Approximate Leave-One-Out Error Estimation for Learning with Smooth, Strictly Convex Margin Loss Functions

Christopher P. Diehl
Research and Technology Development Center
Johns Hopkins Applied Physics Laboratory
Chris.Diehl@jhuapl.edu

September 29, 2004

RTDC
RESEARCH & TECHNOLOGY DEVELOPMENT CENTER

Overview of the Talk

- General Learning Framework
- Model Selection
- LOO Error Estimation
- Unlearning and LOO Margin Estimation
- Comments on the Method
- Evaluating the Approximation
- Results
- Conclusion and Future Work

General Large Margin Learning Framework

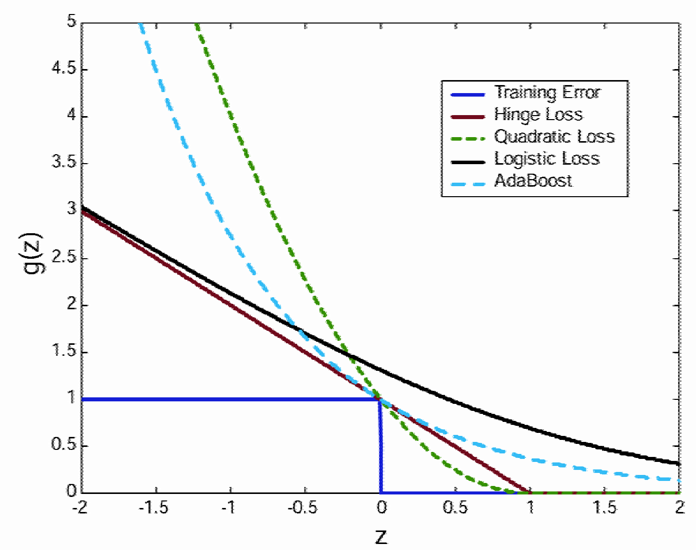
$$\min_{\boldsymbol{\alpha},b} L(\boldsymbol{\alpha},b) = \underbrace{\Omega(\boldsymbol{\alpha},b)}_{\text{Structural Risk}} + \underbrace{\sum_{i=1}^{N} C_i g(z_i)}_{\text{Empirical Risk}}$$

where

$$z_i = y_i f(x_i)$$

$$f(x) = \sum_{j \in I_s} y_j \alpha_j K\left(x_j, x\right) + b$$

Ω and g are strictly convex



- **Class of objective functions includes**
  - Smoothed Support Vector Machines
  - Kernel Logistic Regression

Model Selection

- **The Goal**
  - Identify the model that yields the best generalization performance
- **The Need**
  - An efficient means of estimating the generalization performance (error rate) using the available data
- **Common Approaches**
  - Validation Set
  - K-fold Cross-Validation
  - Leave-One-Out (N-fold Cross-Validation)
    - Almost unbiased estimator of the generalization error
    - Valuable tool when data is scarce for one or more classes

LOO Error Estimation

- **Definition**

$$e_{LOO} = \tfrac{1}{N}\sum_{i=1}^{N} I\left\{ z_k^{(k)} < 0 \right\}$$ where $z_k^{(k)} = y_k f^{(k)}\left(x_k\right)$ and

$f^{(k)}$ is the classifier trained on all data except (x_k, y_k)

- **Approximation Strategy**

Estimate $f^{(k)}$ based on f to derive an estimate of the LOO margin $z_k^{(k)}$

Unlearning and LOO Margin Estimation

- **Repeated Steps in LOO**
  - Unlearn the example (x_k, y_k)
  - Compute the LOO margin $z_k^{(k)}$
- **Exact Unlearning**
  - Set regularization parameter $C_k = 0$ to remove example's influence
  - Update classifier parameters to minimize

$$\min_{\boldsymbol{\alpha},b} L^{(k)}(\boldsymbol{\alpha},b) = \Omega(\boldsymbol{\alpha},b) + \sum_{\substack{i=1 \\ i\neq k}}^{N} C_i g(z_i)$$

via multiple Newton steps

Unlearning and LOO Margin Estimation

- **Approximate Unlearning**
  - Compute only one Newton step to approximate the change in the classifier parameters

$$\begin{bmatrix}\Delta\boldsymbol{\alpha}\\ \Delta b\end{bmatrix} = -\left(\nabla^2{}_{\boldsymbol{\alpha},b} L^{(k)}\Big|_{(\boldsymbol{\alpha}_*,b_*)}\right)^{-1} \nabla_{\boldsymbol{\alpha},b} L^{(k)}\Big|_{(\boldsymbol{\alpha}_*,b_*)} \quad \text{where} \quad (\boldsymbol{\alpha}_*, b_*) = \arg\min L(\boldsymbol{\alpha}, b)$$

  - Estimate of LOO margin is then

$$z_k{}^{(k)} \approx z_k + y_k\left(\sum_{j\in I_S} y_j \Delta\alpha_j K(x_j, x_k) + \Delta b\right)$$

$$z_k{}^{(k)} \approx z_k + \frac{C_k g'(z_k)\hat{\mathbf{q}}_k{}^{\mathsf{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k}{1 - C_k g''(z_k)\hat{\mathbf{q}}_k{}^{\mathsf{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k}$$

where

$$\hat{\mathbf{q}}_k{}^{\mathsf{T}} = \left[y_j y_k K(x_j, x_k)\ \forall\ j \in I_S \ \cdots\ y_k\right]$$

$$\mathbf{H} = \nabla^2{}_{(\boldsymbol{\alpha},b)} L\Big|_{(\boldsymbol{\alpha}_*,b_*)}$$

RTDC
RESEARCH & TECHNOLOGY DEVELOPMENT CENTER

Comments about the Method

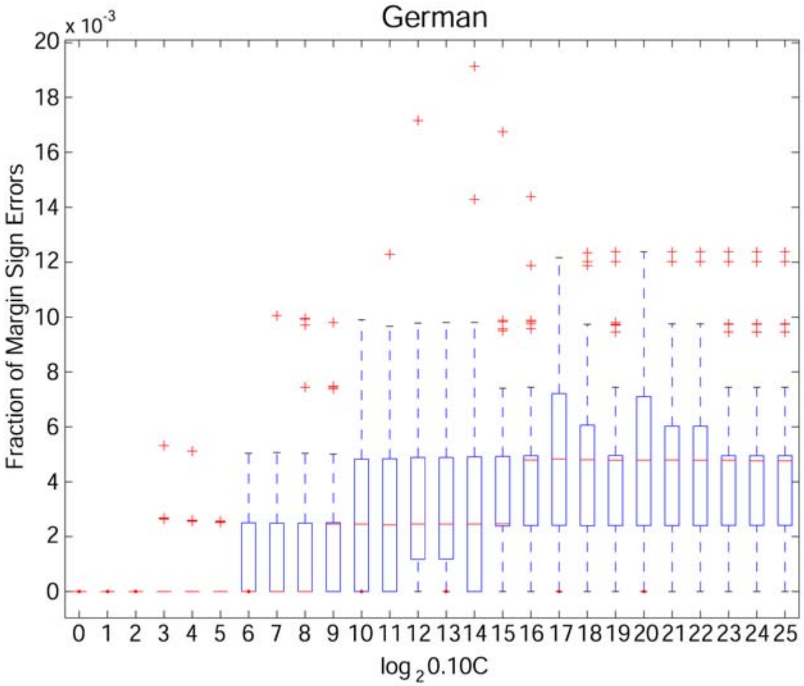
- **Main Computational Expense**
  - Computing $\hat{\mathbf{q}}_k^{\mathsf{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k$ for each example
  - Inverse Hessian available from training phase
- **Margin Sensitivity**
  - Limiting form of approximation yields $\dfrac{\partial z_k}{\partial C_k} = -g'(z_k)\hat{\mathbf{q}}_k^{\mathsf{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k \geq 0$
  - Margin increases monotonically with decreasing regularization (increasing C_k)
  - Implies $\Delta z_k \leq 0$ when unlearning the example – examples classified incorrectly will remain in error after unlearning
- **Generalizations**
  - Approximations can be derived for other strictly convex objective functions and cross-validation methods (e.g. K-fold cross-validation)

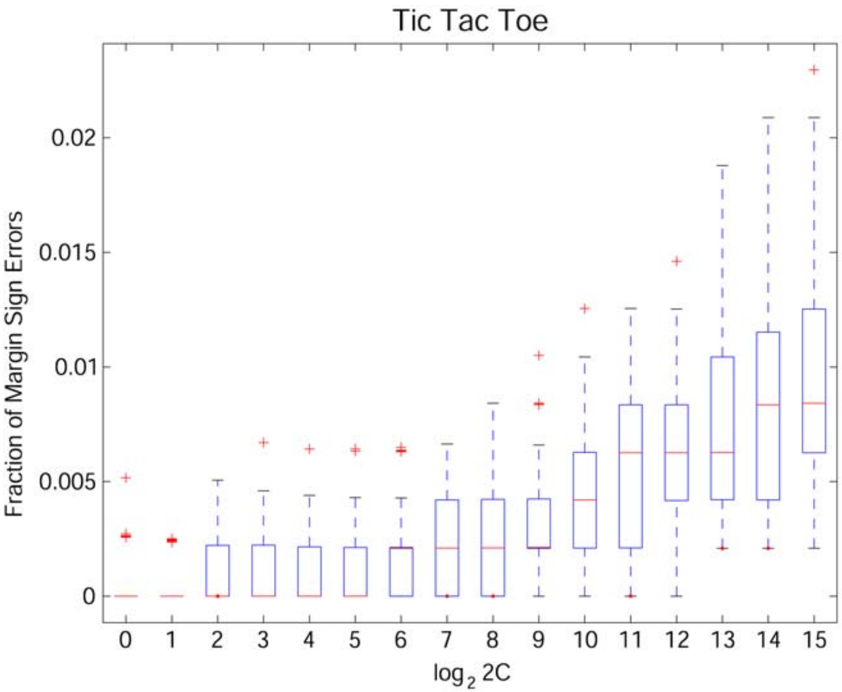
JOHNS HOPKINS
UNIVERSITY
Applied Physics Laboratory

Evaluating the Approximation

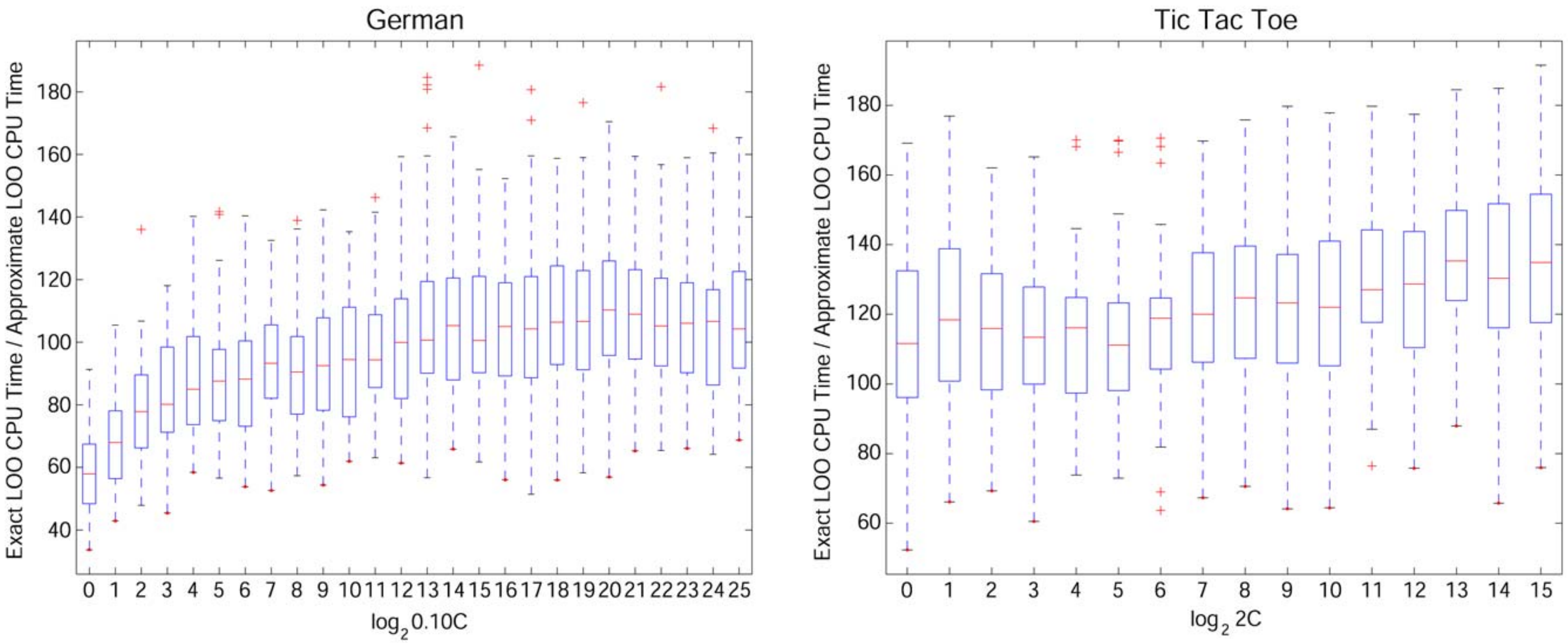
- **Two Questions**
  1. How much approximation error are we incurring?
  2. How much computational savings are we gaining?
- **Evaluation Approach**
  1. Compute fraction of margin sign errors
     - Fraction of examples that yield exact and approximate LOO margins with different signs
  2. Compute ratio of CPU time used to compute exact and approximate LOO error estimates
     - Used *cputime* command in MATLAB to collect measurements

Fraction of Margin Sign Errors



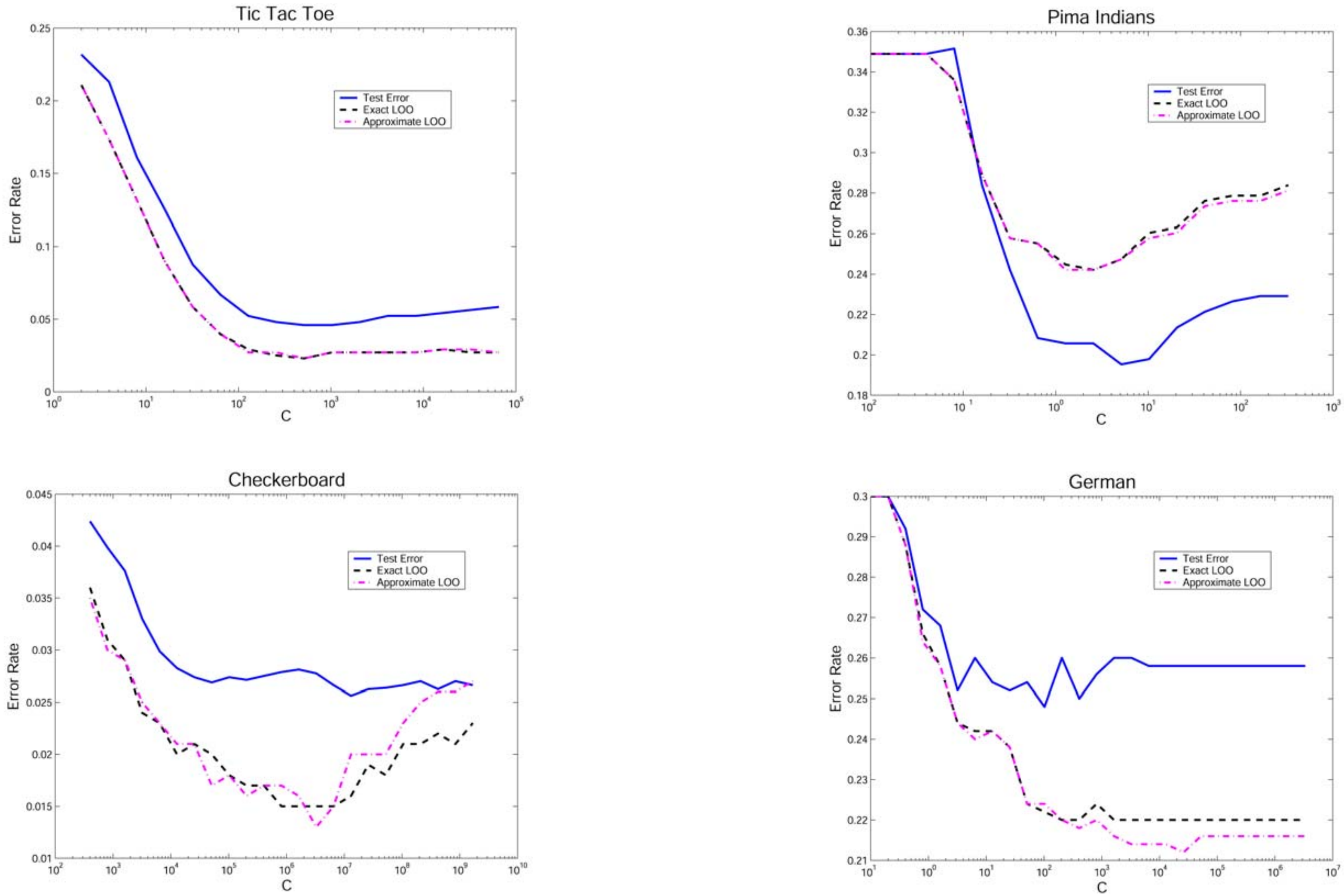


Reduction in CPU Time



RTDC
RESEARCH & TECHNOLOGY DEVELOPMENT CENTER

LOO for Model Selection



JOHNS HOPKINS
UNIVERSITY
Applied Physics Laboratory

Conclusion and Future Work

- **Recap**
  - Introduced approximate LOO error estimation method for strictly convex objective functions
  - Method estimates the change in the margin by computing a single Newton step to approximately unlearn a given example
- **Future Work**
  - Investigate causes of LOO error failing to track the test error
  - Define and evaluate method for model selection based on the LOO margin distribution