[T1 - D1.1] Compare classic Data Structures vs Purely Learned Indexes - Characterization of the space-time-accuracy performance of ML models in terms of distribution of the input data, and their comparison against the known (compressed) data structures

Collaboration with UNIPA unit

- Preliminary experiments about learned indexes with neural networks (NN) models [UNIPA] + NN compression techniques [UNIMI] to reduce the space occupancy + their comparison with some SOA indexes [UNIPA]
- PROBLEM INPUT: sorted sequence of integers $X$ [extendable to other type of elements]
- PROBLEM OUTPUT: a learned indexed for searching elements in $X$
- STEP 1: Coding the input for feeding the NN [e.g. binary representation]
- STEP 2: Coding the labels for training $(x \in X, \hat{F}(x))$
- STEP 3: NN training
- STEP 4 [optional]: NN compression
- STEP 5: Index validation
Some preliminary results.

$X$ uniformly distributed

<table>
<thead>
<tr>
<th>NN model</th>
<th>Compression rate</th>
<th>max error</th>
<th>max error/size</th>
<th>mean error/size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input size 1.05 \cdot 10^6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN3</td>
<td>4.73</td>
<td>1270/660</td>
<td>1.2 \cdot 10^{-1}/6.3 \cdot 10^{-2}</td>
<td>2.40 \cdot 10^{-4}/1.45 \cdot 10^{-4}</td>
</tr>
<tr>
<td>NN2</td>
<td>4.39</td>
<td>1031/699</td>
<td>9.0 \cdot 10^{-2}/6.7 \cdot 10^{-2}</td>
<td>2.33 \cdot 10^{-4}/1.45 \cdot 10^{-4}</td>
</tr>
<tr>
<td></td>
<td>Input size 8192</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN3</td>
<td>4.75</td>
<td>40/32</td>
<td>4.8 \cdot 10^{-1}/3.9 \cdot 10^{-1}</td>
<td>1.14 \cdot 10^{-3}/1.24 \cdot 10^{-4}</td>
</tr>
<tr>
<td>NN2</td>
<td>4.4</td>
<td>33/34</td>
<td>4.0 \cdot 10^{-1}/4.1 \cdot 10^{-1}</td>
<td>1.12 \cdot 10^{-3}/1.19 \cdot 10^{-3}</td>
</tr>
</tbody>
</table>

Ongoing activities:

Experiments with different data distributions

Extending the learned index in the sense of Kraska et al. 2017 (also related to Task 3)
[T1 - D1.1]. Short- Mid-term objectives

- Characterize the effectiveness of NN compression techniques with different data distributions.
- Actually we can relate the expected space occupancy to the compression rate or designing the model to reduce the *mean* error. We aim at investigating whether it is possible to relate (empirically at least) space occupancy with *maximum* error.
- Extending the learned index to more levels and evaluate how the compression effectiveness (space - accuracy) varies with it.
- Comparing with other ‘classical’ indexes.
Compress ML models - Development of compressed ML models, by investigating the trade-off between compressed space, compressed time and the prediction accuracy

**Just UNIMI unit (so far …)**

- Preliminary experiments to evaluate the capability of different compression techniques for NN models to reduce the model space requirements while preserving (or improving) its accuracy
  - Pruning
  - Weight sharing
  - Sparsified NN training
  - ...
  - Combined approaches

- Results might change with the input problem
- NN models for different categories of problems: multi-class/binary classification, regression, dimensionality reduction and so forth
Some preliminary results

**MNIST digit classification**

**INPUT**: feature vector (dim 784) representing images of hand-written digits

**OUTPUT**: softmax 10-neurons layer

**Model structure**: FFW with 1 hidden layer (variable size)

**Ongoing activities**: Studying novel techniques for NN compression

Experiments with different problem categories
[T2 - D2.1]. Some preliminary results
[T2 - D2.1]. Short- Mid-term objectives

- Applying different NN models
- Exploring/designing other NN compression methods. In particular, merging the NN learning and compression phases
  - With the aim of defining a multi-criteria training algorithms (optimizing space and accuracy)
- Compressing other ML models?
We can also give a contribution to

[T1 - D1.2] A collection of known and new implementations of ML-based and compressed data structures, to be used in the next tasks T3 and T4 as “building blocks” for our multicriteria framework

[T2 - D2.2] A collection of software prototypes for succinct ML-models

Possible other collaborations are welcome