UNIVERSITÁ DEGLI STUDI "ROMA TRE"
Dipartimento di Matematica e Fisica
Corso di Laurea Magistrale in Scienze Computazionali

Tesi di Laurea Magistrale

Ricerca della tolopologia ottimale di un sistema di deep learning per identificazioni di oggetti architettonici

Candidato
Désirée Adiutori

Relatore
Prof. Alberto Paoluzzi

Anno Accademico 2017/2018
Luglio 2018

Introduction

Since the invention of computer, man increasingly relies on machines to solve complex computational problems. With the growth of computer performances, the computational algorithms have become progressively more efficient. In 1959, Arthur Samuel, a MIT engineer, defined the "machine learning" as a "field of study that gives computers the ability to learn without being explicitly programmed".

We define machine learning as a set of methods that can automatically detect patterns from data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty. Teaching to machines would be not possible without data and, in generally, the more data are provided to a machine, the more it is able to learn. Thus, with the internet, the concept of "machine learning" has become more and more important since 90s, thanks to the enormous quantity of data that can be found on the internet.

"Deep learning" is a particular kind of machine learning that concerns the imitation of the way people learn. It tackles learning machine problems by learning to represents the world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones. The Deep learning entails using artificial neural networks:*deep artificial neural networks*, algorithms and computational systems inspired by human brain in order to face machine learning problems. Shehzad Noor Taus Priyo expressed a similarity that can help us to understand better what neural networks are:

"We can imagine the neural networks as a sequence of doors a man has to cross. The input is represented by the man and every time he crosses a door, there is a change in his behaviour. At the end of the sequence the man who has become a completely different person represents the output of this process". This thesis focuses on a particular kind of machine learning algorithm: Classification, in particular on Imagines Classification. The main goal is to seek an optimal architecture for the algorithm that enables it to identify the images of architectural objects by finding the correct topology, depth and width for every level of the neural networks.

Learning Algorithms
Most machine learning algorithms can be divided into three categories:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

The choice of the algorithm depends on the kind of data we are provided. However, the final choice needs to be done exclusively once the algorithm is tested, because a set of assumptions that works well in one domain may work poorly in another.
Theorem: No Free Lunch
No machine learning algorithm is universally any better than any other.

0.1 Building a learning algorithm

Building an algorithm requires:

- task, operations that the algorithm must perform;
- performance measures,
- esperiences, data to learn from

Machine learning tasks describes how the machine learning system should process an example.

Definition 1 *An example is a collection of features that have been quantitatively measured from some object or event that we want the machine learning system to process.*

We typically represent an example as a vector $x \in \mathbb{R}^n$, where each entry x_i of the vector is another feature.
In order to evaluate the abilities of a machine learning algorithm, we must design a quantitative measure of its performance, that is specific by the task. Finally, we have to give the algorithm an experience on which it is possible to learn. Such experience will enable to classify the algorithm into one of the three main kind of algorithms and is learnt thanks to the datasets: a collection of many examples. We have different types of datasets:

- training set
- test set
- validation set

The **training set** is a part of the data set that can be used to train a supervised learning system. From this set the algorithm has to build a function that understands which characters describe the various categories.
The **test set**, with the training set, constitute a partition of the starting dataset. Those new pieces of data are used to assess the learning of the "trained" algorithm.
The **validation set** is used similarly to the test set, but we already know the output of the input data (it's possible that a part of them belongs to the training set) and from the outpout we can evaluate if the output is optimal or not.
These three sets can be used at the same time. The choice of the cardinality of these sets is not universal and depends on the type of problem faced.

Definition 2 *The **training error** is an error measure compute on the training set.*

Definition 3 *The **generalization** is the ability to perform well on previously unobserved inputs (test set).*

Definition 4 *The **generalization error** is an error measure compute on the test set.*
It also called the test error.

We may suppose that in each dataset the examples are independent one from each others, and that the train set and test set are identically distributed, drawn from the same probability distribution, as each other.

Definition 5 *A **loss function** $L(y, \hat{y})$ is a function that measure the distance (or error) between the model outputs $\hat{y}$ and the target (truth) values y.*

Several loss functions can be adopted, an example, is the so called Mean Sqared error (MSE)

$$L(y, \hat{y} = \frac{1}{n} \sum_{i=1}^{n} (\hat{y} - y)^2$$

The prediction function will depend on the vector of parameters w and the goal is to choose its optimal value i.e. the value of w which minimize the

training error. According to the learning method and to the specific problem, several uncostrained minimization algorithms can be applied. A first example of unconstrained optimization method for a given problem

$$f(x^*) = \min_{x \in \mathbb{R}^n} f(x), \qquad f \in C^2$$

is the metod of gradient descent, which is based on the following recursive equation

$$x_{k+1} = x_k + \beta_k d_k \tag{1}$$

where x_0 is given, $\beta_k \in \mathbb{R}^+$ is the step and $d_k \in \mathbb{R}^n$ is the direction along which the solution moves; such a direction is descendent i.e. $(d_k, \nabla f(x_k)) < 0$. The step and the direction are suitably chosen at each recursion such that $f(x_{k+1}) < f(x_k)$.
Namely, for the step we solve

$$f(x_k + \beta_k d_k) = \min_{\beta} \{f(x_k + \beta d_k)\} \qquad (exact\ research\ strategy)$$

and for the direction we choose the value

$$d_k = -\nabla f(x_k)$$

Note that the directional derivative of f along d_k is

$$\frac{\partial f}{\partial d_k}(x_k) = \frac{(d_k, \nabla f(x_k))}{\parallel d_k \parallel} = - \parallel \nabla f(x_k) \parallel$$

and from the Cuchy-Schwartz inequality, it follows that

$$\frac{\mid (d_k, \nabla f(x_k)) \mid}{\parallel d_k \parallel} \leq \frac{\parallel d_k \parallel \parallel \nabla f(x_k) \parallel}{\parallel d_k \parallel} = \parallel \nabla f(x_k) \parallel$$

which means that the direction of research is exactly the one such that the directional derivative of f is negative and with maximum absolute value.
The stopping criteria are: $\parallel x_{k+1} - xk \parallel \leq m$, $\parallel \nabla f(x_{k+1}) \parallel \leq m\prime$ or $k > k_{max}$, where m and m' are given thresholds, and k_{max} is the maximum accepted number of iterations.
The above results are guaranteed by the following convergence theorems:

Teorema 1 *Let $f(x) \in C^1$, strictly convex on the set $\Sigma_0 = \{x \in \mathbb{R}^\times : f(x) \leq f(x_0)\}$, and let $\{x_k\}$ be a sequence generated by the algorithm (1). Suppose*

1. *Σ_0 is a compact set;*

2. *the direction* d_k *are such that* $\frac{(d_k, \nabla f(x_k))}{\|d_k \quad \|\|\nabla f(x_k)\|} \leq -\cos\theta$ *for* $k \in I$*, with* I *unlimited set of indices;*

3. *for* $k \in I$*,* β_k *are obtained by exact research.*

Then the sequence $\{x_k\}$ *converges to the only one minimum point* x^* *of* f*.*

Teorema 2 *Let* $f(x) \in C^2$*, strictly convex on the set (suppose it be compact)* $\Sigma_0 = \{x \in \mathbb{R}^\kappa : f(x) \leq f(x_0)\}$*, and let* $\{x_k\}$ *be a sequence generated by the algorithm (1), with* $d_k = -\nabla f(x_k)$*.*
If the β_k *step are obtained by exact research, than the sequence* x_k *converges to the only one minimum point* x^* *of* f*.*

Minimize the training error does not necessarily entail the optimization of the algorithm learning: we can have the underfitting fenomena, it occurs when the model is not able to obtain a sufficiently low error value on the training set. We must also evaluate other factors: test set analysis.
From eq.(**??**) we compute the error training:

$$L(y, \hat{y})_{test} = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_{(test)} - y_{(test)})_i^2 \tag{2}$$

Having found the parameters that minimise the training error, we aim to obtain the minimum error too (the best result is equal to 0). However, as told before, it does not always occurs so that our goal is to have the minimum difference between the two errors. In the opposite case, it occurs the event of the *overfitting* of the model to the set of data that it describes through an excessive number of parameters. Thus, as far as a new dataset is concerned, the model will not be able to be generalised.
Let a pair (X, Y) from test set, we consider the expected test error:

$$\mathbb{E}[L(y, \hat{y})_{test}] = \mathbb{E}[(Y - \hat{y}(X))^2] \tag{3}$$

and we define the true function as:

$$y(X) = \mathbb{E}(Y|X)$$

we would still incure some error, due to noise, what we call estimation bias. Over different copies of the training set, we end up constructing substantially different functions $\hat{y}$, this is another source of error, that we'll call estimation variance. We can write the model:

$$Y = y(X) + \epsilon$$

where we'll assume that ϵ is indipendent of X, with $\mathbb{E}[X] = 0$ and $Var(X) = \sigma^2$.
We'll look at the expected test error equation (3):

$$\begin{aligned} \mathbb{E}[L(y, \hat{y})_{test}] &= \mathbb{E}[(Y - \hat{y}(X))^2 | X = x] \\ &= \mathbb{E}[(Y - y(x))^2 | X = x] + \mathbb{E}[(y(x) - \hat{y}(x))^2 | X = x] \\ &= \sigma^2 + \mathbb{E}[(y(x) - \hat{y}(x))^2]. \end{aligned} \tag{4}$$

The first term σ^2 is the Bayes error. The second term can be further decomposed as

$$\begin{aligned} \mathbb{E}[(y(x) - \hat{y}(x))^2] &= (\mathbb{E}[\hat{y}(x)] - y(x))^2 + \mathbb{E}[(\hat{y}(x) - \mathbb{E}[\hat{y}(x)])^2] \\ &= Bias(\hat{y}(x))^2 + Var(\hat{y}(x)) \end{aligned} \tag{5}$$

We have the bias-variance tradeoff:

$$\mathbb{E}[L(y, \hat{y})_{test}] = \sigma^2 + Bias(\hat{y}(x))^2 + Var(\hat{y}(x)) \tag{6}$$

If we have high bias and low variance, we will incure in underfitting. If we have low bias and high variance, we will incure in overfitting.[**?**] The most common way to negotiate this trade-off is to use cross-validation. This procedure is based on the idea of repeating the training and testing computation on different randomly chosen subsets or splits of the original dataset.

Supervised learning
The supervised learning algorithms are used in order to solve classification and regression problems. We talk about supervised learning when the dataset we are using includes some variables, each one a label.
Given an input vector $x = (x_1, \cdots, x_n)$, each x_i is a d-dimensional vector of numbers representing a features, from those pieces of data we build the training set of cardinalities N: $D = \{(x_i, y_i)\}_{i=1}^N$, where $y = (y_1, \cdots, y_m)$ is the preferred outputs and y_i is the label. Our aim is to learn a general law that links the inputs with the outputs data, so that the algorithm learns to classify a brand new example that does not contains the label.
When y_i is nominal, the problem is known as classification, and when y_i is real-valued, the problem is known as regression. If we indicate with C the number of classes an output is able to belong to: $y \in \{1, ..., C\}$, if $C = 2$, this is called binary classification (in which case we often assume $y \in \{0, 1\}$); if $C > 2$, this is called multiclass classification..

Classification
Classification is used when it is necessary to decide which category a certain

piece of data belongs to. For example, given a photograph, classification is used to understand which category, in this case a monument, it belongs to. The learning algorithm is asked to specify which of k categories some input belongs to. It produce a function $f : \mathbb{R}^n \to \{1, \cdots, k\}$, when $y = f(x)$, the model assigns an input described by vector x to a category identified by numeric code y. There are other variants of the classification task, for example, where f outputs a probability distribution over classes.

Regression

Being known the current value of a piece of data, regression predicts its future value. An example is the prediction of currencies and shares values. It is used in marketing to envisage the feedback rate of a marketing campaign based on a certain customers profile as well as in commerce to estimate how a firm turnover varies through a change in the strategy. It occurs by building a function that best fits to the points that describe the distribution of the Y along the X.

Linear Regression

The goal is to build a system that can take a vector $x \in \mathbb{R}^n$ as input and predict the value of a scalar $y \in \mathbb{R}$, as its output. Where $y = f(x)$, and f is a linear function:

$$y = \alpha + \beta x \qquad (retta\ di\ regressione)$$

Let $\hat{y}$ be the value that our model predicts y should take on. We define the output to be:

$$\hat{y} = \alpha + \beta x + \epsilon$$

where ϵ is that error.

Now, in order to identify the line that best fits to the points that describe the distribution of y along the x, it is crucial to assess the values of the parameters α and β, through the data observed in an example.

We use the least squares method that minimise the error ϵ.

$$\begin{aligned} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 &= \sum_{i=1}^{n} (\hat{y}_i - (\alpha + \beta x_i))^2 \\ &= \sum_{i=1}^{n} (\hat{y}_i - \alpha - \beta x_i)^2 = \min \end{aligned}$$

In order to find the values of α e β, then we solve the system

$$\begin{cases} \dfrac{d(\sum_{i=1}^{n}(\hat{y}_i - \alpha - \beta x_i)^2)}{d\,\alpha} = 0 & (7) \\ \dfrac{d(\sum_{i=1}^{n}(\hat{y}_i - \alpha - \beta x_i)^2)}{d\,\beta} = 0 & (8) \end{cases}$$

Solving from (7) we obtein:

$$\alpha = \mathbb{E}[y] - \beta\mathbb{E}[x]$$

and from (8):

$$\beta = \frac{\sum_{i=1}^{n}(x_i - \mathbb{E}[x])(y_i - \mathbb{E}[y])}{\sum_{i}^{n}(x_i - \mathbb{E}[x])^2} = \frac{Cov(x,y)}{Var(x)}$$

we obtain the values of the parameters we were looking for.

Unsupervised learning
Unsupervised learning algorithms are used to solve clustering problems. Here we are only given inputs,$D = \{x_i\}_{i=1}^{N}$ and the goal is to find "interesting patterns" in the data. This is sometimes called knowledge discovery. Differently from supervised learning, this learning does not have a classification or a final result with which it is possible to assess whether a result is reliable, but it generalizes the characters of the data and, thanks to them, it gives an output to an input: it need to extract pieces of information that are not yet known. Through the deep learning, the aim is to get to the probability distribution that has created a dataset. In the context of deep learning, we usually want to learn the entire probability distribution that generated a dataset.This technique is called *clustering*.

Reinforcement learning
Reinforcement learning algorithm are used to solve regression problems. The goal of this algorithm is to create a system that is able to learn and adapt to changes in the environment in which they work, through a distribution of a "reward" called reinforcement, given by the performances evaluation.
These algorithms are constructed on the idea that the correct results should be remembered, by means of a reinforcement signal, such that they become more likely to be used another time; vice versa if the results is wrong, the signal will be a penalty, such that they will have a lower probability to be used in future.