Support Vector Machine for Learning in Artificial Intelligence Systems (SVM LAIS)

Prof.Chandan Kolvankar*, Mr.Jekin Trivedi**, Mr.Bharat Mani***,
Mr. Ravi Ramanathan****, Ms. Sneha Kadam*****

*(Head of Department, Department of Information Technology, VCET, Mumbai University,
Email: cvkolvankar@yahoo.co.in)
** (Department of Information Technology, VCET, Mumbai University,
Email: jekintrivedi@gmail.com)
*** (Department of Information Technology, VCET, Mumbai University,
Email: bharatmani.mumbai@gmail.com)
**** (Department of Information Technology, VCET, Mumbai University,
Email: gnrravi@gmail.com)
***** (Department of Information Technology, VCET, Mumbai University,
Email: snehakadam85@gmail.com)

ABSTRACT

SVM LAIS stands for Support Vector Machine for learning in Artificial Intelligence System. SVM is a concept in computer science for a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The standard SVM takes a set of input data and predicts, for each given input, which of two possible classes the input is a member of, which makes the SVM a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. The project aims to create an architecture which can be the baseline for learning in Artificial Intelligence system. It involves researching various algorithms and papers to find the best suitable candidate for the job and implementation it in OpenCog (software framework for artificial intelligence programs).

1. INTRODUCTION

SVM is a major concept in the field of Machine Learning. The project aims to make this concept available to Artificial Intelligence Development Environment by creating SVM learning libraries for OpenCog (which is an Open Source AI development platform).

The crux of the project is to use SVM to learn a model distinguishing fit from unfit program trees and to use this model as a filter on generated program trees, to determine which ones are worth fitness-evaluating. SVMs can be applied to complex data types beyond feature vectors (e.g. graphs, sequences, relational data) by designing kernel functions for such data, SVM techniques have been extended to a number of tasks such as regression, principal component analysis and many others including Pattern recognition (Image and text), Bioinformatics` (Protein classification, Cancer classification).

2. REVIEW OF LITERATURE

The OpenCog Framework currently utilizes Bayesian Net modelling methods for solving a problem and finding the optimum solution for the given problem. As per as Moshe Looks' PHD thesis (given in reference [3]), the development of Bayesian net models followed a heuristic path, with applications and extensive experimentation preceding theory. In contrast, the development of SVMs involved sound theory first, then implementation and experiments. A significant advantage of SVMs is that whilst Bayesian net models can suffer from multiple local minima, the solution to an SVM is global and unique. Two more advantages of SVMs are that that have a simple geometric interpretation and give a sparse solution. Unlike Bayesian net models, the computational complexity of SVMs does not depend on the dimensionality of the input space. Bayesian net models use empirical risk minimization, whilst SVMs use structural risk minimization. The reason that SVMs often outperform Bayesian net models in practice is that they deal with the biggest problem with Bayesian net models, SVMs are less prone to overfitting". In contrast to Bayesian Net modelling, SVMs automatically select their model size (by selecting the Support vectors).

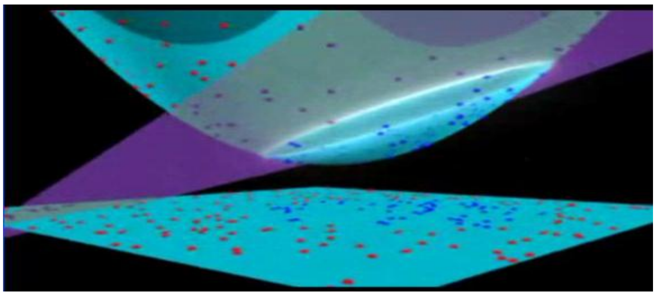

Figure 1: Mapping of linear data to polynomial scale

3. WORKING OF SVM:

Classifying data is a common task in machine learning. A support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite- dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. The goal of SVM modeling is to find the optimal hyperplane that separates clusters of vector in such a way that cases with one category of the target variable are on one side of the plane and cases with the other category are on the other size of the plane. The vectors near the hyperplane are the *support vectors*.

Suppose we are given the following positively labeled data points in X:

{(3,1),(3,-1),(6,1),(6,-1)}

And the following negatively labeled data points in X:

{(1,0),(0,1),(0,-1),(-1,0)}

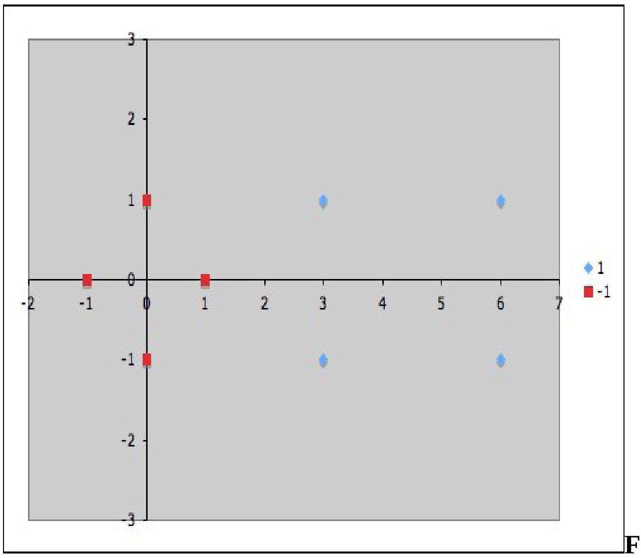

Figure 2: Blue diamond are positive data points and Red diamonds are negative data points

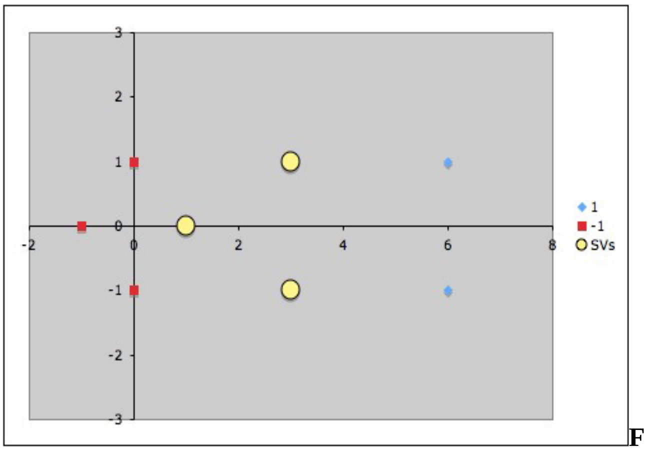

Figure3: Three Support Vectors are marked as yellow circles

We would like to discover a simple SVM that accurately discriminates the two classes. Since the data is linearly separable, we can use a linear SVM (that is, one whose mapping function Φ() is the identity function). By inspection, it should be obvious that there are three support vectors.

$$\{s_1 = (1, 0), s_2 = (3, 1), s_3 = (3, -1)\}$$

In what follows we will use vectors augmented with a 1 as a bias input, and for clarity we will differentiate these with an over-tilde. So, if s_1= (10), then $\hat{s}_1$ = (101). Figure 4.1.6 shows the SVM architecture, and our task is to find values for the α_i such that

$$\alpha_1\Phi(s_1).\Phi(s_1) + \alpha_2\Phi(s_2).\Phi(s_1) + \alpha_3\Phi(s_3).\Phi(s_1) = -1$$

$$\alpha_1\Phi(s_1).\Phi(s_2) + \alpha_2\Phi(s_2).\Phi(s_2) + \alpha_3\Phi(s_3).\Phi(s_2) = +1$$

$$\alpha_1\Phi(s_1).\Phi(s_3) + \alpha_2\Phi(s_2).\Phi(s_3) + \alpha_3\Phi(s_3).\Phi(s_3) = +1$$

Since for now we have let Φ() = I, this reduces to

$$\alpha_1\hat{s}_1.\hat{s}_1 + \alpha_2\hat{s}_2.\hat{s}_1 + \alpha_3\hat{s}_3.\hat{s}_1 = -1$$

$$\alpha_1\hat{s}_1.\hat{s}_2 + \alpha_2\hat{s}_2.\hat{s}_2 + \alpha_3\hat{s}_3.\hat{s}_2 = +1$$

$$\alpha_1\hat{s}_1.\hat{s}_3 + \alpha_2\hat{s}_2.\hat{s}_3 + \alpha_3\hat{s}_3.\hat{s}_3 = +1$$

to get a better algorithmic presentation.

4. IMPLEMENTATION:

Working on the implementation we have used SVMs to do non-linear classication. In particular, we have used SVM with Linear kernels on datasets that are linearly separable and Gaussian kernels on datasets that are not linearly separable.

Types of Kernel used with SVM:

Φ= $x_i * x_j$

(r$x_i * x_j$ + coeffiecient)

exp(-r$|x_i - x_j|$)

tanh(r$x_i x_j + coeffecient)$

OpenCog is an open-source software project which aims to achieve AGI. In OpenCog the system describes each and every attribute for an object and environment. This Knowledge is stored in form of SMEPH. In this system one could create a population of knowledge-sharing agents that operate as a sort of "Borg mind," with intelligence beyond what any of the agents could achieve with its own resources, with each giving contextually appropriate expression to the same knowledge and intelligence. These agents can use the SVM algorithms to classify the knowledge in the SMEPH. One can think of the Gaussian kernel as a similarity function that measures the distance" between a pair of examples, (x(i); x(j)). The Gaussian kernel is also parameterized by a bandwidth parameter, σ , which determines how fast the similarity metric decreases (to 0) as the examples are further apart.

The Gaussian kernel is defined as:

Kgaussian($x^{(i)}, x^{(j)}$) = exp(-$\frac{||x^{(i)}-x^{(j)}||^2}{2\sigma^2}$)

=exp(-$\frac{\sum_{k=1}^{n}(x_k^{(i)}-x_k^{(j)})2}{2\sigma^2}$)

Linear Kernel

The Linear kernel is the simplest kernel function. It is given by the inner product <x,y> plus an optional constant c. Kernel algorithms using a linear kernel are often equivalent to their non-kernel counterparts, i.e. KPCA with linear kernel is the same as standard PCA:

k(x,y)=$x^T y$ + c

Previously we have helped in development of Web Module of OpenCog system. This work done was the creation of an adapter between OpenCog server and the web module. This helps in easy viewing and creation of AtomSpace through using HTML 5 and jquery for making the interaction eye candy and interactive.

Execution no	C	Sigma	Prediction error
1	10.000	30.0000	0.1850
2	30.000	0.0100	0.5650
3	30.000	0.0300	0.0800
4	30.000	0.1000	0.0700

5. RESULTS

The following table shows the iterative formulation of the the values of sigma and C with prediction error for each set of values in the test set. With each iteration it tries to find the best suitable value of C & sigma for optimal prediction.

To test the SVM algorithm in a real world application we are using a Linear SVM to detect SPAM from emails. The SVM has access to a wordlist which denotes the probability of a word being part of a spam (e.g. click, guarantee, winner, etc). This is being used by SVM to generate a feature vector and non zero entities. The program then outputs the training accuracy (The accuracy during training with training set) and the test accuracy (The accuracy during testing with test set).

Test	Accuracy (in percent)
Test accuracy	99.850000
Training accuracy	98.8000

As we can see here the SVM algorithm gives high prediction accuracy for detecting SPAM from emails.

6. CONCLUSION

Our ultimate objective is to apply SVM to complex data types beyond feature vectors (e.g. graphs, sequences, relational data) by designing kernel functions for such data. thus making tasks such as regression, principal component analysis, classification, easy to work with in the OpenCog Development Environment.

Further work is intended and expected to occur on SVM LAIS as it is a part of Open Source Movement and the development work on AI is going to continue for some more decades or so.

REFERENCES

[1] R. Kurzweil. The Singularity Is Near. Penguin Press

[2] Goertzel, Ben and Cassio Pennachin, Editors. Artificial General Intelligence. Springer, New York

[3] http://metacog.org/papers/aaai04.pdf for Moshe Looks' PHD thesis on SVM

[4] Samsonovich, A. V. (2006). Biologically inspired cognitive architecture for socially competent agents. In Upal, M. A., & Sun, R.(Eds.). Cognitive Modeling and Agent-Based Social Simulation:Papers from the AAAI Workshop. AAAI Technical Report, vol. WS-06-02, pp. 36-48. Menlo Park, CA: AAAI Press

[5] R. Hecht-Nielsen R (2005) Cogent confabulation. Neural Networks 18:111-115

[6] W Duch, Oentaryo R.J, Pasquier M, Cognitive architectures: where do we go from here? Proceedings of AGI 2008, IOS Press

[7] P Wang, Rigid Flexibility: The Logic of Intelligence, Springer