San Jose, USA California

GTC 2013 - GPU Technology Conference

20. March 2013

Learn-O-Matic

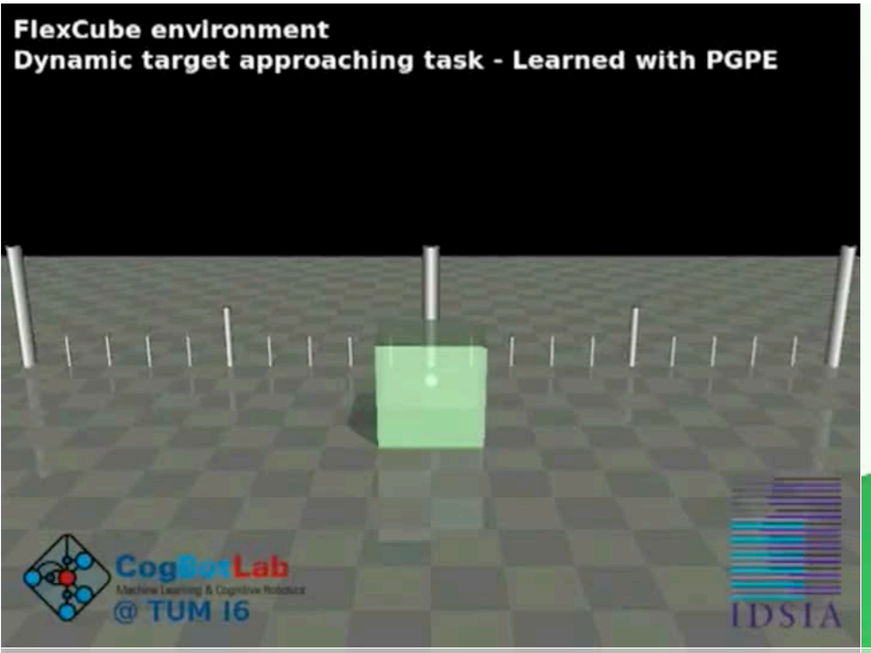
A Fully Automated GPU Machine Learning Suite

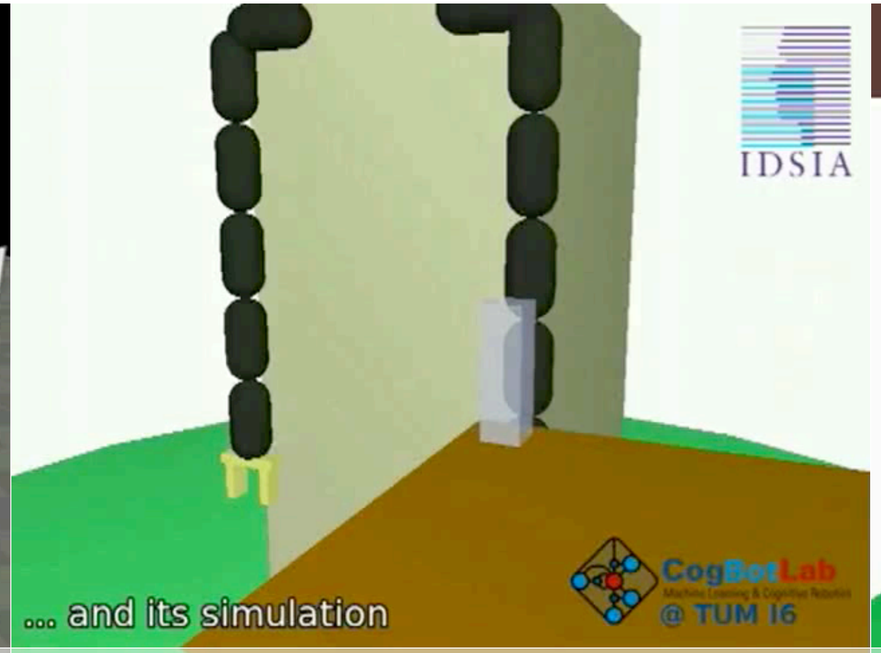
Frank Sehnke, Martin Felder, Anton Kaifel

Zentrum für Sonnenenergie- und Wasserstoff-Forschung
Baden-Württemberg (ZSW)

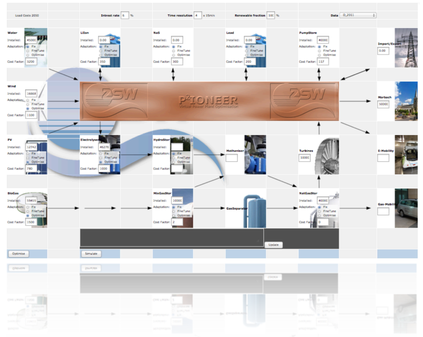
Contents

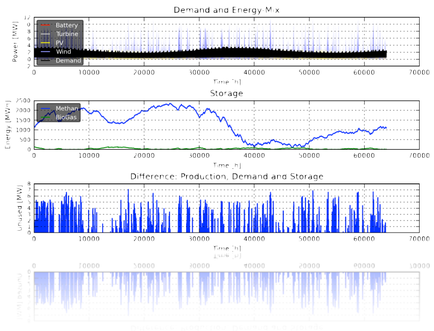
From Robotics ...

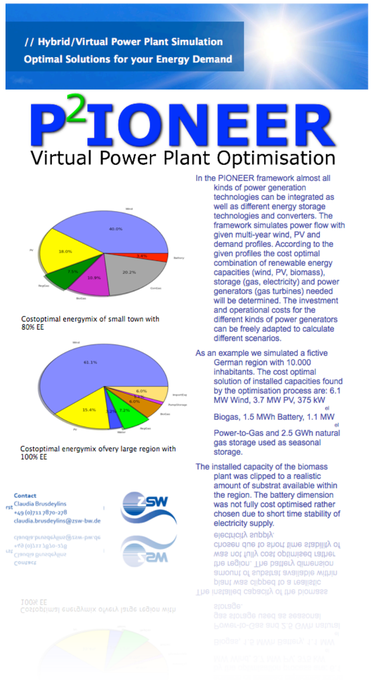


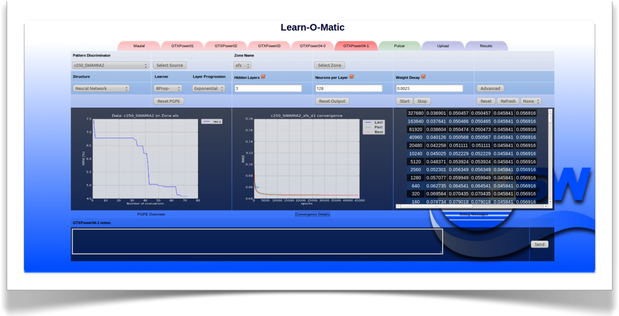


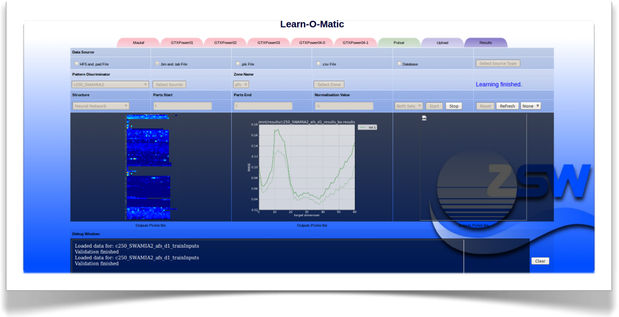
... to Renewable Energies - with a little help from my GPU

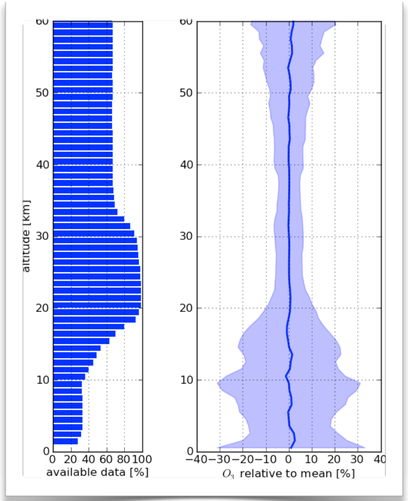


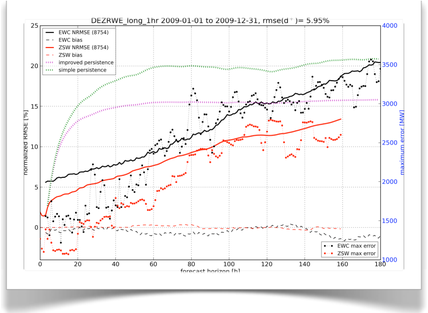


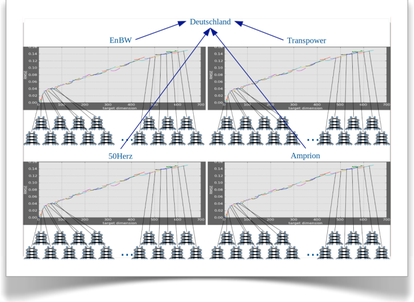










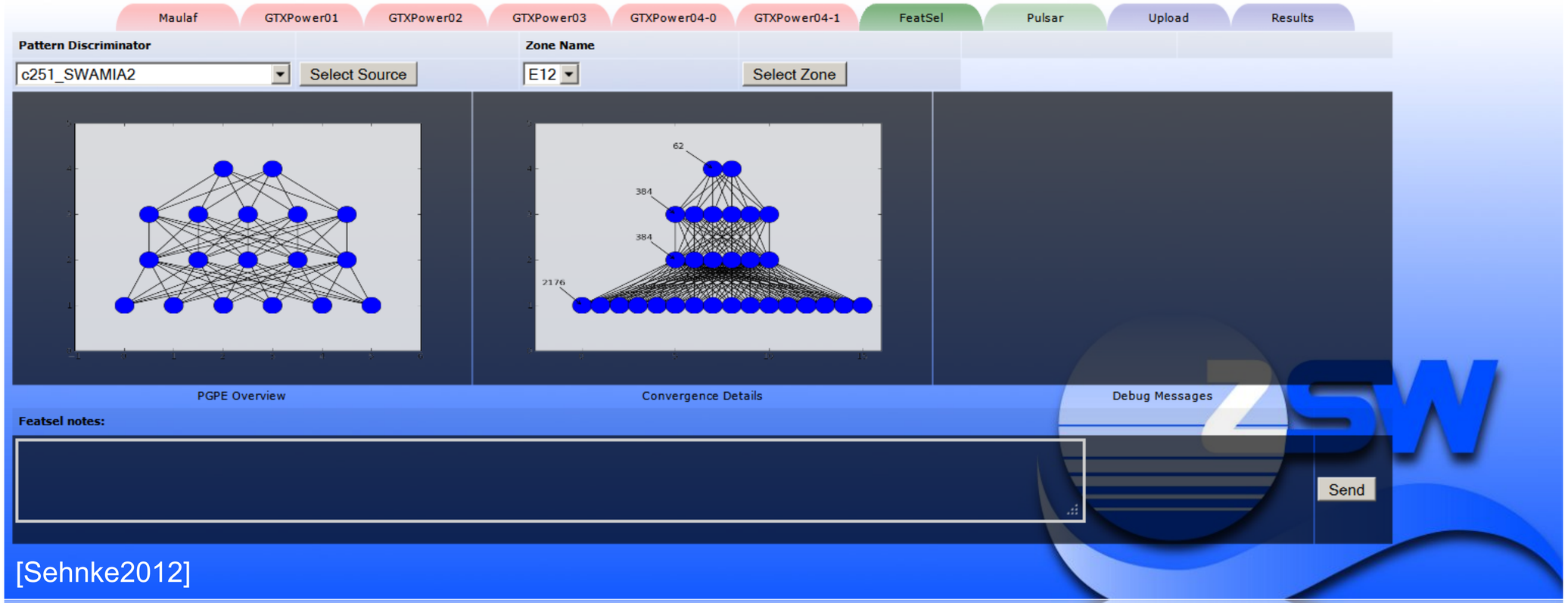


Learn-O-Matic

Machine Learning and Optimization Tool including Deep Learning and Automatic Feature Selection

- ➔ Multi-tier **GPU** based machine learning system with user friendly web frontend.
- ➔ Increased computation power (via GPUs) allows training of networks several times.
- ➔ Meta-learning possible - search in meta parameter space via RL (reward = performance on validation set).

Learn-O-Matic



[Sehnke2012]

Learn-O-Matic: Implemented Features

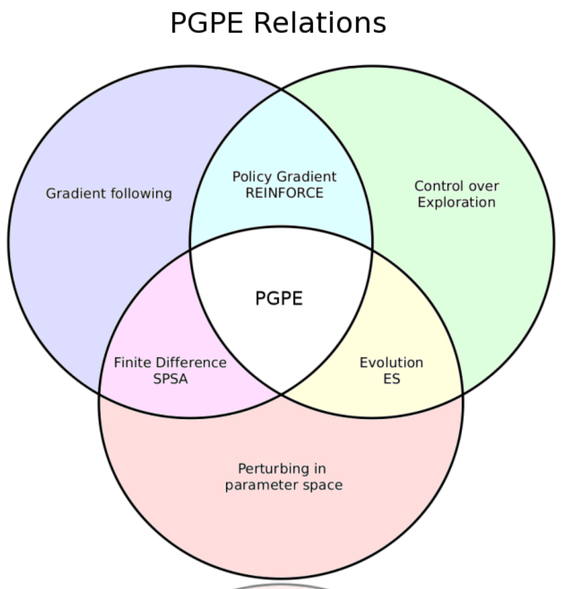
➔ Deep neural networks (NN) (RProp, SGD, RMSProp, ...)

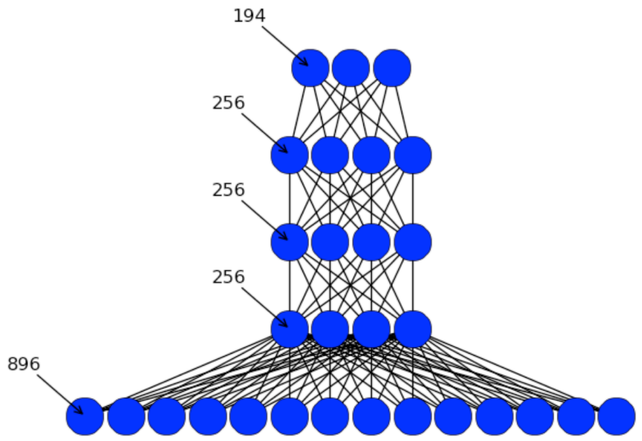
➔ Restricted Boltzmann Machines [Hinton2006]

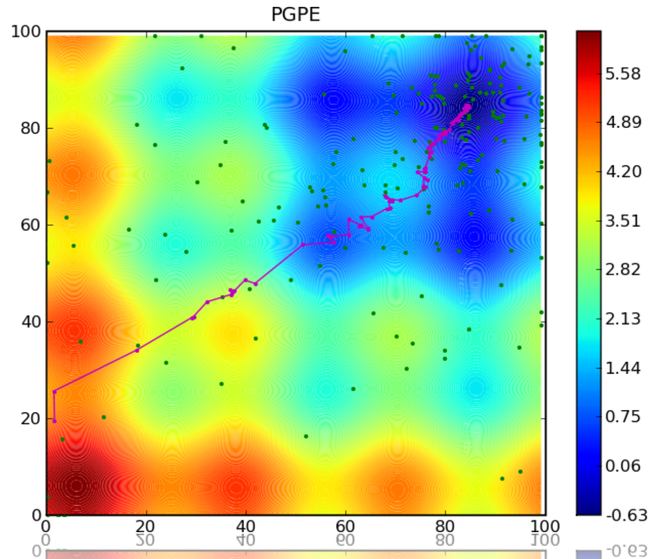
➔ Support vector machines/regression (soon)

➔ Gaussian Processes

➔ Policy Gradient with Parameter-based Exploration (PGPE): reinforcement learning scheme for all kind of optimization tasks [Sehnke2010]







Used Hard- and Software

- We use 3xGTX580, 1xGTX590, 1xTesla C2075, 1xGTX680(4GB, soon 5x)
  - We are able to choose consumer cards over GPGPUs.
  - NNs are very robust vs. infrequent memory error (effects only one of millions of weights)
  - Training of NNs is an iterative process with several thousand epochs. If a GPU has an issue one continues at the latest saved weight matrix.
- We code in Python.
  - Cudamat [Mnih2009] builds directly on Cuda as Python interface.
  - GnumPy [Tielman2010] restores the NumPy Syntax.
  - GPU code for propagating data through a NN condenses down to 3 lines:

```
out = inp.copy()
for i in range(numLayers-1):
    out = gp.dot(weights[i],out).logistic()
```

  - Saves a lot of developer time and costs negligible performance.

NNORSY – Neural Network Ozone Retrieval SYstem

Synergistic ozone profile retrieval using METOP data :

Input data:

- **IASI**: IR spectra (~ **8000** channels)
- **GOME-2**: UV/VIS spectra (~ **4000** channels)
- **AVHRR**: cloud fraction within GOME-2/IASI FOV
- **ECMWF**: temperature profile data

Output data:

- Ozone profile with 61 layers (0.5 km to 60.5 km)

Ozone profile training data: ground and satellite based

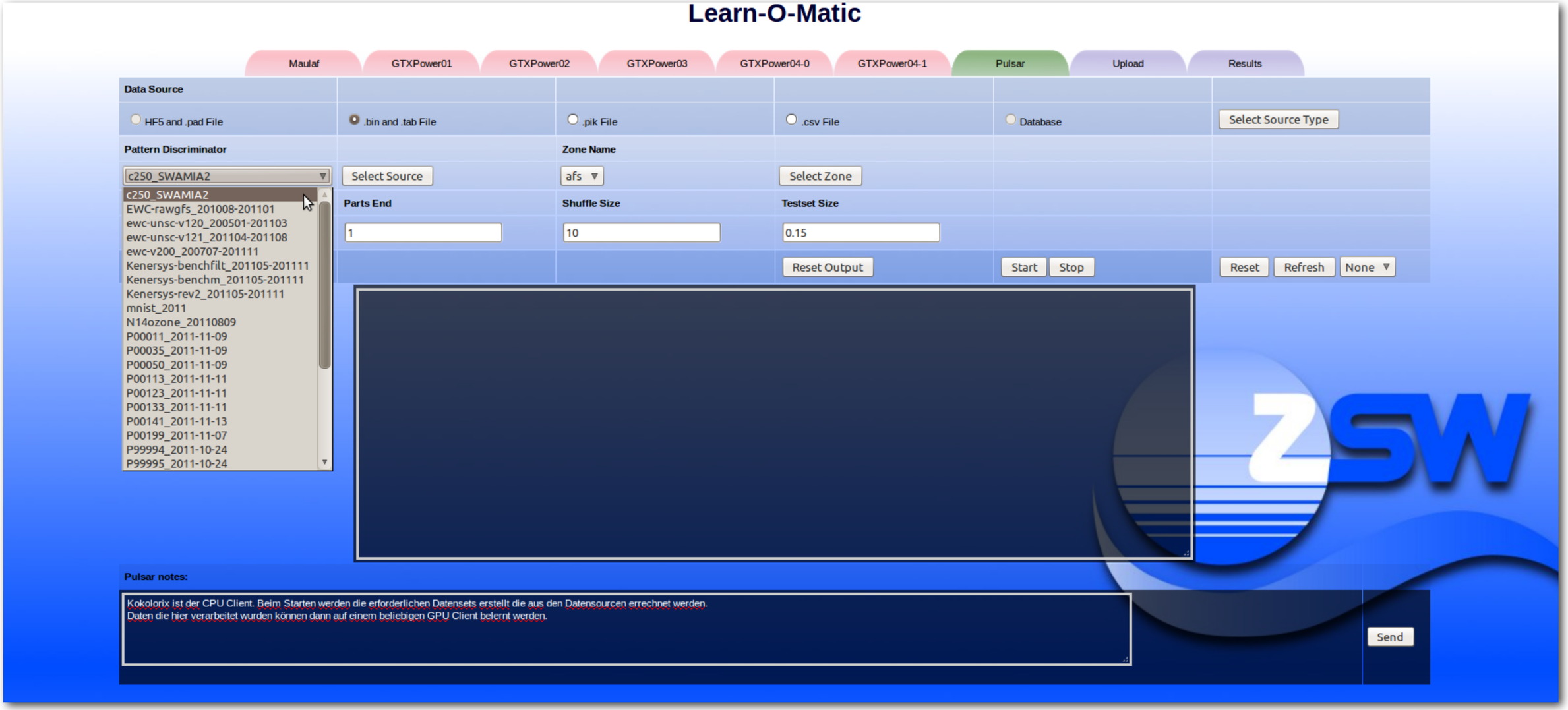
- Ozone sonde measurements (WOUDC and SHADOZ)
- AURA-MLS
- ACE-FTS

Step 1: Data Preparation

- The data are presented as a simple binary dump of an array with dimensions [#patterns, #features]
- The „meaning“ of the data are represented in a so called „feature list file“.
- User can define which parts of the data should be used as inputs for the network.

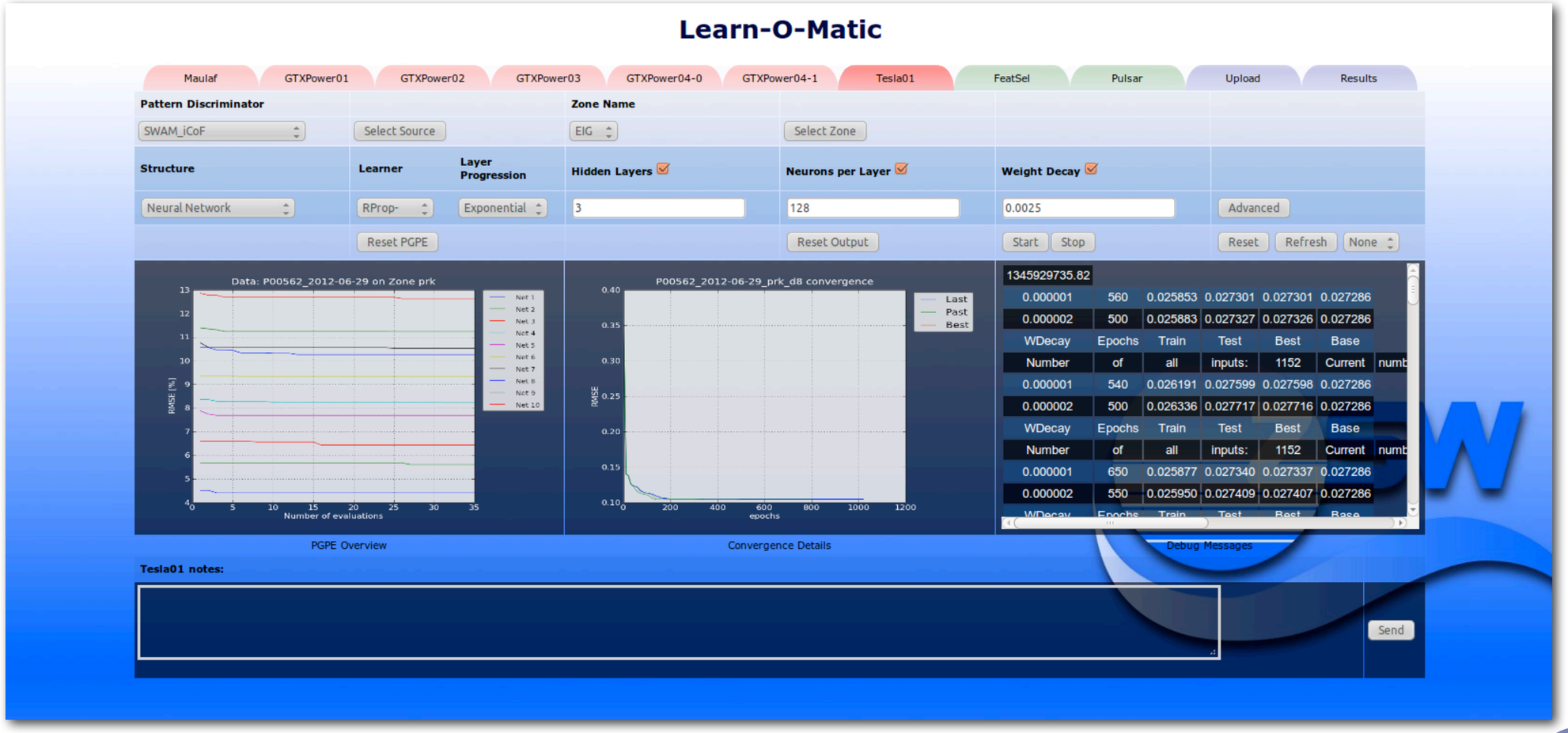
```
 1 [FeatureList] H
 2 Type  Length Handling        Select                    NormType  NormRange   Doc                              Units
 3 Feat_R1A_1  128   "spec"     ""                           1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
 4 Feat_R1A_2  128   "spec"     ""                           1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
 5 Feat_R1A_3  128   "spec"     ""                           1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
 6 Feat_R1A_4  128   "spec"     ""                           1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
 7 Feat_R1A_5  128   "spec"     ""                           1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
 8 Feat_R1A_6  128   "spec"     "50,64-84@4"                 1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
 9 Feat_R1A_7  109   "spec"     "16-32@4,44-68@4,92-109@4"   1      "0.0,0.0"   "GOME-2 band 1a reflectivities"   "[1]"
10 Feat_R1B    147   "spec"     "1-80&4"                     1      "0.0,0.0"   "GOME-2 band 1b reflectivities"   "[1]"
11 Feat_R2B_1  128   "spec"     ""                           1      "0.0,0.0"   "GOME-2 band 2b reflectivities"   "[1]"
```

Step 2: GUI - Data Preprocessing



Step 3: Feature Selection

An automated target driven feature selection can be executed for input dimension reduction.
A L1 weight decay [Ng1998] with probabilistic neuron pruning on the input layer is used to reduce iteratively the number of irrelevant inputs.



Step 4: Deep Supervised Meta Learning

- After preprocessing the data is ready for meta learning on all GPU-Backends. All Meta parameters (hidden layers, neurons per layer, weight decay, ...) are optimised via Policy Gradients with Parameter-based Exploration (PGPE), a state of the art Reinforcement Learning algorithm. [Sehnke2010]
- Statistics of the optimisation as well as of the NN training are presented in real time on the webpage.

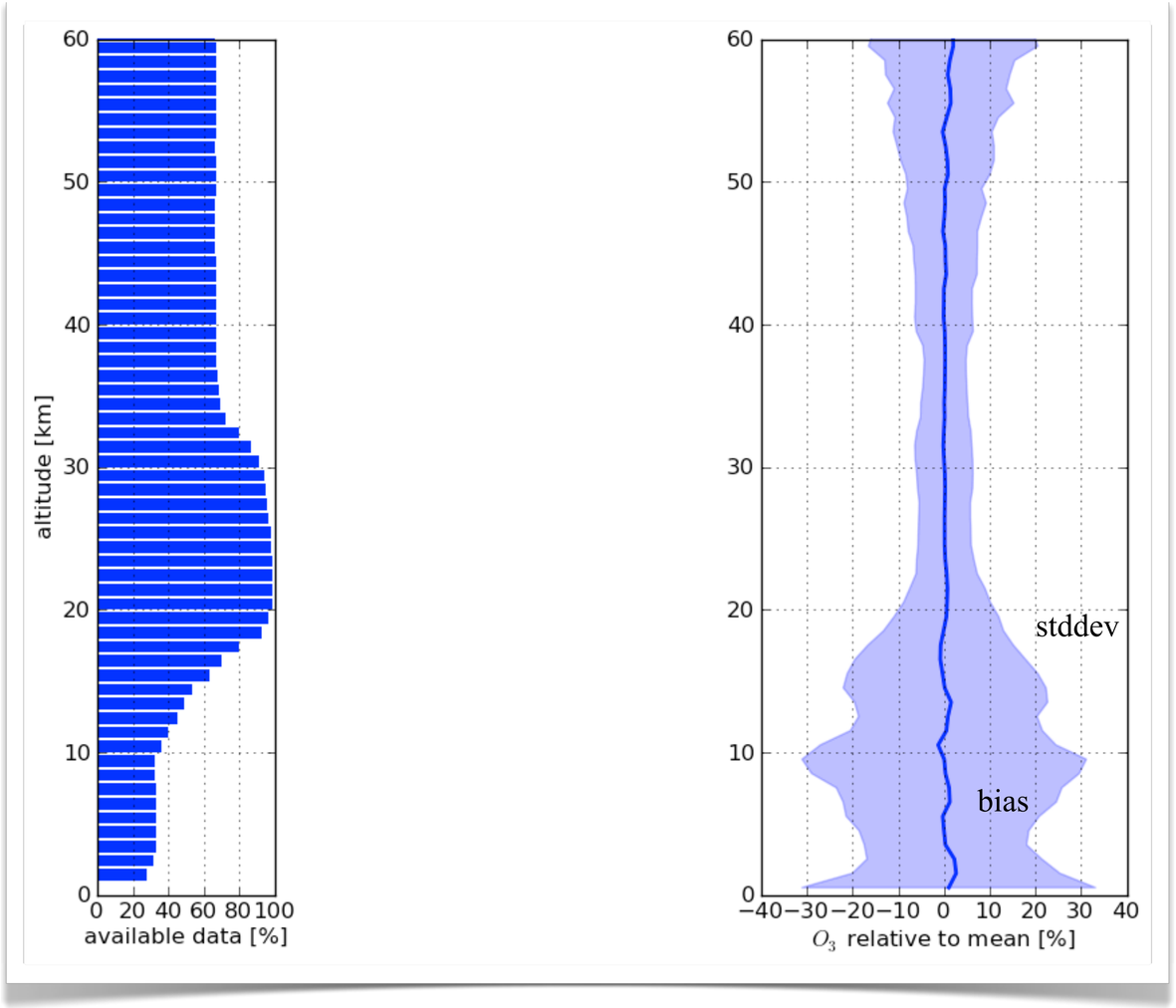
Step 5: Model Validation

After learning has finished, the resulting model can be evaluated. RMSE statistics and feature sensitivity can be viewed and analysed.

The whole learning process is easy to execute and no knowledge about machine learning is needed.

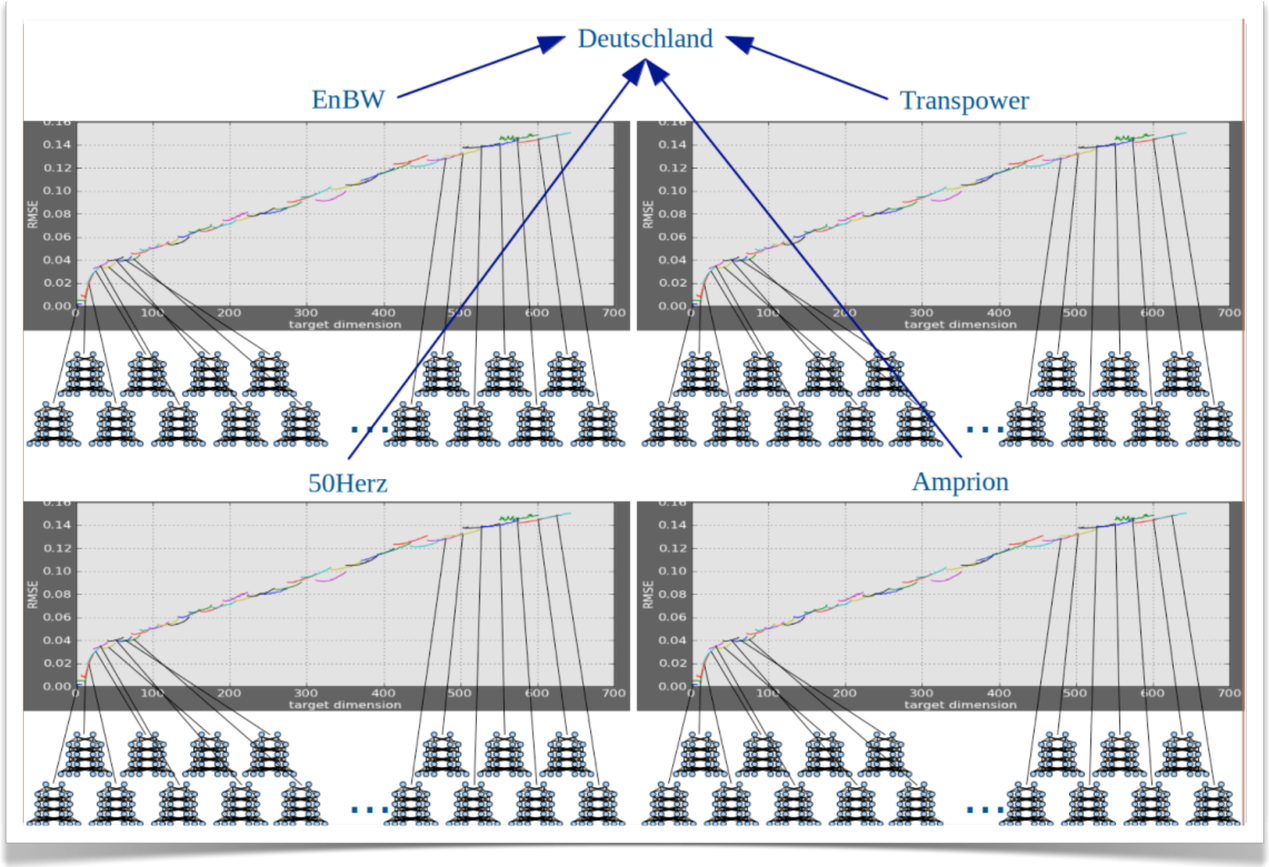
Results - NNORSY

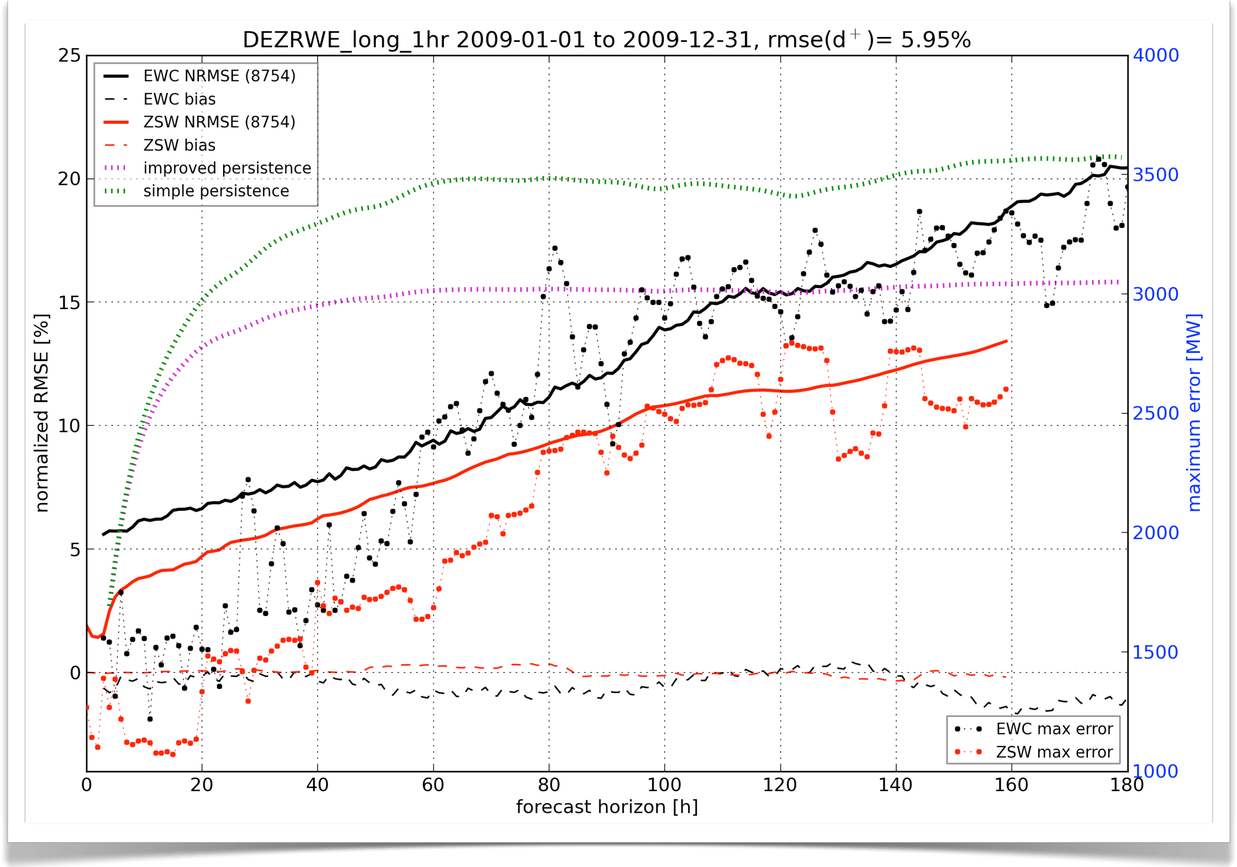
Test set collocation statistics (ca. 27000 profiles)



Wind Power Prediction with Learn-O-Matic

- Different application domain: Energy yield forecasting





Summary

- With Learn-O-Matic arbitrary data for supervised training can be handled.
- There is no need for expert knowledge in machine learning.
- The resulting systems investigated so far performs competitively to expert build systems.
- All tools used can be executed or are executed automatically from the Learn-O-Matic Web Frontend which is accessible over internet.
- For neural network training even consumer cards work very well, due to the robustness of NN to scarce memory errors (single weight values are wrong).

Thank you for your attention!

[Sehnke2012] Sehnke et al. 2012, Learn-O-Matic - Fully automated Machine Learning, ZSW internal Tech. Note

[Igel2000] Christian Igel and Michael Hüsken. Improving the rprop learning algorithm, 2000.

[Riedmiller1993] Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS, pages 586–591, 1993.

[Sehnke2010] F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber. Parameter-exploring policy gradients. Neural Networks, 23(4):551–559, 2010.

[Mnih2009] V. Mnih. Cudamat: a CUDA-based matrix class for python. Department of Computer Science, University of Toronto, Tech. Rep. UTML TR, 2009

[Tielman2010] T. Tieleman. Gnumpy: an easy way to use GPU boards in Python. Technical Report UTML TR 2010-002, University of Toronto, Department of Computer Science, 2010.

[Ng1998] A.Y. Ng. On feature selection: learning with exponentially many irrevelant features as training examples. Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2010