Introduction

It is our belief that researchers and practitioners acquire, through experience and word-of-mouth, techniques and heuristics that help them successfully apply neural networks to difficult real world problems. Often these "tricks" are theoretically well motivated. Sometimes they are the result of trial and error. However, their most common link is that they are usually hidden in people's heads or in the back pages of space-constrained conference papers. As a result newcomers to the field waste much time wondering why their networks train so slowly and perform so poorly.

This book is an outgrowth of a 1996 NIPS workshop called *Tricks of the Trade* whose goal was to begin the process of gathering and documenting these tricks. The interest that the workshop generated motivated us to expand our collection and compile it into this book. Although we have no doubt that there are many tricks we have missed, we hope that what we have included will prove to be useful, particularly to those who are relatively new to the field. Each chapter contains one or more tricks presented by a given author (or authors). We have attempted to group related chapters into sections, though we recognize that the different sections are far from disjoint. Some of the chapters (e.g., 1, 13, 17) contain entire systems of tricks that are far more general than the category they have been placed in.

Before each section we provide the reader with a summary of the tricks contained within, to serve as a quick overview and reference. However, we do not recommend applying tricks before having read the accompanying chapter. Each trick may only work in a particular context that is not fully explained in the summary. This is particularly true for the chapters that present systems where combinations of tricks must be applied together for them to be effective.

Below we give a rough roadmap of the contents of the individual chapters.

Speeding Learning

The book opens with a chapter based on Leon Bottou and Yann LeCun's popular workshop on efficient backpropagation where they present a system of tricks for speeding the minimization process. Included are tricks that are very simple to implement as well as more complex ones, e.g., based on second-order methods. Though many of the readers may recognize some of these tricks, we believe that this chapter provides both a thorough explanation of their theoretical basis and an understanding of the subtle interactions among them.

This chapter provides an ideal introduction for the reader. It starts by discussing fundamental tricks addressing input representation, initialization, target values, choice of learning rates, choice of the nonlinearity, and so on. Subsequently, the authors introduce in great detail tricks for estimation and approximation of the Hessian in neural networks. This provides the basis for a discussion of second-order algorithms, fast training methods like the stochastic Levenberg-Marquardt algorithm, and tricks for learning rate adaptation.

Regularization Techniques to Improve Generalization

Fast minimization is important but only if we can also insure good generalization. We therefore next include a collection of chapters containing a range of approaches for improving generalization. As one might expect, there are no tricks that work well in all situations. However, many examples and discussions are included to help the reader to decide which will work best for their own problem.

Chapter 2 addresses what is one of the most commonly used techniques: early stopping. Here Lutz Prechelt discusses the pitfalls of this seemingly simple technique. He quantifies the tradeoff between generalization and training time for various stopping criteria, which leads to a trick for picking an appropriate criterion.

Using a weight decay penalty term in the cost function is another common method for improving generalization. The difficulty, however, is in finding a good estimate of the weight decay parameter. In Chapter 3, Thorsteinn Rögnvaldsson presents a fast technique for finding a good estimate, surprisingly, by using information measured at the early stopping point. Experimental evidence for its usefulness is given in several applications.

Tony Plate in Chapter 4 treats the penalty terms along the lines of MacKay, i.e. as hyperparameters to be found through iterative search. He presents and compares tricks for making the hyperparameter search in classification networks work in practice by speeding it up and simplifying it. Key to his success is a control of the frequency of the hyperparameter updates and a better strategy in cases where the Hessian becomes out-of-bounds.

In Chapter 5, Jan Larsen et al. present a trick for adapting regularization parameters by using simple gradient descent (with respect to the regularization parameters) on the validation error. The trick is tested on both classification and regression problems.

Averaging over multiple predictors is a well known method for improving generalization. Two questions that arise are how many predictors are "enough" and how the number of predictors affects the stopping criteria for early stopping. In the final chapter of this section, David Horn et al. present solutions to these questions by providing a method for estimating the error of an infinite number of predictors. They then demonstrate this trick for a prediction task.

Improving Network Models and Algorithmic Tricks

In this section we examine tricks that help improve the network model. Even though standard multilayer perceptrons (MLPs) are, in theory, universal approximators, other architectures may provide a more natural fit to a problem. A better fit means that training is faster and that there is a greater likelihood of finding a good and stable solution. For example, radial basis functions (RBFs) are preferred for problems that exhibit local features in a finite region. Of course, which architecture to choose is not always obvious.

In Chapter 7, Gary Flake presents a trick that gives MLPs the power of both an MLP and an RBF so that one does not need to choose between them. This trick is simply to add extra inputs whose values are the square of the regular inputs. Both a theoretical and intuitive explanation are presented along with a number of simulation examples.

Rich Caruana in Chapter 8 shows that performance can be improved on a main task by adding extra outputs to a network that predict related tasks. This technique, known as multi-task learning (MTL), trains these extra outputs in parallel with the main task. This chapter presents multiple examples of what one might use as these extra outputs as well as techniques for implementing MTL effectively. Empirical examples include mortality rankings for pneumonia and road-following in a network learning to steer a vehicle.

Patrick van der Smagt and Gerd Hirzinger consider in Chapter 9 the ill-conditioning of the Hessian in neural network training and propose using what they call a linearly augmented feed-forward network, employing input/output short-cut connections that share the input/hidden weights. This gives rise to better conditioning of the learning problem and, thus, to faster learning, as shown in a simulation example with data from a robot arm.

In Chapter 10, Nicol Schraudolph takes the idea of scaling and centering the inputs even further than Chapter 1 by proposing to center all factors in the neural network gradient: inputs, activities, error signals and hidden unit slopes. He gives experimental evidence for the usefulness of this trick.

In Chapter 11, Tony Plate's short note reports a numerical trick for computing derivatives more accurately with only a small memory overhead.

Representation and Incorporating Prior Knowledge in Neural Network Training

Previous chapters (e.g., Chapter 1) present very general tricks for transforming inputs to improve learning: prior knowledge of the problem is not taken into account explicitly (of course regularization, as discussed in Chapters 2–5, implicitly assumes a prior but on the weight distribution). For complex, difficult problems, however, it is not enough to take a black box approach, no matter how good that black box might be. This section examines how prior knowledge about a problem can be used to greatly improve learning. The questions asked include how to best represent the data, how to make use of this representation for training, and how to take advantage of the invariances that are present. Such issues are key for proper neural network training. They are also at the heart of the tricks pointed out by Patrice Simard et al. in the first chapter of this section. Here, the authors present a particularly interesting perspective on how to incorporate prior knowledge into data. They also give the first review of the tangent distance classification method and related techniques evolving from it such as tangent prop. These methods are applied to the difficult task of optical character recognition (OCR).

In Chapter 13, Larry Yaeger et al. give an overview of the tricks and techniques for on-line handwritten character recognition that were eventually used in Apple Computer's Newton MessagePad ®and eMate®. Anyone who has used these systems knows that their handwriting recognition capability works exceedingly well. Although many of the issues that are discussed in this chapter overlap with those in OCR, including representation and prior knowledge, the solutions are complementary. This chapter also gives a very nice overview of what design choices proved to be efficient as well as how different tricks such as choice of learning rate, over-representation of more difficult patterns, negative training, error emphasis, and so on work together.

Whether it be handwritten character recognition, speech recognition or medical applications, a particularly difficult problem encountered is the unbalanced class prior probabilities that occur, for example, when certain writing styles and subphoneme classes are uncommon or certain illnesses occur less frequently. Chapter 13 briefly discusses this problem in the context of handwriting recognition and presents a heuristic which controls the frequency with which samples are picked for training.

In Chapter 14, Steve Lawrence et al. discuss the issue of unbalanced class prior probabilities in greater depth. They present and compare several different heuristics (prior scaling, probabilistic sampling, post scaling and class membership equalization) one of which is similar to that in Chapter 13. They demonstrate their tricks by solving an ECG classification problem and provide some theoretical explanations.

Many training techniques work well for nets of small to moderate size. However, when problems consist of thousands of classes and millions of examples, not uncommon in applications such as speech recognition, many of these techniques break down. This chapter by Jürgen Fritsch and Michael Finke is devoted to the issue of large scale classification problems and representation design in general. Here the problem of unbalanced class prior probabilities is also tackled.

Although Fritsch and Finke specifically exemplify their design approach for the problem of building a large vocabulary speech recognizer, it becomes clear that these techniques are also applicable to the general construction of an appropriate hierarchical decision tree. A particularly interesting result in this paper is that the structural design for incorporating prior knowledge about speech done by a human speech expert was outperformed by their machine learning technique using an agglomerative clustering algorithm for choosing the structure of the decision tree.

Tricks for Time Series

We close the book with two papers on the subject of time series and economic forecasting. In the first of these chapters, John Moody presents an excellent survey of both the challenges of macroeconomic forecasting and a number of neural network solutions. The survey is followed by a more detailed description of smoothing regularizers, model selection methods (e.g., AIC, effective number of parameters, nonlinear cross-validation), and input selection via sensitivity-based input pruning. Model interpretation and visualization are also discussed.

In the final chapter, Ralph Neuneier and Hans Georg Zimmermann present an impressive integrated system for neural network training of time series and economic forecasting. Every aspect of the system is discussed including input preprocessing, cost functions, handling of outliers, architecture, regularization techniques, and solutions for dealing with the problem of bottom-heavy networks, i.e., the input dimension is large while the output dimension is very small. There is also a thought-provoking discussion of the Observer-Observer dilemma: we want both to create a model based on observed data and, at the same time, to use this model to judge the correctness of new incoming data. Even those people not interested specifically in economic forecasting are encouraged to read this very useful example of how to incorporate prior (system) knowledge into training.

Final Remark

As a final remark, we note that some of the views taken in the chapters contradict each other, e.g., some authors favor one regularization method over another, while other authors make exactly the opposite statement. On the one hand, one can explain these discrepancies by stating that the field is still very active and therefore opposing viewpoints will inevitably exist until more is understood. On the other hand, it may be that both (contradicting) views are correct but on different data sets and in different applications, e.g., an approach that considers noisy time-series needs algorithms with a completely different robustness than in, say, an OCR setting. In this sense, the present book mirrors an active field and a variety of applications with its diversity of views.

August 1998 Jenny & Klaus

Acknowledgments

We would like to thank all authors for their collaboration. Special thanks to Steven Lemm for considerable help with the typesetting. K.-R.M. acknowledges partial financial support from DFG (grant JA 379/51 and JA 379/7) and EU ESPRIT (grant 25387-STORM).