ReLU Neural Networks of Polynomial Size for Exact Maximum Flow Computation

Christoph Hertrich

LSE

Leon Sering

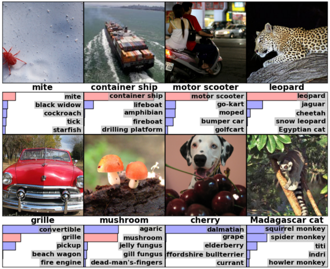
ETH zürich

started at:

TU berlin

IPCO, Madison, WI, USA
June 21, 2023

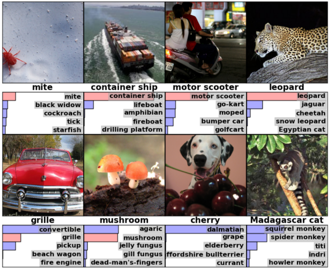
Neural Networks in Action



Krizhevsky et al. “Imagenet classification with deep convolutional neural networks” (NeurIPS 2012)

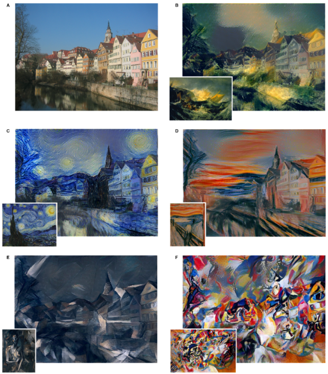
Neural Networks in Action

Gatys et al. "Image style transfer using convolutional neural networks" (CVPR 2016)

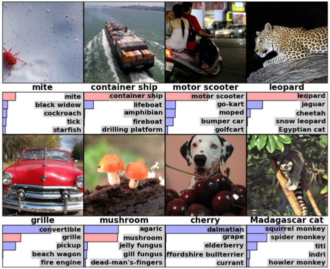


Krizhevsky et al. "Imagenet classification with deep convolutional neural networks" (NeurIPS 2012)

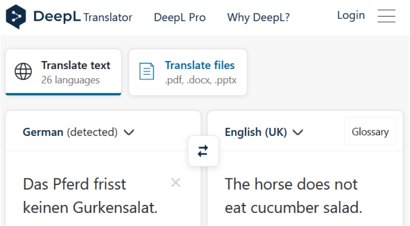
Neural Networks in Action



Gatys et al. “Image style transfer using convolutional neural networks” (CVPR 2016)

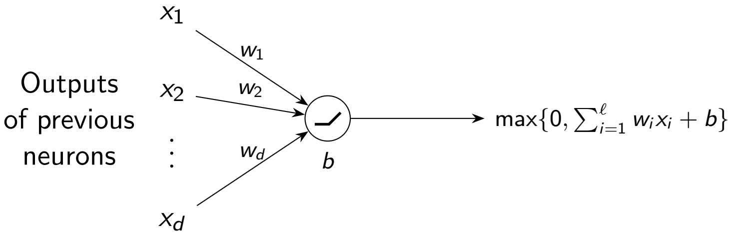


Krizhevsky et al. “Imagenet classification with deep convolutional neural networks” (NeurIPS 2012)

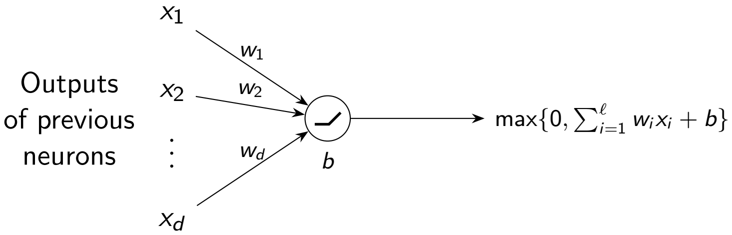


Screenshot `deepl.com` (Feb 18, 2022)

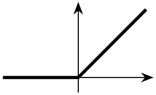
A Single ReLU Neuron



A Single ReLU Neuron



Rectified linear unit (ReLU): $\mathrm{relu}(x) = \max\{0, x\}$



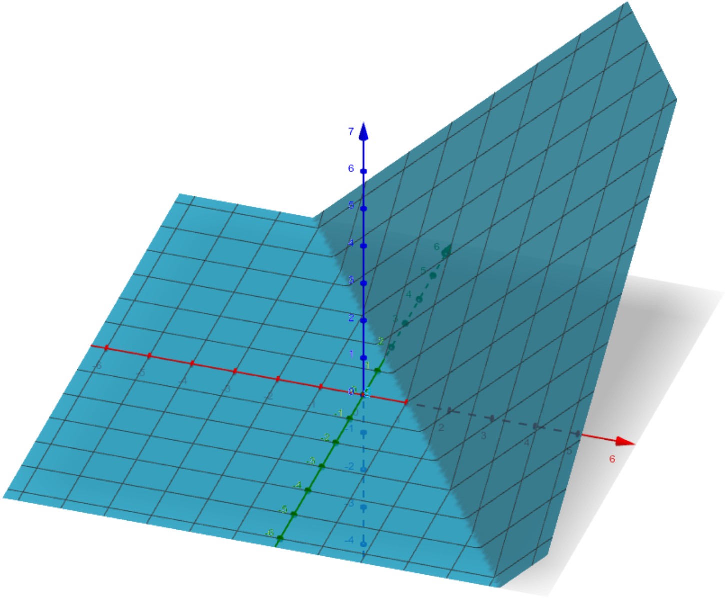
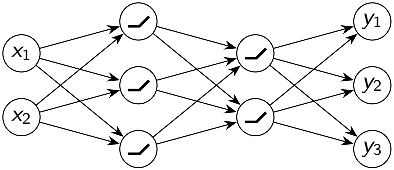


Image created with GeoGebra (geogebra.org)

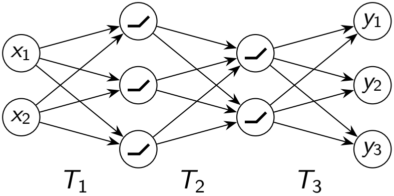
ReLU Feedforward Neural Networks

- Acyclic (layered) digraph of ReLU neurons



ReLU Feedforward Neural Networks

- Acyclic (layered) digraph of ReLU neurons



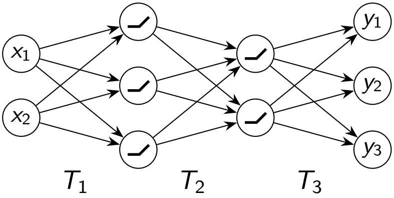
- Computes function

$$T_k \circ \text{relu} \circ T_{k-1} \circ \cdots \circ T_2 \circ \text{relu} \circ T_1$$

with affine transformations T_i.

ReLU Feedforward Neural Networks

- Acyclic (layered) digraph of ReLU neurons



- Computes function

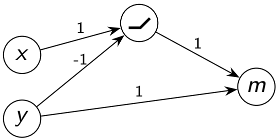
$$T_k \circ \mathrm{relu} \circ T_{k-1} \circ \cdots \circ T_2 \circ \mathrm{relu} \circ T_1$$

with affine transformations T_i.

- Example: depth 3, size 5.

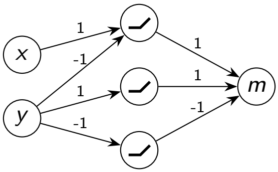
Example: Computing the Maximum of Two Numbers

$$\max\{x, y\} = \max\{x - y, 0\} + y$$

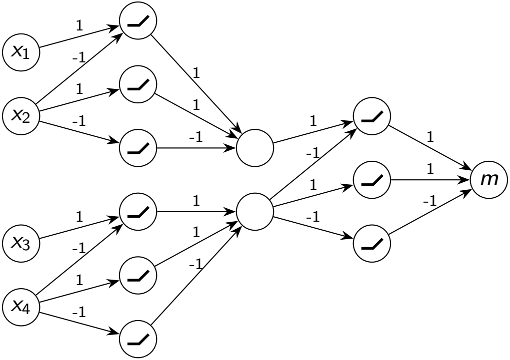


Example: Computing the Maximum of Two Numbers

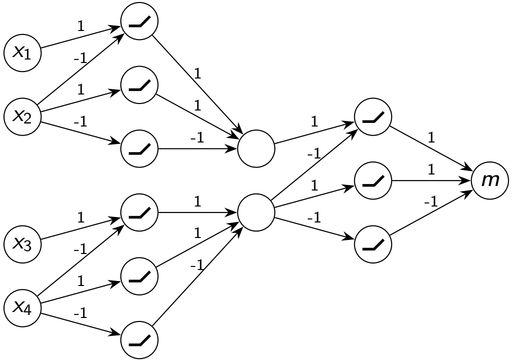
$$\max\{x, y\} = \max\{x - y, 0\} + y$$



Example: Computing the Maximum of Four Numbers

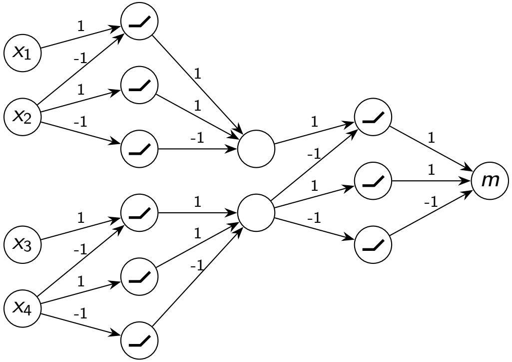


Example: Computing the Maximum of Four Numbers



- Inductively:
  Max of n numbers with depth $\mathcal{O}(\log n)$ and size $\mathcal{O}(n)$.

Example: Computing the Maximum of Four Numbers



- Inductively:
  Max of n numbers with depth $\mathcal{O}(\log n)$ and size $\mathcal{O}(n)$.
- Minimum similarly.

Representing Arbitrary Piecewise Linear Functions

Observation

Every function represented by a ReLU NN is continuous and piecewise linear (CPWL).

Representing Arbitrary Piecewise Linear Functions

Observation

Every function represented by a ReLU NN is continuous and piecewise linear (CPWL).

Theorem (Arora, Basu, Mianjy, Mukherjee (ICLR 2018))

Every CPWL function $f : \mathbb{R}^n \to \mathbb{R}$ can be represented by a ReLU NN with depth $\mathcal{O}(\log n)$.

Open Questions

Theorem (Arora, Basu, Mianjy, Mukherjee (ICLR 2018))

Every CPWL function $f : \mathbb{R}^n \to \mathbb{R}$ can be represented by a ReLU NN with depth $\mathcal{O}(\log n)$.

Open Questions

Theorem (Arora, Basu, Mianjy, Mukherjee (ICLR 2018))

Every CPWL function $f : \mathbb{R}^n \to \mathbb{R}$ can be represented by a ReLU NN with depth $\mathcal{O}(\log n)$.

1. Is logarithmic depth best possible?

Open Questions

Theorem (Arora, Basu, Mianjy, Mukherjee (ICLR 2018))

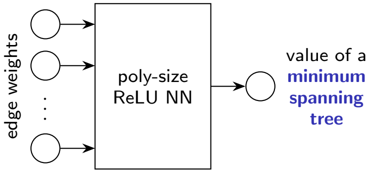
Every CPWL function $f : \mathbb{R}^n \to \mathbb{R}$ can be represented by a ReLU NN with depth $\mathcal{O}(\log n)$.

1. Is logarithmic depth best possible?

2. **Which functions can we represent with polynomial size?**

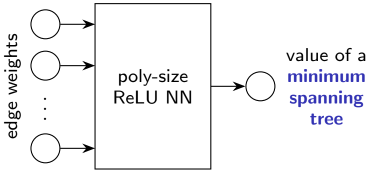
Our Results

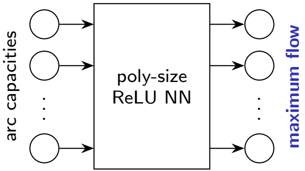
Hertrich, Sering (IPCO 2023)



Our Results

Hertrich, Sering (IPCO 2023)





Neural Networks as Model of **Real-Valued** Computation

Neural Networks as Model of **Real-Valued** Computation

Neural Networks

$=$

Arithmetic circuits with (weighted) sums and maxima gates

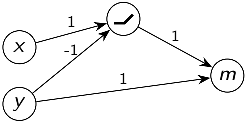
Neural Networks as Model of **Real-Valued** Computation

Neural Networks

$=$

Arithmetic circuits with (weighted) sums and maxima gates

Polynomial-Size NNs

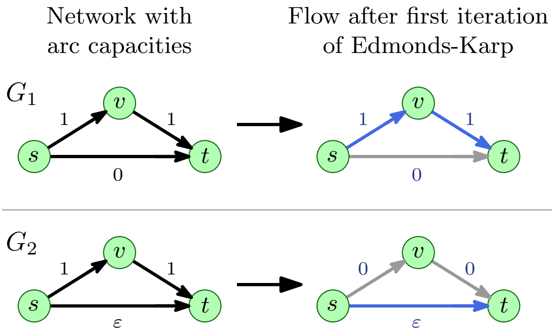


$\Leftrightarrow$

Strongly polynomial algorithms
with restricted set of operations

Input: $x, y \in \mathbb{R}$
$v_1 = x - y$
$v_2 = \max\{0, v_1\}$
$m = v_2 + y$
return m.

Why Edmonds-Karp does not work



What to do instead?

Assume s-t-distance of length $\geq k$ in residual network. Need to find an augmenting s-t-flow which

- uses only arcs along paths of length k,
- is feasible in the residual network, and
- saturates at least one arc (if distance $= k$).

What to do instead?

Assume s-t-distance of length $\geq k$ in residual network. Need to find an augmenting s-t-flow which

- uses only arcs along paths of length k,
- is feasible in the residual network, and
- saturates at least one arc (if distance $= k$).

Then: Similar analysis to Edmonds-Karp

How to find augmenting flow

1. Compute fattest path values $a_{i,v}$: maximum amount of flow that can be pushed on a single path of length exactly i from v to t:

$$a_{i,v} = \max_{vw \in E} \{\min\{c_{vw}, a_{i-1,w}\}\}$$

How to find augmenting flow

1. Compute fattest path values $a_{i,v}$: maximum amount of flow that can be pushed on a single path of length exactly i from v to t:

$$a_{i,v} = \max_{vw \in E} \{\min\{c_{vw}, a_{i-1,w}\}\}$$

Observe: $\mathrm{dist}(v,t) = \min\{i \mid a_{i,v} > 0\}$.

How to find augmenting flow

1. Compute fattest path values $a_{i,v}$: maximum amount of flow that can be pushed on a single path of length exactly i from v to t:

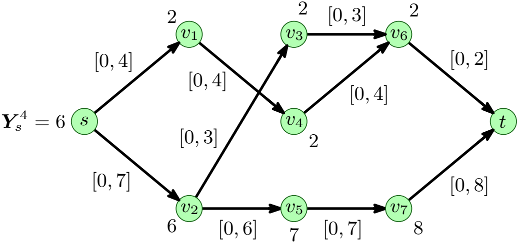
$$a_{i,v} = \max_{vw \in E} \{\min\{c_{vw}, a_{i-1,w}\}\}$$

   Observe: $\mathrm{dist}(v,t) = \min\{i \mid a_{i,v} > 0\}$.

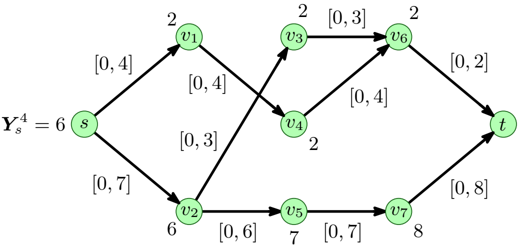
2. Push greedily flow from s to t, while ensuring that in i-th pushing phase flow arriving at v does not exceed $a_{k-i,v}$.

How to find augmenting flow

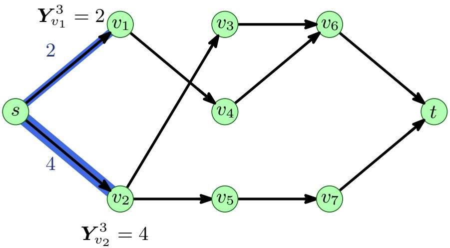
1. Compute fattest path values $a_{i,v}$: maximum amount of flow that can be pushed on a single path of length exactly i from v to t:
$$a_{i,v} = \max_{vw \in E} \{\min\{c_{vw}, a_{i-1,w}\}\}$$
Observe: $\mathrm{dist}(v,t) = \min\{i \mid a_{i,v} > 0\}$.
2. Push greedily flow from s to t, while ensuring that in i-th pushing phase flow arriving at v does not exceed $a_{k-i,v}$.
3. Clean-up to restore flow conservation.

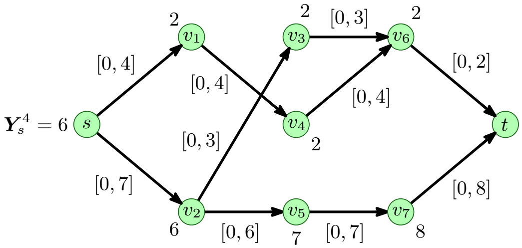


$\boldsymbol{Y}_s^4 = 6$
s
v_1
v_2
v_3
v_4
v_5
v_6
v_7
t
2
2
2
2
6
7
8
$[0,4]$
$[0,7]$
$[0,4]$
$[0,3]$
$[0,6]$
$[0,3]$
$[0,4]$
$[0,7]$
$[0,2]$
$[0,8]$

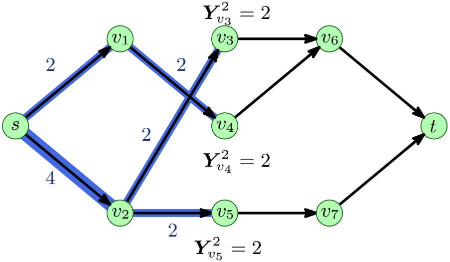


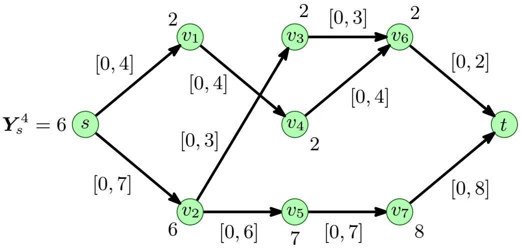
push phase; after $i = 4$



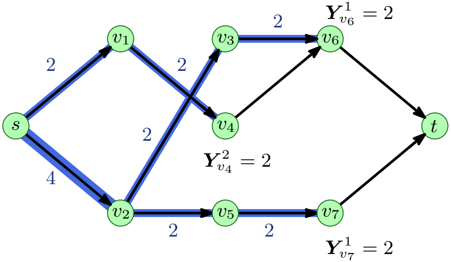


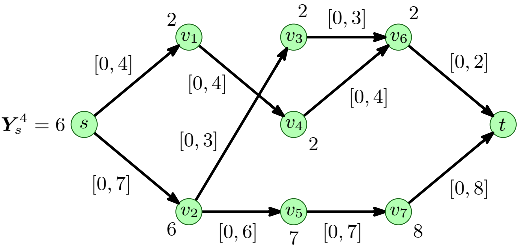
push phase; after $i = 3$



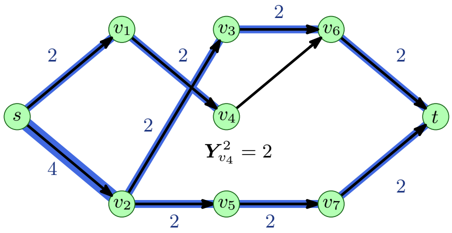


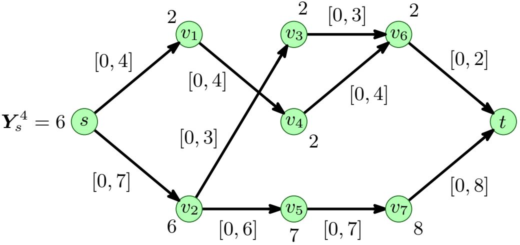
push phase; after $i = 2$



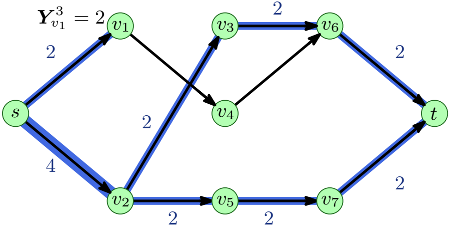


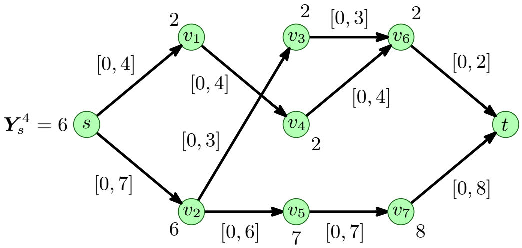
push phase; after $i = 1$



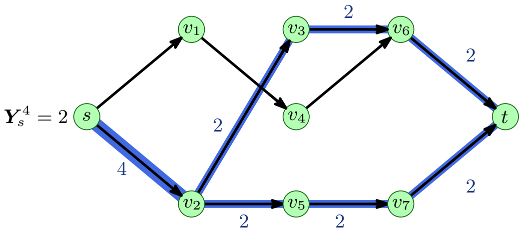


clean-up; after $i = 2$



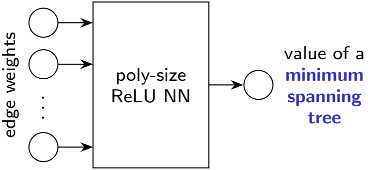


clean-up; after $i = 3$



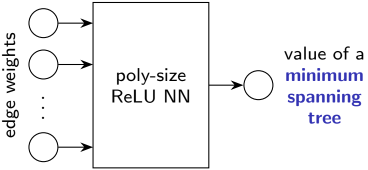
Outlook

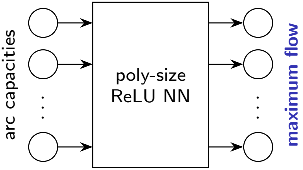
Hertrich, Sering (IPCO 2023)



Outlook

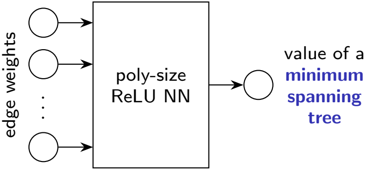
Hertrich, Sering (IPCO 2023)

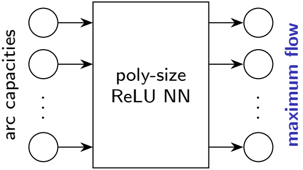




Outlook

Hertrich, Sering (IPCO 2023)

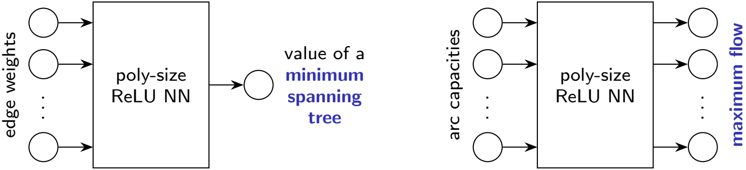




- Which functions can be represented by poly-size NNs?

Outlook

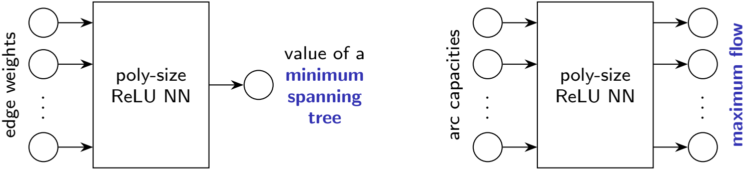
Hertrich, Sering (IPCO 2023)



- Which functions can be represented by poly-size NNs?
- Mininum Cost Flows, Matchings, general LPs?

Outlook

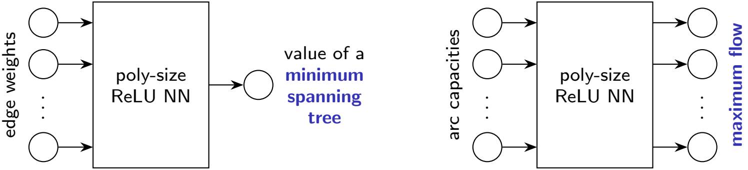
Hertrich, Sering (IPCO 2023)



- Which functions can be represented by poly-size NNs?
- Mininum Cost Flows, Matchings, general LPs?
- Are there CPWL functions computable in strongly polynomial time which are not representable by poly-size NNs?

Outlook

Hertrich, Sering (IPCO 2023)



- Which functions can be represented by poly-size NNs?
- Mininum Cost Flows, Matchings, general LPs?
- Are there CPWL functions computable in strongly polynomial time which are not representable by poly-size NNs?
- Might extended formulations help?