Interactive and Incremental Learning via a Mixture of Supervised and Unsupervised Learning Strategies

Qiong Liu, Stephen Levinson, Ying Wu, Thomas Huang
Email: {q-liu2, sel, yingwu, huang}@ifp.uiuc.edu
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign
Urbana, IL61801

Abstract

Machine learning paradigms are generally separated into supervised learning and unsupervised learning. Both of these paradigms have their own advantages in practice. But existing algorithms of these two paradigms also expose some hard problems in many different applications. In this paper, we first analyze the general problems of these two paradigms, and some successful techniques for boosting their performance. Then we propose a novel algorithm that can overcome some of existing problems through a mixture of these two paradigms. The algorithm is tested with a robot language-learning task. Equipped with this algorithm, our robot is able to acquire short audio information on-line, and gradually understand the audio input through human's intensive teaching.

1.Introduction

Current machine learning paradigms are generally separated into two categories – learning with a teacher (supervised learning), and learning without a teacher (unsupervised learning) [4]. Both of these learning paradigms have their advantages to lead them to a variety of successes. But their disadvantages greatly limit their achievements. We propose an algorithm that can combine the benefits of these two paradigms, and avoid those disadvantages.

1.1 Supervised Learning

The supervised learning paradigm employs a teacher in the machine learning process [4]. The advantage of this paradigm is that all separate classes in the algorithms are meaningful to humans. The disadvantage of this paradigm is that all objective data samples are forced into subjective meaningful classes without considering if these samples are objectively separable or not. This disadvantage of supervised learning can be easily demonstrated with a simple example in Fig 1.

In Fig 1, the dark region is occupied by feature examples from one subjective class; the light region is occupied by feature examples from another subjective class. These two classes may have very clear meanings (e.g. dog or chicken) to humans, but we can see that they are not linearly separable. So, if a scientist wants to separate these two subjective classes with a linear separator, the scientist may experience difficulties doing this.

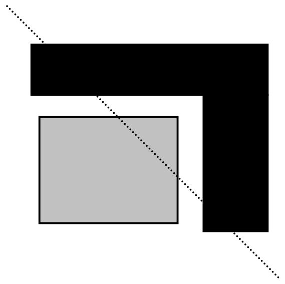

Fig. 1. Classification of two subjective classes with supervised learning

Although we only give a linear classifier example to demonstrate the disadvantage of traditional supervised learning, we believe this phenomenon happens in nearly all traditional supervised learning algorithms, such as single layer perceptron, multilayer perceptron, radial-basis-function networks, even the newly emerging support vector machines (SVM). For example, we need to find a kernel function for a SVM algorithm to calculate the separation surfaces of different classes [1]. Polynomial kernel functions are usually used in SVM papers. What will happen if the real separation surfaces cannot be described properly with polynomial kernels? The same problem will happen just as we described in the linear separation example. That is why some experts in SVM suggest trying several different kernels, and choosing one which has the best performance. These suggestions may help us in some specific learning tasks. But they are not general to many real-life learning tasks. Assume we want to teach a robot to learn human languages with current support vector machine techniques. Suppose the good kernel for learning Chinese is A, and the good kernel for learning English is B, etc. Unfortunately, we do not know which kernel is good for which language. In this situation, we need to stop the robot frequently to try different kernels. What if it is good to separate some English words with kernel A, and separate some other words with kernel B? It will be a really difficult problem in the real world.

Supervised learning sometimes also meets difficulties in collecting labeled data. For example, a multi-sensory robot may acquire input data at a very fast speed. It may acquire a new utterance in seconds; it may also acquire a huge amount of shape and color information in milliseconds. If we ask a human teacher to label all the data a robot acquires, the work is very labor intensive and tedious. Even when we have enough manpower and time to do the labeling work, we may also meet problems in the real world. In our real life, not everything can be clearly labeled. For example, it is hard for us to label hot and cold clearly. When we see an object which is a cross between a loveseat and a bed, it is also hard for us to label it with a name. These difficulties limit the applications of supervised learning to some extent.

In short, supervised learning is limited in many aspects. Blindly using the supervised learning technique is just like fitting our feet into unknown shoes. It is very hard to guarantee a fit. Due to this unpleasant situation, it is better for us to search for some help from another learning paradigm – unsupervised learning.

1.2 Unsupervised Learning

The unsupervised learning paradigm has no external teacher to oversee the training process, and the system forms “natural grouping” of the input patterns. The advantage of this paradigm is that “Natural” is always defined explicitly or implicitly in the clustering system itself [9]. In other words, the finally achieved results with unsupervised learning reflect the input data itself more objectively. The disadvantage of unsupervised learning is that the finally achieved objective classes are not necessary to have subjective meanings. Fig 2 demonstrates an unsupervised learning example for a better understanding of this technique.

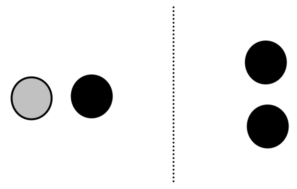

Fig. 2. Classification of two subjective classes with unsupervised learning

In Fig 2, the meanings of dark regions and light regions are similar to those meanings we used in Fig 1. In this figure, an unsupervised learning algorithm (e.g. K-means clustering) gives an objective boundary with no subjective meanings. So, if we use an unsupervised learning algorithm, such as K-means clustering, to separate the data in Fig 2, we may also meet problems.

Some readers may argue that we can separate the feature space into tiny volumes so that parts of these boundaries may have subjective meanings. The question here is how do we know if the algorithm already separates the feature space fine enough and put the classification boundaries at the right places? This is a hard problem that always shadows unsupervised learning. It makes unsupervised learning techniques not as powerful as some of us expected.

1.3 The Trend of Mixing Two Paradigms

Both supervised learning and unsupervised learning paradigms have limitations to achieve good classification results. But their limitations are complementary. Due to this reason, an interesting topic that we should consider is how to avoid as many limitations as possible without losing their major advantages. One possible solution to achieve better classification results is to use classification boundaries from unsupervised learning to approximate the meaningful boundaries expected by supervised learning. Although we rarely saw this idea formally presented in literature, it is implicitly used in several successful algorithms for improving the classification results, such as SOM (Self-Organizing Map) with LVQ (Learning Vector Quantization) [5, 6], GNG (Growing Neural Gas) with LVQ [2, 3], HMM model clustering and splitting [7, 8], and HMM model enhanced with mixture Gaussian distribution [8] etc. All these mentioned algorithms are sorts of mixtures of supervised-learning and unsupervised-learning. The superior achievements of these algorithms reflect an emerging technical trend of combining supervised learning and unsupervised learning in many desired situations.

Although several algorithms imply the merging trend of these two paradigms, these two paradigms are not well blended in the existing algorithms. If we check the detailed implementations of these algorithms, we can easily find that these two paradigms are used separately in all these algorithms. The mixtures of these two paradigms do help us to address the mentioned problems to some extent. But the separated implementation of supervised and unsupervised learning keeps those problems present. For example, we still do not know how many nodes we should use in a SOM before we test several map sizes, and we still do not have enough flexibility to shape the classification boundary according to the task requirement. We believe a successful classifier should adapt the number of clusters based on the task requirement, and permit us to refine the classification boundaries to any level as we want. This idea drives us to design an interactive learning algorithm through properly combining the supervised and unsupervised learning paradigms.

2.An Interactive Learning Algorithm

The algorithm we present here is an interactive and incremental learning algorithm. Equipped with this algorithm, a computer may learn and understand human concepts through its interactions with humans. This algorithm combines supervised and unsupervised learning philosophy through simulating a human's learning process. Its details can be described with set operations. The basic notations we use in our algorithm are briefly described below before the algorithm is presented for better understanding. The goal of these set operations is to increase the knowledge of the computer, and to clarify concepts in the computer.

- K is the knowledge set of the inputs.
- ~K is the non-knowledge set of the inputs. $K \cap \sim K = \Phi$. Φ is an empty set.
- U is the whole input space. $U = K \cup \sim K$.
- K_i is a subset of K. *i* is the index of subset K_i. $K = \cup K_i$, $K_i \cap K_j = \Phi$ for $i \neq j$.
- X is used to represent a data input.
- Y is used to represent a label input.
- A is used to represent an output.
- Y has a fixed relation with A.
- K_i can be represented with five parameters: center C_i, cover range CTH_i, hit times T_i, hit times' threshold TTH_i, and $P(Y_j|K_i)$, where "i" and "j" are indices. The cover range of different subsets can overlap, but one input can only belong to one subset. When an input falls into the cover range of a subset, we say the subset is hit. Many subsets can be hit by one input at the same time when the training begins.

The algorithm can be described with the following 4-step procedure:

1. If $X \in \sim K$, add a new subclass K_m to K. (Before K_m is added to K, there are m-1 subclasses in K). The center of this new subclass will be set at the unknown input. The cover range will be initialized to a large predefined value. The hit times will be initialized to 1. This step simulates human's neuron formation process.
2. If $\forall i$, $\|X-C_i\|<CTH_i$ and $T_i \leq TTH_i$ then set $C_i=(T_i*C_i+X)/(T_i+1)$, and $T_i=T_i+1$. If $\|X-C_i\|<CTH_i$ and $T_i>TTH_i$ then set $C_i=(T_i*C_i+X)/(T_i+1)$, $CTH_i=CTH_i/2$, $T_i=1$. Calculate the conditional probability $P(Y_j|K_i)$ through cumulatively counting related label inputs. This step records the hit times of every subclass, and shrink the cover range of a subclass when it is hit too many times. It also updates the subclass center and "meaning" – action related conditional probability $P(Y_j|K_i)$. We use all these operations to simulate the neuron separation and growing process in the human brain. This also makes it easy for interactive learning.
3. Iterate steps 1 and 2 till the algorithm finds $X \in K_j$, where $j=\text{argmin}_r\{|X-C_r| : |X-C_r|<CTH_r\}$. This is the semantic firing step. After this step, the input is related to a subclass which center has the nearest distance to the input *within all hit subclasses*, and give an output according to the conditional probability estimation. This step simulates the neuron firing process.
4. If the computer does not respond correctly to an input feature, keep feeding it with similar data input and label input till it "understands" the meaning.

In this algorithm, human's supervision guides the unsupervised learning process to put more strength on ambiguous data learning instead of learning all data uniformly. This property makes the unsupervised learning more meaningful than before. On the other hand, supervised learning in this algorithm uses multi-scale (different size hyper-spheres and hyper-planes) separation boundaries from unsupervised learning to construct its meaningful boundaries piece by piece instead of using a fixed form boundary to limit its classification ability.

The key idea of this algorithm is to use misclassified examples to assist the meaningful boundaries' construction according to the teacher's requirements. With this algorithm, the classification boundaries are not limited by fixed forms. The unsupervised learning procedure may also adapt its learning process according to different learning requirements. We introduce a human teacher in our learning example, but it is not a must for this algorithm. In offline training tasks, we may use a computer to fulfill the human's role, and do it in the traditional way. No matter if the algorithm is used online or offline, the good adaptation properties are not changed.

Equipped with this learning model, a computer should be able to learn any unknown feature distributions, including convex feature sets, concave feature sets etc…. Through interacting with humans and its surroundings, the computer should be able to adjust the knowledge subclass boundaries according to subjective meanings.

The above algorithm simulates the neuron evolution process in human brain. It also coincides with our daily experience. For example, we always try to practice more on some confusing words in our primary schools. Maybe those words are not used very frequently in our daily life, but we do spend more effort to avoid confusion. This is different from some existing classification systems. In those systems, the probabilities are estimated according to our daily life without considering the confusing cases in our learning processes.

Through some detailed considerations and analysis of the algorithm, we also find that this algorithm is more general than a simple VQ based classifier. When we set all cover ranges to infinity, this algorithm degenerates to a simple VQ based classifier. This algorithm is also related to the SVM idea and the K-nearest–neighbor estimation algorithm. In a SVM algorithm, researchers try to use "support vectors" to define the classification boundaries. In our algorithm, we allocate more subclasses for hard-to-separate data. In general, the hard-to-separate data is near the classification boundaries. Compared with the K-nearest-neighbor estimation algorithm, our algorithm not only use the probability-estimation-window to estimate the conditional probability (related to the subjective meaning of a subclass), we also use the "window" boundaries as the classification boundaries. Compared with Bayesian decision approaches, the proposed algorithm saves us from calculating covariance matrices with insufficient data. The algorithm is also relatively simple for an on-line real-time learning task. A detailed description of these comparisons will be discussed in a longer paper.

3.Experiments

We are presently using our algorithm to enable a mobile robot to learn spoken languages. The system structure is described below. The algorithm we present in this paper is concentrated on the system classifier.

MIC → Ring Buffer → Loud Sound Detection → End Point Detection → Preemphasis → Autocorrelation → LPC → Cepstral Coefficient → Liftering → Time Warping → **Classifier** → Action Command → Robot

With this system configuration, we may teach the robot short utterances through our interactions with the robot. For example, suppose we want to teach the robot the words "forward" and "back". At the beginning, the robot cannot decide if there is a difference in meaning between these two sounds. So, it is possible to move forward when we say "back". This means the data of sound "back" falls into the cover range of a wrong subclass, whose meaning is to instruct the robot to move forward. Our algorithm can recover from this problem in the following way. In this case, the right thing for us to do is to say "back" again to the robot, and label the audio input through touching the related sensor when we say "back". If the robot cannot understand us, repeat the same word and label it again. When we pronounce the same sound to the robot again and again, the corresponding subclass will be hit again and again, and its cover range will finally shrink. The center of the covered area will be changed gradually during the training. The conditional probability (related to semantic meaning) of the subclass will also be changed through the training procedure. After the cover range shrinks, the computer will not be able to find enough subclasses to represent intensively taught data. In other words, when we pronounce "back" or "forward" to the robot again, it is very likely that the robot would build a new subclass to represent the input data. The building of new subclasses may increase the local resolution for distinguishing different inputs. After we teach the robot the semantic meaning of a new subclass through touching related sensors, the robot will be able to distinguish the semantics of speech "forward" and "back". It is still possible for the robot to misunderstand "forward" and "back" with other sounds, but we can help it to clarify these meanings through our interactions with the robot.

Experiments with this algorithm on the mobile robot have been successful. We tried 6 commands (e.g. "forward", "back", "stop", etc….) together with speech of digits in English, Chinese, and Japanese. When the mobile robot starts to learn, it knows nothing about any sound as well as the meaning of any sound. Through interacting with its human teachers, the robot may acquire audio information on line, and relates different audio inputs to different actions according to human's teaching. The robot can distinguish these commands after we teach it for about ten minutes. Follow the interactive training, when we say a command to the robot in English, Chinese, or Japanese, it will move according to our commands.

4.Conclusion and Future Work

In this short essay, we first analyzed some problems of supervised learning and unsupervised learning paradigms. Then we briefly introduced our new algorithm that combines the philosophy of these paradigms for more flexible applications. Experimental results of the new algorithm are described at the end of the paper.

In the future, we want to try this interactive learning algorithm with more applications including offline learning. We also want to have more comparisons between the result of this novel algorithm and some existing algorithms of supervised and unsupervised learning through offline training. Finally, more concrete proof of the convergence of the algorithm is also what we expect in a longer paper.

References

1. C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Kluwer Academic Publishers, Boston, 1998.
2. B. Fritzke, Some Competitive Learning Methods, http://www.neuroinformatik.ruhr-uni-bochum.de/ini /VDM/research/gsn/.
3. B. Fritzke, Growing Cell Structure – A Self-Organizing Network for Unsupervised and Supervised Learning, Neural Networks, Vol. 7, No. 9, pp. 1441-1460, 1994.
4. S. Haykin, Neural Networks, Prentice-Hall, 1999.
5. J. A. Kangas, T.K. Kohonen, J.T. Laaksonen, Variants of Self-Organizing Maps, IEEE Transaction on Neural Networks, Vol. 1, No. 1, March 1990.
6. T. Kohonen (1995), *Self-Organizing Maps*. Springer, 1995.
7. K.F.Lee, Automatic Speech Recognition – The Development of the SPHINX System, Kluwer Academic Publishers, Boston, 1989.
8. L. Rabiner, B.H. Juang, Fundamentals of Speech Recognition, Prentice-Hall, 1993.
9. D.G. Stork, R.O.Duda, P.E. Hart, Pattern Classification and Scene Analysis, unpublished book section.