Aalto-yliopisto
Perustieteiden
korkeakoulu

Lossless Compression of Deep Neural Networks (topic-presentation)

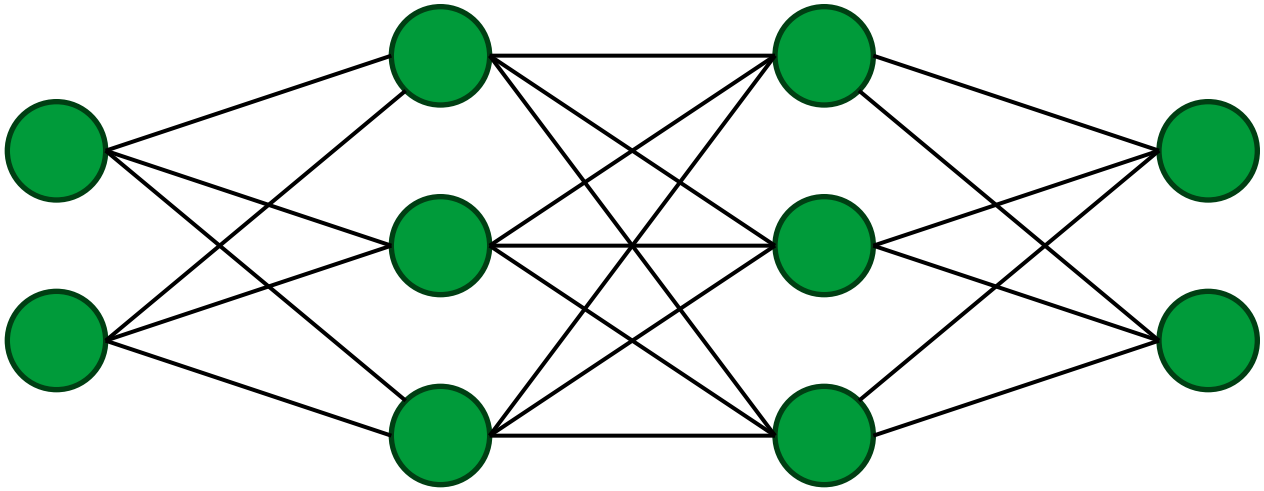
Vilhelm Toivonen

01.11.2023

Instructor: Nikita *Belyak*
Supervisor: Fabricio *Oliveira*

Systeemianalyysin
laboratorio

Background – DNNs



Larger DNNs make calculations more intensive

- Slow forward passes
- Computationally expensive to create mathematical programming problems

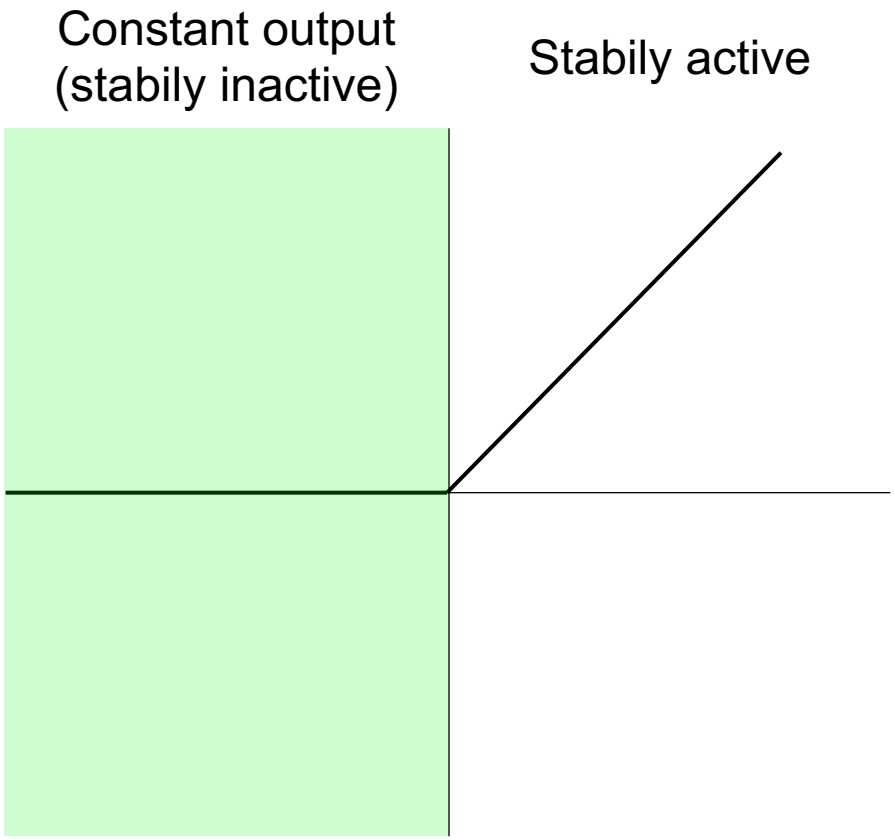
Background – ReLU

Using the Rectified Linear Unit as the activation, we can separate the output in two parts:

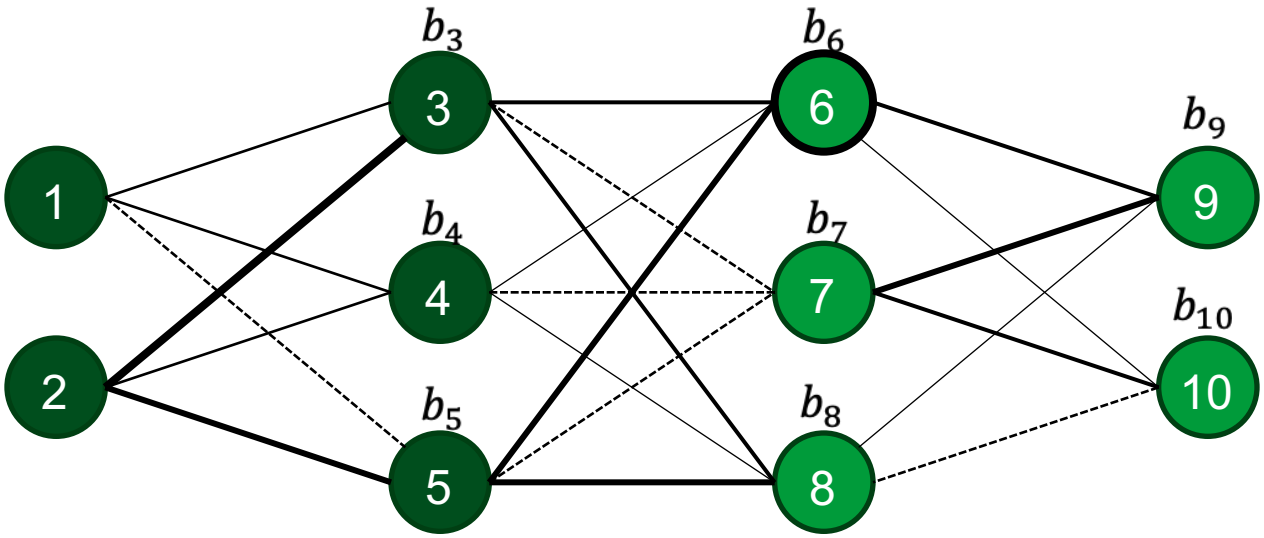
- Constant output
- Stabily active

Neurons with constant outputs can be pruned

Stabily active neurons can be simplified to system of linear equations, and linearly dependent neurons pruned

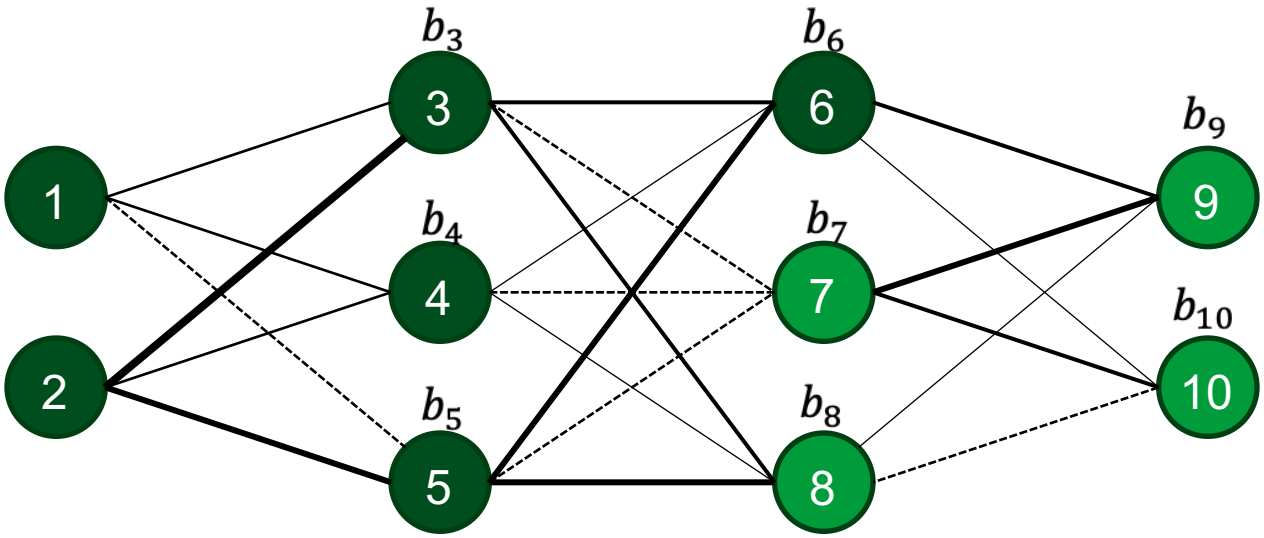


Background – Example of pruning



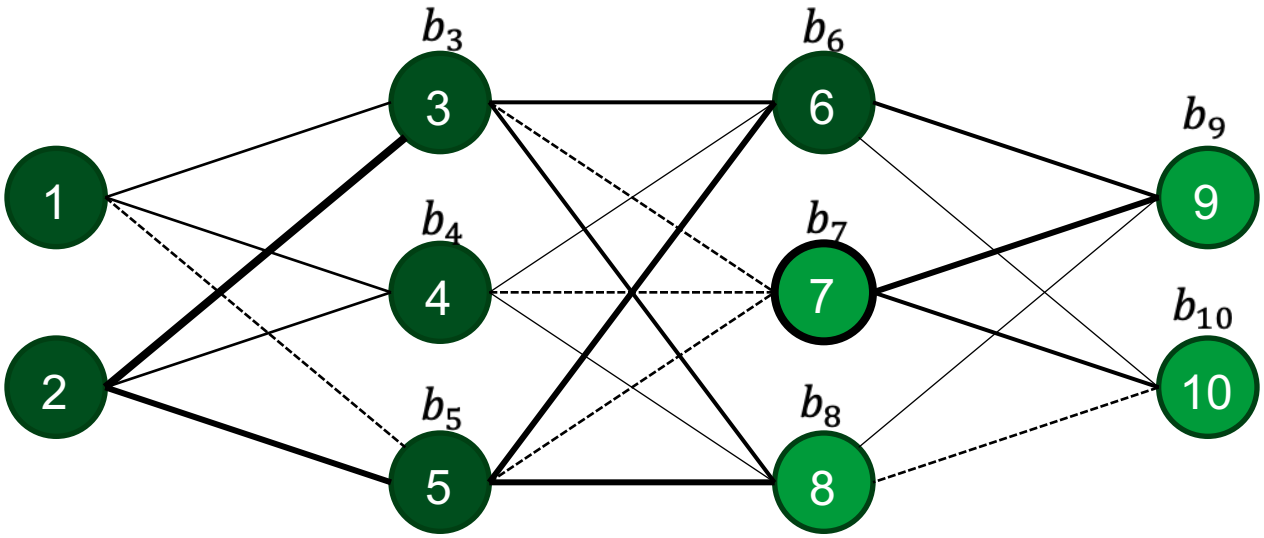
$$l = 2$$
$$i = 1$$
$$\bar{G}_i^l > 0$$
$$S = \{\}$$

Background – Example of pruning



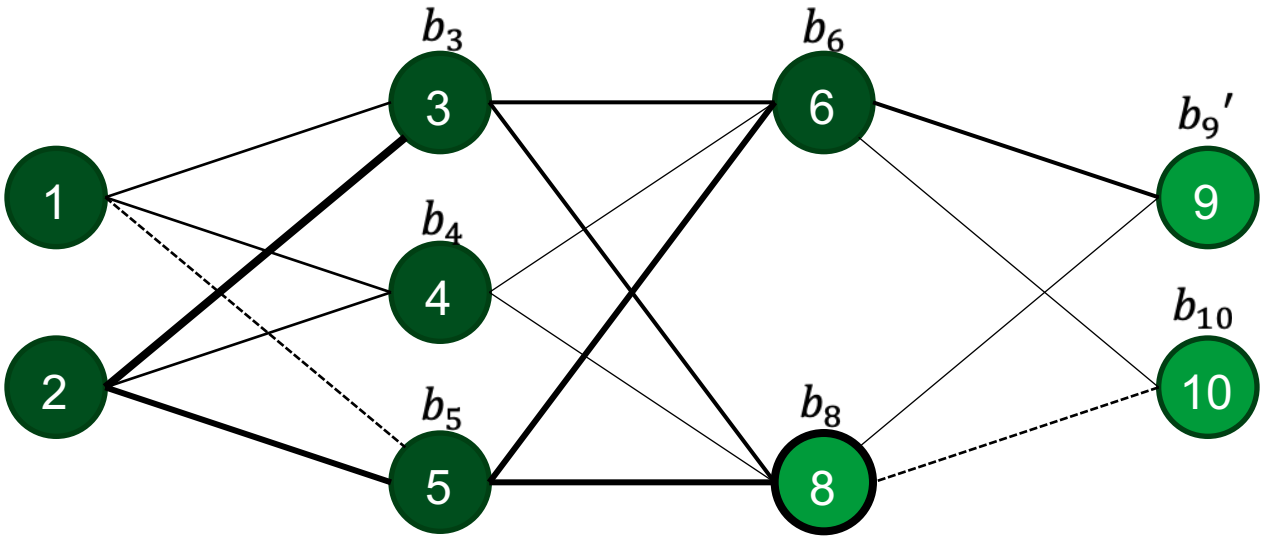
$$l = 2$$
$$i = 1$$
$$\bar{G}_i^l > 0$$
$$S = \{1\}$$

Background – Example of pruning



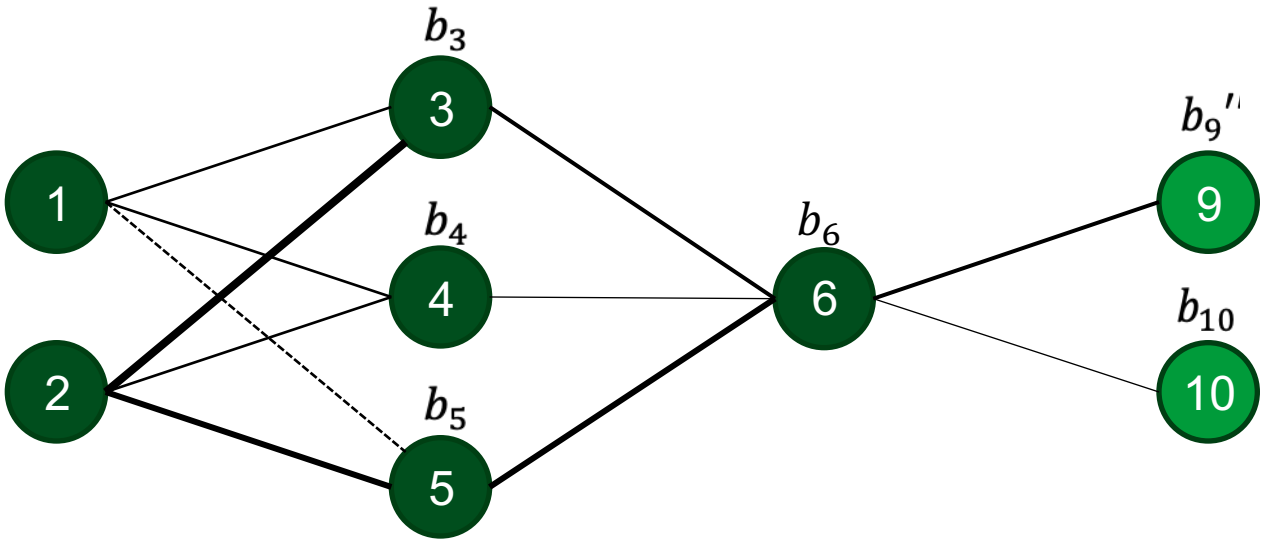
$$l = 2$$
$$i = 2$$
$$W_i^l = 0$$
$$S = \{1\}$$

Background – Example of pruning

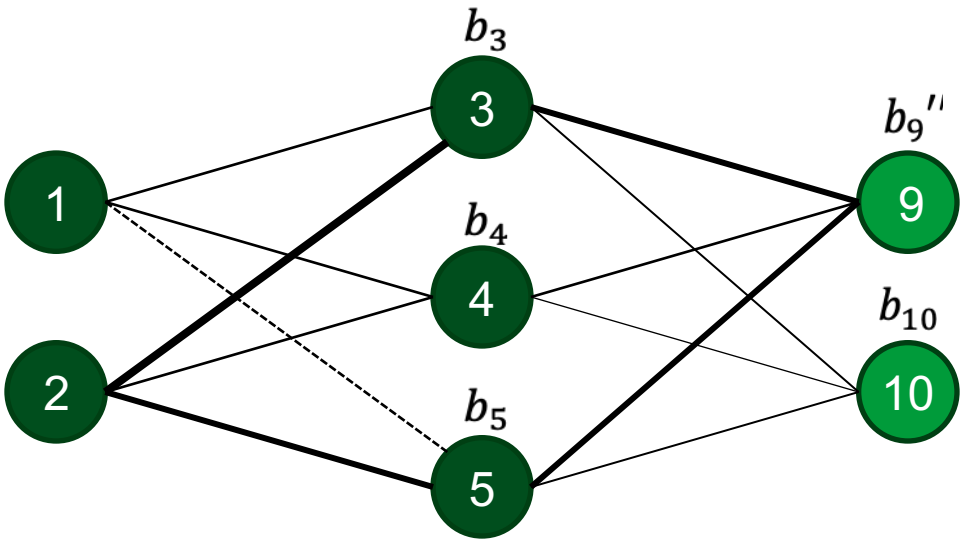


$$l = 2$$
$$i = 3$$
$$\bar{G}_i^l > 0$$
$$S = \{1\}$$

Background – Example of pruning



Background – Example of pruning



Goals

Develop a working algorithm to compress neural networks.

Determine the optimal set of hyperparameters that result in the most effective compression. Consider l1 and l2 norm regularizations with different weights, and at least the SDG, Momentum and Adam optimizers.

for $l \leftarrow 1, \ldots, L$ **do**
  $S \leftarrow \{\}$ ▷ Set of stable units left in layer l
  Unstable $\leftarrow$ False ▷ If there are unstable units in layer l
  for $i \leftarrow 1, \ldots, n_l$ **do**
    if $G_i^l < 0$ for $x \in \mathbb{D}$ or $W_i^l = 0$ **then** ▷ Stabily inactive, constant output
      if $i < n_l$ or $|S| > 0$ or Unstable **then**
        if $W_i^l = 0$ and $b_i^l > 0$ **then**
          for $j \leftarrow 1, \ldots, n_{l+1}$ **do**
            $b_j^{l+1} \leftarrow b_j^{l+1} + w_{ji}^{l+1} b_i^l$
          end for
        end if
        Remove unit i from layer l ▷ Unit i is not necessary
      end if
    else if $\bar{G}_i^l > 0$ for $x \in \mathbb{D}$ **then** ▷ Stabily active
      if $\text{rank}(W^l_{S \cup \{i\}}) > |S|$ **then**
        $S \leftarrow S \cup \{i\}$ ▷ Keep unit in the network
      else
        Find $\{\alpha_k\}_{k \in S}$ such that $w_i^l = \sum_{k \in S} \alpha_k w_k^l$
        for $j \leftarrow 1, \ldots, n_{l+1}$ **do**
          for $k \in S$ **do**
            $w_{jk}^{l+1} \leftarrow w_{jk}^{l+1} + \alpha_k w_{ji}^{l+1}$
          end for
          $b_j^{l+1} \leftarrow b_j^{l+1} + w_{ji}^{l+1} \left(b_i^l - \sum_{k \in S} \alpha_k b_k^l\right)$
        end for
        Remove unit i from layer l ▷ Unit i is no longer necessary
      end if
    else
      Unstable $\leftarrow$ True
    end if
  end for
  if not Unstable **then** ▷ All units left in layer l are stable
    if $|S| > 0$ **then** ▷ The units left have varying outputs
      Create matrix $\bar{\mathbf{W}} \in \mathbb{R}^{n_l \times n_{l+1}}$ and vector $\bar{\mathbf{b}} \in \mathbb{R}^{n_{l+1}}$
      for $i \leftarrow 1, \ldots, n_{l+1}$ **do**
        $\bar{b}_i \leftarrow b_i^{l+1} + \sum_{k \in S} w_{ik}^{l+1} b_k^l$
        for $j \leftarrow 1, \ldots, n_{l-1}$ **do**
          $\bar{w}_{ij} \leftarrow \sum_{k \in S} w_{kj}^l w_{ik}^{l+1}$
        end for
      end for
      Remove layer l; replace parameters in next layer with $\bar{\mathbf{W}}$ and $\bar{\mathbf{b}}$
    else ▷ Only unit left in layer l has constant output
      Compute output Υ for any input $\chi \in \mathbb{D}$
      $(W^{L+1}, b^{L+1}) \leftarrow (0, \Upsilon)$ ▷ Set constant values in output layer
      Remove layers 1 to L and **break** ▷ Remove all hidden layers and leave
    end if
  end if
end for

Limitations

Only consider DNNs with the ReLU activation.

Do not consider other architectures, such as CNNs or RNNs.

Do not focus on calculating the bounds for the neurons.

Sources and materials

- Thiago Serra, Abhinav Kumar, and Srikumar Ramalingam 2020. Lossless Compression of Deep Neural Networks. Bucknell University and the University of Utah. Usa
- Linkola, J. 2023. Reformulating deep neural networks as mathematical programming problems. Bachelor thesis. Aalto-University. School of Science. Espoo.
- ML_as_MO package (https://github.com/gamma-opt/ML_as_MO). Includes implementations for calculating bounds

Tools

Julia, includes an implementation for calculating the bounds for neurons.

Schedule

- Introduction to topic and understanding given materials 06-10/2023
- Experimentation 10-12/2023
- Topic presentation 11/2023
- Writing the thesis 11/2023-01/2024
- Results presentation 12/2023
- Thesis ready 01/2024