ARTICLE IN PRESS
JID: KNOSYS [m5G;October 9, 2017;19:5]
Knowledge-Based Systems 000 (2017) 1–10
Contents lists available at ScienceDirect
Knowledge-Based Systems
journal homepage: www.elsevier.com/locate/knosys

A Siamese Deep Forest

Lev V. Utkin*, Mikhail A. Ryabinin

Department of Telematics, Peter the Great St.Petersburg Polytechnic University, St.Petersburg, Russia

ARTICLE INFO

Article history:
Received 28 May 2017
Revised 23 September 2017
Accepted 4 October 2017
Available online xxx

Keywords:
Classification
Random forest
Decision tree
Siamese
Deep learning
Metric learning
Quadratic optimization

ABSTRACT

A Siamese Deep Forest (SDF) is proposed in the paper. It is based on the Deep Forest or gcForest proposed by Zhou and Feng and can be viewed as a gcForest modification. It can be also regarded as an alternative to the well-known Siamese neural networks. The SDF uses a modified training set consisting of concatenated pairs of vectors. Moreover, it defines the class distributions in the deep forest as the weighted sum of the tree class probabilities such that the weights are determined in order to reduce distances between similar pairs and to increase them between dissimilar points. We show that the weights can be obtained by solving a quadratic optimization problem. The SDF aims to prevent overfitting which takes place in neural networks when only limited training data are available. The numerical experiments illustrate the proposed distance metric method.

1. Introduction

One of the important machine learning tasks is to compare pairs of objects, for example, pairs of images, pairs of data vectors, etc. There are a lot of approaches for solving the task. One of the approaches is based on computing a corresponding pairwise metric function which measures a distance between data vectors or a similarity between the vectors. This approach is called the metric learning [2,17,30]. It is pointed out by Bellet et al. [2] in their review paper that the metric learning aims to adapt the pairwise real-valued metric function, for example, the Mahalanobis distance or the Euclidean distance, to a problem of interest using the information provided by training data. A detailed description of the metric learning approaches is also represented by Le Capitaine [6] and by Kulis [17]. The basic idea underlying the metric learning solution is that the distance between similar objects should be smaller than the distance between different objects.

There are many approaches and methods which take into account the above condition. One of the most important and popular approaches is to use the Mahalanobis distance as a distance metric which assumes some linear structure of data. However, this assumption significantly restricts the applicability of the Mahalanobis distance for comparing pairs of objects. Therefore, in order to overcome this restriction, the kernelization of linear methods is one of the possible ways for solving the metric learning problem. Bellet et al. [2] review several approaches and algorithms to deal with nonlinear forms of metrics. In particular, these are the Support Vector Metric Learning algorithm provided by Xu et al. [29], the Gradient-Boosted Large Margin Nearest Neighbors method proposed by Kedem et al. [14], the Hamming Distance Metric Learning algorithm provided by Norouzi et al. [22].

A powerful implementation of the metric learning dealing with non-linear data structures is the so-called Siamese neural network (SNN) introduced by Bromley et al. [5] in order to solve signature verification as a problem of image matching. This network consists of two identical sub-networks joined at their outputs. The two sub-networks extract features from two input examples during training, while the joining neuron measures the distance between the two feature vectors. The Siamese architecture has been exploited in many applications, for example, in face verification [8], in the one-shot learning in which predictions are made given only a single example of each new class [15], in constructing an inertial gesture classification [3], in deep learning [26], in extracting speaker-specific information [7], for face verification in the wild [13]. This is only a part of successful applications of SNNs. Many modifications of SNNs have been developed, including fully-convolutional SNNs [4], SNNs combined with a gradient boosting classifier [18], SNNs with the triangular similarity metric [30].

One of the difficulties of the SNN as well as other neural networks is that limited training data lead to overfitting when training neural networks. Many different methods have been developed to prevent overfitting, for example, dropout methods [24] which are based on combination of the results of different networks by randomly dropping out neurons in the network. A very interesting

* Corresponding author.
E-mail addresses: lev.utkin@list.ru (L.V. Utkin), mihail-ryabinin@yandex.ru (M.A. Ryabinin).

https://doi.org/10.1016/j.knosys.2017.10.006
0950-7051/

new method which can be regarded as an alternative to deep neural networks is the deep forest proposed by Zhou and Feng [31] and called gcForest. In fact, this is a multi-layer structure where each layer contains many random forests, i.e., this is an ensemble of decision tree ensembles. Zhou and Feng [31] point out that their approach is highly competitive to deep neural networks. In contrast to deep neural networks which require great effort in hyperparameter tuning and large-scale training data, gcForest is much easier to train and can perfectly work when there are only small-scale training data. The deep forest solves tasks of classification as well as regression. Therefore, by taking into account its advantages, it is important to modify it in order to develop a structure solving the metric learning task. We propose the so-called Siamese Deep Forest (SDF) which can be regarded as an alternative to the SNNs and which is based on gcForest proposed by Zhou and Feng [31] and can be viewed as its modification. Three main ideas underlying the SDF can be formulated as follows:

1. We propose to modify training set by using concatenated pairs of vectors.
2. We define the class distributions in the deep forest as the weighted sum of the tree class probabilities where the weights are determined in order to reduce distances between semantically similar pairs of examples and to increase them between dissimilar pairs. The weights are training parameters of the SDF.
3. We apply the greedy algorithm for training the SDF, i.e., the weights are successively computed for every layer or level of the forest cascade.

We consider the case of the weakly supervised learning [2] when there are no information about the class labels of individual training examples, but only information in the form of sets of semantically similar or dissimilar pairs of training data is provided, i.e., we know only semantic similarity of examples. However, the case of the fully supervised learning when the class labels of individual training examples are known can be considered in the same way.

It should be noted that the SDF cannot be called Siamese in the true sense of the word. It does not consist of two gcForests like the SNN. However, its aim coincides with the SNN aim. Therefore, we give this name for the gcForest modification.

The paper is organized as follows. A formal statement of the metric learning problem can be found in Section 2. Section 3 gives a very short introduction into the SNNs. A short description of gcForest proposed by Zhou and Feng [31] is given in Section 4. The ideas underlying the SDF are represented in Section 5 in detail. A modification of gcForest using the weighted averages, which can be regarded as a basis of the SDF is provided in Section 6. Algorithms for training and testing the SDF are considered in Section 7. Numerical experiments with real data illustrating cases when the proposed SDF outperforms gcForest are given in Section 8. Concluding remarks are provided in Section 9.

2. A formal statement of the metric learning problem

Suppose there is a training set $S=\{(\mathbf{x}_i,\mathbf{x}_j,y_{ij}),\ \ (i,j)\in K\}$ consisting of N pairs of examples $\mathbf{x}_i\in\mathbb{R}^m$ and $\mathbf{x}_j\in\mathbb{R}^m$ such that a binary label $y_{ij}\in\{0,1\}$ is assigned to every pair $(\mathbf{x}_i,\mathbf{x}_j)$. If two data vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are semantically similar or belong to the same class of objects, then y_{ij} takes the value 0. If the vectors correspond to different or semantically dissimilar objects, then y_{ij} takes the value 1. This implies that the training set S can be divided into two subsets. The first subset is called the similar or positive set and is defined as

$$\mathcal{S}=\{(\mathbf{x}_i,\mathbf{x}_j):\mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are semantically similar and } y_{ij}=0\}. \quad (1)$$

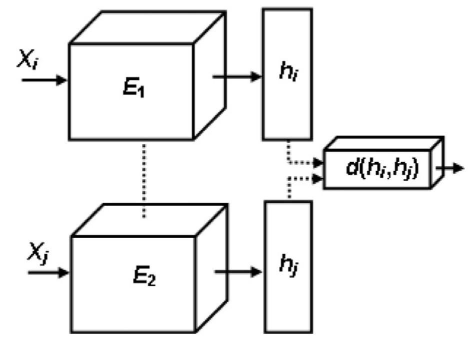

Fig. 1. An architecture of the SNN.

The second subset is the dissimilar or negative set. It is defined as

$$\mathcal{D}=\{(\mathbf{x}_i,\mathbf{x}_j):\mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are semantically dissimilar and } y_{ij}=1\}. \quad (2)$$

If we have two observation vectors $\mathbf{x}_i\in\mathbb{R}^m$ and $\mathbf{x}_j\in\mathbb{R}^m$ from the training set, then the distance $d(\mathbf{x}_i,\mathbf{x}_j)$ should be minimized if $\mathbf{x}_i$ and $\mathbf{x}_j$ are semantically similar, and it should be maximized between dissimilar $\mathbf{x}_i$ and $\mathbf{x}_j$. It has been mentioned that the most general and popular real-valued metric function is the squared Mahalanobis distance $d_M^2(\mathbf{x}_i,\mathbf{x}_j)$ which is defined for vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ as

$$d_M^2(\mathbf{x}_i,\mathbf{x}_j)=(\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}M(\mathbf{x}_i-\mathbf{x}_j). \quad (3)$$

Here $M\in\mathbb{R}^{m\times m}$ is a symmetric positive semi-defined matrix. If $\mathbf{x}_i$ and $\mathbf{x}_j$ are random vectors from the same distribution with covariance matrix C, then $M=C^{-1}$. If M is the identity matrix, then $d_M^2(\mathbf{x}_i,\mathbf{x}_j)$ is the squared Euclidean distance. Learning the Mahalanobis distance metric M implicitly corresponds to seeking a linear transformation which projects data points into a low-dimensional subspace such that the Euclidean distance in the transformed space is equal to the Mahalanobis distance in the original space.

Given subsets $\mathcal{S}$ and $\mathcal{D}$, the metric learning optimization problem can be formulated as follows:

$$M^*=\arg\min_M\,[J(M,\mathcal{D},\mathcal{S})+\lambda\cdot R(M)], \quad (4)$$

where $J(M,\mathcal{D},\mathcal{S})$ is a loss function that penalizes violated constraints; $R(M)$ is some regularizer on M; $\lambda\geq 0$ is the regularization parameter.

3. Siamese neural networks

Before studying the SDF, we consider the SNN which is an efficient and popular tool for dealing with data of the form $\mathcal{S}$ and $\mathcal{D}$. It will be a basis for constructing the SDF.

A standard architecture of the SNN given in the literature (see, for example, [8]) is shown in Fig. 1. Let $\mathbf{x}_i$ and $\mathbf{x}_j$ be two data vectors corresponding to a pair of elements from a training set, for example, images. Suppose that f is a map of $\mathbf{x}_i$ and $\mathbf{x}_j$ to a low-dimensional space such that it is implemented as a neural network with the weight matrix W. At that, parameters W are shared by two neural networks $f(\mathbf{x}_1)$ and $f(\mathbf{x}_2)$ denoted as E_1 and E_2 and corresponding to different input vectors, i.e., they are the same for the two neural networks. The property of the same parameters in the SNN is very important because it defines the corresponding training algorithm. By comparing the outputs $\mathbf{h}_i=f(\mathbf{x}_i)$ and $\mathbf{h}_j=f(\mathbf{x}_j)$ using the Euclidean distance $d(\mathbf{h}_i,\mathbf{h}_j)$, we measure the compatibility between $\mathbf{x}_i$ and $\mathbf{x}_j$.

If we assume for simplicity that the neural network has one hidden layer, then there holds

$$\mathbf{h}=\sigma(W\mathbf{x}+b). \quad (5)$$

Please cite this article as: L.V. Utkin, M.A. Ryabinin, A Siamese Deep Forest, Knowledge-Based Systems (2017), https://doi.org/10.1016/j.knosys.2017.10.006