Adapting RANSAC SVM to Detect Outliers for Robust Classification

Subhabrata Debnath
http://www.cse.iitk.ac.in/~sdebnath
Anjan Banerjee
http://www.cse.iitk.ac.in/~anjan
Vinay Namboodiri
http://www.cse.iitk.ac.in/~vinaypn

Department of Computer Science and Engineering,
Indian Institute of Technology Kanpur
Kanpur
UP, India

Abstract

Most visual classification tasks assume the authenticity of the label information. However, due to several reasons such as difficulty of annotation or inadvertently due to human error, the annotation can often be noisy. This results in wrongly annotated examples. In this paper, we consider the examples that are wrongly annotated to be outliers. The task of learning a robust inlier model in the presence of outliers is typically done through the RANSAC algorithm[6].

We show that instead of adopting RANSAC to obtain the 'right' model, we could use many instances of randomly sampled subsets to build a lot of models. The collective decision of all these models can be used to identify examples that are likely to be outliers. This results in a modification to RANSAC SVM[11] to explicitly obtain probable outliers from the set of given examples. Once the outliers are detected, these examples are excluded from the training set. We also show that the method can be used to identify very hard examples present in the training data. In this case, where we believe that the examples are correctly annotated, we can achieve good generalization when such examples are excluded from the training set. The method is evaluated using the standard PASCAL VOC 2007 dataset[4]. We show that the method is particularly suited for identifying wrongly annotated examples resulting in improvement of more than 12% over the RANSAC-SVM approach. Hard examples in PASCAL VOC dataset are also identified by this method and this even results in a marginal improvement of the mean average precision over the base classifier provided with all clean examples.

1 Introduction

Recognition tasks in computer vision generally rely on the availability of explicit crowd-sourced annotations. However, very often the annotation task is ambiguous and hard [1]. Moreover, the annotator may not be aware of the correct category or may have bias while annotating. These aspects have been studied by Welinder *et al.* [13] where the authors model the profile of annotators to analyse whether a particular annotator is providing accurate labels. In our work we consider the problem where the provided dataset may be noisy for a visual classification task. We address the task of learning a classifier in the presence of noisy data by identifying the wrongly labelled examples as outliers and excluding them while training.

In computer vision the task of learning a model in the presence of noise has been traditionally solved using the classical RANSAC algorithm [7]. The RANSAC algorithm has also been adapted as RANSAC SVM by Nishida and Kurita [11], where the authors adapted RANSAC algorithm to obtain a small set of examples that could be used to scale the support vector machine (SVM) to large scale datasets. Our motivation for adapting RANSAC algorithm is inspired by its original motivation to obtain models that can work in the presence of noise.

Our method involves selection of random subsets of data and using these, we create individual support vector machine (SVM) classifiers. We then classify the whole training set using each of these classifiers. We do this repeatedly using a number of such randomly selected sub-sets. The misclassifications using these models provide us with a measure of outlierness of the samples. While, any single model cannot be used to effectively indicate whether a particular examples is an inlier or an outlier, a quantized score from many such models does provide us with a better estimate.

We use the PASCAL VOC 2007 dataset[4] for our experiments. This is because the dataset is a challenging real world dataset. Another reason for using this dataset is because the annotations are crowd-sourced, thus we can evaluate the dataset by flipping the annotation for certain samples (we choose to do so for the hardest examples). Further, the dataset also provides us with a set of ‘hard’ examples that are labeled 0. These examples enable us to evaluate the outlier identification technique to identify ‘hard’ examples. .

2 Related Work

Our work is inspired by the original RANSAC algorithm [7] that solved the problem of building a model in the presence of noise. The RANSAC algorithm has been widely used by the community and appears in most computer vision textbooks. There have been many variants of the original algorithm and a comparative analysis of the various RANSAC approaches has been done by Choi *et al.* [3]. In our approach we are more interested in the use of RANSAC for image classification. To the best of our knowledge its use for classification has been done primarily through the RANSAC-SVM [11].

As mentioned earlier, the RANSAC SVM method [11], is aimed at selecting a small set of examples to scale the problem of learning with large number of examples. We are motivated more by the original goal of RANSAC of being able to work with data in the presence of noise, which in our case is in the form of wrong or noisy labels. The RANSAC SVM method selects random subsets of the training data and trains small SVMs on them while tuning the hyper-parameters to fit the SVMs to the full training set. The authors show how RANSAC SVM achieves good generalization performance and also has computational advantages over the full SVM solution. However, they also incorporate a genetic algorithm in their approach to choose samples by using multiple generations of evolution. The RANSAC SVM is inherently randomized, however, the original RANSAC does converge in several iterations to an inlier set. The addition of a randomized evolution method results in there being no convergence on an inlier set as the evolution would depend on successive mutations and crossovers that are governed by the genetic algorithm. We compare our method with an implementation of RANSAC SVM that is closer to the original RANSAC algorithm adapted for image classification.

In literature, a number of implicit outlier detection techniques ([9], [10]) have also been explored which aim to perform learning with noisy labels. In this paper, we show how a simple modification to the classical RANSAC algorithm when used in the classification setting can result in very good performance. In the next section we describe our proposed method and highlight the modifications we made to the RANSAC SVM algorithm and explain the differences between our method and RANSAC SVM.

3 Method Description

Our method is based on the following assumptions : i) Outliers differ in their feature space as compared to the normal examples. ii) Misclassifications on smaller random subsets gives us an approximate measure of 'outlierness', indicating dissimilarity with other examples, iii) the number of outliers or wrongly labeled examples is less than 50% of the dataset, i.e. the number of inliers is greater than the number of outliers. We now present the proposed method in detail.

Algorithm 1 Outlier Robust RANSAC-SVM Adaptation

```
1: procedure OUTLIER ROBUST RANSAC-SVM(SetSize $k$, NumIteration $m$,Threshold $\tau$)
2:     for each instance $x_j \in$ Training Set $S$ do
3:         OutlierScore $O(x_j) \leftarrow 0$
4:     end for
5:     for i=1 to NumIteration $m$ do
6:         Choose a random $X_i \subset$ Training Set $S$ s.t. $|X_i|$=SetSize $k$
7:         $w_i \leftarrow BuildSVMModel(X_i)$ using equation 1.
8:         $MisclassifiedSet_i \leftarrow GetMisclassifiedInstances(w_i, S)$
9:         for each instance $x_j \in MisclassifiedSet_i$ do
10:            $IncrementByOne$(OutlierScore $O(x_j)$)
11:        end for
12:    end for
13:    OutlierSet $S_o \leftarrow \{\}$
14:    InlierSet $S_i \leftarrow \{\}$
15:    for each instance $x_j \in$ Training Set $S$ do
16:        if OutlierScore $O(x_j) >$ Threshold $\tau$ then
17:            $AddToSet$(OutlierSet $S_o$, $x_j$)
18:        else
19:            $AddToSet$(InlierSet $S_i$, $x_j$)
20:        end if
21:    end for
22:    InlierModel $w_{in} \leftarrow BuildSVMModel(InlierSet\ S_i)$
23:    AdditionalInliers $S_{ai} \leftarrow GetCorrectlyClassifiedInstances(w_{in}, S_o)$
24:    FinalSet $S_{fi} \leftarrow S_i \cup S_{ai}$
25:    FinalModel $w_{fi} \leftarrow BuildSVMModel(S_{fi})$
26: end procedure
```

We are initially given a training set S of n examples with α positive examples and β negative examples. From this set we draw small subsets of examples. Let one such subset of examples be $X_i = (x_{1i}, x_{2i}, \ldots x_{ki})$. Note that the ratio of positive examples α_i to the ratio of negative examples β_i in the set X_i is same as that of α to β. For each such example set we train a support vector machine to obtain a weight vector w_i using the standard support vector formulation [2] given by:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ & y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, i = 1, \ldots, l. \end{aligned} \tag{1}$$

Once we obtain the weight vector w_i, we classify the whole set of examples S using w_i. By this we obtain the classification scores and the predicted classification labels. The mis-classifications that result from this weight vector are aggregated in a vector O of dimension n. The procedure is repeated for many iterations and the element O_j indicates the likelihood of an example to be an outlier. The detailed algorithm is presented in Algorithm 1.

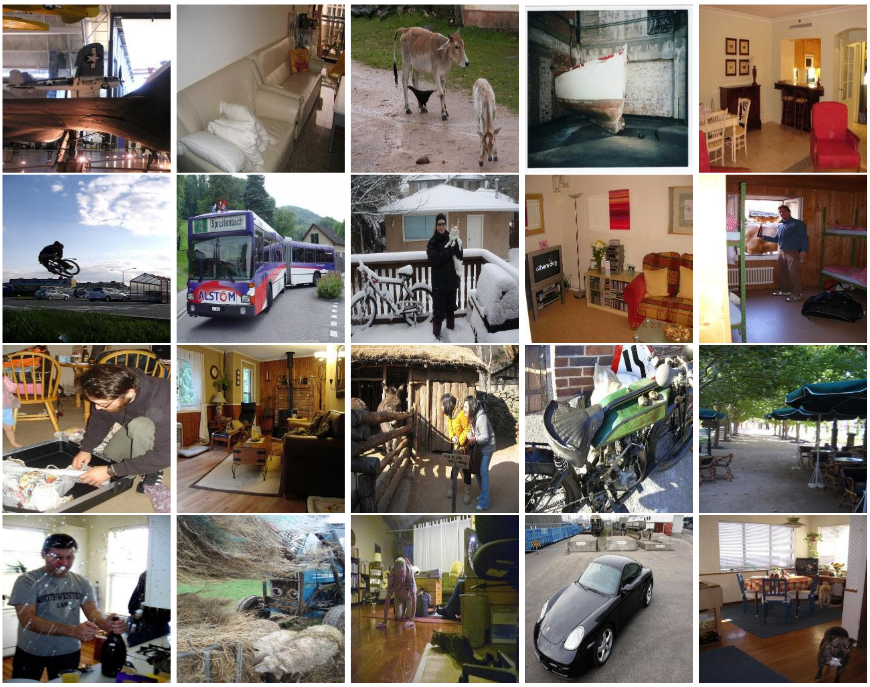

Figure 1: Images from each class of VOC 2007 dataset whose labels were flipped as they were closest to the hyperplane of a linear SVM and thus considered difficult to annotate. The classes are (in row major order): Aeroplane, Bicycle, Bird, Boat, Bottle, Bus, Car, Cat, Chair, Cow, Dining table, Dog, Horse, Motorbike, Person, Potted plant, Sheep, Sofa, Train, and Tv Monitor

The method described so far is outlined in lines 1 to 21 of the Algorithm 1. We used an outlier likelihood threshold τ that determines whether based on the miss-classifications obtained so far, an example x_j is an outlier or not. Based on this threshold τ, all the elements having $O_j > \tau$ are used to obtain the outlier set S_o and an inlier set S_i. A weight vector w_{in} is then learnt using all the samples belonging to the inlier set S_i. In lines 22 to 25 of Algorithm 1, we describe the further processing of the examples. The remaining outlier elements $(S - S_i) = S_o$ are again classified using w_{in} and those examples that are correctly classified form a set of additional inliers S_{ai}. The final inlier set S_{fi} is then obtained by adding the additional inliers S_{ai} to the inlier set S_i. This set is then used to train a final inlier model w_{fi}. The reason we add the additional inliers S_{ai} to our inliers set S_i is because we believe that the model w_{in} being trained on the inliers set S_i, will classify the inliers correctly. Thus the outliers which get properly classified are added as S_{ai} and finally included training.

As can be seen in Algorithm 1, the method takes three parameters as input, the 'SetSize' k, the number of iterations m and the threshold τ. From our experiments we noticed that increasing the number of iterations m beyond a limit does not lead to any significant improvement in accuracy. The set size k in most cases worked best if kept small. However, varying the threshold τ led to significant changes in our results. This is because it directly decides how much misclassifications are to be tolerated. Thus both τ and 'SetSize' k are important parameters in our method. We present in subsection 4.3 an empirical analysis of the effect that the variation of threshold τ and 'SetSize' k has for our method.

The RANSAC-SVM [11], has few key differences from the proposed method. While RANSAC-SVM also samples X_i examples from S and learns model w_i, this model is then checked with respect to all samples to obtain the empirical loss for the set S. This procedure is continued for many iterations and the best such model w_{max} corresponding to the lowest empirical loss is chosen as the final model. In our approach, instead of measuring the efficacy of models w_i, we measure the likelihood for each example to be an outlier. In the end, we relearn the final model w_{fi} using all the inlier examples that we obtain through the procedure. In the subsequent section we show that this difference is key to learning a far more accurate model with an improvement in mean average precision of more than 12%.

4 Evaluation

4.1 Dataset and Experimental Setup

We perform our tests on the PASCAL Voc 2007 dataset [4]. It consists of twenty categories, namely aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, diningtable, dog, horse, motorbike, person, pottedplant, sheep, sofa, train,and tvmonitor. For each category, we have two labels +1 and -1, representing the presence and absence of an object respectively. Also some images are marked as '0' indicating hard positive images. Images may also belong to more than one category. We have used training and validation sets together as our training set. The presence of hard examples in this dataset makes it an obvious choice for the evaluation of our algorithm.

Recently, convolutional neural networks (CNN) have been gaining popularity for feature extraction in various computer vision applications. For describing each image, we have used caffe[8] features pre-trained on the Imagenet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) [12]. The caffe network used, takes as input *number_of_images* x 3 x 227 x 227 sized images and has 8 layers in total. Prior to extracting the features, each input image is re-sized to 221 x 221. The responses of the 7th layer are used for feature representation, giving us a 4096-length feature vector for each input image. The method described is not specific to any descriptor and other descriptors can easily be used instead. We have used

linear SVM [5] in all our experiments as it offers good generalization.
In order to artificially insert noise, we trained a linear SVM using the whole training set and flipped the labels of 20% of the examples which are closest to the hyperplane. We do this with the assumption that examples which are hard in the feature space are also visually hard and hence more susceptible to incorrect annotation. Some of these images are shown in Figure 1. We notice that these images are indeed hard and hence prone to annotation errors. We use mean average precision as our performance measure in the following sections.

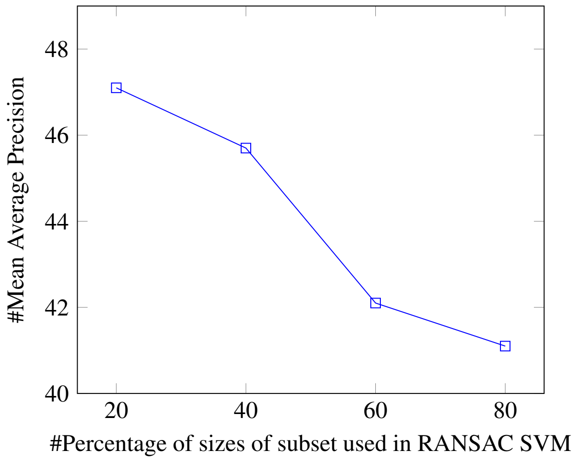

Figure 2: Effect of change of subset on our adaptation of RANSAC SVM method for 20% flip on mAP using the validation set

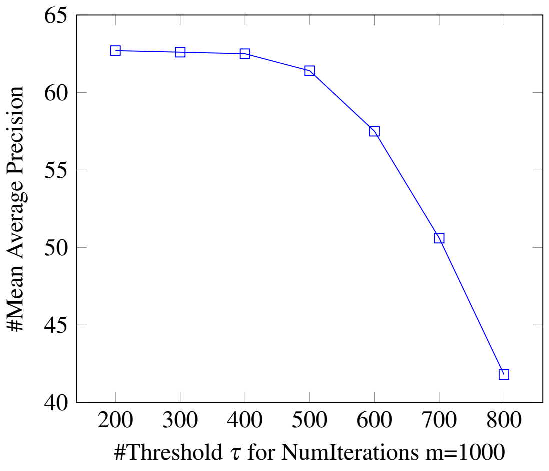

Figure 3: Effect of Threshold on our adaptation of RANSAC SVM method τ for 20% noise on mAP using NumInteration m=1000 and SubsetSize k=20% of training data on test set

We compare the results obtained using RANSAC-SVM, the full SVM solution and our method. We notice from Table 4.1 that using RANSAC-SVM improves the result as compared to using the full SVM solution. This is because the full SVM model naively uses all the examples including the incorrect labels in the training set. But RANSAC-SVM uses only a subset of the training data and hence excludes some examples labelled incorrectly, thus it performs better overall as compared the full SVM solution. We notice significant improvement in average precision by using RANSAC-SVM for the classes : bicycle, bird , boat, bus, cow, diningtable, horse and sheep. However, for the classes : cat and tvmonitor, we notice a minor decrease in accuracy.

Class	Full SVM	RANSAC-SVM	Our method
aeroplane	47.8	51.2	83.7
bicycle	46.1	56.6	70.7
bird	44.9	59.5	80.7
boat	41.0	58.5	76.0
bottle	24.3	27.1	29.8
bus	18.5	26.8	58.0
car	74.7	75.2	79.4
cat	53.8	52.5	72.9
chair	50.4	53.8	56.1
cow	16.8	36.4	46.2
diningtable	23.9	40.8	55.8
dog	54.9	55.1	72.9
horse	38.8	49.7	67.6
motorbike	33.7	42.7	63.3
person	88.4	88.8	89.5
pottedplant	27.2	31.7	35.3
sheep	14.4	34.7	45.0
sofa	36.6	39.1	42.0
train	52.6	57.0	79.6
tvmonitor	34.4	34.1	50.0
mAP	41.2	48.6	62.7

Table 1: Average precision of each class with 20% noisy labels for Full SVM, RANSAC-SVM and Our method. The average precision is computed with respect to the test data of Voc 2007.

4.2 Results

Our method outperforms both the full SVM solution and RANSAC-SVM in all the categories of the dataset. Also the increase in accuracy is quite significant for most of the classes except for the category 'person', where the improvement is only marginal. This happens mainly due to the fact that RANSAC-SVM uses the noisy training data as validation to select the best submodel. This is handled by our method automatically as we use the decisions of many submodels instead of one, and rather try to identify the set of examples which have maximum disagreement with rest of the training data. Thus using multiple validations instead of one provides us with an edge and helps us to achieve better robustness to noisy data. We used 20% of the training data as the subset size for RANSAC-SVM and used one thousand iterations to find the best submodel. And for our method, we have used 20% training data as subset size k and one thousand iterations as m and threshold τ as 20% of number of iterations. All the parameters were tuned using the validation set provided by VOC 2007. We discuss the result of varying these parameters in the next subsection.

4.3 Parameter Analysis

As mentioned earlier one of the most important parameters for our adaptation of RANSAC-SVM is the subset size. We experimented using various subset sizes and found 20% of the full training set to be the optimal size. We notice from Figure 4.1 that on increasing the subset size, the result gradually degrades. And with 80% subset size, we obtain a result which is close to full SVM solution. This can be justified as a smaller subset size will have less probability of containing noise as compared to a larger subset. Using the validation set, we observed that the accuracy degrades on increasing the threshold τ beyond 20% of number of iterations. This can be justified as with a higher threshold lesser outliers are removed and thus more incorrect examples are used for building the final model. From the Figure 4.1, we notice that this holds true for the test set as well and a smaller threshold yields better accuracy.

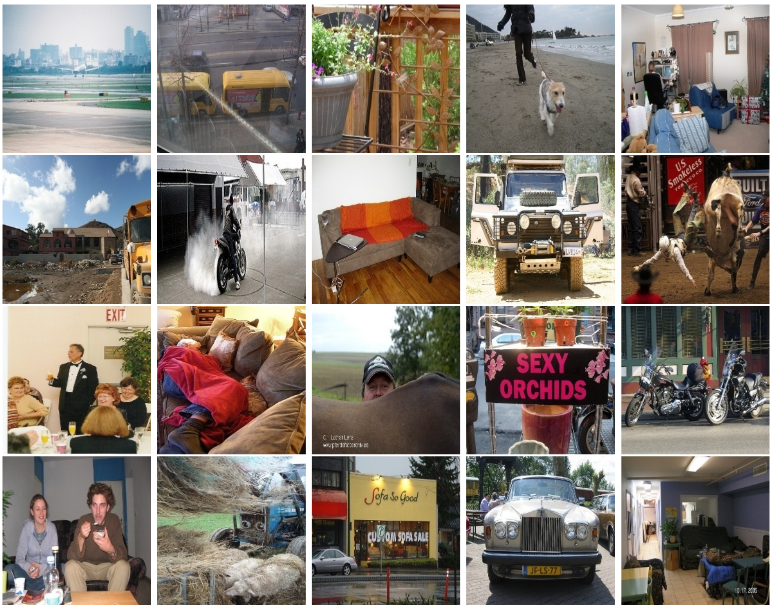

Figure 4: Visually hard images of VOC 2007 which were detected as outliers. The classes are (in row major order): Aeroplane, Bicycle, Bird, Boat, Bottle, Bus, Car, Cat, Chair, Cow, Dining table, Dog, Horse, Motorbike, Person, Potted plant, Sheep, Sofa, Train, and Tv Monitor

4.4 Hardness of examples

As we have defined outliers as any example which highly disagrees with the majority of other examples, thus our method can be adapted to identify hard examples in the training set as well. We use the original PASCAL VOC 2007 dataset to compute our results to prove our claim. We show in Figure 4 some of the images which have been labelled as hard by our algorithm. We notice that these images are indeed visually very difficult to identify. We show in Table 2 that we obtain good generalization when we exclude these images from our training set. We show that even by using 0 labels, our method works well as compared to full SVM without using 0 labels. We notice that although we obtained an overall marginal improvement, but for some classes namely aeroplane, cat, cow, horse, and sofa the full SVM solution gave better results. Thus, courtesy the above results, we show how our method identifies both noisy and difficult examples and also helps in obtaining good generalization

Class	Full SVM without zero labels	Our Method with zero labels	Our Method without zero labels
aeroplane	**88.6**	87.3	87.6
bicycle	80.1	78.9	**81.4**
bird	83.7	83.1	**84.5**
boat	81.3	81.4	**82.1**
bottle	38.3	37.7	**39.2**
bus	67.8	66.8	**68.7**
car	83.9	82.8	**84.5**
cat	**80.7**	79.6	80.6
chair	57.4	**60.2**	59.2
cow	**63.9**	63.2	63.8
diningtable	68.9	66.7	**71.5**
dog	78.2	77.1	**79.1**
horse	**83**	81.6	81.8
motorbike	74.2	72.8	**74.7**
person	90.1	90.2	**90.8**
pottedplant	47.4	47.8	**51.8**
sheep	72.5	72.3	**72.9**
sofa	**60.2**	56.6	60
train	87	86.8	**87.2**
tvmonitor	67.6	68.1	**69.3**
mAP	72.7	72.0	**73.5**

Table 2: Average precision of each class in Pascal VOC 2007 dataset using full SVM excluding examples labelled as 0 , our method in the presence of the 0 labelled examples and our method after removing the 0 labelled examples. Even by using the 0 labels, our method has nominal degradation and without using zero labels performs better than the full SVM solution

5 Conclusion and Future Work

In this paper we show an adaptation of RANSAC SVM by aggregating the misclassification scores for many models, each of which is obtained by training on a small randomly selected set of examples. This approach has been validated for the case of wrongly labeled examples which can be caused due to error in annotation. However, as the method samples examples randomly, thus if the training data contains over or nearly 50% of noisy examples, then the method may not work. However, if majority of labels are flipped in a binary classification problem, then without any additional supervision it is very hard to identify the incorrect examples.

We also show how this method can be used to identify very hard examples in a dataset. Because of the simplicity of the algorithm, its applications can be vast. As our basic outlier algorithm does not impose any restriction on the features to be used, it can be adapted to assist in other machine learning problems like fraud detection.

References

[1] Adela Barriuso and Antonio Torralba. Notes on image annotation. *CoRR*, abs/1210.3448, 2012. URL http://arxiv.org/abs/1210.3448.

[2] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[3] Sunglok Choi, Taemin Kim, and Wonpil Yu. Performance evaluation of ransac family. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 81.1–81.12, 2009. ISBN 1-901725-39-1.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2007 (voc2007) results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[5] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[6] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. ISSN 0001-0782. doi: 10.1145/358669.358692. URL http://doi.acm.org/10.1145/358669.358692.

[7] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 6:381–395, 1981.

[8] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, MM '14, pages 675–678, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3063-3. doi: 10.1145/2647868.2654889. URL http://doi.acm.org/10.1145/2647868.2654889.

[9] T. Leung, Yang Song, and J. Zhang. Handling label noise in video classification via multiple instance learning. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2056–2063, Nov 2011.

[10] Nagarajan Natarajan, Inderjit S. Dhillon, Pradeep D. Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *27th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1196–1204, 2013.

[11] K. Nishida and T. Kurita. Ransac-svm for large-scale datasets. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4, Dec 2008. doi: 10.1109/ICPR.2008.4761280.

[12] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, pages 1–42, April 2015. doi: 10.1007/s11263-015-0816-y.

[13] Peter Welinder, Steve Branson, Serge Belongie, and Pietro Perona. The multidimensional wisdom of crowds. In *Neural Information Processing Systems Conference (NIPS)*, 2010. URL http://vision.ucsd.edu/project/visipedia.