A Qualitative Evaluation of Random Forest Feature Learning

Adelina Tang and Joan Tack Foong

Sunway University, Dept. of Computer Science & Networked Systems, No. 5 Jalan Universiti, Bandar Sunway, 46150 Petaling Jaya, Selangor Darul Ehsan, Malaysia
adelina.tang@ieee.org, 12058590@imail.sunway.edu.my

Abstract. Feature learning is a hot trend in the machine learning community now. Using a random forest in feature learning is a relatively unexplored area compared to its application in classification and regression. This paper aims to show the characteristics of the features learned by a random forest and its connections with other methods.

1 Introduction

1.1 Problem and Motivation

Feature learning has been a hot trend in the machine learning community. It is mainly due to the success of deep learning in traditional machine learning tasks [1] and real world application such as MAVIS (Microsoft Audio Video Indexing Service) [2]. Deep learning itself is the attempt to construct multiple layers of feature representation in such a way that higher level abstractions can be represented.

A feature contributes enormously to the success of machine learning task because it is the input of machine learning algorithms and the only thing they see. Features used to be hand engineered by domain experts to reflect their knowledge of the critical aspects about a particular problem. However, as the problem becomes more complicated, we hope that the machine can take the role of the domain experts and be able to extract most relevant features from the raw data.

1.2 Random Forest as Feature Learning Technique

In this paper, we are going to explore feature learning using random forests [3]. A random forest is an ensemble method that gives good results in classification and regression. However the random forest itself is a much richer structure than can be merely used in these two settings. Criminisi gives a nice overview of using random forest in density estimation, manifold learning, and semi-supervised learning [4].

This paper focuses on the feature learning aspect of the random forest. By analysing the reconstruction of the original data using the learned future, we hope to gain some insight on how it works. Finally, we will discuss briefly its connection with sparse coding [5] and self-taught learning [6].

T. Herawan et al. (eds.), *Recent Advances on Soft Computing and Data Mining SCDM 2014*, Advances in Intelligent Systems and Computing 287,
DOI: 10.1007/978-3-319-07692-8_34,

2 Literature Review

Deep Learning and Representation Learning. Deep learning is the attempt of learning multiple layers of representation, where the higher level representation is the composition of its lower level counterparts [7].

The first breakthrough of deep learning is the success of deep belief nets [8] in the MNIST [9] digit recognition problem. The state-of-the-art result was long held by the Support Vector Machine (SVM). A more recent breakthrough is achieved in the ImageNet dataset, which achieves 15.3% error rate, lower than the state-of-the-art 26.1% [1]. MAVIS (Microsoft Audio Video Indexing Service) speech system, released in 2012, is based on deep learning [2] as well.

Traditional deep learning has been focusing on various type of neural network such as deep belief net [8], autoencoder [10, 11], Restricted Boltzmann Machine [12], and sparse coding [5, 13]. However, as observed in [7], the ensemble of trees such as boosted trees and random forests can be viewed as a three-level deep architecture. What interests us is not that the ensemble serves as a classifier, but that the outputs from all the trees in the ensemble form a distributed representation [14, 15] of the training data. As the exact form of the representation will be spelled out explicitly in the later part of this article, it suffices now to note that the representation provides a very rich description of the input data in the sense that the number of output patterns it can discriminate is exponential to the number of its parameters [16].

Despite all the good properties mentioned above, only two papers are dedicated to this effort [17, 18]. It is the intention of this article to further investigate the properties of this representation and its application in classification.

Ensemble of Decision Trees. Leo Breiman published his seminal book "Classification and Regression Trees (CART)" [19] in 1993, in which he described the fundamental principles in using decision trees for both classification and regression and paved the way for future research. In the same year, JR Quinlan published one of the most popular tree constructing algorithms "C4.5" in his book "C4.5: Programs for machine learning" [20].

Ensemble methods are ways to combine various weak learners in order to get better result. The idea of combining the strengths of many decision trees is not new. Amit and Geman introduced the use of random generated node tests in constructing many decision trees for handwritten digit recognition in their papers [21, 22] published in 1994 and 1997. The term "Random Decision Forest" was introduced by Ho in his paper [23], in which he used the random partition of the feature space to build the trees.

However, the random forest[1] only began to gain serious attention after Leo Breiman published his seminal paper [3] in 2001. He laid the theoretical framework for random forest and introduced a new way of constructing the decision trees by combing his earlier work in "bagging" [24] and Ho's method.

Random forests and their variants enjoy much success in the fields of machine learning, computer vision and medical imaging [25–29]. In this paper, however,

[1] Random forest is the trademark of Leo Breiman.

we are going to explore the potential of using the random forest as a feature learning algorithm.

3 Methodology

3.1 The Basic of Decision Trees

Decision tree can be regarded as the *partitions of the feature space*. Each node in the decision tree ask a question about the features. The feature space is then split into regions which have distinct answers to the question. Fig. 1 illustrates the splitting process.

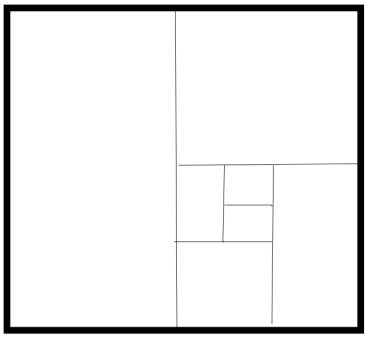

Fig. 1. The decision tree gives rise to the partition of the feature space

3.2 Interpretation of the Partitions

In the common setting of machine learning task, the input data is of the form $[x_i]_{i=1}^n$, where $x_i = (x_i^{(1)}, x_i^{(2)}, ..., x_i^{(m)}) \in \mathbb{R}^m$ is a vector of real number. x_i is reffered as the *data point*, and its components,$(x_i^{(1)}, x_i^{(2)}, ..., x_i^{(m)}$, are referred as *features*. Each node in a decision tree asks a question about the feature, and each distinct answer splits the feature space into corresponding subspaces. Thus each partition in the feature space, and hence each terminal node corresponds to a different configuration and combination of the features. If we consider a feature of a data point as a property that characterizes the data point, then any combination of features can also be regarded as a feature, albeit, a high level feature. Certainly, this high level feature cannot be represented as a real number. However, we can abstract away the detail which is the exact configuration of low level features that correspond to the high level feature, and simply assign each high level feature a terminal node or a distinct symbol. In other words, the decision tree is able to transform the representation of the data point from its standard form $(x_i^{(1)}, x_i^{(2)}, ..., x_i^{(m)})$ to a symbol, say, d (see Fig.2).

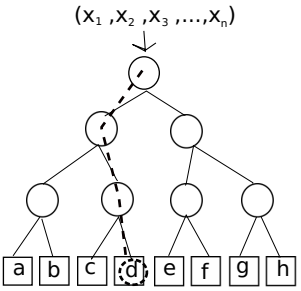

Fig. 2. This diagram shows how a data point is transformed into a symbol

The induced representation of x by $\mathcal{F}$, $\mathcal{F}(x)$ is defined as:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}T_i^{(j)}$$

where $a_{ij} = 1$ if T_i assign the node $T_i^{(j)}$ and $a_{ij} = 0$ otherwise. Note that the summation is purely formal, after all the nodes of decision trees cannot be added, at least not in the usual way. It might just as well be written as a normal vector (a_{ij}). It will be clear in a later section why we choose this notation over a conventional one.

To show that the notation is useful, we use it to introduce an important concept introduced by Breiman: *proximity* [3]. First we have to define a "norm"[2] $||$ for a formal summation of the form $w = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}T_i^{(j)}$ as $|w| = \frac{|a_{ij}|}{n}$.

Given a random forest $\mathcal{F} = \{T_i\}_{i=1}^{n}$ and two data points x and y, the proximity of these two points with respect to $\mathcal{F}$ is the number of identical symbols between the data points divided by n.

Now suppose

$$\mathcal{F}(x) = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}T_i^{(j)}, \mathcal{F}(y) = \sum_{i=1}^{n}\sum_{j=1}^{m} b_{ij}T_i^{(j)}$$

then

$$\begin{aligned} |\mathcal{F}(x) - \mathcal{F}(y)| &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(a_{ij} - b_{ij})T_i^{(j)}}{n} \\ &= \frac{\text{number of different symbols}}{n} \\ &= 1 - \frac{\text{number of identical symbols}}{n} \\ &= 1 - \text{proximity of x and y} \end{aligned}$$

The derivation above shows the natural connection between the "norm" that we defined and the concept of proximity.

[2] Not a norm in the strict mathematical sense.

3.3 Reconstruction of Image

In this section, we are going to show how to reconstruct a binary image from the features induced by the random forest. Suppose an image is represented by a vector of its pixel intensities, i. Given a random forest $\mathcal{F}$, the image can be represented as $\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}T_i^{(j)}$ as shown above. In this case, $T_i^{(j)}$ tells us, partially, which pixel is on and which pixel is off. Thus $T_i^{(j)}$ can be regarded as a vector which captures a certain property of the original image. Now the formal sum above can actually be calculated, and the value will be the reconstructed image.

3.4 Tree Building Algorithm

In this paper, we follow closely the algorithm known as *Extremely Randomized Forest* [30]. First of all, a feature,x_i, and a threshold,θ, are chosen randomly. Then the feature space is split into two parts, i.e. $x_i \leq \theta$ and $x_i \geq \theta$. A *score* for this particular split is then calculated. If the score is greater than a predetermined value, repeat the process on the subspaces. although there are many ways to calculate the score of a particular split, the one we are using here is the information gain.

4 Results

The result in this section shows the general properties of the learned representation using the MNIST dataset [9].

Fig. 3 shows the first 15 digits from the dataset.

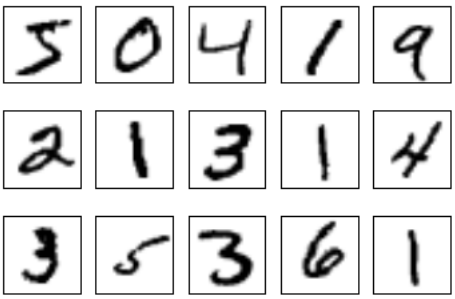

Fig. 3. First 15 digits from the train dataset

A random forest consisting of 30 trees is trained using randomly generated data. To be precise, the data used here consists of 50,000 vectors of dimension 784×1, drawn from random uniform distribution. There is no relationship at all with the MNIST dataset. However, it is possible to use this random forest to

transform the MNIST dataset into a new representation. The left diagram in Fig. 4 shows the reconstruction of the first 15 digits using the new representation.

For comparison, another random forest is trained using 50,000 digits from the dataset. However, unlike the case of using the random forest in classification, a random label is given for each digit. As shown by the comparison in Fig. 4, the reconstructed digits are more visually recognizable.

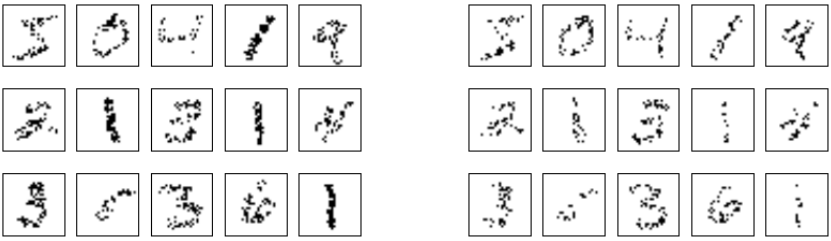

Fig. 4. The left diagram shows the reconstruction using the MNIST dataset, and the right reconstruction using random data

To show the individual contribution of the trees inside the random forest, here is the progressive reconstruction of the digit "5". The diagram is to be read from left to right and from top down. The first image shows the reconstruction using only the first tree; the second image uses the first and second; and the final one uses all of the trees.

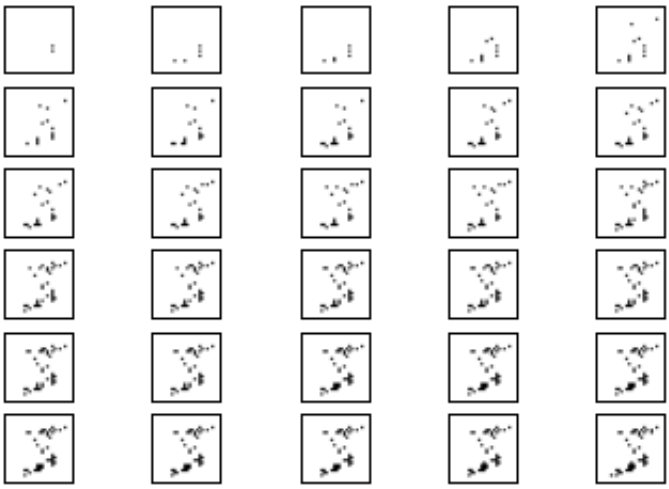

Fig. 5. Progressive reconstruction using random forest trained with random data

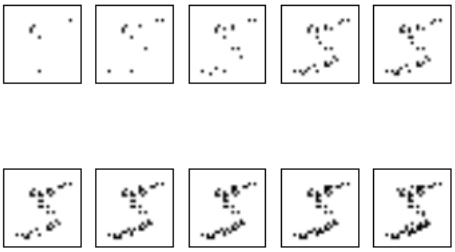

Fig. 6. Progressive reconstruction using random forest trained with original dataset

5 Discussion

In [31], the authors show empirically that the power of sparse coding , as a feature learning technique, is not the learned basis functions but rather the non-linear coding scheme. It corresponds to the facts showed in the paper that the data used to train the random forest is not that important. Instead of justifying our claim using classification accuracy, we choose to reconstruct the images using learned representations. The visual similarity between original images and reconstructed images gives us better intuition.

As shown in Fig. 2, each data point falls to a terminal node through a series of split nodes. Each split node dictates the pixel value of a particular point. Thus a terminal node represents a certain configuration of the pixels. The typical configuration is shown in the top leftmost image in Figs. 5 and 6. It could be just a few points arranged in a particular order, but as they layer up on each other, the digit take its shape gradually (see Figs. 5 and 6).

Take note that the random forest $\mathcal{F}$ can be trained on one set of data X, and yet it can be used in constructing the representation of the data point from another set of data Y. X and Y can have no relation at all, with the exception that their data points must have the same dimensions. In fact, X can be totally random data. As shown in the comparison in Fig. 4, the random forest trained on random data can nevertheless represent the basic shapes of the digits as well as the random forest trained on the digits dataset. The notable difference here is that the pixel density is lower for the digits reconstructed using the random forest trained on random data. The idea of training a learner, using different data from the one on which it eventually applies, is explored in the paper [6], in which the authors coined the term *self-taught learning* as an alternative to the other learning paradigms such as supervised learning, unsupervised learning, transfer learning, and reinforcement learning.

Motivated by this observation, we propose another interpretation of the formal summation[3] $\sum a_{ij} T_i^{(j)}$. Given that data point x is in the form of $(x^{(i)})_{i=1}^n$ and the $T_i^{(j)}$ specifies the values of a certain subset of the features, say $(x^{(k_i)})_{i=1}^m$,

[3] Abbreviated form of $\sum_{i=1}^n \sum_{j=1}^m a_{ij} T_i^{(j)}$.

then $T_i^{(j)}$ can be represented as $(v_i)_{i=1}^n$ where $v_i = x^{(k_j)}$ if $i = k_j$ else $v_i = 0$. We can now say $\sum a_{ij} T_i^{(j)}$ approximates the data point x, that is $x \approx \sum a_{ij} T_i^{(j)}$. Observe that most of the a_{ij} that is zero for each data point is assigned with a single terminal node, and in general there are 2^d terminal nodes for a binary tree with depth d. Suppose n trees in a forest have the same depth, then the ratio of non-zero coefficients in the sum $\sum a_{ij} T_i^{(j)}$ is

$$\frac{(n \times 1)}{(n \times 2^d)} = \frac{1}{2^d} \to 0 \text{ as } d \to \infty$$

In other words, the representation induced by the random forest is very *sparse*. On the other hand, sparse coding [5] is the method of representing a data point x in the form of $\sum a_i T_i$ that minimizes

1. the difference $|x - \sum a_i T_i|$
2. the sum of coefficients $|\sum a_i|$

The second constraint encourages the coefficients to have as many zeros as possible, thus the term *sparse*. Notice the similarity between sparse coding and the method outlined in this paper although the method here achieves sparsity, but not by direct optimization.

The possibility of using a different dataset in the training phase is a plus point for this method. Hence, we can use a combination of a large number of seemingly unrelated datasets to train the learner and then apply the learner to yet another dataset of interest. The more data that we can feed into a learner, the better its performance. The method in this paper can exploit existing patterns as it can learn from unrelated datasets.

In conclusion, the method outlined here shows basic learning capacity and shares a lot of interesting connections with other methods too. The future work will be focused on the elaboration of the connections as well as the application of the learned feature(s) in the classification task.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25, pp. 1106–1114 (2012)
2. Seide, F., Li, G., Yu, D.: Conversational speech transcription using context-dependent deep neural networks. In: Proc. Interspeech, vol. 11, pp. 437–440 (2011)
3. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
4. Criminisi, A.: Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. Foundations and Trends in Computer Graphics and Vision 7(2-3), 81–227 (2011)
5. Lee, H., et al.: Efficient sparse coding algorithms. In: Advances in Neural Information Processing Systems, pp. 801–808 (2006)
6. Raina, R., et al.: Self-taught learning: Transfer learning from unlabeled data. In: Proceedings of the Twenty-fourth International Conference on Machine Learning (2007)

7. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning 2(1), 1–127 (2009)
8. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18(7), 1527–1554 (2006)
9. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition Proceedings of the IEEE, Vol. Proceedings of the IEEE 86(11), 2278–2324 (1998)
10. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. Tech. rep. DTIC Document (1985)
11. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
12. Salakhutdinov, R., Mnih, A., Hinton, G.: Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th International Conference on Machine Learning, pp. 791–798. ACM (2007)
13. Olshausen, B.A., Field, D.J.: Sparse coding with an overcomplete basis set: A strategy employed by V1? Vision Research 37(23), 3311–3325 (1997)
14. Hinton, G.E.: Learning distributed representations of concepts. In: Proceedings of the Eighth Annual Conference of the Cognitive Science Society, Amherst, MA, pp. 1–12 (1986)
15. Bengio, Y., et al.: Neural probabilistic language models. In: Holmes, D.E., Jain, L.C. (eds.) Innovations in Machine Learning. STUDFUZZ, vol. 194, pp. 137–186. Springer, Heidelberg (2006)
16. Bengio, Y., Delalleau, O., Simard, C.: Decision trees do not generalize to new variations. Computational Intelligence 26(4), 449–467 (2010)
17. Moosmann, F., Nowak, E., Jurie, F.: Randomized clustering forests for image classification. Pattern Analysis and Machine Intelligence 30(9), 1632–1646 (2008)
18. Vens, C., Costa, F.: Random Forest Based Feature Induction. In: 2011 IEEE 11th International Conference on Data Mining (ICDM), pp. 744–753. IEEE (2011)
19. Breiman, L.: Classification and regression trees. CRC Press (1993)
20. Quinlan, J.R.: C4. 5: Programs for machine learning, vol. 1. Morgan Kaufmann (1993)
21. Amit, Y., Geman, D.: Randomized Inquiries About Shape: An Application to Handwritten Digit Recognition. Tech. rep. DTIC Document (1994)
22. Amit, Y., Geman, D.: Shape quantization and recognition with randomized trees. Neural Computation 9(7), 1545–1588 (1997)
23. Ho, T.K.: Random decision forests. In: Proceedings of the Third International Conference on Document Analysis and Recognition, vol. 1, pp. 278–282. IEEE (1995)
24. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
25. Bosch, A., Zisserman, A., Muoz, X.: Image classification using random forests and ferns. In: IEEE 11th International Conference on Computer Vision, ICCV 2007, pp. 1–8. IEEE (2007)
26. Criminisi, A., Shotton, J., Robertson, D., Konukoglu, E.: Regression forests for efficient anatomy detection and localization in CT studies. In: Menze, B., Langs, G., Tu, Z., Criminisi, A., et al. (eds.) MICCAI 2010. LNCS, vol. 6533, pp. 106–117. Springer, Heidelberg (2011)
27. Fanelli, G., Gall, J., Van Gool, L.: Real time head pose estimation with random regression forests. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 617–624. IEEE (2011)

28. Geremia, E., Menze, B.H., Clatz, O., Konukoglu, E., Criminisi, A., Ayache, N.: Spatial decision forests for MS lesion segmentation in multi-channel MR images. In: Jiang, T., Navab, N., Pluim, J.P.W., Viergever, M.A., et al. (eds.) MICCAI 2010, Part I. LNCS, vol. 6361, pp. 111–118. Springer, Heidelberg (2010)
29. Leistner, C., et al.: Semi-supervised random forests. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 506–513. IEEE (2009)
30. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning 63(1), 3–42 (2006)
31. Coates, A., Ng, A.: The Importance of Encoding Versus Training with Sparse Coding and Vector Quantization. In: Proceedings of the 28th International Conference on Machine Learning, pp. 921–928 (2011)