Efficient Cross-Validation of Echo State Networks

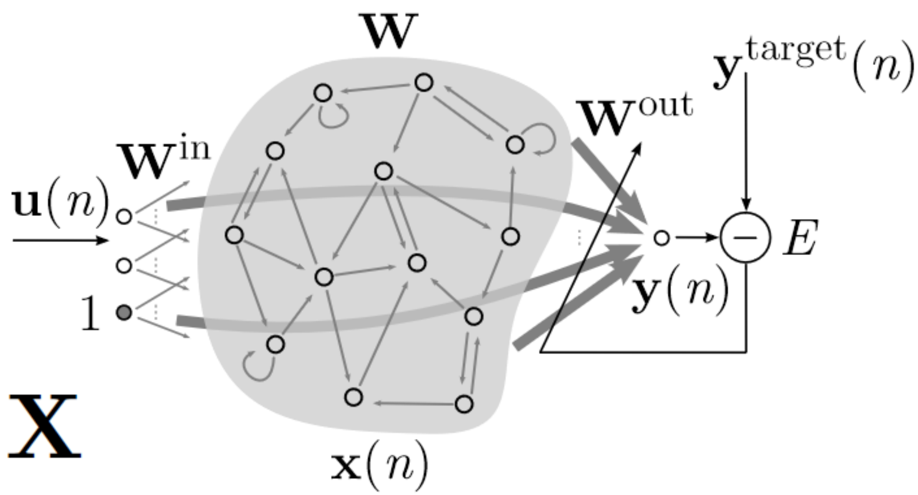
Mantas Lukoševičius and Arnas Uselis
Kaunas University of Technology, Lithuania

19.09.2019 Munich, Germany

Outline

- Cross-validation in time series
  - How, why, and when you should use it
- Efficient implementation in ESNs
  - And other RC methods
- Empirical results

ESN training classics



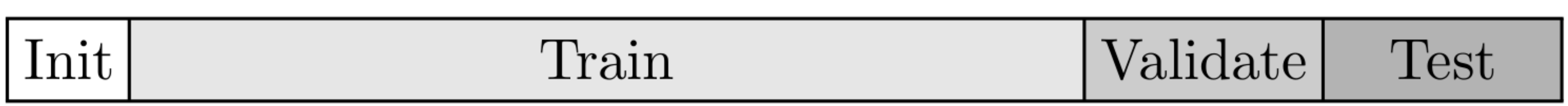
- Generate ESN
- Learn

$$\mathbf{Y}^{\text{target}} = \mathbf{W}^{\text{out}}\mathbf{X}$$

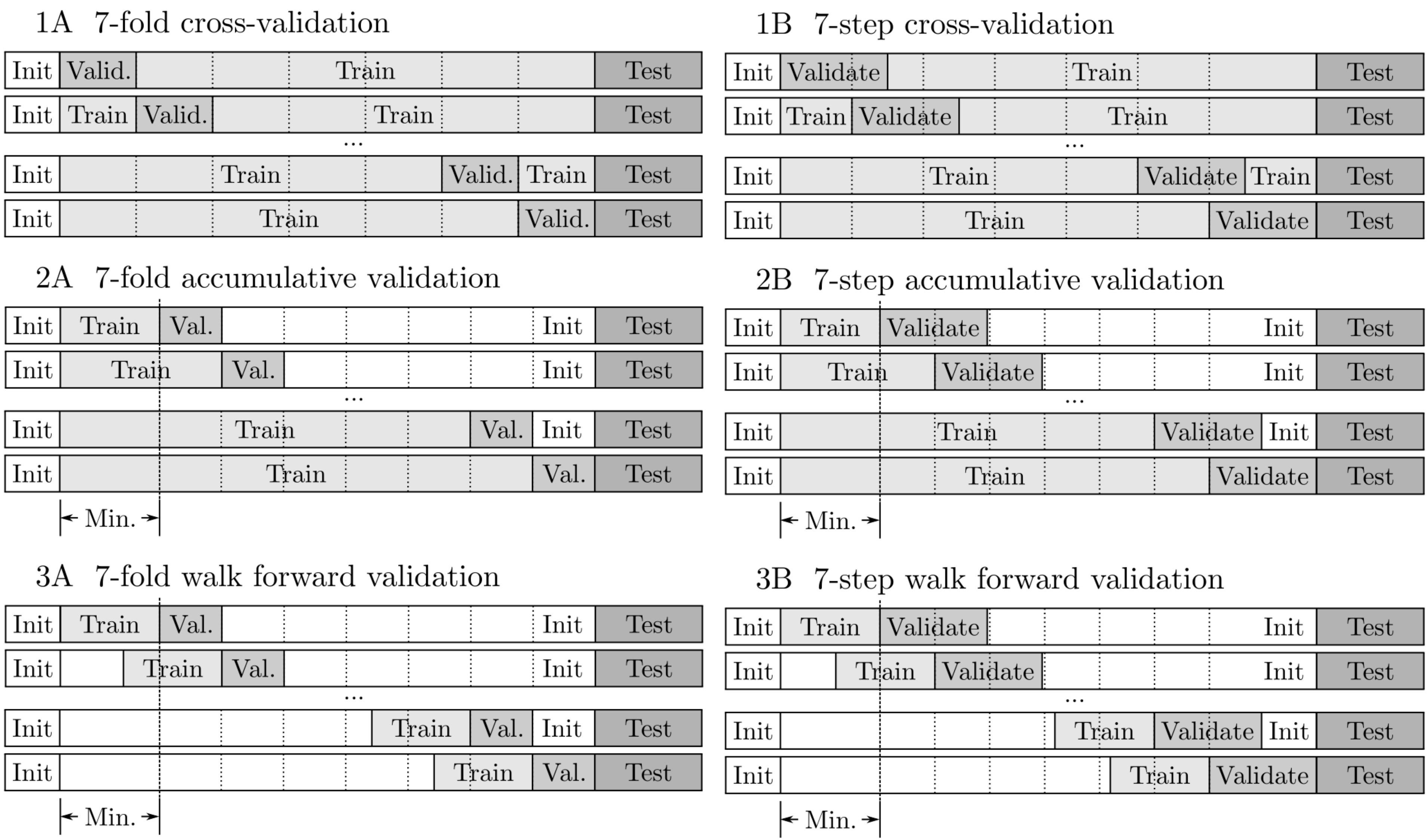
  - One-shot $$\mathbf{W}^{\text{out}} = \mathbf{Y}^{\text{target}}\mathbf{X}^{\mathrm{T}}\left(\mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta\mathbf{I}\right)^{-1}$$
  - Data / time / memory – efficient
- Hyper-parameters are important
- Standard (static) validation:

Cross-validating time series?

- Standard in static tasks
- Less in temporal
  - Breaks continuity – bad for backprop
  - Very expensive
  - Peeking into the future?
- Reservoir computing is well-suited



Cross-validation schemes



Producing the final model

We have the best hyper-params, to get the final model we can:

- Retrain on all the folds
- Average over the splits
- Take the best split

Complexity of ESN training

- foreach $n \in (1,...,T)$:
  - update
    $$\tilde{\mathbf{x}}(n) = \tanh\left(\mathbf{W}^{\mathrm{in}}[1;\mathbf{u}(n)] + \mathbf{W}\,\mathbf{x}(n-1)\right)$$
  - collect $\mathbf{X}\mathbf{X}^{\mathrm{T}}$
- Train

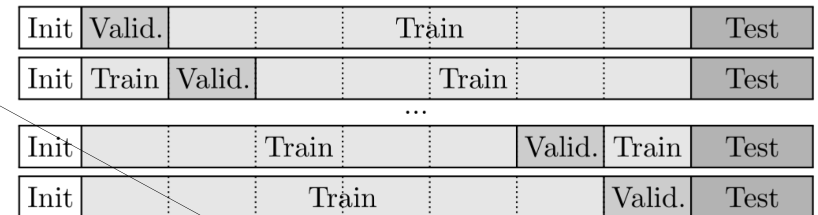
<u>Time:</u> $\mathcal{O}({N_{\mathrm{r}}}^2 T)$ > $\mathcal{O}({N_{\mathrm{r}}}^3)$

$$\mathbf{W}^{\mathrm{out}} = \mathbf{Y}^{\mathrm{target}}\mathbf{X}^{\mathrm{T}}\left(\mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta\mathbf{I}\right)^{-1}$$

<u>Space:</u> $\mathcal{O}({N_{\mathrm{r}}}^2)$

Efficient k-fold x-val. algorithm

Naive time: $\mathcal{O}(k{N_\mathrm{r}}^2T)$



- Collect global $\mathbf{X}\mathbf{X}^\mathrm{T}$
- foreach k fold:
  - Collect $\mathbf{X}\mathbf{X}^\mathrm{T}$ for val. fold, subtract from global $\mathbf{X}\mathbf{X}^\mathrm{T}$
  - Train $\left(\mathbf{X}\mathbf{X}^\mathrm{T} + \beta\mathbf{I}\right)^{-1}$
  - Validate on val. fold

$\mathcal{O}({N_\mathrm{r}}^2T)$

$\mathcal{O}(k{N_\mathrm{r}}^3)$

- Time complexity
  - Time: $\mathcal{O}({N_\mathrm{r}}^2T + k{N_\mathrm{r}}^3)$
  - Dominated by $\mathcal{O}({N_\mathrm{r}}^2T)$ when $k < T/N_\mathrm{r}$

Space: $\mathcal{O}({N_\mathrm{r}}^2)$

Datasets used

- Generative prediction

Dataset	Samples T	Valid/test samples	Folds k	Min. ratio
Labour	360	10	34	50%
Gasoline	1355	67	18	50%
Electricity	4033	200	18	50%
Sunspots	3177	200	10	50%

- Time series classification
  - Japanese wovels

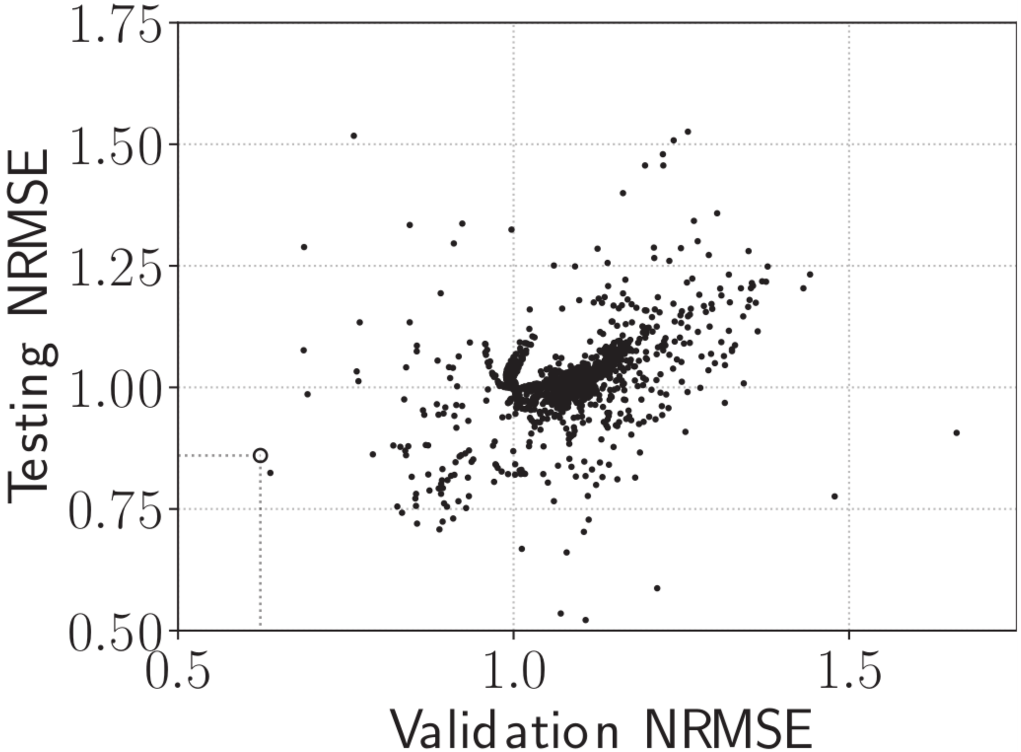
Results: generative prediction

Method		**Labour**		**Gasoline**		**Electricity**		**Sunspots**	
Validation	**Final**	**Valid**	**Test**	**Valid**	**Test**	**Valid**	**Test**	**Valid**	**Test**
SV	As is	1.034	1.927	0.891	0.881	0.623	0.860	0.749	0.784
	Retrained		1.957		1.132		0.835		0.755
k-fold CV	Averaged	2.009	1.835	1.000	0.914	0.834	0.990	1.060	0.924
	Retrained		1.833		0.913		1.006		0.970
	Best		1.838		0.901		0.995		1.008
k-step AV	Averaged	1.927	4.469	1.040	0.867	0.812	0.829	0.703	0.835
	Retrained		1.171		0.962		1.006		0.742
	Best		4.546		**0.829**		0.820		0.855
k-step FV	Averaged	2.188	3.413	1.065	0.925	0.783	**0.733**	0.726	0.640
	Retrained		**0.681**		0.949		1.006		**0.612**
	Best		2.799		0.894		0.769		0.649

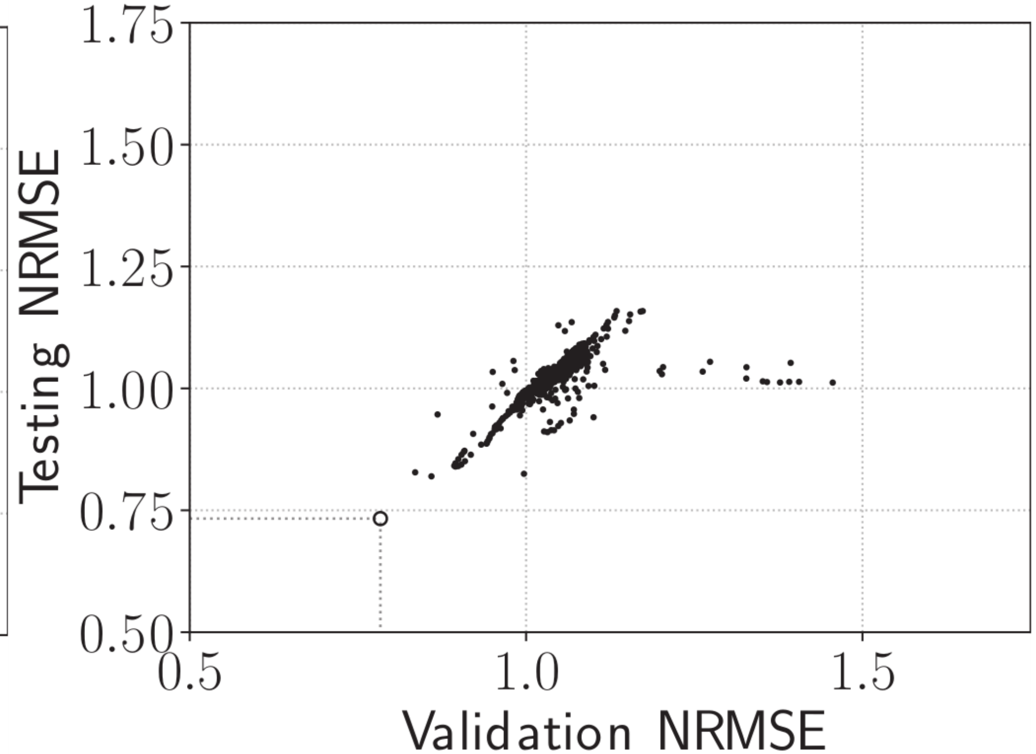
Results: Japanese wovels

Method	**Final**	**Validation error**	**Test error**	**Missclasifications**
SV	As is	0.504 ± 0.017	0.491 ± 0.005	5.0 ± 1.5
	Retrained		0.486 ± 0.003	4.8 ± 1.6
CV	Averaged	0.493 ± 0.004	0.472 ± 0.008	4.2 ± 1.8
	Retrained		0.468 ± 0.006	4.4 ± 2.1
CV IReg	Averaged	0.489 ± 0.004	$\mathbf{0.468} \pm 0.009$	4.4 ± 1.2
	Retrained		0.470 ± 0.008	$\mathbf{3.8} \pm 1.8$

How it works



(a) SV Retrained



(b) k-step FV Averaged

When Cross-validating time series makes sense

- When data are scarse, use it efficiently
- Averaging models is a form of aditional regularization
- It’s about (non-)stationarity of the process
  – stationary: V scheme doesn’t matter much
  ~ „wanders“ around – CV
  / „drifts“ in one direction – AV, FV, SV
  (! strongly non-stationary – hard to learn at all)

Conclusions

- The speedup applies to other Reservoir Computers too
  - No need to run the reservoir k times
- Cross-validation in RC
  - Is easy, efficient
  - Predicts testing performance better
  - Exploits RC strengths
- Extensions?
  - Weighting splits
  - Ensembling
  - ...

Questions?

Contacts:
https://mantas.info/