Yueh Chen, Ching-Hua Kuo, Hao-Hua Kang, and Hung-Hsuan Lin



$$O(I * k)$$

I: grid size

k: average number of

integers in one constraint

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

Nonogramok és gépi tanulás

Solving nonogram puzzles by reinforcement learning

Frédéric Dandurand (frederic.dandurand@gmail.com)

Department of Psychology, Université de Montréal, 90 ave. Vincent-d'Indy Montréal, QC H2V 2S9 Canada

Denis Cousineau (denis.cousineau@uottawa.ca)

École de psychologie, Pavillon Vanier, Université d'Ottawa 136 Jean Jacques Lussier, Ottawa, Ontario, K1N 6N5, Canada

Thomas R. Shultz (thomas.shultz@mcgill.ca)

Department of Psychology and School of Computer Science, McGill University, 1205 Penfield Avenue Montreal, QC H3A 1B1 Canada

Nonogramok és gépi tanulás

(n+n)! combinations of sequences

```
n = 5 \rightarrow 3.6 million combinations
```

$$n = 6 \rightarrow 497$$
 million combinations

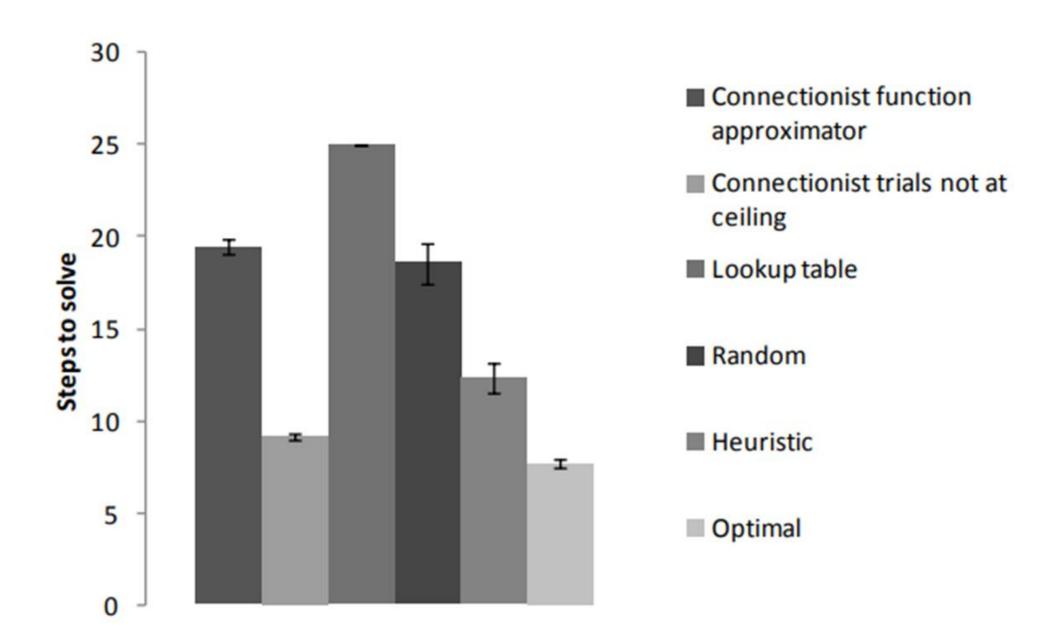
$$n = 7 \rightarrow 8.7 * 10^{10}$$
 million combinations

Nonogramok és gépi tanulás

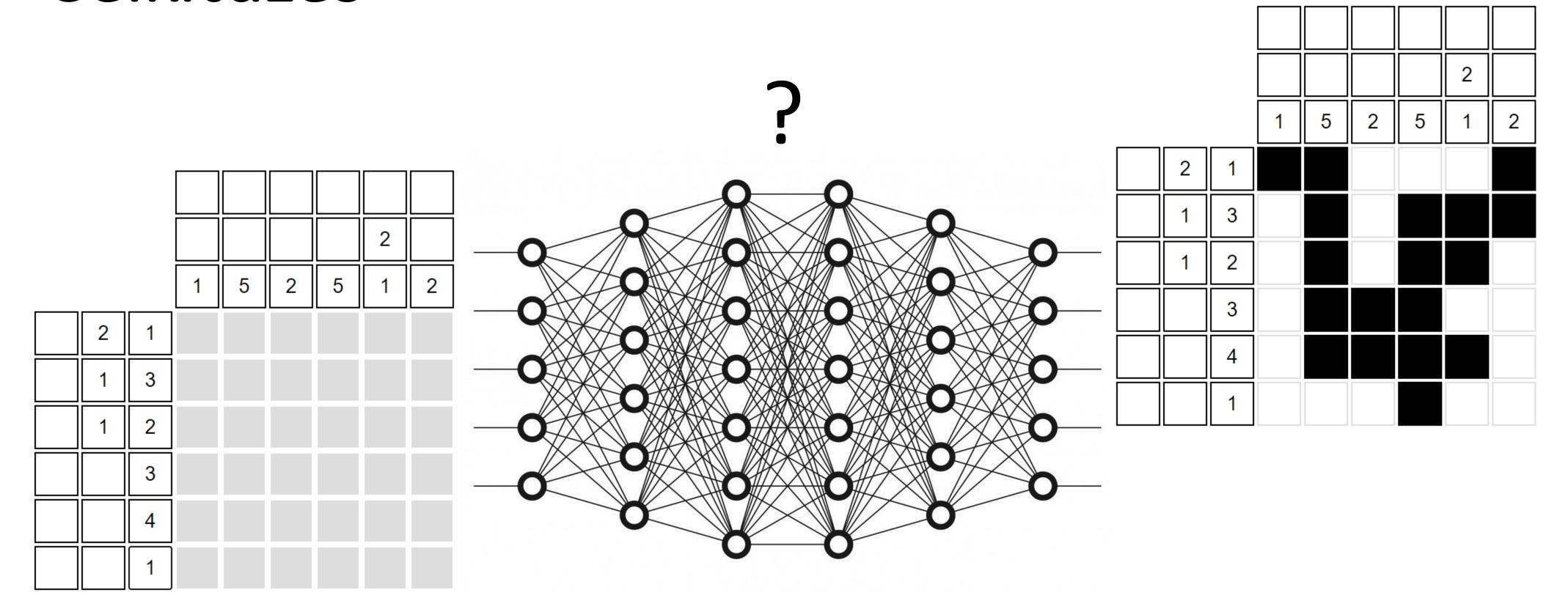
State: of the cells on the corresponding line

Action: choice of the next line to solve

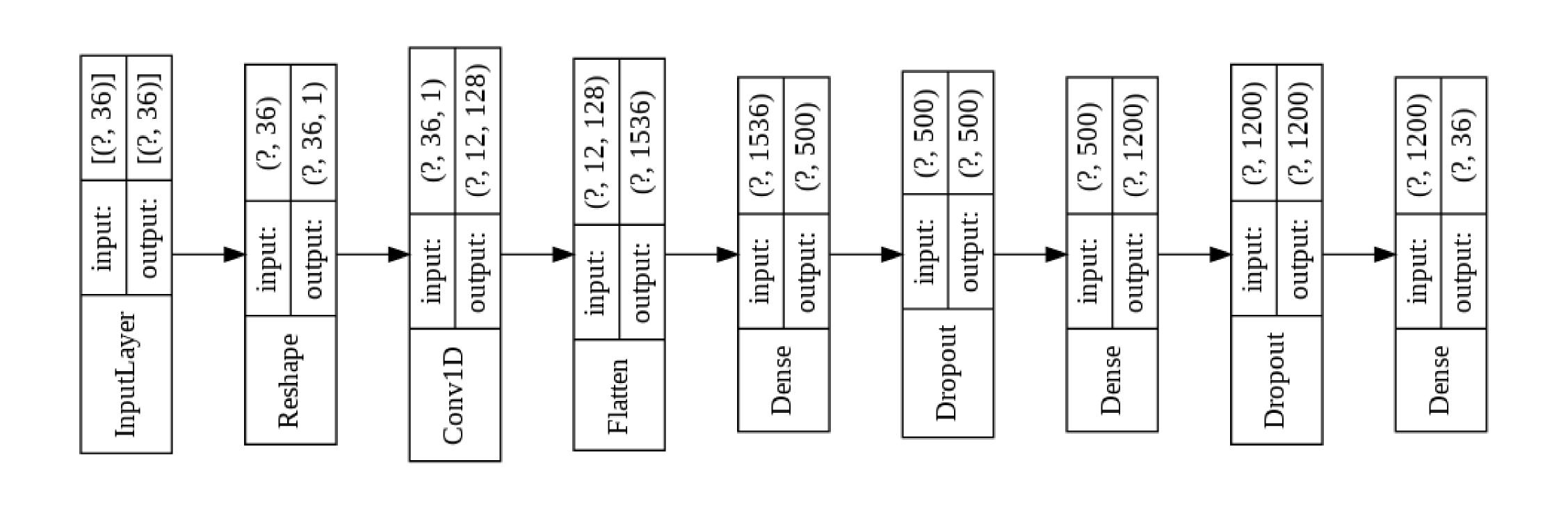
Reward: proportion of the currently unknown cells that are determined as filled or empty



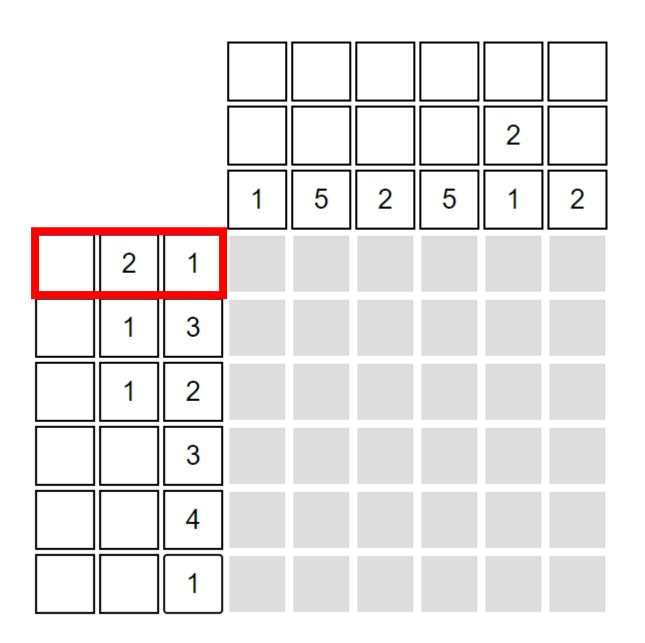
Célkitűzés

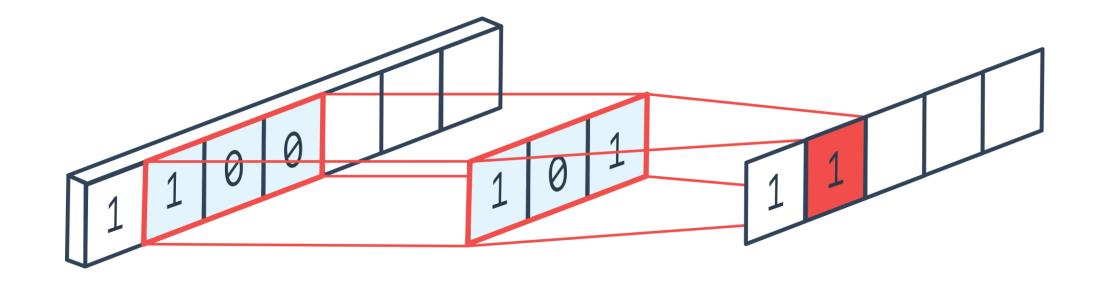


Rendszerterv



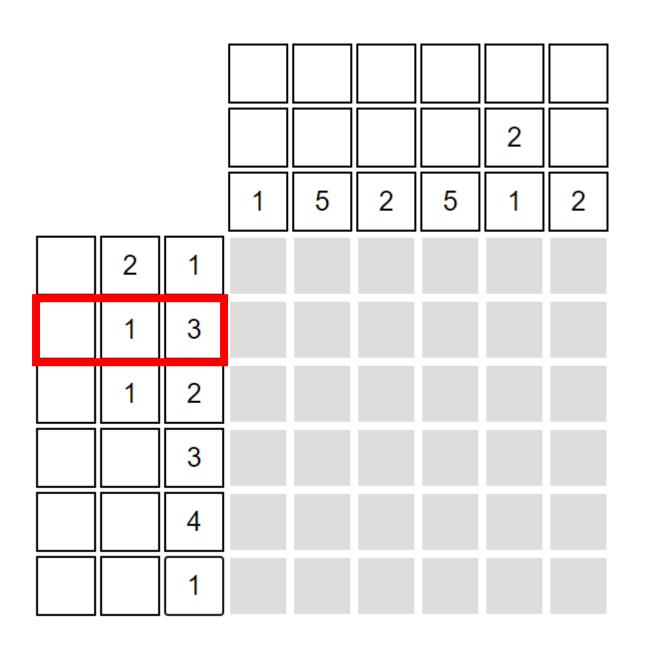
Konvolúciós réteg

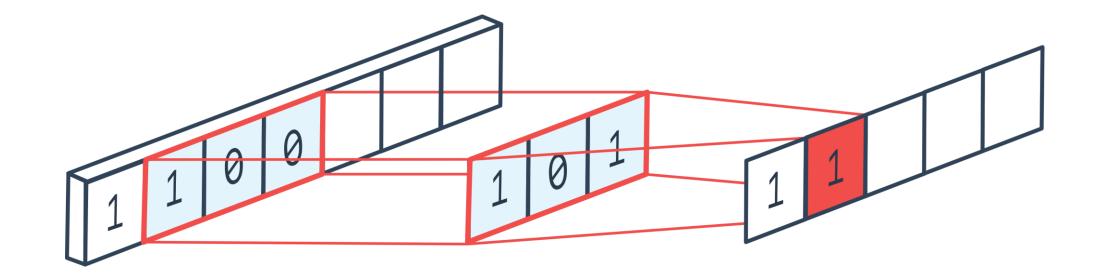




0210130120030040010010...]

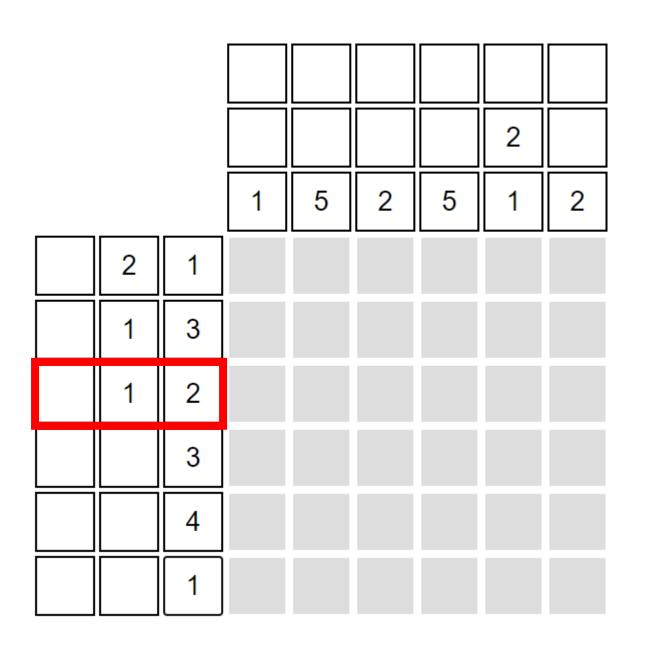
Konvolúciós réteg

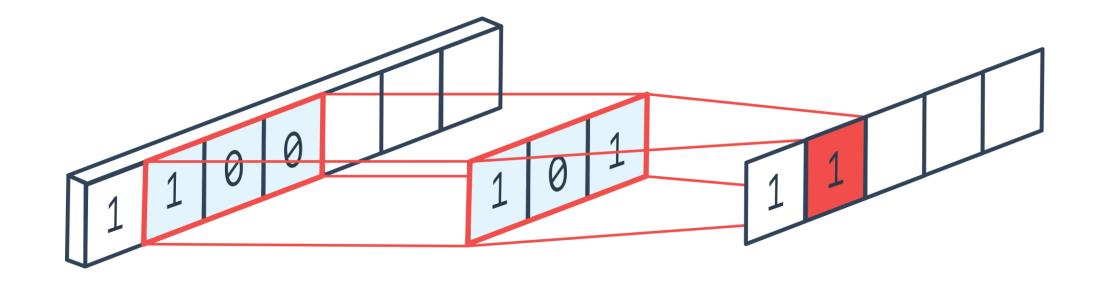




[0 2 1 0 1 3 0 1 2 0 0 3 0 0 4 0 0 1 0 0 1 0 ...]

Konvolúciós réteg





[0 2 1 0 1 3 0 1 2 0 0 3 0 0 4 0 0 1 0 0 1 0 ...]

Adatgenerálás

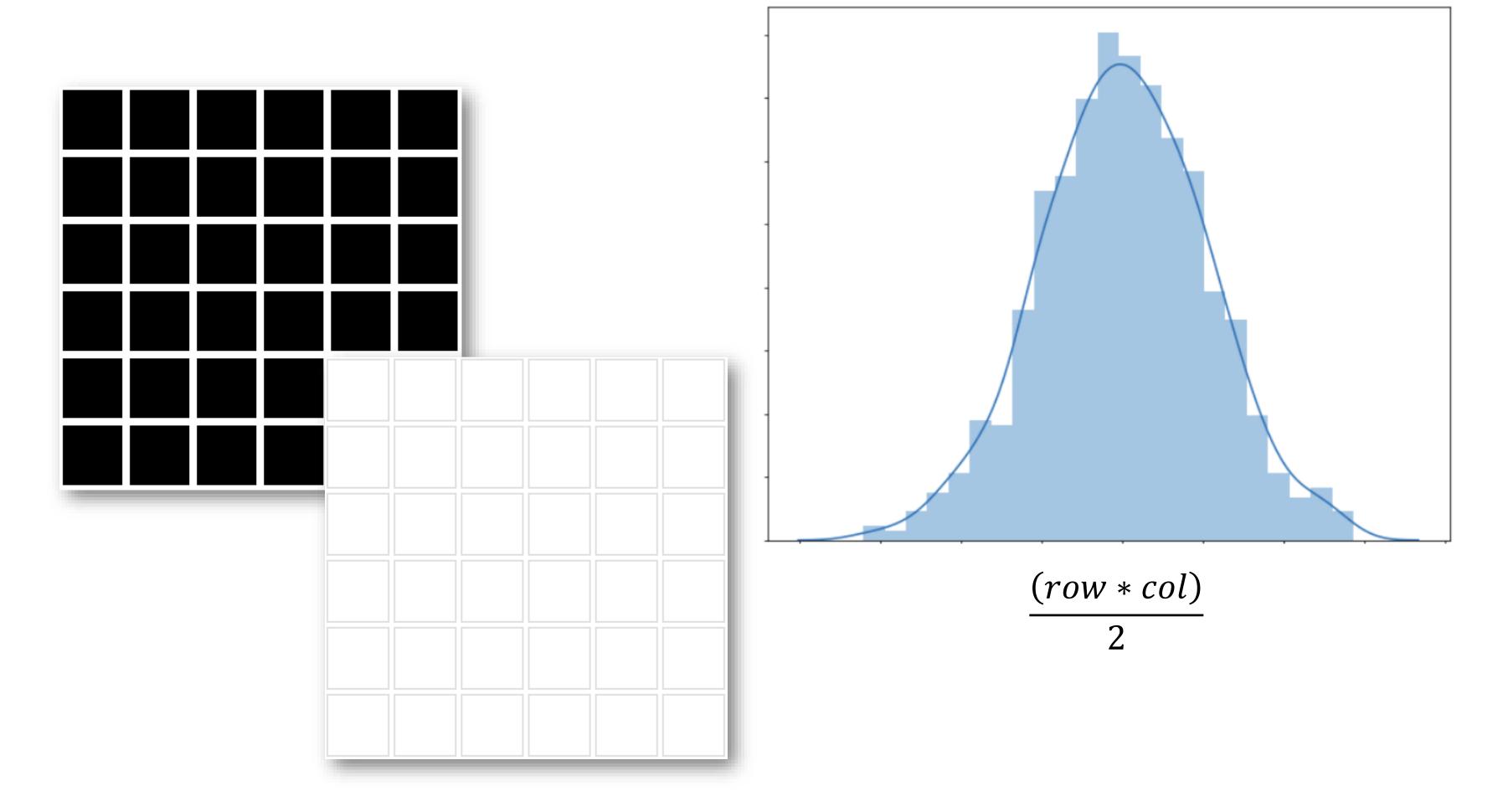
Generating data

```
datagen.py
-s, --samples Number of generated samples
-r, --rows Number of rows in a nonogram
-c, --columns Number of columns in a nonogram
-t, --train Train split [0.0, 1.0]
-v, --valid Valid split [0.0, 1.0]
-o, --output Output folder
```

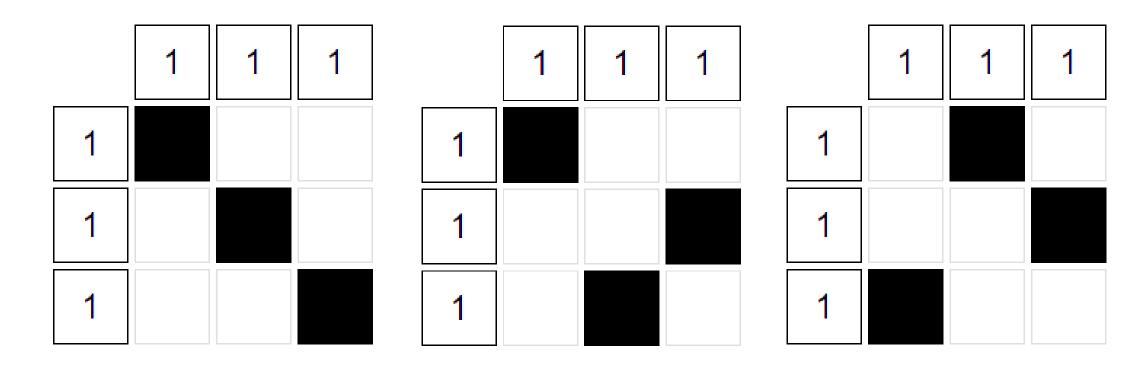
	4 4				
1	1 1	1 1			
1 1 2 1 1	1 1 1	1 1 1	2 1	1 1	2 11
113 1 4	2 2 1 4 2 1		3 2 1 4 4 1 4	1 111113	3 1 2 1 1 1 2 3
2 3 2 1 6 1 3 1	1 4 4 1 1 2 3 1	1 4 4 1 1 2 3 1	3 4 2 2 3 3 1 1	2 1 2 3 2 1 4 4	1 1 4 1 2 2 2 3
11000000000	2 1 1	2 1 1	1 1 2 1	5 0 0 0	2 1 1 0
3 1 0 0	1208088000	12000000	113		3 2
15000000	113	11380808880	1 3 1	11000 - 000	111
2 2 2	7 🗆 🗷 🗷 🗷 🗷 🗷	7	1400000		1 1 2 1
110000 - 00	3 1 • • • • • • •	3 1	41		3000000
1110	4 1 1		2 3 1		1400000
4	11000000000		3 2		
2 1 □ ■ ■ □ □ □ □ ■	1 2			1111	
			~ ~ 1		
	1		2		
1	1 31 11	1 1	_		
2 2 1 2 1 4	1 1 31 11 23112211	1 1 5 1131 1	1 1 1 1	1 1 4 4 1 3	1 3 1 3 2 2 1
			1 1 1 1	1 1 4 4 1 3 6 1 3 3 1 1 2 1	$\begin{array}{cccccccccccccccccccccccccccccccccccc$
2 2 1 2 1 4	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2	5 1 1 3 1 1 1 3 2 2 1 3 6 1	1 1 1 1 1 1 1 1 3 4 3 2 4 1 1 2 1 2 2	1 1 4 4 1 3 6 1 3 3 1 1 2 1	1 3 1 3 2 2 1
2 2 1 2 1 4 3 6 1 2 1 3 1 1	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 8 8 8 0 8 8	5 1 1 3 1 1 1 3 2 2 1 3 6 1 1 2 1 • • • • • • •	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 \square \square \square \square \square \square \square	$\begin{array}{cccccccccccccccccccccccccccccccccccc$
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 \square	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 0	5 1 1 3 1 1 1 3 2 2 1 3 6 1 1 2 1 8 0 0 0 8 8 0 8	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0	1 3 1 3 2 2 1 3 1 2 2 1 1 3 2 1 2 1 0 8 0 0 8 8 0 8
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 3 2 0 0 0 0 0 0 0 0	5 1 1 3 1 1 1 3 2 2 1 3 6 1 1 2 1 • • • • • • • • • • • • • • • • •	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0	1 3 1 3 2 2 1 3 1 3 2 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 1 3 2 1 1 1 1
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 3 2 0 0 0 0 0 0 0 0	5 1 1 3 1 1 1 1 3 2 2 1 3 6 1 1 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 1 1 1 3 4 3 1 1 1 1 2 1 2 2 3 2 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0 0 1 3 1 0 0 0 0 0 0 0 0	1 3 1 3 2 2 1 3 2 2 1 3 2 2 1 1 3 2 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 1 3 2 1 1 1 3 1 1 1 1
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 3 0 8 8 8 0 8 8 0 0 8 8 8 0 0 8 8 0	5 1 1 3 1 1 1 1 3 2 2 1 3 6 1 1 1 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0 1 3 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0	1 3 1 3 2 2 1 3 2 2 1 3 2 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 1 3 1 1 1 3 1 1 1 1
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 3 1 1 2 2 1 1 2 3 3 1 1 3 5 1 2 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	5 1 1 3 1 1 1 1 3 2 2 1 3 6 1 1 1 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 1 1 1 3 4 3 2 4 1 1 2 1 2 2 3 2 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 2 2 1 0 0 0 0 0 0 0 0 4 1 0 0 0 0 0 0 0 0	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0 0 1 3 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 0 1 2 0 0 0 0 0 0 0 0 0	1 3 1 3 2 2 1 3 2 3 1 3 2 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 3 2 1 1 1 3 2 1 1 1 3 1 1 1 1
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 3 0 8 8 8 0 8 8 0 0 8 8 8 0 0 8 8 0	5 1 1 3 1 1 1 3 2 2 1 3 6 1 1 2 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 4 4 1 3 6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0 0 1 3 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 0 1 2 0 0 0 0 0 0 0 0 0 4 2 0 0 0 0 0 0 0 0 0	1 3 1 3 2 2 1 3 2 2 1 3 2 2 1 1 3 2 2 1 1 3 2 2 1 1 3 2 2 1 1 3 2 1 1 3 2 1 1 1 3 2 1 1 1 1

Cellák értéke

Normáleloszlás



Egyértelműség



Tanítás

Hibafüggvény

$$ext{MAE} = rac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

MAE = mean absolute error

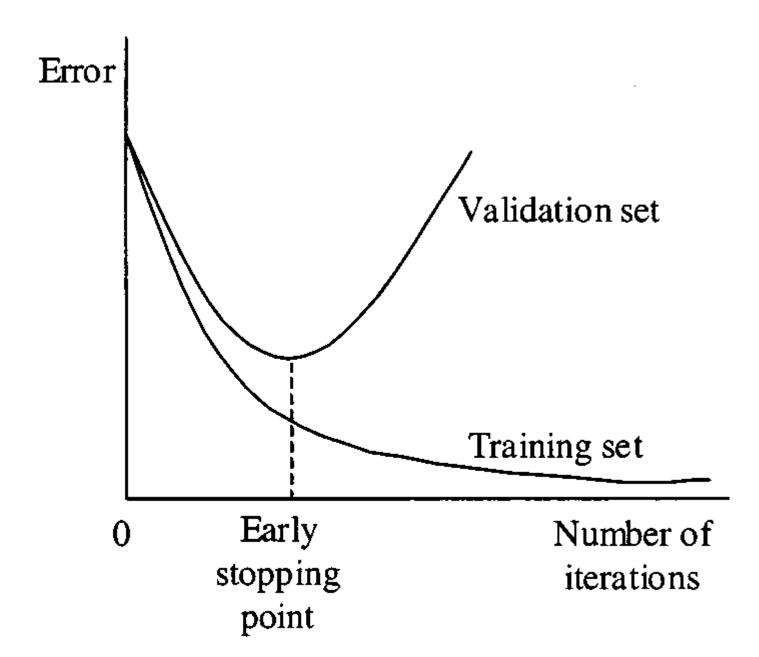
 y_i = prediction

 x_i = true value

n = total number of data points

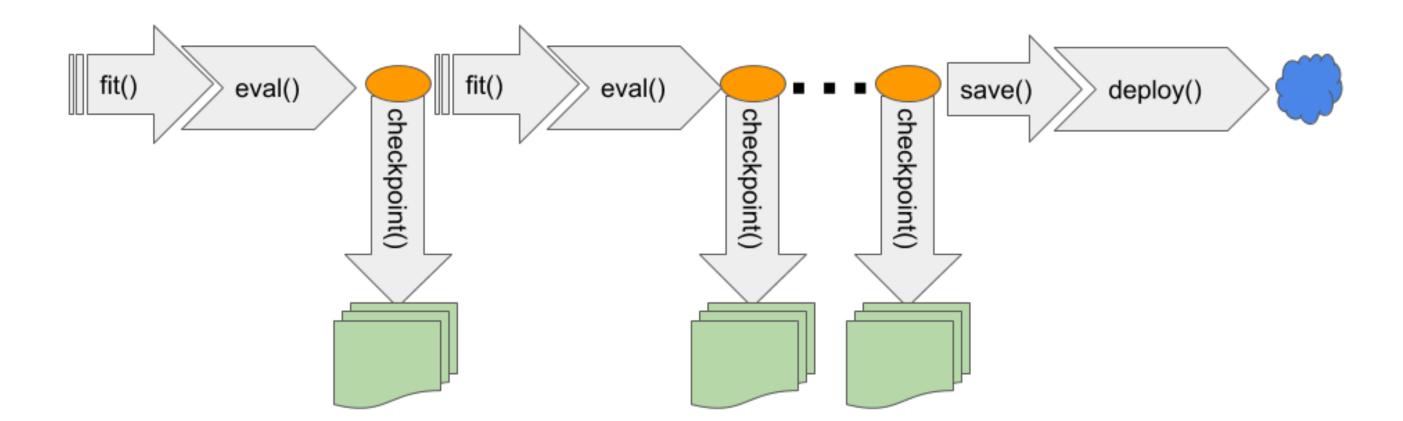
Early Stopping

early_stopping = EarlyStopping(patience=10, verbose=1)

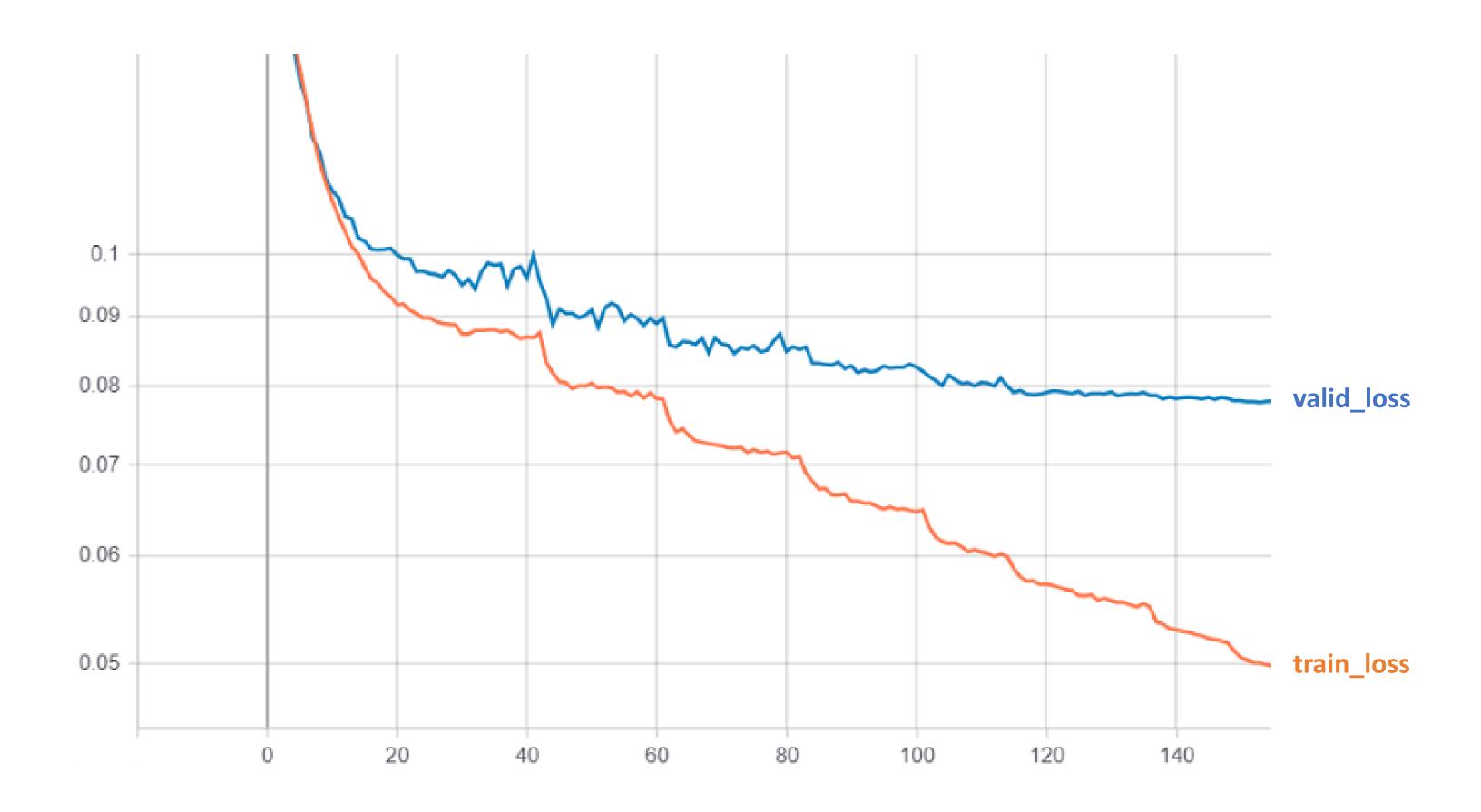


Checkpoints

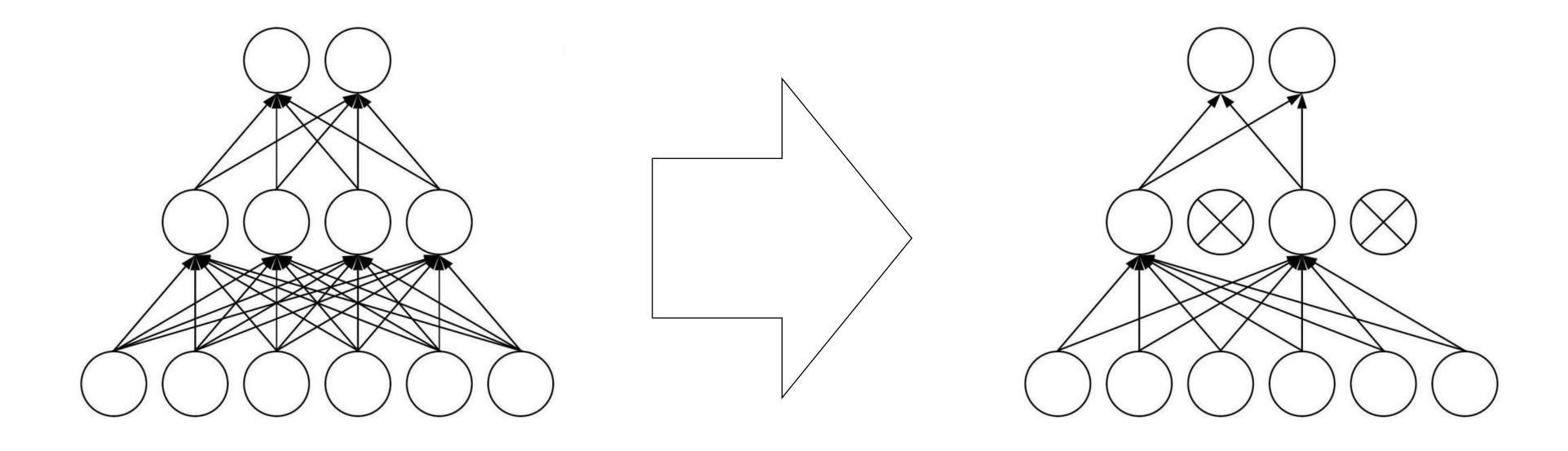
checkpoint = ModelCheckpoint(filepath='cnn_model_best', save_best_only=True, verbose=1)



Túltanulás



Dropout

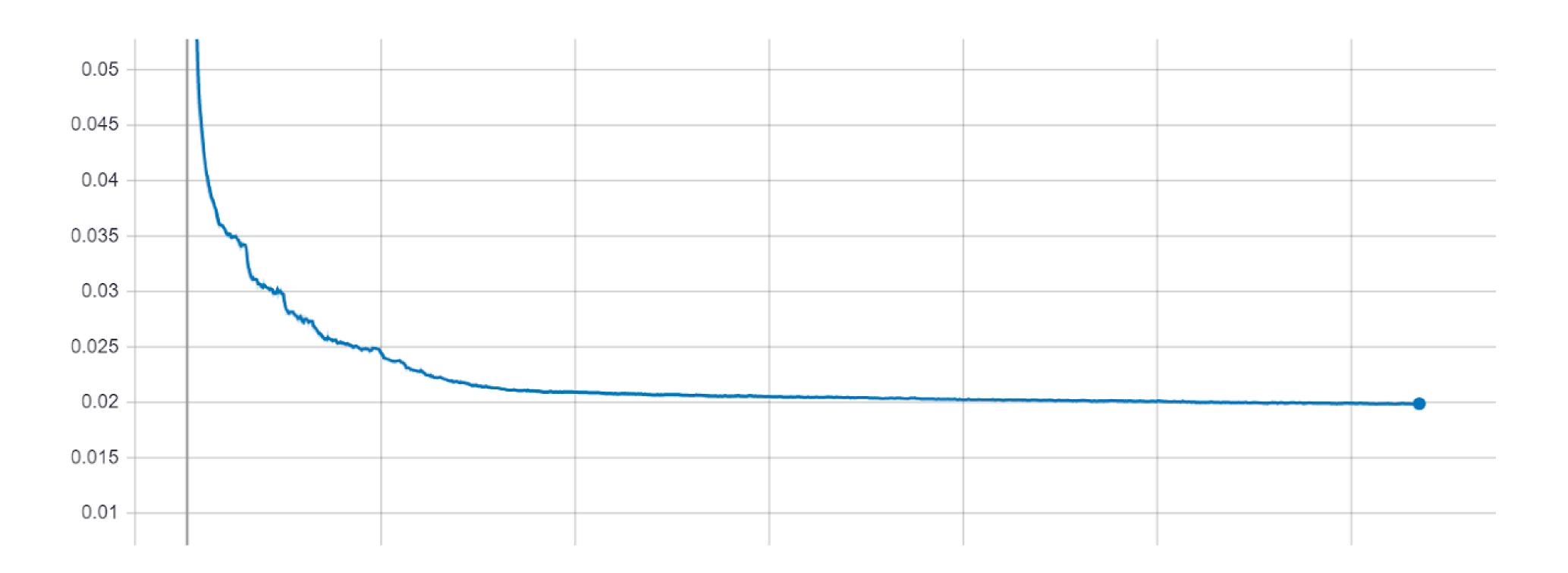


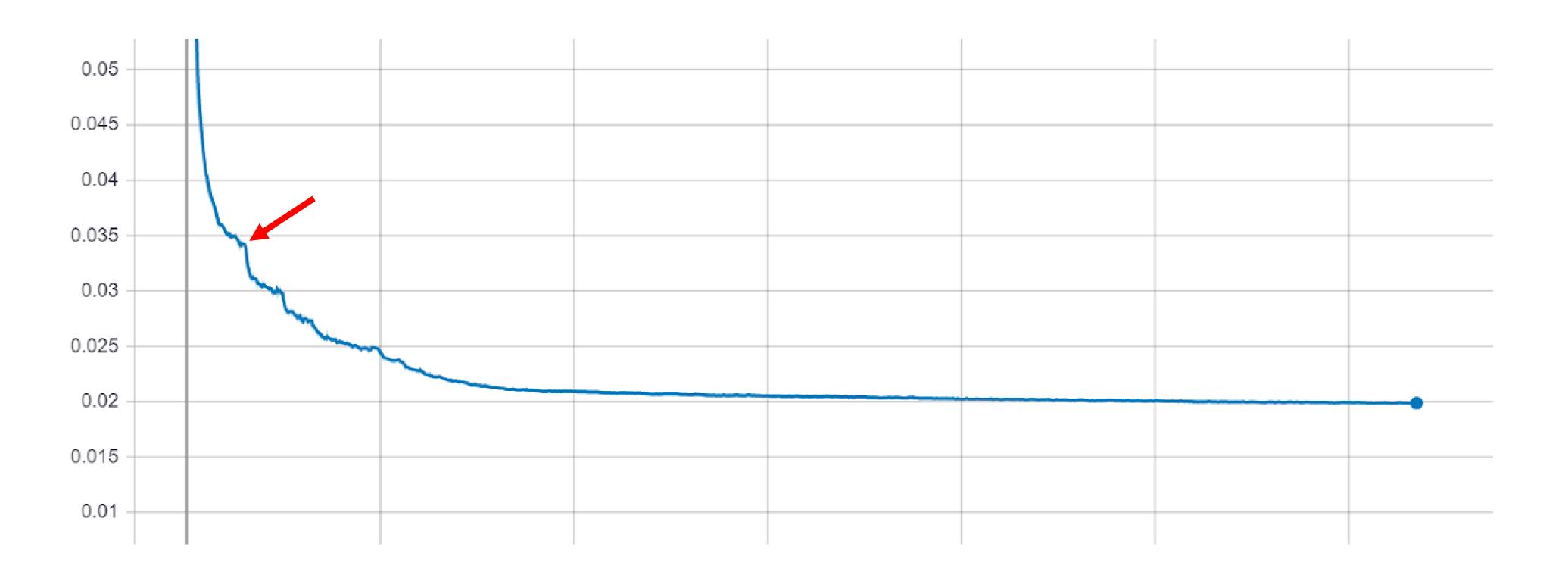
Adatgenerálás

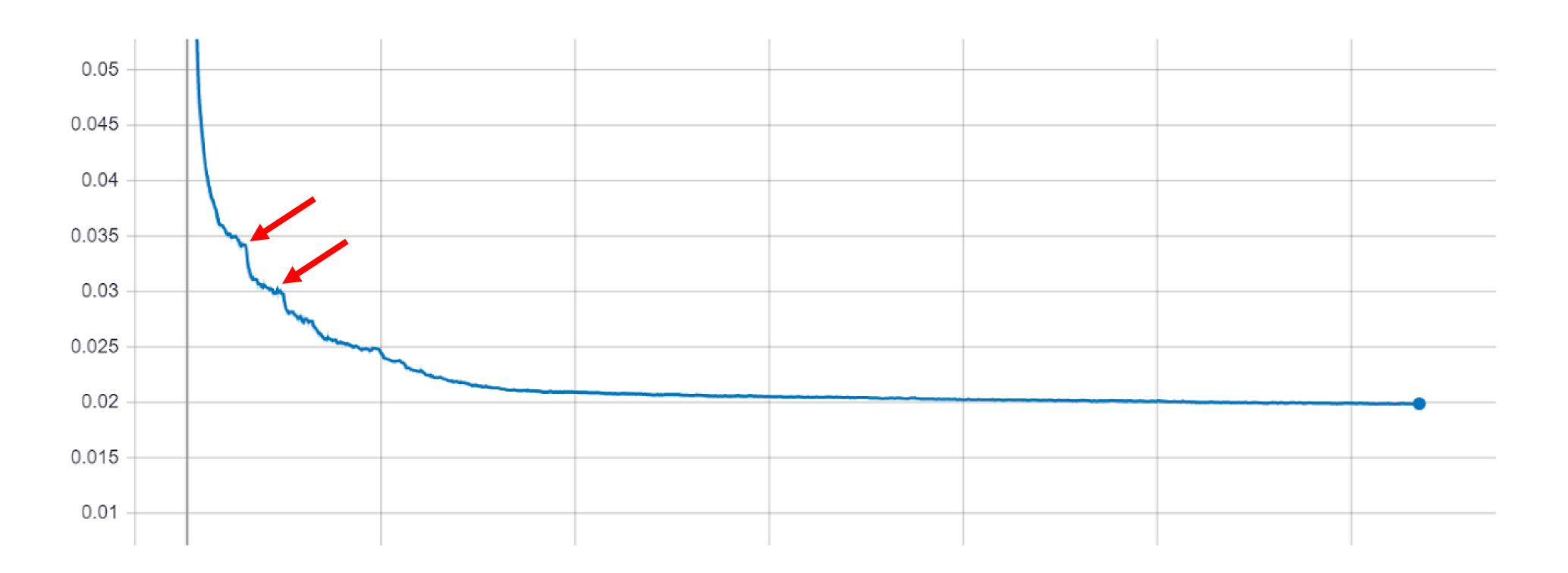
	1 1	1 1			
1	1 1 1	1 1 1 1 1 2 2 1 4 2 1		4.4	2 4 4
113 1 4	22 442 4	1 1 1	2 1	1 1 1 1 1 1 1 1 3	2 1 1
2 3 2 1 6 1 3 1	22 142 1				
1100000	1 4 4 1 1 2 3 1		3 4 2 2 3 3 1 1	2 1 2 3 2 1 4 4	1 1 4 1 2 2 2 3
	2 1 1		1 1 2 1 🗷 🗆 🗆 🗆 🗆 🗆	5 0 0 0	2 1 1 0
3 1 0 0	1200000		11380808880	111	3 2
15000000	1138080888	113	13180088808	11000 🛮 000 🖜	111
2 2 2	7	7	1400000	1100000000	1 1 2 1
110000 - 00	3 1	3 1	41	2 2 0 0	3 0
11108008008	4 1 1	4 1 1	2 3 1	13000 - 0	1400000
4 0 • • • • 0 0 0	11000000000	1108000800	3 2	1 3 2	2 4 0
2 1 0	12				
			2 2 1 0 8 8 0 8 8 0 8		
	1		2		
1	1 31 11	1 1	2 1 1 1 1	2 1 2 2	1 1
2 2 1 2 1 4	2 3 1 1 2 2 1 1	1 1 5 1131 1	2 1 11 1 1111 343	2 1 2 2 1 1 4 4 1 3	1 1 1 3 1 3 2 2 1
2 2 1 2 1 4 3 6 1 2 1 3 1 1	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2	1 3 2 2 1 3 6 1	2 1 11 1 1 1111 343 24112122	6 1 3 3 1 1 2 1	3 1 2 2 1 1 3 2
2 2 1 2 1 4	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2	1 1 5 1 1 3 1 1 1 1 1 3 2 2 1 3 6 1 1 1 2 1 1 0 0 0 1 1 0 1	2 1 11 1 1 1111 343 24112122 320 88 800 0	6 1 3 3 1 1 2 1	1210 0 0 0 0 0
2 2 1 2 1 4 3 6 1 2 1 3 1 1	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 • • • • • • •	1 3 2 2 1 3 6 1	2 4 1 1 2 1 2 2		
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 \square \square \square \square \square \square	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 • • • • • • •	1 3 2 2 1 3 6 1 1 2 1 8 0 0 0 8 8 0 8 1 2 1 8 0 0 8 8 0 8 0	2 4 1 1 2 1 2 2 3 2 3 5 5 6 7		1210 0 0 0 0 0
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0	1 3 2 2 1 3 6 1 1 2 1 8 0 0 0 8 8 0 8 1 2 1 8 0 0 8 8 0 8 0 3 1 1 8 8 8 0 8 0 8 0	2 4 1 1 2 1 2 2 3 2 0 8 8 8 0 0 8 8 1 1 1 8 0 8 0 0 0 8 0		
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 3 2 0 0 0 0 0 0 0 0 4 1 0 0 0 0 0 0 0	1 3 2 2 1 3 6 1 1 2 1 8 0 0 0 8 8 0 8 1 2 1 8 0 0 8 8 0 8 0 3 1 1 8 8 8 0 8 0 8 0	2 4 1 1 2 1 2 2 3 2 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0		
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 4 1 0 0 0 0 0 0 0 0	1 3 2 2 1 3 6 1 1 2 1 8 0 0 0 8 0 0 8 1 2 1 8 0 0 0 8 0 0 0 3 1 1 8 8 0 0 0 0 0 0 0 2 1 2 8 8 0 0 0 0 8 0	2 4 1 1 2 1 2 2 3 2 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 0		
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 4 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0	1 3 2 2 1 3 6 1 1 2 1 8 0 0 0 8 0 0 8 1 2 1 8 0 0 0 8 0 0 0 3 1 1 8 8 0 0 0 0 0 0 0 2 1 2 8 0 0 0 0 0 0 0 2 2 2 0 0 0 0 0 0 0 0	2 4 1 1 2 1 2 2 3 2 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 0 2 2 1 0 0 0 0 0 0 0 0		1 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 1 2 1 4 3 6 1 2 1 3 1 1 2 1 0 0 0 0 0 0 0 2 1 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0 0 0 2 1 2 0 0 0 0 0 0 0 0 0	2 3 1 1 2 2 1 1 1 3 1 1 3 5 1 2 3 3 0 0 0 0 0 0 0 0 4 1 0 0 0 0 0 0 0 0 2 2 1 0 0 0 0 0 0 0	1 3 2 2 1 3 6 1 1 2 1	2 4 1 1 2 1 2 2 3 2 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 0	6 1 3 3 1 1 2 1 4 1 0 0 0 0 0 0 0 0 1 3 1 0 0 0 0 0 0 0 0 2 3 1 0 0 0 0 0 0 0 0 1 2 0 0 0 0 0 0 0	1 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

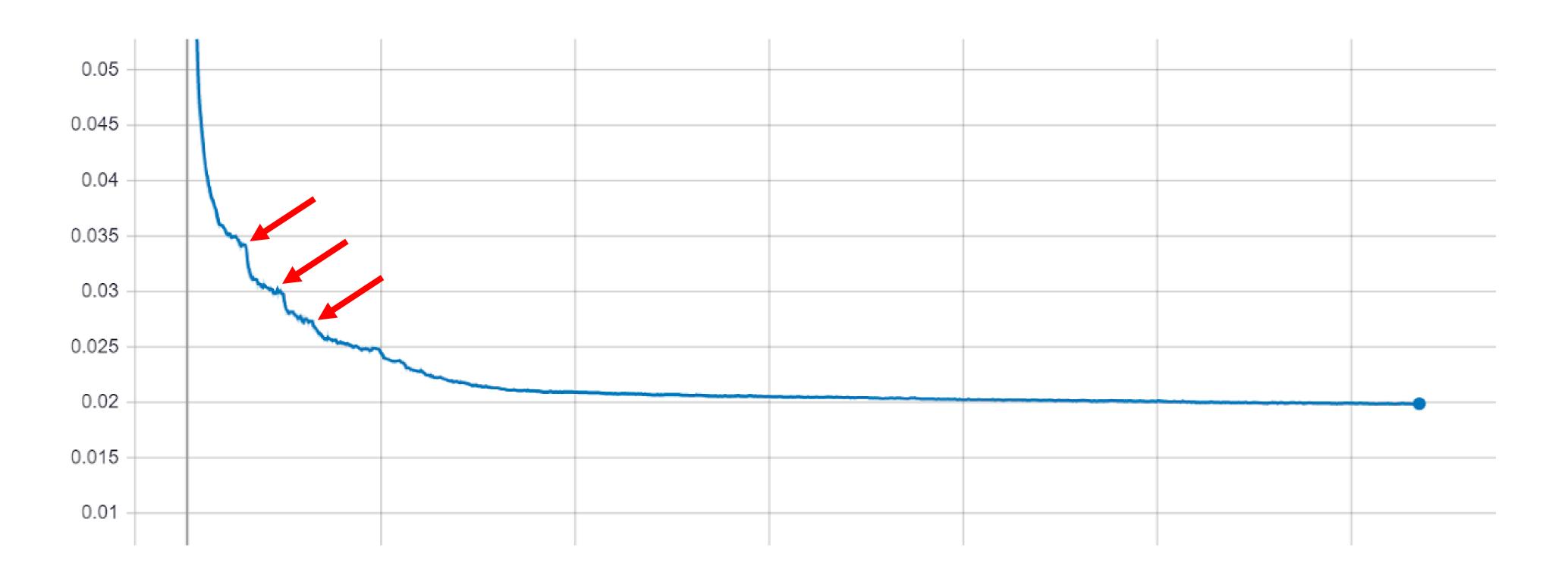
Adatgenerálás

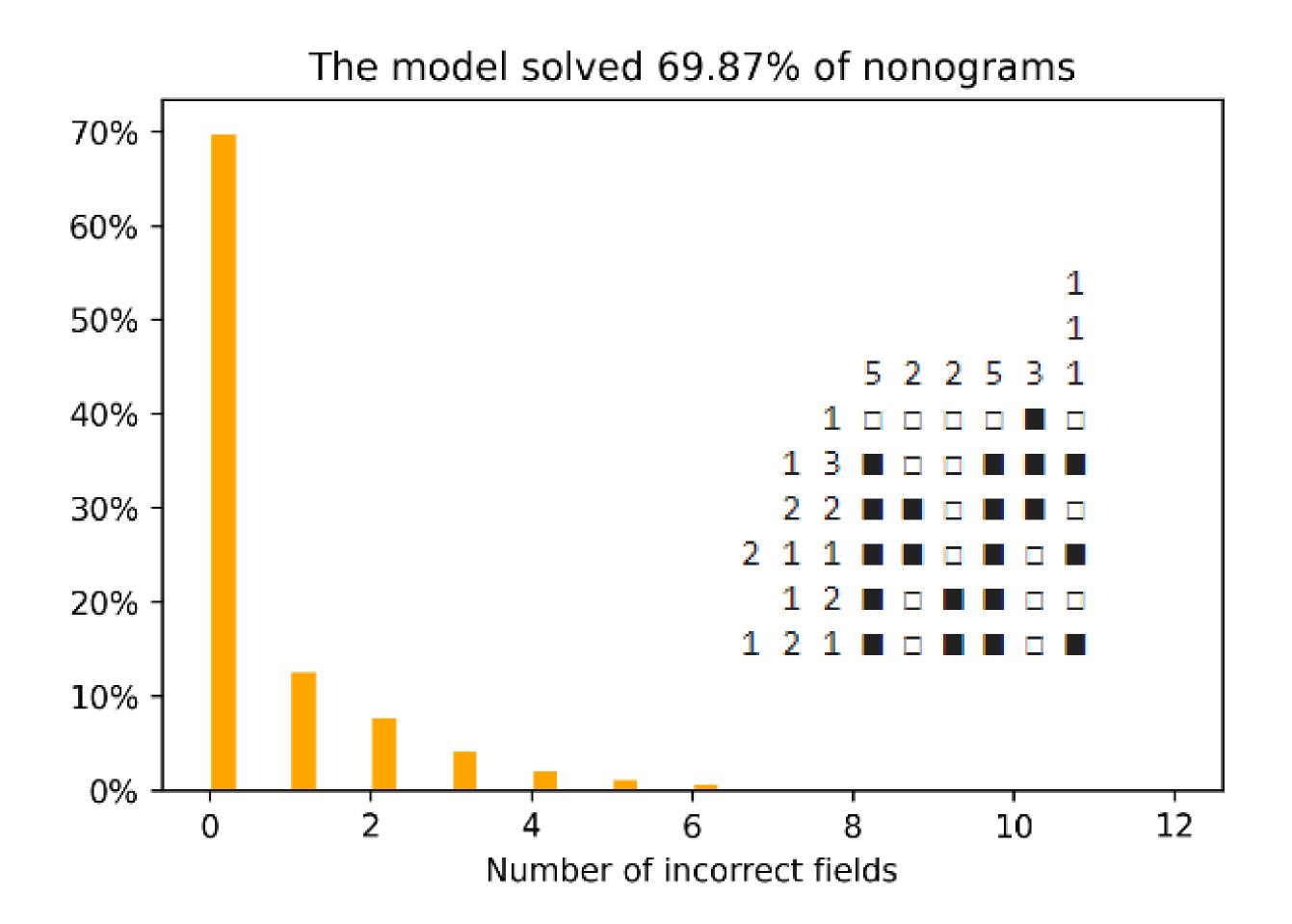
							1 1	1 1			
1	1 1	1 1				1	1 1 1	1 1 1	2 1	1 1	2 11
113 1 4	22 142 1	22 142 1	2 1	11 11113	2 1 1 3 1 2 1 1 1 2 3	113 1 4	2 2 1 4 2 1	2 2 1 4 2 1	3 2 1 4 4 1 4	1 111113	3 1 2 1 1 1 2 3
2 3 2 1 6 1 3 1	1 4 4 1 1 2 3 1		2 2 2 1 1 2 1	2 1 2 3 2 1 4 4	11412223		1 4 4 1 1 2 3 1	1 4 4 1 1 2 3 1	3 4 2 2 3 3 1 1	2 1 2 3 2 1 4 4	1 1 4 1 2 2 2 3
1100000000	2 1 1 \blacksquare \blacksquare \square \square \blacksquare \square \square			5 0 0 0					1 1 2 1 8 0 8 0 8 8 0 8	5 0 0 0 0 0 0 0 0	2 1 1 0
3 1 0 0 0 0 0 0	120 🗷 0 🗷 🗷 0 0 0	1 2 0 🗷 0 🗷 🗷 0 0 0		$1\ 1\ 1\ \blacksquare\ \square\ \blacksquare\ \square\ \square\ \square\ \square\ \blacksquare$	3 2	158008888				11180800008	
		113		11000 0000	111	2 2 2	7	7			
110000	7	7 - 					3 1 ■ ■ ■ □ □ □ ■ □	3 1	4 1	2 2 0 0 0 0 0 0	3
1110 - 00 - 00 -		411			1400	1110000000		$4\ 1\ 1\blacksquare\blacksquare\blacksquare\blacksquare\square\blacksquare\square\blacksquare$	2 3 1	13000	1 4 0 0 8 0 8 8 8 8
4	1108000800				2 4 0		110		3 2		2 4 🗆 🗷 🗆 🗷 🗷 🗷
2 1 0	12 • • • • • • • •	12 🗷 🗆 🗆 🗷 🗆 🗆		$1\ 1\ 1\ 1\ \blacksquare\ \square\ \blacksquare\ \square\ \blacksquare\ \square\ \blacksquare$			12 • • • • • • •	12	2 2 1 0	1111	111
	1		2			4	1		2	2.4 2 2	4 4
1	1 31 11	1 1	1 1 1 1	2 1 2 2		2 2 1 2 1 4	1 3 1 1 1 1 2 3 1 1 2 2 1 1	1 1	1 11 1	21 2 2 1 1 4 4 1 3	1 1 131322 1
2 2 1 2 1 4	2 3 1 1 2 2 1 1	5 1131 1		11441 3	1 3 1 3 2 2 1	3 6 1 2 1 3 1 1	13113512	5 1131 1 13221361	1 1 1 1 3 4 3 2 4 1 1 2 1 2 2	6 1 3 3 1 1 2 1	3 1 2 2 1 1 3 2
3 6 1 2 1 3 1 1 2 1 0 8 8 0 0 0 8 0	1 3 1 1 3 5 1 2			61331121	3 1 2 2 1 1 3 2 1 2 1 0 0 0 0 0 0 0	2 1 0		121	3 2 0 0 0 0 0 0		1 2 1 0 8 0 0 8 8 0 8
				13180088808	5	2 1 1 0 0 0 0		1 2 1 • • • • • • •	1110000000	1 3 1	5
2 3 1		311			1130			3 1 1 • • • • • • •	3 0 0 0 0 0		1130 0 0 0 0 0 0 0
2 1 2				2 3 1	6 🗆 🗆 🗷 🗷 🗷 🗷			2 1 2	2 3 0 0 0 0 0 0 0		
11100000000	2 2 1			1 2	1000000000			2 4		4 2	12
	6			4 2		1 1 4 8 0 8 0 8 8 8	2 2 1	2 2 0 0	2 2		1 2 1
						1 0 0 0 0 0 0 0	13000000		1 2 1 1	5 1 0	2 🔳 🖶 🗆 🗆 🗆 🗆
1 - 0 0 0 0 0 0 0	1 1	1 1		312 111 22			1 1	1 1			
1	1 1 1	1 1 1	2 1	1 1	2 11	113 1 4	1 1 1	1 1 1	2 1	1 1	2 11
113 1 4	2 2 1 4 2 1	2 2 1 4 2 1	3 2 1 4 4 1 4	1 111113	3 1 2 1 1 1 2 3	23216131	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	3 2 1 4 4 1 4	1 111113	3 1 2 1 1 1 2 3
2 3 2 1 6 1 3 1 1 1 0 0 0 0 0 0 0 0	1 4 4 1 1 2 3 1	1 4 4 1 1 2 3 1	3 4 2 2 3 3 1 1	2 1 2 3 2 1 4 4	1 1 4 1 2 2 2 3	1100000 - 0	211		3 4 2 2 3 3 1 1 1 1 2 1 8 0 8 0 8 0 8	2 1 2 3 2 1 4 4 5 0 0 0 0 0 0 0 0	1 1 4 1 2 2 2 3 2 1 1 0 0 0 0 0 0 0 0 0
3 1 0 0 0 0 0 0 0 0			1 1 2 1			3 1 0 0 8 8 8 0 8 0	1 2 0 • 0 • • 0 0 0	1 2 0 • 0 • • 0 0 0	113	111000000	3 2
15		1 2 0 8 0 8 8 0 0 0	1 1 3 	1110000000	3 2 8 8 8 0 0 0 8 8 1 1 1 8 0 0 8 0 0 8	150000000	113		1 3 1	11000 - 000	
2 2 2	7 🗆 🗷 🗷 🗷 🗷	7		110000000000	1 2 1 0 0 0 0 0 0 0		7	7	1 4 0 0 8 0 8 8 8	11000000001	
110000	3 1	3 1 • • • • • • •	4 1 • • • • • • •	2 2 0 0 8 8 0 0 8 8			3 1 	3188800080	41		3 0 8 8 8 0 0 0 0
	411		2 3 1	13000000	1 4 0 0 8 0 8 8 8	4	110 000 00	1100000000		1 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	2 4 0 0 0 0 0 0 0 0 0
				132	2 4 0	2 1 0	12	1 2		111	
					111		1		2		
1	1 31 11		2	2 1 2 2	1 1	1	1 3 1 1 1	1 1	1 1 1 1	21 2 2	1 1
2 2 1214	1 31 11 2311	5 1131 1	1 11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	11441 3	131322 1	2 2 1 2 1 4	2 3 1 1 2 2 1 1	5 1131 1	1111 343	1 1 4 4 1 3	1 3 1 3 2 2 1
3 6 1 2 1 3 1 1	1 3 1 1 3 5 1 2	13221361	2 4 1 1 2 1 2 2	6 1 3 3 1 1 2 1	3 1 2 2 1 1 3 2	3 6 1 2 1 3 1 1	1 3 1 1 3 5 1 2	1 3 2 2 1 3 6 1	2 4 1 1 2 1 2 2	6 1 3 3 1 1 2 1	3 1 2 2 1 1 3 2
2 1 0 0 0 0 0 0		1 2 1 • • • • • • •	3 2 0		1 2 1 0 8 0 0 8 8 0 8		3 3 0 8 8 8 0 8 8 8		3 2 0 8 8 8 0 0 8 8		5
		$1\ 2\ 1\ \blacksquare\ \square\ \square\ \blacksquare\ \square\ \square\ \square$	1118080080	1 3 1 8 0 0 8 8 8 0 8	5	2 3 1					1130
		3 1 1	3 0 0 0 0 0 0			2 1 2	1180000800		2 3 0	2 3 1	6
		2 1 2		12 0 0 0 0 0	100000000	1110 - 000 - 0	2 2 1	2 4	2 2 1	12 0 0 0	1000000000
2 1 • • • • • •			4108888080	4 2	12 • • • • • • • •			2 2 0 0	410	4 2	12
1 1 4	2 2 1	2 2 0 0	2 2 • • • • • • •		1 2 1 • • • • • • • •						
1 • 0 0 0 0 0 0	1300 - 00 - 0	1 0 0 0 0 0 0 0 0 0	1 2 1 1 8 0 8 8 0 8 0 8	5 1 0	2 🖩 🖷 🗆 🗆 🗆 🗆			1 - 0 0 0 0 0 0			

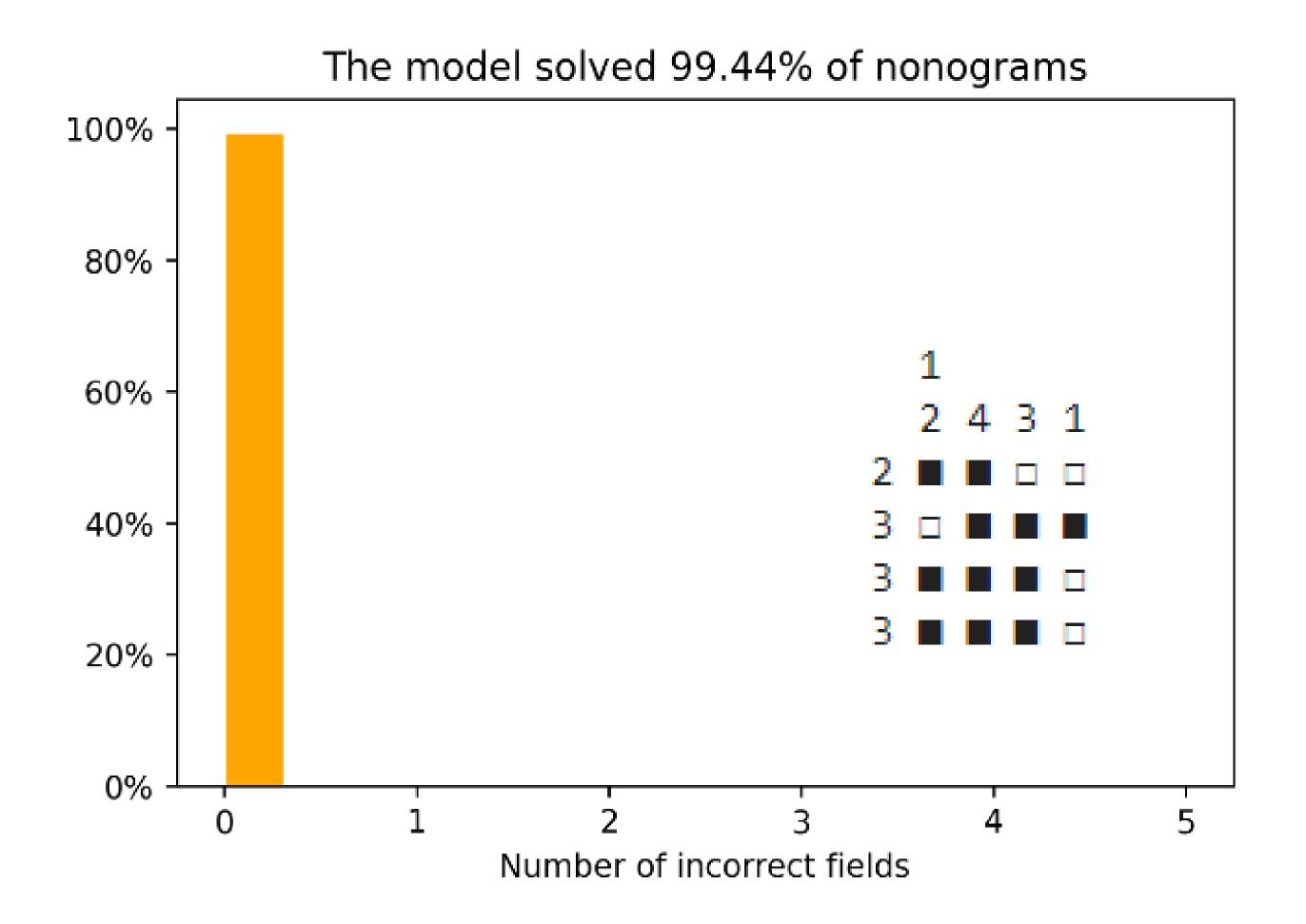


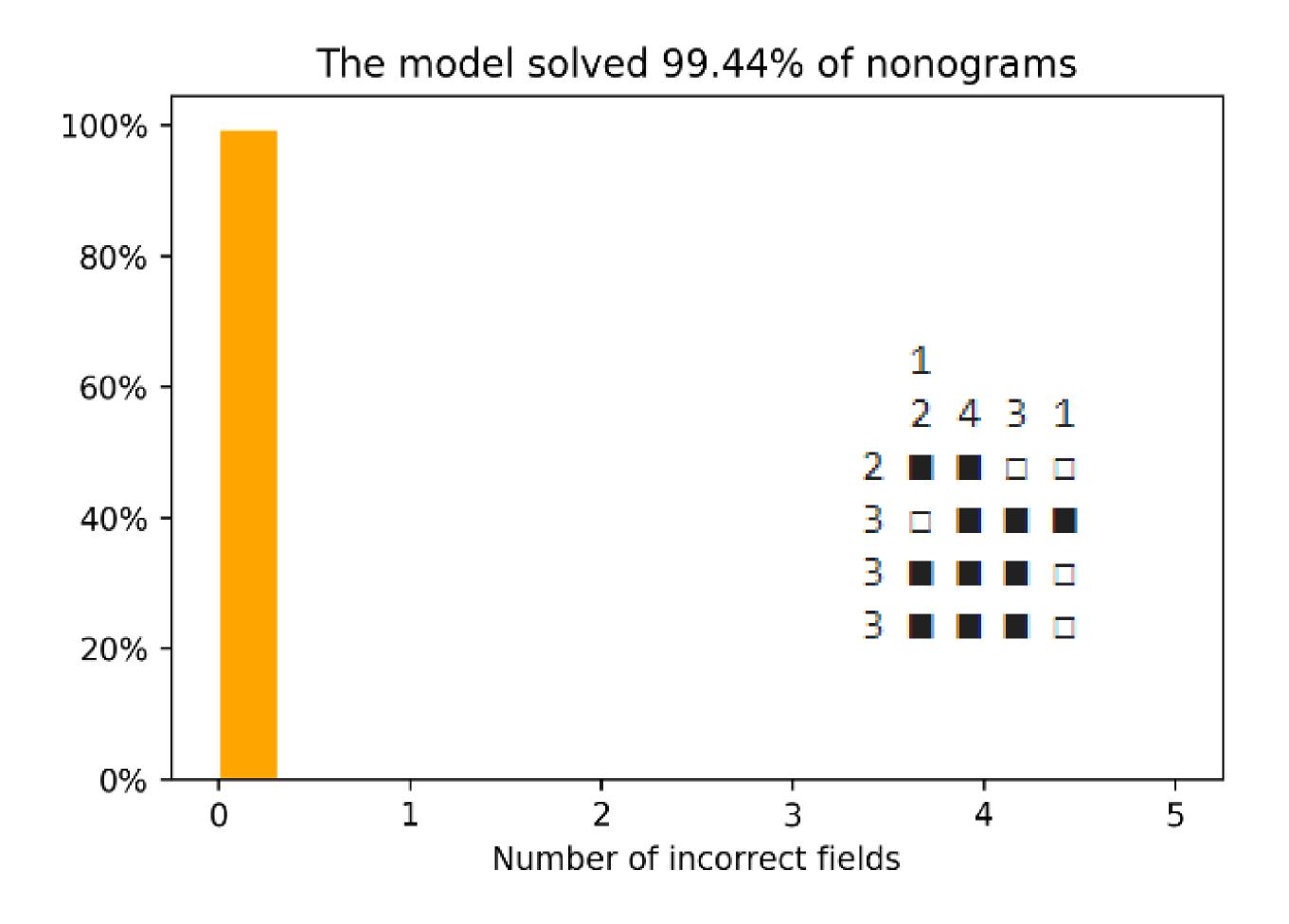












$$2^{(4*4)} = 65536$$

Deployment

TensorFlow.js is a library for machine learning in JavaScript

Develop ML models in JavaScript, and use ML directly in the browser or in Node.js.

See tutorials

Tutorials show you how to use TensorFlow.js with complete, end-to-end examples.

See models

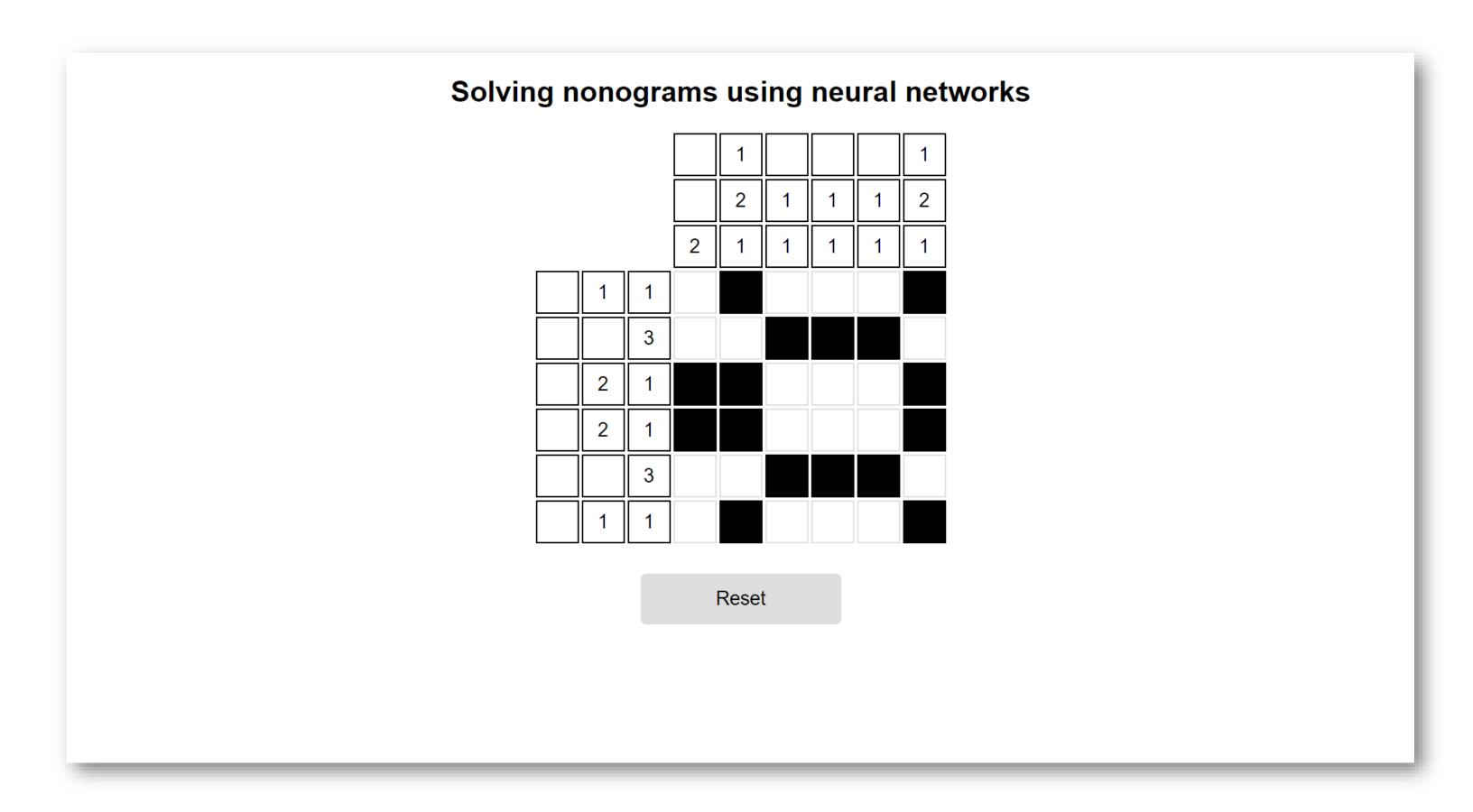
Pre-trained, out-of-the-box models for common use cases.

See demos

Live demos and examples run in your browser using TensorFlow.js.



Demo



https://nonogram-solver.web.app/

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					

					2		1
			2	2	1	5	4
	2	1					
		3					
2	1	1					
1	2	1					
	1	1					
		1					