The SHOGUN Machine Learning Toolbox

(and its R interface)

Sören Sonnenburg[1,2], Gunnar Rätsch[2],Sebastian Henschel[2], Christian Widmer[2],Jonas Behr[2],Alexander Zien[2],Fabio De Bona[2],Alexander Binder[1],Christian Gehl[1], and Vojtech Franc[3]

[1] Berlin Institute of Technology, Germany
[2] Friedrich Miescher Laboratory, Max Planck Society, Germany
[3] Center for Machine Perception, Czech Republic

Outline

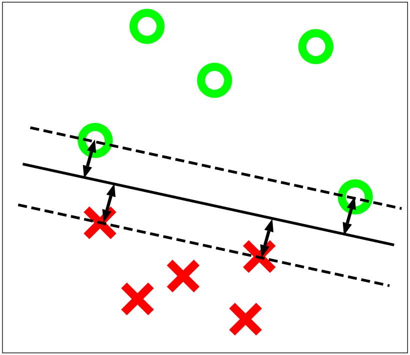
Introduction

What can you do with the SHOGUN Machine Learning Toolbox [6]?

- Types of problems:
  - Clustering (no labels)
  - **Classification** (binary labels)
  - Regression (real valued labels)
  - Structured Output Learning (structured labels)
- Main focus is on **Support Vector Machines** (SVMs)
- Also implements a number of other ML methods like
  - Hidden Markov Models (HMMs)
  - Linear Discriminant Analysis (LDA)
  - Kernel Perceptrons

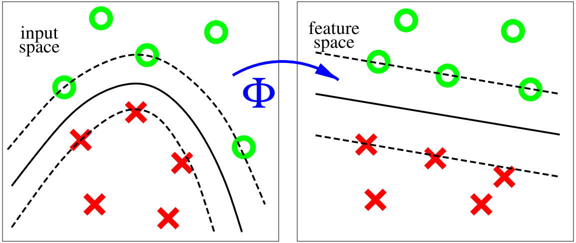
Support Vector Machine

- Given: Points $\mathbf{x}_i \in \mathcal{X}$ $(i = 1, \ldots, N)$ with labels $y_i \in \{-1, +1\}$
- Task: Find hyperplane that maximizes margin



Decision function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$

SVM with Kernels



- SVM decision function in kernel feature space:

$$f(\mathbf{x}) = \sum_{i=1}^{N} y_i \alpha_i \underbrace{\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}_i)}_{=k(\mathbf{x},\mathbf{x}_i)} + b \qquad (1)$$

- Training: Find parameters α
- Corresponds to solving quadratic optimization problem (QP)

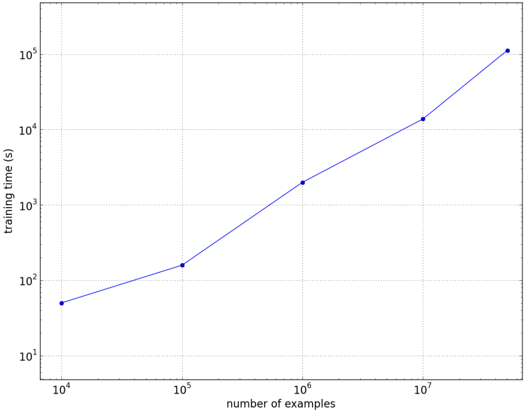
Large-Scale SVM Implementations

- Different SVM solvers employ different strategies
- Provides generic interface to 11 SVM solvers
- Established implementations for solving SVMs with kernels
  - `LibSVM`
  - `SVM`^{light}
- More recent developments: Fast linear SVM solvers
  - `LibLinear`
  - `SvmOCAS` [1]
- Support of Multi-Threading

⇒ We have trained SVMs with up to 50 million training examples

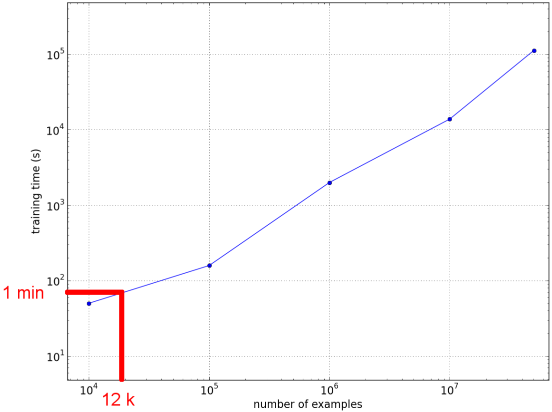
Large Scale Computations

- Training time vs sample size



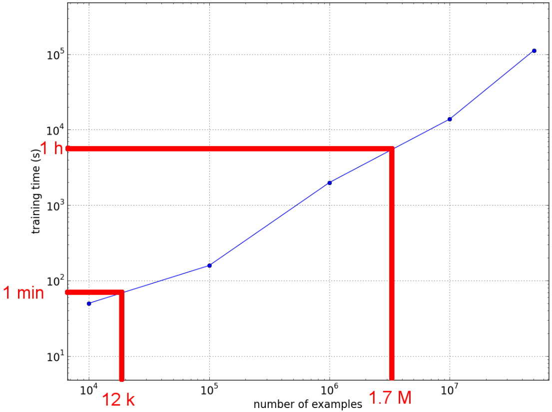
Large Scale Computations

- Training time vs sample size



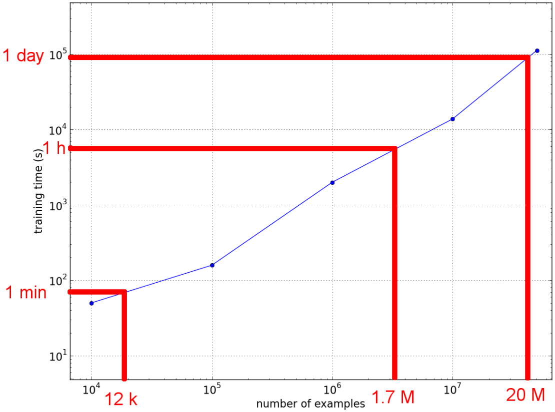
Large Scale Computations

- Training time vs sample size



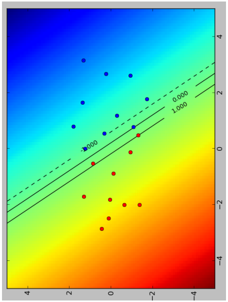
Large Scale Computations

- Training time vs sample size

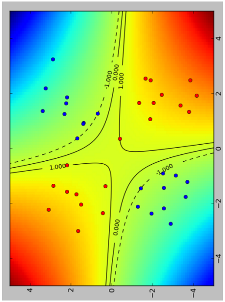


Various Kernel Functions

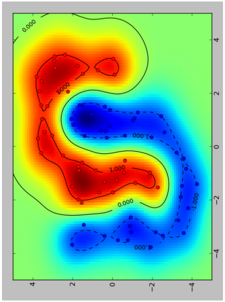
- Kernels for real-valued data



(a) Linear



(b) Polynomial

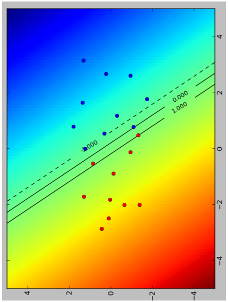


(c) Gaussian

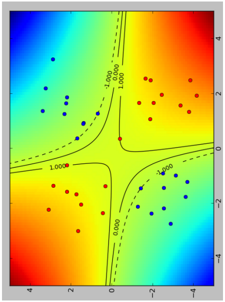
⇒ What if my data looked like...

Various Kernel Functions

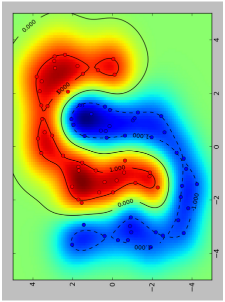
- Kernels for real-valued data



(d) Linear



(e) Polynomial



(f) Gaussian

$\Rightarrow$ What if my data looked like...

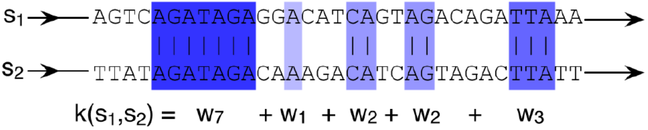
Various Kernel Functions

...this?!

TTTTGGAAGGAACGGAGGATCCGGATAGGCTTCATCCAAATATTCGATTATTGCCAAGCTTTCTGTCAGTGATAAGCCATTTATCACTAAAGTTGGAACTTTTTTCGCTGGGTTGTGCTTGACGAACTCTGCATTGTTCTGAAATAC
AGCTATATTTCTATGGAAGTGGGAGTAGGGCAGGGGTGGATGGCAAATTGCCGGTTTGCCGGAAATTTTCAATTCCGGCAATTTACCGATTTACCGTTTGCCGAATTTTACCGATTAAAATTGAAATTCTGAAATTTCCAGAAAAA
AATGTGCAAAACCACAAGTTGCCGAAAGTTTTCGGCAAATTGTGTTTTTCCGGCAAATCAGCGATTTGCCGTTCTGCCGAATTTTGCCAGAAATTTTCAATTCCGGCAATTTTCCAATTTACCGGAAATTTTCAATTCTAGCAATTT
GCTGGAAATTTCAATTCAGGCAATATGCCGAATTTGCCGAAAATTTTCAATTCCGGCAATTTTCCAATTTGCCCGAAAATTTTGATTTTCATTAAATTTTCGATTTGCCGGAAATTGTAATTTTGGGCAAATTGCCGATTTACCGTTG
CCAAAAGTTTTCGACAAATTGTGGTTTTTTCCGGCAAATCGGCAATTTGCCGCTAGGCCGAATTTTGCCGGAAATTTAAATTTCGGCACTTTGCCGATTTGCCGAAAATTTTCAATCCCGGCAATTTGCCGCTCTGGCGAATTTTGC
CGGAAATTTTCAATTCCGGCAATTTTCCAATTTGCCGGAAATTTTCTAACCTTGGATTCTTCAGAAAATAAGTCGATTGGCCGGTATTCGTAGTCGATATTTTTCAGAGCAAGTGCTATAAATCAGTATTTTATAGAATTTCCAAGCTA
AAATTACCGATGCGAACTCTCCAGGCACACGACGAACGCCAATATGAATATAAAATCGGTTTTGCCATGGAGTTAGAAGCTTCTAAAGAAGAAAATTGAAATATTTAAGCAATAAATGACATTTTGAGAAGAAATTAAATATGAAC
ACGTGGGAAAAATCTTCAAAAACCCTAATTTTTCAGTTAAAAACTACTTTCTCCGAATAAAACCTCCATTTTCCTCAATTTTCAGCACTTTTTTATGTCAATTCCCACTTAAAATTCCCCTTTAACGGTCATTTTTCCTAAACTGTGC
AGTTTCGCGTCGCCGTGTTTGCACACATTCCGTTTGATGCGCAGGTTTCCACGTTGTCCTCTTTTCTCCATTTTCTGTCTCTTTTAACGTGGATTTACCGATTTTTATCTCTGAAATTCTTAATTTTTATGGTATTTTTCGTAGTTTCTTT
ATCAATTTAATCGAAATTCCAAGTTTTTTCATTGATTTTTTGCATTTTTGTCTGTTTTTTTTAAACATTCTTATATATCCACGTGTCATTTCCATAATTTCAGAACCTTCATTCACGCCGAGGAGCAGTTTTTGACTGGTTACCGACTATA
AATCGGATATCATGTCGCTCCTTATCAAATTTGAATCAAAATCCGCTAGGGTTAAAGGTAAATATTTTGGTTTTTGGCATTATTTTTCAATAATTTCCCAATTTTCCAGGAATTTCGTTCCACCCAACTCGGCCATGGGTGCTCACATC
GCTGCACAGTGGAGTCATTCAACTCTGGGACTACCGAATGTGCGTTCTTCTCGAGAAGTTCGACGAGCACGATGGTCCAGTGAGAGGAATTTGCTTCCATCATGATCAGCCGATCTTTGTGTCTGGAGGAGATGATTATAAGATTA
AAGTGTGGAATTACAAGCAGAAACGCTGTATTTTCACACTTCTCGGACATTTGGATTATATTCGTACCACTTTCTTCCATCATAAGTATCCATGGATCATTTCTGCGTCCGACGATCAAACGGTTCGGATTTGGAATTGGCAGAGCC
GAAACTCGATCGCCATCCTCACCGGACACAATCACTACGTCATGTGTGCTCAATTCCACCCAACCGAAGATCTTGTGGCTTCTGCCTCCTTGGATCAGACCGTCCGTGTTTGGGATATTTCTGGACTTCGGAAGAAGCAAATGCCT
GGTGGAGGTGCTCCAAGCCGGCCAACCGGAGGGCAACAAGCTGAGCTATTCGGGCAACCTGATGCAGTTGTAGAGGGACACGATCGTGGAGTCAACTGGGTTGCATTCCATCATACCAATCCAATTCTAGTC
TCGGGATCCGATGATCGTCAAGTCAAGATTTGGCGTTACAATGAGACGAAAGCCTGGGAGCTCGACTCGTGCCGTGGACACTACAATAACGTCTCATCTGTGATCTTCCATCCAAATGCTGACTTGATTCTTTCCAACTCGGAGGA
TAAGTCGATTCGTGTGTGGGATATGCAGAAGCGTACCAGTCTTCACGTTTTCCGACATGAAAATGAGCGTTTCTGGGTTTTGGCCGCCCATCCAAGCTTGAACATGTTCGCCGCAGGACATGACAATGGAATGGTCGTTTTTAAAA
TTCAAAGAGAACGCCCGGCTTATTGTGTCAGTGATAATTTGGTTTTCTATGTGAAGGGACAACAAATTCGCAAGCTCGATTTGACAACCAAGTGAGTAGTTTGATGCCAAGAAAAGTGTGCCGCAAAATGCATTTTTGCACAGAA
TTAGCGTTTTTTTTTTGCGAAAAAAAGCGTGCGAGAGTTGCGAAAATGCATTTTCCGGCGCACTGAATCATGTTTTTGTGGAGTCTCAGTAGTTTTTAAATGAAAAACTTTAAATGTGCCCATTAAGCGAATTTCGTATTTAGAGC
CTCCAAAAAATCGGAAAAAAGGCGTAATTTCAATAGCTAAATTCTCCAACTTCCACTGAAAATAGCAGAGTTTTTTTTGAGATTTATCAAATTTTTAGCTTTTCAAAAAATGTTTTCTGCTTTTTACGAACAAATCTTTTGTTAAAC
AATGAAAATCGAATTTAAAATGACTAAAATTAACTTTTTCCGTCCACTGGTTGTGAAATGGGTTGAAATTGAAGAAATCAACGGGATTTTTCTGATTTTCTGAATATTTTTCTATTAGAAATCGGTTTTAAACCATTTTTTTTACTAA
AACCGATTTTTAATAGGAGAATATTTAGAAAATGAGAAAAATCCCGTTGATTTCTTCAATTTCAACCCATTTCACAACCAGTGGACGAAAAAAGTTAATTTTAGTCATTTTAAATTCGATTTTCACTGTTTAACAGAAGGTTTGTTC
GTAAAATGCAGGAAAAAAAATTTTTAAAGCTAAAAATTTGATAAATCTAAAAAAATCTGTTATTTTCAGTACGAAGTTGGAGATCTTAGCTAAATTTAGAGCTTTTCTCCGGAAATTTTGAGGACTCTCACTACAAAATTCGTTATTTT
TTGTACATTCAGAGGGGATGTACAAGCAAAAACCCCTCTAAAATCAAATTTTCAGCAAGGATGTTGCTCTGTGCAAGCTCCGTTACCCACAACCATTCATGCAACCATACTACTCTCTTTCCTTCAATCCTGCCGAGGGAACCTTC
CTTCTCACAAGCCGTACTCACAATAAGGATCTATGCGCCTTTGAGCTGTACAAAGTTGCCAGTAACTCTGATGGAACCACTGAAGCTGCGTGCGTCAAGAGCACCGGAATCAATGCACTTTGGGTTGCTCGTAATCGTTTTGCAG
TTCTTGACAAGGCTCAGAATGTGTCGTTGCGTGATTTGACGAACAAGGAACTTCGCAAACTCGAGAATATCAACACGGCTGTCGACGATATTTTCTATGCCGGAACTGGAATGCTCCTTTTGAGAAATGATGATGGGCTTCAACT
GTTTGATGTTCAGCAAAAGATTGTGACTGCTTCAGTGAAAGTGTCGAAAGTTCGTTATGTGATTTGGAGCAAGTCTATGGAATTCGCAGCTCTTCTTTCAAAGCACACGCTGACCTTGGTCAACCGTAAATTGGAGATTCTGTGTA
CCCAACAGGAGTCGACTCGTGTGAAGAGTGGAGCTTGGGATGATGACTCGGTATTCCTCTACACCACTAGCAATCACATCAAATATGCAATCAATTCTGGAGATTGTGGAATCGTTCGTACTCTGGATCTTCCTCTGTACATTCTG
GCTATTCGAGGAAACGTGCTCTACTGCTTGAATCGTGACGCTACACCAGTTGAAGTTCCAATTGATAATTCTGATTACAAGTTTAAGCTTGCTTTGATTAACAAACGAATCGATGAAGTTGTCAACATGGTTCGATCCGCAAATCTC
GTCGGTCAATCCATCATTGGATATCTCGAGAAAAAGGGTTACCCAGAGATTGCTCTTCACTTTGTCAAGGATGAGAAGACTCGCTTCGGGTTGGCAATTGAATGTGGAAACCTGGAAATCGCGTTGGAAGCCGCCAAGAAGCTC
GACGAGCCAGCAGTCTGGGAGGCTCTCGGAGAGACTGCTCTTCTTCAAGGCAATCATCAAATTGTTGAGATGAGCTATCAACGAACGAAAAACTTTGAGAAGCTCTCGTTTTTGTATTTTGTTACTGGAAATACGGATAAGCTTG
TCAAGATGATGAAGATTGCACAGGCTAGAAATGATGCTCACGGACATTTCCAGACAGCTCTCTACACTGGTGATGTTGAGGAGAGAGTTAAAGGTAAAAATGTGCAGCTGGATTTCAGAAGGGGGCAAAAGTCAAAGTTTTATT
GATTTTTAGCTCCTCTTTGCGTATTTTTTTCGTGAATTTCTGCAAATTCTAGACGATTTTGAAATTATTGAACCTAAAAATGGTCCAAATCTACCGAATATCGTCATGAGATTTCTCAAAAAAAAATTTGGAAATCTACCAATTTTCAG
CTAAAATCTCATATTTTGCCAACTTTTCAGTTTCGCAGCTGACGGAACTTGATTTTTATTCAATTATCGCTATTTAATACGTATTTTGGTCGTTTTAACGAATAAAAGTGTGATAGACGACCAAAATATGTATTAAAAAGTAATAATCA
ATGACACACTAAAGTTGGCAAAATTTGAGATTTTAGCTGAAAATAGGTTTTTTTCCAAATCTCAAAACCGCCATAACTTTTTTTGTGAGTTTTCAGAAACTTTTCAGGATTTTTCATGACGAAATTCGTTACTTTTTGGGTCATTTTC
AACAAAATAAAGTATAAAGTTAATTTTTCAGTGCTCCGAAACTGTGGACAAACTTCTCTGGCCTACCTCGCCGCAGCCACTCATGGATATTCTGCTGAAGCTGAGGAGTTGAAGGCTGAACTCGAGAGTCGTCAGCAACCAATCC
CACCAGTTGATCCAAATGCTCGCCTCCTTGTCCCACCACCACCAGTTGCTCGTTTAGAAGAGAATTGGCCACTTCTTGCCTCGGCTCGAGGAACTTTCGATGCACAGCTTCTTGGATTAGGAGGACAATCTGCTCCAACTAATGTT
GGCGGTGTGAAGCCTGCTGCTGCTGCCTTTGCTGTTATGGATGATGATGTTAGTTTATTGATTTCTGAGCTATCTGAGCTATAAAATATTAATTTTTGCAGGATGGTGATGTTGGAAATGAAGCATGGGGAGATGATGAGTATCTTGT
CGGAGAAGATGGTGAACTTGATGTTGATGAAGGAGAAGGACCAGTTGATGGAGATGAGGAAGGTGATGGGATGTGGATGATGATCTGGCTCTTCCTGATGTTACCGATGATCAAGGTGGAGATGATGATGAGGAAGTGGTTCC
TAACCCAGCTCCAGCTGTCTCATCGGAATGGCCGAATGTTTCTCGTCTTGCTGCTGATCATGTTGCAGCTGGTTCATTCGGAACTGCCATCAAGCTTCTTCATGATACAATTGGAGTCGTTGAAGCAGCACCATTCAAGGATGTATT
CGTGAAGGCTTATGCAGCTTCTAGACTCTCACATCGTGGATGGGGAGGACTCGGACCAGCTGGACCAGTGTCCATTCATCCGGTTCGTAATTTCCAGGAGGATAAGAATCATCTTCCAGTCGCTGCGTTCAAACTTTCTCAACTTG
CAAAGAAACTGCAAAAGGCTTATCAAATGACGACGAATGGAAAGTTTGGAGATGCTGTCGAGAAACTGCGGGAGATTCTTCTATCTGTTCCACTGATCGTTGTGTCTTCGAAACAAGAAGTCGCCGAAGCGGAGCAGCTCATC
ACAATCACACGGGAATATCTTGCAGCACTTCTCCTTGAAACTTATCGAAAAGATTTGCCAAAAACGAATTTGGAGGATGCGAAACGCAACGCCGAACTAGCTGCCTACTTTACACACTTTGAGCTTCAACCAATGCATCGTATCC
TGACACTGAGATCTGCTATCAATACTTTCTTCAAAATGAAGCAAATGAAGACTTGTGCATCACTTTGCAAACGTCTATTGGAGCTCGGTACGTGCTTGCGTTGTTGATTTTTTTAAATTTTATTTCTAATTCTTAATTTTTTTTTCCAG
CTCCAAAACCAGAAGTTGCTGCTCAAATCCGAAAAGTGCTGACTGCTGCCGCGAGAAGGATAACACTGACGCGCATCAGTTGACCTATGATGAGCATAATCCATTCGTCGTGTGCTCTCGTCAATTTGTGCCTCTTTACCGTGGACG
TCCACTCTGTAAATGCCCGTATTGTGGAGCATCGTACTCTGAAGGGCTCGAGGGAGAAGTGTGCAACGTGTGCCAGGTTGCAGAAGTTGGAAAGAATGTTCTTGGGCTCCGAATCTCAACTCTGAAGAGCTAAAAGTTGTGAAA
TATGATAATTTTTTCGTTTTTTGTGTGATATTAATTATTGTAATTTATTTGCCTTGTTCTTCCACAATTCCCATTGCTTTTGGTCCCTTCATATTATTTTCCCCGTTGGGCGGCGGTTCATGTTTTCTATTTAAATTATGTGTTTTTAAAAATT
ATTTAAACCCCACAGAAAGATTATGATTGCATGCTGGTCTATTTTATTTGAATAAATCATGATTTTTTAACTTTTTTTCATTTATCTTTGGGGTAAGAAAATGTTCTCTAAATCAGTAATTTACATGGAGAAATTTGCTCGAAAAAATT
CAAAAAAAATACAGTAAGAAAATTGTCGTAAAGTCTAGGGTTTTGGTATTGCGGACTACGACAAAATGCACACAAACCATTTTGGCACCGGAAAAAGGCAAAAGCCAGGATTTTATCGCGTTTTGGCTCATTTTTCTTGCAAATT
TCCATGGAATTCAGTTAACATATACCTAATTATTCAGTTATTTCGAATTAATCAATAATTTTCAGTGAATTTCGCAATTTCTAAGTGGTCCAATGATGTCTGTCCACATATTTCCTTGAAAACTGATTGTTTTATCGGAATTTATTGATTT
AAAATAATATATATAATAAAATCTGAATTTTAGAATCATTTTTGACGTGGAAAAGACCCGAAAAATCCAAAAATCACGAATAAACTGAACAATTAAGGTATTAATTTTTGATCCCCCACCCCCAGTTTCTCAGTTTCTCAGCAAAGG
ATTAAAACAATTGTTCTTCTTCTTTTCCCCCTACAAGAATTTCCTGTGAATTTCAATCGTAAAAATATCGATTTTCTGTTTATTCTTTGGATGAGTCACCGCACTCGCCGATTGTTTAACTAGTCAGTTCGGCAGTTTTACGTTATTT
TTTCACGATTTTTTGGATTCTGATCCTCTAGGTGACTTGAAAATACCGAATTTCATATCGTAATATGACAGAAAAAATTTATGGCGGTAAAAAAAATGCGGAAAAAATTTTCTAATTTTAACGGAAAACTCCAAAAAGCCGTTGGAAAAT
GATTTTTATTTGATATTGGCATATTTAAATAATTTTTATCCTGTTTTTTAGTGGATTTGGATATTTTAACCCGATAGTTATGTATAACGGCTTAAAAATTGCCAAAATCAGCGAAAAAACGCGAGAAAAATCACTTAAATATGGAAAATA

Various Kernel Functions

- String Kernels
  - Applications in Bioinformatics [3, 5, 7], Intrusion Detection
  - Idea of Weighted Degree String Kernel



- Heterogeneous Data Sources
  - CombinedKernel class to construct kernel from weighted linear combination of subkernels

$$K(x,z) = \sum_{i=1}^{M} \beta_i \cdot K_i(x,z)$$

  - β_i can be learned using Multiple Kernel Learning [4, 2]

Interoperability

- Supports many programming languages
  - Core written in C++ ($> 130,000$ lines of code)
  - R-bindings using SWIG (Simple Wrapper Interface Generator)
  - Additional bindings: Python, Matlab, Octave
  - More to come, e.g. Java
- Supports many data formats
  - `SVM`^{light}, `LibSVM`, `CSV`
  - HDF5
- Community Integration
  - Documentation available, many many examples (> 600)
  - Source code is freely available
  - There is a Debian Package, MacOSX
  - Mailing-List, public SVN repository (read-only)
  - Part of MLOSS.org

Simple Code Example

Simple code example: SVM Training

```
# given: features, labels, test as R-data structures
lab <- Labels(labels)
train <- RealFeatures(features)
gk <- GaussianKernel(train, train, 1.0)
svm <- LibSVM(10.0, gk, lab)
svm$train()
out <- svm$predict(test)
```

- It's easy to train & predict
- Generic interface to many solvers (e.g. LibSVM $\rightarrow$ SVMLight)
- SVM accepts any kernel (e.g. GaussianKernel $\rightarrow$ PolyKernel)

When is SHOGUN for you?

- You want to work with SVMs (11 solvers to choose from)
- You want to work with Kernels (35 different kernels)
  $\Rightarrow$ Esp.: String Kernels / combinations of Kernels
- You have large scale computations to do (up to 50 million)
- You use one of the following languages:
  R, Python, octave/MATLAB, C++
- Community matters: mloss.org, mldata.org

Thank you!

Thank you for your attention!!

For more information, visit:

- Implementation `http://www.shogun-toolbox.org`
- More machine learning software `http://mloss.org`
- Machine Learning Data `http://mldata.org`

References I

V. Franc and S. Sonnenburg.
Optimized cutting plane algorithm for large-scale risk minimization.
The Journal of Machine Learning Research, 10:2157–2192, 2009.

M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K.R. Müller, and A. Zien.
Efficient and accurate lp-norm multiple kernel learning.
Advances in Neural Information Processing Systems, 22(22):997–1005, 2009.

G. Schweikert, A. Zien, G. Zeller, J. Behr, C. Dieterich, C.S. Ong, P. Philips, F. De Bona, L. Hartmann, A. Bohlen, et al.
mGene: Accurate SVM-based gene finding with an application to nematode genomes.
Genome research, 19(11):2133, 2009.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf.
Large scale multiple kernel learning.
The Journal of Machine Learning Research, 7:1565, 2006.

References II

S. Sonnenburg, A. Zien, and G. Rätsch.
ARTS: accurate recognition of transcription starts in human.
Bioinformatics, 22(14):e472, 2006.

Sören Sonnenburg, Gunnar Rätsch, Sebastian Henschel, Christian Widmer, Jonas Behr, Alexander Zien, Fabio de Bona, Alexander Binder, Christian Gehl, and Vojtech Franc.
The SHOGUN machine learning toolbox.
Journal of Machine Learning Research, 2010.
(accepted).

C. Widmer, J. Leiva, Y. Altun, and G. Raetsch.
Leveraging Sequence Classification by Taxonomy-based Multitask Learning.
In *Research in Computational Molecular Biology*, pages 522–534. Springer, 2010.