Object Recognition with Näive Bayes-NN via Prototype Generation

Hugo Jair Escalante¹, Mauricio Sotomayor²,
Manuel Montes¹, and A. Pastor Lopez-Monroy¹

¹ Computer Science Department, INAOE,
Luis Enrique Erro No. 1, Puebla, 72840, Mexico
{hugojair,mmontesg,pastor}@inaoe.mx

² Instituto Tecnológico de Orizaba,
Av. Oriente 9 No. 832, Orizaba, Ver., 66451, Mexico

Abstract. Näive Bayes nearest neighbors (NBNN) is a variant of the classic KNN classifier that has proved to be very effective for object recognition and image classification tasks. Under NBNN an unseen image is classified by looking at the distance between the sets of visual descriptors of test and training images. Although NBNN is a very competitive pattern classification approach, it presents a major drawback: it requires of large storage and computational resources. NBNN's requirements are even larger than those of the standard KNN because sets of raw descriptors must be stored and compared, therefore, efficiency improvements for NBNN are necessary. Prototype generation (PG) methods have proved to be helpful for reducing the storage and computational requirements of standard KNN. PG methods learn a reduced subset of prototypical instances to be used by KNN for classification. In this contribution we study the suitability of PG methods for enhancing the capabilities of NBNN. Throughout an extensive comparative study we show that PG methods can reduce dramatically the number of descriptors required by NBNN without significantly affecting its discriminative performance. In fact, we show that PG methods can improve the classification performance of NBNN by using much less visual descriptors. We compare the performance of NBNN to other state-of-the-art object recognition approaches and show the combination of PG and NBNN outperforms alternative techniques.

Keywords: NBNN, Prototype generation, KNN, Object recognition.

1 Introduction

Näive Bayes nearest neighbors (NBNN) is a variant of the KNN classifier that has proved to be very effective in diverse pattern recognition and computer vision tasks. The underlying idea behind NBNN is that the category of an object can be determined by comparing all of the visual descriptors associated to training and test objects. NBNN has outperformed other state-of-the-art methodologies

(e.g., the bag-of-visual-words) in diverse domains and has motivated research that tries to extend and take advantage of similar ideas [1,26].

In spite of the competitive performance of NBNN, it has a major disadvantage that limits its use for certain applications/domains: it requires of large storage and computational resources. Notwithstanding this limitation of NBNN, its effectiveness and theoretical justification plentifully justify studying ways of improving the efficiency of this methodology. This paper aims at determining whether well known techniques from pattern recognition could amend the efficiency drawback of NBNN without compromising its competitive performance.

We focus on the application of prototype generation (PG) approaches with the goal of reducing the set of visual descriptors that NBNN requires for classifying test instances. PG methods are a type of instance reduction methodology in which a small set prototypical instances are learned, with the constraint that prototypes should be able to effectively represent a whole data set and using the smallest number of prototypes [22]. PG methods have proved to be very effective for KNN classifiers [22,19]. Thus, we hypothesize that PG methods could also be helpful for boosting the performance of NBNN. To the best of our knowledge, the use of PG methods for improving the efficiency of NBNN is a research topic that has not been explored yet.

Along this paper we report a comparative study that considers the most effective PG methods proposed so far and evaluate its usefulness for object recognition using NBNN as classifier. We report experimental results in a database that has been used in previous studies [2] and show that some PG methods indeed are able of reducing the storage and computational requirements of NBNN without compromising its discriminative capabilities. In fact, we show that some methods can even improve the classification performance of NBNN by using much less instances. We compare the performance of NBNN to alternative state-of-the-art approaches and confirm its competitiveness.

2 Considered Object Recognition Methods

This section describes the NBNN [1] method, the pattern classification technique that is focus of our study, and the bag-of-visual-words approach [28], a state-of-the-art object recognition methodology against we compare NBNN.

2.1 Naïve Bayes Nearest Neighbors Method

Recently, it has been shown that pattern recognition techniques based on image-to-image comparisons can obtain state-of-the-art performance in computer vision tasks. This type of techniques rely on direct comparison between descriptors of training and testing objects. One of the most representative methods in this category is NBNN, which is closely related to KNN classification [1].

Under NBNN a set of visual descriptors (e.g., SIFT [16]) are extracted from each training image and they are stored. Each visual descriptor is a numerical vector, thus each image is associated to a set of vectors; in the data set we consider each image has ≈ 450 visual descriptors on average. When a new image has

to be classified, we extract its visual descriptors. For each test-image descriptor we estimate its nearest-neighbors in the set of training visual descriptors of each class. That is, for each descriptor we extract nearest-neighbors for each of the classes. For each class, we add the distances of test descriptors to their nearest-neighbors of the class. The class for a test image is assigned by looking at the minimum cumulative distance across classes. NBNN's classification criterion is as follows:

$$\hat{C} = \arg \min_C \sum_{i=1}^n ||d_i - NN_C(d_i)|| \quad (1)$$

where $d_{1,...,n}$ is the set of visual descriptors from the test image and $NN_c(d_i)$ is the nearest training neighbor of d_i with class C , note we assume a single nearest neighbor. Boiman et al., argued that the direct comparison of descriptors is more discriminative than vector quantizing the descriptors (e.g., under the bag-of-visual words framework). In fact, they show that under the naïve-Bayes assumption, the theoretically optimal image classifier is effectively approximated by NBNN.

One should note that NBNN is a highly inefficient method as one has to estimate distances between all of the training and testing descriptors. Hence, instance reduction methodologies can be applied to improve its efficiency. In this work, we apply PG methods to the set of training visual descriptors, that is, we consider each visual descriptor from each training image an instance of a pattern classification problem (the class of a descriptor is the class corresponding to the image from which the descriptor was extracted). Under these assumptions, PG methods can be applied directly. The goal is to keep only a reduced subset of descriptors and speedup the classification process of NBNN.

2.2 The Bag-of-Visual-Words Representation

The bag-of-visual-words (BoVW) is one of the most used representations for images and videos in many computer vision tasks [28]. The underlying hypothesis is that the content of objects (e.g., images or videos) can be effectively described by histograms that account for the presence of representative features, called visual words. This is in analogy to the bag-of-words representation widely used in information retrieval and text mining. The standard approach for generating visual words is by means of clustering methods, k -means in most of the cases. Low-level descriptors extracted from images are grouped into k -clusters, the k -centers of the learned clusters are considered as visual words. Under the BoVW approach each image is represented by a single vector and standard classifiers can be used, e.g., SVM or KNN. In this paper we compare the performance of NBNN to the BoVWs formulation.

One should note that although unsupervised methods have been used for generating visual vocabularies in BoVWs, object recognition is a supervised classification task and then class information is available for each training visual

descriptor. Therefore, it makes sense to ask ourselves whether PG methods could also be used to learn the visual vocabularies: using a PG method to generate discriminative prototypes, playing the role of visual words, instead of applying k -means, and use the resultant centers. With the goal of answering this question we also evaluate the suitability of PG methods for generating codebooks for BoVWs representations. Hence, using PG methods for codebook generation is another contribution of this work.

3 Prototype Generation

Prototype-based classification can be seen as a KNN classifier using a subset of the total of training instances [10]. Usually the classification rule is 1NN, and the core of this type of methods lies in the way to obtain a small set of prototypical instances that effectively summarize the whole training set. Many PG¹ methods have been proposed so far. Triguero et al. recently performed a complete experimental study that considered most of the successful PG methods proposed up to 2012.

Among the most effective type of methods are those based on position adjustment and evolutionary algorithms. In evolutionary algorithms the general idea is to consider prototypes as points in the d -dimensional input space. The positions of points (prototypes) are updated trying to maximize a criterion related to the classification performance using 1NN in a training or validation set [6,17,7,24,25,4]. Vector quantization methods follow a similar principle, although these methods are based on a gradient-descend like optimization algorithm [27,13,10,8]. Other competitive methods include those based on edition [14] or condensation [3] principles. Alternative strategies include those based on sub-sampling [9].

For this study we consider all of the PG methods available in the comprehensive KEEL toolbox [11]. This toolbox contains implementations of most PG methods proposed so far. Initially all of the PG methods in the KEEL toolbox were considered, however, we report results on PG for NBNN only for those methods that completed the processing of the considered data set within a 2-weeks period². Table 3 provides a brief description of each of the methods that were used in this study. These methods comprise a wide diversity of methodologies, besides, these are the methods that could be applied efficiently for mid-size data sets. PG methods are applied using the set of visual descriptors extracted from all of the training images, the selected prototypes are then used with NBNN and the BoVWs formulation for object recognition.

¹ Prototype selection is also an option [18], in this study we focused on PG techniques as these methods are more general than selective ones (e.g., they can generate prototypes outside the training set). However, for future work we will also study the suitability of prototype selectors.

² Experiments were performed in a workstation with Core i7-3820 processor at 3.60Ghz and 64GB in RAM.

Table 1. Considered PG methods <http://sci2s.ugr.es/pr/> for details

Method	Description	Ref.
BST3	Bootstrap technique for nearest neighbor	[9]
Condensation	Condensation PG method	[3]
AVQ	Adaptive vector quantization	[27]
DEGL	Differential Evolution using a Neighborhood-Based Mutation Operator	[24]
DSM	Decision surface mapping	[8]
GENN	Removes instances conservatively trying to maximize classification perf.	[14]
ENPC	Evolutionary algorithm that tries to maximize classification perf.	[6]
VQ/LVQ-X	Variants of the Learning Vector Quantization algorithm	[13]
IPADE	Iterative prototype adjustment based on differential evolution	[25]
MSE	PG using gradient descend and simulated annealing	[4]
POC-NN	Pairwise Opposite Class Nearest Neighbor	[20]
PSCSA	Prototype reduction using an artificial immune system	[7]
PSO	Particle Swarm Optimization	[17]
RSP	Reduction by space partitioning	[21]
SADE	Self-Adaptive Differential Evolution	[24]
SGP	Self-generating Prototypes	[5]

4 Experiments and Results

Experiments were performed using a subset of images from the Caltech-101 data set. We consider 4 categories: *camera*, *motorcycle*, *dollar bill* and *wristwatch*, see Figure 1. For each category 20 images were randomly selected for training and another 100 images for testing. This data set was collected by Chang et al. and we use in this paper exactly the same training/test partitions [2].

Fig. 1. Sample images from the training set for each of the considered categories

From each image (either for training or testing) we extracted its SIFT descriptors as described in [16], thus for each image we have a set of 128-dimensional feature vectors. Raw SIFT-descriptors were used directly for image classification with NBNN. For the BoVW approach, training descriptors were processed with k -means to obtain a visual codebook, which was used to represent images; in this aspect, we performed extensive experimentation for determining the optimal value of k . A linear SVM was used as classifier for all of our experiments with BoVW. Table 2 shows some statistics for each category of the data set.

We were able to run all of the PG methods available in the KEEL toolbox. However, only 22 of these methods returned a solution within 400 hours. As future work we will explore the use of stratification [23] for making feasible the application of the rest of PG methods. Default parameters were used for all PG methods (except for k -means, see below).

Table 2. Characteristics of the considered data set [2]

Category	Training		Test	
	<i>Imgs.</i>	<i>Descriptors</i>	<i>Imgs.</i>	<i>Descriptors</i>
Camera	20	8,477	100	43,777
Dollar bill	20	9,986	100	66,787
Motorcycle	20	9,236	100	25,836
Wristwatch	20	9,614	100	43,158
Total	80	37,313	400	179,558

Table 3 shows the main results of this paper (see also Figure 2). We show the test-set performance obtained by the two recognition methods (NBNN and BoVW) when using each of the 22 PG techniques. Also, we show for each PG method the number of prototypes/visual words obtained ($|P|$) and, its processing time (comprising the generation of prototypes only). For completion, we report in rows 2 and 3 the performance obtained by the different methods when using (i) all of the descriptors, and (ii) k -means for generating prototypes/visual words, respectively. We emphasize that for k -means we report results obtained with the best value of k we found in preliminary experimentation, this is in order to have a strong baseline.

Several interesting findings can be drawn from Table 3. First, the results when using all of the descriptors confirm those reported in [1]: NBNN performs similarly to the BoVWs formulation (compare columns (i)-NBNN and (ii)-BOW). One should note that even when NBNN did not outperform the performance of BoVWs with k -means, for the latter method we are reporting the best configuration we found after extensive experimentation.

Regarding the suitability of PG methods for NBNN, in terms of accuracy, it can be seen that not all PG methods were useful to obtain prototypes that can compare favorably with NBNN using all of the descriptors. In fact, the performance of NBNN decreases at different rates when using most PG methods. However, the number of instances is reduced dramatically in some cases. For instance, the IPAD [25] method learned only 4 prototypes (one per class) and still it was able to obtain a performance of $\approx 65\%$. On the other hand, methods like VQ, LVQ2 [13], and DSM [8] obtained performances ($\approx 82\%$) that are close to that obtained by NBNN with all descriptors (87%), however, for these PG methods only 10% of the number of original descriptors were used.

Interestingly, prototypes generated with GENN [14] and ENPC [6] methods improved the recognition performance of NBNN, with reduction rates of 39% and 69%, respectively. In fact, results obtained with prototypes generated with the ENPC method outperform any other combination of vocabulary-generation / object-recognition methods. The combination outperforms by more than 5% to the result obtained by NBNN when using all of the descriptors. Hence, by using discriminative prototypes we are able to outperform the popular NBNN approach and using 69% less of descriptors. Therefore, this PG method not only improves the recognition performance of NBNN but also makes more efficient the classification task.

Table 3. Recognition performance for the considered PG methods. For reference we also show the performance obtained when using all of the descriptors.

ID	Description	$ P $	Time	NBNN	BOW
(i)	All descriptors	37313	-	87.04	75
(ii)	K-means - <i>optimal k</i>	22780	01:06:41	81.50	90.50
1	AVQ	8	02:55:41	55.75	56.50
2	BST3	3731	01:03:36	74.75	87.00
3	Condensation	3731	04:04:02	72.50	82.50
4	DEGL	746	96:37:32	74.75	81.50
5	DSM	3731	01:10:36	82.00	89.75
6	ENPC	11244	201:04:57	92.50	89.75
7	GENN	22781	372:39:00	89.25	86.50
8	IPADE	4	01:06:14	65.25	65.00
9	LVQ 1	3731	04:32:45	81.50	90.00
10	LVQ 2	3731	02:02:17	82.25	90.00
11	LVQ 2.1	3731	02:03:45	81.50	89.50
12	LVQ 3 method	746	00:03:51	70.50	86.50
13	LVQ with pruning	3729	02:34:50	80.25	88.50
14	LVQ training counter	4	00:03:07	25.50	31.50
15	MSE	14	00:00:49	25.00	67.75
16	POC-NN	1904	245:55:10	62.25	88.00
17	PSCSA	14	25:27:40	25.00	67.75
18	PSO	746	09:35:57	53.50	78.00
19	RSP	18726	368:50:02	83.00	85.50
20	SADE	746	131:34:07	66.50	83.50
21	SGP	220	00:08:38	32.00	68.00
22	VQ	3731	02:14:36	81.50	89.75
MAX		22781	372:39:00	92.50	90.00
AVG.		3988.6	19:04:58	66.23	79.22
MIN		4	00:00:49	25.00	31.50

NBNN with ENPC prototypes outperforms by 2% to the best result obtained with the BoVWs formulation. This improvement is larger than results reported in previous works that have used discriminative prototypes for codebook generation [12,15]. Actually, ENPC-NBNN outperforms the method in [2] that uses the same database we consider. This approach is based on clustering of descriptors and a Bayesian network. The method in [2] achieves average (throughout classes) recall, precision and accuracy of 90.7%, 93.3% and 94.5%, respectively. Whereas ENPC-NBNN method obtains 92.5%, 92.8% and 96.5%, in the same measures. To the best of our knowledge the results reported in this paper are the best ones so far for the data set we consider.

Regarding the performance of PG methods under the BoVWs formulation, we found that the use of some PG methods for generating the vocabulary can be beneficial: PG methods did not improve the performance of BoVWs but reduced considerably the vocabulary size. For instance, with codebooks generated with LVQ1 and LVQ2 methods one can obtain virtually the same performance as when

using k -means with optimized k , but using much less visual words (a reduction of $\approx 83\%$). This result gives evidence that PG methods could be a promising solution for enhancing the BoVWs formulation. Nevertheless, we emphasize that k was optimized for k -means, whereas default parameters were used for all of the other techniques.

Fig. 2. Performance of NBNN and BoVWs when using the outputs of PG methods. The bottom plot is a zoom of the one in the top.

Figure 2 depicts the performances of NBNN and the BoVWs when using the outputs of PG methods (results are sorted by the number of generated prototypes used). It can be seen that for the data set we considered, the larger the number of prototypes, the better the performances of NBNN and BoVWs. We can clearly see that, for the BoVWs formulation, size seems to be more important than discriminative information: *the larger the size of the vocabulary the higher the performance*, although after 10,000 BoVWs performance does not change significantly. Under NBNN, several PG methods that obtain more than $\approx 3,000$ prototypes perform similarly to k -means. For instance, methods 5, 9, 10, 11 and 22 obtain differences in performance of less than 1% by using 83% less visual words.

Finally, regarding processing time, the most effective PG methods for BoVWs / NBNN classification we identified are also the more computationally expensive (e.g., ENPC and GENN). Although, PG is an off-line process that is performed a single time, it is important to mention that for larger data sets it would be very difficult to apply straight PG-methods. Therefore, efficient PG methods are highly needed, meanwhile stratification approaches can be adopted [23].

5 Conclusions

We described an experimental study that aims at evaluating the suitability of PG for NBNN. We reported experimental results in an object recognition task where we evaluated the performances of NBNN and BoVWs when using prototypes generated with PG methods. The conclusions of this work are as follows:

- PG methods resulted very useful for reducing the training set of descriptors for NBNN. Dramatic reductions were observed and still acceptable performance was obtained, even, when using default parameters for PG methods.
- GENN and ENPC methods improved the recognition performance of NBNN by using less training instances. ENPC achieved the best result overall and reduced significantly ($\approx 69\%$) the size of the data set. In fact, NBNN using prototypes generated with ENPC outperformed the best reported results for the considered data set.
- In general, computational efficiency is a problem on the use of PG methods for object recognition under NBNN. The best performer, ENPC, took more than 200 hours to return a solution for the data set we consider.
- Under the BoVWs formulation, PG methods can obtain comparable performance to that obtained with a traditional k -means approach with optimal selection of k , but using a much less number of prototypes/visual words.
- Vocabulary size is crucial for recognition purposes under BoVWs, but using discriminative visual words does not seem to help considerably.

Future work directions include the development of PG methods that directly optimize the NBNN criterion, and the study of stratification approaches to make scalable the application of PG methods for object recognition problems.

Acknowledgements. This work was partially supported by the LACCIR program under project id R1212LAC006. We are grateful with I. Triguero for his valuable support on the use of KEEL.

References

1. Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: CVPR, pp. 1–8 (2008)
2. Chang, L., Duarte, M.M., Sucar, L.E., Morales, E.F.: A bayesian approach for object classification based on clusters of sift local features. *Expert Syst. Appl.* 39, 1679–1686 (2012)
3. Chen, C.H., Jozwik, A.: A sample set condensation algorithm for the class sensitive artificial neural network. *Pattern Recog. Lett.* 17(8), 819–823 (1996)
4. Decaestecker, C.: Finding prototypes for nearest neighbour classification by means of gradient descent and det. annealing. *Pattern Recog.* 30(2), 281–288 (1997)
5. Fayed, H.A., Hashem, S.R., Atiya, A.F.: Self-generating prototypes for pattern classification. *Pattern Recog.* 40(5), 1498–1509 (2007)
6. Fernandez, F., Isasi, P.: Evolutionary design of nearest prototype classifiers. *Journal of Heuristics* 10, 431–454 (2004)

7. Garain, U.: Prototype reduction using an artificial immune system. *Pattern Analysis and Applications* 11(3-4), 353–363 (2008)
8. Geva, S., Sitte, J.: Adaptive nearest neighbor pattern classification. *IEEE Transaction on Neural Networks* 2(2), 318–322 (1991)
9. Hamamoto, Y., Uchimura, S., Tomita, S.: A bootstrap technique for nearest neighbor classifier design. *IEEE Trans. Pattern Anal. Mach. Intell.* 19(1), 73–79 (1997)
10. Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning*. Springer (2001)
11. Alcalá-Fdez, J., et al.: Keel: A software tool to assess evolutionary algorithms to data mining problems. *Soft Computing* 13(3), 307–318 (2009)
12. Jiu, M., Wolf, C., Garcia, C., Baskurt, A.: Supervised learning and codebook optimization for bag-of-words models. *Cogn. Comput.* 4(4), 409–419 (2012)
13. Kohonen, T.: The self organizing map. *Proc. of IEEE* 78(9), 1464–1480 (1990)
14. Koplowitz, J., Brown, T.: On the relation of performance to editing in nearest neighbor rules. *Pattern Recog.* 13(3), 251–255 (1981)
15. Lazebnik, S., Raginsky, M.: Supervised learning of quantizer codebooks by information loss minimization. *IEEE Trans. PAMI* 31(7), 1294–1309 (2009)
16. Lowe, D.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60(2), 91–110 (2004)
17. Nanni, L., Lumini, A.: Particle swarm optimization for prototype reduction. *Neurocomputing* 72(4-6), 1092–1097 (2008)
18. Olvera, A., Carrasco-Ochoa, J.A., Martinez-Trinidad, J.F., Kittler, J.: A review of instance selection methods. *Artificial Intell. Rev.* 34, 133–143 (2010)
19. Paredes, R., Vidal, E.: Learning prototypes and distances: a prototype reduction technique based on nearest neighbor error minimization. *Pattern Recog.* 39(2), 180–188 (2006)
20. Raicharoen, T., Lursinsap, C.: A divide-and-conquer approach to the pairwise opposite class-nearest neighbor (poc-nn) algorithm. *Pattern Recog. Lett.* 26, 1554–1567 (2005)
21. Sánchez, J.S.: High training set size reduction by space partitioning and prototype abstraction. *pattern recognition. Pattern Recog.* 37(7), 1561–1564 (2004)
22. Triguero, I., Derrac, J., García, S., Herrera, F.: A taxonomy and experimental study on prototype generation for nearest neighbor classification. *IEEE Trans. on Systems, Man, and Cybernetics–Part C* 42(1), 86–100 (2012)
23. Triguero, I., Derrac, J., Herrera, F., Garcia, S.: A study of the scaling up capabilities of stratified prototype generation. In: *NaBIC*, pp. 297–302. IEEE (2011)
24. Triguero, I., García, S., Herrera, F.: Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification. *Pattern Recog.* 44(4), 901–916 (2011)
25. Triguero, I., Garcia, S., Herrera, F.: Ipade: Iterative prototype adjustment for nearest neighbor classification. *IEEE Trans. Neural N.* 21(12), 1984–1990 (2010)
26. Tuytelaars, T., Fritz, M., Saenko, K., Darrell, T.: The NBNN kernel. In: *Proc. of the International Conference on Computer Vision*, pp. 1824–1831. IEEE (2011)
27. Yen, C.W., Young, C.N., Nagurka, M.L.: A vector quantization method for nearest neighbor classifier design. *Pattern Recog. Lett.* 25(6), 725–731 (2004)
28. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision* 73, 213–238 (2007)