LIBLINEAR: a Library for Linear Classification

Chih-Jen Lin
Department of Computer Science
National Taiwan University

Talk at Open Machine Learning Workshop, August 22, 2014

Outline

Outline

History

- Before 2006, my group focused on developing a general kernel SVM package called LIBSVM (Chang and Lin, 2011)
- While visiting Yahoo! in 2006-2007, I found that training large documents via kernel is time consuming
- But for this type of data, accuracy of using linear and Gaussian (RBF) kernels does not differ much

History (Cont'd)

Data set	Linear Time	Linear Accuracy	RBF Kernel Time	RBF Kernel Accuracy
MNIST38	0.1	96.82	38.1	99.70
ijcnn1	1.6	91.81	26.8	98.69
covtype	1.4	76.37	46,695.8	96.11
news20	1.1	96.95	383.2	96.90
real-sim	0.3	97.44	938.3	97.82
yahoo-japan	3.1	92.63	20,955.2	93.31
webspam	25.7	93.35	15,681.8	99.26

Size reasonably large: e.g., yahoo-japan: 140k instances and 830k features

History (Cont'd)

- Without kernels, training/testing can be much faster
- Therefore, we started developing LIBLINEAR in around 2007

Outline

Binary Linear Classification

- Given training instances $\{(y_i, \mathbf{x}_i)\}_{i=1}^{l}$, $y_i \in \{-1, 1\}$, $\mathbf{x}_i \in \mathbf{R}^n$

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi(\mathbf{w}; \mathbf{x}_i, y_i)$$

- Example:

$$\xi(\mathbf{w}; \mathbf{x}_i, y_i) = \begin{cases} \log(1 + e^{-y_i\mathbf{w}^T\mathbf{x}_i}) & \text{logistic regression} \\ \max(0, 1 - y_i\mathbf{w}^T\mathbf{x}_i) & \text{L1-loss SVM} \\ \max(0, 1 - y_i\mathbf{w}^T\mathbf{x}_i)^2 & \text{L2-loss SVM} \end{cases}$$

Current Solvers

L2-regularized classification

- Newton method for primal LR and l2-loss SVM (Lin et al., 2008)
- Dual coordinate descent for dual SVM and LR (Hsieh et al., 2008; Yu et al., 2011)

L1-regularized classification

- Primal coordinate descent for l1-regularized l2-loss SVM (Yuan et al., 2010)
- A hybrid method (Newton + coordinate descent) for l1-regularized LR (Yuan et al., 2012)

Current Solvers (Cont'd)

L2-regularized regression

- Newton method for primal l2-loss SVR (Ho and Lin, 2012)
- Dual coordinate descent for dual SVR (Ho and Lin, 2012)

In addition to optimization methods, many implementation techniques were developed

Selection of Solvers

- We studied other methods such as primal coordinate descent, primal quasi-Newton
- They are also good.
- In the end (for l2-requlariized solvers) we chose one 1st-order method (coordinate descent), and one 2nd-order method (Newton)
- The former quickly gets a model for easy situations. The latter is for difficult situations

Design Principle

- The setting mainly follows LIBSVM
  - same (sparse) data format
  - command line and library functions
  - simplicity and easy of use
- However, their target users are different
  - LIBSVM: general users who may not know machine learning much
  - LIBLINEAR: more advanced users
- Currently LIBLINEAR is used by many Internet companies

Interfaces to Other Languages

- Matlab/Octave
- Python
- Ruby
- Perl
- Weka
- R
- and others

Parameter Selection

- The only parameter is C
- The same parameter-selection script (by cross validation) in LIBSVM can be used

Extensions

In the website LIBSVM Tools, we provide some extensions of LIBLINEAR. Examples:

- Large-scale linear rankSVM
- LIBLINEAR for incremental and decremental learning
- Disk-level linear classification
- Distributed LIBLINEAR

Outline

Parameter Selection

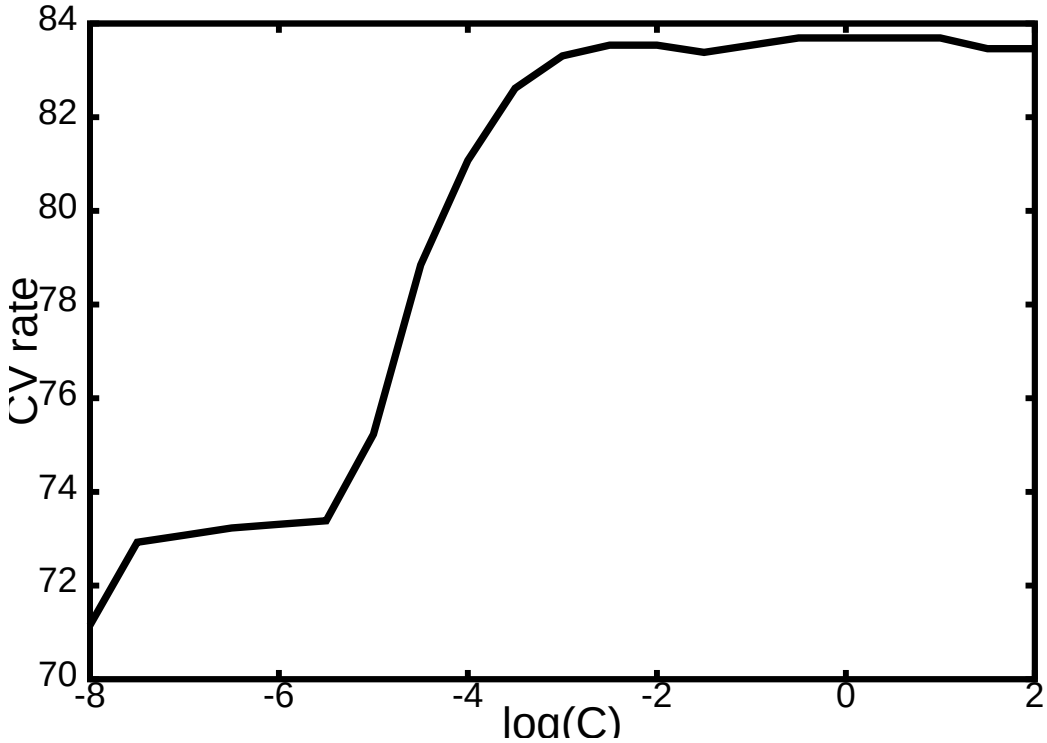
- If $C \to \infty$, optimal $\mathbf{w}^*$ approaches the solution of

$$\min_{\mathbf{w}} \quad \sum_{i=1}^{l} \xi(\mathbf{w}; \mathbf{x}_i, y_i)$$

- Indeed for l1-loss SVM, after $C \geq C^*$, $\mathbf{w}^*$ remains the same

Parameter Selection (Cont'd)

- Thus cross-validation accuracy is like



- When C is large, the default solver (dual coordinate descent) can be slow

Parameter Selection (Cont'd)

- But very large C is not necessary
- However, users may not be aware of this fact
- We hope to have some settings so users easily know that the model has stabilized

Data Scaling

- For kernel classifiers, data scaling is important. If feature values are large,

$$e^{-\gamma\|\mathbf{x}_i-\mathbf{x}_j\|^2} \to 0, \text{ if } i \neq j.$$

  Without feature-wise scaling, accuracy can be bad
- For linear classifiers, this problem does not exist
- However, range of feature values still affect the training speed

Data Scaling (Cont'd)

- Usually people normalize each document instance to have unit length
- But this normalization may not be good for non-document data
- We don't have a clear guideline for users yet

Distributed LIBLINEAR

- We are interested in extending LIBLINEAR to handle very large data
- After some studies, we recently released an extension for distributed classification
- See `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/distributed-liblinear`

Distributed LIBLINEAR

- We support both MPI and Spark
- Currently we have distributed Newton there, but not sure if that's the most suitable method
- The development is still in an early stage.

Conclusions

- In this talk, we have summarized the past development and current status of LIBLINEAR
- We have learned a lot from users in different application areas
- Overall, the experience of developing LIBLINEAR is quite rewarding

Acknowledgments

- All users have greatly helped us to make improvements

  Without them we cannot get this far
- We also thank all our past group members