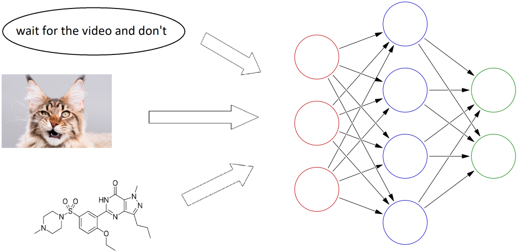
Deep processing of structured data

Łukasz Maziarka, Marek Śmieja, Aleksandra Nowak, Jacek Tabor, Łukasz Struski, Przemysław Spurek

GMUM Group of Machine Learning Methods,
Jagiellonian University, Kraków

Structured / non-vector data

- Typical machine learning models assume that data are represented as vectors of fixed size.
- In practice, raw data often do not have a vector form:
    - texts
    - images of varied sizes
    - graphs
    - sets with varied sizes
    - data with missing attributes



Problem

"Structured data" – data which are not given as vectors of fixed dimension.

Question

How to process structured data by neural networks?

In details:

- how to represent structured data?
- how to process these representations?

Outline

- processing of sets
- application in processing of text, graphs, images

Ł. Maziarka, et al., *Deep processing of structured data*, arXiv:1810.01868, 2018,

M. Zaheer, et al. *Deep sets*, NIPS 2017

Processing sets by neural networks

Problem

- Data set X is a family of sets $X_1, \ldots, X_N$
- Every set $X_i \subset \mathbb{R}^D$ is an individual element of a data set.

Question

Typical networks process vectors one by one:

- how to feed the whole set to the network?
- how to obtain a single output for the whole set?

Examples of tasks

We are given a family of sets, where each one has label (X_i, y_i).
Possible tasks:

- Learning the number of clusters (centers of clusters) for the input set.
- Learning a decision boundary for a given set.
- Finding the entropy (or any other statistic) of set.

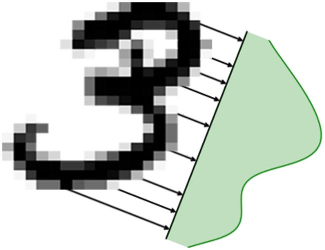
Structured data are usually represented as a set of features, so the set processing is essential.

Idea: projection

Cramer-Wold Theorem

Two sets are equal if they are equal on all one-dimensional projections.

Without loss of information, we can process sets X through their one-dimensional projections $v^T X$, where $v \in \mathbb{R}^D$



Step 1:

Pick some $v_1, \ldots, v_M \in \mathbb{R}^D$ and project X onto them.

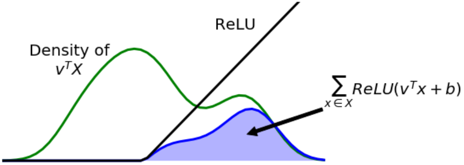
Idea: aggregation

- Let $v^T X \subset \mathbb{R}$ be 1D set and let $b \in \mathbb{R}$ be a fixed bias
- We define aggregated ReLU:

$$\begin{aligned} \mathrm{ReLU}_{v,b}(X) &= \textstyle\sum_{x \in X} \mathrm{ReLU}_{v,b}(x) \\ &= \textstyle\sum_{x \in X} \max(v^T x + b, 0) \in \mathbb{R}. \end{aligned}$$

Fact

We can reconstruct $v^T X \subset \mathbb{R}$ iff we know $\mathrm{ReLU}_{v,b}(X)$ for all $b \in \mathbb{R}$.



Step 2:

Pick some $b_1, \ldots, b_M \in \mathbb{R}$ and compute $\mathrm{ReLU}_{v,b_i}(X)$, for all i.

Procedure

Take a neural network with M output neurons (each parameterized by v, b)

- Project $X \subset \mathbb{R}^D$ onto 1D by $v^T X$
- Apply ReLU for every element

$$\mathrm{ReLU}_{v,b}(x) = \max(v^T x + b, 0)$$

- Summarize the result:

$$\begin{aligned} \mathrm{ReLU}_{v,b}(X) &= \textstyle\sum_{x \in X} \mathrm{ReLU}_{v,b}(x) \\ &= \textstyle\sum_{x \in X} \max(v^T x + b, 0) \end{aligned}$$

Representation:

Set aggregation network (SAN) with M output neurons gives M-dimensional representation of the set.

Processing structured data by neural networks.

General view

Let X be structured data, e.g. text, image, graph, etc.

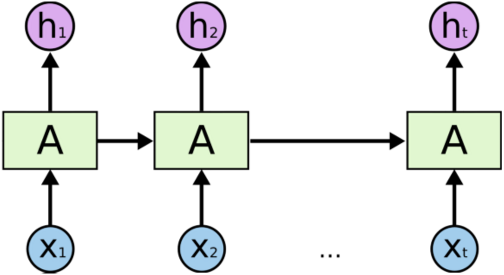
Processing pipeline

$$X = (x_i)_i \xrightarrow{\Psi} (\Psi x_i)_i \xrightarrow{\mathrm{Pool}} \mathrm{Pool}\{\Psi(x_i) : i\} \xrightarrow{\Phi} \mathbb{R}^N.$$

1. Ψ – feature extraction
2. Pool – set aggregation
3. Φ – final output

Step 1a

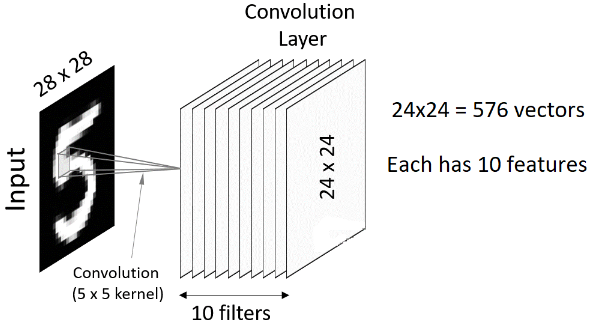
Recurrent network for extracting sequential patterns (e.g. for texts)



ψ returns a sequence of vectors.

Step 1b

Convolutional networks for extracting local patterns (e.g. for images)



ψ returns the image.

Step 2

Let $X \subset \mathbb{R}^K$ be a set of extracted features

- Typically, take max over each attribute to produce a fixed length vector
- Better idea is to replace pooling layer by set aggregation network (SAN) to preserve necessary information from a set.

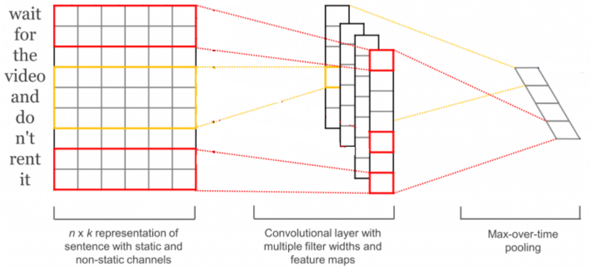


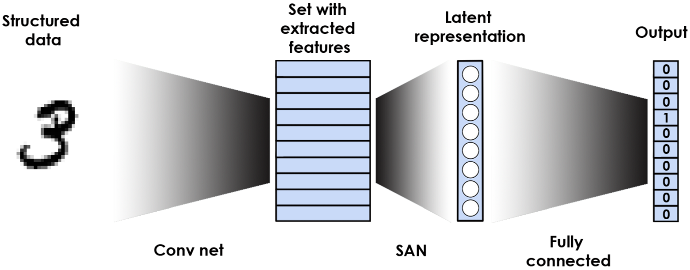
Figure: Max-pooling for 1D-convolutions.

Step 3

Let $x \in \mathbb{R}^K$ be an aggregated vector.

- Use a classical (fully connected) neural network ϕ to produce a final output

Summary

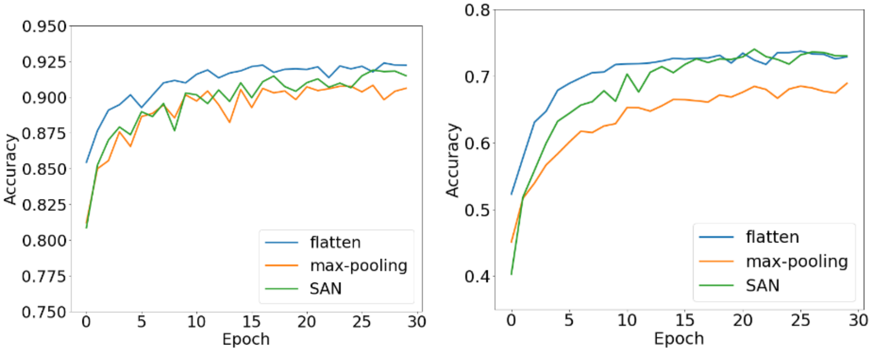

Structured data
Set with extracted features
Latent representation
Output
0
0
0
1
0
0
0
0
0
0
Conv net
SAN
Fully connected

Images

Table 3: Classification accuracy for images with varied resolutions.

		Trained on all resolutions		Trained only on original resolution		
Dataset	Image size	max-pooling	SAN	max-pooling	SAN	SAN-raw
Fashion MNIST	14x14	0.8788	**0.8810**	0.2519	**0.2884**	0.2148
	22x22	0.8969	**0.9064**	0.7380	**0.8247**	0.4563
	28x28	0.9023	**0.9111**	0.9062	**0.9150**	0.8114
	42x42	0.9020	**0.9033**	0.5548	**0.6893**	0.3946
	56x56	0.8913	**0.8966**	0.3274	**0.4515**	0.3605
CIFAR-10	16x16	0.5830	**0.6167**	0.3213	0.4145	**0.4862**
	26x26	0.6689	**0.7037**	0.5974	**0.6706**	0.4957
	32x32	0.6838	**0.7292**	0.6891	**0.7302**	0.4968
	48x48	0.6813	**0.7080**	0.5542	**0.5921**	0.4932
	64x64	0.6384	**0.6413**	0.3904	0.3658	**0.4922**

Images



(a) Fashion MNIST (b) CIFAR-10

Figure 4: Classification accuracy on images with a fixed resolution.

Text Cheminfortics

Cheminfortics

Table 1: AUC scores for classifying active chemical compounds .

	ECFP4	KlekFP	Mordred	JTVAE	set of atoms
Random Forest	0.726	0.604	0.493	0.527	-
Extra Trees	0.713	0.667	0.501	0.470	-
k-NN	0.532	0.569	0.546	0.521	-
Dense network	0.702	0.574	0.593	0.560	-
SAN	-	-	-	-	0.643

Text

Table 2: Accuracy scores obtained for text data.

dataset	CNN	SAN
MR	0.7623	**0.7646**
imdb	**0.8959**	0.8480

Regularization