AutoEncoder by Forest

Machine Learning 2020 Course

Olga Novitskaia, Maria Begicheva, Egor Sevriugov, Kirill Shcherbakov

Skoltech

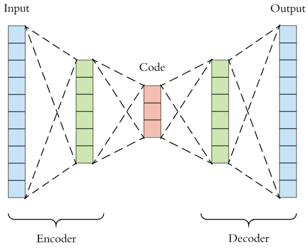
April 6, 2020

Plan

- Introduction
- Goal and related tasks
- Data description
- Algorithm and models
  - EncoderFores (eForest) AE
  - Multilayer perceptrons (MLP) based auto-encoders (AE)
  - Convolutional Neural Network (CNN) based AE
- Experiments and Results
  - Image Reconstruction
  - Computation Efficiency
  - Damage Tolerable
  - Model Reuse
- Conclusion

Intro

- AE is a class of models, compressed the input data to a lower-dimensional code and mapped it back to the original space with low reconstruction error as its objective[1].
- Ensemble learning is one of the most powerful techniques of improving the performance of a model, trained and combined various learners to solve a problem.
- eForest is original idea[2] by Feng J. and Zhou Z-H., tree ensemble based AE which can be trained in both supervised or unsupervised fashion.



https://blog.paperspace.com/autoencoder-image-compression-keras/

Skoltech
Skolkovo Institute of Science and Technology

Goal and related tasks

The main goal is to **check performance** of eForest AE.

Related tasks:

- Implementation of the eForest algorithm
- Demonstration of its usage in supervised and unsupervised setting
- Comparison with MLP and CNN based AE by the following criteria:
  - Accuracy
  - Efficiency
  - Damage-tolerance (resistance of the model to partial damage)
  - Reusability (application of the model trained from one dataset to the other dataset)

Data description

- **MNIST**[3]: $60,000$ grayscale 28×28 images for training and 10,000 for testing (784 dimensional vector per sample).
- **CIFAR-10**[4]: $50,000$ colored 32×32 images for training and $10,000$ colored images for testing (each image is in R^{1024} per channel).
- **Omniglot**[5]: $19,280$ grayscale 105×105 images (used just for testing).

The test size of each dataset was **reduced** to **1,000** images because of computationally expensive eForest performance evaluation.

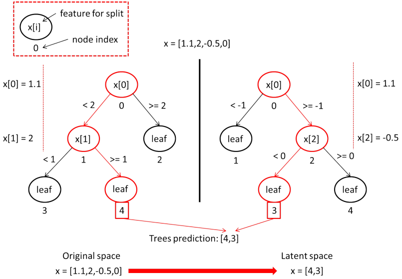
All models were evaluated on this test size.

eForest AE

Algorithm 1 Encoding

Input: trained forest F with T trees, object x
Output: x_{enc}
$x_{enc} = empty(T)$
for i in $range(T)$ **do**
  $x_{enc}[i] = F.tree[i].apply(x)$
  % "$apply$" returns the index of leaf in tree i corresponding to object x
end for
return x_{enc}

Encoding algorithm



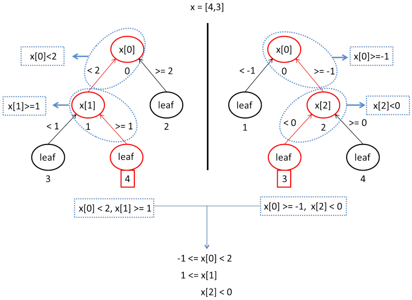
Encoding procedure

eForest AE

Algorithm 2 Calculate set of rules for tree

Input: tree t, n - leaf index of object for tree t
Output: rule set(array of size (N,2); for each feature min and max value)
% N - number of features in original space
$set = [[-infty, infty]\ for\ i\ in\ range(N)]$
$path = t.decision\ path(n)$
% path - array of nodes from root to leaf
for $node\ in\ path$ **do**
  $f = node.feature$
  $thres = node.threshold$
  if $next\ node\ in\ decision\ path\ is\ left$ **then**
    % the threshold for the split is upper bound
    $set[f][1] = min(set[f][1], thres)$
  else
    % the threshold for the split is lower bound
    $set[f][0] = max(set[f][0], thres)$
  end if
end for
return set

Set of rules for one tree



Decoding procedure

Skoltech
Skolkovo Institute of Science and Technology

eForest AE

Algorithm 3 MCR

Input: trained forest F with T trees, object x in latent space
Output: final set of rules
$result = [[-infty, infty]\ for\ i\ in\ range(N)]$
for $tree\ in\ F.trees$ **do**
  $set =$ Calculate set of rules for tree$(tree, x[tree])$
  $\%x[tree]$ returns component of x corresponding to $tree$
  $result = intersection(result, set)$
  $\%\ intersection(a, b)$ returns the intersection of rules a and b
end for
return $result$

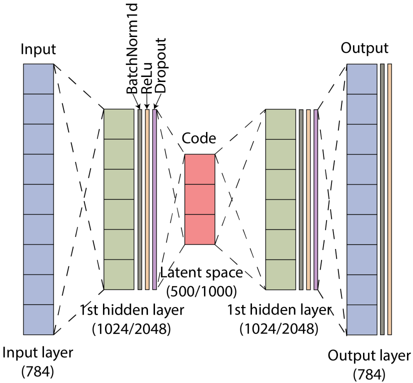
Maximal compatible rule(MCR)

Algorithm 4 Decoding

Input: trained forest F with T trees, object x in latent space
Output: x_{dec}
$x_{dec} = empty(N)$
% N - number of features in original space
$set = MCR(F, x)$
for $i\ in\ range(N)$ **do**
  if $set[i][0] == -infty\ and\ set[i][1] == infty$ **then**
    $x_{dec}[i] = median[i]$
    $\%\ median[i]$ - median value for feature i
  else if $set[i][0] \neq -infty\ and\ set[i][1] == infty$ **then**
    $x_{dec}[i] = set[i][0]$
  else if $set[i][0] == -infty\ and\ set[i][1] \neq infty$ **then**
    $x_{dec}[i] = set[i][1]$
  else
    $x_{dec}[i] = mean(set[i][0], set[i][1])$
  end if
end for
return x_{dec}

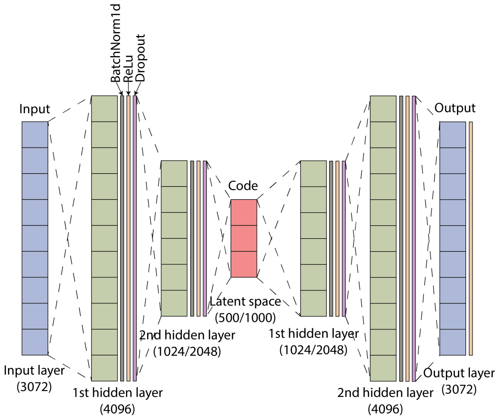
Decoding

MLP based AE



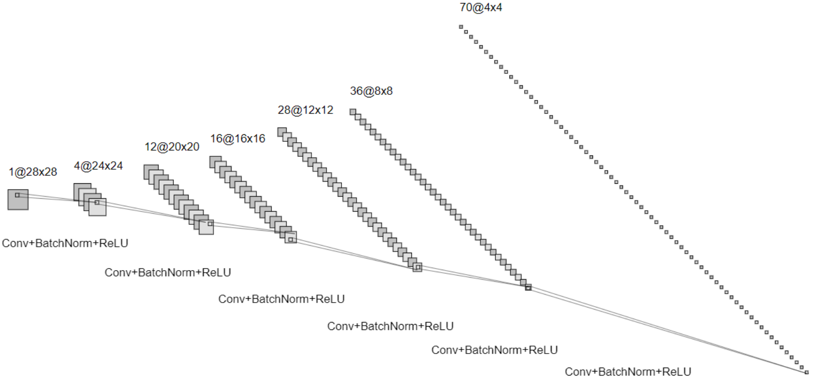
MLP_1/MLP_2 based AE architecture on MNIST

MLP based AE



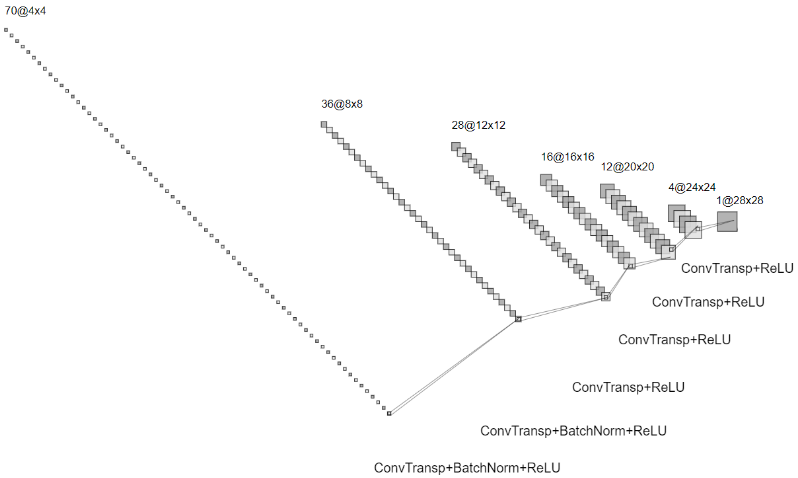
MLP_1/MLP_2 based AE architecture on CIFAR-10

CNN based AE



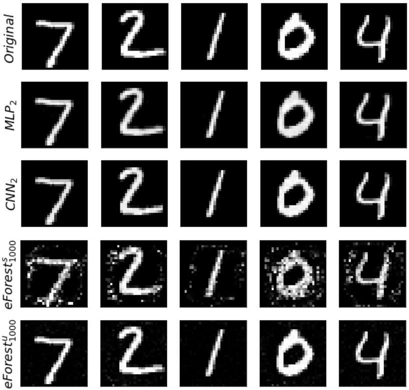
CNN encoder architecture on MNIST

CNN based AE

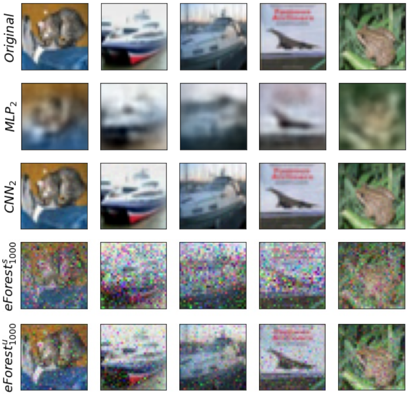


CNN decoder architecture on MNIST

Image Reconstruction



Reconstructed samples on MNIST



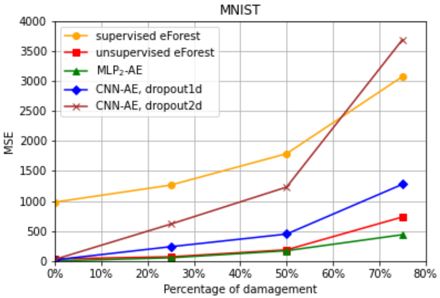
Reconstructed samples on CIFAR-10

Computation Efficiency

Time cost (in seconds). Decoding time is measured in sample per seconds.

MODEL	MNIST		CIFAR-10	
	TRAIN	DECODE	TRAIN	DECODE
MLP_1	1050	0.0003	1237	0.0004
MLP_2	1777	0.0003	1579	0.0005
CNN	2394	0.0006	7988	0.0007
eForest^s_{500}	268	0.9071	1180	1.7208
eForest^s_{1000}	561	1.4393	2333	3.0264
eForest^u_{500}	118	1.0096	90	1.7483
eForest^u_{1000}	223	1.8121	172	3.2822

Damage Tolerable



Performance on MNIST and CIFAR-10 when model is partially damaged

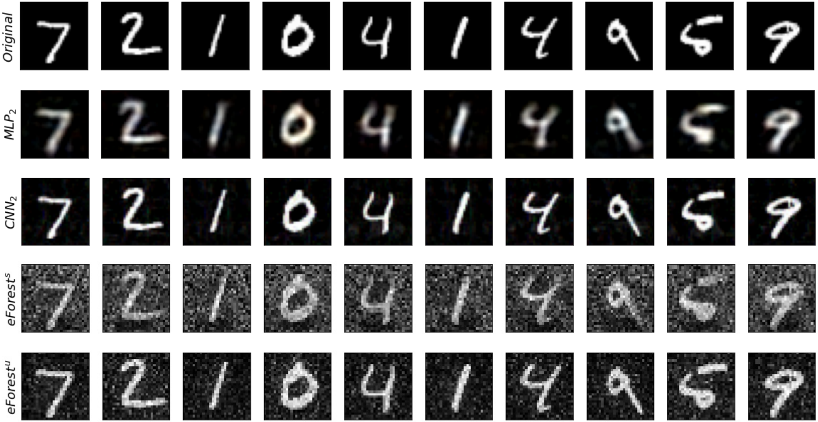
Skoltech
Skolkovo Institute of Science and Technology

Model Reuse

Performance comparison for model reuse (measured by MSE)

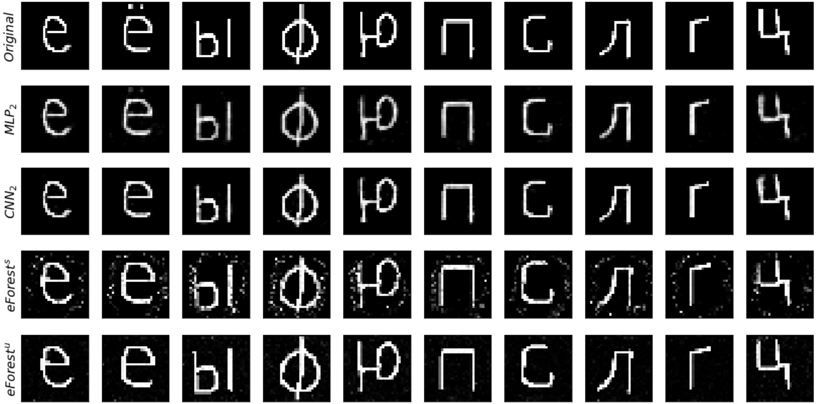
Model	cifar train mnist test	mnist train omniglot test	cifar train omniglot test
MLP_2	931.195	461.723	1855.313
CNN	123.392	210.599	269.476
eForest^s	4132.411	1173.529	4357.183
eForest^u	1438.885	103.896	1671.229

Model Reuse



Model reuse with training on CIFAR-10 and testing on MNIST

Model Reuse



Model reuse with training on MNIST and testing on Omniglot

Conclusion

- **Not accurate**: the eForest approach experimental reconstruction error is much higher than reasonable CNN based autoencoders, but comparable with MLP based autoencoders both on MNIST and CIFAR10 datasets.
- **Efficient**: the eForest on a many-core CPU runs even faster than a CNN autoencoder runs on a Tesla K80 GPU for training.
- **Not damage-tolerable**: the eForest is less resistant to partial damage than MLP and CNNs (to dropout specific neurons and to dropout channels in the layers).
- **Reusable**: In addition to replicating the original experiment, reusability of the eForest model were tested and compared additionally on a new experiment (train on CIFAR10 dataset and perform encoding/decoding task on Omniglot dataset), and the model trained from one dataset can be directly applied on the other dataset in the same domain.

Code and contact info

- https://github.com/Olga013/Skoltech-ML-2020-AutoEncoder-by-Forest/
- kirill.shcherbakov@skoltech.ru

References

1. Vincent, P., *et al*. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2013.
2. Feng, J. and Zhou, Z.-H. *Autoencoder by forest*. ArXiv, pp.1–14, 2017.
3. LeCun, Y., *et al*. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
4. Krizhevsky, A. Learning multiple layers of features from tiny images. *Tech. rep.*, University of Toronto, 2009.
5. Lake, B. M., *et al*. Human-level concept learning through probabilistic program induction. *Science*, 350(6266), 1332-1338, 2015.