

A Fast Dual Method for HIK SVM Learning

Jianxin Wu

Nanyang Technological University

Abstract. Histograms are used in almost every aspect of computer vision, from visual descriptors to image representations. Histogram Intersection Kernel (HIK) and SVM classifiers are shown to be very effective in dealing with histograms. This paper presents three contributions concerning HIK SVM classification. First, instead of limited to integer histograms, we present a proof that HIK is a positive definite kernel for non-negative real-valued feature vectors. This proof reveals some interesting properties of the kernel. Second, we propose ICD, a deterministic and highly scalable dual space HIK SVM solver. ICD is faster than and has similar accuracies with general purpose SVM solvers and two recently proposed stochastic fast HIK SVM training methods. Third, we empirically show that ICD is not sensitive to the C parameter in SVM. ICD achieves high accuracies using its default parameters in many datasets. This is a very attractive property because many vision problems are too large to choose SVM parameters using cross-validation.

1 Introduction

Recently, the Histogram Intersection Kernel (HIK) has attracted a lot of attention in the computer vision community. The success of HIK can be attributed to at least two important factors:

- First, histograms are frequently used in solving vision problems. At the feature level, many visual descriptors are histograms of various image measurements, e.g., SIFT [1], HOG [2], CENTRIST [3], or histogram of LBP [4], just to name a few. At the image level, histogram is also a popular representation, e.g., color histogram [5] or bag of visual words.
- Second, it is shown that HIK, as a measure for comparing the similarity (or dissimilarity) of *two histograms*, achieves better performances in various machine learning tasks than other commonly used measures, e.g., l_2 distance or RBF kernel. HIK is shown to have higher accuracies in SVM classification [6, 7], and in the clustering of *histograms* [7].

Recently HIK becomes even more attractive for its fast evaluation speed [6, 7]. It is shown that

$$\sum_{i=1}^n c_i \kappa_{\text{HI}}(\mathbf{q}, \mathbf{x}_i) \quad (1)$$

can be computed in $O(d)$ steps, where $\{\mathbf{x}_i\}_{i=1}^n$ are a set of n d -dimensional histograms, \mathbf{q} is a query histogram, and $\kappa_{\text{HI}}(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^d \min(x_j, y_j)$ is the

37 histogram intersection kernel. Although the effectiveness of HIK and other mea- 37
 38 sures for comparing histogram (e.g., the χ^2 distance) have not been extensively 38
 39 compared, the fast computing of Eqn. 1 makes HIK particularly attractive. 39

40 In this paper we make three contributions related to HIK SVM learning: 40

- 41 1. We show that HIK is a positive definite (PD) kernel for non-negative real- 41
 42 valued histograms. HIK is known to be a valid kernel for non-negative integer 42
 43 histograms [8]. We give a proof for non-negative real valued histograms. 43
 44 Our proof also completes the missing part of [9], which proved that HIK is 44
 45 conditionally positive definite (CPD). 45
- 46 2. We propose ICD, a fast dual HIK SVM training algorithm. ICD solves the 46
 47 SVM problem without re-encoding the input (which is a necessary step 47
 48 in [9]). It explicitly finds the feature space decision boundary, while the 48
 49 computations are carried out in the input space efficiently. ICD is a deter- 49
 50 ministic algorithm and do not need to choose a step size for optimization. 50
 51 We empirically show that ICD not only converges faster than the methods 51
 52 of [9, 10], but also yields higher accuracies. 52
- 53 3. We show that ICD is robust to the SVM parameter C *in practice*. Choosing 53
 54 SVM parameters by cross validation is very time consuming, but crucial for 54
 55 linear and RBF kernels, or the methods in [9, 10]. We empirically show that 55
 56 SVM parameters have only slight effects in ICD thus parameter selection is 56
 57 not necessary. 57

58 2 Related Work 58

59 The histogram intersection kernel (HIK) is originally proposed by Swain and 59
 60 Ballard for color-based object recognition [5]. It is further shown to be a positive 60
 61 definite kernel when the histograms only contain non-negative integers [8], which 61
 62 makes HIK suitable for SVM classification. HIK is shown to be a conditionally 62
 63 positive definite (CPD) kernel in real-valued cases [9]. We will give a proof that 63
 64 HIK on non-negative real-valued vectors is a positive definite kernel in Sec. 3.1. 64

65 HIK has shown to be a suitable similarity measure for comparing two his- 65
 66 tograms in different machine learning tasks. For example, it achieved higher 66
 67 accuracies in SVM classification than linear or RBF kernel [6, 7] in different 67
 68 domains, including object recognition, object detection, place recognition, and 68
 69 scene recognition (whose feature vectors are histograms). In unsupervised learn- 69
 70 ing tasks, kernel k-means clustering using HIK was also shown to produce better 70
 71 visual codebooks (and consequently achieved consistently higher accuracies in 71
 72 the resulting bag of visual words model) than the normal k-means algorithm [7]. 72

73 A naive method to compute Eqn. 1 will take $O(nd)$ steps, which is very 73
 74 expensive when either n or d is large. However, Eqn. 1 (with different assignment 74
 75 of the weights c_i) is crucial in the training and testing of HIK SVM classifiers, 75
 76 and in HIK based clustering. Recently, [6] showed that Eqn. 1 can be computed 76
 77 in $O(d \log n)$ steps (and $O(d)$ steps if an approximation is allowed with only slight 77
 78 loss of accuracy). Furthermore, [7] showed that by first quantizing the vectors 78
 79 to integers, exact answer can be obtained in $O(d)$ steps with less overhead than 79

80 the method in [6]. Fast computation of Eqn. 1 enables the testing of HIK SVM 80
 81 classifiers to have the same complexity as that of linear SVM [6, 7], and make 81
 82 HIK clustering almost as fast as the usual k-means clustering [7]. In Sec. 3.2 we 82
 83 will show that the method in [7] is not only a way to accelerate computation, 83
 84 but has a physical interpretation. 84

85 These computational methods are also applied in fast training of HIK SVMs, 85
 86 in which Eqn. 1 is again the speed bottleneck. Stochastic gradient descent (SGD) 86
 87 methods are used in [10, 9] to train an HIK SVM. More than 10 fold acceleration 87
 88 can be achieved by using the fast method to compute Eqn. 1. PWLSGD [9] is 88
 89 based on the stochastic method Pegasos (Primal Estimated sub-GrAdient SOLver 89
 90 for SVM) [11], and SIKMA [10] is another SGD method. One drawback of these 90
 91 methods is that it is subtle to choose a step size in the gradient descent update, 91
 92 which is important to the success of SGD methods. Also, SGD methods give 92
 93 different results in different runs on the same dataset. 93

94 3 The Histogram Intersection Kernel 94

95 3.1 HIK in \mathbb{R}_+ is a positive definite kernel 95

96 Let \mathbb{R}_+ be the set of non-negative real numbers $\{x \geq 0 | x \in \mathbb{R}\}$. We will prove 96
 97 that the histogram intersection kernel $\kappa_{\text{HI}}(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^d \min(x_{1,j}, x_{2,j})$ is a 97
 98 valid positive definite kernel for $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}_+^d$. 98

99 We use n to denote the number of data points and d for the dimension, and 99
 100 $x_{i,j}$ as the j -th component of a vector \mathbf{x}_i . We will always use i to index a training 100
 101 example, and use j to index a feature dimension. 101

102 We first prove this fact for $d = 1$. Given n real numbers $x_1, \dots, x_n \in \mathbb{R}_+$, we 102
 103 assume that $x_i \leq x_{i'}$ whenever $1 \leq i < i' \leq n$ without the loss of generality. 103
 104 Thus the kernel matrix K of this set has the property that 104

$$K_{ii'} = \min(x_i, x_{i'}) = x_{\min(i,i')}. \quad (2)$$

105 It is easy to verify that $\Lambda = R^T K R$, where R and Λ are defined as: 105

$$R_{ij} = \begin{cases} 1 & \text{if } i = j \\ -1 & \text{if } i = j - 1, \\ 0 & \text{otherwise} \end{cases}, \quad \text{and} \quad \Lambda_{ij} = \begin{cases} x_1 & \text{if } i = j = 1 \\ x_i - x_{i-1} & \text{if } i = j > 1. \\ 0 & \text{otherwise} \end{cases}. \quad (3)$$

106 The diagonal matrix Λ is positive semidefinite, so is K . Thus κ_{HI} is a positive 106
 107 definite kernel when $d = 1$. The generalization to $d > 1$ is straight forward, 107
 108 because the sum of Mercer kernels is again a Mercer kernel [12]. 108

109 HIK is proved to be conditionally positive definite (CPD) in \mathbb{R} [9], i.e., 109
 110 $\mathbf{x}^T K \mathbf{x} \geq 0$ when $\sum_j x_j = 0$. However, the proof in [9] is incomplete. The fi- 110
 111 nal step of the proof of [9] used the fact that HIK is a positive definite kernel in 111
 112 \mathbb{R}_+ , which we have just proved. CPD kernels can be safely used in an SVM if 112
 113 the bias term is included, but may have problem if we do not use the bias term 113
 114 (e.g., the SVM in Eqn 11 does not include the bias term.) 114

115 One important note is that “HIK is p.d. in \mathbb{R}_+ ” can be proved by setting 115
 116 $\beta = 1$ in Proposition 3 of [13]. Our method, though, provides a new intuitive 116
 117 proof that reveals interesting structures of HIK. It is also worth mentioning that 117
 118 Proposition 3 in [13] can also be easily proved using our technique. 118

119 3.2 Equation 1 and its feature space interpretation 119

120 There is a more intuitive way to illustrate that HIK is a Mercer kernel when 120
 121 the histograms only contain non-negative integers [8]. Given a d dimensional 121
 122 histogram \mathbf{x} , whose elements are all smaller than or equal to an upper bound \bar{v} . 122
 123 We define a mapping $B: \mathbb{N} \rightarrow \mathbb{R}^{\bar{v}}$ as (i.e., the unary representation) 123

$$B(x) = [\underbrace{1, 1, \dots, 1}_{x \text{ times}}, \underbrace{0, 0, \dots, 0}_{\bar{v}-x \text{ times}}], \quad (4)$$

124 Then the feature space spanned by κ_{HI} is $d\bar{v}$ dimensional, and a vector $\mathbf{x} \in \mathbb{N}^d$ 124
 125 is mapped to $B(\mathbf{x}) = [B(x_1), \dots, B(x_d)] \in \mathbb{R}^{d\bar{v}}$. 125

126 This fact is easy to prove because $xy = \min(x, y)$ when $x, y \in \{0, 1\}$. Thus 126
 127 $\kappa_{\text{HI}}(\mathbf{x}, \mathbf{y}) = B(\mathbf{x})^T B(\mathbf{y})$. Using the feature space for integer histograms, we can 127
 128 give a clear interpretation of the method presented in [7] for computing Eqn. 1. 128

129 Given a dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ in which we assume that the elements of \mathbf{x}_i 129
 130 are non-negative integers not larger than \bar{v} , and $y_i \in \{-1, +1\}$. An HIK SVM 130
 131 classifier will be 131

$$f(\mathbf{q}) = \sum_{i=1}^n \alpha_i y_i \kappa_{\text{HI}}(\mathbf{q}, \mathbf{x}_i) - \theta \quad (5)$$

132 for a test example \mathbf{q} , in which α_i are the Lagrange multipliers. Note that if we 132
 133 set $c_i = \alpha_i y_i$, this equation is a special case of Eqn. 1. 133

134 We define a matrix $T \in \mathbb{R}^{d\bar{v}}$ as (in which $c_i = \alpha_i y_i$) $T_{j,k} = \sum_{i:k \geq x_{i,j}} c_i x_{i,j} +$ 134
 135 $k \sum_{i:k < x_{i,j}} c_i$. Then it is shown in [7] that 135

$$f(\mathbf{q}) = \sum_{j=1}^d T_{j,q_j} - \theta. \quad (6)$$

136 Now consider the dataset $\{(B(\mathbf{x}_i), y_i)\}_{i=1}^n$, i.e., we study the same problem in 136
 137 the feature space instead. Let us assume that a linear SVM in the feature space 137
 138 results in the solution vector $\mathbf{w} = [\mathbf{w}_1, \dots, \mathbf{w}_d] \in \mathbb{R}^{d\bar{v}}$, where $\mathbf{w}_i \in \mathbb{R}^{\bar{v}}$ is the 138
 139 weights corresponding to $B(\mathbf{x}_i)$. Then we must have 139

$$f(\mathbf{q}) = \mathbf{w}^T B(\mathbf{q}) - \theta = \sum_{j=1}^d \mathbf{w}_j^T B(q_j) - \theta. \quad (7)$$

140 Comparing Eqn. 7 and 6, we get that 140

$$\mathbf{w}_j^T B(q_j) = T_{j,q_j} \quad \forall j \in \{1, \dots, d\}, q_j \in \{0, 1, \dots, \bar{v}\}. \quad (8)$$

141 In other words, we have: for all $j \in \{1, \dots, d\}$ and $k \in \{0, 1, \dots, \bar{v}\}$ 141

$$T_{j,k} = \sum_{t=1}^k w_{j,t}. \quad (9)$$

142 In short, we just revealed that there is a bijection between the table T and 142
 143 the decision boundary \mathbf{w} in the feature space. The key benefit of using the table 143
 144 T is that we do not need to explicitly store $\{B(\mathbf{x})\}_{i=1}^n$. Also, Eqn. 6 is very 144
 145 efficient ($O(d)$). 145

146 3.3 ICD: Intersection Coordinate Descent 146

147 This intuition can be used in fast training of HIK SVM classifiers. After quan- 147
 148 tizing the dataset to integers (with maximum feature value \bar{v}), we will solve a 148
 149 linear SVM problem in the feature space $\mathbb{R}^{d\bar{v}}$. However, instead of creating and 149
 150 storing $B(\mathbf{x}_i) \in \mathbb{R}^{d\bar{v}}, i = 1, \dots, n$, we will make use of data structures like the 150
 151 table T to carry out computations in the input space \mathbb{N}^d . We do not need to 151
 152 re-encode the input data as the PWLSGD method in [9]. 152

153 Our method is based on the SVM solver in LIBLINEAR [14], which uses 153
 154 a dual coordinate descent method. Given a dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $y_i \in$ 154
 155 $\{-1, +1\}$, its corresponding dataset in the feature space is $\{(B(\mathbf{x}_i), y_i)\}_{i=1}^n$. A 155
 156 linear SVM in the feature space solves the following problem: 156

$$\min_{\mathbf{w}} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^n \xi(\mathbf{w}; B(\mathbf{x}_i), y_i), \quad (10)$$

157 where $\xi(\mathbf{w}; B(\mathbf{x}_i), y_i) = \max(1 - y_i \mathbf{w}^T B(\mathbf{x}_i), 0)^2$ is the L2-loss function. The 157
 158 parameter C controls a trade-off between maximum margin and empirical errors 158
 159 on the training set. 159

The primal problem (Eqn. 10) is equivalent to the following dual form

$$\begin{aligned} \min_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}) &= \frac{1}{2} \boldsymbol{\alpha}^T \bar{Q} \boldsymbol{\alpha} - \mathbf{e}^T \boldsymbol{\alpha} \\ \text{subject to } &0 \leq \alpha_i \leq U, \forall i, \end{aligned} \quad (11)$$

160 where $U = \infty$, $\bar{Q} = Q + D$, $Q_{ii'} = y_i y_{i'} B(\mathbf{x}_i)^T B(\mathbf{x}_{i'})$, D is a diagonal matrix 160
 161 and $D_{ii} = 1/(2C)$ in an L2-loss SVM. Note that $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\mathbf{w} = \sum_i \alpha_i y_i B(\mathbf{x}_i)$. 161

162 In [14] the dual problem is solved using coordinate descent. The values of 162
 163 α_i are updated sequentially for $i = 1, 2, \dots, n$. When updating α_i , a new α'_i is 163
 164 chosen such that it will reduce $g(\boldsymbol{\alpha})$ by the largest amount, while still in the range 164
 165 $[0 U]$. The discriminant function \mathbf{w} is then incremented by $(\alpha'_i - \alpha_i) y_i B(\mathbf{x}_i)$, i.e., 165
 166 updated using the i -th data point $B(\mathbf{x}_i)$. A hypothetical algorithm to solve SVM 166
 167 in the feature space is shown in Algorithm 1. 167

168 However, we will not use this algorithm in practice. The main difficulty of ap- 168
 169 plying Algorithm 1 is to compute line 1, 4, and 9 without explicitly constructing 169
 170 the high dimensional vectors \mathbf{w} and $B(\mathbf{x}_i)$. Instead we will use the table T . 170

171 We do not need to compute line 1 because there is a bijection between \mathbf{w} 171
 172 and T . A proper initialization of T will replace line 1. We simply initialize all 172
 173 elements of T to 0, which is equivalent to initialize \mathbf{w} to 0. 173

174 Similarly, Eqn. 6 can be used to efficiently compute $\mathbf{w}^T B(\mathbf{x}_i)$ in $O(d)$ steps, 174
 175 which makes line 4 easy to compute. The remaining difficulty is then how to 175
 176 update \mathbf{w} , or equivalently, how to update T because we do not store \mathbf{w} . 176

Algorithm 1 A hypothetical algorithm for HIK SVM in the feature space

```

1: Given  $\alpha$  and correspondingly  $\mathbf{w} = \sum_i \alpha_i y_i B(\mathbf{x}_i)$ 
2: while  $\alpha$  is not optimal do
3:   for  $i = 1, \dots, n$  do
4:      $G = y_i \mathbf{w}^T B(\mathbf{x}_i) - 1 + D_{ii} \alpha_i$ 
5:      $PG = \begin{cases} \min(G, 0) & \text{if } \alpha_i = 0 \\ \max(G, 0) & \text{if } \alpha_i = U \\ G & \text{if } 0 < \alpha_i < U \end{cases}$ 
6:     if  $|PG| \neq 0$  then
7:        $\bar{\alpha}_i \leftarrow \alpha_i$ 
8:        $\alpha_i \leftarrow \min(\max(\alpha_i - G/\bar{Q}_{ii}, 0), U)$ 
9:        $\mathbf{w} \leftarrow \mathbf{w} + (\alpha_i - \bar{\alpha}_i) y_i B(\mathbf{x}_i)$ 
10:    end if
11:  end for
12: end while

```

Algorithm 2 ICD: A method for training HIK SVM

{Replace the following lines in Algorithm 1, the remaining lines of Algorithm 1 will be omitted here. }

line 1' : $T_{j,k} \leftarrow 0$, for all $j \in \{1, \dots, d\}, k \in \{1, \dots, \bar{v}\}$

{Note that the below commands update the table T using \mathbf{x}_i }

line 4' : $G = y_i \sum_{j=1}^d T_{j,x_{i,j}} - 1 + D_{ii} \alpha_i$

line 9' : $T_{j,k} \leftarrow T_{j,k} + (\alpha_i - \bar{\alpha}_i) y_i \min(x_{i,j}, k), \forall j \in \{1, \dots, d\}, k \in \{1, \dots, \bar{v}\}$

177 Let us denote $(\alpha_i - \bar{\alpha}_i) y_i$ as δ_{α_i} . Then the change of \mathbf{w} (line 9) is now 177
 178 $\Delta \mathbf{w} = \delta_{\alpha_i} B(\mathbf{x}_i)$, or equivalently $(B(x_{i,j})_t$ is the t -th element of $B(x_{i,j})$), 178

$$\Delta w_{j,t} = \delta_{\alpha_i} B(x_{i,j})_t, \text{ for all } 1 \leq j \leq d, 1 \leq t \leq \bar{v}.$$

Using Eqn. 9, it is easy to find that

$$\begin{aligned} \Delta T_{j,k} &= \sum_{t=1}^k \Delta w_{j,t} \\ &= \sum_{t=1}^k \delta_{\alpha_i} B(x_{i,j})_t = \delta_{\alpha_i} \sum_{t=1}^k B(x_{i,j})_t \\ &= \delta_{\alpha_i} \min(x_{i,j}, k). \end{aligned} \tag{12}$$

179 The last equality in Eqn. 12 follows from the identity $\sum_{t=1}^k B(x)_{t} = \min(x, k)$. 179

180 In summary, solving an HIK SVM optimization can be done using the dual 180
 181 coordinate descent approach. The computations are carried out on the original 181
 182 histograms instead of in the high dimensional feature space, which maintains the 182
 183 fast training speed. The HIK SVM training algorithm is shown in Algorithm 2, 183
 184 in which we assume that $T_{j,0} = 0$ for any $j \in \{1, \dots, d\}$. We will refer to 184
 185 Algorithm 2 as the ICD (Intersection Coordinate Descent) method. 185

186 After an SVM is trained using ICD, a new example \mathbf{q} can be classified in 186
 187 $O(d)$ steps using Eqn. 6, which is the same complexity as that of a linear SVM. 187

188 3.4 L1-loss, multi-class, convergence, and all that 188

189 The ICD algorithm is provided at <http://www.ntu.edu.sg/home/jxwu> inside 189
 190 the libHIK package. Beyond Algorithm 2, fast HIK SVM training method 190
 191 using L1-loss, in primal space, and for multi-class datasets are also provided. 191
 192 Due to the space limit, we will only briefly discuss some important issues. 192

193 ICD can use L1-loss function $\xi(\mathbf{w}; \mathbf{x}_i, y_i) = \max(1 - y_i \mathbf{w}^T \mathbf{x}_i, 0)$ [14], by 193
 194 setting $U = C$ and $D_{ii} = 0$ in Eqn. 11. 194

195 We can accelerate the solving of the primal problem (Eqn. 10) using the same 195
 196 idea of ICD. We maintain both \mathbf{w} and the table T during training. There is no 196
 197 need to explicitly create $B(\mathbf{x}_i)$. Primal method is preferred when $d \gg n$. 197

198 We can solve multi-class problems using the one versus rest method. The 198
 199 Crammer-Singer formulation [15] can also be greatly accelerated by implicit 199
 200 feature space computations using Eqn. 6. 200

201 The global convergence Theorem 1 of [14] readily applies to ICD. Thus the 201
 202 ICD method obtains an ϵ -accurate solution in $O(\log(1/\epsilon))$ iterations. 202

203 In ICD there are $d\bar{v}$ numbers to change when updating each α_i ,¹ while in 203
 204 linear SVM we only need to update d numbers. However, in practice ICD requires 204
 205 a much smaller number of iterations to converge than that of linear SVM. On 205
 206 many real world datasets, ICD converges within 30 iterations, while linear SVM 206
 207 is not converged after 1000 iterations. Thus the training time of ICD is much 207
 208 faster than \bar{v} times of the linear SVM training time. On some difficult datasets 208
 209 ICD is even faster than LIBLINEAR (c.f. Sec. 4). 209

210 3.5 Quantization, default bin number, and default C parameter 210

211 When the feature vectors are not natural integer histograms, we use a simple 211
 212 method to quantize it so that we can apply ICD. Given a dataset, we find v_{\min} , 212
 213 the minimum feature value in the training set. We also find v_{\max} , which is the 213
 214 97.5-th percentile of all training feature values.² A feature value v is mapped 214
 215 (quantized) to an integer in $[0 \bar{v}]$ as follows 215

$$v \rightarrow (\text{int}) (\bar{v} \times (v - v_{\min}) / (v_{\max} - v_{\min})). \quad (13)$$

216 With this simple quantization strategy, we can apply ICD to a much broader 216
 217 range of problems (e.g., whose feature vectors are not natural histograms and 217
 218 have negative feature values). 218

219 Choosing \bar{v} is a very important decision. The number of quantization bins, 219
 220 \bar{v} , not only affects the training time and storage requirements. It is also directly 220

¹ In practice we do not need to update all these $d\bar{v}$ numbers. If \bar{v}_j is the maximum feature value in dimension j , then we only need to update $T_{j,k}$, $k = 1, \dots, \bar{v}_j$ for the j -th dimension. We usually observe that $\bar{v}_j \ll \bar{v}$.

² We do not use the maximum feature value as v_{\max} because in computer vision it may have an artificial mode at the largest feature value. For example, in the densely sampled bag of visual words model, possibly more than half of the image patches will be mapped to the visual word that corresponds to a uniform image region.

related to the accuracy of trained classifiers. Obviously a small \bar{v} value will result in low accuracy. However, it is adverse to have a large \bar{v} . A large \bar{v} will eventually cause over-fitting and uses more memory and CPU cycles.

We experimented with $\bar{v} = 50, 100, \text{ and } 200$. In our experiments different problems acquired best accuracies at different \bar{v} values. However, $\bar{v} = 100$ achieved a fair balance between the memory/computation cost and classification accuracy across almost all the datasets. We use $\bar{v} = 100$ if quantization is needed, and if we do not explicitly specify \bar{v} otherwise.

The default value for the C parameter in LIBLINEAR is 1, and feature vectors are usually normalized to the range $[-1, 1]$. In ICD, κ_{HI} usually generates much large kernel values. Consequently, we choose $C = 10^{-3}$ as the default value for ICD.

4 Experimental results

We conducted 4 sets of experiments to test various aspects of the ICD algorithm. First we compare ICD with two recently proposed fast HIK SVM training algorithm (Sec. 4.1). We then test ICD on a large scale pedestrian detection dataset (Sec. 4.2). The third set of experiments deal with three different object and scene recognition problems in computer vision (Sec. 4.3). Finally, we show that when cross-validation based SVM parameter selection is infeasible for huge datasets, ICD achieves both faster speed and higher accuracies comparing with linear and RBF kernels, using their default parameter settings (Sec. 4.4). Our empirical results show that ICD is very robust to the C parameter in practice.

Before applying ICD, we use Eqn. 13 to quantize a problem if necessary. We set $\bar{v} = 100$ if not otherwise specified. The one versus rest strategy is used for multi-class problems. We use the default value $C = 10^{-3}$ whenever ICD is used.

4.1 Comparing with PWLSGD and SIKMA

PWLSGD [9] and SIKMA [10] are two recent stochastic gradient descent (SGD) methods for fast training of HIK SVM. In this section we compare ICD with these two, using the software and datasets provided with [9] and [10].

Note that in PWLSGD and SIKMA, we use the SVM parameters that are carefully chosen using cross validation by their corresponding authors. While in ICD we simply use the default value $C = 10^{-3}$. PWLSGD provides sample data on Caltech 101 [16]. We report in Table 1(a) the results when 15 training and testing examples are used in each category. The SIKMA software provides sample data on the PASCAL VOC 2007 images [17]. The comparison results are reported in Table 1(b). SGD methods yield different results on the same dataset in multiple runs. Since SIKMA does not fix the seed of its random number generator, we report its average result in 5 runs. Following the setup of SIKMA, we set $\bar{v} = 50$ for this problem.

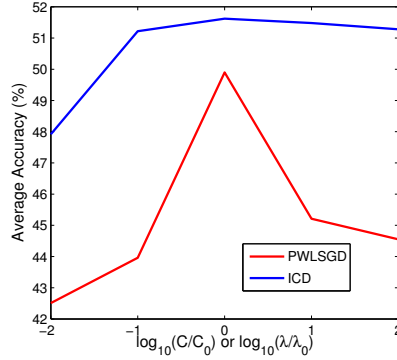
For every method, we show the training time (in seconds) followed by the classification accuracy. As shown in Table 1(a) and 1(b), ICD not only reduces

Table 1. Comparing training time and accuracy of HIK SVM methods.

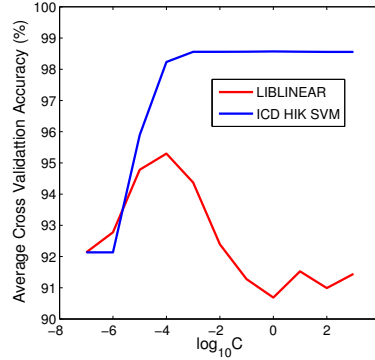
ICD		PWLSGD		LIBLINEAR		ICD		SIKMA		LIBLINEAR	
56.5	51.62%	188.2	49.90%	43.3	48.05%	9.2	97.21%	13.2	96.97±.19%	1.9	96.00%

(a) Comparing ICD with PWLSGD.

(b) Comparing ICD with SIKMA.



(a) Result on Caltech 101



(b) Result on INRIA

Fig. 1. Effect of different SVM parameters.

262 training time by a large percentage, it also has higher classification accuracies, 262
 263 despite the fact that we do not tune the C parameter in ICD. An additional 263
 264 comparison with LIBLINEAR [14] is provided (with the default settings of LI- 264
 265 BLINEAR). The proposed method enjoys higher accuracy with a reasonable 265
 266 amount of increase in training time. For example, on the Caltech 101 dataset, 266
 267 ICD only uses 30% more training time than LIBLINEAR (Table 1(a)). 267

268 One attractive property of ICD is that empirically it is not sensitive to SVM 268
 269 parameters. Let $\lambda_0 (= 0.0015)$ and $C_0 (= 0.001)$ denote the SVM parameters used 269
 270 in Table 1 for PWLSGD and ICD respectively. In Fig. 1a we show Caltech 101 270
 271 results of different C and λ values where $\log_{10}(C/C_0)$ or $\log_{10}(\lambda/\lambda_0)$ ranges from 271
 272 -2 to 2 . ICD has high accuracy at the default value C_0 , and its accuracy is stable 272
 273 with larger C values. However, large variations are observed for PWLSGD. Time 273
 274 consuming cross-validation based parameter selection is needed for PWLSGD to 274
 275 choose an appropriate λ , but in ICD it is not necessary to choose C . ICD has 275
 276 stable accuracies when $C \geq 10^{-4}$. We observe in Sec. 4.2 again that ICD is 276
 277 robust to C . 277

278 4.2 Pedestrian detection 278

279 Next we use the INRIA pedestrian dataset [2] to evaluate ICD in a large scale 279
 280 vision problem. We use the 256 dimensional CENTRIST [3] visual descriptor 280

Table 2. Results on INRIA pedestrian.

	Time	Accuracy	Iterations
ICD	160 s	98.56%	21.4
LIBLINEAR	681 s	90.69%	1000

281 as our base feature descriptor. A 108×36 image patch is divided into 9×4 281
 282 blocks. Any neighboring 2×2 blocks are formed into a super-block. The con- 282
 283 catenation of CENTRIST in all super-blocks generates a feature vector that has 283
 284 6,144 dimensions. This feature vector is a natural histogram with $\bar{v} = 352$. We 284
 285 evaluate the SVM training algorithms in a “hard” dataset that is the result of 285
 286 bootstrapping the INRIA dataset (using the procedures in [2]). There are 30,711 286
 287 examples. This dataset is mostly dense, resulting in a large scale problem with 287
 288 approximately 82 million non-zero feature values. 288

289 We compare ICD with LIBLINEAR. Five-fold cross validation is applied. The 289
 290 total training time, average accuracy, and average number of iterations needed 290
 291 to finish the optimization are reported in Table 2. Default C values are used in 291
 292 both methods in Table 2. 292

293 One interesting observation from Table 2 is that ICD only takes about 24% 293
 294 of the training time of LIBLINEAR. This is related to the number of iterations 294
 295 which is required to terminate the SVM optimization. HIK SVM has higher 295
 296 discrimination capability than a linear SVM, and ICD usually requires a small 296
 297 number of iterations to converge in practice.³ In summary, ICD scales well to 297
 298 large problems, and is particularly suitable when the feature vectors are natural 298
 299 histograms. 299

300 The effect of SVM parameters are studies in Fig 1b, where the C value ranges 300
 301 from 10^{-7} to 10^3 , with step size 10. Linear SVM is sensitive to C , and an overly 301
 302 large C will lead to a lower accuracy. In this dataset HIK SVM is not sensitive 302
 303 to C : the accuracy increases with C and a large C value will not lead to a lower 303
 304 accuracy. The same phenomenon is observed in almost all datasets we tested in 304
 305 this paper. 305

306 It is also interesting to compare with general purpose SVM learners with 306
 307 linear, RBF, or the histogram intersection kernel. However, on this large scale 307
 308 dataset, general purpose SVM solvers (e.g., LIBSVM [18]) requires more than 10 308
 309 hours to converge. This fact makes these solvers impractical for large problems. 309

310 4.3 Object and scene recognition 310

311 In this section we evaluate ICD in 3 more benchmark vision problems: Caltech 311
 312 101 [16], 15 class scene recognition [19], and 8 class sport events [20]. Images 312
 313 are represented using the bag of visual words model, and feature vectors are 313

³ LIBLINEAR terminates when the iteration number is 1000. So the actual iterations needed for convergence is higher than 1000 in this problem.

Table 3. Results on various vision problems.

	caltech			scene			sports		
	Time	Acc	Acc(cv)	Time	Acc	Acc(cv)	Time	Acc	Acc(cv)
ICD	71.5	60.0%	59.9%	20.1	81.9%	82.0%	10.4	81.3%	81.5%
LIBSVM+HIK	66.8	54.6%	54.9%	40.1	81.6%	81.7%	10.7	81.0%	81.0%
LIBLINEAR	6.3	54.1%	54.2%	1.4	75.3%	75.5%	0.5	76.7%	78.5%
LIBSVM+LIN	65.2	51.3%	51.4%	33.5	76.8%	76.8%	8.5	78.8%	78.8%
LIBSVM+RBF	67.3	18.2%	53.4%	58.2	35.2%	79.6%	14.1	12.5%	76.5%

generated by libHIK [7] with k-means visual codebooks. The results from the first train/test split of libHIK are reported.⁴

Three kernel types (linear, RBF, and HIK) are compared. The features are quantized for ICD and LIBSVM+HIK, because we want these two methods to use exactly the same data. In Table 3, the first two columns for each dataset report the training time and accuracy of a method when we use the default SVM parameters. We also use training set cross validation to choose SVM parameters in the range $\log_{10} C \in [-5, 3]$, $\log_{10} \gamma \in [-5, -1]$, whose accuracies are reported in the ‘Acc(cv)’ column.

In terms of classification accuracy, HIK has clear advantages over linear and RBF kernel types. ICD achieves slightly higher accuracies than the general purpose LIBSVM solver with HIK. The Caltech 101 dataset shows an exception where ICD has a large 5.4% advantage over LIBSVM. This might be due to the fact that there are 101 classes in this problem, while the one versus one strategy of LIBSVM is not suitable for handling large number of classes.

In terms of training time, LIBLINEAR trains much faster than other methods, while all other methods have comparable training time. It is worth noting that the feature vectors are $d = 6,200$ dimensional in these problems, while the training set size n ranges from 560 to 1515. Dual space algorithms (including the proposed method) is not effective while $d \gg n$. However, ICD still has approximately the same training speed as LIBSVM.

Again we observed the phenomenon that ICD is not sensitive to SVM parameters, since ‘Acc(cv)’ only has slight advantage over ‘Acc’ (the accuracy using default ICD SVM parameters). The robustness to SVM parameters is very attractive because cross validation parameter selection is infeasible for huge datasets, e.g., accuracy of the RBF kernel is heavily affected by SVM and kernel parameters.

4.4 Working with non-histograms and default SVM parameters

Nowadays many problems are too large to perform cross validation for SVM parameter selection. Thus it is important to emphasize *high accuracies using the default SVM parameters*. In the final set of experiments, we will evaluate ICD on problems that are not natural histograms and on huge datasets. The four

⁴ Note that the Caltech 101 features are different than those used in Sec. 4.1.

Table 4. Properties of the non-histogram datasets.

	train size	test size	dimension	#class
ijcnn1	4,990	91,701	22	2
shuttle	43,500	14,500	9	7
acoustic	78,823	19,705	100	3
rcv1	677,399	20,242	47,236	2

Table 5. Comparing performances using default SVM parameters.

	ijcnn1		shuttle		acoustic		rcv1	
	Time	Accuracy	Time	Accuracy	Time	Accuracy	Time	Accuracy
ICD	0.6	94.82%	0.7	99.50%	137.2	82.98%	34.3	97.95%
LIBSVM+HIK	42.9	95.27%	3.7	99.57%	1362.0	83.12%	> 50,000	
LIBLINEAR	0.4	91.79%	0.8	92.35%	14.6	80.18%	6.4	97.96%
LIBSVM+LIN	48.4	92.12%	7.4	97.10%	1736.0	80.46%	> 50,000	
LIBSVM+RBF	60.1	92.79%	14.3	97.23%	1824.0	79.69%	> 50,000	

346 problems we experimented with are chosen from the LIBSVM dataset collection: 346
 347 ijcnn1, shuttle, acoustic (combined), and rcv1 (binary). We choose these datasets 347
 348 whose features are not histograms, and whose sizes range from medium to huge. 348
 349 The particulars of these problems are collected in Table 4. We switched the 349
 350 training and testing set of the rcv1 problem so that we have a huge training set 350
 351 to test the scalability of ICD. 351

352 We compare with LIBLINEAR and LIBSVM using linear, RBF, and HIK. 352
 353 We use the default value $C = 1$ for LIBLINEAR and LIBSVM on the original 353
 354 feature vectors, and use $C = 10^{-3}$ on quantized versions. Experimental results 354
 355 are reported in Table 5. 355

356 One important observation is that on these datasets both HIK SVM classifiers 356
 357 (ICD and LIBSVM+HIK) achieve higher accuracies even when their feature 357
 358 vectors are not natural histograms. We also compare the two pairs of methods 358
 359 that use the same kernel. Although the LIBSVM+LIN solver sometimes have 359
 360 noticeable advantage over LIBLINEAR at the cost of much longer training time 360
 361 (e.g., in the shuttle problem), the proposed ICD method has almost the same 361
 362 accuracy as LIBSVM+HIK. 362

363 ICD, however, trains a lot faster than LIBSVM+HIK, and its speedup is 363
 364 related to size of the datasets. ICD is about 5 times faster than LIBSVM+HIK 364
 365 on the shuttle dataset. However, in the rcv1 dataset, the speedup is more than 365
 366 3 orders of magnitude. For all three kernel types, the LIBSVM solver did not 366
 367 converge after 50,000 seconds when running the rcv1 problem. Thus LIBSVM's 367
 368 accuracies on this dataset is not available in Table 5. 368

369 LIBLINEAR is faster than ICD. However, the speedup is usually smaller 369
 370 than 10. Given the fact that the training time is smaller than 1 minute even 370
 371 in the rcv1 dataset, we believe that the proposed algorithm is preferable for its 371
 372 higher classification accuracies. 372

373 It is generally accepted that for problems with a large number of feature 373
 374 dimensions, linear SVM usually works as well as other more complex kernel 374
 375 types. Thus it is not surprising to observe that on the `rcv1` dataset, ICD requires 375
 376 more training time than LIBLINEAR, while both methods have approximately 376
 377 the same classification accuracy. Our experiments on `rcv1`, though, further illus- 377
 378 trate the scalability of ICD. In computer vision, we usually work with a medium 378
 379 dimensional feature vector (e.g., around 5000, smaller than that of `rcv1`). The 379
 380 experiments on `rcv1` illustrate that ICD is able to handle even millions of training 380
 381 examples in computer vision problems. 381

382 5 Conclusions and future work 382

383 Our contributions are threefold. First, we prove that the histogram intersection 383
 384 kernel (HIK) is a positive definite kernel for non-negative real numbers. Second, 384
 385 we give the physical meaning of the computational method that accelerates the 385
 386 kernel evaluation of HIK. Based on this interpretation, we propose ICD, a fast, 386
 387 accurate, and scalable HIK SVM solver. Third, we empirically show that ICD is 387
 388 not sensitive to the C parameter in SVM, and achieve high accuracies using its 388
 389 default settings on huge datasets. 389

390 As a summary of the theoretical analyses and experimental results, we list 390
 391 the advantages and limitations of the proposed method (+ for advantages and 391
 392 - for limitations). 392

393 **Speed (+)** ICD trains much faster than general purpose SVM solvers. It also 393
 394 trains faster than two recent SGD based methods (PWLSGD and SIKMA). 394
 395 The testing speed has the same complexity as linear classifiers. 395

396 **Insensitivity to C (+)** Accuracy of ICD generally increases with the SVM 396
 397 parameter C . However, a large C does not lead to low accuracy. This empirical 397
 398 property is particularly attractive on huge datasets where cross validation 398
 399 parameter selection is infeasible. 399

400 **Scalability (+)** It scales easily to large problems, and is most efficient for prob- 400
 401 lems with a medium number of feature dimensions and a huge number of 401
 402 training examples. Many vision problems fit into this category. 402

403 **Accuracy (+)** It has comparable accuracies to general purpose SVM solvers, 403
 404 and the PWLSGD or SIKMA HIK SVM solver. 404

405 **Simplicity (+)** There is only 1 parameter in ICD (C), and the default value 405
 406 $C = 10^{-3}$ works well in most problems. It is also a deterministic algorithm, 406
 407 producing the same result on multiple runs. There is no need to re-encode 407
 408 input data. 408

409 **Storage (-)** The table T increases the storage cost, especially when d is large, 409
 410 and when the problem contains a large number of classes. 410

411 **Quantization (-)** The need for quantization introduces additional costs (al- 411
 412 though this cost is small), and the quantized version does not guarantee 412
 413 higher accuracy than linear SVM in a few cases (e.g., `rcv1`). 413

414 There are possible directions to address certain limitations of the proposed 414
 415 method. For example, we can make the table T sparse, which will answer the 415

416 storage problem at the cost of a reasonable increase in training and testing time. 416
 417 A rule of thumb can be developed to automatically choose between a linear SVM 417
 418 or HIK SVM. Finally, we can explore adaptive quantization methods to achieve 418
 419 better quantized feature vectors (and higher accuracies). 419

420 References 420

- 421 1. Lowe, D.: Distinctive image features from scale-invariant keypoints. *IJCV* **60** 421
 422 (2004) 91–110 422
- 423 2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 423
 424 CVPR. Volume 1. (2005) 886–893 424
- 425 3. Wu, J., Rehg, J.M.: CENTRIST: A visual descriptor for scene categorization. 425
 426 Technical Report GIT-GVU-09-05, GVU Center, Georgia Institute of Technology 426
 427 (2009) 427
- 428 4. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation 428
 429 invariant texture classification with local binary patterns. *IEEE TPAMI* **24** (2002) 429
 430 971–987 430
- 431 5. Swain, M.J., Ballard, D.H.: Color indexing. *IJCV* **7** (1991) 11–32 431
- 432 6. Maji, S., Berg, A.C., Malik, J.: Classification using intersection kernel support 432
 433 vector machines is efficient. In: CVPR. (2008) 433
- 434 7. Wu, J., Rehg, J.M.: Beyond the euclidean distance: Creating effective visual code- 434
 435 books using the histogram intersection kernel. In: ICCV. (2009) 435
- 436 8. Odone, F., Barla, A., Verri, A.: Building kernels from binary strings for image 436
 437 matching. *IEEE Trans. Image Processing* **14** (2005) 169–180 437
- 438 9. Maji, S., Berg, A.C.: Max-margin additive classifiers for detection. In: ICCV. 438
 439 (2009) 439
- 440 10. Wang, G., Hoiem, D., Forsyth, D.: Learning image similarity from flickr groups 440
 441 using stochastic intersection kernel machines. In: ICCV. (2009) 441
- 442 11. Shalev-Shwartz, S., Singer, Y., Srebro, N.: Pegasos: Primal estimated sub-gradient 442
 443 solver for svm. In: ICML. (2007) 807–817 443
- 444 12. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines 444
 445 and Other Kernel-based Learning Methods. Cambridge University Press (2000) 445
- 446 13. Boughorbel, S., Tarel, J.P., Boujemaa, N.: Generalized histogram intersection 446
 447 kernel for image recognition. In: ICIP. (2005) 447
- 448 14. Hsieh, C.J., Chang, K.W., Lin, C.J., Keerthi, S.S., Sundararajan, S.: A dual 448
 449 coordinate descent method for large-scale linear SVM. In: ICML. (2008) 408–415 449
- 450 15. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel- 450
 451 based vector machines. *JMLR* **2** (2001) 265–292 451
- 452 16. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few 452
 453 training example: an incremental bayesian approach tested on 101 object cate- 453
 454 gories. In: CVPR 2004, Workshop on Generative-Model Based Vision. (2004) 454
- 455 17. Everingham, M., Gool, L.V., Williams, C., Winn, J., Zisserman, A.: The PASCAL 455
 456 visual object classes challenge 2007 (VOC 2007) results. Technical report (2007) 456
- 457 18. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) 457
 458 Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>. 458
- 459 19. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid 459
 460 matching for recognizing natural scene categories. In: CVPR. Volume II. (2006) 460
 461 2169–2178 461
- 462 20. Li, L.J., Fei-Fei, L.: What, where and who? Classifying events by scene and object 462
 463 recognition. In: ICCV. (2007) 463