

The Shogun Machine Learning Toolbox

将軍
sho gun

heiko.strathmann@gmail.com

Europython 2014

July 24, 2014

Outline

Overview

Machine Learning Features

Technical Features

Community

A bit about me

- ▶ Heiko Strathmann - <http://herrstrathmann.de/>
- ▶ 2004-2008 - Jazz guitar
- ▶ 2008-2013 - BSc in CS, MSc in ML

- ▶ Phd in Neuroscience & Machine Learning, UCL
- ▶ Kernels, Bayesian & MCMC, Bioinformatics, [Open-Source](#)
- ▶ Involved in Shogun since 2010

A bit about Machine Learning

- ▶ Science of **patterns** in information
- ▶ For example for **recognising**
 - ▶ Frauds in aeroplane wings
 - ▶ Skin cancer
 - ▶ Abuse of bank accounts
- ▶ ...or **predicting**
 - ▶ Treatment resistance of HIV
 - ▶ Brain activity
 - ▶ Products on Amazon/Spotify/Netflix
- ▶ Not limited to, but including
 - ▶ Statistics
 - ▶ Big Data & <place-buzzword>
 - ▶ Robot world domination



A bit about Shogun



The banner features a central graphic of a white box with a colorful abstract design and the Japanese characters '将' (Shō) and '軍' (gun) in black. A yellow 'new' tag is on the top left of the box. Below the box is a 'DOWNLOAD NOW' button. To the right, the text 'SHOGUN 3.2.0' is displayed in large yellow letters, with the tagline 'In \$DEITY we trust all others bring data.' below it. A yellow banner in the top right corner says 'Fork me on GitHub'. At the bottom left, it says 'What's New: SHOGUN Release version 3.2.0 (libshogun 16.0, data 0.8, parameter 1)'. At the bottom right, there is a blue penguin icon and the text 'Follow us on Twitter'.

- ▶ Open-Source tools for ML problems
- ▶ Made public in 2004
- ▶ Currently 8 core-developers + 20 regular contributors
- ▶ Originally academic background
- ▶ In Google Summer of Code since 2010 (29 projects!)
- ▶ **Upcoming workshop:** July 27, 28, c-base

Ohloh - Summary

In a Nutshell, SHOGUN...

- ... has had 25,187 commits made by 126 contributors representing 588,445 lines of code
- ... is mostly written in C++ with a very low number of source code comments
- ... has a well established, mature codebase maintained by a very large development team with stable Y-O-Y commits
- ... took an estimated 162 years of effort (COCOMO model) starting with its first commit in June, 2006 ending with its most recent commit about 3 hours ago

Ohloh - Code

Languages



C++

50%

C

35%

Python

7%

12 Other

8%

Lines of Code

1,000k

500k

0k

2008

2010

2012

2014

Code

Comments

Blanks

Ohloh - Commits



Outline

Overview

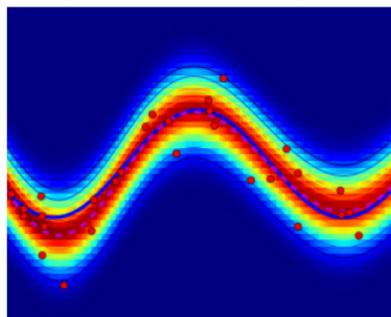
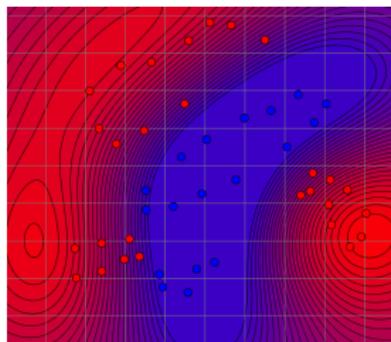
Machine Learning Features

Technical Features

Community

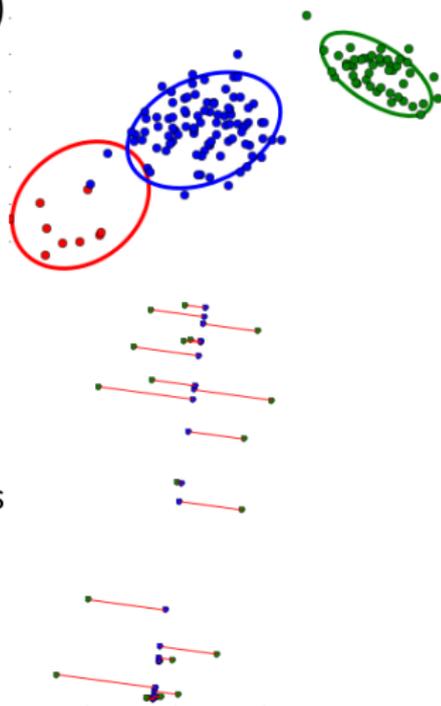
Supervised Learning

- ▶ Given: $\{(x_i, y_i)\}_{i=1}^n$, want: $y^*|x^*$
- ▶ Classification: y discrete
 - ▶ Support Vector Machine
 - ▶ Gaussian Processes
 - ▶ Logistic Regression
 - ▶ Decision Trees
 - ▶ Nearest Neighbours
 - ▶ Naive Bayes
- ▶ Regression: y continuous
 - ▶ Gaussian Processes
 - ▶ Support Vector Regression
 - ▶ (Kernel) Ridge Regression
 - ▶ (Group) LASSO



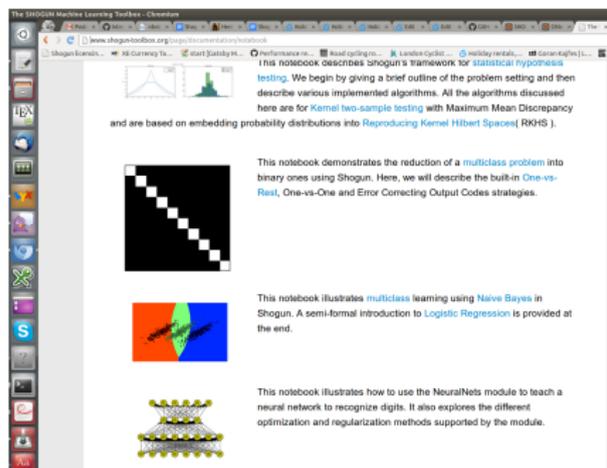
Unsupervised Learning

- ▶ Given: $\{x_i\}_{i=1}^n$, want notion of $p(x)$
- ▶ Clustering:
 - ▶ K-Means
 - ▶ (Gaussian) Mixture Models
 - ▶ Hierarchical clustering
- ▶ Latent Models
 - ▶ (K) PCA
 - ▶ Latent Discriminant Analysis
 - ▶ Independent Component Analysis
- ▶ Dimension reduction
 - ▶ (K) Locally Linear Embeddings
 - ▶ Many more...



And many more

- ▶ Multiple Kernel Learning
- ▶ Structured Output
- ▶ Metric Learning
- ▶ Large-Scale log-determinants
- ▶ Variational Inference
- ▶ Deep Learning (whooo!)
- ▶ ...



The screenshot shows a web browser window displaying the Shogun Machine Learning Toolbox documentation. The page title is "The SHOGUN Machine Learning Toolbox - Overview". The browser's address bar shows the URL "http://www.shogun-toolbox.org". The page content includes several sections with text and images:

- A section with a graph icon and text: "This notebook describes shogun's framework for **statistical hypothesis testing**. We begin by giving a brief outline of the problem setting and then describe various implemented algorithms. All the algorithms discussed here are for **Kernel two-sample testing** with Maximum Mean Discrepancy and are based on embedding probability distributions into **Reproducing Kernel Hilbert Spaces** (RKHS)." Below this text is a black square with a white diagonal line of pixels.
- A section with a colorful abstract image icon and text: "This notebook demonstrates the reduction of a **multiclass problem** into binary ones using Shogun. Here, we will describe the built-in **One-vs-Real**, **One-vs-One** and **Error Correcting Output Codes** strategies." Below this text is a square divided into three colored regions (red, green, blue) with a black diagonal line.
- A section with a neural network diagram icon and text: "This notebook illustrates **multiclass learning** using **Naive Bayes** in Shogun. A semi-formal introduction to **Logistic Regression** is provided at the end." Below this text is a diagram of a neural network with multiple layers of nodes.

<http://www.shogun-toolbox.org/page/documentation/notebook>

Some Large-Scale Applications

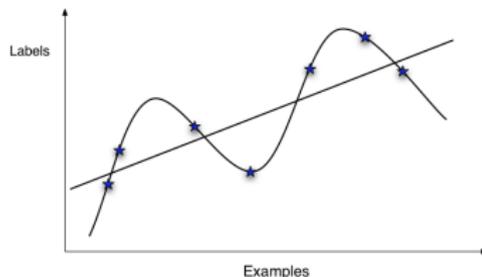
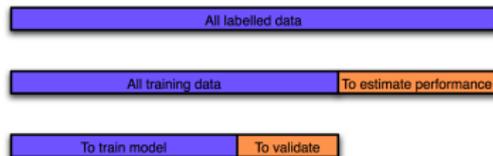
- ▶ Splice Site prediction: 50m examples of 200m dimensions
- ▶ Face recognition: 20k examples of 750k dimensions

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG  
AAGATTAACAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT  
CACTCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC  
TTAATTTCACTTCCACATACTCCAGATCATCAATCTCCAAAACCAACAC  
TTGTTTTAATATTTCAATTTTTACAGTAAGTTGCCAATTCATGTTCCAC  
TACCTAATTATGAAATTAATTCAGTGTGCTGATGAAACGGAGAAGTC
```



ML in Practice

- ▶ Modular data representation
 - ▶ Dense, Sparse, Strings, Streams, ...
 - ▶ Multiple types: 8-128 bit word size
 - ▶ Preprocessing tools
- ▶ Evaluation
 - ▶ Cross-Validation
 - ▶ Accuracy, ROC, MSE, ...
- ▶ Model Selection
 - ▶ Grid-Search
 - ▶ Gradient based
- ▶ Various native file formats, generic multiclass, etc



Outline

Overview

Machine Learning Features

Technical Features

Community

Geeky details

- ▶ Written in (proper) C/C++
- ▶ Modular, fast, memory efficient
- ▶ Unified interface for Machine Learning
- ▶ Linear algebra & co: Eigen3, Lapack, Arpack, pthreads, OpenMP, recently GPUs

Class list:

<http://www.shogun-toolbox.org/doc/en/latest/namespeshogun.html>

Modular language interfaces

- ▶ SWIG - <http://www.swig.org/>
- ▶ We write:
 - ▶ C/C++ classes
 - ▶ Typemaps (i.e. 2D C++ matrix \Leftrightarrow 2D numpy array)
 - ▶ List of classes of expose
- ▶ SWIG generates:
 - ▶ Wrapper classes
 - ▶ Interface files
- ▶ Automagically happens at compile time
- ▶ **Identical** interface for all modular languages:
 - ▶ C++, Python, Octave, Java, R, Ruby, Lua, C#
- ▶ We are in Debian/Ubuntu, but also Mac, Win, Unix

C/C++

```
#include <shogun/base/init.h>
#include <shogun/kernel/GaussianKernel.h>
#include <shogun/labels/BinaryLabels.h>
#include <shogun/features/DenseFeatures.h>
#include <shogun/classifier/svm/LibSVM.h>

using namespace shogun;

int main()
{
    init_shogun_with_defaults();
    ...
    exit_shogun();
    return 0;
}
```

C/C++

```
DenseFeatures<float64_t>* train=new  
    DenseFeatures<float64_t>(...);  
DenseFeatures<float64_t>* =new DenseFeatures<  
    float64_t>(...);  
BinaryLabels* labels=new BinaryLabels(...);  
  
GaussianKernel* kernel=new GaussianKernel(  
    cache_size, width);  
svm=new LibSVM(C, kernel, labels);  
svm->train(train);  
  
CBinaryLabels* predictions=CLabelsFactory::  
    to_binary(svm->apply(test));  
predictions->display_vector();  
  
SG_UNREF(svm);  
SG_UNREF(predictions);
```

Python

```
from modshogun import *

train=RealFeatures(numpy_2d_array_train)
test=RealFeatures(numpy_2d_array_test)
labels=BinaryLabels(numpy_1d_array_label)

kernel=GaussianKernel(cache_size, width)
svm=LibSVM(C, kernel, labels)
svm.train(train)

predictions=svm.apply(test)

# print first prediction
print predictions.get_labels()[0]
```

Octave

```
modshogun

train=RealFeatures(octave_matrix_train);
test=RealFeatures(octave_matrix_train);
labels=BinaryLabels(octave_labels_train);

kernel=GaussianKernel(cache_size, width);
svm=LibSVM(C, kernel, labels);
svm.train(train);

predictions=svm.apply(test);

% print first prediction
disp(predictions.get_labels()[1])
```

Java

```
import org.shogun.*;
import org.jblas.*;
import static org.shogun.LabelsFactory.to_binary;

public class classifier_libsvm_modular {
    static {
        System.loadLibrary("modshogun");
    }

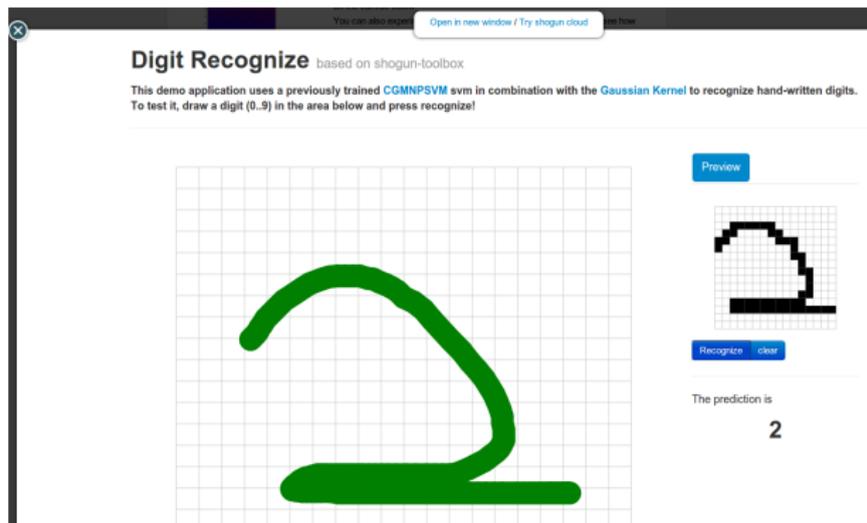
    public static void main(String argv[]) {
        modshogun.init_shogun_with_defaults();

        RealFeatures train=new RealFeatures(new CSVFile(train_file));
        RealFeatures test=new RealFeatures(new CSVFile(test_file));
        BinaryLabels labels=new BinaryLabels(new CSVFile(label_fname));
        GaussianKernel=new GaussianKernel(cache_size, width);
        LibS svm=new LibSVM(C, kernel, labels);
        svm.train(train);

        // print predictions
        DoubleMatrix predictions=to_binary(svm.apply(test)).get_labels();
        System.out.println(predictions.toString());
    }
}
```

Shogun in the Cloud

- ▶ We love (I)Python notebooks for documentation
- ▶ IPython notebook server: try Shogun **without installation**
- ▶ Interactive web-demos (Django)



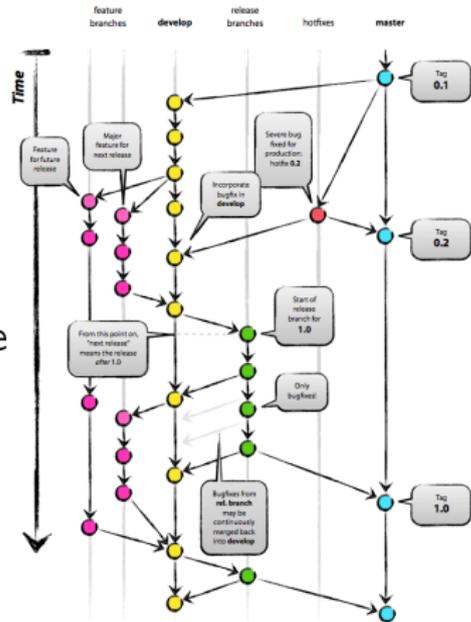
The screenshot shows a web interface for a digit recognition demo. At the top, there's a navigation bar with a close button (X) and links for "Open in new window / Try shogun cloud". The main heading is "Digit Recognize" followed by "based on shogun-toolbox". Below this, a paragraph explains: "This demo application uses a previously trained CGMNPsvm svm in combination with the Gaussian Kernel to recognize hand-written digits. To test it, draw a digit (0..9) in the area below and press recognize!". The central part of the interface features a large grid where a digit '2' has been drawn in green. To the right of the grid, there is a "Preview" button above a smaller grid showing the digit as a black binary image. Below the preview grid are "Recognize" and "Clear" buttons. At the bottom right, the text "The prediction is" is followed by the number "2".

<http://www.shogun-toolbox.org/page/documentation/notebook>

<http://www.shogun-toolbox.org/page/documentation/demo>

Development model

- ▶ Github
 - ▶ Pull requests
 - ▶ Intense discussions
 - ▶ Git flow model
- ▶ Extensive test coverage
- ▶ Travis: Tests pass **before** merge
- ▶ Buildbot: All OS, nightly binaries, memory checks, notebooks, documentation



<http://buildbot.shogun-toolbox.org/>

shogun-toolbox	a4e7ce28b8dd... in develop	eb1f080e75db... in develop	67d55553fa37... in develop	d923cb10ed5e... in develop	4bfa55e61686... in develop
FC19 - libshogun	OK	OK		OK	OK
FCRH - libshogun	OK	OK		OK	OK
bsd1 - libshogun	OK	OK		OK	OK
clang34 - static analysis	OK	OK			OK
clang34 - thread analysis	warnings test	warnings test			warnings test
clang34 - undefined behaviour analysis	warnings test	warnings test	warnings test		warnings test
coverity_analysis (waiting)					
cyg1 - libshogun (offline, plus 18)					
deb1 - libshogun	OK	OK		OK	OK
deb2 - static interfaces	OK	OK			OK
deb3 - modular interfaces	OK	OK			OK
deb4 - python3	OK	OK	OK	OK	OK
debian wheezy - memcheck	failed memory check	failed memory check			failed memory check
nightly_all (waiting)		OK			
nightly_default (waiting)		failed notebooks			
nightly_fedora (waiting)		failed git			
nightly_none (waiting)		OK			
osx1 - libshogun (offline, plus 48)					
osx2 - python (offline)					
precise - libshogun	OK	OK		OK	OK
rpm1 - libshogun	OK	OK		OK	OK

Outline

Overview

Machine Learning Features

Technical Features

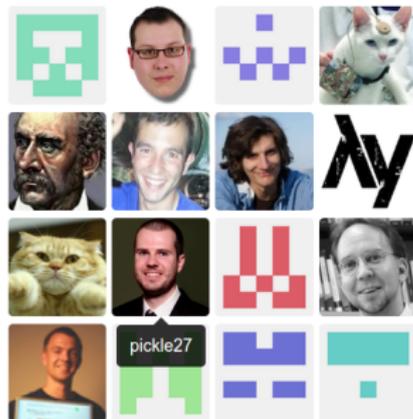
Community

Strong vibrations

- ▶ Active mailing list
- ▶ Populated IRC (come say hello)
- ▶ Cool team & backgrounds

Members

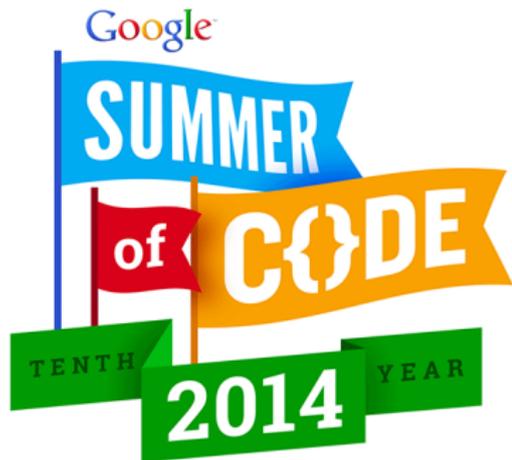
18 >



<http://www.shogun-toolbox.org/page/contact/contacts>

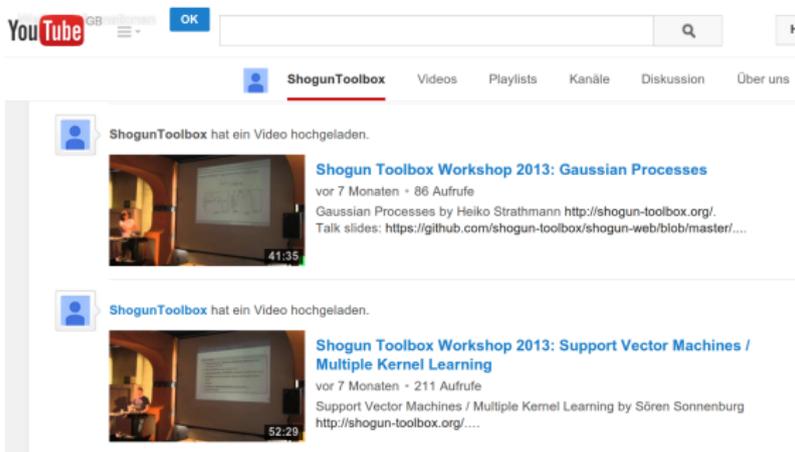
Google Summer of Code

- ▶ Student works full time during the summer
- ▶ Receives \$5000 stipend
- ▶ Work remains open-source
- ▶ At this moment!
- ▶ 29 x 3 months (we have lots of impact)



Future Ideas

- ▶ We just founded a non-profit association
- ▶ Goal: Take donations and hire a full-time developer
- ▶ GPL → BSD (industry friendly)
- ▶ Shogun in education, fundamental ML
- ▶ We organise workshops (next July 27, 28, c-base)



The screenshot shows the YouTube channel page for 'ShogunToolbox'. At the top, there is a search bar and navigation tabs for 'Videos', 'Playlists', 'Kanäle', 'Diskussion', and 'Über uns'. The channel name 'ShogunToolbox' is highlighted. Below the navigation, there are two video uploads:

- ShogunToolbox hat ein Video hochgeladen.**
Shogun Toolbox Workshop 2013: Gaussian Processes
vor 7 Monaten · 86 Aufrufe
Gaussian Processes by Heiko Strathmann <http://shogun-toolbox.org/>.
Talk slides: <https://github.com/shogun-toolbox/shogun-web/blob/master/...>
- ShogunToolbox hat ein Video hochgeladen.**
Shogun Toolbox Workshop 2013: Support Vector Machines / Multiple Kernel Learning
vor 7 Monaten · 211 Aufrufe
Support Vector Machines / Multiple Kernel Learning by Sören Sonnenburg
<http://shogun-toolbox.org/...>

Help!

- ▶ We don't sleep.
- ▶ You could:
 - ▶ Use Shogun and give us feedback
 - ▶ Fix bugs (see github), help us with framework design
 - ▶ Write (Python) examples and notebooks
 - ▶ Write documentation and update our website (Django)
 - ▶ Implement Super-parametric Massively Parallel Manifold Tree Classification Samplers (tm)
 - ▶ Mentor GSoC projects, or join as a student
 - ▶ Give us your money (we expose your logo to a large community)
 - ▶ **Come to the workshop on Sunday/Monday**

Thanks!

Questions?

<http://www.shogun-toolbox.org>