An expert system for automatically pruning vines

Sam Corbett-Davies[*]
University of Canterbury
New Zealand

Tom Botterill[†]
University of Canterbury
New Zealand

Richard Green[‡]
University of Canterbury
New Zealand

Valerie Saxton[§]
Lincoln University
New Zealand

ABSTRACT

Vine pruning is an important part of vineyard management, and pruning is the most expensive task in the vineyard which has not yet been automated. Every year, most new canes must be removed from the vine, and the choice of canes to retain impacts vine yield. To automate the process of vine pruning, a vine pruning robot must make decisions on what canes to remove or to keep, based on a 3D topological model of the structure of the vine. In this paper we present an Artificial Intelligence (AI) system for making these decisions, developed and evaluated using simulated vines. A viticulture expert evaluated our approach by comparing it to pruning decisions made by a pruner with a skill level typical of human pruners. Our system successfully pruned 30 % of vines better than the human and 89 % at least as well. These results demonstrate that the vine pruning problem is solvable using current computing technologies, and that automating the pruning process has the potential to improve vine quality and yield.

Categories and Subject Descriptors

I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems—*Industrial automation*; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding—*Perceptual reasoning*

Keywords

Agricultural automation, vine pruning, artificial intelligence, machine learning

1. INTRODUCTION

Computer vision and artificial intelligence techniques are advancing to the point where complex agricultural and horticultural tasks can be automated [13]. Automating these tasks requires the expert knowledge of humans to be computerised, which is usually achieved by developing an expert system [17, 12]. This paper describes the development of an an expert system for determining how to best prune vines, for use in a vine pruning robot. The system is developed to make pruning decisions based on a 3D topological model of vines from a computer vision vine modelling system, and determines the best way to prune a vine (henceforth known as a "pruning scheme"), based on knowledge, rules and training data collected from vine pruning experts.

We begin by investigating standard AI approaches in Section 3, including classification and search algorithms. Our completed AI system uses a cost function with parameters learned from examples of well-pruned vines. Brute force search over all possible pruning schemes for a vine is used to find the optimal one. Expert knowledge is used to design *features* of a pruning scheme such as the length and height of the canes, which form a low-dimensional representation of pruning decisions. Section 4.2 describes the development of the cost function for a pruning scheme, which is a linear combination of 12 features. The relative weightings of the features is determined using training data acquired from a pruning expert, as explained in Section 4.3. Adaptive random search is used to optimise the weightings to best match how the training vines are pruned.

In Section 5.1 we evaluate our AI algorithm against a pruner with a skill level typical of vineyard workers. 100 simulated vines were pruned by both our system and the human pruner, with a viticulture expert rating the quality of every decision. Our system outperformed the human and was well received by the viticulture expert. In Section 6 we discuss the implications of our research and how our system could be extended to real data.

2. PRUNING PRELIMINARIES

This section provides a background into the task of pruning vines sufficient to understand the rest of this paper. Figure 1 shows a simulated vine with relevant features labelled. *Vine* is used to refer to the entire plant, while a *cane* is a single branch. Pruning occurs in winter, at which time a number of canes are *tied down* to wires to become this sea-

[*] samcorbettdavies@gmail.com
[†] tom.botterill@grcnz.com
[‡] richard.green@canterbury.ac.nz
[§] valerie.saxton@lincoln.ac.nz

IVCNZ '12, November 26 - 28 2012, Dunedin, New Zealand
Copyright 2012 ACM 978-1-4503-1473-2/12/11 ...$15.00.

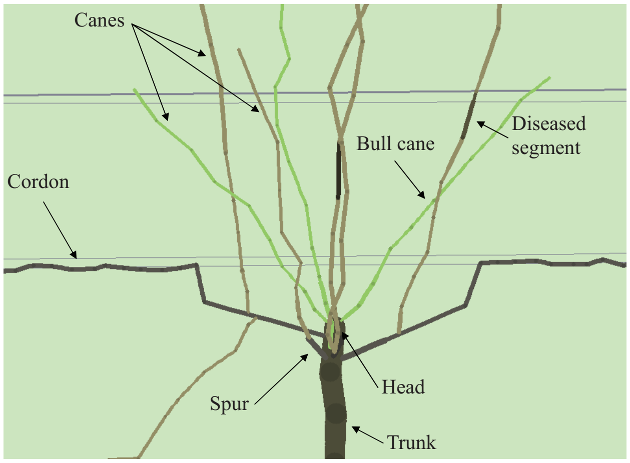

Figure 1: A computer-generated vine.

son's *cordons*. Canes that grow out of cordons bear fruit in summer. Pruning also involves cutting a few canes 10-20 cm from their base to become *spurs*, where the candidates to become next year's cordons will grow out of. "Spur" is also used as a verb to describe cutting canes in this way. The *head* of a vine is the wooden mass at the top of the trunk; canes that grow directly out of the head are known as *bull canes*. An example of a real pruned vine is shown in Fig. 2.

A typical pruning strategy is to leave one cane on either side of the head to be tied down to become a cordon. Another one or two canes (often the lowest to the ground) are spurred, and all other growths are pruned away. *The task of pruning vines can therefore be reduced to finding the best canes to spur and tie down for a given plant.* For brevity, in this paper we often refer to "canes to spur" as "spurs", and "canes to tie down" as "cordons".

2.1 Vine Simulator

The AI for the vine pruning robot has been developed in parallel with the robot's computer vision systems. However, the computer vision systems are still some time off being finished, so vine structure data from real world vines is not available. In order to facilitate the development of the AI without this data, we developed a vine simulator (see Fig. 1), which generates vine structures that are qualitatively similar to real vines. The simulated data is designed to have the same characteristics as what will be extracted by the robot's vision, but there is no guarantee this will still be the case when real data becomes available. Because of this we investigate general solutions to the AI problem.

3. EXPERT SYSTEMS

In this section we evaluate existing approaches to expert systems, and discuss their suitability to the problem of pruning vines. All AI algorithms require parameters to be set to determine how the algorithm operates. These parameters can either be designed using expert knowledge of the problem, or determined using machine learning algorithms on a training set of observations [16]. When parameterising a machine learning algorithm care must be taken to avoid overfitting and underfitting by choosing the simplest possible representation that captures the essence of the problem [18].

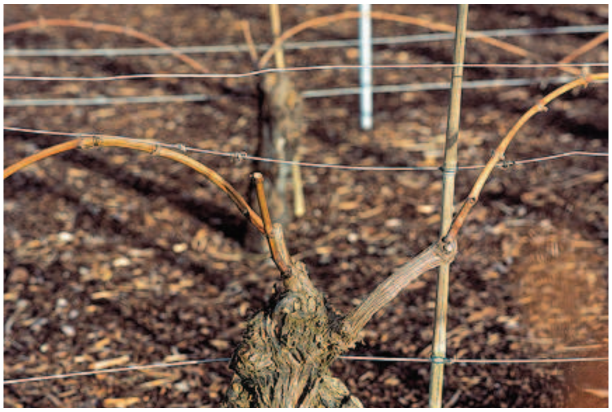

Figure 2: A close-up of the head of pruned vine, with two cordons and one spur clearly visible.

The most successful approaches to expert systems use domain knowledge to develop the structure of the algorithm, before parameterising it with large quantities of observed data [9]. This is because a huge number of features can be conceived for any data set, and only a small fraction of these will be relevant to predictions. Although algorithms have been conceived to automatically determine which features are relevant (a problem known as feature selection), for small problems it is simpler to utilise existing expert knowledge [8]. Section 6 discusses the use of such algorithms in future approaches.

3.1 Algorithms

The approaches investigated to solve the problem can be broadly classified as either classification or search algorithms. Classification algorithms could be used to classify individual canes as being good candidates to spur or tie down by developing a model of what constitutes such a candidate. These algorithms use the vector-space model, making decisions based on a vector of features of a cane, and therefore rely on the availability of suitable training data. They have the advantage of being fast, with running time proportional to the number of canes on a vine, but do not (natively) consider connections between vines when making decisions.

An alternative approach is to evaluate pruning schemes holistically, and use a search algorithm to find the best of all possible schemes for a vine. These algorithms rely on the development of a fitness functions to quantify the quality of a particular pruning scheme. The development of such a fitness function allows good pruning schemes to be identified using well-studied search algorithms over the space of all possible pruning schemes. Search algorithms have the advantage that they are better able to reason about relationships between vines, but can face running time exponential in the number of canes on a vine. Brute-force search was the only such algorithm we investigated, because it is feasible for a problem of this size (see Section 3.1.2).

A number of potential algorithms, including fuzzy logic and local search methods, were not considered for this problem.

3.1.1 Classification Algorithms

Our first approach used a decision tree [3] to classify canes, designed from pruning rules developed by viticulture experts. The implementation of this algorithm was very simple, so it was done to establish a worst-case performance benchmark. The tree performed poorly, adequately pruning less than 50 % of vines. The reason for this poor performance is that the rules were under-specified for vines without good candidates to spur and tie down. This problem, where experts do not (and often cannot) fully explain their decisions, occurs often in the development of expert systems [9]. However, even if the rules acknowledged every influencing factor, it was clear decision trees were too brittle to be successful in this problem. This is because there will always be situations where the algorithm must choose between two undesirable situations, and enumerating rules for every such decision is infeasible. We proceeded to investigate algorithms more capable of making more nuanced decisions.

Numerous classification algorithms could be applied to this problem, including decision trees [3], artificial neural networks [20], support vector machines [6] and random decision forests [2]. An advantage of these techniques is that they are extremely well-studied and proven on a wide variety of data sets [5]. Of particular interest are random forests, which Caruana *et al.* found to be the most accurate classification algorithm on many standard data sets [5].

Classification algorithms were ultimately deemed to be unsuitable for this problem because canes cannot be classified in isolation; for example, one cane may excessively shade another, making them good choices to tie down individually but bad together. Research into vine pruning suggested that that such interrelations between classification decisions would be prevalent, so these algorithms were not implemented.

3.1.2 Search Algorithms

The simplest way to select the best pruning scheme given a fitness function is to iterate through all possible pruning schemes and find the one with the best score. This is known as brute-force search, and quickly becomes impractical as the size of the problem grows [18]. However, selecting 4 canes to leave and 2 to spur from a vine of 20 canes requires only:

$$^{20}C_4^{16}C_2 = \frac{20!}{(20-4)!4!}\frac{16!}{(16-2)!2!} = 581400 \qquad (1)$$

cut sets to be evaluated, which could be achieved on a modern computer in well under a second. This suggests that the relatively small size of the vine-pruning problem makes a brute-force search feasible. In addition, this number could be significantly decreased by first applying some expert knowledge to remove from the search canes that will definitely be pruned. For instance, healthy canes close to the head will always be preferred over similar canes further away, and it can be safely assumed that a vine with an excessive number of canes will have enough good candidate canes that the outer ones can be ignored. Excluding pruning scheme from the search in this way turns a brute-force search into a heuristic search.

Search algorithms also have the advantage of being able to easily handle uncertainty. For example, if the robot's vision system can only determine features of vines in terms of probability distributions, the target for maximisation could be the expected value of the fitness function as opposed to its value.

The greatest challenge in the implementation of a search algorithm is the development of the evaluation function. Hand-designing such a function to measure the quality of a cut set will require a large amount of parameter tuning. For example, both weak canes and short canes are bad candidates to be tied down, but how would one quantify which is the worse pruning decision? Section 4.3 describes how we learned the parameters for an evaluation function from a set of simulated vines "pruned" by an expert.

3.2 Optimisation

"Optimization problems lie at the heart of most machine learning approaches" [1]. Many machine learning algorithms are designed to make certain optimisation methods effective; for example, artificial neural networks use backpropagation [16] while support vector machines use quadratic programming [6]. These optimisation methods make assumptions about the shape of the objective function, so are ineffective if these assumptions do not hold.

If the objective function cannot be assumed to be convex, continuous and/or differentiable, stochastic global optimisation strategies are most often used [4]. These methods are also known as Monte Carlo methods, and search for the global optimum by randomly exploring the parameter space according to some distribution. The simplest Monte Carlo method is Pure Random Search (PRS), which samples the parameter space uniformly. The solution returned by the algorithm is the best point found so far, which converges to the global optimum with probability tending to 1 as the number of iterations goes to infinity [19]. Adaptive random search methods change the sampling distribution based on previous search iterations, typically in order to focus the search on promising regions of the parameter space [11]. In practice, adaptive search methods converge faster than PRS, and will converge if the sampling distribution is always nonzero over the entire parameter space $\mathbb{P}$. We used adaptive random search for optimisation in our system.

More complex stochastic optimisation algorithms include genetic algorithms [10], simulated annealing [15], swarm optimisation [14] and tabu search [7]. These algorithms use various techniques to find new candidates in the parameter space, often combining local search with some means of escaping local optima. We elected not to implement these algorithms as they are difficult to implement and require parameter tuning in order to operate well [11]. Having achieved satisfactory results with random search, there was no need to implement a more complicated optimisation algorithm.

4. IMPLEMENTATION

This section describes the development of our expert system for automatically pruning vines.

4.1 Brute Force Search

As discussed in Section 3.1.1, classification-based algorithms were deemed to be unsuitable for this problem because of the tightly coupled nature of decisions about canes. Because of this, we pursued a search-based approach which considered pruning schemes as a whole. Having shown (in Section 3.1.2) that brute force search across all possible pruning schemes for a vine was feasible, we proceeded to develop this algorithm. The algorithm's task was to determine the best n spurs and m cordons on each side, selecting $2(n+m)$ canes in total.

First, a list of candidate canes was compiled. This included all canes out of the head, spurs or cordons, regardless of whether they were diseased or of otherwise dubious suitability. Also added to the list were $2(n+m)$ "null canes", to allow the search to include pruning schemes where less than $2n$ or $2m$ canes were spurred or tied down. The search was carried out using the combinations extension to the Boost C++ library, developed by Hervé Brönnimann[1]. This library allowed every pruning scheme (combination of spurs and cordons) to be iterated over and evaluated. This library was also used to enumerate every possible pruning scheme during the training phase.

4.2 The Cost Function

The next step was to develop a means of evaluating the quality of a pruning scheme, so that brute force search could be used to find the best scheme. To this end, expert knowledge was used to develop a set of features ($\mathbf{x} \in \mathbb{X}$) to parameterise a pruning scheme, which represented the most important factors in making pruning decisions. A brief summary of these features is shown in Table 1.

Table 1: Pruning scheme feature set. The features in italics are global to the pruning scheme; the others are aggregated over all cordons and spurs chosen.

Cordons	Spurs
Number missing	*Number missing*
Expected viable length	Probability viable
Height	Height
Distance from head	Distance from head
Angle from ideal	Angle from ideal
Bull cane?	Bull cane?

The 12 features that make up the feature vector for a pruning scheme include 2 global features and 5 local features each for the spurs and cordons chosen in the scheme. All features are normalised to values between 0 and 1, and designed so that they increase as the pruning scheme gets worse. A simple cost function was developed to describe the quality of a pruning scheme from its feature vector ($\mathbf{x}$). In order to minimise the number of parameters need to define the cost function, a linear form was used:

$$f_{\mathbf{p}}(\mathbf{x}) = \mathbf{p} \cdot \mathbf{x}. \tag{2}$$

The cost function is therefore a weighted sum of the features of a pruning scheme. Minimising the number of parameters in the cost function decreases the possibility of overfitting when the parameters are trained.

4.3 Training

The remaining challenge was to find the weightings of the features ($\mathbf{p}$) that result in the cost function that best reflects the decisions made by human experts. The principal expert on the project was Dr Valerie Saxton, a viticulture lecturer from Lincoln University. She provided the training examples by virtually pruning 100 simulated vines using 1-cane pruning ($n = m = 1$). A random selection of 80 of these vines were chosen as the training set, with the remaining 20 making up the validation set.

All we can infer with confidence from each pruned vine in the training set is that the pruning scheme as a whole is no worse than any other possible scheme. We therefore developed a training method to learn the cost function parameters ($\mathbf{p}$) from the training examples. This was framed as a global optimisation problem in the parameter space ($\mathbb{P}$). The optimisation problem can be expressed as:

$$\mathbf{p}^* = \operatorname*{argmax}_{\mathbf{p} \in \mathbb{P}} g(\mathbf{p}) \tag{3}$$

$$g(\mathbf{p}) = \frac{\sum_{i=1}^{80} |\{\mathbf{x} \in \mathbb{X}_i : f(\mathbf{x}, \mathbf{p}) \geq f(\mathbf{x_i^*}, \mathbf{p})\}|}{\sum_{i=1}^{80} |\mathbb{X}_i|}, \tag{4}$$

where $\mathbf{p}^*$ is the optimal parameter vector, $\mathbb{X}_i$ is set of feature vectors for all possible pruning schemes in training example i, and $\mathbf{x_i^*}$ is the feature vector for the scheme the expert chose. Concretely, $g(\mathbf{p})$ expresses, across all training examples, the percentage of pruning schemes the cost function $f_{\mathbf{p}}$ scores worse than the expert's scheme. We assume that all other pruning schemes *are* worse that the expert's, so we want to find the $\mathbf{p}$ that maximises $g(\mathbf{p})$.

Random search was used to solve (3). The search space $\mathbb{P}$ was the 12-dimensional hypercube with every element of $p \in [0, 100]$, and the search proceeded as follows:

1. Randomly sample $\mathbf{p}$ from $\mathbb{P}$ according to the distribution μ.
2. For every vine in the training set:
   (a) Compute the cost $f^* = f_{\mathbf{p}}(\mathbf{x}^*)$ for the scheme chosen by the expert.
   (b) For every other possible pruning scheme, compute the cost $f = f_{\mathbf{p}}(\mathbf{x})$.
   (c) Count the number of *incorrectly classified schemes*, that is, the number of schemes with a lower cost than than the expert's choice (ie $f < f^*$).
3. Sum the incorrectly classified schemes over all vines (e) and record the pair ($\mathbf{p}$, e).
4. Return to 1 and repeat a given number of times.
5. The optimal parameter vector $\mathbf{p}^*$ is the one paired with the lowest value of e.

Initially pure random search was used, with the distribution μ uniform over $\mathbb{P}$. An adaptive search method was introduced to try to accelerate convergence, where μ was changed after every 10000 iterations. For adaptive search μ was set such that points within the hyperrectangle bounding the 100 best parameter vectors (with the lowest e values) were sampled 9 times as often as points outside. Section 5 shows how convergence was sped up by this simple adaptive search method.

5. RESULTS

The cost function was trained on 80 vines, which could be pruned in a total of 139,258 different ways. Figure 3 shows how $g(\mathbf{p}^*)$ converged over 1 million iterations, or 780 minutes of optimisation. Using adaptive search, the most accurate cost function scored the expert's decisions better than 99.35 % of possible pruning schemes. While this does indicate that the cost function reflects the expert's pruning strategy well, it does not ensure that brute force search will produce good decisions. Brute force search selects the pruning scheme with the lowest cost for each vine, so its success

[1] http://photon.poly.edu/~hbr/boost/combinations.html

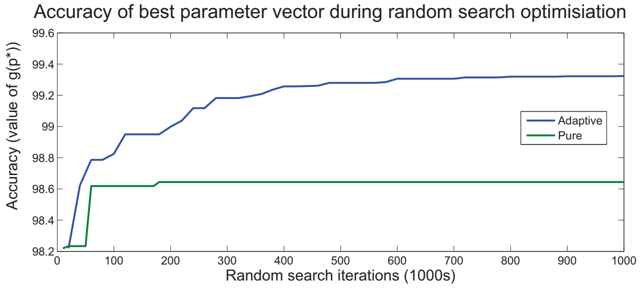

Figure 3: The convergence of pure and adaptive random search optimisation.

depends critically on whether the top 0.65 % are good pruning schemes. In Section 5.1 we determine if this is the case.

In order to ensure the parameters were not creating a cost function that overfit the training data, we measured its accuracy on an validation set of 20 pruned vines. The vines in this set could be pruned in 24,759 ways, 99.59 % of these we correctly scored worse than the expert's decisions by the trained cost function. The cost function therefore generalises very well, as it is least as accurate on the validation set as on the training set.

The brute force search takes on average 0.32 s to execute on a commodity PC, for a vine with 15 candidate canes (the largest vines simulated). This is sufficiently fast for the purposes of our robot, and is small compared to the computation required by the robot's computer vision systems. Although the training process took 13 hours it only has to be performed once, and is therefore not prohibitive.

5.1 Evaluation

With no objective way of determining the quality of pruning schemes, we again made use of Dr Saxton's expertise to evaluate the AI. Because every vineyard manager will have a slightly different opinion of what constitutes good decisions, it is quite reasonable for the same expert to evaluate the system and supply the training data. We specifically wanted to compare the performance of our AI against the status quo in vine pruning: low-skilled, seasonal labour. Our novice pruner was a Lincoln University student who had experience in vine pruning but had not studied it at a tertiary level. He had, in our expert's estimation, above-average vine pruning ability compared to vineyard workers.

Using the simulator, we generated an evaluation set of 100 new vines (not part of the training or validation sets), which were pruned by both the novice and our algorithm. Dr Saxton then evaluated the pruning decisions made on a five-point scale:

1. Major problems with the pruning.
2. Should have been pruned differently, will have a negative effect on the vine.
3. Should have been pruned differently but no harm done.
4. Not perfect but no damage done.
5. As good as possible.

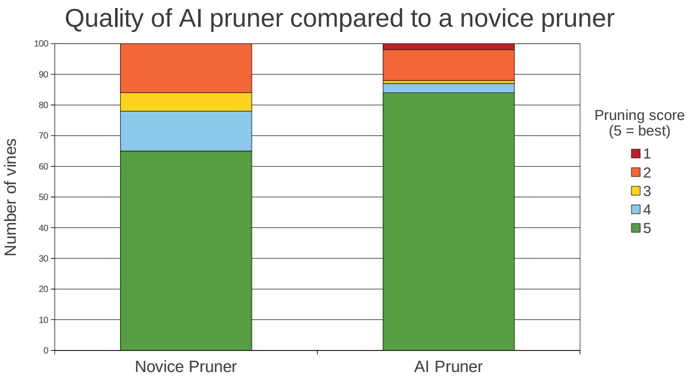

Figure 4: Quality of the decisions made by the AI compared to a novice human pruner.

Table 2: Average evaluation score (out of 5) of human and AI pruners.

Novice	AI	AI (2-cane pruning)
4.27	4.57	4.10

Finally, we tested the cost function for its ability to generalise to pruning strategies it was not trained on. The cost function was trained using examples of 1-cane pruning, so we used our algorithm to prune 20 vines using 2-cane pruning ($n = 1,\ m = 2$). These decision were also evaluated on the 1-5 scale by Dr Saxton.

The results of the evaluation were extremely positive, with Dr Saxton remarking on "how good the routine is", saying she "would let this pruner loose in [her] vineyard". Figure 4 and Table 2 show how our algorithm performed compared to a novice pruner. The pruning decisions made by or algorithm were scored on average 0.3 points better than the novice pruner's decisions.

Comparing pruning decisions head-to-head, our algorithm pruned 30 % of vines better than the novice pruner and 89 % at least as well. Our algorithm also satisfactorily (averaging a score of 4.1) pruned a set of 20 vines using the 2-cane pruning method. This demonstrates the ability of our algorithm to adapt to different pruning strategies. The following section discusses these results, particularly the two vines for which our algorithm failed.

6. DISCUSSION AND FUTURE WORK

Our algorithm performed successfully on the majority of the 100 vines it was evaluated on, but did fail (achieving a score of 1) on two vines. These are shown in Fig. 5 and illustrate limitations of our AI approach. Vine (a) was pruned badly because downward facing spurs were chosen. Although the cost function included a term to penalise spurs at bad angles, this penalty was extremely small. The reflects the fact that the angle of a cane has almost no impact on its suitability as a spur, *unless* that spur is facing down. Because this occurred so rarely in the training set, a suitable penalty was not learned. This illuminates two drawbacks of our approach: decision factors cannot be learned if they do not occur in the training set, and linear penalties do not always model how decisions are made. It should be noted that there is nothing in our approach to stop higher-order

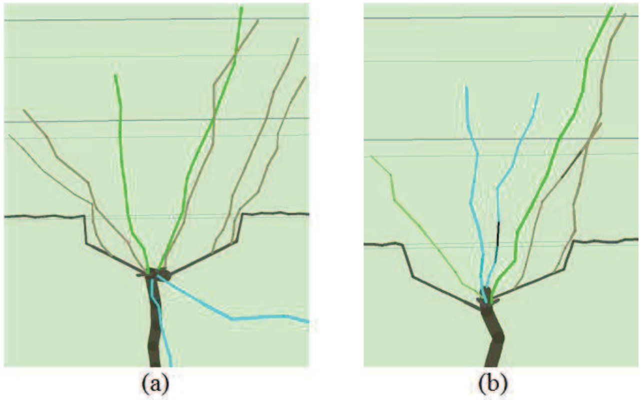

Figure 5: The two worst decisions made by the AI. Canes chosen to spur are marked in blue, canes chosen to tie down are marked in dark green.

features being added to effectively create non-linear penalties, but we decided against this to minimise the size of the feature vectors and therefore the chance of overfitting.

In vine (b) only one cane was tied down, a critical mistake. This occurred because all the canes on the left side are bad choices to tie down, as they are short, diseased, bull canes. The penalty for tying down such canes is less more the penalty for tying down too few canes. Again, the algorithm could not learn a greater missing cordon penalty because there were so few situations in the training set where such poor canes were chosen.

It is clear that machine learning algorithms will struggle when tested in situations they were not trained for, but equally clear that it is infeasible to supply them with every possible relevant training example. This is a fundamental problem in machine learning [16]; the challenge becomes designing algorithms that generalise as well as possible. Many algorithms use regularisation in the learning phase [18], which biases the search towards simpler (and assumed to be more general) models, at the expense of training set accuracy. Taking inspiration from this approach, our algorithm could be made more general by biasing the search towards larger parameters. Because all feature values are designed to increase as a pruning scheme gets worse, it is generally true that larger parameters better match how experts prune vines. Indeed, both failures of the AI could have been addressed by better parameterisation.

7. CONCLUSION

This paper has presented a detailed account of the successful development of an expert system for pruning vines. We have demonstrated that computers are able to make good vine pruning decisions based on a model of vines generated by a computer vision system. Therefore, the task of vine pruning can be automated. We have also shown that automating the task could improve vine quality, as our system outperforms an above-average human pruner. It pruned 84 % of vines in an evaluation set perfectly, and 89 % at least as well as the human pruner. Our approach is now ready to be tested on data extracted from real vines.

8. REFERENCES

[1] K. P. Bennett and E. Parrado-HernÃąndez. The interplay of optimization and machine learning research. *The Journal of Machine Learning Research*, 7:1265–1281, 2006.

[2] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, Oct. 2001.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Chapman & Hall, New York, NY, 1984.

[4] S. H. Brooks. A discussion of random methods for seeking maxima. *Operations Research*, pages 244–251, 1958.

[5] R. Caruana, N. Karampatziakis, and A. Yessenalina. An empirical evaluation of supervised learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 96–103, New York, NY, USA, 2008. ACM.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[7] F. Glover and R. Marti. Tabu search. *Metaheuristic Procedures for Training Neutral Networks*, pages 53–69, 2006.

[8] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[9] F. Hayes-Roth, D. A. Waterman, and D. B. Lenat. *Building expert systems.* Addison-Wesley Pub. Co., Reading, Mass., 1983.

[10] J. H. Holland. *Adaptation in natural and artificial systems.* Number 53. University of Michigan press, 1975.

[11] R. Horst and P. M. Pardalos. *Handbook of global optimization.* Kluwer Academic Publishers, Dordrecht; Boston, 1995.

[12] P. Jackson. *Introduction to Expert Systems.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1990.

[13] M. Kassler. Agricultural automation in the new millennium. *Computers and electronics in agriculture*, 30(1-3):237–240, 2001.

[14] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948, 1995.

[15] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[16] T. Mitchell. *Machine Learning (Mcgraw-Hill International Edit).* McGraw-Hill Education (ISE Editions), 1st edition, Oct. 1997.

[17] G. N. R. Prasad and A. V. Babu. A study on various expert systems in agriculture. *Extension of Technologies: From Labs to Farms*, page 297, 2008.

[18] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*, volume 1st. Prentice Hall, 1995.

[19] F. J. Solis. Minimization by random search techniques. *Mathematics of operations research*, pages 19–30, 1981.

[20] J. Zurada. *Introduction to artificial neural systems.* West Publishing Co., St. Paul, MN, USA, 1992.