Establishing Safety Criteria for Artificial Neural Networks

Zeshan Kurd, Tim Kelly

Department of Computer Science
University of York, York, YO10 5DD, UK.
{zeshan.kurd, tim.kelly}@cs.york.ac.uk

Abstract. Artificial neural networks are employed in many areas of industry such as medicine and defence. There are many techniques that aim to improve the performance of neural networks for safety-critical systems. However, there is a complete absence of analytical certification methods for neural network paradigms. Consequently, their role in safety-critical applications, if any, is typically restricted to advisory systems. It is therefore desirable to enable neural networks for highly-dependable roles. This paper defines the safety criteria which if enforced, would contribute to justifying the safety of neural networks. The criteria are a set of safety requirements for the behaviour of neural networks. The paper also highlights the challenge of maintaining performance in terms of adaptability and generalisation whilst providing acceptable safety arguments.

1 Introduction

Typical uses of ANNs (Artificial Neural Networks) in safety-critical systems include areas such as medical systems, industrial process & control and defence. An extensive review of many areas of ANN use in safety-related applications has been provided by a recent U.K. HSE (Health & Safety Executive) report [1]. However, the roles for these systems have been restricted to advisory, since no convincing safety arguments has yet been produced. They are typical examples of achieving improved performance without providing sufficient safety assurance.

There are many techniques for traditional multi-layered perceptrons [2, 3] which aim to improve generalisation performance. However, these performance-related techniques do not provide acceptable forms of safety arguments. Moreover, any proposed ANN model that attempts to satisfy safety arguments must carefully ensure that measures incorporated do not diminish the advantages of the ANN.

Many of the existing approaches which claim to contribute to safety-critical applications focus more upon improving generalisation performance and not generating suitable safety arguments. One particular type of ANN model employs diversity [2] which is an ensemble of ANNs where each is devised by different methods to encapsulate as much of the target function as possible. Results demonstrate improvement in generalisation performance over some test set but lack in the ability to analyse and determine the function performed by each member of the ensemble.

Other techniques such as the validation of ANNs and error bars [3] may help deal with uncertainty in network outputs. However, the ANN is still viewed as a black-box and lacks satisfactory analytical processes to determine the overall behaviour. Finally, many development lifecycles [4] for ANNs lack provision for analytical processes such as hazard analysis and focuses more upon black-box approaches. However, work performed on fuzzy-ANNs [5] attempts to approach development based upon refining partial specifications.

2 The Problem

The criticality of safety-critical software can be defined as directly or indirectly contributing to the occurrence of a hazardous system state [6]. A hazard is a situation, which is potentially dangerous to man, society or the environment [7]. When developing safety-critical software, there are a set of requirements which must be enforced and are formally defined in many safety standards [8]. One of the main tools for determining requirements is the use of a safety case to encapsulate all safety arguments for the software. A safety case is defined in Defence Standard 00-55 [8] as:

> *"The software safety case shall present a well-organised and reasoned justification based on objective evidence, that the software does or will satisfy the safety aspects of the Statement of Technical Requirements and the Software Requirements specification."*

Some of the main components in formulating safety requirements are hazard analysis and mitigation. Function Failure Analysis (FFA) [9] is a predictive technique to distinguish and refine safety requirements. It revolves around system functions and deals with identifying failures to provide a function, incorrect function outputs, analysing effects of failures and determining actions to improve design. Another technique is Software Hazard Operability Study (SHAZOP) [10] which uses 'guide words' for qualitative analysis to identify hazards not previously considered. It attempts to analyse all variations of the system based on these guide words and can uncover causes, consequences, indications and recommendations for particular identified hazards.

Arguing the satisfaction of the safety requirements is divided into analytical arguments and arguments generated from testing. This is defined in a widely accepted Defence Standard 00-55 [8]. Arguments from testing (such as white-box or black-box) can be generated more easily than analytical arguments. Unlike neural networks, the behaviour of the software is fully described and available for analysis. However, generating analytical arguments is more problematic. Some of these problems are mainly associated with providing sufficient evidence that all hazards have been identified and mitigated.

There are several safety processes performed during the life-cycle of safety-critical systems [11]. Preliminary Hazard Identification (PHI) is the first step in the lifecycle, it forms the backbone of all following processes. It aims to identify, manage and control all potential hazards in the proposed system. Risk Analysis and Functional Hazard Analysis (FHA) analyses the severity and probability of potential accidents for each

identified hazard. Preliminary System Safety Assessment (PSSA) [9] ensures that the proposed design will refine and adhere to safety requirements and help guide the design process. System Safety Analysis (SSA) demonstrates through evidence that safety requirements have been met. It uses inductive and deductive techniques to examine the completed design and implementation. Finally, the Safety Case [11] generated throughout development delivers a comprehensible and defensible argument that the system is acceptably safe to use in a given context.

Any potential safety case must overcome problems associated with typical neural networks. Some typical problems may be concerned with ANN structure and topology. These are factors that may influence the generalisation performance of the ANN. Another problem lies in determining the training and test set where they must represent the desired function using a limited number of samples. Dealing with noise during training is also problematic to ensure that the network does not deviate from the required target function. Other issues related to the learning or training process may involve forgetting of data particularly when using the back-propagation algorithm. This could lead to poor generalisation and long training times. Furthermore, problems during training may result in learning settling in local minima instead of global and deciding upon appropriate stopping points for the training process. This could be aided by cross-validation [12] but relies heavily on test sets.

One of the key problems is the inability to analyse the network such that a white-box view of the behaviour can be presented. This contributes to the need for using test sets to determine generalisation performance as an overall error and the lack of explanation mechanisms for network outputs. The inability to analyse also makes it difficult to identify and control potential hazards in the system and provide assurance that some set of requirements are met.

3 Safety Criteria for Neural Networks

We have attempted to establish safety criteria for ANNs (Artificial Neural Networks) that defines minimum behavioural properties which must be enforced for safety-critical contexts. By defining requirements from a high-level perspective, the criteria are intended to apply to most types of neural networks. Figure 1.0 illustrates the safety criteria in the form of Goal Structuring Notation (GSN) [11] which is commonly used for composing safety case patterns.

Each symbol in figure 1.0 has a distinct meaning and is a subset of the notation used in GSN. The boxes illustrate goals or sub-goals which need to be fulfilled. Rounded boxes denote the context in which the corresponding goal is stated. The rhomboid represents strategies to achieve goals. Diamonds underneath goals symbolise that the goal requires further development leading to supporting arguments and evidence.

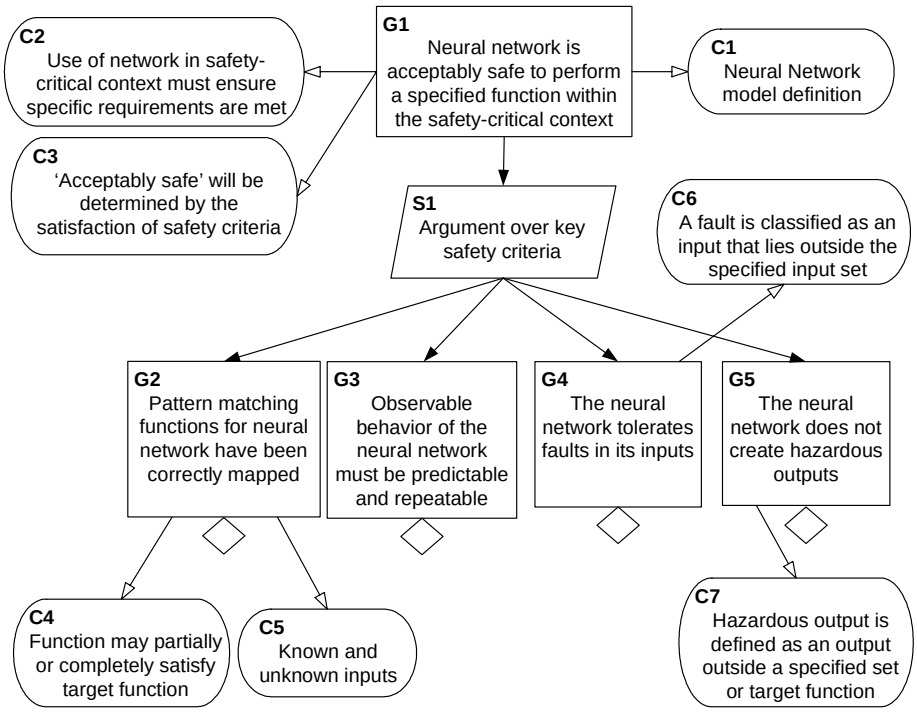

Fig. 1. Preliminary Safety Criteria for Artificial Neural Networks

The upper part of the safety criteria consists of a top-goal, strategy and other contextual information. The goal G1 if achieved, allows the neural network to be used in safety-critical applications. C1 requires that a specific ANN model is defined such as multi-layered perceptron or other models. C2 intends for the ANN to be used when conventional software or other systems cannot provide the desired advantages. C3 attempts to highlight that 'acceptably safe' is related to product and process based arguments and will rely heavily on sub-goals. The strategy S1 will attempt to generate safety arguments from the sub-goals (which form the criteria) to fulfil G1. The goals G2 to G5 presents the following criteria:

Criterion G2 ensures that the function performed by the network represents the target or desired function. The function represented by the network may be considered as input-output mappings and the term 'correct' refers to the target function. As expressed in C4, the network may also represent a subset of the target function. This is a more realistic condition, if all hazards can be identified and mitigated for the subset. This may also avoid concerns for attempting to solve a problem where analysis may not be able to determine whether totality of the target function is achieved. Previous work on dealing with and refining partial specification for neural networks [5] may apply. However, additional methods to analyse specifications in terms of performance and safety (existence of potential hazards) may be necessary.

Forms of sub-goals or strategies for arguing G2 may involve using analytical methods such as decompositional approaches [13]. This attempts to extract behaviour by

analysing the intrinsic structure of the ANN such as each neuron or weight. This will help analyse the function performed by the ANN and help present a white-box view of the network. Techniques to achieve this may involve determining suitable ANN architectures whereby a meaningful mapping exists from each network parameter to some functional description. On the other hand, pedagogical approaches involve determining the function by analysing outputs for input patterns such as sensitivity analysis [14]. This methodology however, maintains a black-box perspective and will not be enough to provide satisfactory arguments for G2. Overall, approaches must attempt to overcome problems associated with the ability of the ANN to explain outputs and generalisation behaviour.

Criterion G3 will provide assurance that safety is maintained during ANN learning. The 'observable behaviour' of the neuron means the input and output mappings that take place regardless of the weights stored on each connection. The ANN must be predictable given examples learnt during training. 'Repeatable' ensures that any previous valid mapping or output does not become flawed during learning. The forgetting of previously learnt samples must not occur given adaptation to a changing environment. Possible forms of arguments may be associated with functional properties. This may involve providing assurance that learning maintains safety by abiding to some set of behavioural constraints identified through processes such as hazard analysis.

Criterion G4 ensures the ANN is robust and safe under all input conditions. An assumption is made that the ANN might be exposed to training samples that do not represent the desired function. Achievement of this goal is optional as it could be considered specific to the application context. The network must either detect these inputs and suppress them, or ensure a safe state with respect to the output and weight configuration. Other possible forms of argument and solutions may involve 'gating networks' [15] which receives data within a specific validation area of the input space.

Criterion G5 is based upon arguments similar to G2. However, this goal focuses solely on the network output. This will result in a robust ANN, which through training and utilisation, ensures that the output is not hazardous regardless of the integrity of the input. For example, output monitors or bounds might be used as possible solutions. Other possible forms of arguments may include derivatives of 'guarded networks' [15] that receives all data but is monitored to ensure that behaviour does not deviate too much from expectations.

The above criteria focuses on product-based arguments in contrast to process-based arguments commonly used for conventional software [16]. This follows an approach which attempts to provide arguments based upon the product rather than a set of processes that have been routinely carried out [16]. However, process and product-based arguments may be closely coupled for justifying the safety ANNs. For example, methods such as developing ANNs using formal languages [17] might reduce faults incorporated during implementation. However the role of analytical tools is highly important and will involve hazard analysis for identifying potential hazards. Some factors which prevent this type of analysis can be demonstrated in typical monolithic neural networks. These scatter their functional behaviour around its weights in incomprehensible fashion resulting in black-box views and pedagogical approaches. Refining various properties of the network such as structural and topological factors may help overcome this problem such as modular ANNs [18]. It may help improve the relation-

ship between intrinsic ANN parameters and meaningful descriptions of functional behaviour. This can enable analysis in terms of domain-specific base functions associated with the application domain [19]. Other safety processes often used for determining existence of potential hazards may require adaptation for compatibility with neural network paradigms.

4 Performance vs. Safety Trade-off

One of the main challenges when attempting to satisfy the criteria is constructing suitable ANN models. The aim is to maintain feasibility by providing acceptable performance (such as generalisation) whilst generating acceptable safety assurance. Certain attributes of the model may need to be limited or constrained so that particular arguments are possible. However, this could lead to over-constraining the performance of ANNs. Typical examples may include permissible learning during development but not whilst deployed. This may provide safety arguments about the behaviour of the network without invalidating them with future changes to implementation. The aim is to generate acceptable safety arguments providing comparable assurance to that achieved with conventional software.

5 Conclusion

To justify the use of ANNs within safety critical applications will require the development of a safety case. For high-criticality applications, this safety case will require arguments of correct behaviour based both upon analysis and test. Previous approaches to justifying ANNs have focussed predominantly on (test-based) arguments of high performance.

In this paper we have presented the key criteria for establishing an acceptable safety case for ANNs. Detailed analysis of potential supporting arguments is beyond the scope of this paper. However, we have discussed how it will be necessary to constrain learning and other factors, in order to provide analytical arguments and evidence. The criteria also highlight the need for adapting current safety processes (hazard analysis) and devising suitable ANN models that can accommodate them. The criteria can be considered as a benchmark for assessing the safety of artificial neural networks for highly-dependable roles.

References

1. Lisboa, P., Industrial use of safety-related artificial neural networks. Health & Safety Executive 327, (2001).
2. Sharkey, A.J.C. and N.E. Sharkey, Combining Diverse Neural Nets, in Computer Science. (1997), University of Sheffield: Sheffield, UK.

3. Nabney, I., et al., Practical Assessment of Neural Network Applications. (2000), Aston University & Lloyd's Register: UK.
4. Rodvold, D.M. A Software Development Process Model for Artificial Neural Networks in Critical Applications. in Proceedings of the 1999 International Conference on Neural Networks (IJCNN'99). (1999). Washington D.C.
5. Wen, W., J. Callahan, and M. Napolitano, Towards Developing Verifiable Neural Network Controller, in Department of Aerospace Engineering, NASA/WVU Software Research Laboratory. (1996), West Virginia University: Morgantown, WV.
6. Leveson, N., Safeware: system safety and computers. (1995): Addison-Wesley.
7. Villemeur, A., Reliability, Availability, Maintainability and Safety Assessment. Vol. 1. (1992): John Wiley & Sons.
8. MoD, Defence Standard 00-55: Requirements for Safety Related Software in Defence Equipment. (1996), UK Ministry of Defence.
9. SAE, ARP 4761 - Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment. (1996), The Society for Automotive Engineers,.
10. MoD, Interim Defence Standard 00-58 Issue 1: HAZOP Studies on Systems Containing Programmable Electronics. (1996), UK Ministry of Defence.
11. Kelly, T.P., Arguing Safety – A Systematic Approach to Managing Safety Cases, in Department of Computer Science. (1998), University of York: York, UK.
12. Kearns, M., A Bound on the Error of Cross Validation Using the Approximation and Estimation Rates, with Consequences for the Training-Test Split, AT&T Bell Laboratories.
13. Andrews, R., J. Diederich, and A. Tickle, A survey and critique of techniques for extracting rules from trained artificial neural networks. (1995), Neurocomputing Research Centre, Queensland University of Technology.
14. Kilimasaukas, C.C., Neural nets tell why. Dr Dobbs's, (April 1991): p. 16-24.
15. Venema, R.S., Aspects of an Integrated Neural Prediction System. (1999), Rijksuniversiteit, Groningen: Groningen, Netherlands.
16. Weaver, R.A., J.A. McDermid, and T.P. Kelly. Software Safety Arguments: Towards a Systematic Categorisation of Evidence. in International System Safety Conference. (2002). Denver, CO.
17. Dorffner, G., H. Wiklicky, and E. Prem, Formal neural network specification and its implications on standardisation. Computer Standards and Interfaces, (1994). **16**(205-219).
18. Osherson, D.N., S. Weinstein, and M. Stoli, Modular Learning. Computational Neuroscience, Cambridge - MA, (1990): p. 369-377.
19. Zwaag, B.J. and K. Slump. Process Identification Through Modular Neural Networks and Rule Extraction. in 5th International FLINS Conference. (2002). Gent, Belgium: World Scientific.