Example

Large Scale Learning with Kernels
-
Understanding deep learning requires rethinking generalization

Corentin Ambroise Luis Montero Margaux Zaffran

Generalization properties

26/02/2020

Outline

Introduction

- Kernel Ridge Regression or Kernel Regularized Least Square (KRR / KRLS):

$$\min_f \mathbb{E}[(f(x) - y)^2]$$

$$\hat{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2$$

$$\hat{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n}(\langle f(\cdot), K_{x_i}\rangle - y_i)^2$$

$$f(x) = \sum_{i=1}^{n}\alpha_i K_{x_i}(x) \in \mathcal{H}_n \subset \mathcal{H}$$

Nyström approximation

- Computational constraint
  - Memory; $\mathcal{O}(n^2)$, Computation; $\mathcal{O}(n^3)$
- Idea: Approximate $\mathcal{H}_n$ by $\mathcal{H}_m$ smaller
- Sampling $\tilde{x}_k$ of size m of x_i
  - Uniform
  - Approximated Leverage Scores
- Find the optimal $\tilde{\alpha}$ such that $f(x) = \sum_{i=1}^{m} \tilde{\alpha}_i K(\tilde{x}_i, x)$ minimizes our loss
  - Closed form
  - Gradient Descent
  - Stochastic Gradient Descent
  - ...

Nyström approximation

Recent proposals

- Less is More: Nystrom Computational Regularization [3]
  - Theoretical results on the optimality of Nystrom approaches
  - The sub-sampling also regularize the model
- FALKON: An Optimal Large Scale Kernel Method [4] (3 authors in common with "Less is More")
  - They proposes another technique to optimize α base iterative solvers plus preconditioning
  - Scale training up

Idea

Theorem (Bochner)

A continuous kernel $k(x, y) = k(x - y)$ on $\mathbb{R}^d$ is positive definite if and only if $k(\delta)$ is the Fourier transform of a non-negative measure.

$$k(x - y) = \int_{\mathbb{R}^d} p(w) e^{jw'(x-y)} dw = \mathbb{E}_w[\xi_w(x)\xi_w(y)^*]$$

with $\xi_w(x) = e^{jwx}$ as defined in [1].

If p is scaled correctly, we then can generate w from p, and with $\varphi_w(x) = [\cos(wx)\sin(wx)]$, we have $\varphi_w(x)^T \varphi_w(y) \approx k(x - y)$.

Monte-Carlo simulation: $\frac{1}{D} \sum_{i=1}^{D} \varphi_{w_i}(x)^T \varphi_{w_j}(y)$.

Gaussian kernel

Example (gaussian kernel)

Let $G(x) = e^{-\frac{||x||^2}{2\sigma^2}}$, with $x \in \mathbb{R}^d$. Its Fourier tranform is given by $\widehat{G}(\xi) = (2\pi\sigma^2)^{\frac{d}{2}} e^{-\frac{\sigma^2||\xi||^2}{2}}$ (see [2] for details of the computation).

Numerical Study

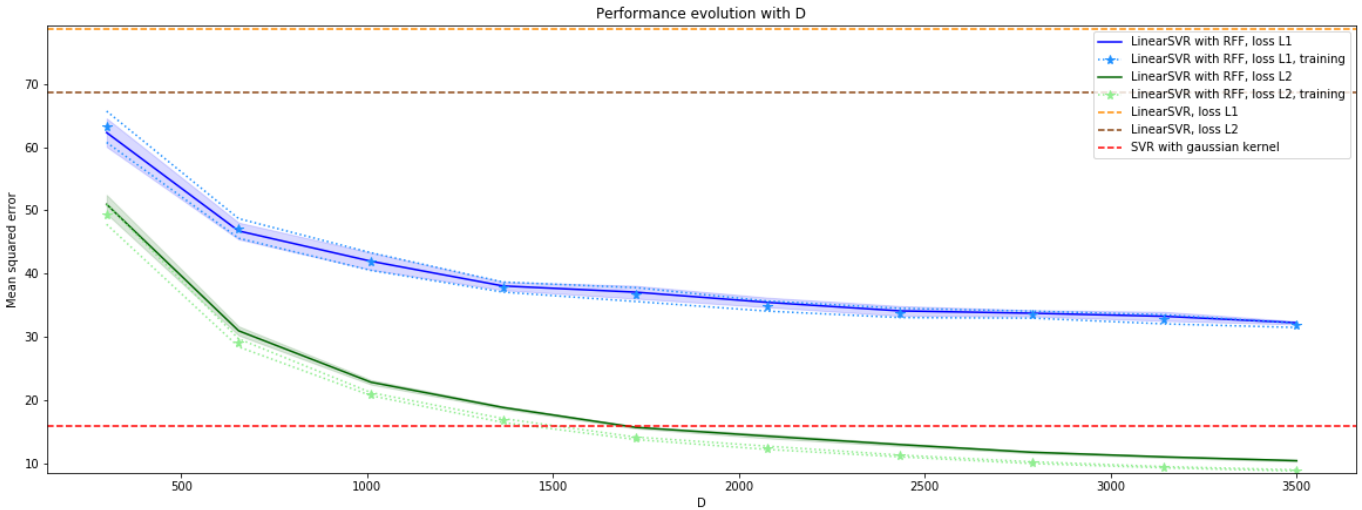
We chose to benchmark the different methods, on two datasets. There is a regression problem and a classification problem.

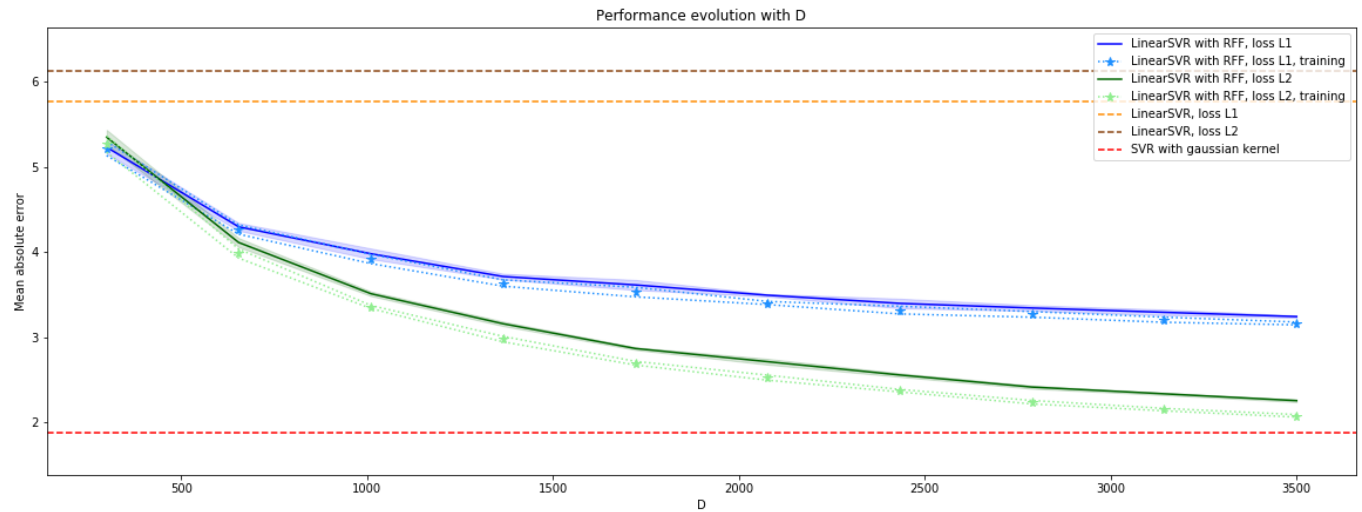
For each method, we studied what was their best parameter values for different type of regularization, for different losses according to the problem, while comparing with other algorithms.

Regression problem

The regression problem is a dataset of 53500 observations, each one of them has 385 features. The features are CT-scan acquisition parameters, the goal is to predict the height coordinate of the CT-Scan slice on the patient.

Random feature Fourrier





Random feature Fourrier

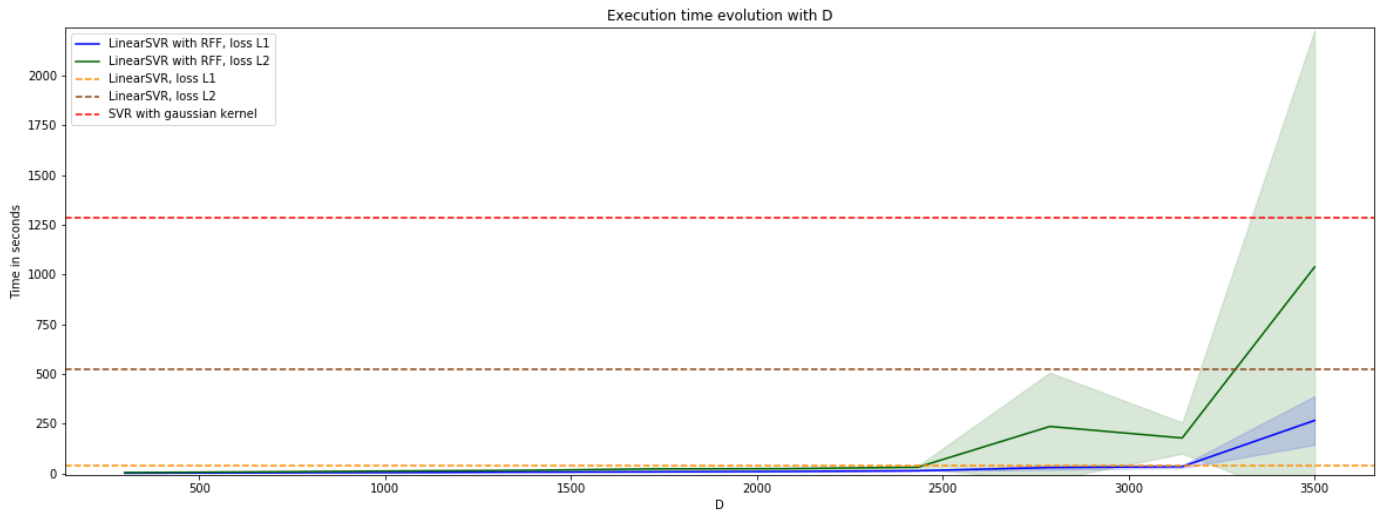
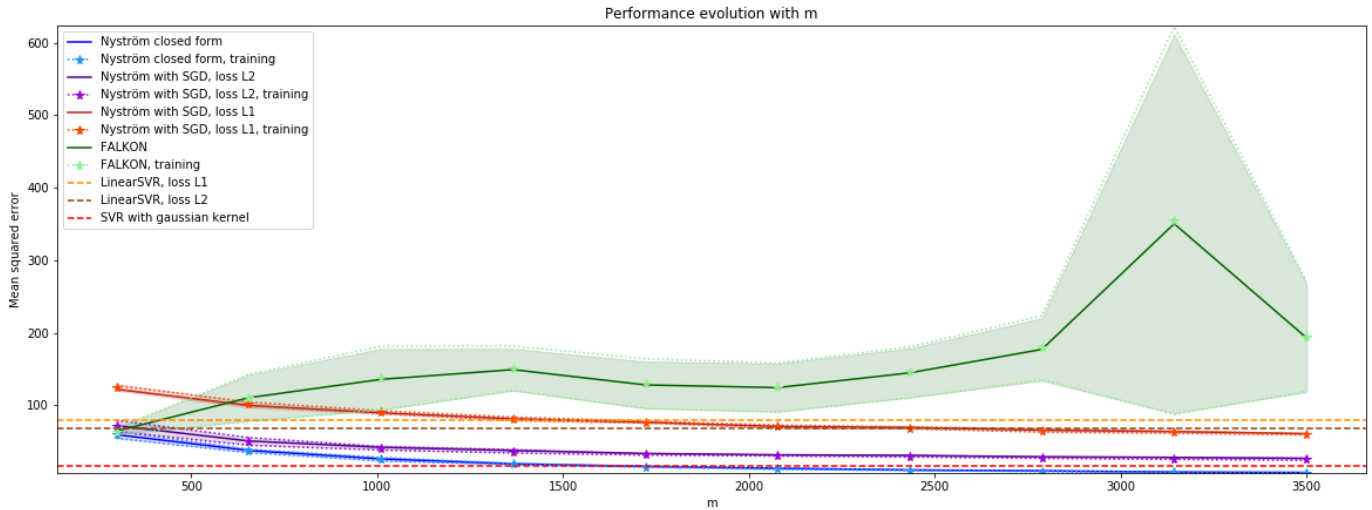
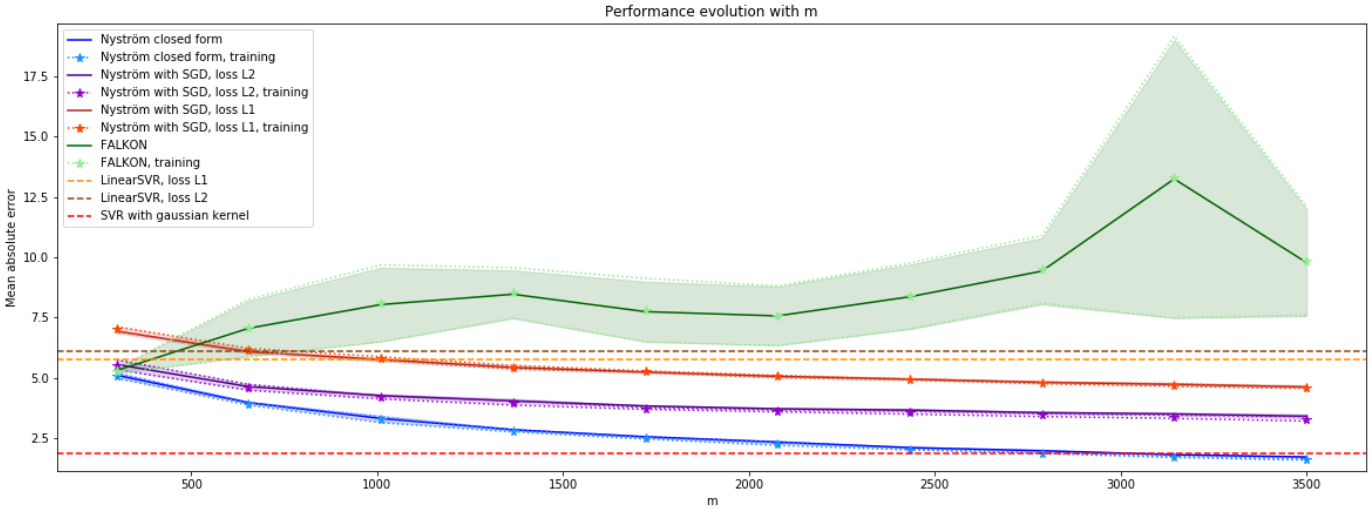


Figure 1: Training time against the number of feature

FALKON


Performance evolution with m
Nyström closed form
Nyström closed form, training
Nyström with SGD, loss L2
Nyström with SGD, loss L2, training
Nyström with SGD, loss L1
Nyström with SGD, loss L1, training
FALKON
FALKON, training
LinearSVR, loss L1
LinearSVR, loss L2
SVR with gaussian kernel
Mean squared error
m


Performance evolution with m
Nyström closed form
Nyström closed form, training
Nyström with SGD, loss L2
Nyström with SGD, loss L2, training
Nyström with SGD, loss L1
Nyström with SGD, loss L1, training
FALKON
FALKON, training
LinearSVR, loss L1
LinearSVR, loss L2
SVR with gaussian kernel
Mean absolute error
m

Nystrom and FALKON

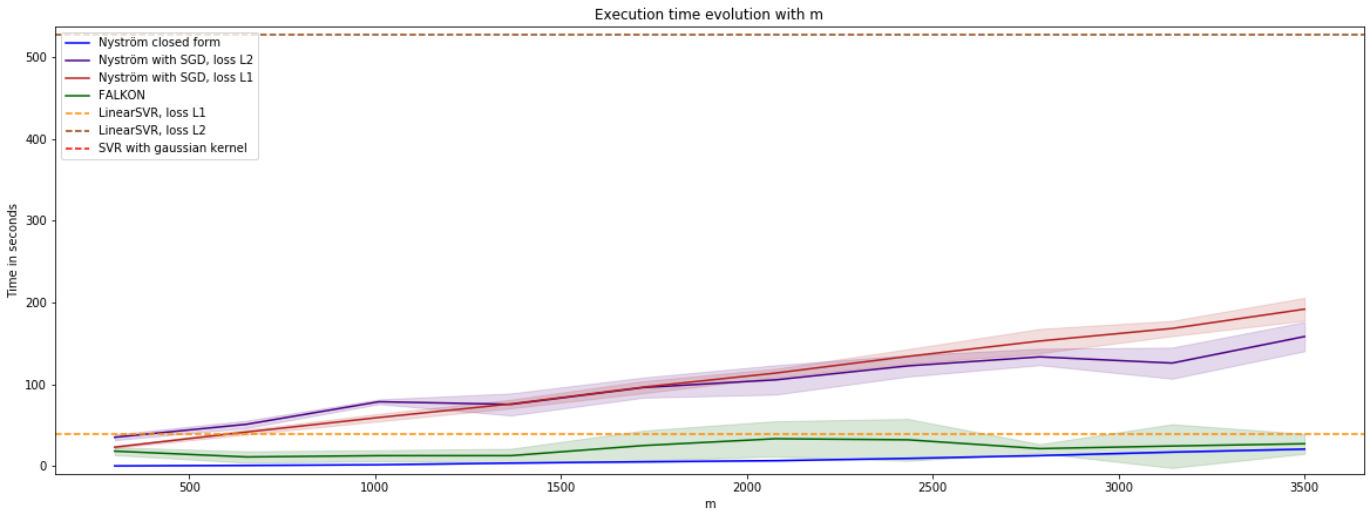
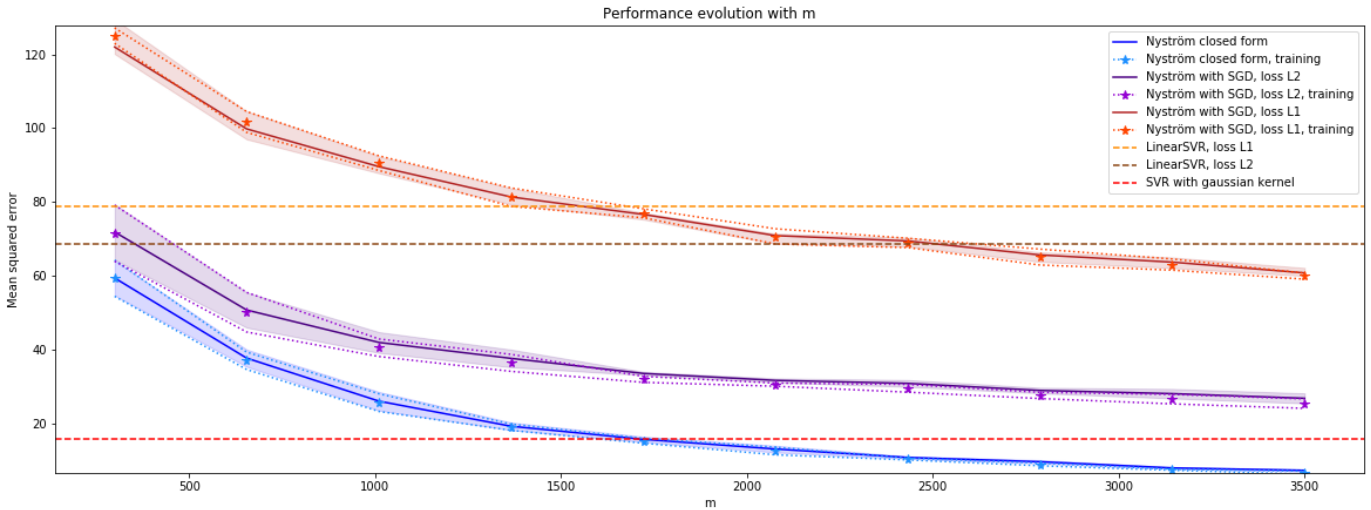
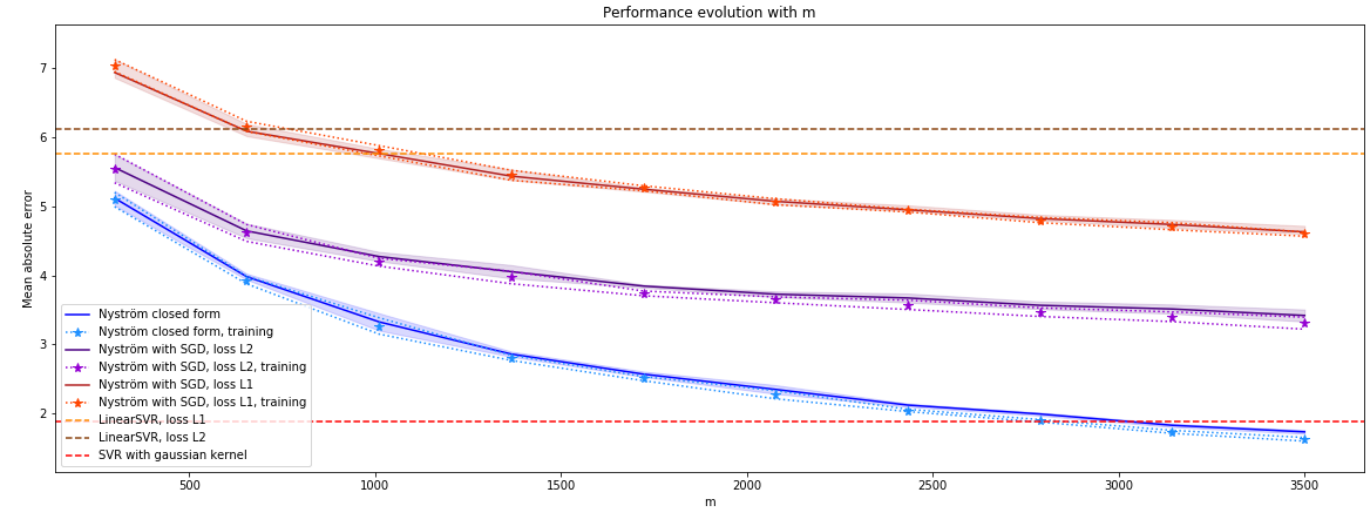


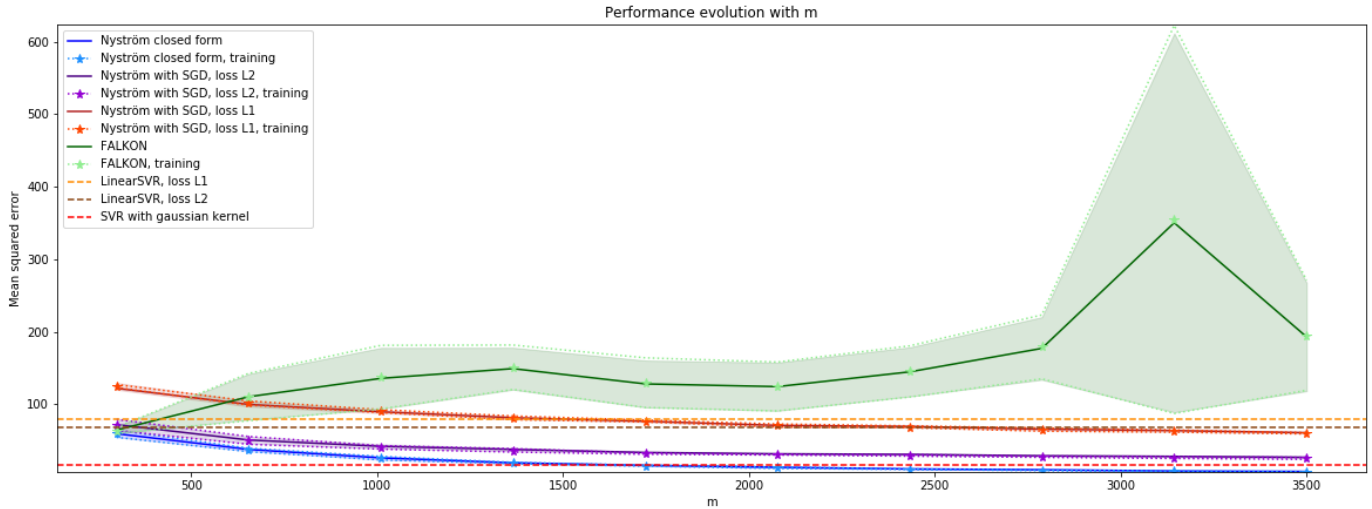
Figure 2: Training time against the number of feature

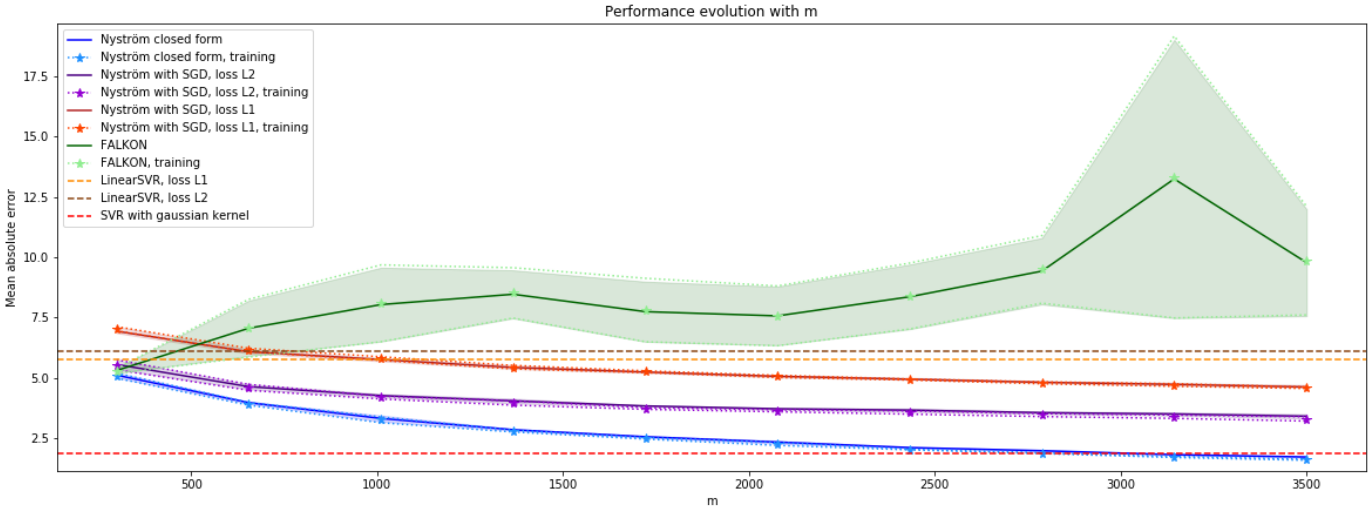
Nystrom


Performance evolution with m
Mean squared error
m
Nyström closed form
Nyström closed form, training
Nyström with SGD, loss L2
Nyström with SGD, loss L2, training
Nyström with SGD, loss L1
Nyström with SGD, loss L1, training
LinearSVR, loss L1
LinearSVR, loss L2
SVR with gaussian kernel

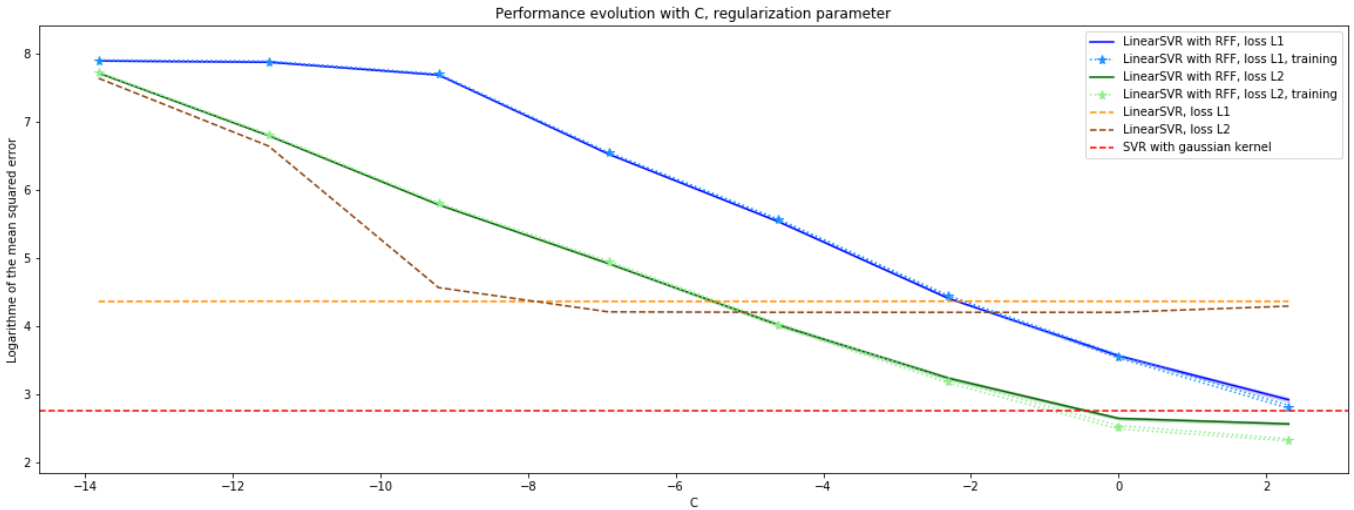

Performance evolution with m
Mean absolute error
m
Nyström closed form
Nyström closed form, training
Nyström with SGD, loss L2
Nyström with SGD, loss L2, training
Nyström with SGD, loss L1
Nyström with SGD, loss L1, training
LinearSVR, loss L1
LinearSVR, loss L2
SVR with gaussian kernel

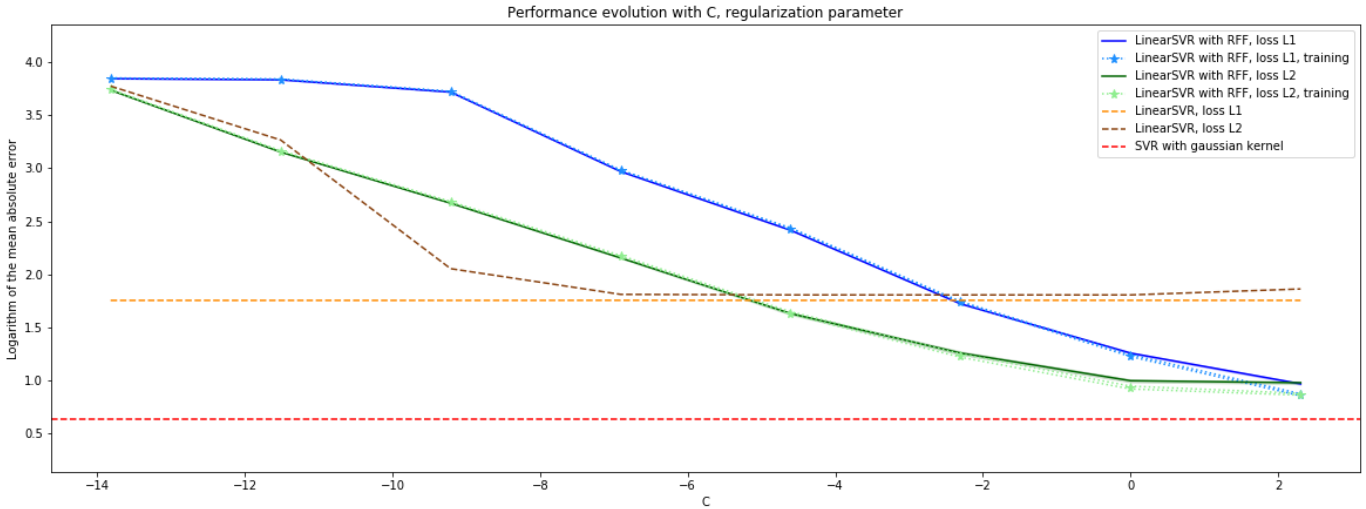
FALKON


Performance evolution with m
Nyström closed form
Nyström closed form, training
Nyström with SGD, loss L2
Nyström with SGD, loss L2, training
Nyström with SGD, loss L1
Nyström with SGD, loss L1, training
FALKON
FALKON, training
LinearSVR, loss L1
LinearSVR, loss L2
SVR with gaussian kernel
Mean squared error
m


Performance evolution with m
Nyström closed form
Nyström closed form, training
Nyström with SGD, loss L2
Nyström with SGD, loss L2, training
Nyström with SGD, loss L1
Nyström with SGD, loss L1, training
FALKON
FALKON, training
LinearSVR, loss L1
LinearSVR, loss L2
SVR with gaussian kernel
Mean absolute error
m

Regularization with RFF





Regularization with RFF

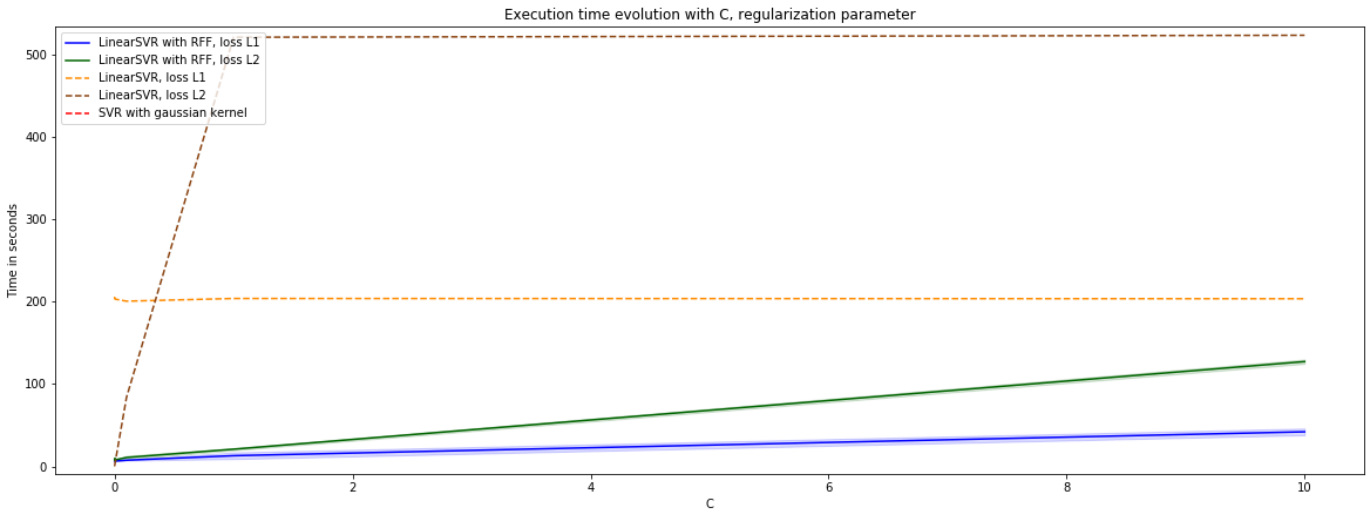
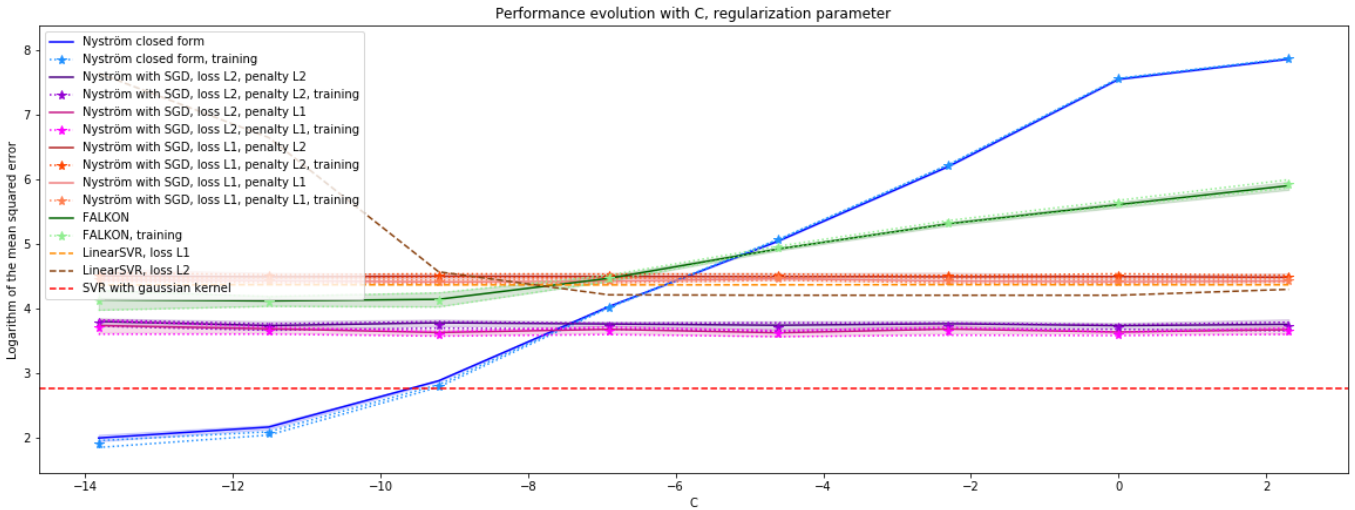
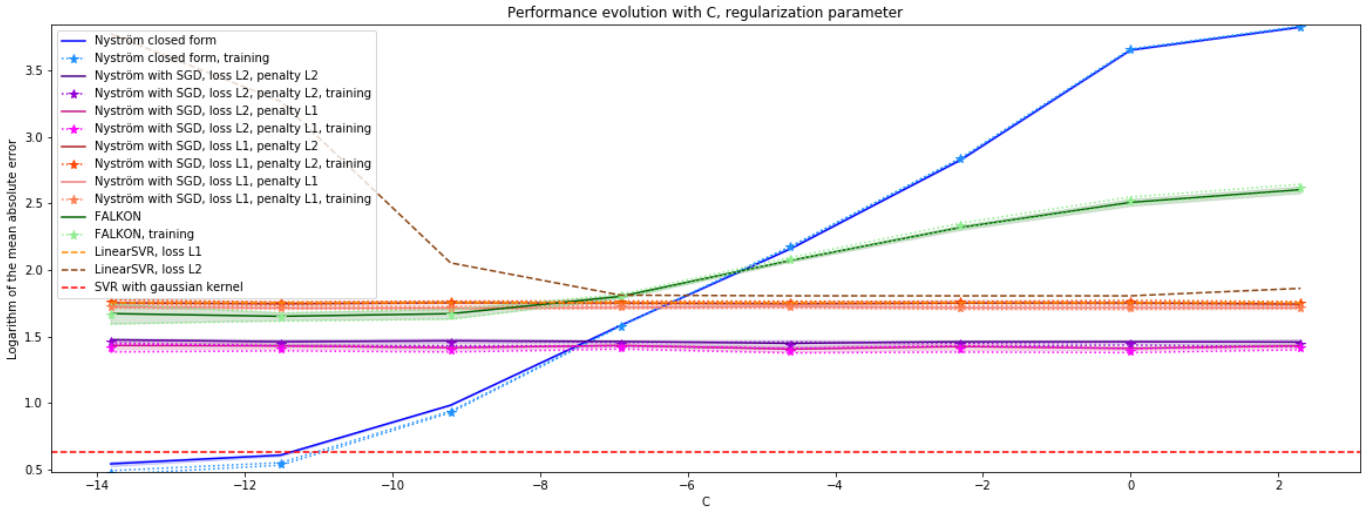


Figure 3: Training time against regularization strength

Regularization with Nystrom adn FALKON





Regularization with Nystrom and FALKON

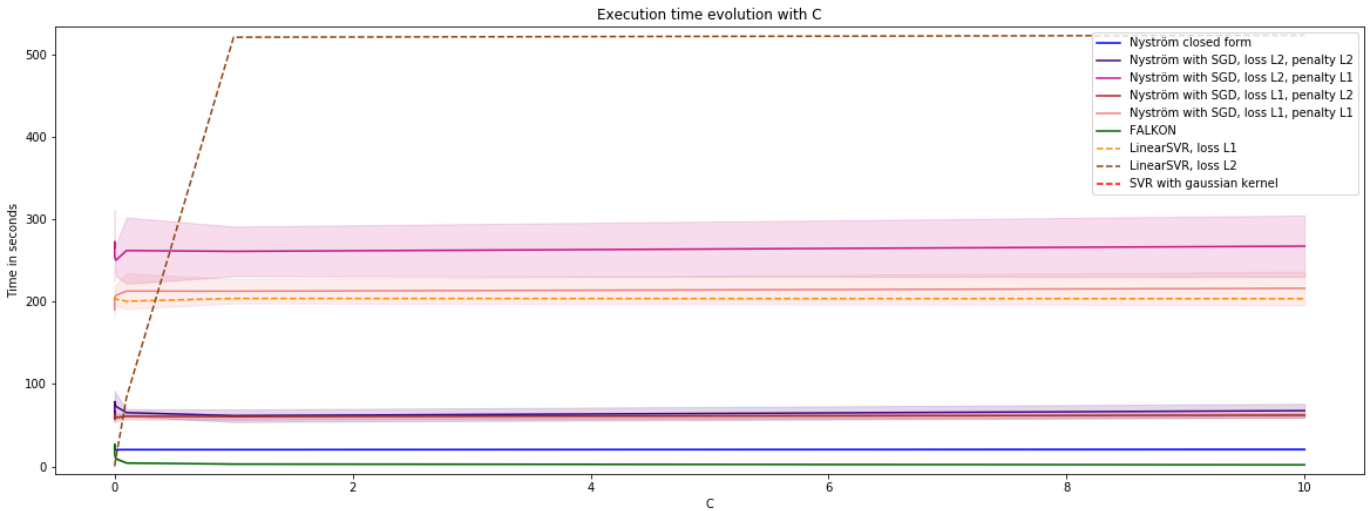


Figure 4: Training time against regularization strength

Classification

The classification problem is the one provided by the *dataset* API of scikit-learn with the function `make_classification`. We simulated this way 50000 samples with 120 features, 15 of them being informative. It is a binary classification problem.

Regularization with RFF

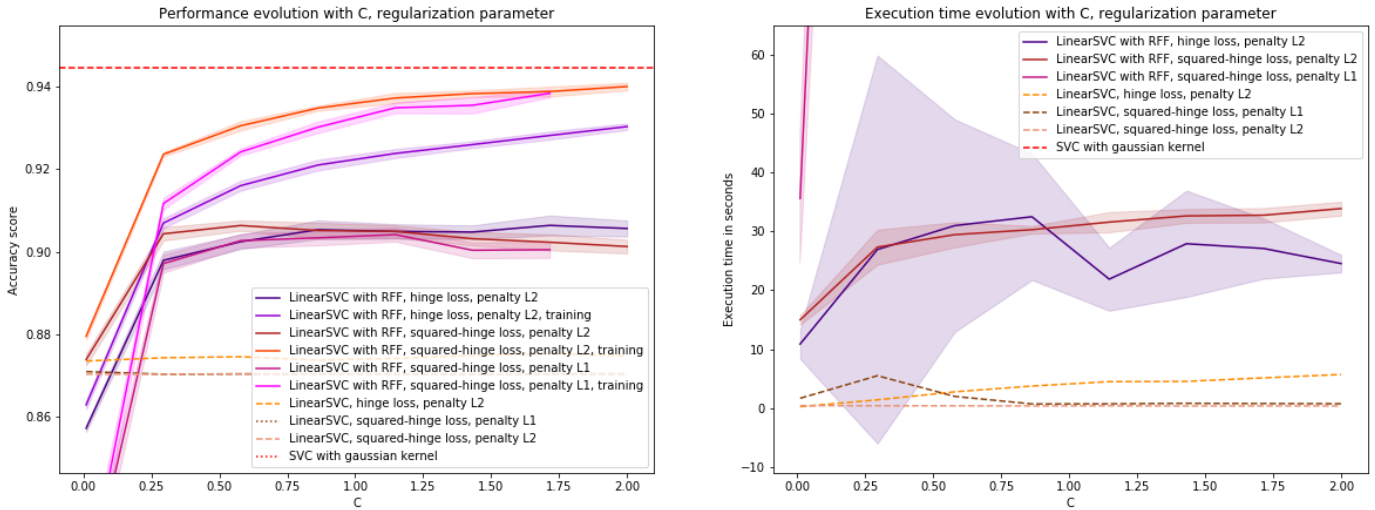


Figure 5: Performance and training time against the regularization strength

Regularization with Nystrom

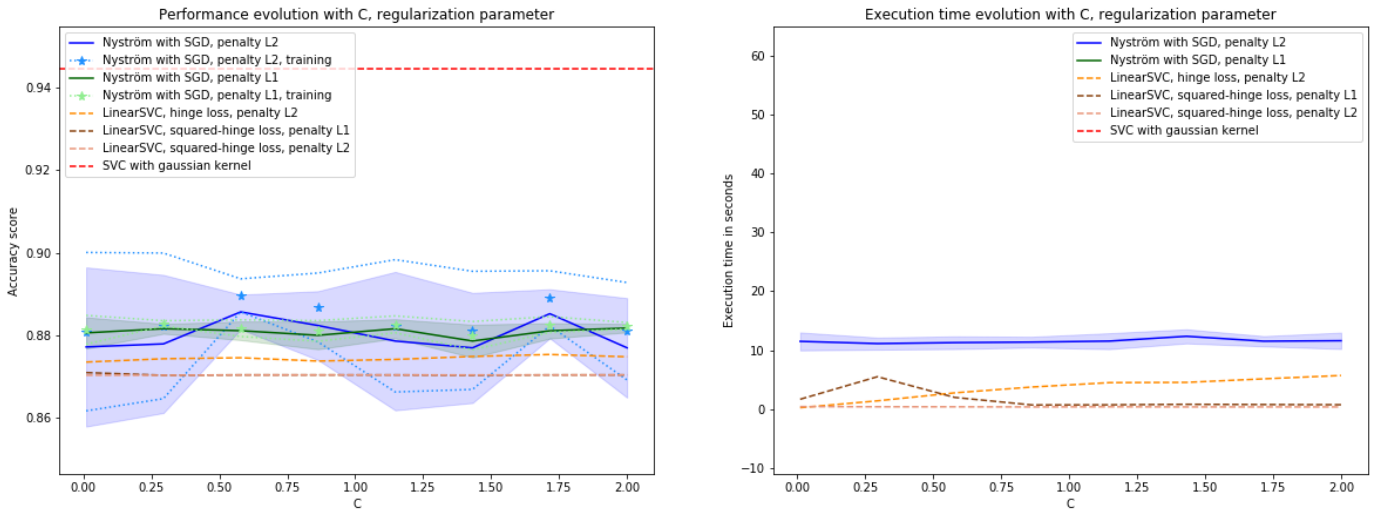


Figure 6: Training time against regularization strength

Benchmark

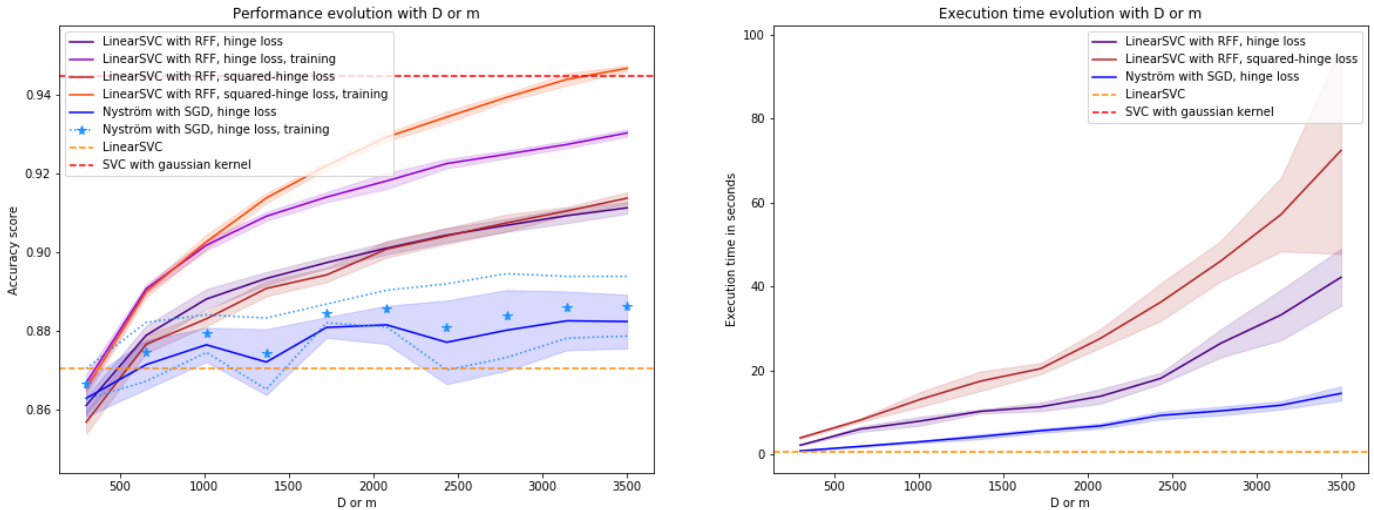


Figure 7: Performance and training time against the regularization strength

Outline

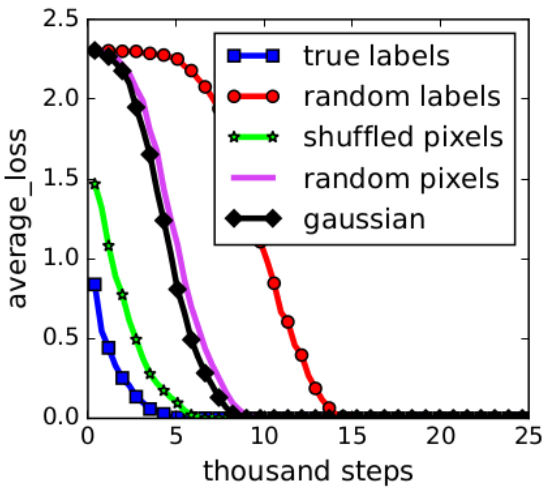
Introduction

Neural networks have proven to generalize quite well, compared to other parametric models. However, they have usually a lot more parameters than data points they are trained on. It is also easy to find a network that generalizes poorly. So then, why do neural networks generalize so well?

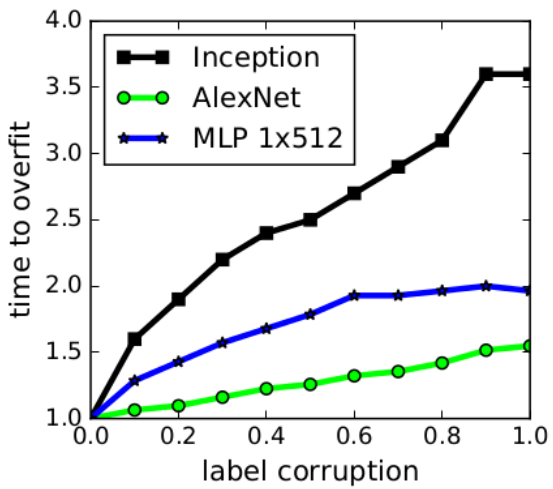
One might think it is due either to properties of the function family associated to these networks, or regularization techniques used during training.

Randomization tests

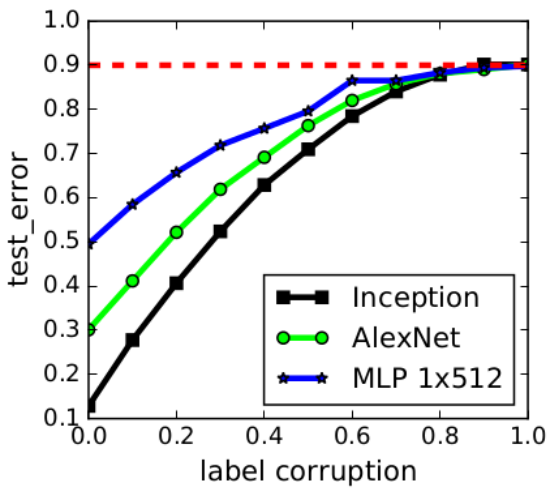
- Corrupt the dataset by inserting noise in the data or the label
- AlexNet, Inception and MLPs trained on CIFAR10 and ImageNet
- Always fit perfectly CIFAR10 during training



(a) learning curves



(b) convergence slowdown



(c) generalization error growth

Randomization tests

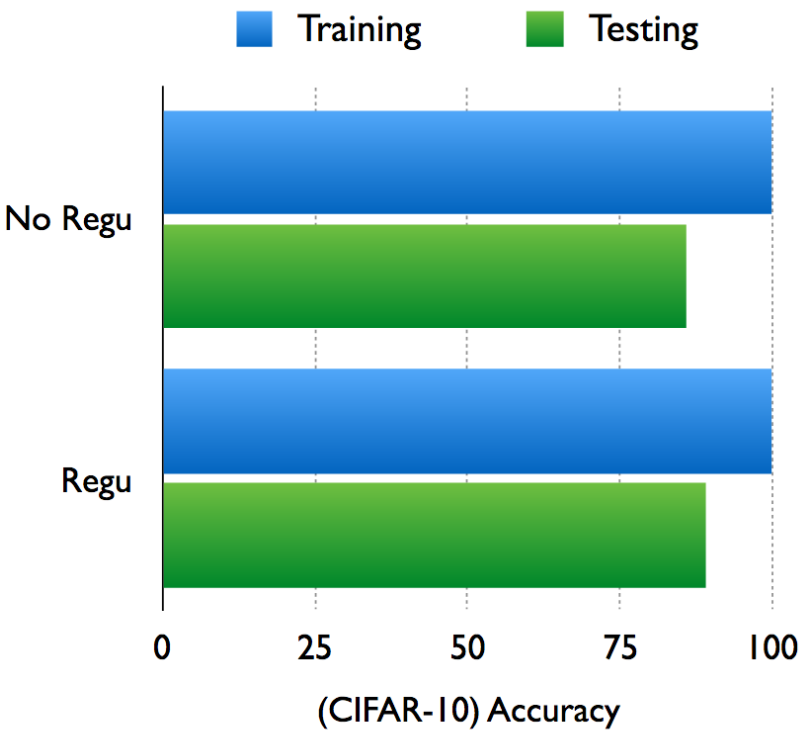
- Same training errors (0%), but Inception which has more parameters than the other, generalizes better

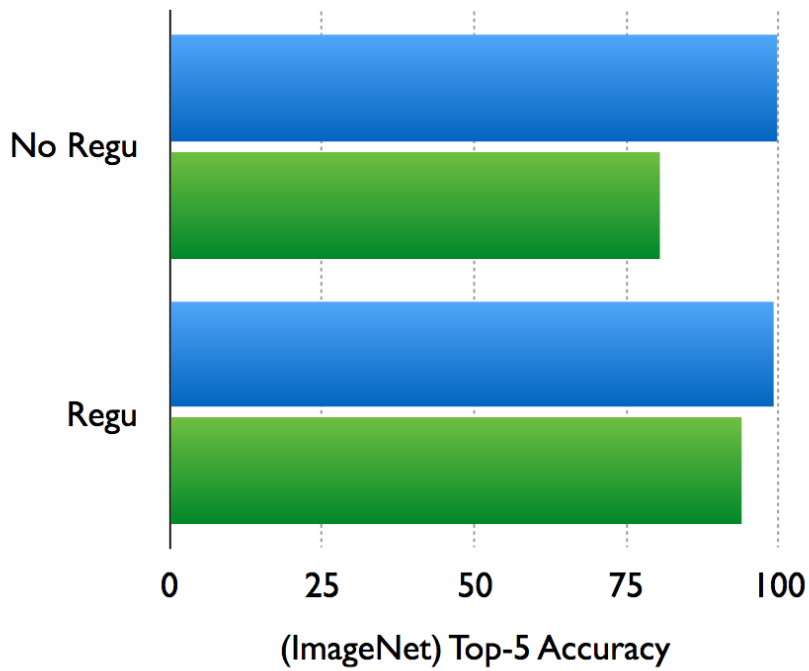
model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.05
		yes	no	100.0	89.31
		no	yes	100.0	86.03
		no	no	100.0	85.75
(fitting random labels)		no	no	100.0	9.78
Inception w/o BatchNorm	1,649,402	no	yes	100.0	83.00
		no	no	100.0	82.00
(fitting random labels)		no	no	100.0	10.12
Alexnet	1,387,786	yes	yes	99.90	81.22
		yes	no	99.82	79.66
		no	yes	100.0	77.36
		no	no	100.0	76.07
(fitting random labels)		no	no	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
(fitting random labels)		no	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
(fitting random labels)		no	no	99.34	10.61

- Able to completely memorize the dataset
- Rademacher, VC-dimension and uniform stability give no useful bound on the generalization error

Regularization

- 3 main regularization techniques in DL
- Dropout, Weight decay and data augmentation
- Applied to both CIFAR10 and ImageNet, with the same networks





Regularization

- Does not necessarily bother training
- Show how other techniques like the SGD or batch normalization can act as an implicit regularizer
- No significantly enough performances change to say generalization in DL is due to regularization techniques

Theoretical result

Finite sample expressivity:

There exists a two-layer neural network with ReLU activations and $2n + d$ weights that can represent any function on a sample of size n in d dimensions

Conclusion

- Neural networks are able to fit any function, provided they have enough parameters
- Their generalization properties are not mainly due to regularization techniques applied
- Optimization is easy
- Today's machine learning tools to understand generalization are not applicable to deep learning

Critics

- Rewarded by the ICLR 2017 committee with the Best Paper Award
- No concrete results explaining the generalization properties of a neural network
- Not always in agreement with the conclusions drawn from the result of the experiments


Ali Rahimi and Benjamin Recht.
Random features for large-scale kernel machines.
In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184. Curran Associates, Inc., 2008.


Salim Rostam.
Transformée de fourier de la gaussienne.
https://perso.ens-rennes.fr/math/people/salim.rostam/files/agreg/D%C3%A9veloppements/Transform%C3%A9e%20de%20Fourier%20de%20la%20gaussienne.pdf, 2014.
[Online; accessed 23-Febrary-2020].


Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco.
Less is more: Nyström computational regularization.

In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1657–1665. Curran Associates, Inc., 2015.

Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco.
Falkon: An optimal large scale kernel method.
In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3888–3898. Curran Associates, Inc., 2017.