

Training Support Vector Machines
with Particle Swarms

deur

Ulrich Paquet

Voorgelê as 'n deel van die vereistes vir die graad

Magister Scientiae

in die Fakulteit Ingenieurswese, Bouomgewing, en Inligtingstechnologie

Universiteit van Pretoria

November 2003

Die finansiële ondersteuning van die *National Research Foundation* (NRF) tot hierdie navorsing word hierdeur erken. Opinies en gevolgtrekkings in hierdie verhandeling is dié van die outeur, en kan nie noodwendig aan die NRF toegeskryf word nie.

Training Support Vector Machines with Particle Swarms

deur

Ulrich Paquet

Opsomming

Deelswerms kan met gemak gebruik word om 'n funksie, wat beperk word deur 'n stel lineêre beperkings, te optimeer. 'n "Lineêre Deelswermoptimeerder" en 'n "Konvergente Lineêre Deelswermoptimeerder" word ontwikkel om sulke beperkte funksies te optimeer. As die hele swerm aanvanklik slegs uit geldige oplossings bestaan, dan kan die swerm die beperkte funksie optimeer sonder om ooit weer die stel beperkings te oorweeg. Die Konvergente Lineêre Deelswermoptimeerder" oorkom die waarskynlikheid van vroegtydige konvergensie, wat deur die Lineêre Deelswermoptimeerder" getoon word. Om 'n Steunvektormasjien te leer moet 'n beperkte kwadratiese programmeringsprobleem opgelos word, en die Konvergente Lineêre Deelswermoptimeerder voldoen aan die behoeftes van 'n optimeringsmetode vir Steunvektormasjiene. Deelswerms is intuïtief en maklik om te implementeer, en word aangebied as 'n alternatief tot huidige metodes om Steunvektormasjiene te leer.

alternative to current methods Paquet, Ulrich. Training Support Vector Machines with Particle Swarms

Studieleier: Prof. A.P. Engelbrecht

Departement Rekenaarwetenskap

Degree: Magister Scientiae

Training Support Vector Machines with Particle Swarms

by

Ulrich Paquet

Preface

Abstract

The question that originally started the research in this thesis was: can a Particle Swarm Optimiser be used to train a Support Vector Machine, and if so, what would a Linear Particle Swarm Optimiser look like?

Particle swarms can easily be used to optimise a function with a set of linear equality constraints, by restricting the swarm's movement to the constrained search space. A "Linear Particle Swarm Optimiser" and "Converging Linear Particle Swarm Optimiser" is developed to optimise linear equality-constrained functions. It is shown that if the entire swarm of particles is initialised to consist of only feasible solutions, then the swarm can optimise the constrained objective function without ever again considering the set of constraints. The Converging Linear Particle Swarm Optimiser overcomes the Linear Particle Swarm Optimiser's possibility of premature convergence. Training a Support Vector Machine requires solving a constrained quadratic programming problem, and the Converging Linear Particle Swarm Optimiser ideally fits the needs of an optimisation method for Support Vector Machine training. Particle swarms are intuitive and easy to implement, and is presented as an alternative to current numeric Support Vector Machine training methods.

problems

The main contributions made by this thesis are:

1. An algorithm for SVM training, which is transformed into a constrained optimisation problem.
2. The development of LPSP for solving the SVM problem, based on the Linear Particle Swarm Optimiser.
3. A proof that LPSP is ideally suited for solving the SVM problem, under the condition needed for LPSP to always converge to a local minimum.
4. The extension of LPSP to CLPSP to provide a method for solving the SVM problem, and a proof that CLPSP will at least converge to a local minimum.

Supervisor: Prof. A.P. Engelbrecht

Department of Computer Science

Degree: Magister Scientiae

In a sense this thesis has delivered more than its original aim. The new PSO algorithms will probably be of greater importance to further advances in the Swarm Intelligence community, than its application to SVM training.

Chapter 2 poses SVM under the magnifying glass, and sees the more optimisation problem (a quadratic programming problem) that forms the backbone of this thesis. SVM poses a problem of its own, primarily because the training problem shows quadratic growth as the training set size increases. Methods for SVM training are discussed in Chapter 3, and a training algorithm, based on standard methods of decomposing the more optimisation problem into sub-problems, are used to construct a new SVM training algorithm. Chapter

The question that originally spurred the research in this thesis was - “can a Particle Swarm Optimiser be used to train a Support Vector Machine, and to what extent will it be successful?”

Training a Support Vector Machine (SVM) involves solving a quadratic programming problem, with a single linear constraint, and a set of non-negativity constraints. At first this problem seemed trivial - the objective function that needs to be minimised is convex, and the Particle Swarm Optimiser (PSO) will not be trapped in any local minima.

The difficulty in the problem arose with developing a method to handle the linear constraint. This has led to the development of the Linear (and Converging Linear) PSO algorithms (LPSO and CLPSO), which both have unique properties needed not only for handling the single linear constraint, but any set of (feasible) linear constraints. The non-negativity constraints have led to the extension of both new Particle Swarm algorithms to include cases when constraints appear as boxed constraints. With the addition of slack variables to an optimisation problem with linear constraints, it becomes possible to solve any of these problems.

The main contributions made by this thesis are therefore:

1. An algorithm for SVM training, which is based on analysis of a method for decomposing the SVM quadratic programming problem.
2. The development of LPSO for general optimisation problems, and a proof of a condition on the initial swarm guaranteeing that any point in the search space can be reached.
3. A proof that LPSO is ideally suited for linearly constrained optimisation, with a precondition needed for LPSO to always satisfy linear equality constraints.
4. The extension of LPSO to CLPSO to preclude premature convergence, and a proof that CLPSO will at least converge to a local minimum.
5. The addition of a method to LPSO and CLPSO needed for inequality constraint handling, and the implementation of CLPSO with inequality constraints as an optimiser in SVM training.

In a sense this thesis has delivered more than its original aim. The new PSO algorithms will probably be of greater importance to further milestones in the Swarm Intelligence community, than its application in SVM training.

Chapter 1 puts SVMs under the magnifying glass, and sets the main optimisation problem (a quadratic programming problem) that forms the backbone of this thesis. SVM training has unique problems of its own, primarily because the training problem shows quadratic growth as the training set size increases. Methods for SVM training are discussed in *Chapter 2*, and a training algorithm, based on standard methods of decomposing the main optimisation problem into subproblems, are used to construct a correct training algorithm. *Chapter 3* introduces PSO as an optimisation algorithm, and discusses some of the recent advancements to the PSO method itself. The PSO is extended to handle constrained problems in *Chapter 4*, and LPSO and CLPSO are developed. This extension includes a rigorous analysis of the newly developed algorithms. The successes and failures of LPSO and CLPSO are empirically shown in *Chapter 5*. It is also shown how the CLPSO can be used to train SVMs from a very large character recognition dataset. Finally, *Chapter 6* provides an overview, and gives some thoughts for further research.

Many people have contributed to the successful completion of this thesis. Foremost, I am greatly indebted to professor Andries Engelbrecht for introducing me to the world of artificial intelligence, and for his patient guidance throughout my research.

Ulrich Paquet

Pretoria, South Africa

June 2003

Commit thy works unto the LORD, and thy thoughts shall be established. Proverbs 16:3

Contents

Preface	v
1 Support Vector Machines	1
1.1 Introduction to Support Vector Machines	1
1.2 Pattern recognition	3
1.3 Linear Support Vector Machines	3
1.4 Soft margin hyperplanes	7
1.5 Non-linear Support Vector Machines	8
1.6 Concluding	13
2 Support Vector Machine Training Methods	15
2.1 Introduction to Support Vector Machine training methods	15
2.1.1 Chunking	16
2.1.2 Decomposition	17
2.1.3 Sequential Minimal Optimisation	18
2.2 Conditions for optimality	18
2.3 A decomposition method	21
2.3.1 Optimality of the working set	23
2.3.2 Selecting the working set	23
2.3.3 Shortcuts and optimisations to the decomposition algorithm	27
2.4 The training algorithm	29
3 Particle Swarm Optimisation	32
3.1 Introduction to unconstrained optimisation	32
3.2 Introduction to Particle Swarm Optimisation	33
3.2.1 Global best (<i>gbest</i>)	35
3.2.2 Local best (<i>lbest</i>)	35
3.2.3 The PSO algorithm	36
3.2.4 Improvements	37

3.3	Concluding	39
4	Constrained Particle Swarm Optimisation	40
4.1	Introduction to constrained optimisation	40
4.1.1	Terminology	40
4.1.2	Expressing problems in the standard form	42
4.1.3	Slack variables	43
4.1.4	Convex optimisation	43
4.1.5	Duality	44
4.1.6	Equality-constrained optimisation	45
4.2	Linear Particle Swarm Optimisation	45
4.2.1	Criteria on the initial swarm	47
4.3	Equality-constrained optimisation	49
4.3.1	Current methods	49
4.3.2	PSO for equality-constrained optimisation	50
4.3.3	Overcoming premature convergence	56
4.3.4	Proof of convergence for CLPSO	58
4.4	Inequality-constrained optimisation	61
4.5	Concluding	64
5	Experimental results	65
5.1	Linear Particle Swarm Optimiser	65
5.1.1	Experimental results	65
5.1.2	LPSO and CLPSO Convergence characteristics	81
5.2	Support Vector Machine Training	81
5.2.1	Implementing the SVM training algorithm	81
5.2.2	Practical concerns and improvements	83
5.2.3	Experimental results	84
5.3	Concluding	88
6	Conclusion and Future Work	89
	Publications derived from this thesis	91
	Bibliography	92

List of Figures

1.1	An example of a classification problem in two dimensions, with the support vectors encircled.	4
1.2	Constructing an optimal hyperplane	5
1.3	An example of a linear separating hyperplane for the non-separable case. . . .	7
1.4	An example of two-dimensional classification. The three-dimensional feature space is defined by monomials x_1^2 , $\sqrt{2}x_1x_2$, and x_2^2 , where a linear decision surface is constructed. This construction corresponds to a non-linear ellipsoidal decision boundary in \mathbb{R}^2	9
1.5	Architecture of a Support Vector Machine: The input vector \mathbf{x} and the support vectors \mathbf{x}_i (in this example optical digits) are non-linearly mapped (by Φ) into a feature space \mathcal{F} , where dot products between their mapped representations are computed. By the use of the kernel k , these two steps are in practice combined. The results are linearly combined by weights α_i found by solving a quadratic program. The linear combination is then fed into a decision function f , which determines the classification of \mathbf{x}	12
1.6	Classifying with different kernel functions. The support vectors, with nonzero α_i , are shown with a double outline, and define the decision boundaries between the two classes.	13
2.1	Selecting a working set of size four.	25
4.1	Comparing the possible search spaces resulting from different initial swarms in LPSO, with $\mathbf{v}_i^{(0)} = \mathbf{0}$	47
4.2	Progressive reduction of the feasible domain.	50
4.3	Minimising f under a linear equality constraint.	51
4.4	Particles becoming a linear combination of each other.	64
5.1	Results of 100 <i>Genocop II</i> simulations on the constrained parabola f_1 defined in equation (5.2).	67

5.2	Results of 100 simulations of LPSO on the constrained parabola f_1 defined in equation (5.2).	68
5.3	Results of 100 simulations of CLPSO on the constrained parabola f_1 defined in equation (5.2).	70
5.4	Results of 100 <i>Genocop II</i> simulations on the constrained quadratic function f_2 defined in equation (5.3).	72
5.5	Results of 100 simulations of LPSO on the constrained quadratic function f_2 defined in equation (5.3).	74
5.6	Results of 100 simulations of CLPSO on the constrained quadratic function f_2 defined in equation (5.3).	75
5.7	Results of 100 <i>Genocop II</i> simulations on the constrained Rosenbrock function f_3 defined in equation (5.4).	77
5.8	Results of 100 simulations of LPSO on the constrained Rosenbrock function f_3 defined in equation (5.4).	78
5.9	Results of 100 simulations of CLPSO on the constrained Rosenbrock function f_3 defined in equation (5.4).	79
5.10	A few examples from the MNIST dataset.	85

List of Tables

Support Vector Machines

5.1	Results of 100 <i>Genocop II</i> simulations on the constrained parabola f_1 defined in equation (5.2), after 250 generations. ('chromosomes' is abbreviated as chrms.)	67
5.2	Results of 100 LPSO and CLPSO simulations on the constrained parabola f_1 defined in equation (5.2), after 250 iterations.	71
5.3	Results of 100 <i>Genocop II</i> simulations on the constrained quadratic function f_2 defined in equation (5.3), after 1000 generations. ('chromosomes' is abbreviated as chrms.)	73
5.4	Results of 100 LPSO and CLPSO simulations on the constrained quadratic function f_2 defined in equation (5.3), after 1000 iterations.	76
5.5	Results of 100 <i>Genocop II</i> simulations on the constrained Rosenbrock function f_3 defined in equation (5.4), after 2000 generations. ('chromosomes' is abbreviated as chrms.)	78
5.6	Results of 100 LPSO and CLPSO simulations on the constrained Rosenbrock function f_3 defined in equation (5.4), after 2000 iterations.	80
5.7	Influence of different working set sizes on the first 20,000 elements of the MNIST dataset	86
5.8	Scalability: training on the MNIST dataset	87