www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 1 January, 2014 Page No.3778-3783

Pattern Recognition and Their Different Approach

Ajit Kumar, Rajeev Ranjan, Dr. B.Mishra

*(Computer Science, S.R.K.G. College/ B.R.Ambedkar Bihar Universiy, Bihar, India)
*(Computer Science, S.R.K.G. College/ B.R.Ambedkar Bihar Universiy, Bihar, India)
*(Department Of Mathematics, S.R.K.G. College/ B.R.Ambedkar Bihar Universiy, Bihar, India)

E-mail: - ajit.phd2013@gmail.com , rajeev_smg@gmail.com

Abstract

"Pattern recognition is the research area that studies the operation and design of systems that machine diagnostics, person identification and industrial inspection."

"A branch of artificial intelligence concerned with the classification or description of observations. Pattern recognition aims to classify data (patterns) based on either a priori knowledge recognize patterns in data. It encloses sub disciplines like discriminant analysis, feature extraction, error estimation, cluster analysis (together sometimes called statistical pattern recognition), grammatical inference and parsing (sometimes called syntactical pattern recognition). Important application areas are image analysis, character recognition, speech analysis, man and or on statistical information extracted from the patterns. The patterns to be classified are usually groups of measurements or observations, defining points in an appropriate multidimensional space."
Pattern recognition is the science of making inferences based on data. Two of the main forms of pattern recognition are classification and regression. In classification problems, data are collected and given discrete class labels. In a regression problem, on the other hand, data labels are typically continuous values, not categorical. Basic pattern recognition approaches seek a function, f, that takes an observation and predicts the unseen label, y:=f(x_j). The goals of learning in pattern recognition are to develop the function, f, given only a (possibly small) set of training data, X={{x_i,y_i}}, I=1....N. As such, pattern recognition is fundamentally an ill-posed problem, since it is trivially easy to define a function that performs arbitrarily well on the training data (($f(x_j) = y_j)\forall x_j, y_j \in X$).The present paper reviews the techniques for automated extraction of information from signals. The techniques may be classified broadly into two categories—the conventional pattern recognition approach and the artificial intelligence (AI) based approach.

Keywords: - Artificial intelligence, Feature Variables, Knowledge representation, Pattern recognition,

Probabilistic classifiers

Introduction:-

Considering a simple example can often make the problem of pattern recognition clearer. Assume that we run a scrap metal processing yard, and we'd like to have a machine automatically sort pieces of scrap metal into different piles depending on if they are zinc or copper. As pieces of metal move down our conveyor belt, we use our Trusty ACME DensAndRef machine to measure the density of each piece of scrap, and it's reflectance at a particular wavelength (how much light the object reflects). As pieces come down the conveyor belt, we need to make decisions (copper or zinc) based on these two pieces of data. Since we might know something ahead of time about the reflectance and densities of these two types of metal, it might make sense to make up some rules even if we haven't seen any data from our sensors. For example, we might develop an algorithm that says:

If (density < 0.15 lb/in^3) and (reflectance > 80%) object is aluminium.

However making decision boundaries like this is an error-prone approach. First of all, we haven't incorporated any information about our actual sensing apparatus into these rules. It might turn out that our sensing apparatus has a strong bias, and all reflectance are measured at 1/2 their true reflectance. Or we might find out that our density measuring equipment has a large degree of noise, so that densities of very light objects are sometimes recorded as very large. Second, we haven't incorporated any information about the kind of data we might actually see. The equations above are suitable for accurate and precise measurements of pure metals, but are not suitable for detecting some amount of aluminium mixed in alloys with other components. To overcome these limitations, we need to have some training data.

Luckily, in most learning tasks, data will be available on which we can perform learning. This data will typically consist of N sets of *observations* and *labels* (or *targets*). Usually observation vectors are designated with x_ {i} and targets or labels with y_ {i}, where i indexes each observation/target pair. A set of N measurements can then be denoted $X = x_i, y_i$ for i = 1...N. In our example, they x_i are 1 x 2 vectors where $x_i(1)$the measured density is, and $x_i(2)$ is the measured reflectance. y_i In our case is a binary variable where 0 represents "measured from copper", and 1 represents "measured from aluminium". The goal now is, given some set of training data, how we can define a boundary .In machine learning, pattern recognition is the assignment of a label to a given input value. An example of pattern recognition is classification, which attempts to assign each input value to one of a given set of *classes* (for example, determine whether a given email is "spam" or "non-spam"). However, pattern recognition is a more general problem that encompasses other types of output as well. Other examples are regression, which assigns a real-valued output to each input; sequence labelling, which assigns a class to each member of a sequence of values (for example, part of speech tagging, which assigns a part of speech to each word in an input sentence); and parsing, which assigns a parse tree to an input sentence, describing the syntactic structure of the sentence.

Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs and to perform "most likely" matching of the inputs, taking into account their statistical variation. This is opposed to pattern matching algorithms, which look for exact matches in the input with pre-existing patterns. A common example of a pattern-matching algorithm is expression matching, which looks for patterns of a given sort in textual data and is included in the search capabilities of many text editors and word processors. In contrast to pattern recognition, pattern matching is generally not considered a type of machine learning, although pattern-matching algorithms (especially with fairly general, carefully tailored patterns) can sometimes succeed in providing similar-quality output to the sort provided by pattern-recognition algorithms.

Pattern recognition is generally categorized according to the type of learning procedure used to generate the output value. Supervised learning assumes that a set of *training data* (the *training set*) has been provided, consisting of a set of instances that have been properly labelled by hand with the correct output. A learning procedure then generates a *model* that attempts to meet two sometimes conflicting objectives: Perform as well as possible on the training data, and generalize as well as possible to new data (usually, this means being as simple as possible, for some technical definition of "simple", in accordance with Occam's Razor, discussed below). Unsupervised learning, on the other hand, assumes training data that has not been hand-labelled, and attempts to find inherent patterns in the data that can then be used to determine the correct output value for new data instances. A combination of the two that has recently been explored is semi, which uses a combination of labelled and unlabeled data (typically a small set of labelled data combined with a large amount of unlabeled data). Note that in cases of unsupervised learning, there may be no training data at all to speak of; in other words, the data to be labelled *is* the training data.

Note that sometimes different terms are used to describe the corresponding supervised and unsupervised learning procedures for the same type of output. For example, the unsupervised equivalent of classification is normally known as clustering, based on the common perception of the task as involving no training data to speak of, and of grouping the input data into *clusters* based on some inherent similarity measure (e.g. the distance between instances, considered as vectors in a multi-dimensional vector space), rather than assigning each input instance into one of a set of pre-defined classes. Note also that in some fields, the terminology is different: For example, in community ecology, the term "classification" is used to refer to what is commonly known as "clustering".

The piece of input data for which an output value is generated is formally termed an *instance*. The instance is formally described by a vector of *features*, which together constitute a description of all known characteristics of the instance. (These feature vectors can be seen as defining points in an appropriate multidimensional space, and methods for manipulating vectors in spaces can be correspondingly applied to them, such as computing the dot product or the angle between two vectors.) Typically, features are either categorical (also known as nominal, i.e., consisting of one of a set of unordered items, such as a gender of "male" or "female", or a blood type of "A", "B", "AB" or "O"), ordinal (consisting of one of a set of ordered items, e.g., "large", "medium" or "small"), integer-valued (e.g., a count of the number of occurrences of a particular word in an email) or real-valued (e.g., a measurement of blood pressure). Often, categorical and ordinal data are grouped together; likewise for integer-valued and real-valued data. Furthermore, many algorithms work only in terms of categorical data and require that real-valued or integer-valued data be*d is cretized* into groups (e.g., less than 5, between 5 and 10, or greater than 10).

Probabilistic classifiers

Many common pattern recognition algorithms are *probabilistic* in nature, in that they use statistical inference to find the best label for a given instance. Unlike other algorithms, which simply output a "best" label, often probabilistic algorithms also output a probability of the instance being described by the given label. In addition, many probabilistic algorithms output a list of the *N*-best labels with associated probabilities, for some value of *N*, instead of simply a single best label. When the number of possible labels is fairly small (e.g., in the case of classification), *N* may be set so that the probability of all possible labels is output. Probabilistic algorithms have many advantages over non-probabilistic algorithms:

- They output a confidence value associated with their choice. (Note that some other algorithms may also output confidence values, but in general, only for probabilistic algorithms are this value mathematically grounded in probability theory. Non-probabilistic confidence values can in general not be given any specific meaning, and only used to compare against other confidence values output by the same algorithm.)

- Correspondingly, they can *abstain* when the confidence of choosing any particular output is too low.
- Because of the probabilities output, probabilistic pattern-recognition algorithms can be more effectively incorporated into larger machine-learning tasks, in a way that partially or completely avoids the problem of *error propagation*.

Feature variables

Feature selection algorithms, attempt to directly prune out redundant or irrelevant features. A general introduction to feature selection which summarizes approaches and challenges has been given. The complexity of feature-selection is, because of its non-monotonous character, an optimization problem where given a total of n features the power set consisting of all 2^n - 1 subsets of features need to be explored. The Branch-and-Bound algorithm does reduce this complexity but is intractable for medium to large values of the number of available features n. For a large-scale comparison of feature-selection algorithms .Techniques to transform the raw feature vectors (feature extraction) are sometimes used prior to application of the pattern-matching algorithm. For example, feature extraction algorithms attempt to reduce a large-dimensionality feature vector into a smaller-dimensionality vector that is easier to work with and encodes less redundancy, using mathematical techniques such as Principal Component Analysis (PCA). The distinction between feature selection and feature extraction is that the resulting features after feature extraction has taken place are of a different sort than the original features and may not easily be interpretable, while the features left after feature selection are simply a subset of the original features.

Problem statement (supervised version)

Formally, the problem of supervised pattern recognition can be stated as follows: Given an unknown function : g x-->y (the *ground truth*) that maps inputes $\boldsymbol{x} \in \mathcal{X}$ to output labels $y \in \mathcal{Y}$, along with training data $\mathbf{D} = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}$ assumed to represent accurate examples of the mapping, produce a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that approximates as closely as possible the correct mapping g. (For example, if the problem is filtering spam, then $\boldsymbol{x}_i$ is some representation of an email and y is either "spam" or "non-spam"). In order for this to be a well-defined problem "approximates as closely as possible" needs to be defined rigorously. In decision theory, this is defined by specifying a loss function that assigns a specific value to "loss" resulting from producing an incorrect label. The goal then is to minimize the expected loss, with the expectation taken over the probability distribution of $\mathcal{X}$. In practice, neither the distribution of $\mathcal{X}$ nor the ground truth function $g : \mathcal{X} \rightarrow \mathcal{Y}$ are known exactly, but can be computed only empirically by collecting a large number of samples of x and hand-labelling them using the correct value of y (a time-consuming process, which is typically the limiting factor in the amount of data of this sort that can be collected). The particular loss function depends on the type of label being predicted. For example, in the case of classification, the simple zero-one loss function is often sufficient. This corresponds simply to assigning a loss of 1 to any incorrect labelling and implies that the optimal classifier minimizes the error rate on independent test data (i.e. counting up the fraction of instances that the learned function $h : \mathcal{X} \rightarrow \mathcal{Y}$ labels wrongly, which is equivalent to maximizing the number of correctly classified instances). The goal of the learning procedure is then to minimize the error rate (maximize the correctness) on a "typical" test set.

For a probabilistic pattern recognizer, the problem is instead to estimate the probability of each possible output label given a particular input instance, i.e., to estimate a function of the form

$$p(\text{label}|\boldsymbol{x}, \boldsymbol{\theta}) = f(\boldsymbol{x}; \boldsymbol{\theta})$$

Where the feature vector input is $\boldsymbol{x}$, and the function *f* is typically parameterized by some parameters $\boldsymbol{\theta}$. In a discriminative approach to the problem, *f* is estimated directly. In a generative approach, however, the inverse probability $p(\boldsymbol{x}|\text{label})$ is instead estimated and combined with the prior probability $p(\text{label}|\boldsymbol{\theta})$ using Bayes' rule, as follows:

$$p(\text{label}|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{p(\boldsymbol{x}|\text{label})p(\text{label}|\boldsymbol{\theta})}{\sum_{L \in \text{all labels}} p(\boldsymbol{x}|L)p(L|\boldsymbol{\theta})}.$$

When the labels are continuously distributed (e.g., in regression analysis), the denominator in-
v
o
l
v

$$p(\text{label}|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{p(\boldsymbol{x}|\text{label})p(\text{label}|\boldsymbol{\theta})}{\int_{L \in \text{all labels}} p(\boldsymbol{x}|L)p(L|\boldsymbol{\theta}) \, \mathrm{d}L}.$$

e
s integration rather than summation:

The value of $\boldsymbol{\theta}$ is typically learned using maximum a posteriori (MAP) estimation. This finds the best value that simultaneously meets two conflicting objects: To perform as well as possible on the training data (smallest error-rate) and to find the simplest possible model. Essentially, this combines maximum likelihood estimation with a regularization procedure that favours simpler models over more complex models. In a Bayesian context, the regularization procedure can be viewed as placing a prior probability $p(\boldsymbol{\theta})$ on different values of $\boldsymbol{\theta}$. Mathematically:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{D})$$

Where $\boldsymbol{\theta}^*$ is the value used for $\boldsymbol{\theta}$ in the subsequent evaluation procedure, and $p(\boldsymbol{\theta}|\mathbf{D})$, the posterior probability of $\boldsymbol{\theta}$, is given by

$$p(\boldsymbol{\theta}|\mathrm{D}) = \left[\prod_{i=1}^{n} p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta})\right] p(\boldsymbol{\theta}).$$

In the Bayesian approach to this problem, instead of choosing a single parameter vector $\boldsymbol{\theta}^*$, the probability of a given label for a new instance $\boldsymbol{x}$ is computed by integrating over all possible values of $\boldsymbol{\theta}$, weighted according to the posterior probability:

$$p(\text{label}|\boldsymbol{x}) = \int p(\text{label}|\boldsymbol{x},\boldsymbol{\theta})p(\boldsymbol{\theta}|\text{D})\,\mathrm{d}\,\boldsymbol{\theta}.$$

Frequentist or Bayesian approach to pattern recognition:-

The first pattern classifier – the linear discriminant presented by Fisher – was developed in the Frequentist tradition. The frequentist approach entails that the model parameters are considered unknown, but objective. The parameters are then computed (estimated) from the collected data. For the linear discriminant, these parameters are precisely the mean vectors and the Covariance matrix. Also the probability of each class $p(\text{label}|\boldsymbol{\theta})$ is estimated from the collected dataset. Note that the usage of 'Bayes rule' in a pattern classifier does not make the classification approach Bayesian.

Bayesian statistics has its origin in Greek philosophy where a distinction was already made between the 'a priori' and the 'a posteriori' knowledge. Later Kant defined his distinction between what is a priori known – before observation – and the empirical knowledge gained from observations. In a Bayesian pattern classifier, the class probabilities $p(\text{label}|\boldsymbol{\theta})$ can be chosen by the user, which are then a priori. Moreover, experience quantified as a priori parameter values can be weighted with empirical observations – using e.g., the Beta- (conjugate prior) and Dirichlet-distributions. The Bayesian approach facilitates a seamless intermixing between expert knowledge in the form of subjective probabilities, and objective observations.

Probabilistic pattern classifiers can be used according to a frequentist or a Bayesian approach.

Within medical science, pattern recognition is the basis for computer-aided diagnosis (CAD) systems. CAD describes a procedure that supports the doctor's interpretations and findings. Other typical applications of pattern recognition techniques are automatic speech recognition, classification of text into several categories (e.g., spam/non-spam email messages), the automatic recognition of handwritten postal codes on postal envelopes, automatic recognition of images of human faces, or handwriting image extraction from medical forms. The last two examples from the subtopic image analysis of pattern recognition that deals with digital images as input to pattern recognition systems.

Optical character recognition is a classic example of the application of a pattern classifier, see OCR-example. The method of signing one's name was captured with stylus and overlay starting in 1990. The strokes, speed, relative min, relative max, acceleration and pressure are used to uniquely identify and confirm identity. Banks were first offered this technology, but were content to collect from the FDIC for any bank fraud and did not want to inconvenience customers.

Neural networks (neural net classifiers) have many real-world applications in image processing, a few examples:

1. identification and authentication: e.g., license plate recognition, fingerprint analysis and face detection/verification;]
2. medical diagnosis: e.g., screening for cervical cancer (Pap net) or breast tumours';
3. Defence: various navigation and guidance systems, target recognition systems, etc.

For a discussion of the aforementioned applications of neural networks in image processing, see e.g.

In psychology, pattern recognition, making sense of and identifying the objects we see is closely related to perception, which explains how the sensory inputs we receive, are made meaningful. Pattern recognition can be thought of in two different ways: the first being template matching and the second being feature detection. A template is a pattern used to produce items of the same proportions. The template-matching hypothesis suggests that incoming stimuli are compared with templates in the long term memory. If there is a match, the stimulus is identified. Feature detection models, such as the Pandemonium system for classifying letters (Selfridge, 1959); suggest that the stimuli are broken down into their component parts for identification. For example, a capital E has three horizontal lines and one vertical line.

Algorithms for pattern recognition depend on the type of label output, on whether learning is supervised or unsupervised, and on whether the algorithm is statistical or non-statistical in nature. Statistical algorithms can further be categorized as generative or discriminative. Categorical sequence labelling algorithms (predicting sequences of categorical labels)

Supervised:

- Conditional random fields (CRFs)
- Hidden Markov models (HMMs)
- Maximum entropy Markov models (MEMMs)

Unsupervised:

- Hidden Markov models (HMMs)

Classification algorithms (supervised algorithms redicting categorical labels) Parametric

- Linear discriminant analysis
- Quadratic discriminant analysis
- Maximum entropy classifier (aka logistic regression, multinomial logistic regression): Note that logistic regression is an algorithm for classification, despite its name. (The name comes from the fact that

logistic regression uses an extension of a linear regression model to model the probability of an input being in a particular class.)

Nonparametric:

- Decision trees, decision lists
- Kernel estimation and K-nearest-
neighbour algorithms
- Naive Bayes classifier
- Neural networks (multi-layer perceptrons)
- Perceptrons
- Support vector machines
- Gene expression programming

Clustering algorithms (unsupervised algorithms predicting categorical labels)

- Categorical mixture models
- Deep learning methods
- Hierarchical clustering (agglomerative or divisive)
- K-means clustering
- Kernel principal component analysis (Kernel PCA)

Ensemble learning algorithms (supervised meta-algorithms for combining multiple learning algorithms together)

- Boosting (meta-algorithm)
- Bootstrap aggregating ("bagging")
- Ensemble averaging
- Mixture of experts, hierarchical mixture of experts

General algorithms for predicting arbitrarily-structured (sets of) labels

- Bayesian networks
- Markov random fields

Multilinear subspace learning algorithms (predicting labels of multidimensional data using tensor representations)

Unsupervised:

- Multilinear principal component analysis (MPCA)

Parsing algorithms (predicting tree structured labels)

Supervised and unsupervised:

Probabilistic context free grammars (PCFGs)

Real-valued sequence labelling algorithms (predicting sequences of real-valued labels)

Supervised (?):

- Kalman filters
- Particle filters

Regression algorithms (predicting real-valued labels)

Supervised:

Gaussian process regression (kriging)

- Linear regression and extensions
- Neural networks

Unsupervised:

- Independent component analysis (ICA)
- Principal components analysis (PCA)

Classifier to choose for a classification task: - we are giving an extensive list of statistical classifiers for supervised and unsupervised classification tasks, clustering and general regression prediction. When considering building a classifier e.g., for a software application, a number of different aspects influence the choice of the preferred classifier type to use.

Building or *training* a classifier is essentially statistical inference. This means that an attempt is made to identify stochastic (often unknown) relations between feature variables and the categories to be predicted. For example, the influence of increased cholesterol on the risk of a heart attacks for a patient, within the next year. Which other variables besides the current cholesterol level determine this risk? The two categories to 'predict' by a classifier are 'heart attack likely', or 'heart attack unlikely'.

The theoretically optimal classifier is called the Bayes classifier. It minimizes the loss-function or *risk* as defined here. When all types of misclassifications are associated with equal losses (outcome A becomes B is as undesired as when outcome B becomes A), the Bayes classifier with the minimal error rate (on a test set) is the optimal one for the classification task. In general, it is unknown what is the optimal classifier type and true parameters θ. However, bounds on the optimal Bayes error rate have been derived. For, for example, the K-nearest neighbour classifier theoretical results are derived that bound the error rate, in relation to the optimal Bayes error rate.

In essence, building a classifier brings model selection with it. Feature selection – using only a subset of the available feature variables to predict the most likely categorical outcome – by itself entails model selection. Choosing among the extensive set of different classifiers makes model selection even more complex. A theoretical analysis of this search problem has been presented as the no free lunch theorem: No particular classification algorithm is 'the best' for all problems. The pragmatic approach to this open problem is to combine prior knowledge of the classification task (e.g. distributional assumptions) with a search process where different types of classifiers are developed and their performance compared.

The search for the 'best' model that predicts observations and relations between these is a problem that was discovered already in ancient Greece. In medieval times, Occam's razor was formulated:

"Plurality should not be posited without necessity".

In this context, it means that if a simple classifier with only a few parameters (small θ) does the job as well as a much more complex classification algorithm, choose the simpler one. Only to add that the performance of a classifier is but one of the criteria to apply when choosing the best classification model. Distributional assumptions, insight provided into the discovered relations between variables, whether the classification algorithm can cope with missing feature variables, whether a change in class prior probability can be incorporated, speed of the training algorithm, memory requirements, parallelization of the classification process, resemblance to the human perceptual system, and other aspects as well. In visual pattern recognition invariance to variations in colour, rotation and scale are extra properties that need to be accounted for.

Supervised classification

When choosing the most appropriate supervised classifier, the generally accepted heuristic is to:

1. Separate the available data, at random, into training set and a test set. Use the test set only for the final performance comparison of the trained classifiers.

2. Experiment by training a number of classification algorithms, including parametric(discriminant analysis , multinomial classifier) and non-parametric algorithms (k-nearest neighbour, a support vector machine, a feed-forward neural network, a standard decision-tree algorithm).

3. Test distributional assumptions of the (continuous) feature-distributions per category. Are they Gaussian?

4. Which subset of feature variables contributes mostly to the discriminative performance of the classifier?

5. Are elaborate confidence intervals needed for the error-rates and class-predictions

6. White-box versus black-box considerations may render specific classifiers unsuited for the job.

Examples of Patterns

1. Crystal patterns at atomic and molecular levels .These structures can be represented by graphs or by grammars. This is often called syntactic pattern recognition with generative models. One may view a compiler for a programming language (e.g. matlab, c) as a syntactic pattern recognition system. A syntactic pattern recognition system not only classifies the input, but also extracts hierarchical (compositional) structures. Pattern, in English usually, refers to regular repeated structures. But in pattern recognition, anything that you can perceive is a pattern.
2. A pattern often exhibits a wide range of variations with nuisance factors : e.g. faces
   1. Expression –geometric deformation
   2. Lighting --- photometric deformation
   3. 3D pose transform
   4. Noise and occlusion
   Each pattern corresponds to a set (sometimes a manifold) in the signal space.
   The nuisance factors are called attributes when they are "useful"
3. Face recognition becomes tractable with infrared camera in constrained environments.

References

[1.] Duda, Hart, Stork, Pattern Classification http://www.amazon.com/Pattern-Classification-2nd-Richard-Duda/dp/0471056693

[2.] Bishop, Pattern Recognition and Machine learning http://www.amazon.com/Pattern-Recognition-Learning-Information-Statistics/dp/0387310738

[3.] http://www.stat.ucla.edu/~sczhu/Courses/UCLA/Stat_231/Lect_note/Lect1_intro_to_PR.pdf