Pattern Recognition 48 (2015) 60–72

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Iterative Nearest Neighbors

Radu Timofte [a,b,*], Luc Van Gool [a,b]

[a] VISICS, ESAT-PSI/iMinds, KU Leuven, Kasteelpark Arenberg 10, 3001 Leuven, Belgium
[b] Computer Vision Lab, D-ITET, ETH Zurich, Sternwartstrasse 7, 8092 Zurich, Switzerland

ARTICLE INFO

Article history:
Received 5 May 2013
Received in revised form
6 June 2014
Accepted 10 July 2014
Available online 21 July 2014

Keywords:
Iterative Nearest Neighbors
Least squares
Sparse representation
Collaborative representation
Classification
Dimensionality reduction

ABSTRACT

Representing data as a linear combination of a set of selected known samples is of interest for various machine learning applications such as dimensionality reduction or classification. k-Nearest Neighbors (k NN) and its variants are still among the best-known and most often used techniques. Some popular richer representations are Sparse Representation (SR) based on solving an l_1-regularized least squares formulation, Collaborative Representation (CR) based on l_2-regularized least squares, and Locally Linear Embedding (LLE) based on an l_1-constrained least squares problem. We propose a novel sparse representation, the Iterative Nearest Neighbors (INN). It combines the power of SR and LLE with the computational simplicity of k NN. We empirically validate our representation in terms of sparse support signal recovery and compare with similar Matching Pursuit (MP) and Orthogonal Matching Pursuit (OMP), two other iterative methods. We also test our method in terms of dimensionality reduction and classification, using standard benchmarks for faces (AR), traffic signs (GTSRB), and objects (PASCAL VOC 2007). INN compares favorably to NN, MP, and OMP, and on par with CR and SR, while being orders of magnitude faster than the latter. On the downside, INN does not scale well with higher dimensionalities of the data.

1. Introduction

It is a common assumption that data has an intrinsic structure which significantly influences the performance of the subsequent applications using it. This is the case especially for classification and dimensionality reduction.

Dimensionality reduction techniques set out to preserve or enhance one or the other property under the reducing projection. Some linear techniques are used for this: Principal Component Analysis (PCA) [1] enhances the concentration of variance, Locality Preserving Projection (LPP) [2] conserves the local affinities, Sparse Representation Linear Projection (SRLP) [3] embeds a sparse representation, and Linear Discriminant Analysis (LDA) [4,5] maximizes the inter-class vs. intra-class variance. Among the non-linear techniques we have Locally Linear Embedding (LLE) [6] which preserves the local neighborhood, Laplacian Eigenmaps (LE) [7] and Supervised Laplacian Eigenmaps (SLE) [8] preserving the distances to neighbors, and Sparse Representation Embedding (SRE) [3] preserving the sparse representation.

For classification purposes we have to define those properties and measures used to match a new sample or 'query' against the known, labeled samples or the learned class model. Picking the label of the nearest neighbor as the class label for the query is the simplest such classification decision. 'Nearest' is defined on the basis of some similarity, distance, or metric [9]. The Sparse Representation-based Classifier (SRC) starts from the l_1-regularized least squares decomposition, a Sparse Representation (SR). The decision is based on the class labels of the samples that contribute to the representation of the query. SRC yields state-of-the-art performance in e.g. face recognition [10]. The Collaborative Representation (CR) Classifier (CRC) [11] uses instead the l_2-regularized least squares decomposition.

Apart from the quality of the results, the computational efficiency can be critical in practice. The high performance of SR and CR comes at the price of a computational burden. k NN in comparison is very fast, but less performant.

We aim at combining the high speed of k NN and the performance of SR. Our proposed method is coined the Iterative Nearest Neighbors (INN) representation and is inspired by feedback loop methods. First, INN searches for the nonzero terms in the representation, instead of the weights like most current l_1 solvers would. Each new selection of a sample in the representation is meant to compensate for the errors introduced by the previous selections and, as a consequence, to further reduce the residual between the query and its reconstruction. We have an iterative procedure depicted in Fig. 1. Starting from the query

* Corresponding author at: Computer Vision Lab, D-ITET, ETH Zurich, Sternwartstrasse 7, 8092 Zurich, Switzerland. Tel.: +41 44 63 25279; fax: +41 44 63 21199.
E-mail addresses: radu.timofte@vision.ee.ethz.ch (R. Timofte), vangool@vision.ee.ethz.ch (L. Van Gool).
URL: http://www.vision.ee.ethz.ch/~timofter/ (R. Timofte).

http://dx.doi.org/10.1016/j.patcog.2014.07.011
0031-3203/

sample, INN first gets its nearest neighbor from the training pool of samples. In the next iterations, it each time picks the sample best suited to reduce the residual between the query sample and its reconstruction on the basis of the previously selected samples. The reconstruction is obtained as a linear combination. This process is repeated until a stopping criterion is met. This can be a maximum number of iterations, a residual threshold, or when the sum of the remaining weights in a potentially infinite representation gets too small compared to the sum of the weights already assigned to samples. We transform the selection of each new term into a NN decision by adapting the query at each step by adding the weighted residual introduced by the previous selection. This weight is the regularization parameter of the method. As will become clear, the monotonously increasing sum of the imposed weights in the representation is guaranteed to have a limited value and after a limited number of iterations, the remaining terms can be discarded given an application-specific (error) tolerance to noise.

INN is an iterative procedure and its complexity is dominated by the maximum number of iterations (nonzeros in the representation) $\times$ the time of the NN search. Experimentally, we found the performance to be robust for a wide range of values for the parameter regulating the importance of the residual introduced by each newly picked NN. It is comparable to the Lagrangian parameter used in l_1 and l_2-regularized least squares solvers. Nevertheless, tuning the parameter for specific data usually still brings some improvement (as with l_1 and l_2-solvers).

This paper is an extension of our previous work [12] which introduced INN in its original form (in the sequel referred to as INN0). We show INN0 to be a valid approach in cases with moderate amounts of noise, and show that a proposed refinement brings no significant advantages. Moreover, we also extend the empirical validation of the method by comparing it against similar greedy iterative recovery algorithms, i.e. Matching Pursuit (MP) [13] and its variant Orthogonal Matching Pursuit (OMP) [14,15].

The rest of the paper is structured as follows. First, we review and refine the Iterative Nearest Neighbors method in Section 2. We emphasize similarities with other iterative methods by benchmarking its performance and we provide an analysis of its complexity, bounds, and approximations. Then, in Section 3 we derive our INN-based variant of the SRLP dimensionality reduction method, our INN-based Classifier, and INN-based variants for the Naive Bayes classifier and the Sparse Coding Spatial Pyramid Matching method. Section 4 experimentally validates the proposed INN representation and its derived applications, using real-life datasets. The conclusions are drawn in Section 5.

2. Iterative Nearest Neighbors

First we briefly review the main motivation and assumptions we made, then we introduce our INN proposal and provide a theoretical and empirical analysis.

2.1. Motivation and assumptions

We want to combine the speed and simplicity of kNN or MP with the performance of SR or CR. This is our main motivation. Regarding these properties, we make two observations. Firstly, kNN and MP rely on nearest neighbors and this to a large extent accounts for their speed. Secondly, SR solvers optimize over the weights, and seem to draw much of their performance from the fact that the main supporting samples in the SR representation tend to belong to the data class/cluster of the input query. Thus, we propose a method that combines these elements. Moreover, we introduce a controlling parameter that allows our algorithm to trade off performance for speed.

Three important design choices we make in defining our INN representation are

(1) We try to ensure that the most important samples in the representation are similar to the input query. This motivates the use of nearest neighbor search as main operation and not a regression involving scaling.

(2) We envisage a sequential selection process of the samples, with their weights going down as the algorithm proceeds. We will impose guiding weights for the representation and, with these weights steadily decreasing, the importance of each subsequent sample decreases. This said, a specific sample can be selected multiple times, thus even if the weights are predefined discrete values of a decreasing function, the actual weight of such repeatedly selected sample is the sum of the corresponding weights. As a result, the first selected samples are not necessarily the ones with the largest impact (cumulated weight) in the representation.

(3) We use a working query updated after each selection. The update is an addition of the weighted residual between the working query and the selected sample. In this way:

(i) the energy (i.e. the l_2-norm) of the working query is slowly changing,
(ii) the working query embeds the history of our selections,
(iii) the impact of the residual is controlled by a weighting parameter, the same controlling the decay of the weights, and

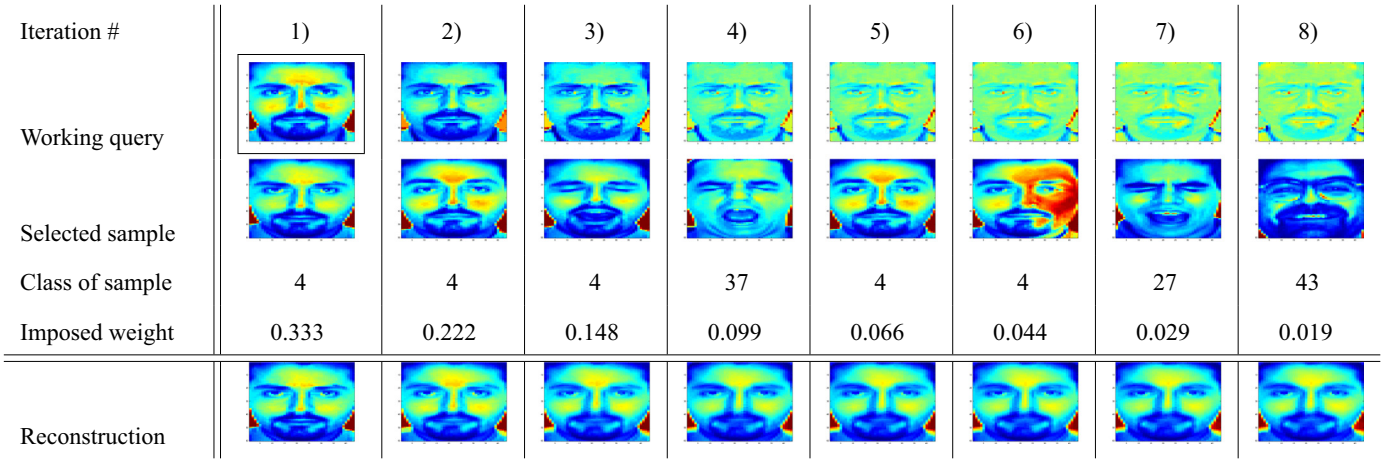

Fig. 1. INN0 algorithm iteration by iteration. 8 iterations for $\lambda=0.5$ and $\beta=0.95$ (95%). At each iteration we show the working query, the selected NN sample from the training pool, its class, the imposed weight, and the current reconstruction. The input query has a summed confidence of 0.813 to belong to class 4.