Convolutional structures and marginal statistics–a study based on K-nearest neighbours

Jugurta Montalvão, Jânio Canuto, and Elyson Carvalho

***Abstract*—This paper addresses statistical tricks found in deep convolutive neural networks. First, the most relevant statistical tricks are studied under the perspective of data scarcity, then one of them, directly related to convolution-like structures, is regarded as a random variable marginalization. The same kind of marginalization is implemented in an ensemble of K-nearest neighbours cells, where each cell yields scores instead of class labels. Scores are then combined to improve classification accuracy, as compared to a conventional K-nearest neighbours classifier in experiments with two emblematic datasets—MNIST and CIFAR-10. This improvement is regarded as evidence of the variable marginalization effect over performance, whereas it is discussed the potential for further lessons learned from deep neural networks to be transferred to KNN based classifiers, whose advantage is to allow for explainable artificial intelligence.**

***Index Terms*—Data scarcity, explainable Artificial Intelligence, KNN Network.**

I. Introduction

Researchers in different domains are paying attention to the so-called deep learning structures. While most published effort are aimed at exploring the practical advantages of this renewed tool, some researchers, since Bengio et al. [1], are struggling to understand why modern deep neural networks (DNN) eventually started to consistently outperform other pattern classification structures, in spite of many years of experimental disappointment with multiple layers of artificial neural networks (ANN). Beyond the scientific curiosity, this effort to provide theoretical explanations to the outstanding performances of DNN is also driven by the increasing need for artificial intelligence explainability, mainly in sensible applications such as self-driven cars, or medical and military uses.

Relevant works, such as [2] and [3] seem to converge to the conclusion that functions implemented by DNN, although very complicated, preserve an intrinsic rigidity caused by pattern space foldings that help the resulting classifier to generalize unseen samples better than shallow models. However, it was observed, for many years of experimentation with ANN with more than one hidden layer that these networks do not necessarily yield good pattern recognition machines by themselves, without some carefully crafted training procedures, which can be roughly separated into three main (non-exclusive) approaches, namely:

(a) Training each layer as an auto-encoder (stacked auto-associators).
(b) Greedy layer-wise supervised training.
(c) Weight sharing.

Approaches (a) and (b) were carefully studied by Bengio et al. [1], whereas (c) has been studied since the 80's by Yann le Cun [4]. Record-breaking results have been obtained with deep structures that combine (a) and (c), but impressive results with (c) alone were pointed out many years even before the term *deep learning* was coined [4]. On the other hand, strategy (a) seems to be the starting point of the renewed interest in DNN, through works such as [5], [6] and [7].

Whatever the understanding of how DNN works, it is undisputed that the backpropagation used during the training of DNN somehow encodes the training dataset as neuron weights. Unfortunately, the resulting codes are too entangled to allow explainable artificial intelligence. Indeed, the extraction of rules from trained artificial neural networks is not a new concern [8], and it seems to come back with the renewed interest in ANN, through the DNN.

Unlike ANN, K-nearest neighbour (KNN) is a much simpler pattern recognition strategy whose performance has been studied for more than half a century, as detailed in Section III. Simplicity and theoretical background are the main reasons why we chose KNN for this study. Besides, KNN classifiers explicitly use labelled patterns given for *training* in their structures, which allows for straightforward manners to find out why a given decision is made, which can be very attractive, mainly in critical applications involving human lives, such as in healthcare, military devices and self-driven cars. The downside of KNN, however, is its apparent poor performance, as compared to state of the art DNN. However, the theoretical statements regarding KNN, recalled in Section III, suggests that this difference can be partially explained by data scarcity, as defined in Section II, which makes KNN even more well fitted to this study.

Therefore, this research does not propose a new method, but it rather studies (through KNN) the fundamental relationship between intrinsic dimension of data, data scarcity and classification performance, highlighting the statistical role of convolution-like structures, where processing image patches can be regarded as extracting marginal statistics (i.e. marginal statistics of the random variable that models the image source). Our goal is not to outperform DNN with alternative structures, but to use KNN as a tool to understand at least some elements that make state-of-the-art DNN to excel. Besides, explainability is perceived as a by-product of using either KNN or ensembles of KNN cells, which may suggest the ensembles of KNN cells not only as a tool for theoretical studies, as we did, but also as an actual classifier in critical applications.

J. Montalvão, Jânio Canuto, and Elyson Carvalho are with the Federal University of Sergipe (UFS), São Cristóvão, Brazil e-mail: jmontalvao(at)ufs.br.
Digital Object Identifier: 10.14209/jcis.2018.19

In this work, however, we emphasize that only one of the underlying statistical tricks found in DNN (as explained in Section II) is isolated and transferred to the ensembles of KNN cells.

This paper is organized as follows: in Section II a brief study of the concept of scarcity is presented, along with a definition of extreme scarcity measure, which is applied in Section IV to set patch sizes according to the corresponding effective manifold dimension. In Section III a network of KNN cells is used as a structure to implement a trade-off between data manifold dimension and training dataset size. Back to Section IV, some experimental evidences are gathered from experiments with two emblematic datasets, namely MNIST [9] and CIFAR-10 [10]. In Section V we discuss the experimental evidences and their implications, whereas in Section VI we conclude this paper by delineating some straightforward perspectives for future research.

II. Data scarcity

A well-known issue in pattern recognition is that the number of available patterns for training/learning must grow with the dimension of manifolds were patterns are found. To illustrate this issue through a plain reasoning, consider a one-dimensional Real space with C classes. Lets consider, for a while, that we accept to work with only 2 training patterns per class (which can be referred to as *extreme scarcity*), which gives at least a non-null probability of occupying both positive and negative semiaxis, but are far from being statistically representative of underlying pattern distributions.

Analogously, for a two-dimensional space the equivalent extremely scarce number of training patterns per class is $2^2 = 4$. In this case, the non-null probability of observing one pattern in each quadrant, per class, is $4!/4^4 \approx 0.09$. Likewise, for patterns in three-dimensional spaces, at least $2^3 = 8$ samples per class are necessary to yield a non-null probability ($8!/8^8 \approx 0.002$) of observing one pattern in each octant. In general, the extremely scarce number of training patterns is $N = 2^L$ per class, for an L-dimensional (L-D) space, which yields a non-null probability of occupying the space with at least one pattern per L-hyperoctant of $N!/N^N \approx \sqrt{2\pi N}e^{-N}$ (de Moivre's approximation). Thus, considering all C classes, we define

$$N_{scarce} = C \times 2^L \tag{1}$$

as a generalization of the extremely scarce number of training patterns.

It is noteworthy, however, that in this definition L is the effective space dimension spanned by all observable patterns, i.e. the effective manifold dimension where pattern are expected to be found. Therefore, to avoid data scarcity, the number of training patterns would be a function of its effective manifold dimension. Unfortunately, for real-world patterns representing images and sounds, for example, manifold dimensions are typically too high, which would demand far too large training datasets. For instance, training patterns in the MNIST dataset [9] are represented as 784-D vectors (28×28 monochromatic pixels), although they lie in an approximately 14-D manifold. Likewise, training images from the CIFAR-10 dataset [10] are represented as 3072-D vectors (3 color channels, each with 32×32 pixels), although they lie in an approximately 35-D manifold.

Both manifold dimensions were estimated through the method by Farahmand et al. [11], where each image was encoded as a real-valued vector, and the estimator parameter was set to 22 near neighbours, as suggested by the authors. Moreover, about 500 randomly draw patterns were used to produce partial estimates that were averaged to yield a final dimension estimate. All dimension estimates used further on were obtained through the same method.

According to Eq. (1), even for MNIST, a training dataset of 60000 patterns is a small one, as compared to $10 \times 2^{14} = 163840$, for the corresponding 14-D manifold dimension. Indeed, experiments with data augmentation done by Wong et al. [12] emphasize the smallness of the MNIST training dataset. As for the CIFAR-10 training dataset, scarcity is even worse, for only 50000 patterns are available, whereas $10 \times 2^{35} \gg 50000$.

Similarly, by considering coarse 40-band spectral representations of half-overlapping time-windows of 20 ms, even a single second of speech signal is represented as a pattern in a 3960-D space (99 frames of 40-D spectral representations). Again, even if strong lossy compression is imposed, by taking only 12 Mel-Frequency Cepstral Coefficients (MFCC) to represent 40-band spectral contours, the single second of speech would be represented as a 1188-D pattern. Time redundancies may be removed yet, but there is no reason to expect a much smaller dimensional manifold than those found for MNIST and CIFAR-10, since speech representations through spectrograms do yield complex images as well.

Through these examples, we are revisiting the sampling aspect of the *curse of dimensionality* phenomenon, explained in pattern recognition textbooks, but we believe it also addresses a more recent controversy, namely: *data versus models*, where arguments range from references to the learning curve experiment [13] to the recent results published by Sun et al. [14]. In this last publication huge image databases are used, but even there, it would not be a surprise to realise that such datasets can still be scarce.

In this work we conjecture that data scarcity is an important factor behind the success of DNN. That is to say that if data scarcity is the main concern, either the manifold dimension or the number of training examples should be modified, and DNN do both through two complementary statistical tricks, namely:

T1: *Lossy compression trick*: keeping the number of training samples while reducing the manifold dimension.
Auto-encoders are able to find structural redundancies in raw input variables and map these variables into a less redundant space. Another simpler T1 implementation is through the subdivision of patterns into subpatterns of lower dimensionality, such as small image patches. This strategy is the one we arbitrarily chose to study in this paper, and it is further explained in Section III. By regarding the whole image as an instance of a high-dimensional random variable, features or scores extracted from image patches are then regarded as marginal statistics of the

random variable that models the image source.

T2: *Data augmentation trick*: keeping the effective dimension while artificially increasing the amount of training data. If high-dimensional patterns correspond to images taken without constraints regarding position, rotation and scale, then image patches are instances of identically distributed random variables (r.v.).

The first trick encompasses many traditional approaches in pattern recognition, such as Principal Component Analysis (PCA), Nonlinear PCA, Restricted Boltzmann Machines and Factor Analysis. By contrast, the second one has been intrinsically used through the shared weights of convolutional structures, as mentioned by Lin et al. [15], through the remark that, in lower layers of convolutional networks, the scarcity of training data is compensated by weight-sharing, "which increases the effective number of training examples (patches in that case) per weight by a factor of several thousand."

In the next Section, a KNN network is used as a straightforward structure for implementing T1 in its simplest manner, through the partial processing of small image patches whose sizes are set to compensate for data scarcity.

III. Multilayer KNN network

The KNN is a rather old-fashioned classifier, but despite its conceptual simplicity, theoretically rich studies of KNN properties have been provided since Fix and Hodges [16]. For instance, Cover and Hart [17] proved that,

> "in the large sample case, this simple rule has a probability of error which is less than twice the Bayes probability of error, and hence is less than twice the probability of error of *any other decision rule*, nonparametric or otherwise, based on the infinite sample set"

In other words, if a given classification task based on an infinite sample set is bounded by the minimum Bayesian error at, say, 1%, then a KNN is theoretically able to attain error rates lower than 2%. The other way around, if an actual classifier such as a DNN is finely tuned to attain 1%, given that it is expected to be greater than the Bayesian lower bound [17], this lower bound is less than or equal to 1%, and the theoretical proof by Cover and Hart [17] assures that a KNN can afford a competitive performance of less than or equal to 2%, in the large sample case.

This plain reasoning leads to the following analytical statement: if in any practical applications the KNN performs much worse than a DNN, that is because the large sample condition does not hold, whereas DNN excels mostly because it somehow makes better use of scarce training data. If this is true, the application of the same statistical tricks implicitly used by DNN to other classifiers should improve their performance as well, including the KNN.

To test this hypothesis, we first define a KNN cell that explicitly yields C scores instead of hard decisions. This structure can be reused in a multilayer network of KNN cells, which in turn implements T1 in its simplest way, as explained further on.

A KNN cell yields one score per class, as follows: given an instance x of a random variable X, K nearest patterns from each class are used to locally approximate the conditional probability density function (pdf) $p(x|\Omega_c)$, where Ω_c stands for the c-th class, $c \in \mathcal{C} = \{1, 2, \dots C\}$. More precisely, scores per class are given as estimates of $p(x|\Omega_c)$, as

$$\hat{p}(x|\Omega_c) = \frac{K}{\Delta_c \times N_c},$$

where Δ_c stands for an estimation of volume (or hypervolume) spanned by the K nearest neighbours of x in $\mathcal{D}_c$, and $\mathcal{D}_c$ stands for the set of training/reference samples from the c-th class. In this definition, the K nearest neighbours of a pattern x correspond to the K-*coverage* of this pattern, as defined by Zhao et al. [18].

Because patterns in our experiments are high-dimensional images (or image patches), we chose to replace distances with inner products between normalized patterns. Consequently, $\hat{p}(x|\Omega_c)$ is replaced by the average K highest inner products. Therefore, this KNN implementation maps every instance x into C values that are roughly proportional to the probability density conditioned to each class. In general, these values would be multiplied by the *a priori* probabilities of each class, $P(\Omega_c)$, and normalized to yield C scores that could be used in a straightforward approximation to the minimum error Bayesian classifier, according to the following decision rule:

$$\text{DEC}: \ \ \Omega_c \ \ \text{if} \ \ \hat{p}(x|\Omega_c)P(\Omega_c) \geq \hat{p}(x|\Omega_r)P(\Omega_r), \quad \forall r \in \mathcal{C}.$$

In our experiments with MNIST and CIFAR-10 datasets, we assume that *a priori* probabilities are always the same, so that $\hat{p}(x|\Omega_c)$ is just normalized to yield C scores, according to:

$$s_{\Omega_c}(x) = \frac{\hat{p}(x|\Omega_c)}{\sum_{c \in \mathcal{C}} \hat{p}(x|\Omega_c)}. \tag{2}$$

A bold $\mathbf{s}(x)$ is used in this text to represent a vector of C scores associated to a given pattern. Because experiments were done with either MNIST or CIFAR-10, both with 10 classes of images, C is fixed to 10 in all experiments.

This KNN classifier can be used either to classify whole images, or to extract score vectors from many (possibly overlapping) image patches. This ensemble of patch processors results in an image classifier as represented in Fig. 1, which will be referred to as the KNN network. This network is formed by many KNN cells, where each cell yields C scores for every $d \times d$ patch, based on the training samples from the corresponding position of the patch.

A fusion strategy for the pool of score vectors is necessary before a final decisions is made. This strategy was designed to be simple, as follows: each score vector, $\mathbf{s}(\mathbf{x}(i,j))$, yielded by the image patch $\mathbf{x}(i,j)$, centred at row i and column j, is first parsed to avoid null values (null values are replaced with 10^{-6} and the vector is renormalized to sum one). Then, each value is compared to the guess probability $P_0 = 1/C$ and log-transformed so that positive/negative values of $\log_2\left(\frac{\mathbf{s}(\mathbf{x}(i,j))}{P_0}\right)$ can be regarded as bits of evidence (in terms of Shannon

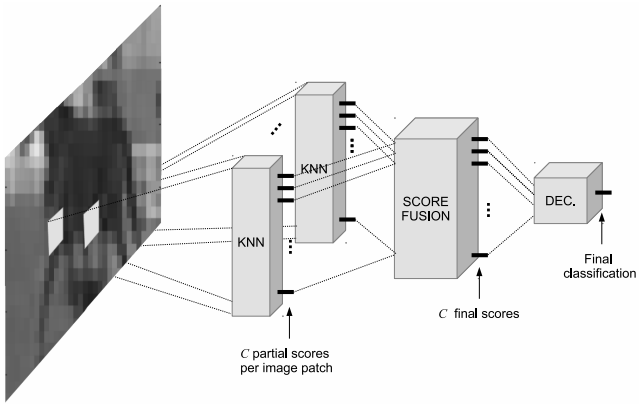

Fig. 1. KNN network illustration. Each KNN cell yields C scores for its corresponding $d \times d$ patch. All partial scores are fused to yield a final set of C scores, which is eventually used to classify the whole image.

information) in favour/against a given class. The final score vector is the sum of all resulting vectors, according to

$$\mathbf{s}_{final} = \sum_{i,j} \log_2 \frac{\mathbf{s}(\mathbf{x}(i,j))}{P_0},$$

which corresponds to the sum of all bits of information from all patches, as if they were independent sources. The final classification is done according to the index of the higher score.

As compared to neural networks, in terms of computational cost, the KNN network is spared of the burden of adaptive learning. By contrast, for an $M \times M$ image to be classified according to R pre-labelled images (training dataset), each KNN cell does about $d^2 \times R$ multiplications, resulting in an overall computational cost of about $(M-d+1)^2 \times d^2 \times R$ multiplications.

IV. Experimental evidences

Although every image to be classified can be taken as a pattern, to cope with high dimensional issues (and scarcity), patches are taken instead, whereas patch dimension, d, remains a free parameter that allows effective dimension control. Therefore, single $d \times d$ image patches, as illustrated in Fig. 2 with images from the CIFAR-10 dataset, are modelled either as d^2-D random variables, if only gray images are considered, or as $(3 \times d^2)$-D r.v., if RGB color channels are considered instead. In both cases, r.v. $\mathbf{X}(i,j)$ models a patch around row i and column j in all available images in the corresponding dataset.

The main point in this approach is the choice of the patch size in order to compensate for data scarcity. For instance, a 5×5 (i.e. $d = 5$) patch around the center of all training images in the CIFAR-10 dataset is associated to the r.v. $\mathbf{X}(16,16)$, as illustrated in Fig. 2, whose effective manifold dimension is about 15-D. Similar manifold dimensions are estimated for $\mathbf{X}(3,3)$ and $\mathbf{X}(3,16)$, what corroborates the perception that images gathered in CIFAR-10 are barely controlled for position and scale. Indeed, without pose control there is no reason to expect manifold dimension as a function of the patch position.

According to the analysis proposed in Section II, for a 15-D pattern space, the amount of labelled images for training in CIFAR-10 (50000 patterns) is critical, because it is less than $N_{scarse} = 10 \times 2^{15} = 327680$. By reducing the patch size to 3×3, manifold dimension is reduced to about 10-D, and the training dataset becomes almost 6 times greater than the corresponding critical scarcity. From this point of view, 3×3 patches would be preferable. However, shrinking the patch size has a price: some dependencies between neighbouring pixels are discarded, thus causing overall performance degradation.

As for MNIST dataset, the patch sizes whose manifold dimensions yields critical values of scarcity just below 60000 are 11×11 and 13×13, with corresponding dimensions of about 11-D and 12-D, respectively.

To study the trade-off between manifold dimension and overall performance after score fusion, three patch sizes were tested in each dataset, as presented in Table I.

TABLE I
CLASSIFICATION PERFORMANCES OF THE KNN AND THE KNN NETWORK.

Patch size	MNIST	CIFAR-10
3x3	-	54.90%
5x5	-	**58.23%**
7x7	-	56.63%
9x9	98.69%	-
11x11	**98.82%**	-
13x13	98.69%	-
KNN (full image)	97.64%	39.73%

The single KNN parameter, K, was empirically set to 3 for both MNIST and CIFAR-10 (full) images, whereas K was set to 1 in all experiments with patches. It is known that values of K higher than 1 are used in conventional KNN to regularize classification boundaries. However, the network of KNN includes a fusion of scores that also regularizes classification boundary, thus playing the role of K values greater than 1.

In Figs. 3 and 4, further experimental results are presented with patch sizes 5x5, for CIFAR-10, and 11x11 for MNIST. These results highlight the massive improvement the KNN network provides for the most difficult dataset, CIFAR-10, as compared to the single KNN applied to full images. Besides, individual patches yield weak partial classifiers with roughly similar performances. By contrast, patches in the easier MNIST database yield very irregular partial performances, strongly dependent on the patch position. Another difference with regard to CIFAR-10 is that, with MNIST, the fusion of patch scores yields a rather subtle accuracy improvement, as compared to the KNN applied to the full image. These results are taken as evidences that trick T1 also works in the convolution-like structure of this KNN network.

V. Discussion

The classifier used in this work is an ensemble of KNN working on parts of a pattern (image patches), therefore it is conceptually equivalent to many existing approaches to image

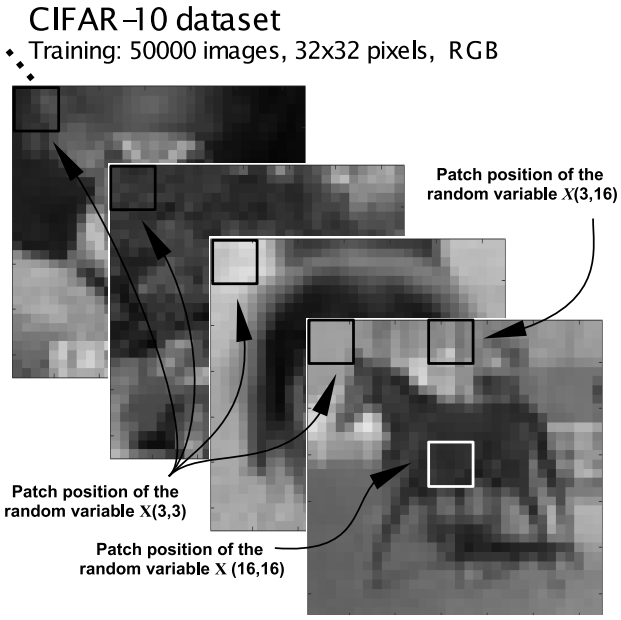

Fig. 2. Illustration of the correspondence between random variables and image patches in CIFAR-10 dataset.

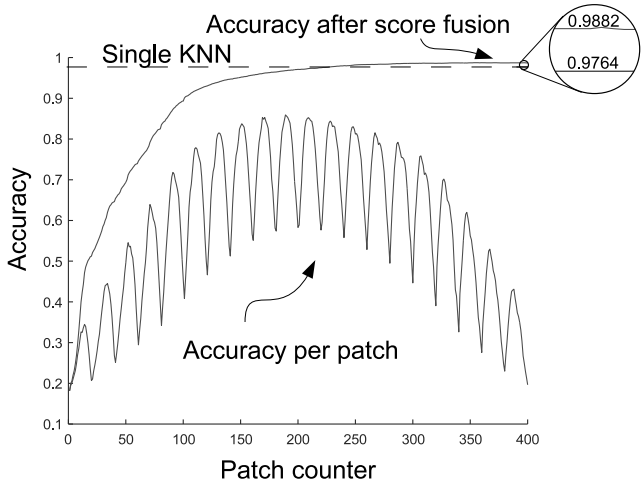

Fig. 3. Accuracy evolution in terms of added evidences from patches in the MNIST datase. Individual accuracies associated to each patch are also presented for comparison, as well the performance of a single KNN applied to full images.

classification. The novelty of this work, to the extent of the authors knowledge, is to use this kind of structure as a tool to study the role of the fundamental trade-off between intrinsic dimension of data and number of available observation, and how it can explain, in part, the good results obtained with DNN-like structure. Thus, we highlight that outperforming state-of-the-art classifiers is not the goal of this work.

In terms of decisions explainability, an advantage of using KNN comes from the fact that training patterns are explicitly used in its structure. Therefore, unlike DNN, where these patterns are somehow encoded in the neuronal connections, every KNN in the network can be *interrogated* regarding which training patterns were the most important for their partial contribution (in terms of scores).

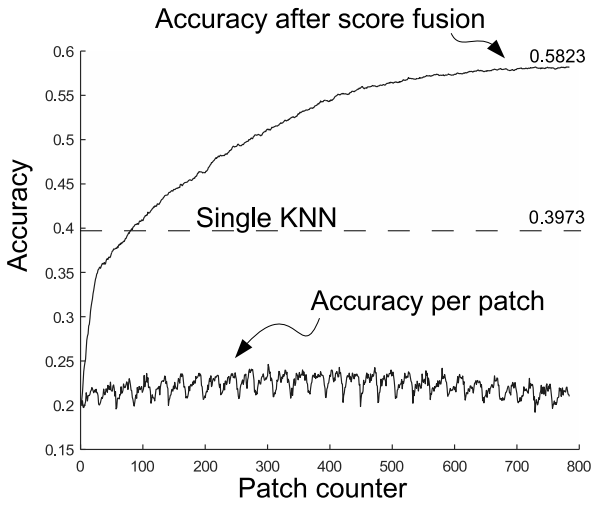

Fig. 4. Accuracy evolution in terms of added evidences from patches in the CIFAR-10 datase. Individual accuracies associated to each patch are also presented for comparison, as well the performance of a single KNN applied to full images.

This is the most straightforward manner to address the problem, but we believe that many more elaborated approaches for explained decisions can be easily deployed. To give a simple example, Fig. 5 illustrates how the accumulated scores can be used to explain the classifier decision. For this illustration, we created an artificial image from two samples from the CIFAR-10 test database, namely: image 106 (labelled automobile) and image 217 (labelled horse). Evidences in favour of class *horse* rises consistently until about patch 252. Given the indicated progression of patches over the image, it can be noticed that the change in evidence accumulation trend roughly corresponds to the edition boundary between the two original images.

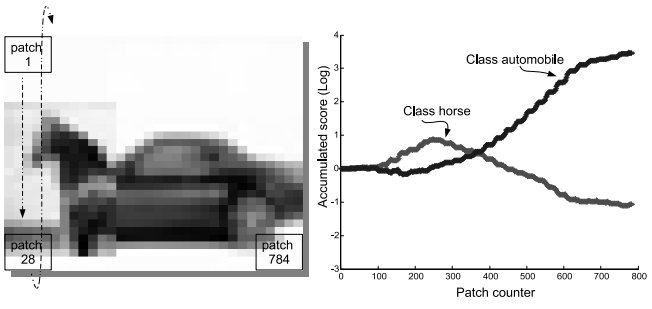

Fig. 5. Illustration of how the accumulated scores can be used to explain the classifier decision. The image results from a combination of images 106 (labelled automobile) and 217 (labelled horse) in the CIFAR-10 test database. Evidences in favour of class *horse* rise consistently until about patch 252. Afterwards, a clear rising of evidences in favour of class *automobile* dominates until the last patch.

If KNN based strategies could be improved to state-of-the-art performances obtained with DNN, it would yield a potentially advantageous alternative to DNN, mainly in critical applications involving human lives, such as in healthcare, military devices and self-driven cars. In this first attempt, we were able to isolate at least one strategy to compensate for data

scarcity in DNN, and successfully transfer it to a network of KNN cells.

Interestingly, accuracies of 98.82% and 58.23%, for MNIST and CIFAR-10, respectively, are comparable to results reported in publications during the early days of the deep learning era for the same databases. For instance, as reported by Bengio et al. [1], for the MNIST test set, a DNN with layers initialized as auto-encoders (prior to back-propagation final adjustments) yielded error rate of 1.6%, whereas another DNN with supervised greedy layer-wise algorithm to pre-train each layer yielded error rate of 1.9%.

Likewise, some DNN used by Krizhevsky and Hinton [10] to classify images from the test set of CIFAR-10 yielded accuracies below the 58.23% attained by the KNN network. More recently, Lin et al. [15] studied the limits of deep neural nets without convolution, and according to their results, we can conclude that the KNN network outperforms at least four ReLU[1] based DNN with different configurations, including the ReLU-Lin network, whose structure (represented by the authors as 4000ReLU-1000Linear-4000ReLU) interleaves a linear layer between two ReLU layers, all trained with supervised back propagation using stochastic gradient descent with momentum. It is noteworthy that their ReLU-Lin network outperforms all the pure ReLU networks, reaching an accuracy of 56.84%.

Lin et al. [15] further report that performances without convolution are boosted when they give up on permutation-invariance (by using data augmentation), thus attaining 78.62% with the same structure. This indirectly corroborates the ideas pursued in this paper concerning the importance of data scarcity in DNN scenarios. Besides, it suggests that the KNN network can also be improved through data augmentation.

We highlight that the single statistical trick transferred to the KNN network was the raw dimensional reduction imposed by the use of image patches. Further tricks are to be studied in the sequel of this work, including another important lesson learned from DNN, namely, the implicit data augmentation in convolutional strategy. For now, the convolution-like structure in Fig. 1 is just partially exploited, since it has no equivalent to the weight sharing of Convolutional Neural Networks, what would correspond to trick T2.

Another important lesson not taken into account in the KNN network is the hierarchical combination of information from patches. The KNN network is *naive* in the sense that patches are independently processed, and evidences are fused as if score vectors were statistically independent. It is noteworthy that shrinking patch size to fit scarcity requirements has a cost, which is the degradation of classification performance due to the loss of dependence information between pixels from different patches. The current implementation of the KNN network discards this information. Thus, a potentially relevant improvement to the KNN network is the inclusion of some hierarchical recombination of information, as in DNN, but it is beyond the purpose of this work.

[1]ReLU is an acronym for Rectified Linear Unit, a kind of activation function used in artificial neurons.

VI. Conclusion and perspectives

In this paper, we conjecture that what makes DNN consistently outperform other classifiers are the many ways it tackles data scarcity. To test it, we first define a critical value of scarcity which is dependent of the dimension of the manifold spanned by the available training patterns. We further propose the categorization of relevant strategies to cope with data scarcity into either dimension reduction or data augmentation. The first encompasses many traditional pattern classification tools, including auto-encoders and projections such as image patches, whereas the second can be done explicitly (e.g. inclusion of translated/rotated/distorted images into the database, as in [19]), or implicitly through convolutional strategies, as pointed out by [15], concerning shared weights in CNN.

Processing separately image patches can be regarded as the simplest dimension reduction strategy, conceptually equivalent to using marginal statistics of the random variable that models the image source. We arbitrarily chose to isolate this effect in a KNN structure, yielding a simple network of KNN cells. The choice of a KNN based approach was also motivated by the explainability issue in DNN. Indeed, KNN makes decisions based on stored patterns, what allows for a myriad of straightforward strategies to trace back and understand decisions. Moreover, KNN is suitable in cases were it is advantageous to trade the training period for memory. Another potentially desirable consequence is that classification results are not dependent on arbitrary initialization of learning parameters choices.

The KNN network provided results that, mainly for the CIFAR-10 dataset, confirmed the expected performance gain, due to a careful choice of the manifold dimension corresponding to patches in each dataset. We recall that, according to Cover and Hart [17], when data is abundant, we should not expect an attractive advantage of DNN over KNN. Purposely, we also remark that, for the MNIST dataset, whose scarcity is moderate, the performance of a simple KNN is rather competitive, whereas for CIFAR-10, whose scarcity is much bigger, KNN yields a very modest accuracy. These two results, added to the significant improvement of performance with the KNN network on the CIFAR-10 dataset corroborate the idea that the main hinging factor for limited performances of usual KNN is data scarcity.

This work is a step towards the understanding of theoretical principles that give power to DNN, and building alternative structures using these principles. In this first attempt, only one of the principles was isolated, and we believe that the next significant performance factors to be studied are implicit data augmentation and hierarchical recombination of information, as discussed in the former section. We believe that, concerning the pattern recognition field, diversity of methods (with their different strengths and weakness) is a healthy goal in itself.

Acknowledgment

This work was supported by grant from the CNPq to J.M. (grant 304853/2015-1).

REFERENCES

[1] Y. Bengio, P. Lamblin, D. Popovici and H. Larochelle, "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, 2007, pp. 153–160.

[2] H. W. Lin, M. Tegmark and D. Rolnick, "Why does deep and cheap learning work so well?" *Journal of Statistical Physics*, Springer, 2016, pp. 1–25.

[3] G. F. Montufar, R. Pascanu, K. Cho and Y. Bengio, "On the number of linear regions of deep neural networks," *Advances in neural information processing systems*, 2014, pp. 2924–2932.

[4] Y. LeCun, "Generalization and network design strategies," *Connectionism in perspective*, 1989, pp. 143–155.

[5] G. E. Hinton, "To recognize shapes, first learn to generate images," *Progress in brain research*, vol. 165, 2007, pp. 535–547.

[6] H. Larochelle, D. Erhan, A. Courville, J. Bergstra and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," *In Proceedings of the 24th international conference on Machine learning*, 2007, pp. 473–480.

[7] R. Salakhutdinov and G. E. Hinton, "Deep boltzmann machines," *Artificial Intelligence and Statistics.*, 2009, pp. 448–455.

[8] R. Andrews, J. Diederich and A. B. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks," *Knowledge-based systems*, Elsevier, vol. 8, no. 4, 1995, pp. 373–389.

[9] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, 1998, pp. 2278–2324.

[10] A. Krizhevsky and G. E. Hinton, "Learning multiple layers of features from tiny images," *Computer Science Department, University of Toronto*, Tech. Rep., 2009.

[11] A. M. Farahmand, C. Szepesvári and J.-Y. Audibert, "Manifold-adaptive dimension estimation," *Proceedings of the 24th international conference on Machine learning (ICML '07), Zoubin Ghahramani (Ed.).*, 2007, pp. 265–272.

[12] S. C. Wong, A. Gatt, V. Stamatescu and M. D. McDonnell, "Understanding data augmentation for classification: when to warp?" *Digital Image Computing: Techniques and Applications (DICTA), 2016 IEEE International Conference on*, 2016, pp. 1–6.

[13] M. Banko and E. Brill, "Scaling to very very large corpora for natural language disambiguation," *In Proceedings of the 39th annual meeting on association for computational linguistics, Association for Computational Linguistics*, 2001, pp. 26–33.

[14] C. Sun, A. Shrivastava, S. Singh and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," *arXiv preprint arXiv:1707.02968*, 2017, pp. 1–13.

[15] Z. Lin, R. Memisevic and K. Konda, "How far can we go without convolution: Improving fully-connected networks," *arXiv preprint arXiv:1511.02580*, 2015, pp. 1–10.

[16] E. Fix and J. L. Hodges Jr., "Discriminatory analysis-nonparametric discrimination: consistency properties," *California Univ Berkeley*, 1951, pp. 1–21.

[17] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, 1967, pp. 21–27.

[18] K.-P. Zhao, S.-G. Zhou, J.-H. Guan and A.-Y. Zhou, "C-pruner: an improved instance pruning algorithm," *In Machine Learning and Cybernetics, 2003 IEEE International Conference on*, vol. 1, 2003, pp. 94–99.

[19] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *In Advances in neural information processing systems*, 2012, pp. 1097–1105.

Jugurta Montalvão was born in Aracaju, Brazil, in 1968. He received the title of Electrical Engineer (1992) from the University of Campina Grande (UFPB II), Master in Electrical Engineering (1995) from the University of Campinas (UNICAMP) and Doctor in "Automatique et traitement du signal" (2000) from the University Paris-Sud XI. He joined the Department of Electrical Engineering of the Federal University of Sergipe (UFS) in 2005. His main research interests are: pattern recognition and signal processing.

Jânio Canuto was born in Maceió, Brazil, in 1984. He received the title of Electrical Engineer (2007) from the Federal University of Sergipe (UFS), M.Sc. in Electrical Engineering (2010) from the State University of Campinas (UNICAMP) and Ph.D. in Computer Science (2014) from Télécom SudParis. He is currently a postdoc fellow at the Department of Computer Science of the Federal University of Sergipe (UFS). His main research interests are: pattern recognition and signal processing.

Elyson Carvalho was born in Arapiraca, Alagoas, Brazil, in 1985. He received the title of Electrical Engineer (2006) from the Federal University of Sergipe (UFS), M.Sc. in Electrical Engineering (2007) from the Federal University of Campina Grande (UFCG) and Ph.D. in Electrical Engineering (2012) from the Federal University of Campina Grande (UFCG). He joined the Department of Electrical Engineering of the Federal University of Sergipe (UFS) in 2010. His main research interests are: robotics, nonlinear systems, instrumentation and signal processing.