International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 18 (2015) pp 39208-39210

Comparison Of Case Based Reasoning And Support Vector Machines Algorithm

Ponni. J
Research Scholar of Vels University Department of Computer Science & Engineering Ponni. sakthivel@gmail. com
.
Dr. Shunmuganathan. K. L
HOD of Computer Science & Engineering R. M. K. Engineering College, Chennai klsnathan@gmail. com

Abstract
We present a paper for comparison of Case Based Reasoning and Support Vector Machines Algorithm(SVM). Text Categorization(TC) is an important component in many information organization and information management tasks. Support Vector Machines are powerful machine learning systems, which combine remarkable performance with an elegant theoretical framework. The SVMs well fits the Text Categorization task due to the special properties of text itself. Experiments show that the SVM improves clustering performance by focusing attention of Support Vector Machines onto informative subspaces of the feature spaces. Textual case-based reasoning approach allows to integrate both textual and domain knowledge aspects in order to carry out text categorization. CBR solves a new problem by identifying its similarity to one or several previously solved problems stored in a case base and by adapting their known solutions. Cases of our framework are created from text. The results indicate that SVM performances much better than CBR method without character data. Considering the running time, the choice of CBR or SVM method for text categorization should be based on the number of samples and variables.

Keywords-CBR, SVM, Text Categorization, support vector, case based reasoning.

I. INTRODUCTION
In machine learning support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. 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 other, making it a non-probabilistic binary linear classifier. 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. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.
In case-based reasoning (CBR) systems expertise is embodied in a library of past cases, rather than being encoded in classical rules. Each case typically contains a description of the problem, plus a solution and/or the outcome. The knowledge and reasoning process used by an expert to solve the problem is not recorded, but is implicit in the solution.
To solve a current problem: the problem is matched against the cases in the case base, and similar cases are retrieved. The retrieved cases are used to suggest a solution which is reused and tested for success. If necessary, the solution is then revised. Finally the current problem and the final solution are retained as part of a new case.

II. SUPPORT VECTOR MACHINE ALGORITHM
A support vector machine constructs a hyper plane or set of hyper planes 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 hyper plane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. Whereas the original problem may be stated in a finite dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space.
To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function $\mathbf{K(x, y)}$ selected to suit the problem. The hyperplanes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters $\boldsymbol{\alpha}_i$ of images of feature vectors that occur in the data base. With this choice of a hyperplane, the points $\mathbf{x}$ in the feature space that are mapped into the hyperplane are defined by the relation:

$$\sum_i \alpha_i K(x_i, x) = \text{constant}$$

Note that if $\mathbf{K(x, y)}$ becomes small as $\mathbf{y}$ grows further away from $\mathbf{x}$, each term in the sum measures the degree of closeness of the test point $\mathbf{x}$ to the corresponding data base point $\mathbf{x_i}$. In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points $\mathbf{x}$ mapped into any hyperplane can be quite convoluted as a result, allowing much more complex discrimination between sets which are not convex at all in the original space.

Representation of classes and Hyperplane

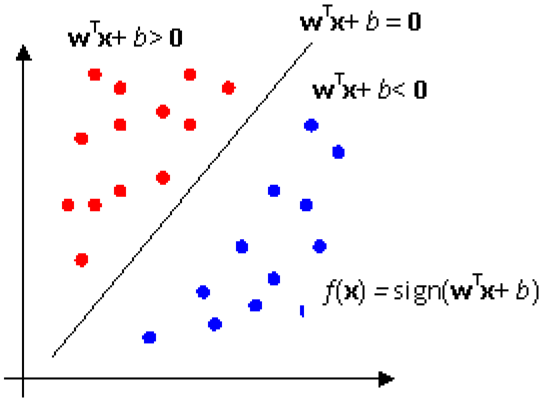

Figure (a)

Normal Vector for Hyperplane

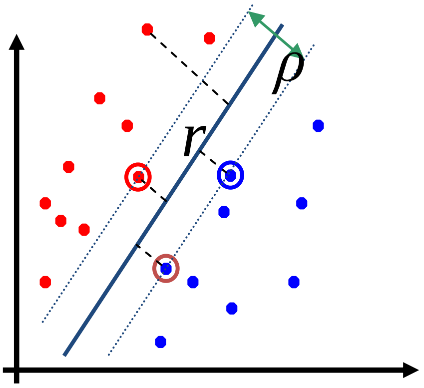

Figure (b)

III. CASE BASED REASONING ALGORITHM

A critical issue in case-based reasoning (CBR) is to retrieve not just a similar past case but a usefully similar case to the problem. For this reason, the integration of domain knowledge into the case indexing and retrieving process is highly recommended in building a CBR system. However, this task is difficult to carry out as such knowledge often cannot be successfully and exhaustively captured and represented.
The case based reasoning (CBR) process relies on three main operations: retrieval, adaptation and case memorization. Adaptation is at the heart of the CBR process and plays a central role.

Case-based reasoning has been formalized for purposes of computer reasoning as a four-step process:[1]

1. **Retrieve:** Given a target problem, retrieve from memory cases relevant to solving it. A case consists of a problem, its solution, and, typically, annotations about how the solution was derived. For example, suppose Fred wants to prepare blueberry pancakes. Being a novice cook, the most relevant experience he can recall is one in which he successfully made plain pancakes. The procedure he followed for making the plain pancakes, together with justifications for decisions made along the way, constitutes Fred's retrieved case.
2. **Reuse:** Map the solution from the previous case to the target problem. This may involve adapting the solution as needed to fit the new situation. In the pancake example, Fred must adapt his retrieved solution to include the addition of blueberries.
3. **Revise:** Having mapped the previous solution to the target situation, test the new solution in the real world (or a simulation) and, if necessary, revise. Suppose Fred adapted his pancake solution by adding blueberries to the batter. After mixing, he discovers that the batter has turned blue – an undesired effect. This suggests the following revision: delay the addition of blueberries until after the batter has been ladled into the pan.
4. **Retain:** After the solution has been successfully adapted to the target problem, store the resulting experience as a new case in memory. Fred, accordingly, records his new-found procedure for making blueberry pancakes, thereby enriching his set of stored experiences, and better preparing him for future pancake-making demands.

CBR cycle

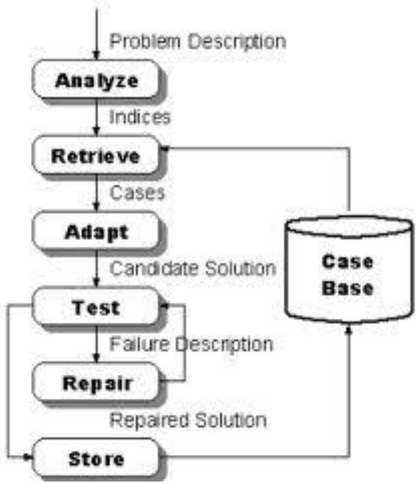

Figure (c)

IV. COMPARISON

Multi-Agent System with Case based reasoning:

Multi-Agent System technique is employed in CBR system for the purpose of retrieving, reusing, adapting the cases in CBR System. This section explores a framework that integrates the multi-agent and case-based reasoning techniques to support the dynamic and problem-oriented knowledge sharing among supply chain members. The framework is characterized by differential knowledge sharing levels depending upon the applications as well as the knowledge creation and reuse based on the previous knowledge in the problem area. It also provides a new tool for the field of inter-organizational knowledge management.

The CBR system retrieves cases relevant to the present problem situation from the case base and decides on the solution to the current problem on the basis of the outcomes from previous cases CBR System based on MA system consists of the following main agents:

Retriever Agent

When a new problem is entered into a case based system, a retriever decides on the features similar to the stored cases. Retrieval is done by using features of the new cases as indexes into the case base.

Adapter Agent

An adapter examines the differences between these cases and the current problem. It then applies rules to modify the old solution to fit the new problem

Refiner Agent

A refiner critiques the adapted solution against prior outcomes. One way to do this is to compare it to similar solutions of prior cases. If a known failure exists for a derived solution, the system then decides whether the similarities are sufficient to suspect that the new solution will fail.

Executer Agent

Once a solution is critiqued, an executer applies the refined solution to the current problem.

Evaluator Agent

If the results are as expected, no further analysis is made, and the cases and its solution are stored or use in future problem solving. If not, the solution is repaired.

Multi-Agent System with Support Vector Machines:

User interface agent

This agent is mainly used for interaction between the user and the system. The communication between users and the system is finished though the user interface agent. The users don't have to communicate with other agents. Communications of the users are always through the user interface agent. The alternation between the users and the system is realized using the user interface agent.

Task management agent

All the task of establishing, managing, start-up and executing the data mining project is done through the task management agent. The handling steps required by the data mining project like data picking up method, data pre-handling method, data dispersing method, data mining algorithm and so on can be packed into the data mining project using the task management agent. This agent manages the mining activity, start-up the mining project and also records the state information about the mining execution.

Data mining agent

To analyze the local database, the data mining agent is used. It got three functions, data picking up, data pre-handling and data mining activities. Different sub agents are used to express the functions. The first two functions are data preparing functions, which is very important in the data mining function.

V. CONCLUSION

The results indicate that SVM performances much better than CBR method without character data. Considering the running time, the choice of CBR or SVM method for text categorization should be based on the number of samples and variables. SVM consistently achieve good performance on text categorization tasks, outperforming existing methods substantially and significantly. With their ability to generalize well in high dimensional feature spaces, SVMs eliminate the need for feature selection, making the application of text categorization considerably easier. Another advantage of SVMs over the conventional methods is their robustness. SVMs show good performancein all experiments, avoiding catastrophic failure, as observed with the conventional methods on some tasks. Furthermore, SVMs do not require any parametertuning, since they can and good parameter settings automatically. All this makes SVMs a very promising and easy-to-use method for learning text classifiers from examples.

REFERENCES

[1] K. D. Althoff, R. Bergmann, and L. K. Branting, *(ICCBR-99). Case-Based Reasoning Reasearch and Development, Third International Conference on Case-Based Reasoning*, Lecture Notes in Artificial Intelligence 1650, Springer, Berlin, 1999.

[2] R. Bergmann and W. Wilke, *(ECAI-98)*. Towards a New Formal Model of Transformational Adaptation in Case-Based Reasoning in *Proceedings of the 13th European Conference on Artificial Intelligence Brighton, United Kingdom*, ed., H. Prade, pp. 53–57, (1998).

[3] Gireesh Kumar. T, Vijayan. V. P, Dec 2007. A multi-agent optimal path planning approach to robotics environment. InInternational Conference on Computational Intelligence and Multimedia Applications, pp. 400-404.

[4] Poornaselvan, K. J, Gireesh Kumar. T, Vijayan. V. P, July 2008. Agent-based ground flight control using type-2 fuzzy logic and hybrid ant colony optimisation to a dynamic environment. 1st International Conference on Emerging Trends in Engineering and Technology, July 2008. pp. 343-48.

[5] Rajan, J. Saravanan, V. (200. A Framework of an Automated Data Mining System Using Autonomous Intelligent Agents, International Conference on Computer Science and Information Technology, 700-704.

[6] Seydim, A. Y, (1999). Intelligent Agents: A Data Mining Perspective, Dept. of Computer Science and Engineering, Southern Methodist University, Dallas, TX 75275.

[7]. Thuraisingam. B, (2000). DataMining: Technologies, Techniques, Tools, and Trends, CRC Press, 4-6.