Supervised Machine Learning under Constraints

Jean-Michel Begon

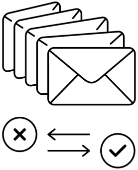
University of Liege, Belgium

21/12/2021

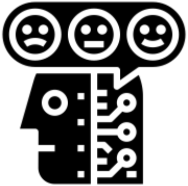
Introduction

Supervised learning—Common tasks

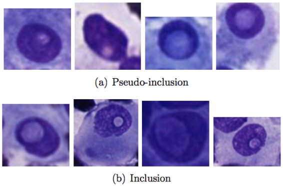
(a) Speech recognition.



(b) Spam detection.



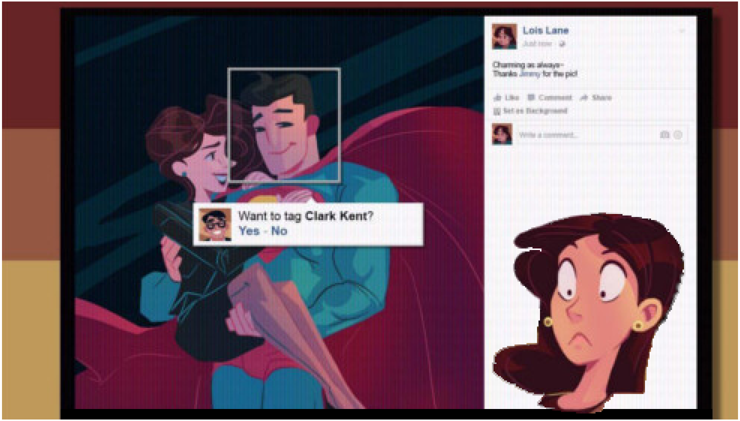
(c) Sentiment analysis.



(d) Medical diagnosis (Mormont et al., 2016).

Figure 1 Examples of tasks suited for supervised learning.

Supervised learning—Common tasks



(e) Face detection/recognition.

Figure 1 Examples of tasks suited for supervised learning.

Supervised learning—Overview

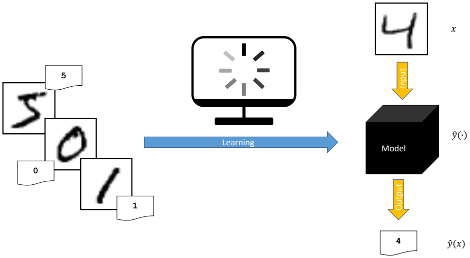
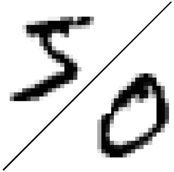


Figure 2 Schematic of supervised learning.

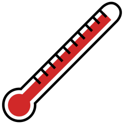
Classification

A few modalities.



Regression

A continuous scale.



Supervised learning example

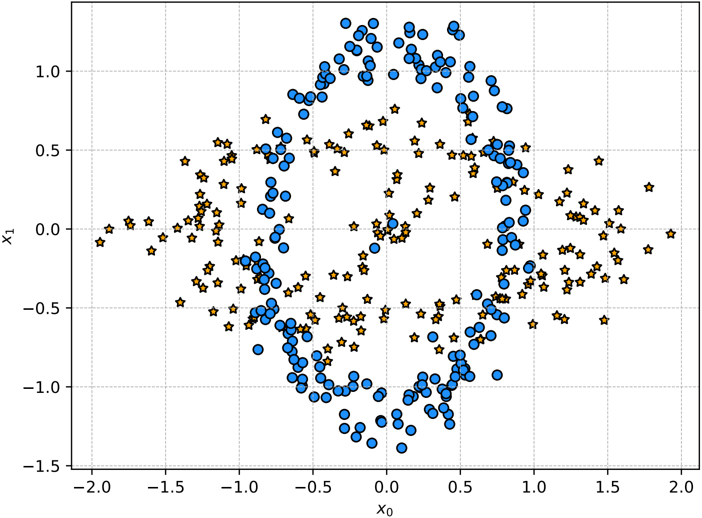
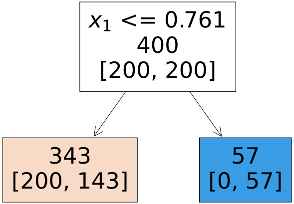
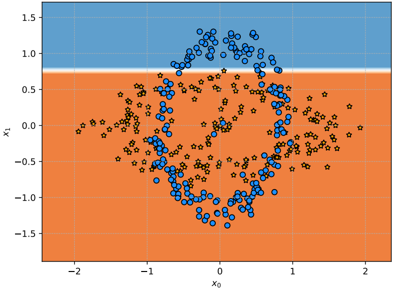


Figure 3 A classification problem.

Supervised learning example: decision tree



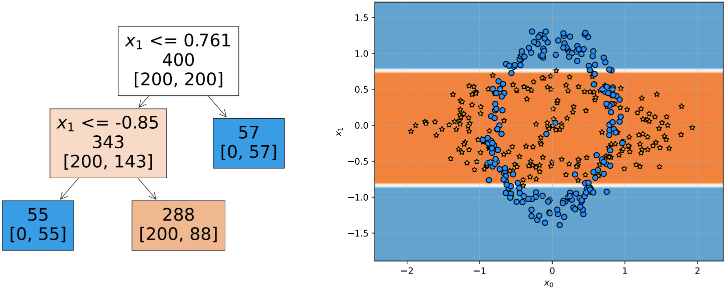
(a) Decision tree.



(b) Boundary and decision function.

Figure 4 A decision tree (maximum depth = 1) for the toy classification problem.

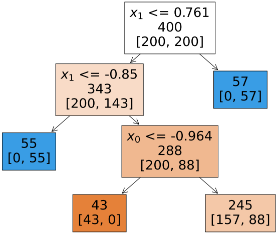
Supervised learning example: decision tree



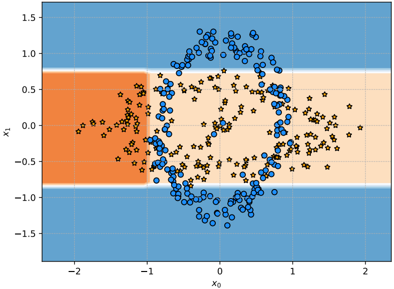
(a) Decision tree. (b) Boundary and decision function.

Figure 5 A decision tree (maximum depth = 2) for the toy classification problem.

Supervised learning example: decision tree



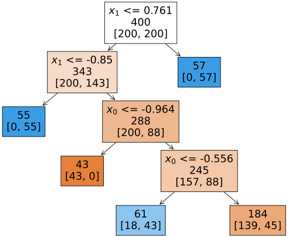
(a) Decision tree.



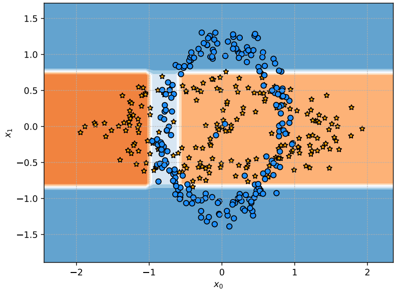
(b) Boundary and decision function.

Figure 6 A decision tree (maximum depth = 3) for the toy classification problem.

Supervised learning example: decision tree



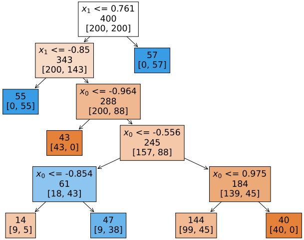
(a) Decision tree.



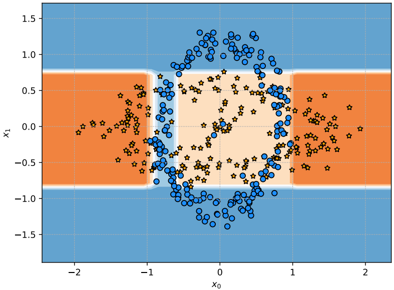
(b) Boundary and decision function.

Figure 7 A decision tree (maximum depth = 4) for the toy classification problem.

Supervised learning example: decision tree



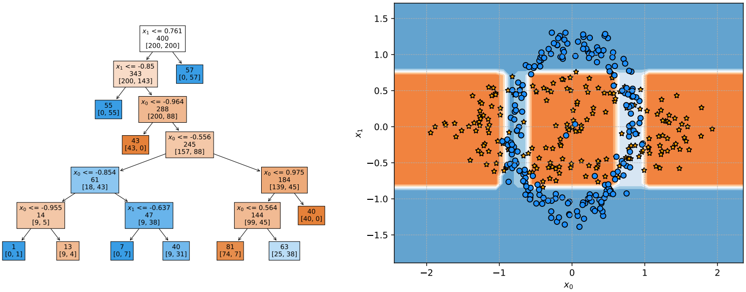
(a) Decision tree.



(b) Boundary and decision function.

Figure 8 A decision tree (maximum depth = 5) for the toy classification problem.

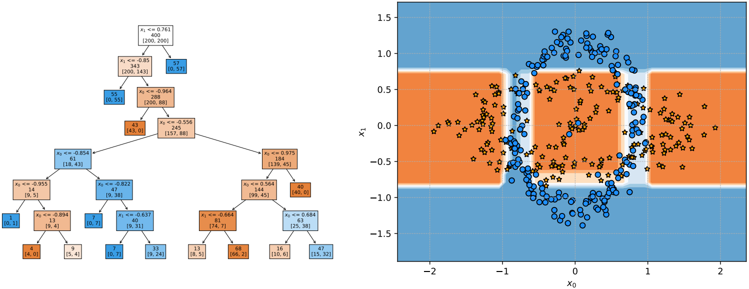
Supervised learning example: decision tree



(a) Decision tree. (b) Boundary and decision function.

Figure 9 A decision tree (maximum depth = 6) for the toy classification problem.

Supervised learning example: decision tree

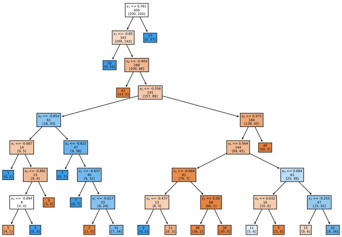


(a) Decision tree.

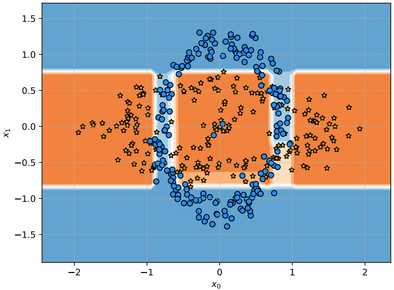
(b) Boundary and decision function.

Figure 10 A decision tree (maximum depth = 7) for the toy classification problem.

Supervised learning example: decision tree



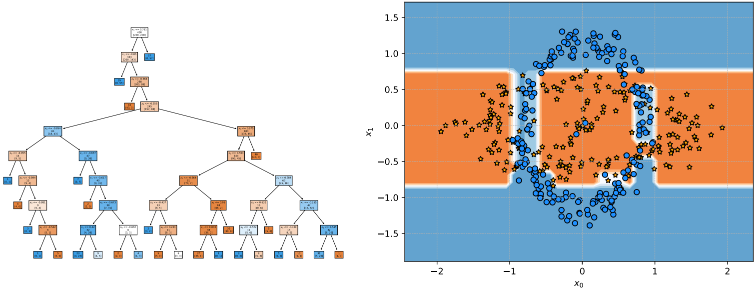
(a) Decision tree.



(b) Boundary and decision function.

Figure 11 A decision tree (maximum depth = 8) for the toy classification problem.

Supervised learning example: decision tree

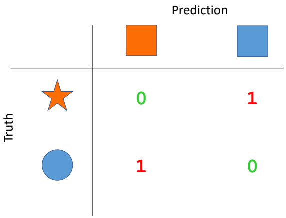


(a) Decision tree. (b) Boundary and decision function.

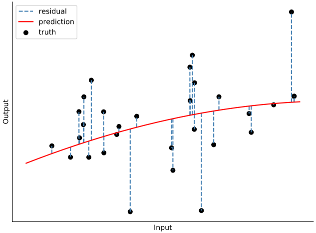
Figure 12 A decision tree (maximum depth = 9) for the toy classification problem.

Objective—loss ℓ and risk

Loss function ℓ



(a) Zero-one loss ℓ_{0-1}.



(b) Squared loss ℓ_2.

Figure 13

Minimizing the risk

$$\min_{\hat{y}(\cdot)\in\mathbb{H}} \mathbb{E}_{(x,y)\sim\mathcal{I}}\{\ell\left(y, \hat{y}(x)\right)\} \tag{1}$$

Objective—Overfitting

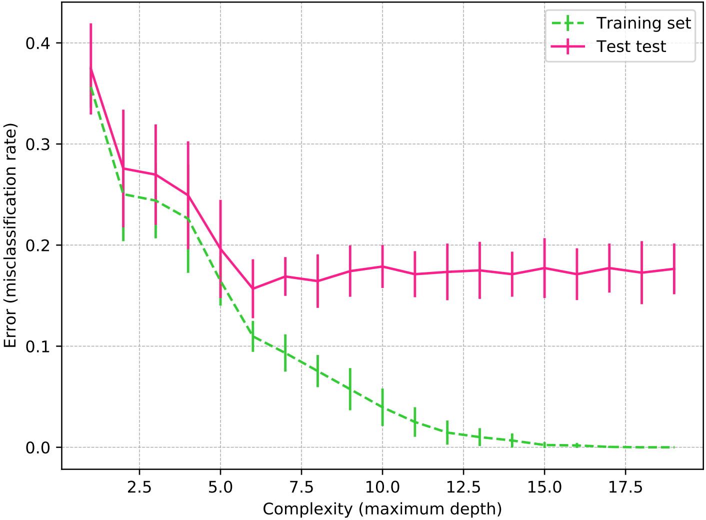


Figure 14 Generalization and re-substitution errors for the two-ellipses problem and decision tree.

Supervised learning under constraints

Supervised learning

Given data, find, with reasonable resources, the best model $\hat{y}(\cdot) \in \mathbb{H}$ for a problem according to some learning objective.

Constraints

Anything (extrinsic to the problem) which conditions or limits learning.

(a) Low latency. (b) Lack of data. (c) Interpretability.

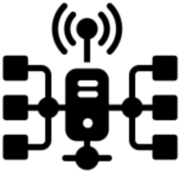
Figure 15

Contributions

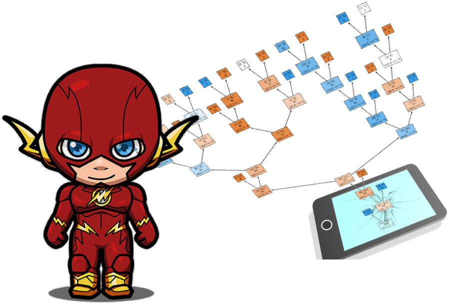
	Model	
	Small	n/a
Traditional learning	Forest pre-pruning (Chap. 6)	
Sample-free post-processing	Network compression (Chap. 8) ←	Enforcing robustness (Chap. 7)

Interpretability (Chap. 9)

Small models

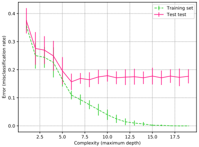


(a) Big data/hard problem.



(b) Speed.

(c) Energy.



(d) Reduced overfitting.

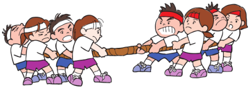
Figure 16 The “whys” of small models

Data unavailability

(a) Privacy.

(b) Size.

(c) Cost.



(d) Business reasons.

Figure 17 The “whys” behind data unavailability.

Outline

Globally Induced Forests

Outline

Foreword—Decision forest

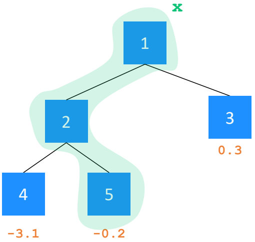


Figure 18 Prediction with a decision tree

Foreword—Decision forest

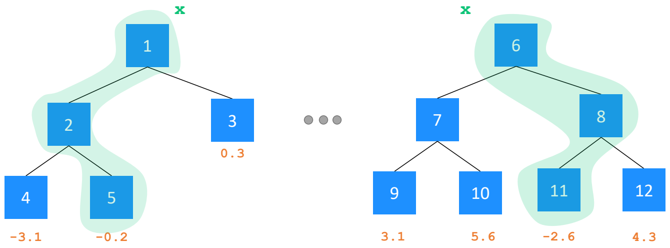


Figure 19 Prediction with a decision forest

Forest

Learning introduce randomness to produce different trees.

Prediction propagate to all trees and aggregate prediction.

Goal and motivation

What? Building accurate yet lightweight decision forests quickly (*i.e.* without building the whole model first).

Why? Decision forests are heavy models memory-wise:

- $\propto$ Number of nodes in a tree is (at worst) linear with the size of the data;
- $\propto$ number of required trees grows with the problem complexity.

How? Globally Induced Forests (GIFs):

- add one node at a time;
- choose globally.

GIF algorithm—Illustration

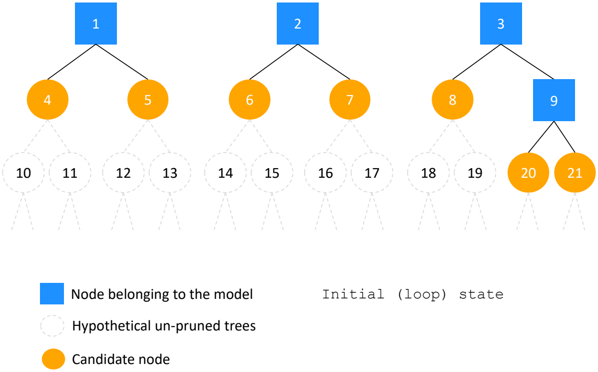


Figure 20 GIF algorithm: an illustration

GIF algorithm—Illustration

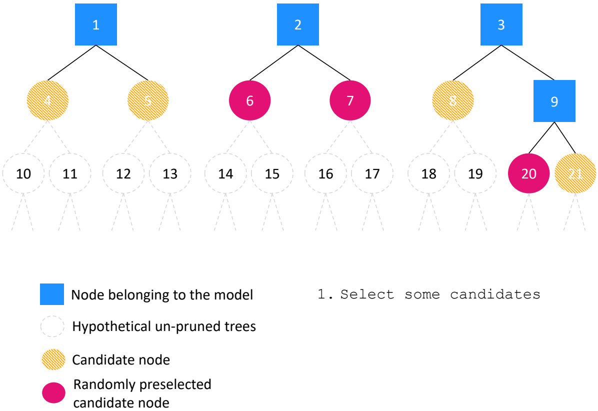


Figure 20 GIF algorithm: an illustration

GIF algorithm—Illustration

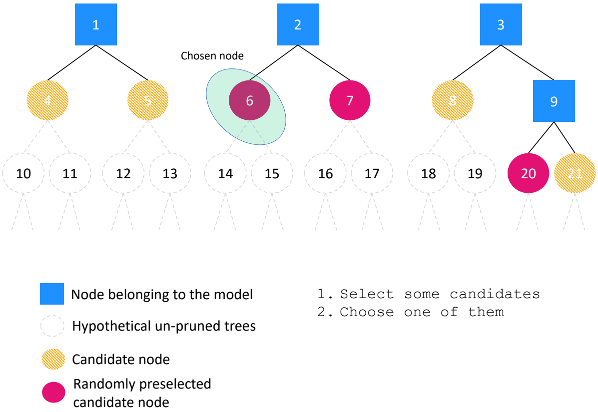
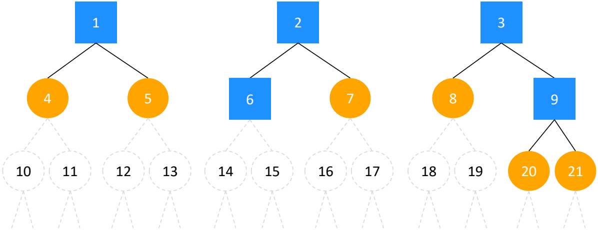


Figure 20 GIF algorithm: an illustration

GIF algorithm—Illustration



Node belonging to the model

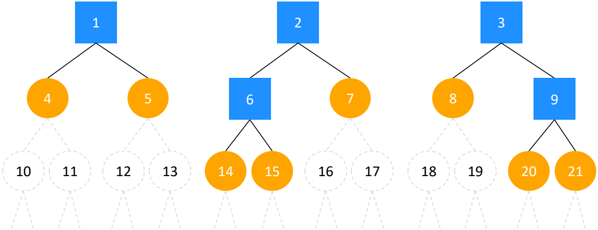
Hypothetical un-pruned trees

Candidate node

```
1. Select some candidates
2. Choose one of them
3. Add it to the model
```

Figure 20 GIF algorithm: an illustration

GIF algorithm—Illustration



Node belonging to the model
Hypothetical un-pruned trees
Candidate node

1. Select some candidates
2. Choose one of them
3. Add it to the model
4. Add its children (if any) to the candidate list

Figure 20 GIF algorithm: an illustration

GIF algorithm—Illustration

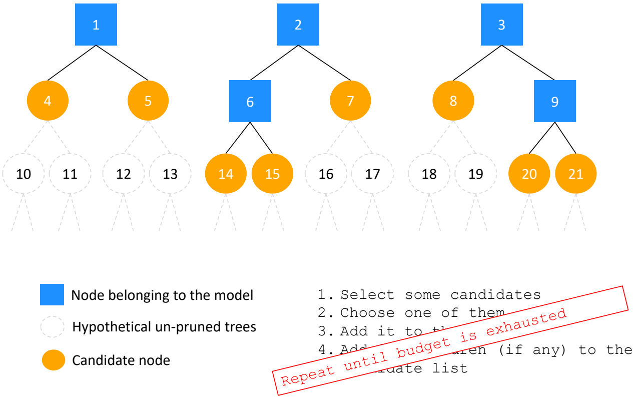


Figure 20 GIF algorithm: an illustration

GIF algorithm—Node selection: the forest space

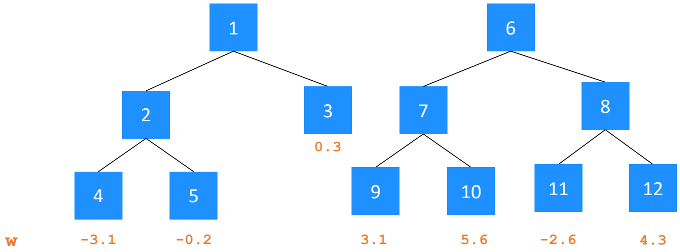
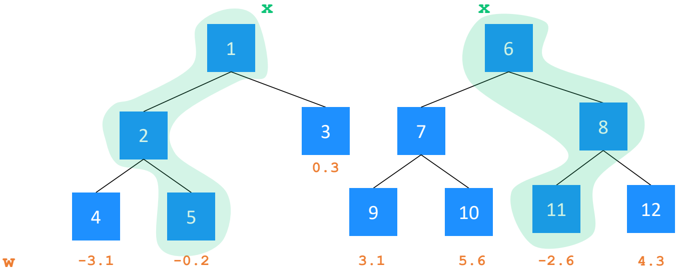


Figure 21 A decision forest

j	1	2	3	4	5	6	7	8	9	10	11	12
w_j	0	0	0.3	−3.1	−0.2	0	0	0	3.1	5.6	−2.6	4.3

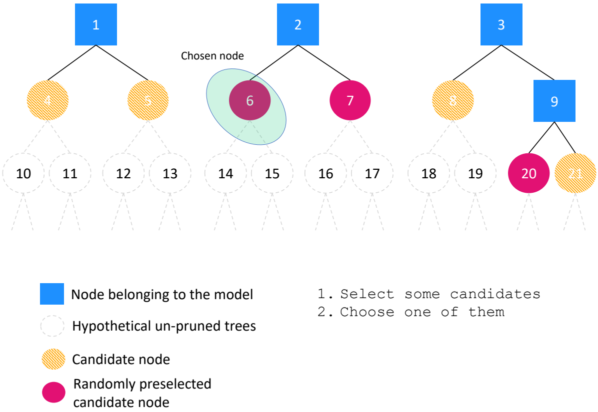
GIF algorithm—Node selection: the forest space



j	1	2	3	4	5	6	7	8	9	10	11	12
w_j	0	0	0.3	−3.1	−0.2	0	0	0	3.1	5.6	−2.6	4.3
$z_j(x)$	1	1	0	0	1	1	0	1	0	0	1	0
$w_j z_j(x)$	0	0	0	0	−0.2	0	0	0	0	0	−2.6	0

$$\hat{y}(x) = \sum_{j=1}^{12} w_j z_j(x) = -0.2 + -2.6 = -2.8 \tag{2}$$

GIF algorithm—Illustration




```
1. Select some candidates
2. Choose one of them
```

Figure 22 GIF algorithm: an illustration

GIF algorithm—Node selection: the global program

$$\hat{y}_{[t]}(x) = w_0 + \sum_{\tau=1}^{t} w_{j^{[\tau]}}^{[\tau]} z_{j^{[\tau]}}(x) = \hat{y}_{[t-1]}(x) + w_{j^{[t]}}^{[t]} z_{j^{[t]}}(x) \qquad (3)$$

where w_0 is the best constant over the learning set

GIF algorithm—Node selection: the global program

$$\hat{y}_{[t]}(x) = w_0 + \sum_{\tau=1}^{t} w_{j^{[\tau]}}^{[\tau]} z_{j^{[\tau]}}(x) = \hat{y}_{[t-1]}(x) + w_{j^{[t]}}^{[t]} z_{j^{[t]}}(x) \qquad (3)$$

where w_0 is the best constant over the learning set

The best node $j^{[t]}$, together with its optimal weight $w_{j^{[t]}}^{[t]}$, are the ones minimizing some loss ℓ over the training set $\{(x_i, y_i)\}_{i=1}^{n}$:

$$\left(j^{[t]}, w_{j^{[t]}}^{[t]}\right) = \underset{j \in C_{[t]}, w \in \mathbb{R}^K}{\arg\min} \sum_{i=1}^{n} \ell\left(y_i, \hat{y}_{[t-1]}(x_i) + w z_j(x_i)\right) \qquad (4)$$

where $C_{[t]}$ is the subsample of candidates.

GIF algorithm—Node selection: the global program

The problem is solved in two steps

1. for a candidate j, compute the best weight $w_j^{[t]}$ (closed form):

$$w_j^{[t]} = \arg\min_{w \in \mathbb{R}^K} \sum_{i=1}^{n} \ell \left(y_i, \hat{y}_{[t-1]}(x_i) + w z_j(x_i) \right) \tag{5}$$

2. select the best candidate (exhaustive search):

$$j^{[t]} = \arg\min_{j \in C_{[t]}} \sum_{i=1}^{n} \ell \left(y_i, \hat{y}_{[t-1]}(x_i) + w_j^{[t]} z_j(x_i) \right) \tag{6}$$

Results — Protocol

1. Grow a forest of a thousand fully-developed Extremely randomized trees ($ET_{100\%}$) and count the number of nodes M.
2. Compare how different methods fare (in average over ten runs) under a constraint of 1% and 10% of that budget.

   - $GIF_{x\%}$ grow the forest of a thousand trees until the node budget is met with the GIF algorithm.
   - $RAND_{x\%}$ grow a forest of a thousand trees randomly.
   - $ET_{x\%}$ grow only $10x$ fully-developed trees.

Hyper-parameters

$$\lambda = 10^{-1.5}$$
$$CW = 1$$
$$m = 1000$$

Splits: extremely randomized trees (ET), default hyper-parameters

Results — Regression

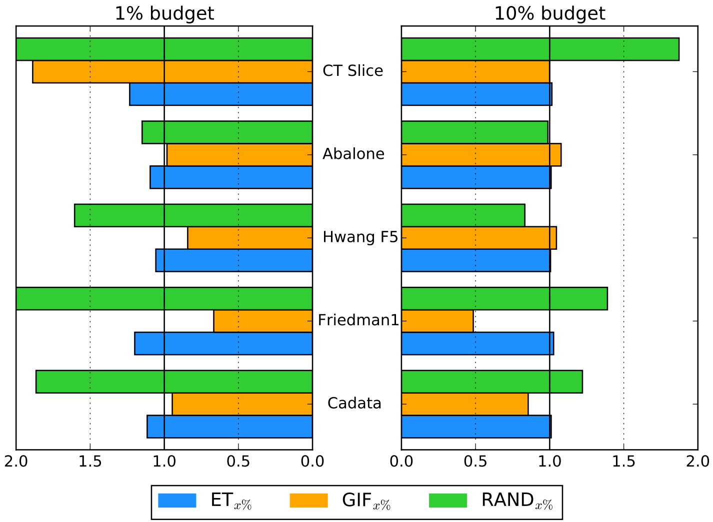


Figure 23 Relative average mean square error to $ET_{100\%}$.

Conclusion and future works

See thesis for more

- Experiments and discussion regarding hyper-parameters;
- comparison with more methods (baselines, post-pruning, boosting);
- producing interpretable models.

Take home message

- GIF allows for lightweight yet accurate forests;
- global optimization of the weight usually helps;
- optimizing the choice of node might lead to overfitting.

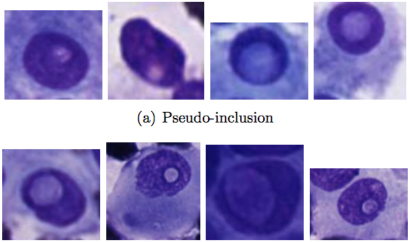
TODOs

- Handle multiclass problems better.

Sample-free Out-of-distribution detection

Outline

Out-of-distribution (OOD) detection

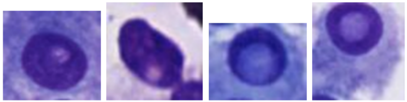


(a) Pseudo-inclusion

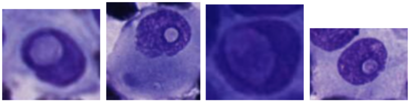
(b) Inclusion

Figure 24 Training data (from Mormont et al., 2016).

Out-of-distribution (OOD) detection



(a) Pseudo-inclusion



(b) Inclusion

Figure 24 Training data (from Mormont et al., 2016).

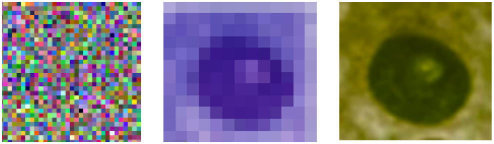


Figure 25 Anomalies.

Goal and motivation

What? Detecting OOD samples *a posteriori*, *i.e.* without data.

What for?
- Robustness;
- does the model know what it should receive as inputs?
- useful for other tasks.

Context? Deep networks for image classification.

How? With white-box indicators.

Deep learning 101

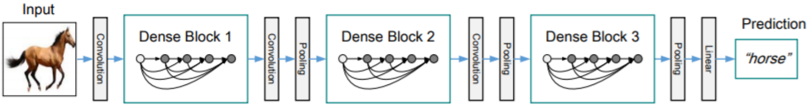


Figure 26 DenseNet (Huang et al., 2017), an example of a deep network.

$$\hat{p}(\cdot, \Theta) = \underbrace{\mathrm{softmax}(\cdot) \circ (W \cdot + b)}_{\text{softmax classifier}} \circ \underbrace{f_{L-1}(\cdot; \theta_{L-1}) \circ \ldots \circ f_1(\cdot; \theta_1)}_{\text{feature extractor } u(\cdot)} \quad (7)$$

The trainable weights $\Theta = [\theta_1, \ldots, \theta_{L-1}, (W, b)]$ are learned by gradient descent (over a cross-entropy loss).

Deep learning—Optimization (feature extractor)

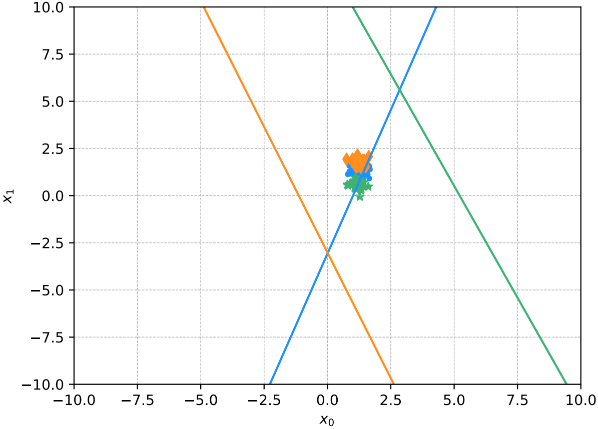


Figure 27 Feature extractor optimization (toy problem). At initialization.

Deep learning—Optimization (feature extractor)

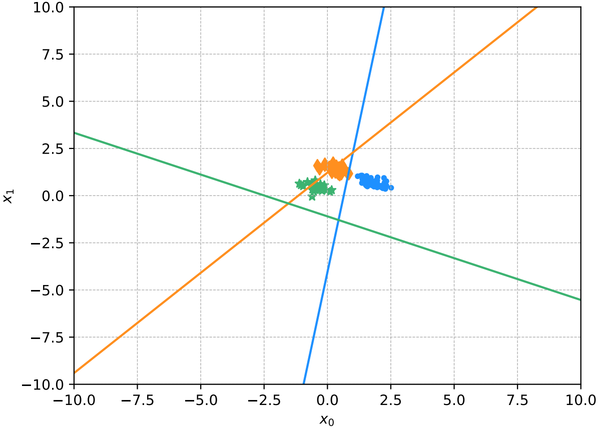


Figure 28 Feature extractor optimization (toy problem). After 10 iterations.

Deep learning—Optimization (feature extractor)

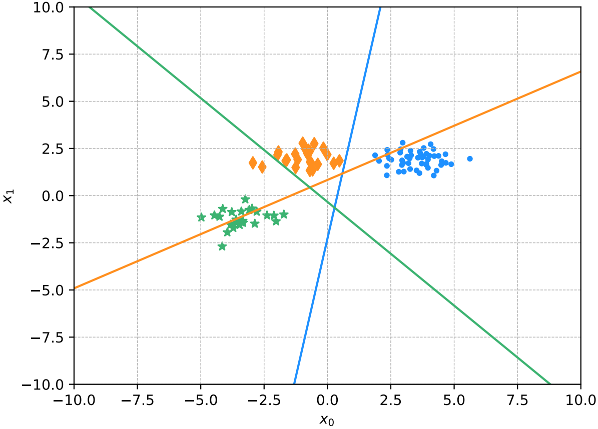


Figure 29 Feature extractor optimization (toy problem). After 15 iterations.

Deep learning—Optimization (feature extractor)

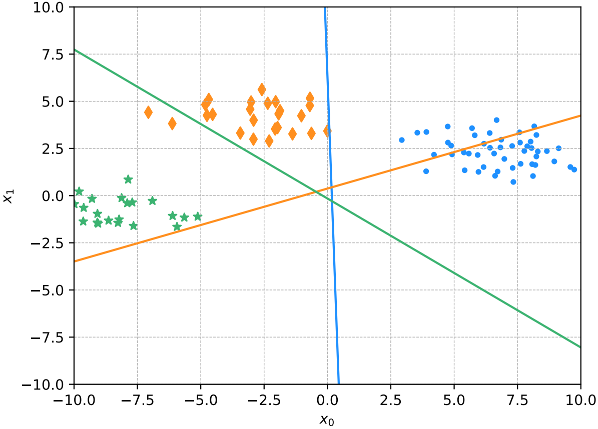


Figure 30 Feature extractor optimization (toy problem). At convergence.

OOD white-box indicators

- Sample-free;
- white-box: details of the model are known (Θ);
- indicator:

$$g(x;\Theta) \text{ is } \begin{cases} \text{low if } x \text{ is from the training distribution;} \\ \text{high otherwise.} \end{cases} \tag{8}$$

How?

OOD white-box indicators—baseline

Example (from Hendrycks and Gimpel, 2017):

$$MP(x) = 1 - \max_{1 \le j \le K} \hat{p}^{(j)}(x) \tag{9}$$

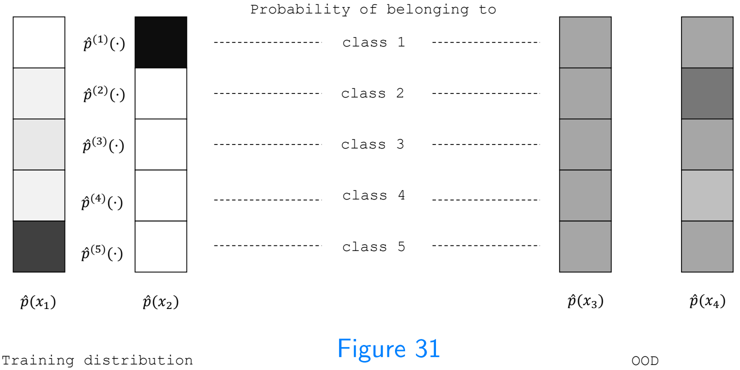


Figure 31

OOD white-box indicators—ang

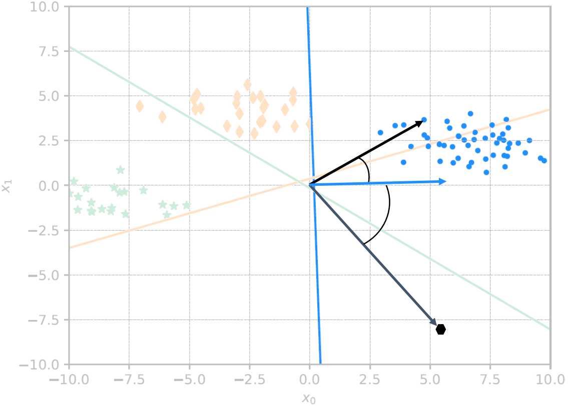


Figure 32

OOD white-box indicators—how

Optimality-based indicators (11)

`ODIN*`	`T1000`	`H`	`Proj`	`Norm`	`Norm+`
`Act`	`Act+`	`MP*`	`Ang`	`Ang++`	

Statistically-based indicators (7)

`In-DMS`	`In-DSS`	`DMS-AOS`	
`In-DMS-AOS`	`DSS`	`DMS`	`DSS-Ext`

Aggregation (1)

`1C-Sum`

Results—datasets

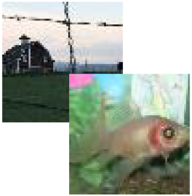
Figure 33 Original task: CIFAR 10.

(a) Gaussian. (b) SVHN. (c) MNIST.

Figure 34 OOD datasets.

Results—datasets

Figure 33 Original task: CIFAR 10.



(a) Tiny ImageNet.

(b) LSUN.

Figure 34 OOD datasets.

Results—main experiment

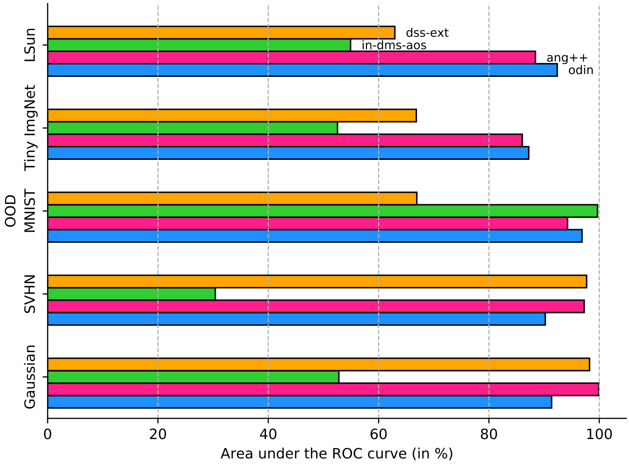


Figure 35 Performance of several indicators for OOD detection. Base task is CIFAR 10 on a DenseNet 50 network.

Results—main experiment

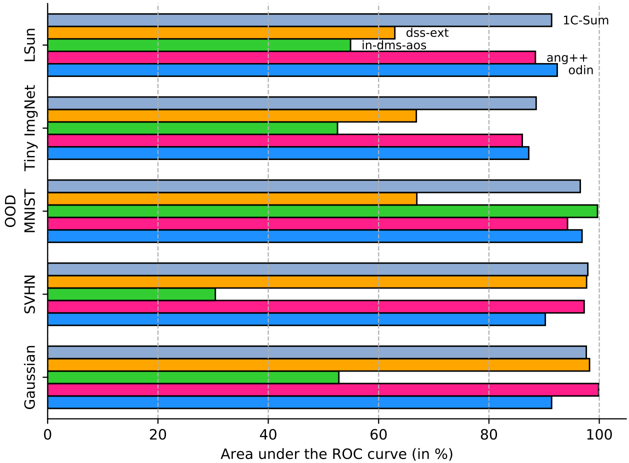


Figure 35 Performance of several indicators for OOD detection. Base task is CIFAR 10 on a DenseNet 50 network.

Conclusion

See thesis for more

- Redundancy analysis;
- discussion regarding the model quality;
- study of indicators for misclassifcation, with and without OOD detection;
- discussion on how to use indicators in practice.

Take home message

- Sample-free OOD detection works quite well on some problems;
- hard tasks require data.

TODOs

- Other indicators?

Distillation from heterogeneous collections

Outline

Goal and motivation

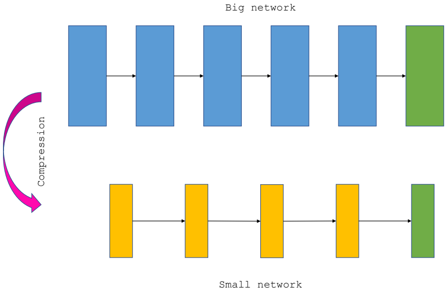


Figure 36 Compression.

What? Compress a big network into a lightweight one without data of training task.

Context? Deep networks for image classification.

How? Distillation from heterogeneous collections.

Distillation

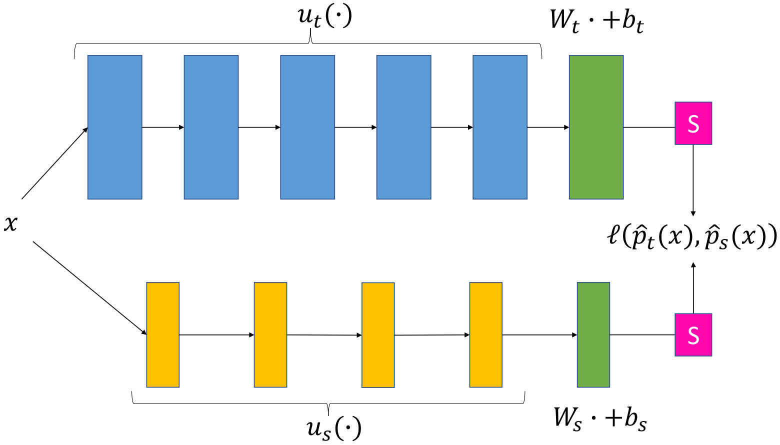


Figure 37 Teacher-student transfer: the memory requirements are met by choosing an appropriate student architecture.

Adaptations

No data

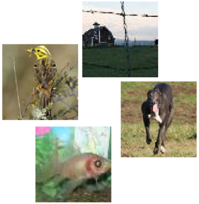
- heterogeneous collection.

Imperfect data

- learn more from teacher;
- focus on relevant data.

Collections

Figure 38 Original task: CIFAR 10.



(a) Relevant.

(b) Irrelevant.

Figure 39

Adaptations

No data

- heterogeneous collection.

Imperfect data

- learn more from teacher;
- focus on relevant data.

Distillation—Fixed softmax classifier

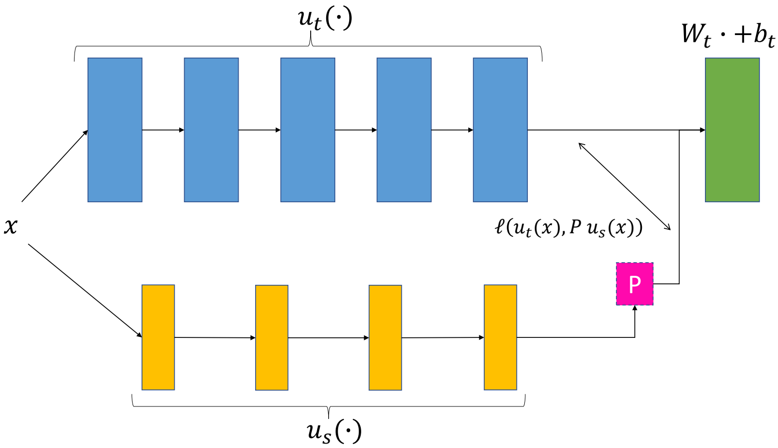


Figure 40 Fixed linear distillation (FL+P): learning the same feature extractor and keeping the softmax classifier of the teacher.

Results

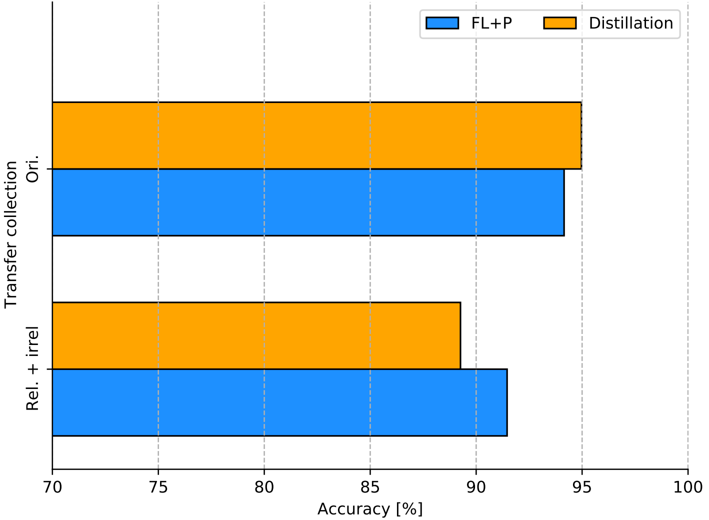


Figure 41 Performance of transfer from unlabeled collection. The task being transferred is CIFAR 10 from ResNet 50 to MobileNet v2.

Adaptations

No data

- heterogeneous collection.

Imperfect data

- learn more from teacher;
- focus on relevant data.

Distillation—biasing towards good data

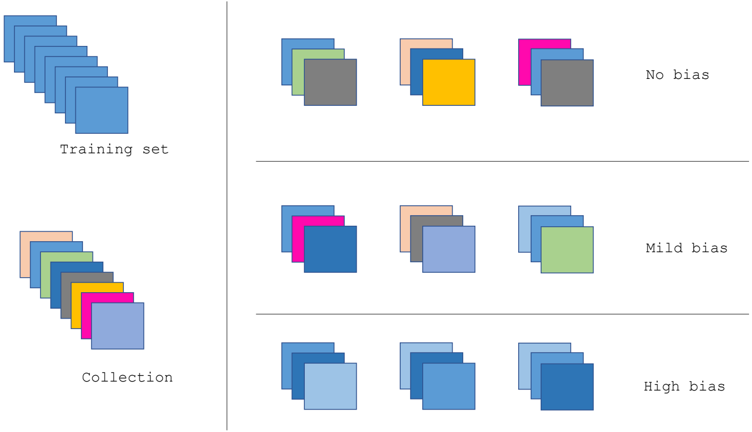


Figure 42 Biasing towards good data: select the data proportionally to how they "resemble" the training data (colors indicate resemblance).

Look for good data with an OOD indicator.

Results

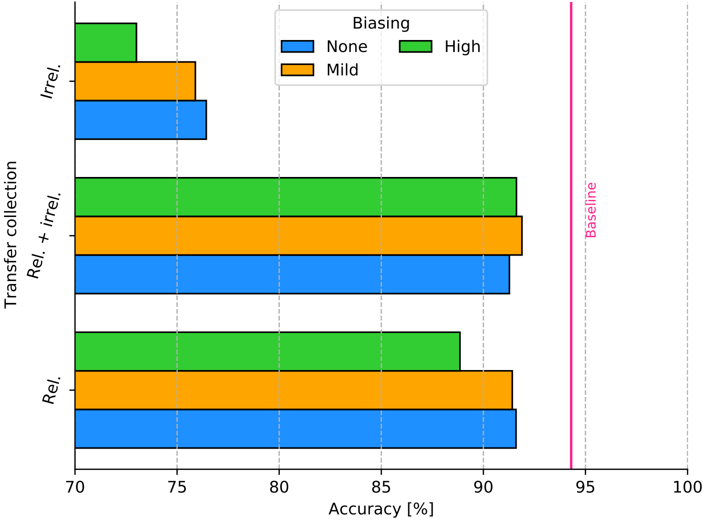


Figure 43 Performance of transfer from unlabeled collection. The task being transferred is CIFAR 10 from DenseNet 121 to MobileNet v2.

Results

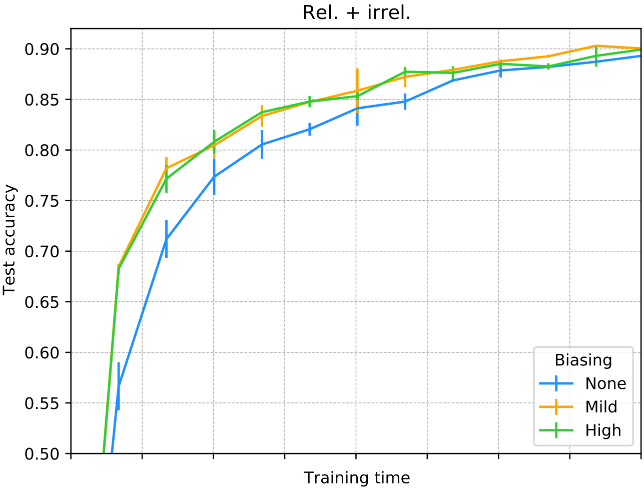


Figure 44 Convergence of transfer from unlabeled collection. The task being transferred is CIFAR 10 from DenseNet 121 to MobileNet v2.

Conclusion

See thesis for more

- Same collection, different base tasks;
- more teacher/student pairs;
- effects of the biasing mechanism.

Take home message

- Works quite well and (relatively) fast;
- relevant data > fixed-linear distillation > biasing

TODOs

- Improves the biasing mechanism;
- further improve transfer from teacher.

Conclusion

Conclusion

Supervised machine learning under constraints

- Producing small decision forests;
- detecting out-of-distribution samples in a sample-free regime;
- compressing a deep network without data;

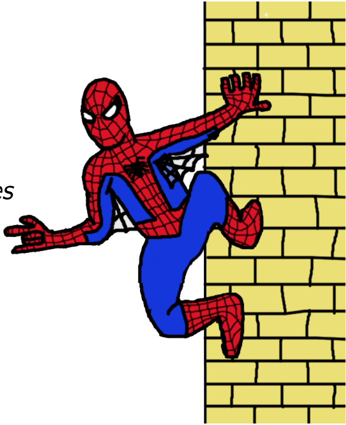
Overall take home message

Even though working with severe constraints is challenging, good results are achievable.

Meeting more and more constraints efficiently is the logical evolution.

Conclusion

With great success come great challenges



Backup

OOD—Results—Protocol

- Three architectures: DenseNet 121 (Huang et al., 2017), ResNet 50 (He et al., 2016), WideResNet (Zagoruyko and Komodakis, 2016);
- three base tasks: CIFAR 10, CIFAR 100 (Krizhevsky, Hinton, et al., 2009), ImageNet (Deng et al., 2009);
- several OOD datasets (pure noise, gray images, very different label space, close input statistics).
- metric: area under the ROC curve (auroc); aggregate of
  - OOD correctly identified (TPR);
  - ID taken for OOD (FPR).

True positive rate (y-axis): OOD catched rate False positive rate (x-axis): ID mistakenly taken for OOD

Distillation—Formally

Teacher

$$\hat{p}_t(\cdot, \Psi) = \text{softmax}(\cdot) \circ (W_t \cdot + b_t) \circ f_{t;L_t-1}(\cdot; \psi_{L_t-1}) \circ \ldots \circ f_{t;1}(\cdot; \psi_1) \tag{10}$$

$$= \text{softmax}(\cdot) \circ (W_t \cdot + b_t) \circ u_t(\cdot; \psi_{L_t-1:1}) \tag{11}$$

Student

$$\hat{p}_s(\cdot, \Theta) = \text{softmax}(\cdot) \circ (W_s \cdot + b_s) \circ u_t(\cdot; \theta_{L_t-1:1}) \tag{12}$$

Teacher-student transfer

$$\min_{\Theta} \mathbb{E}_{x \sim \mathcal{I}} \{\ell(\hat{p}_s(x, \Theta), \hat{p}_t(x, \Psi))\} \tag{13}$$

Meet the requirement (memory, latency, etc.) by choosing the student architecture properly.

Distillation—biasing towards good data

$$\mathbb{E}_{x\sim\mathcal{I}}\{\ell\left(\hat{p}_s(x,\Theta),\hat{p}_t(x,\Psi)\right)\} = \mathbb{E}_{x\sim\mathcal{O}}\{\beta(x)\,\ell\left(\hat{p}_s(x,\Theta),\hat{p}_t(x,\Psi)\right)\} \tag{14}$$

$$\beta(x) = \frac{\log \mathbb{P}_{\mathcal{I}}(x)}{\log \mathbb{P}_{\mathcal{O}}(x)} \tag{15}$$

Idea

Biasing the sampling select data randomly but proportionally to $\beta(x)$.

Characterizing score $\beta(x) \propto \frac{1}{\lambda} e^{g(x)}$

- λ controls the biasing;
- OOD indicator can be used as proxy for $g(x)$.

Distillation—Fixed softmax classifier

Idea

Increase the knowledge transfer (more information per sample) by

- learning only the feature extractor $u(\cdot)$;
- projecting the feature vectors onto the teacher latent space;
- keeping the same softmax classifier.

$$\begin{cases} W_s = PW_t \\ b_s = b_t \\ \min_{\theta,P} \mathbb{E}_x \, ||Pu_s(x;\theta) - u_t(x)||_2^2 \end{cases} \tag{16}$$

Distillation—Results—Protocol

- Two teacher architectures: DenseNet 121 (Huang et al., 2017) and ResNet 50 (He et al., 2016);
- Two student architectures: MobileNet v2 (Sandler et al., 2018) and ShuffleNet v2 (Ma et al., 2018);
- Two base tasks: CIFAR 10 (Krizhevsky, Hinton, et al., 2009) mainly and KMNIST (Clanuwat et al., 2018);
  - `Rel.`: Tiny ImageNet (Le and Yang, 2015) and STL 10 (Coates, Ng, and Lee, 2011);
  - `Irrel.`: MNISTx2 (LeCun et al., 1998), Fashion MNIST (Xiao, Rasul, and Vollgraf, 2017) and SVHN (Netzer et al., 2011).

Bibliography I

Clanuwat, Tarin et al. (2018). "Deep Learning for Classical Japanese Literature". In: CoRR abs/1812.01718. arXiv: 1812.01718. URL: http://arxiv.org/abs/1812.01718.

Coates, Adam, Andrew Ng, and Honglak Lee (2011). "An analysis of single-layer networks in unsupervised feature learning". In: Proceedings of the fourteenth international conference on artificial intelligence and statistics, pp. 215–223.

Deng, Jia et al. (2009). "Imagenet: A large-scale hierarchical image database". In: 2009 IEEE conference on computer vision and pattern recognition. Ieee, pp. 248–255.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

Bibliography II

- Hendrycks, Dan and Kevin Gimpel (2017). "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks". In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net. URL: https://openreview.net/forum?id=Hkg4TI9xl.
- Huang, Gao et al. (2017). "Densely connected convolutional networks". In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708.
- Krizhevsky, Alex, Geoffrey Hinton, et al. (2009). "Learning multiple layers of features from tiny images". In.
- Le, Ya and Xuan Yang (2015). "Tiny imagenet visual recognition challenge". In: CS 231N 7.7, p. 3.
- LeCun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: Proceedings of the IEEE 86.11, pp. 2278–2324.

Bibliography III

Ma, Ningning et al. (2018). "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design". In: Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV. Ed. by Vittorio Ferrari et al. Vol. 11218. Lecture Notes in Computer Science. Springer, pp. 122–138. DOI: 10.1007/978-3-030-01264-9_8. URL: https://doi.org/10.1007/978-3-030-01264-9_8.

Max Karpsten (2016). AI recognition drawing. [Online; accessed December 8, 2021]. URL: https://www.facebook.com/182161158594228/photos/a.182284588581885/912506332226370/?type=3.

Monkik (2019). Sentiment analysis drawing. [Online; accessed December 8, 2021]. URL: https://static.thenounproject.com/png/3383100-200.png.

Bibliography IV

Mormont, Romain et al. (2016). "SLDC: an open-source workflow for object detection in multi-gigapixel images". In.

Netzer, Yuval et al. (2011). "Reading digits in natural images with unsupervised feature learning". In.

Oleksandr Panasovskyi (2019). Spam detection drawing. [Online; accessed December 8, 2021]. URL: https://thenounproject.com/icon/email-spam-filter-2863991/.

Paula Helit (2019). Speech recognition drawing. [Online; accessed December 8, 2021]. URL: https://pixabay.com/vectors/voice-recognition-recognize-google-4414962/.

Bibliography V

Sandler, Mark et al. (2018). "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. IEEE Computer Society, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474. URL: http://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html.

Xiao, Han, Kashif Rasul, and Roland Vollgraf (Aug. 28, 2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv: cs.LG/1708.07747 [cs.LG].

Bibliography VI

Zagoruyko, Sergey and Nikos Komodakis (2016). "Wide Residual Networks". In: Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016. Ed. by Richard C. Wilson, Edwin R. Hancock, and William A. P. Smith. BMVA Press. URL: http://www.bmva.org/bmvc/2016/papers/paper087/index.html.

Credits I

Fig. 1a Paula Helit (2019). Speech recognition drawing. [Online; accessed December 8, 2021]. URL: https://pixabay.com/vectors/voice-recognition-recognize-google-4414962/

Fig. 1b Oleksandr Panasovskyi (2019). Spam detection drawing. [Online; accessed December 8, 2021]. URL: https://thenounproject.com/icon/email-spam-filter-2863991/

Fig. 1c Monkik (2019). Sentiment analysis drawing. [Online; accessed December 8, 2021]. URL: https://static.thenounproject.com/png/3383100-200.png

Fig. 1g Max Karpsten (2016). AI recognition drawing. [Online; accessed December 8, 2021]. URL: https://www.facebook.com/182161158594228/photos/a.182284588581885/912506332226370/?type=3

Credits II

Fig. 2 Hand-written digits taken from MNIST (LeCun et al., 1998)