

Solving Non-Linear SVM in Linear Time? – A Nyström Approximated SVM with Applications to Image Classification

Ming-Hen Tsai, Academia Sinica, Taiwan (now at Google Inc.)

Joint work with

Yi-Ren Yeh, Intel-NTU Connected Context Computing Center

Yuh-Jye Lee, Dept. CSIE, National Taiwan University Science & Technology

Yu-Chiang Frank Wang, Research Center for IT Innovation, Academia Sinica

May 21, 2013

Outline

- Introduction
- Nyström Method for Kernel Approximation
- Nyström Method for Linear Representation
- Experiments and Conclusions

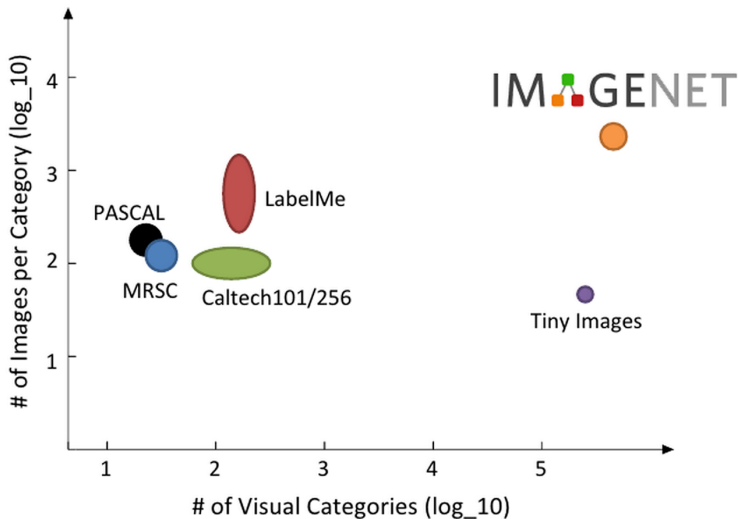
Outline

- Introduction
- Nyström Method for Kernel Approximation
- Nyström Method for Linear Representation
- Experiments and Conclusions

Visual Classification: Faces, Objects, and Beyond



Example: Some Image Classification Data Sets



We Have Known...

- Non-linear SVM: powerful but slow
- Linear SVM: simple but fast

Paper: "Training Linear SVMs in Linear Time" by Joachims

Software: LIBLINEAR, VW, etc.

We Had Always Wondered...

- But we want something **powerful** and **fast**: train faster than non-linear SVM and generate a more accurate model than linear SVM.

We Had Always Wondered...

- But we want something **powerful** and **fast**: train faster than non-linear SVM and generate a more accurate model than linear SVM.

What should we do?

We Had Always Wondered...

- But we want something **powerful** and **fast**: train faster than non-linear SVM and generate a more accurate model than linear SVM.

What should we do?

- Can we save some computations (for speed) and still get non-linearity (for more power)?

We Had Always Wondered...

- But we want something **powerful** and **fast**: train faster than non-linear SVM and generate a more accurate model than linear SVM.

What should we do?

- Can we save some computations (for speed) and still get non-linearity (for more power)?

Yes. Do approximation!

Outline

- Introduction
- **Nyström Method for Kernel Approximation**
- Nyström Method for Linear Representation
- Experiments and Conclusions

Definitions and a Brief Review (1/2)

- Input Data: $\{(\mathbf{y}, X)\} = \{(y_i, \mathbf{x}_i)\}_{i=1}^{\ell}$. y_i : label, \mathbf{x}_i : feature vector.
- (Non-zero) dimension of \mathbf{x}_i : n
- Non-linear mapping $\phi(\mathbf{x})$ (e.g. bi-gram features)
- Kernel function: $K(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{u})^T \phi(\mathbf{v})$, usually computed in linear time.
- For ease of representation, for $U = [\mathbf{u}_1, \dots, \mathbf{u}_{\ell}]$ and $V = [\mathbf{v}_1, \dots, \mathbf{v}_{\ell}]$, we define $Q = K(U, V)$, where $Q_{ij} = K(\mathbf{u}_i, \mathbf{v}_j)$

Definitions and a Brief Review (1/2)

- Input Data: $\{(\mathbf{y}, X)\} = \{(y_i, \mathbf{x}_i)\}_{i=1}^{\ell}$. y_i : label, \mathbf{x}_i : feature vector.
- (Non-zero) dimension of \mathbf{x}_i : n
- Non-linear mapping $\phi(\mathbf{x})$ (e.g. bi-gram features)
- Kernel function: $K(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{u})^T \phi(\mathbf{v})$, usually computed in linear time.
- For ease of representation, for $U = [\mathbf{u}_1, \dots, \mathbf{u}_{\ell}]$ and $V = [\mathbf{v}_1, \dots, \mathbf{v}_{\ell}]$, we define $Q = K(U, V)$, where $Q_{ij} = K(\mathbf{u}_i, \mathbf{v}_j)$
- Lower bound for data storage and training: $\Omega(\ell n)$.

Definitions and a Brief Review (2/2)

- Primal:

$$\min_{\mathbf{w}, b} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{\ell} \max(1 - y_i \mathbf{w}^T \phi(\mathbf{x}_i), 0)$$

- Dual:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

- Primal-Dual Correspondence: $\mathbf{w} = \sum_{i=1}^{\ell} y_i \alpha_i \phi(\mathbf{x}_i)$

Kernel in Non-linear SVM

- Dual form:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

- Kernel matrix Q with $Q_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$.

Kernel in Non-linear SVM

- Dual form:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

- Kernel matrix Q with $Q_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$.

Computational time: $O(\ell^2)$ times kernel products
 $\sim O(\ell^2 n) \sim \ell$ times data size.

Space: $O(\ell^2)$.

Kernel in Non-linear SVM

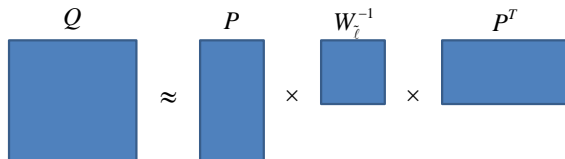
- Dual form:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

- Kernel matrix Q with $Q_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$.
Computational time: $O(\ell^2)$ times kernel products
 $\sim O(\ell^2 n) \sim \ell$ times data size.
Space: $O(\ell^2)$.
- $1M$ images, $1k$ dimensional features: $1M$ times more computational time, and $1M/1k = 1k$ times more space than the linear counterpart.

Nyström Method

- Sample $\tilde{\ell}$ feature vectors from X to a set $B = \{\mathbf{b}_i\}_{i=1}^{\tilde{\ell}}$.
- The approximated kernel: $Q \sim \tilde{Q} = P^T W^{-1} P$, where $P_{ij} = K(\mathbf{x}_i, \mathbf{b}_j)$ and $W_{ij} = K(\mathbf{b}_i, \mathbf{b}_j)$.

$$Q \approx P \times W_{\tilde{\ell}}^{-1} \times P^T$$


Nyström Method

- Sample $\tilde{\ell}$ feature vectors from X to a set $B = \{\mathbf{b}_i\}_{i=1}^{\tilde{\ell}}$.
- The approximated kernel: $Q \sim \tilde{Q} = P^T W^{-1} P$, where $P_{ij} = K(\mathbf{x}_i, \mathbf{b}_j)$ and $W_{ij} = K(\mathbf{b}_i, \mathbf{b}_j)$.

$$\begin{array}{ccccccc}
 & Q & & P & & W_{\tilde{\ell}}^{-1} & & P^T \\
 & \square & \approx & \square & \times & \square & \times & \square
 \end{array}$$

- Time and space complexity:
 Time: Basis selection time + $O(\ell^2 \max(\tilde{\ell}, n))$ overall.
 (Alright...)
 Space: $O(\ell \max(\tilde{\ell}, n))$. (Good! But we don't know how large is $\tilde{\ell}$.)

Nyström Method

- Sample $\tilde{\ell}$ feature vectors from X to a set $B = \{\mathbf{b}_i\}_{i=1}^{\tilde{\ell}}$.
- The approximated kernel: $Q \sim \tilde{Q} = P^T W^{-1} P$, where $P_{ij} = K(\mathbf{x}_i, \mathbf{b}_j)$ and $W_{ij} = K(\mathbf{b}_i, \mathbf{b}_j)$.

$$\begin{array}{ccccccc}
 & Q & & P & & W_{\tilde{\ell}}^{-1} & & P^T \\
 & \begin{array}{|c|} \hline \square \\ \hline \end{array} & \approx & \begin{array}{|c|} \hline \square \\ \hline \end{array} & \times & \begin{array}{|c|} \hline \square \\ \hline \end{array} & \times & \begin{array}{|c|} \hline \square \\ \hline \end{array}
 \end{array}$$

- Time and space complexity:
 Time: Basis selection time + $O(\ell^2 \max(\tilde{\ell}, n))$ overall.
 (Alright...)
 Space: $O(\ell \max(\tilde{\ell}, n))$. (Good! But we don't know how large is $\tilde{\ell}$.)
- Exists studies on how to get a compact and representative B .

Outline

- Introduction
- Nyström Method for Kernel Approximation
- **Nyström Method for Linear Representation**
- Experiments and Conclusions

An Equivalent Representation (1/2)

- Bottleneck for the previous method: kernel matrix Q of size $O(\ell^2)$. But observe the following...

An Equivalent Representation (1/2)

- Bottleneck for the previous method: kernel matrix Q of size $O(\ell^2)$. But observe the following...
- $Q = \tilde{X}^T \tilde{X}$ by Cholesky decomposition. (This is valid because Q is PSD by definition.)
- Investigate the columns of $\tilde{X} = [\tilde{\mathbf{x}}_1 \dots \tilde{\mathbf{x}}_\ell]$. We have $\tilde{\mathbf{x}}_i^T \tilde{\mathbf{x}}_j = Q_{ij} = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$.

An Equivalent Representation (1/2)

- Bottleneck for the previous method: kernel matrix Q of size $O(\ell^2)$. But observe the following...
- $Q = \tilde{X}^T \tilde{X}$ by Cholesky decomposition. (This is valid because Q is PSD by definition.)
- Investigate the columns of $\tilde{X} = [\tilde{\mathbf{x}}_1 \dots \tilde{\mathbf{x}}_\ell]$. We have $\tilde{\mathbf{x}}_i^T \tilde{\mathbf{x}}_j = Q_{ij} = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$.
- We represent the kernelized linear products as regular linear products. Let's call \tilde{X} a compact representation.

An Equivalent Representation (2/2)

- Dual:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

An Equivalent Representation (2/2)

- Dual:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

- Primal:

$$\min_{\mathbf{w}, b} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{\ell} \max(1 - y_i \mathbf{w}^T \tilde{\mathbf{x}}_i, 0)$$

An Equivalent Representation (2/2)

- Dual:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - \mathbf{e}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned}$$

- Primal:

$$\min_{\mathbf{w}, b} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{\ell} \max(1 - y_i \mathbf{w}^T \tilde{\mathbf{x}}_i, 0)$$

- Issue: computing $\tilde{\mathbf{x}}_i$ requires the computation of the ℓ^2 -sized kernel.

Nyströms' Equivalent Representation

- Let's go back to kernel approximation!

Nyströms' Equivalent Representation

- Let's go back to kernel approximation!
- As $\tilde{Q} = P^T W^{-1} P$, we can find a matrix R such that $W^{-1} = R^T R$, and RP is our compact representation.

Nyströms' Equivalent Representation

- Let's go back to kernel approximation!
- As $\tilde{Q} = P^T W^{-1} P$, we can find a matrix R such that $W^{-1} = R^T R$, and RP is our compact representation.

Time: Basis selection time + $O(\ell \tilde{\ell} n)$ overall.

Space: $O(\ell \max(\tilde{\ell}, n))$.

Nyströms' Equivalent Representation

- Let's go back to kernel approximation!
- As $\tilde{Q} = P^T W^{-1} P$, we can find a matrix R such that $W^{-1} = R^T R$, and RP is our compact representation.
Time: Basis selection time + $O(\ell \tilde{\ell} n)$ overall.
Space: $O(\ell \max(\tilde{\ell}, n))$.
- If we set our basis size, $\tilde{\ell}$, as n . The space consumption is optimal, but the speed still depends on the basis selection part.

Another View of Basis Selection (1/2)

- We view RP as input data and do feature selection on it to get a more compact feature set.
- Recall that: $P_{:,i} = [K(\mathbf{x}_i, \mathbf{b}_1), \dots, K(\mathbf{x}_i, \mathbf{b}_{\tilde{\ell}})]^T$
- This is not a basis selection algorithm because R introduces dependency between dimensions.

Another View of Basis Selection (1/2)

- We view RP as input data and do feature selection on it to get a more compact feature set.
- Recall that: $P_{:i} = [K(\mathbf{x}_i, \mathbf{b}_1), \dots, K(\mathbf{x}_i, \mathbf{b}_{\tilde{\ell}})]^T$
- This is not a basis selection algorithm because R introduces dependency between dimensions.
- We remove R ! Each dimension is then independent of other basis vector, we can view P as our linear SVM input data and do feature selection for basis selection.
- L1-regularized linear SVM which runs linear time in data size is used in the work.

Another View of Basis Selection (2/2)

- Does it hurt by removing R ?

Another View of Basis Selection (2/2)

- Does it hurt by removing R ?
- Say \mathbf{p}_+ and \mathbf{p}_- are two samples with different labels.

Another View of Basis Selection (2/2)

- Does it hurt by removing R ?
- Say \mathbf{p}_+ and \mathbf{p}_- are two samples with different labels.
- If for a \mathbf{w} , $\mathbf{w}^T \mathbf{p}_+ > \mathbf{w}^T \mathbf{p}_-$ (separable), then $\mathbf{w}^T R^{-1} R \mathbf{p}_+ > \mathbf{w}^T R^{-1} R \mathbf{p}_-$.

Another View of Basis Selection (2/2)

- Does it hurt by removing R ?
- Say \mathbf{p}_+ and \mathbf{p}_- are two samples with different labels.
- If for a \mathbf{w} , $\mathbf{w}^T \mathbf{p}_+ > \mathbf{w}^T \mathbf{p}_-$ (separable), then $\mathbf{w}^T R^{-1} R \mathbf{p}_+ > \mathbf{w}^T R^{-1} R \mathbf{p}_-$.
- $R^{-1} \mathbf{w}$ separates $R \mathbf{p}_+$ and $R \mathbf{p}_-$.

Another View of Basis Selection (2/2)

- Does it hurt by removing R ?
- Say \mathbf{p}_+ and \mathbf{p}_- are two samples with different labels.
- If for a \mathbf{w} , $\mathbf{w}^T \mathbf{p}_+ > \mathbf{w}^T \mathbf{p}_-$ (separable), then $\mathbf{w}^T R^{-1} R \mathbf{p}_+ > \mathbf{w}^T R^{-1} R \mathbf{p}_-$.
- $R^{-1} \mathbf{w}$ separates $R \mathbf{p}_+$ and $R \mathbf{p}_-$.
- When doing the linear transformation by R , we preserve separability.

A Linear Feature Selection Algorithm

Algorithm 2 Selection Algorithm, \mathcal{A}

Input Training data with labels (\mathbf{y}, X) . A kernel functions K . A number s , the total size of the bases. (Optional)

1. Get a set of basis, B^{big} , each with size $\sqrt{\ell}$. Randomly sample $\sqrt{\ell}$ instances with labels, $(\mathbf{y}^{subset}, X^{subset})$.
2. $X^{select} \leftarrow K(X^{subset}, B^{big})$.
3. Train L1-regularized SVM on $(\mathbf{y}^{subset}, X^{select})$ and a sparsity pattern \mathbf{w} for each class.
4. $B \leftarrow$ columns in B^{big} where the corresponding element in \mathbf{w} is not zero.
5. If s is defined, $B \leftarrow$ the s centroid generated by k-means of B_i (as proposed in [12].)

Return B

The Algorithm Workflow

- 1 Input data: (\mathbf{y}, X) , a kernel K .
- 2 Run some basis selection algorithm in linear time and get the basis B .
- 3 Compute the new data $\tilde{X} = K(X, B)\text{Chol}(K(B, B)^{-1})$.
- 4 Train on (\mathbf{y}, \tilde{X})

The Algorithm Workflow

- 1 Input data: (\mathbf{y}, X) , a kernel K .
- 2 Run some basis selection algorithm in linear time and get the basis B .
- 3 Compute the new data $\tilde{X} = K(X, B)\text{Chol}(K(B, B)^{-1})$.
- 4 Train on (\mathbf{y}, \tilde{X})

Time: $O(\sqrt{\ell}\sqrt{\ell}n) + O(\ell\tilde{\ell}n)$ overall.

Space: $O(\ell \max(\tilde{\ell}, n))$.

Outline

- Introduction
- Nyström Method for Kernel Approximation
- Nyström Method for Linear Representation
- Experiments and Conclusions

Experimental Settings

- We conduct experiments on two benchmark datasets: USPS and MNIST.
- We randomly split 70% of the data for training and the remaining 30% for testing.
- Repeat random split five times.
- We perform a five-fold cross validation to select the parameters γ and C .

Table: Dataset descriptions (with the number ℓ of instances and the dimension d of the data). The sizes for storing the data ℓd and the associated kernel matrices ℓ^2 are also listed.

	ℓ	d	kernel size ℓ^2	data size ℓd
USPS	7291	256	53M	2M
MNIST	60000	780	3.6G	47M

Discussions

- Our method achieved improved accuracy than linear SVMs
- The time for training and testing using our proposed model is comparable to that of linear SVMs
- The standard nonlinear SVM utilizes the full kernel matrix whose time complexity is quadratically scaled-up with ℓ .

Table: Comparisons of accuracy and computation time (in sec.) on three datasets. We denote our proposed Nyström approximated SVM as Nyström primal SVM below.

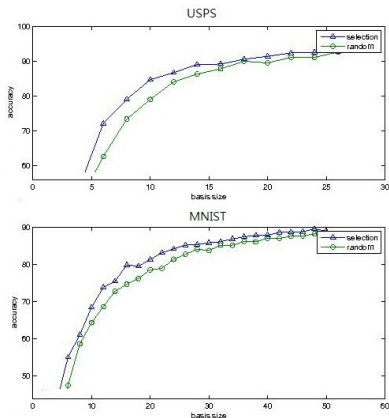
USPS	Accuracy	Training Time	Testing Time
Nyström primal SVM	97.057 ± 0.402	3.764	0.320
nonlinear SVM	98.007 ± 0.198	14.507	4.129
linear SVM	95.009 ± 0.275	2.274	0.079
MNIST	Accuracy	Training Time	Testing Time
Nyström primal SVM	93.833 ± 0.115	24.6092	2.006
nonlinear SVM	98.547 ± 0.066	3650.334	524.487
linear SVM	91.917 ± 0.077	14.510	0.381

Comparisons to Random Basis Selection

- To verify the effectiveness of our proposed basis selection approach, we compare the performance
 - Using the basis matrix determined by our method
 - Using the random sampling strategy
- We plot the recognition rates using these two different methods on the two datasets.
- Our selection method yields higher accuracy than random sampling when choosing the same basis size for most scenarios.

Comparisons to Random Basis Selection

Figure: Performance using our basis selection method (in blue) vs. random sampling (in green) on USPS (top) and MNIST (bottom) datasets.



Conclusion

- We proposed a method to solve an approximated dual SVM in the primal based on Nyström approximation.
- Our proposed method automatically determines the optimal basis matrix.
- The proposed method allows us to solve linear SVMs whose time complexity is simply scaling up with the dataset size.
- Experiments compared by linear SVM, our method
 - achieved improved recognition accuracy
 - reported comparable computation time
- Both training and testing time using our approximated model was remarkably reduced compared to that of nonlinear SVMs.