Prevention of occurrence of dead units in self-organizing maps

Student: Bc. Jakub Novák
Supervisor: doc. RNDr. Martin Takáč, PhD.

Contents

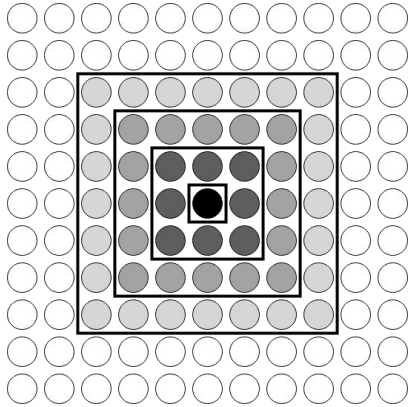
Goals of the thesis

- review existing approaches
- suggest SOM modifications
- implement modifications
- analyze to what extent the proposed modifications succeed in eliminating dead units

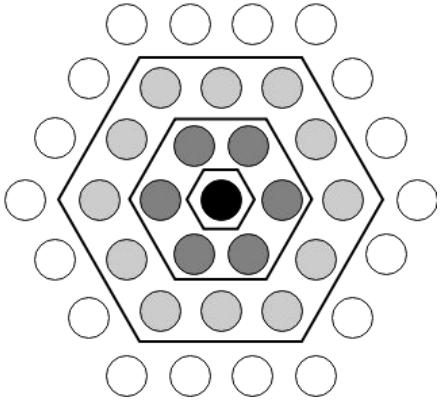
Overview of self-organizing map algorithm

Self-organizing map

- model of artificial neural network trained using unsupervised learning
- preserves topology
- two-dimensional representation of the input space
- neighbourhood function
  - adapts neighbourhood of winning neuron
  - neighbourhood shrinks with time



(1) Rectangular neighbourhood

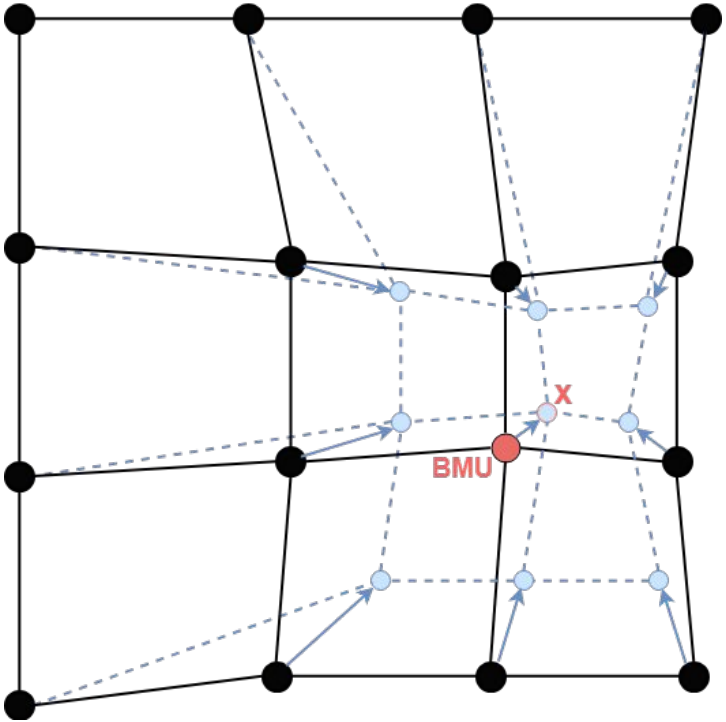


(2) Hexagonal neighbourhood

Self-organizing map

- SOM algorithm has 2 steps
  - competition
  - learning
- competition
  - neurons compete, which one has weights the closest to the input
- learning
  - neurons adapt weights within the neighbourhood
  - SOM parameters update (learning rate, neighbourhood size, ...)

Self-organizing map



(3) SOM update

Dead units

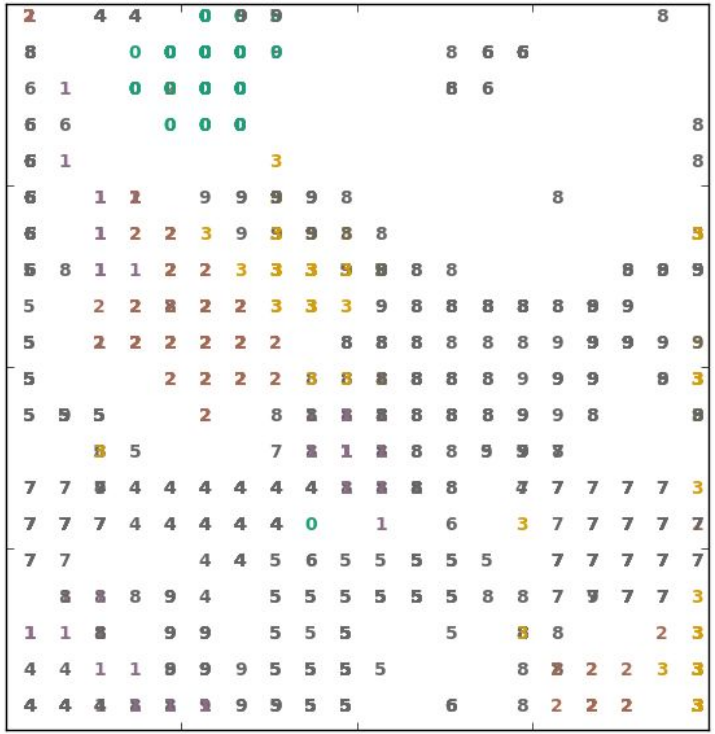
Dead units

- neurons remember information about available data in their weights
- small map forces neurons to represent more different data
- bigger map can represent data more precisely
  - unused neurons, called dead units, might occur

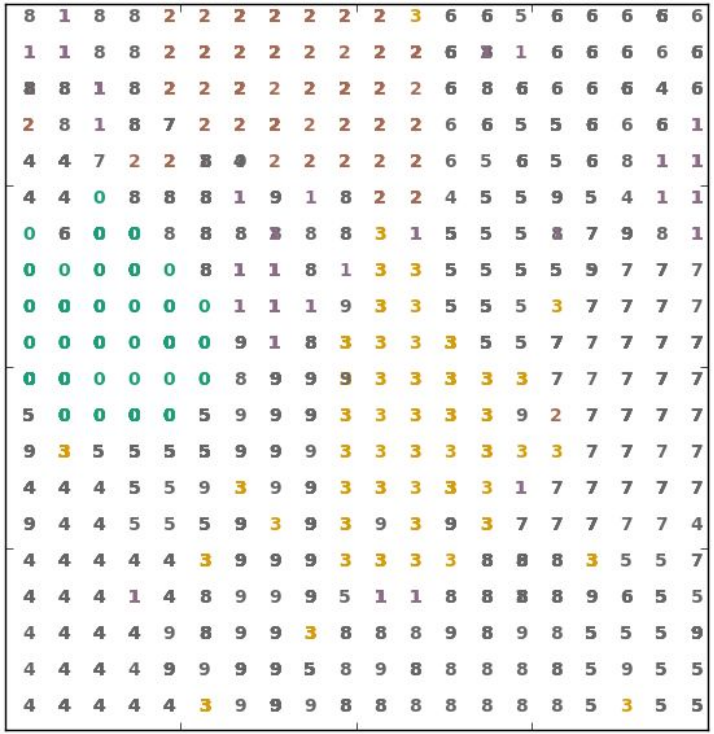
Dead units

- common
- as a result, network capacity is not being fully utilized
- usually caused by badly initialized weights in the SOM
  - some neurons have weights far from any data
- rich get richer
- losers adapt too little

Dead units problem



(4) 20x20 SOM with dead units (MNIST dataset)



(5) 20x20 SOM without dead units (MNIST dataset)

Experiments

General setup for experiments

- 3 different SOM (map) **sizes**
- 3 **datasets** representing 3 different data designs
- 3 **methods** of handling dead units
- each ran for 30 **epochs**
- using Gaussian neighborhood function
- using various parameter combinations

Methods

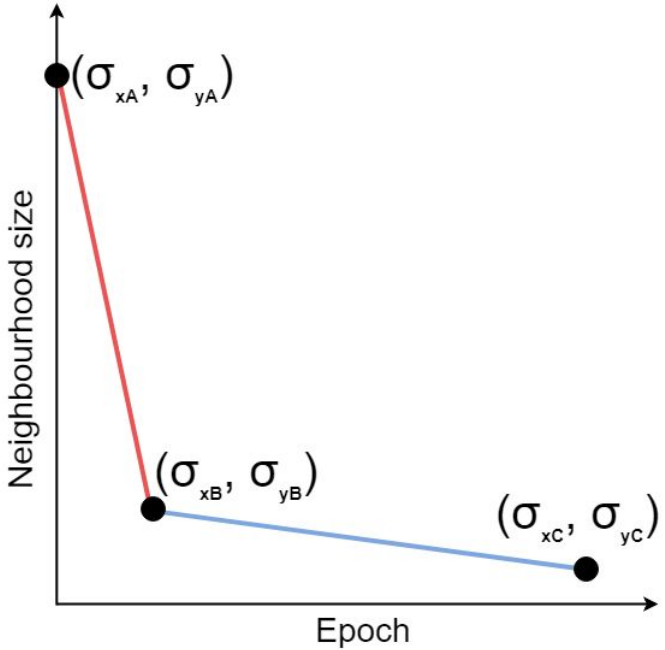
1. standard self-organizing map
2. training random dead unit
3. training dead units for novel inputs

Method 1:
Standard self-organizing map

Standard self-organizing map

- standard approach
- two phases
  - initial organization phase
    - start off with big neighbourhood size and quickly (within few epochs) get to small neighbourhood size
  - fine-tuning phase
    - fine-tuning neuron weights

  - (xA, yA) - starting point
    (xB, yB) - breaking point
    (xC, yC) - end point



(6) Learning rate annealing

Standard self-organizing map

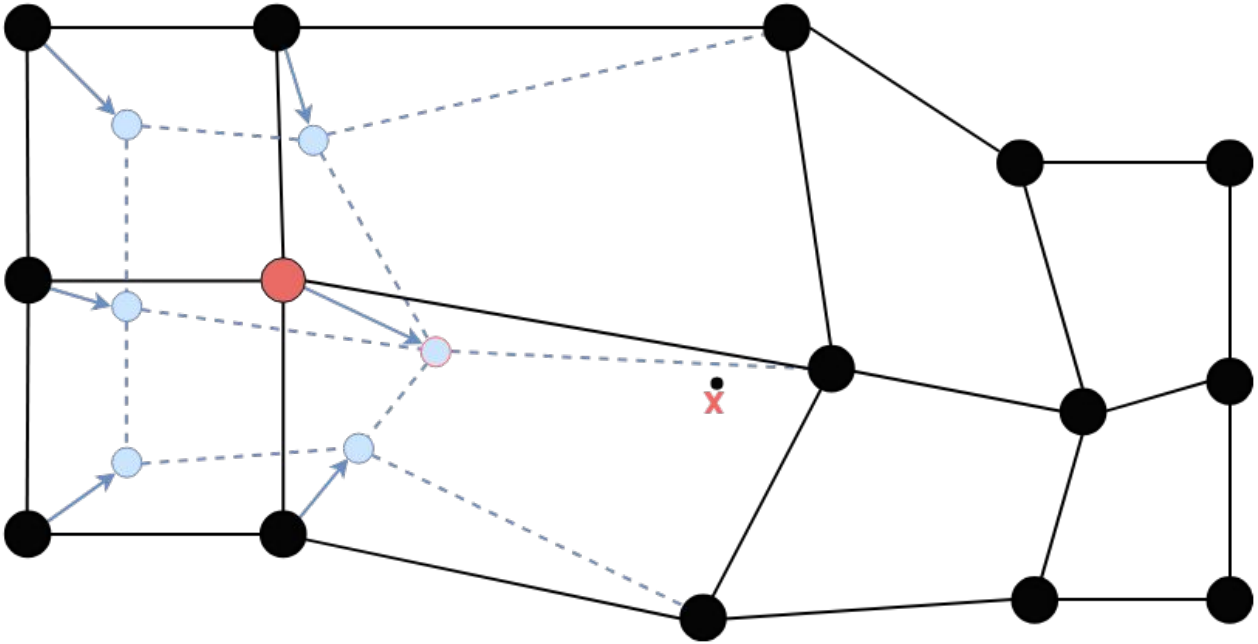
- explored all combinations for
  - breaking points at 1st, 3rd, 7th and 12th epoch
  - breaking point learning rates: 0.1, 0.2, 0.5
  - initial neighbourhood sizes: 2, 5, 10, 20, 50, 100
- hypothesis
  - **starting with larger neighbourhood sizes can help minimize the number of dead units**

Method 2:
Training random dead units

Training random dead unit

- extra step after each epoch
- randomly choose dead unit
- find closest input sample
- adapt dead units weights and weights of its neighbourhood with this input sample

Training random dead unit



(7) Update of randomly chosen dead unit (red dot)

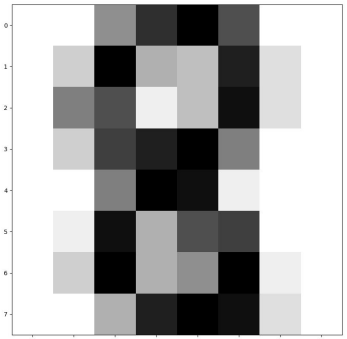
Training random dead unit

- hypothesis
  - **eliminate dead neurons by forcing them to adapt**

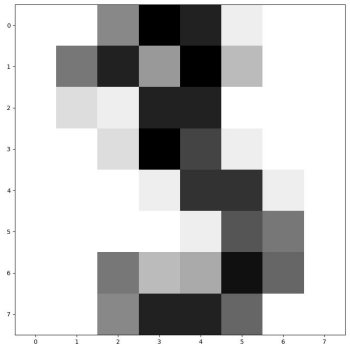
Method 3:
Training dead units for novel inputs

Training dead units for novel inputs

- are neurons weights similar enough to the input datum?
- threshold variable
  - some samples are not clearly separable



(8) Digit eight



(9) Digit three

(10) Letter dha

(11) Letter ayb

Training dead units for novel inputs

- choosing threshold
  - calculate mean Euclidean distance for inputs of the same category
  - calculate cross-category Euclidean distances

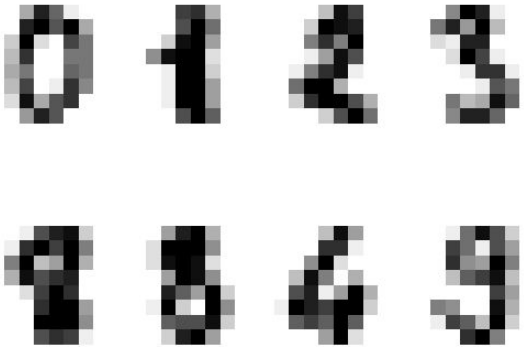
- hypothesis
  - **potential better clusterization**
  - **novelty inputs mapped to unused neurons**

Datasets and methods comparison

Datasets

- scikit-learn MNIST dataset
  - 1797 handwritten digits
  - 8x8 pixels
  - **multidimensional input with 10 classes and relatively uniform distribution of classes**
- Raster
  - 9801 digits from range (0, 1) with a step of 0.01
  - subset of 2000 samples used to train
  - **low-dimensional input**
- OMNIGLOT
  - 1623 different handwritten characters from 50 different alphabets
  - 105x105 pixels
  - subset of 600 samples used to train
  - **multidimensional input with many classes having few samples per class**

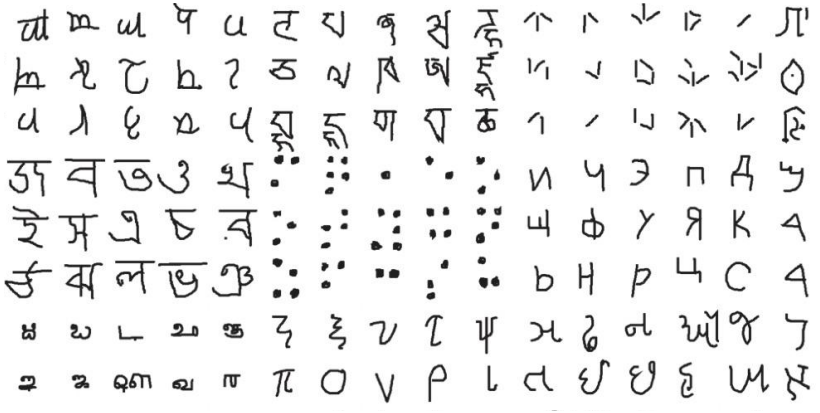
Datasets



(12) MNIST dataset

```
[0.07, 0.09],
[0.03, 0.14],
[0.21, 0.25],
...,
[0.51, 0.74],
[0.42, 0.46],
[0.54, 0.31]]
```

(13) Raster dataset



(14) OMNIGLOT dataset

Methods comparison

- computational restrictions
- one run for each parameter combination
  - 1350 combinations
- cannot give statistically clear answer
  - more runs needed
- exploratory analysis

MNIST

Standard SOM

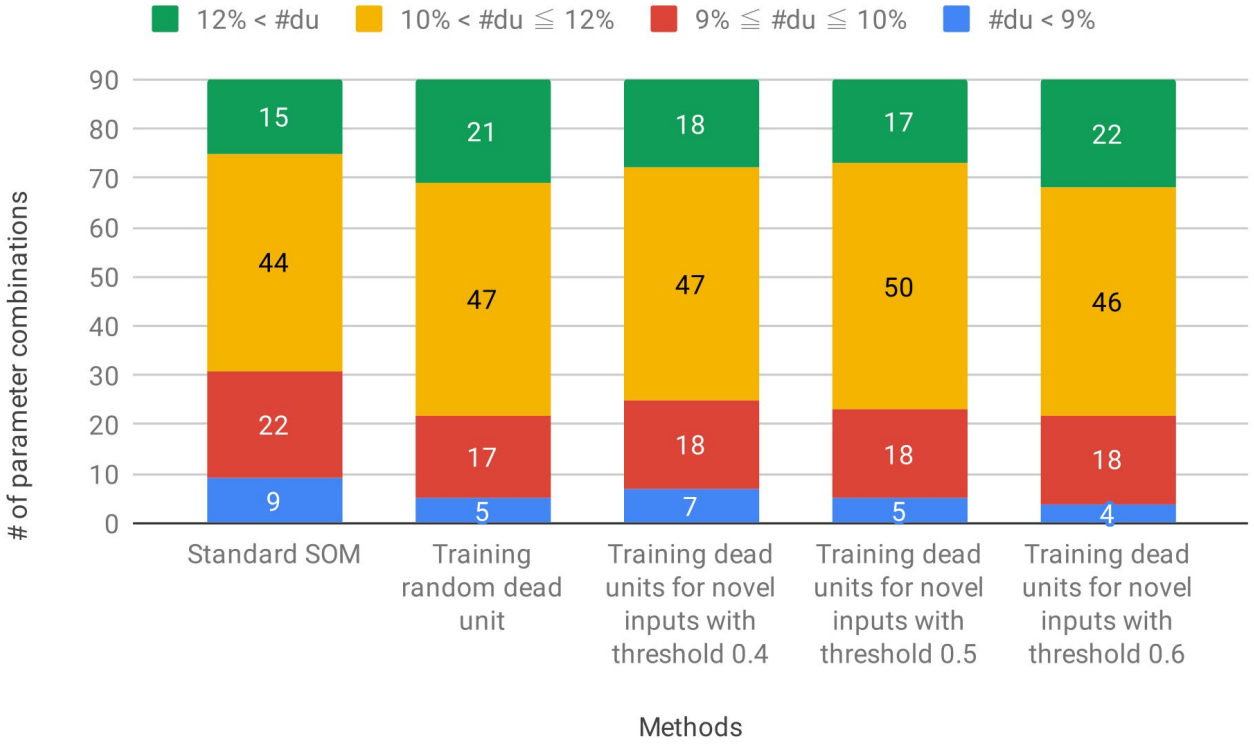
Breaking point alpha: 0.1	Initial sigma					
Breaking point	2.0	5.0	10.0	20.0	50.0	100.0
1	13.50%	11.50%	10.50%	11.75%	12.25%	10.50%
3	12.25%	10.75%	12.25%	12.75%	11.50%	11.75%
7	10.30%	11.30%	9.50%	10.00%	11.50%	12.80%
12	8.75%	8.25%	11.75%	9.75%	10.00%	10.25%
20	11.25%	10.50%	11.00%	10.50%	11.25%	11.25%

Breaking point alpha: 0.2	Initial sigma					
Breaking point	2.0	5.0	10.0	20.0	50.0	100.0
1	13.00%	11.25%	9.50%	12.75%	12.00%	11.75%
3	10.50%	9.00%	13.50%	7.50%	13.25%	11.75%
7	10.50%	13.00%	7.00%	10.50%	11.30%	10.50%
12	12.25%	8.75%	9.50%	11.25%	7.75%	9.00%
20	10.00%	10.50%	9.75%	10.00%	10.75%	9.75%

Breaking point alpha: 0.5	Initial sigma					
Breaking point	2.0	5.0	10.0	20.0	50.0	100.0
1	8.75%	9.50%	9.75%	11.25%	9.75%	9.25%
3	10.25%	8.50%	11.00%	9.25%	11.00%	11.25%
7	11.00%	12.00%	11.50%	12.50%	11.50%	10.50%
12	10.00%	8.50%	9.50%	10.00%	10.75%	10.00%
20	11.50%	12.00%	11.25%	9.75%	12.25%	12.50%

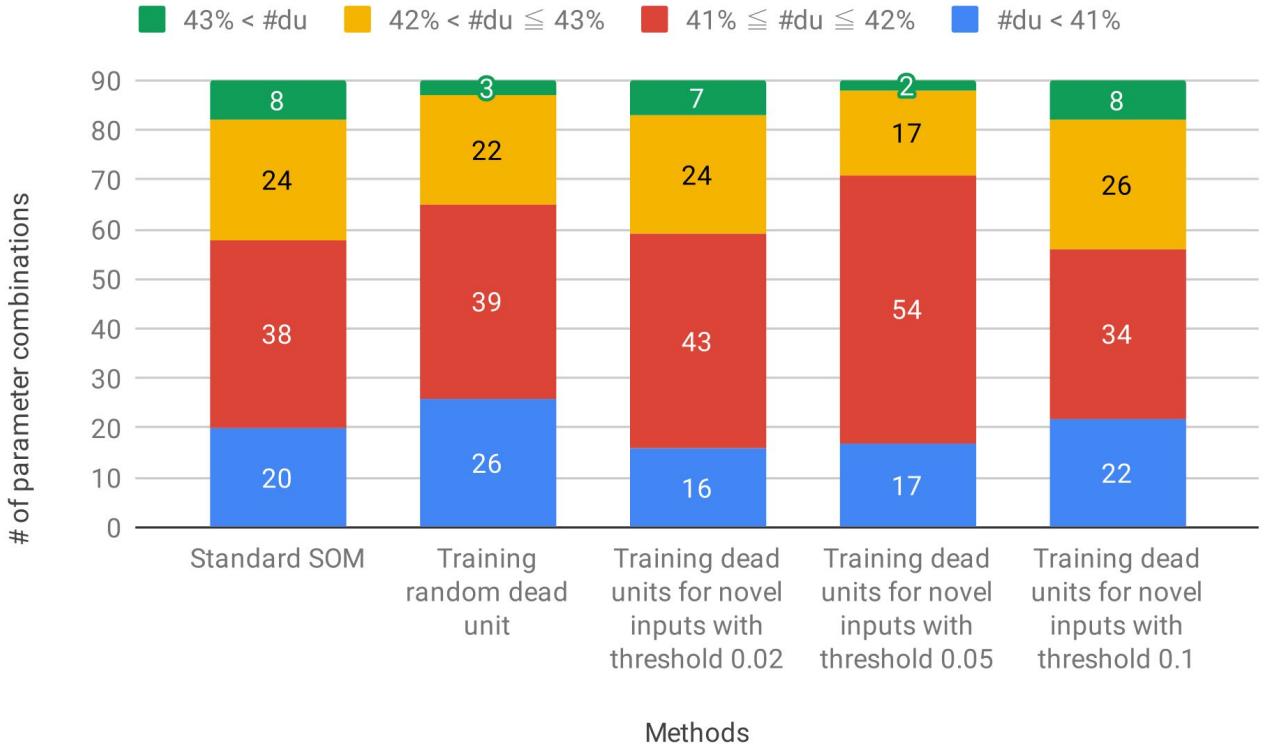
(15) Results of first method trained on MNIST dataset and map size 20x20

MNIST



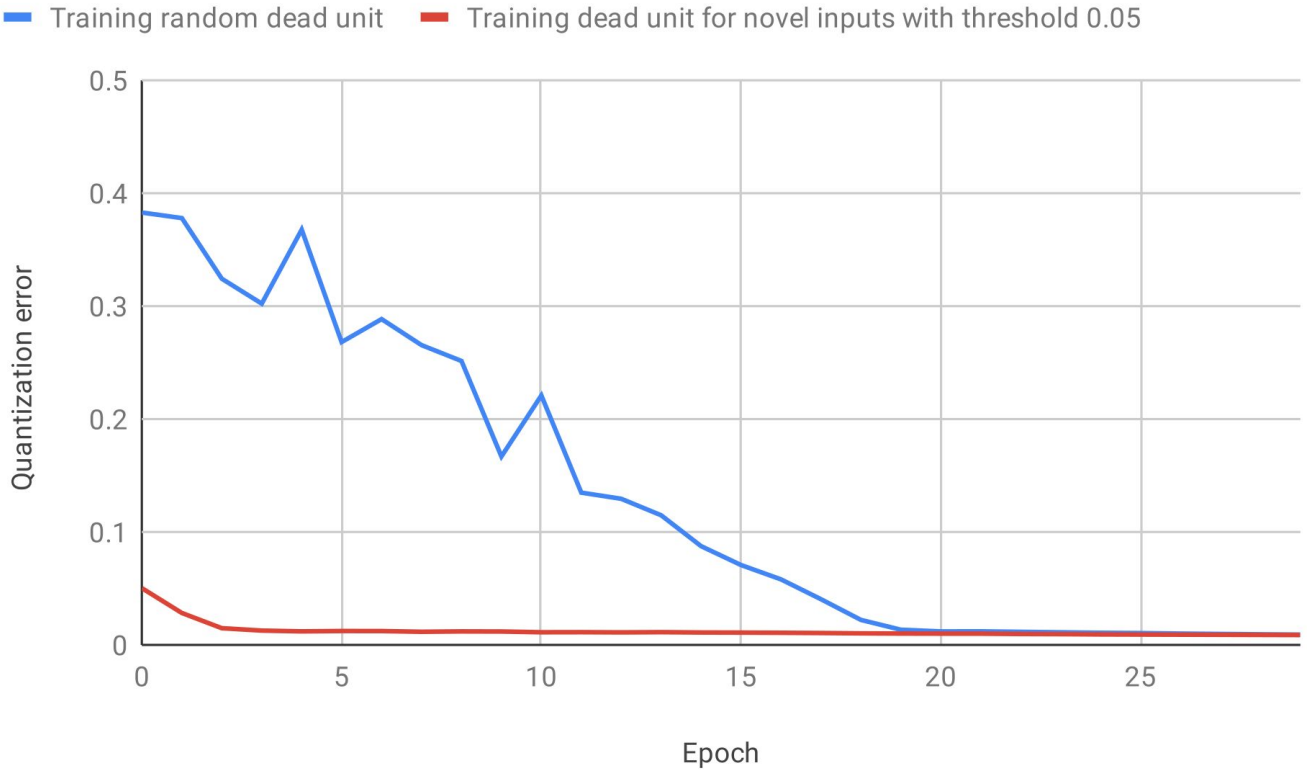
(16) 20x20 map - performance of each method in terms of the percentage of dead units (#du)

RASTER



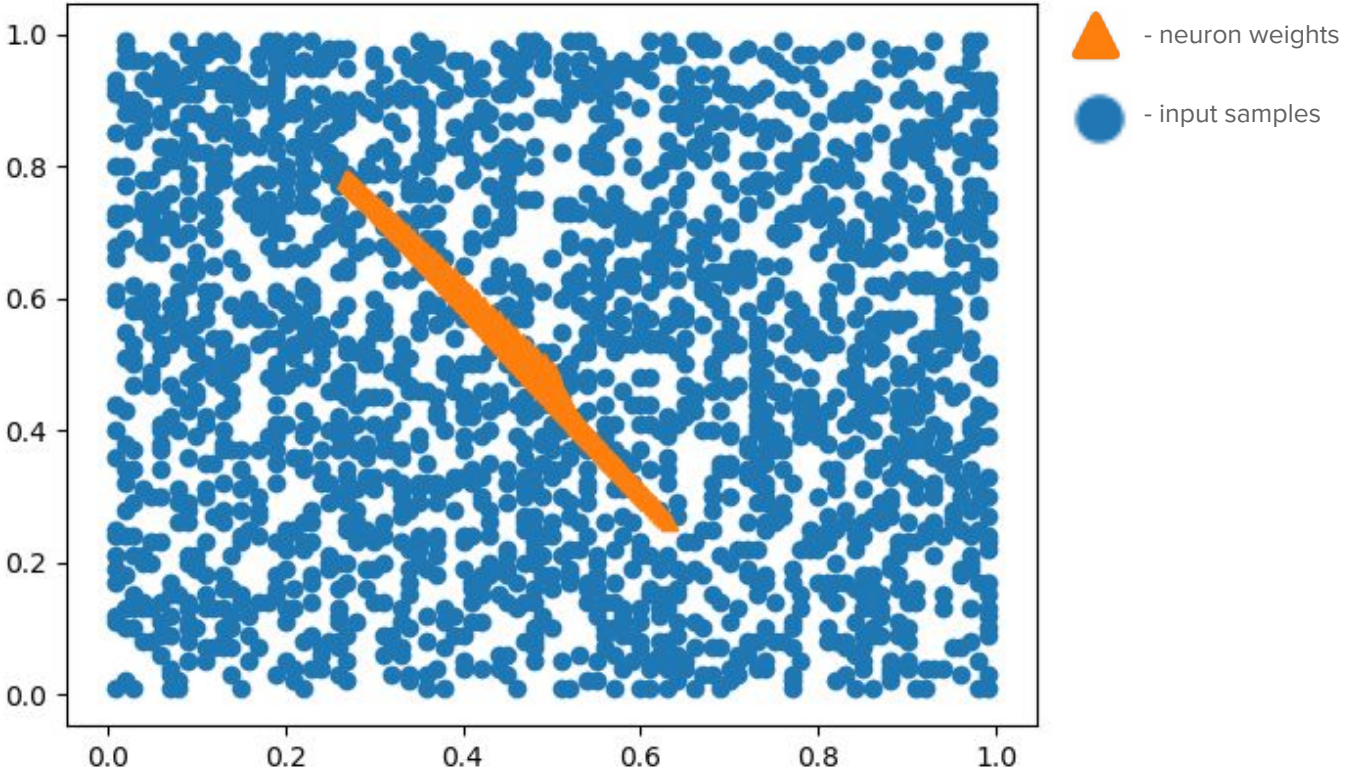
(17) 40x40 map - performance of each method in terms of the percentage of dead units (#du)

RASTER



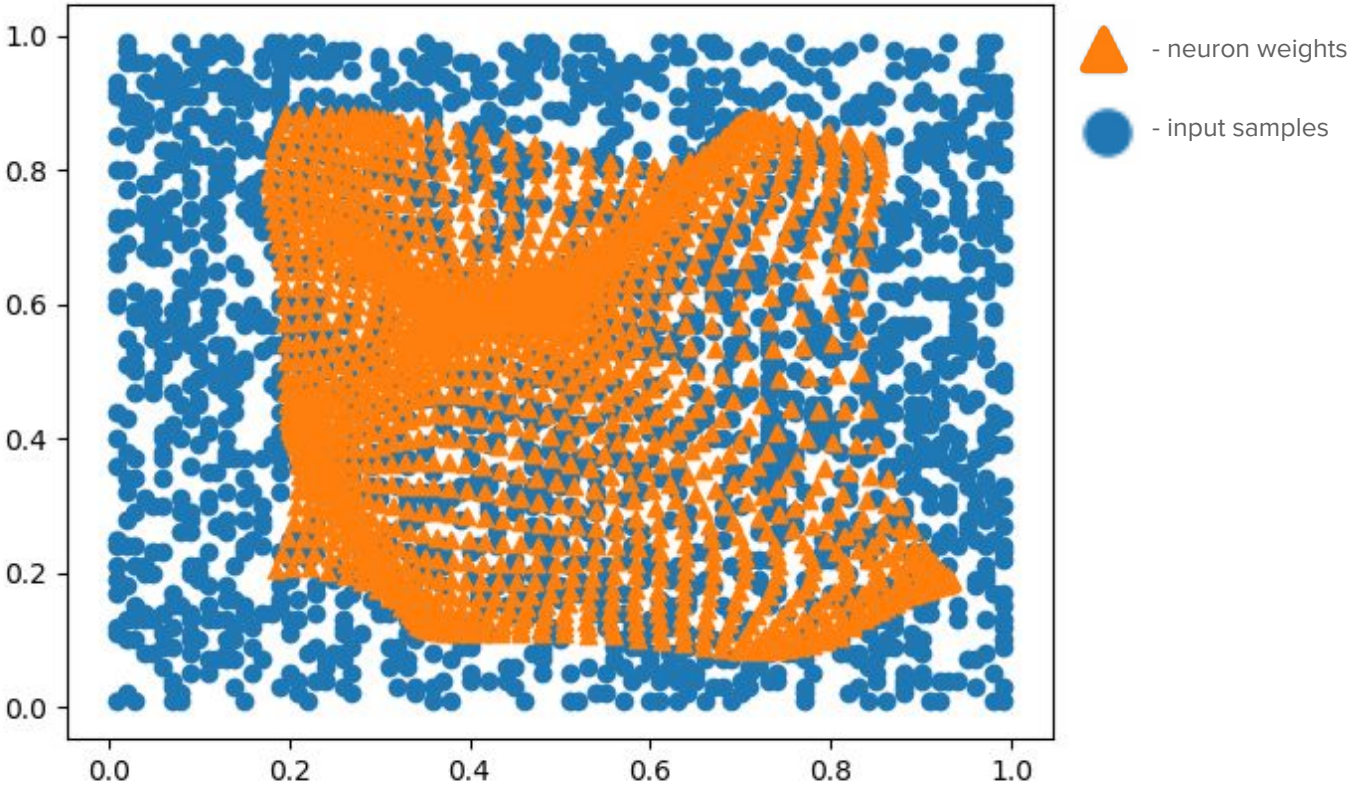
(18) 40x40 map - comparison of quantization errors for two of the best performing parameter combinations

RASTER



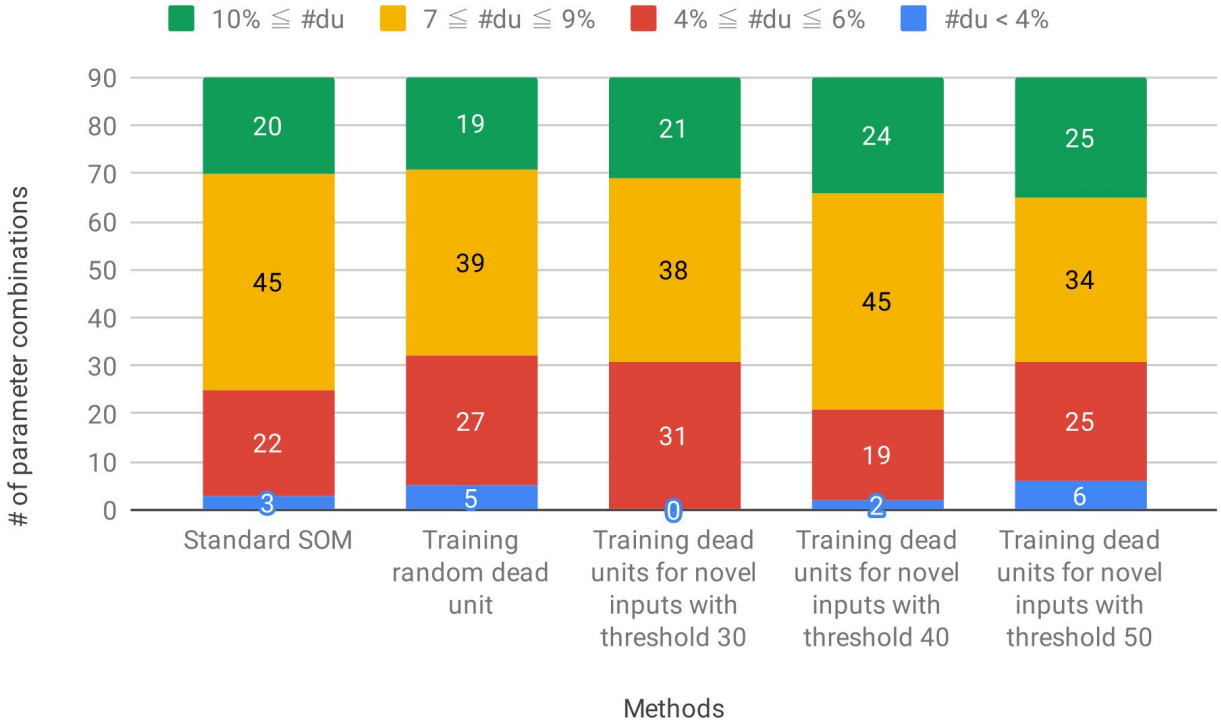
(19) 40x40 map - neuron weights at the 1st epoch of training random dead unit method

RASTER



(20) 40x40 - neuron weights at the 1st epoch of training dead units for novel inputs with threshold 0.05 method

OMNIGLOT



(21) 10x10 map - performance of each method in terms of the percentage of dead units (#du)

...compared to Raster

(22) 10x10 map - performance of each method in terms of the percentage of dead units (#du)

OMNIGLOT

Training dead units for novel inputs with threshold 50

Breaking point alpha: 0.1	Initial sigma					
Breaking point	2.0	5.0	10.0	20.0	50.0	100.0
1	5.00%	10.00%	3.00%	6.00%	9.00%	10.00%
3	7.00%	10.00%	13.00%	7.00%	6.00%	9.00%
7	9.00%	11.00%	3.00%	8.00%	7.00%	8.00%
12	10.00%	7.00%	16.00%	6.00%	6.00%	10.00%
20	7.00%	7.00%	13.00%	11.00%	6.00%	8.00%

Breaking point alpha: 0.2	Initial sigma					
Breaking point	2.0	5.0	10.0	20.0	50.0	100.0
1	6.00%	8.00%	7.00%	8.00%	13.00%	10.00%
3	11.00%	11.00%	5.00%	7.00%	2.00%	10.00%
7	6.00%	7.00%	5.00%	7.00%	10.00%	7.00%
12	10.00%	6.00%	6.00%	10.00%	7.00%	9.00%
20	4.00%	9.00%	10.00%	10.00%	8.00%	8.00%

Breaking point alpha: 0.5	Initial sigma					
Breaking point	2.0	5.0	10.0	20.0	50.0	100.0
1	5.00%	5.00%	7.00%	10.00%	10.00%	5.00%
3	9.00%	7.00%	7.00%	6.00%	8.00%	3.00%
7	8.00%	6.00%	5.00%	12.00%	6.00%	8.00%
12	5.00%	3.00%	8.00%	4.00%	5.00%	10.00%
20	9.00%	11.00%	8.00%	5.00%	3.00%	6.00%

(23) Results of third method trained on OMNIGLOT dataset and map size 10x10

Conclusion

Conclusion - parameter space

- trivially, the smallest map size performed the best
  - high quantization error
- the higher breaking point, the better
  - learning rate is higher for more epochs
  - neurons can move around more significantly for longer
- high values of initial neighbourhood size did not have expected impact
  - increases volatility
  - the input data which is iterated through last has significantly more impact

Conclusion - methods and data designs

- MNIST and RASTER showed some promise
- OMNIGLOT results were, comparatively, disappointing
  - 1-shot learning might perform better on this data design
- overlap between methods
- second and third method performed slightly better overall
- further experimentation needed

Future work

Future work

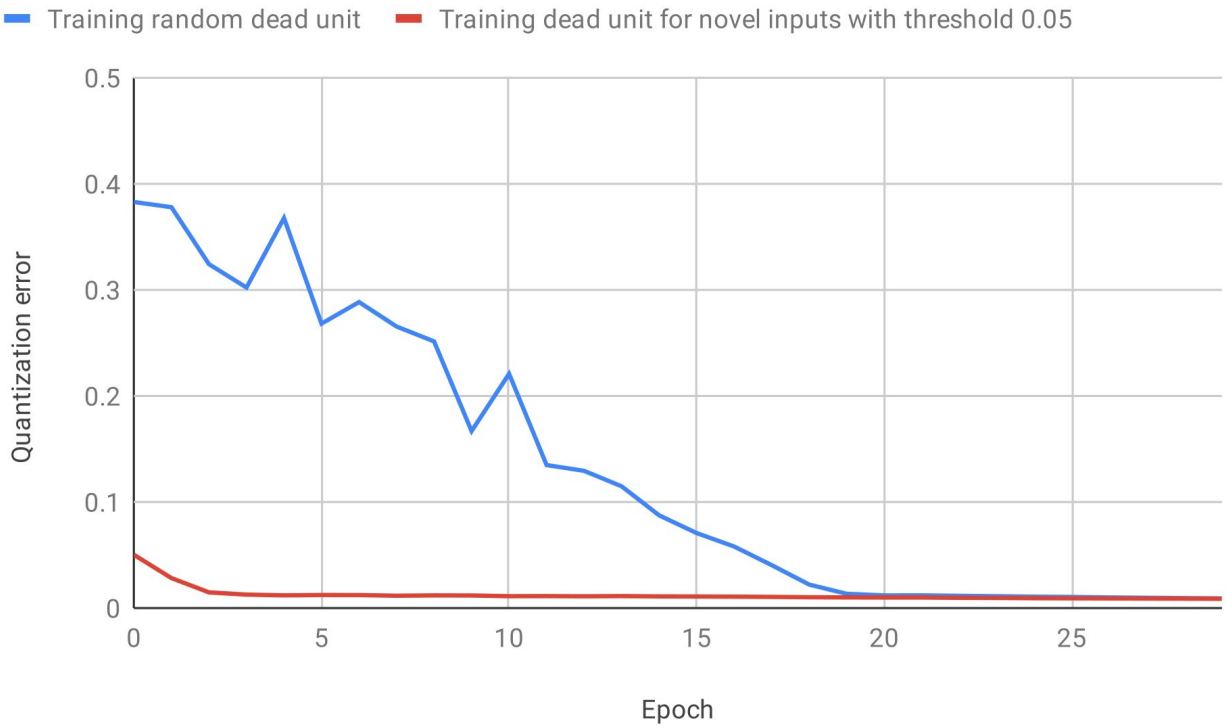
- technology and code
  - enabling quicker runs
- more random initializations of the neurons
  - enabling statistical tests
- parameter space - more granular exploration

Thank you

Questions

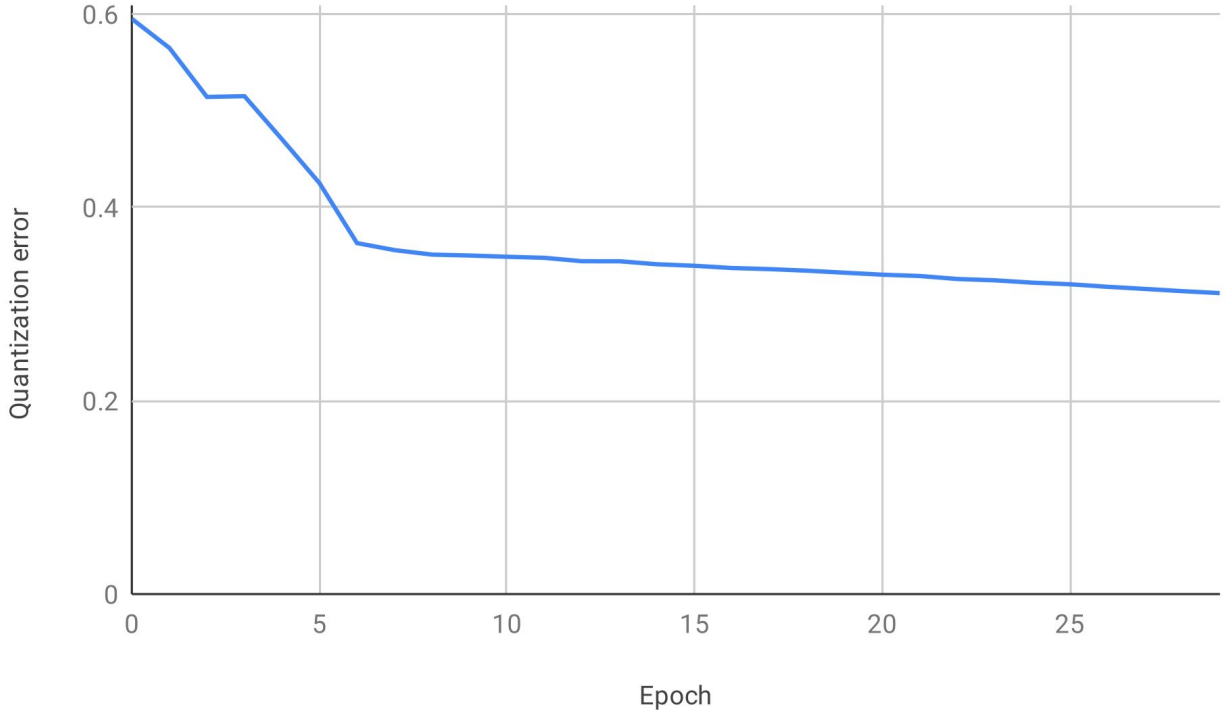
- Vysvetlite rozdiely, resp. zdôvodnite čísla v grafoch kvantizačnej chyby.
  - Euklidovská vzdialenosť v rámci dát je iná v závislosti od typu datasetu.

Questions



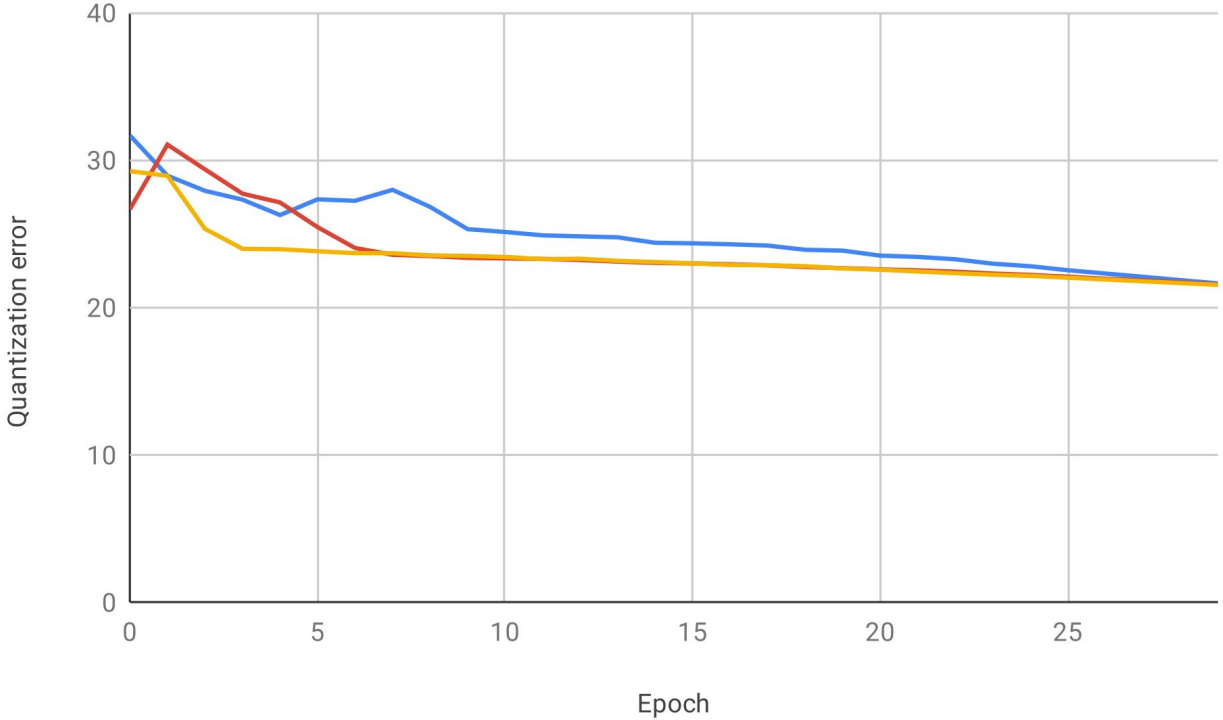
(24) 40x40 map - comparison of quantization errors for two of the best performing parameter combinations on Raster dataset

Questions



(25) 20x20 map - quantization error of the best performing parameter combination on MNIST dataset

Questions



(26) 10x10 map - comparison of quantization errors for three of the best performing parameter combinations on OMNIGLOT dataset

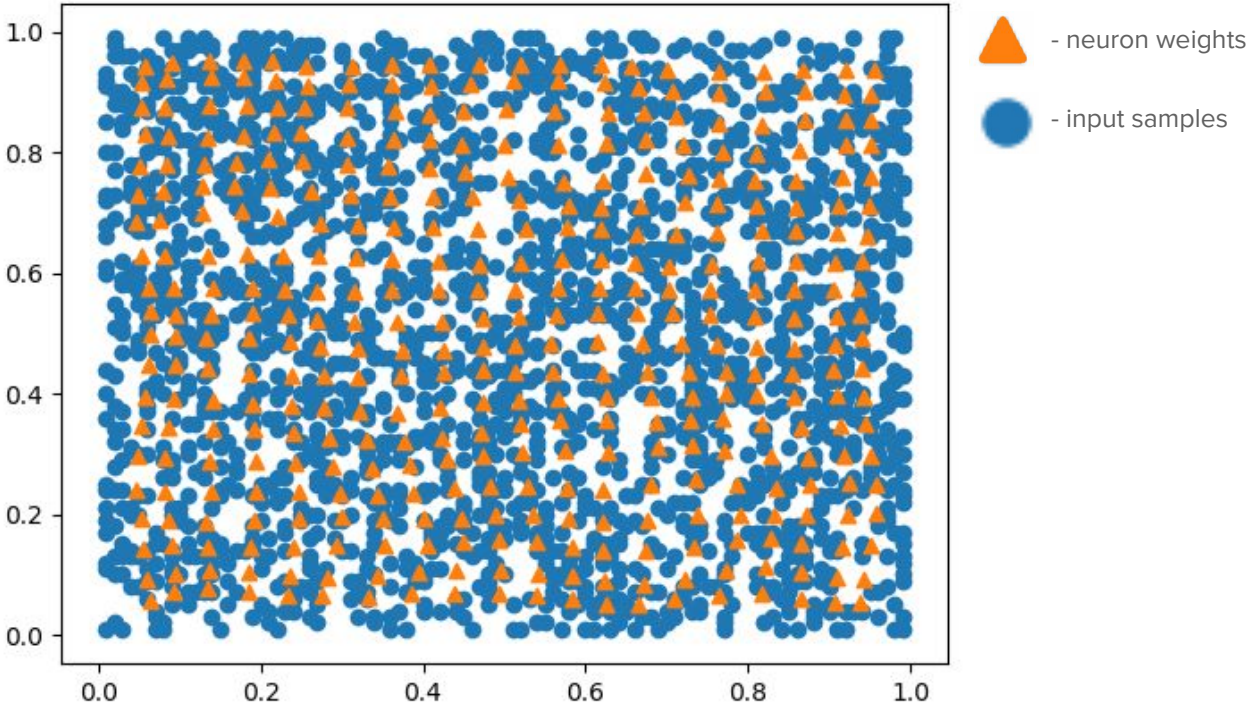
Questions

- Popíšte, ktoré časti jednotlivých metód by sa dali paralelizovať.
  1. počiatočná vzdialenosť neurónov od vstupného samplu a update váh
  2. počítanie vzdialeností
  3. počítanie vzdialeností

Questions

- Čo presne znamenajú body a trojuholníky na obrázku 4.17 (resp. 4.21, 4.25) a čo z týchto obrázkov usudzujete?
  - Body reprezentujú dáta
  - Trojuholníky reprezentujú váhy neurónov
  - Z obrázkov vieme usúdiť, ako dobre sa neuróny naučili vstup

Questions

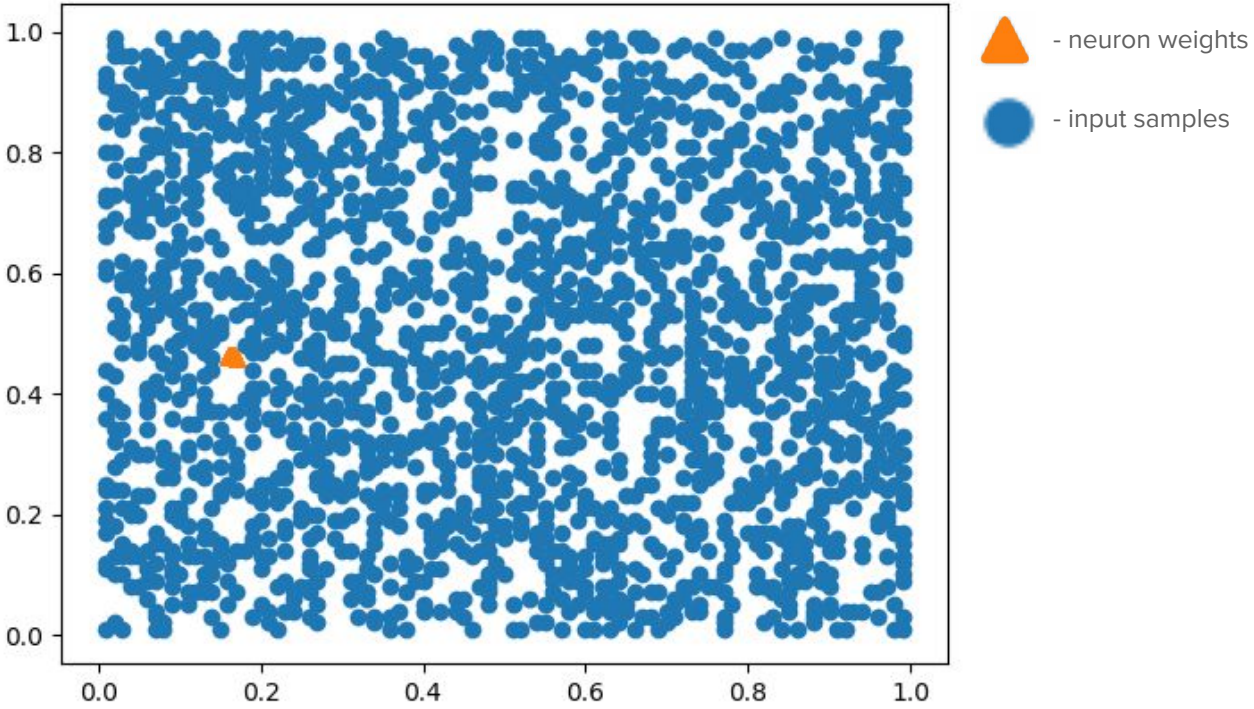


(27) 20x20 map - neuron weights at the last epoch of training dead units for novel inputs with threshold 0.05 method

Questions

- Prečo sú na začiatku všetky trojuholníky spolu?
  - Obrázky ukazujú stav mapy po prvej epoche
  - Mapa sa už niečo naučila
  - Vysoká sigma (veľkosť okolia) spôsobila, že sa všetky váhy neurónov prepísali tou istou hodnotou

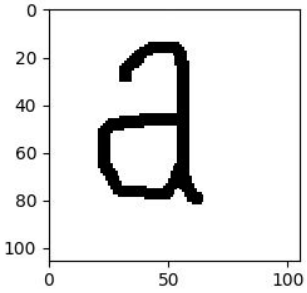
Questions



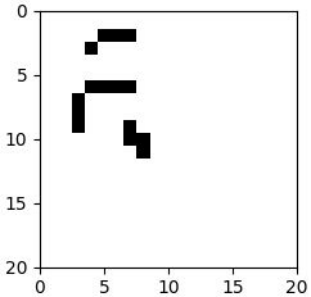
(28) 20x20 map - neuron weights at the 1st epoch of training dead units for novel inputs with threshold 0.05 method

Questions

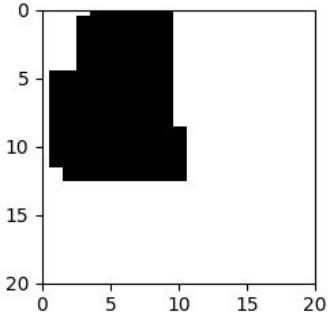
- Aký je rozdiel medzi resize, rescale a downscale?



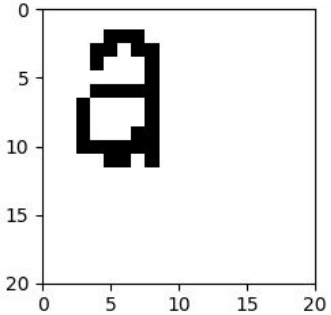
(29) Original



(30) Resize



(31) Rescale



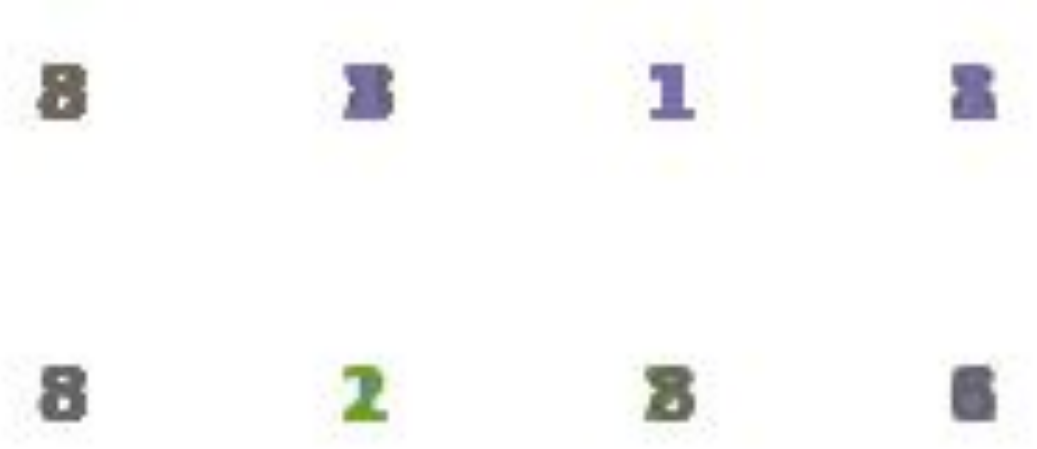
(32) Downscale

Questions

- Prečo boli ako výsledky týchto operácií použité binárne obrázky a nie odtiene šedej ako pri MNIST dátach?
  - OMNIGLOT dataset obsahuje binárne obrázky

Questions

- Ako je definovaný winmap v prípade, že jeden neurón je BMU pre viacero obrázkov s potenciálne rôznymi ciframi?
  - Vizualizačná pomôcka
  - V kóde si každý neurón pamätá, aké vstupy sa naň zobrazili



Questions

Map size	Time per combination	Total time
10x10	$\sim$ 4 minutes	$\sim$ 6 hours
20x20	$\sim$ 19 minutes	$\sim$ 28.5 hours
40x40	$\sim$ 89 minutes	$\sim$ 133.5 hours

(1) First method

Map size	Time per combination	Total time
10x10	$\sim$ 4 minutes	$\sim$ 6 hours
20x20	$\sim$ 21.5 minutes	$\sim$ 32 hours
40x40	$\sim$ 114.5 minutes	$\sim$ 172 hours

(2) Second method

Map size	Time per combination	Total time
10x10	$\sim$ 4 minutes	$\sim$ 18 hours
20x20	$\sim$ 20 minutes	$\sim$ 90 hours
40x40	$\sim$ 107 minutes	$\sim$ 481.5 hours

(3) Third method