Roundtable: Machine Learning for Embedded Systems: Hype or Lasting Impact?
A panel at ESWEEK 2017

X. Sharon Hu, University of Notre Dame
Rolf Ernst, TU Braunschweig
Petru Eles, Linköping University
Gernot Heiser, UNSW Sydney and Data61 CSIRO
Kurt Keutzer, UC Berkeley
Daehyun Kim, Samsung
Tetsuya Tohdo, DENSO

A previous IEEE Design&Test "Roundtable" already discussed the aspect of machine learning (ML) test and verification, but the impact of ML is wider, including hardware, software, and communication architectures and design, as well as behavioral guarantees, just to name a few important fields. ML has also started to develop a strong impact on important embedded systems design and applications. While the initial success raises high expectations for the reinvention of engineering, a discussion is overdue on where this development will eventually lead us in research and engineering.

A highly attended plenary panel at Embedded Systems Week (ESWEEK) 2017 in Seoul, South Korea, with the provocative title "Machine Learning for Embedded Systems: Hype or Lasting Impact?" spurred a lively and controversial discussion that is continued in this roundtable. It is moderated by the panel organizers and moderators X. Sharon Hu, University of Notre Dame, and Rolf Ernst, Technische Universität Brunswick. Panelists include Petru Eles, Linköping University; Gernot Heiser, UNSW Sydney; Kurt Keutzer, University of California at Berkeley; Daehyun Kim, Samsung; and Tetsuya Tohdo, DENSO CORP.

Sharon Hu: Thank you for joining the roundtable. Let me start with an introductory statement. ML for embedded systems has a strong impact on embedded systems applications and architectures. This refers particularly—but not exclusively—to deep learning. ML comes with high requirements in terms of performance, power consumption, and timing, which conflict with limited resources of embedded systems. Verification and guaranteed behavior remain challenges. Therefore, there must be good reasons to extend its use beyond the current applications.

I would like to start the discussion with a general question. Do you agree or do you see ways to control or to adapt ML to reach predictable and guaranteed behavior? Should we even give up on predictability and guarantees to achieve higher typical performance?

Gernot Heiser: I think the question, as posed, misses the point somewhat. An aircraft pilot doesn't show "predictable and guaranteed behavior," yet planes operate extremely reliably.Part of that is redundancy, part is that decisions aren't black and white,and there is a feedback loop that corrects for temporarily incorrect or imprecise behavior.The same is achievable with ML. In terms of RT behavior, what is required is that the time to execute one iteration of the control loop is bounded. In ML, that is essentially one matrix multiply, which is straightforward to bound.

Tetsuya Tohdo: I would propose two other approaches instead of pursuing "guarantee" on ML for practical reasons. One is combining other complementary technologies with ML to achieve required characteristics, for example, introducing an independent safety mechanism to mitigate the unsafe behavior of ML featured functionalities. The other approach is evaluating the performance of ML features by testing, depending on their use cases in order to get reasonably acceptable confidence, like an evaluation of pedestrian recognition based on ML. I do not deny the studies that pursue metrics able to guarantee ML performances because they will also contribute to the other approaches I am proposing here.

Petru Eles: I think that the question implies a scenario in which there is a safety critical function implemented with a piece of ML software in its middle. In this context, it is reasonable to argue that proper guarantees and a certain degree of predictability is needed and, for certain applications, even requested by certification bodies. Not only academia has argued that formal verification or testing with quantifiable coverage is needed in such a context. If we imagine an image processing ML software in a driverless car, it is reasonable to expect such a degree of predictability in order not to miss a pedestrian.

Gernot's argument is interesting here. But it, in some way, takes out the ML component from the safety-critical loop by assuming a lower level of the safety system. I'm not sure how this would work with our pedestrian detection. If the image processing is in doubt the emergency control, e.g., breaks the car? But we still have to make sure that this"level of doubt" is triggered properly (and we also want to avoid too many false positives and break all the time).And,then,let's not forget.Our drivers and pilots are strictly "certified" when they get their license. Their level of training is checked and quantified over a carefully designed exam. There is also a health check and some implicit assumption regarding their ethical and social maturity (like minimal age limit). When software is put in place,certification is exactly this licensing.

Kurt Keutzer: When we are dealing with a real-world problem, we apply formal methods by first mathematically modeling the problem domain and then we use formal methods to reason about that object. The presumption is that if our modeling is accurate enough, our results will be applicable to the original problem. An integrated circuit, for example, is a very complicated physical object; however, through a series of simplifications, we are able to model a combinational circuit as a Boolean function. We can then reason about them. For example, we can prove, or at least verify after the fact, that a series of logic optimizations produces a function that is equivalent to the original. Our modeling methods are well tested in practice and we can have confidence that logic optimization has not altered the functionality of the original circuit.

In the areas that I'm researching, like computer vision for embedded systems, and particularly perceptual systems for autonomous vehicles, the problem is not with the Deep Learning methods that we apply. The problem is with our inability to formally model the problem domain. For example, we don't have a formalism in which we can "prove" that a camera image of a stop sign is actually a stop sign or an image of a pedestrian is a pedestrian. In computer vision research, the method by which we assess our object detection techniques is by comparing the results to human beings performing the same task.

This doesn't mean that formal methods are useless. We are using formal methods such as automated counterexample generation to help us to improve the accuracy of computer vision systems. This has a positive practical impact.

Rolf Ernst: If achieving "perfect" predictability/guarantee should not be the goal for ML when applied in safety-critical systems, what could be acceptable metrics for measurable predictability/guarantee? Will such metrics be significantly influenced by ML implementation options (such as word length reduction, etc.)?

Eles: This is the million dollar question! Considering that the neural network (NN) is an uninterpretable black box, both formal verification and structural testing with the traditional coverage metrics is difficult to imagine. Therefore, to my knowledge, extensive functional testing is the practice, with “metrics” like driving a certain (very large) amount of miles (physically or simulated) and running through a sequence of collected scenarios. These are, of course, not the metrics that really provide a quantification of predictability and trust. When we, traditionally, talk about coverage in functional testing, we mean which of the features of the application have been covered by the test, assuming a set of typical scenarios. But in the context of an application like autonomous driving, the problem is with the scenarios. How do we know that we have covered the corner-case scenarios and how can we get a quantification of the extent to which we did it? I think that this is an extremely exciting research area and the answers to these questions are still to come.

Daehyun Kim: Important and difficult question. I guess a kind of probability can be given as a metric. For example, we can say that a vision algorithm can detect a person with 95% accuracy. However, even giving such probability is not possible in an absolute sense. If it can be done, it is actually the same as “perfect” guarantee. Therefore, in my opinion, a practical metric would be like “for a given test set, an ML algorithm showed x% accuracy,” which is, for example, what the ImageNet challenge does. I think the most important thing is that people understand the imperfectness of ML. Then, the question is whether people will accept such a metric and use ML in safety-critical systems. I think we will. We have always done it in our history. Of cause, we need to put efforts to make our test cases better and better to increase their coverage.

Heiser: In line with my answer to the first question, what is needed is a bound on the execution time of one control-loop iteration, plus presumably some convergence criterion. In general, I think it is unrealistic (for now anyway) to expect more precision/predictability than from a human operator, except that ML will eliminate human factors like fatigue, distraction, etc. And, unlike a human operator, we can demand auditability of the decision-making process, and a degree of reproducibility. If we achieve all these (and solve the “small” problem of cybersecurity), then we’ve already made a huge improvement in safety over human operators. And I don’t think we’re anywhere near defining a meaningful metric of the safety of software systems (as we don’t have a metric of security). Classical safety arguments are based on assumptions on the randomness and independence of hardware failures. The software is neither random nor are its failure mode independent. Trying to come up with a measure of software safety is likely to be delusional.

Keutzer: The automotive systems that we have today consist of ad hoc tested control components with an unpredictable human-in-the loop. I don’t think it will be hard to improve over that. We’re working with Dreossi and Seshia on a variety of approaches for verifying cyber-physical systems such as autonomous vehicles. For something like an Automatic Emergency Braking System that automatically brakes to avoid obstacles, one approach is to model the system in a cyber-physical system analyzer, which then identifies regions of interest with regard to the ML/deep learning object detector. This in turn can then be used to generate test cases for the object detector in the region of interest. Finally, in simple cases, we can establish a kind of coverage metric over the test cases. On the one hand, this only increases our confidence in the system; there are no guarantees. On the other hand, this is still much more formal and disciplined than the currently deployed state-of-the-art.

Tohdo: In the automotive domain, ISO26262 requires hierarchical decomposition in system and software design in order to achieve safety. This means that end-to-end learning approaches are hard to apply, as Salay et al. [1] have discussed, and also means that the evaluation of ML components is much important in practice to achieve sufficient confidence in accordance with the requirements allocated on them. Measuring sufficiency is crucial for the data set used for this evaluation as well as the training data set. Metrics for this purpose would depend on the requirements and domain models behind, for example, ranges of physical parameters of the object identification application, which can contribute to increasing robustness of the evaluation.

Hu: How will ML functions change the design process including system synthesis, test, and verification? Is online learning a realistic option in a manageable design process and systems operation?

Kim: I do not have expertise, so provide my non-expert opinion. In ML, data sets (training set and test set) are actually as important as algorithms. So, the design process should include how to acquire and validate the data sets. I would like to argue that managing data sets are the most important process in the ML system design, which is a totally new addition to the conventional design process. Online learning will provide opportunities to keep acquiring data sets even after the products are shipped. However, due to the uncertainty of ML, it is not guaranteed that the system will keep improving with more data sets. So, the application of online learning will be limited as an incremental learning to change the system behavior gradually.

Tohdo: ML technologies allow using training data sets instead of the specification and this fact would pose a huge impact on the development process. Considering a case to develop new functionalities based on an existing system, we usually analyze the impact of modification and/or reuse of the existing components in the traditional approach. But when replacing the specification of some components with training data sets, we need to keep the consistency between such training data sets and the specifications of other components. I am not confident in impact analysis, as the current state-of-the-art, of the system including ML-based components, but I hope research activities to reveal solutions for effective management of the design specifications and the training data sets.

Eles: Yes, I can perfectly agree that the collection, evaluation, and management of data sets are main factors that are going to impact the design process. From another direction, there might come an interesting impact as well. We can look at ML-based functions not only as part of the delivered end-user functionality. ML techniques might also impact the design process since ML-based and data-driven-based approaches can and will be applied inside the design space exploration and synthesis tools in order to increase their efficiency.

Keutzer: In most conventional software systems when a bug in the field is found then, depending on the severity, a more or less experienced engineer is assigned to review the code, make changes, and verify, to whatever extent, that the bug has been eliminated. Later the change will be included in a software update. In systems which use deep learning, we have the opportunity to dramatically improve this process. For example, in a deployed autonomous vehicle that depends on deep learning, in an anomalous scenario in which the driver takes over steering, a relevant window of the camera and other sensor data can be uploaded for humans to review at the first opportunity. This data can then be annotated, added to the training set of a Deep Neural Net model, which will then be trained on the new training data, and then the updated Deep Neural Net model can be downloaded into all the relevant vehicles. This process can be significantly faster than the conventional scenario of bug identification, bug fixing, regression testing, etc., described above. Moreover, my group has a focus on the development of very small Deep Neural Net models that can even be communicated in Over The Air updates via cellular communication systems; these models are so small that they do not even require WiFi. Finally, another frontier of research is to use the window of real-world data from the anomalous scenario described above as a seed for simulating a variety of related scenarios. With a technique known as domain adaptation, it is hoped that the resulting simulation data can be combined with real-world training data to better prepare the vehicle for the most challenging real-world scenarios.

Heiser: I don't think I can add anything intelligent that hasn't been already said.

Ernst: How about the integration of dependable ML functions in complex systems, such as cars—are there credible design strategies?

Tohdo: I don't see any difference from the conventional design strategies that are adopted for reuse existing semi-black box components. An essential issue is not the complexity of systems but the complexity of the contexts that systems should interact. Using automated driving as an example, it is important to consider enough scenarios and situations which often depend on complex conditions such as the existence of opposed vehicle and weather. ML technologies might offer an easier way to realize prototype systems, but it is hard to get confidence without considering the sufficiency of assumed conditions. This discussion is not specific to ML technologies and that is the reason why I don't see any difference from the conventional design strategies.

Keutzer: One approach to create reliable complex systems is to use a formal approach to compositional reasoning; we decompose the problem into modules, and then we apply a strategy to verify the composition of the modules. One such approach is the assume/guarantee paradigm. In such an approach, the verification of the complex system is decoupled from the details of the individual models. It seems to me that this approach is appropriate for complex systems with individual elements built using ML/deep learning. So, for example, in an autonomous vehicle, we can decompose the system naturally into a sensor subsystem, a perceptual subsystem, a mapping subsystem, a localization subsystem, a motion planning and control subsystem, and so forth. We can then reason about global properties of the overall system using an assume/guarantee paradigm with each of the modules. On the other hand, another approach to autonomous driving is to encapsulate the perception and motion-planning and control subsystems in a single deep learning module that is trained end to end. There, the same assume-guarantee paradigm can be applied, but quite a lot has to be guaranteed about the behavior of a single deep learning-based module. I think that reasoning these end-to-end driving systems that use deep learning is challenging.

Kim: I can clearly say that it is the most important challenge we have to overcome to use ML in mission-critical systems, as we have just seen the tragic accident caused by an autonomous driving car. Unfortunately, I do not think that it is currently well studied, though it is being studied intensively such as "explainable AI." In my opinion, the only option we have right now is just testing. We need extensive "real-world" testing before deploying the products. And, the real-world testing scenarios should be included in ML design strategies until better technical solutions have been developed. By the way, I hope someone who has deeper technical expertise in this problem can answer the question. I really hope we have a good practical solution for it.

Hu: Will online learning fundamentally change the design of critical systems or should it be limited to what can be handled in the current safety-oriented design process? Could complementary protective methods help and, if so, which are likely candidates?

Eles: It has to change the way we look at the design of, at least, certain safety-critical systems. We have here a component very different from what we were used to handle. We have to accommodate these new ML-based modules in our systems and current processes are not fit to handle them, at least not in the way we were used to.

Tohdo: Talking about "learning performance or functionality related safety on line," I believe we need to consider broader topics (e.g., ethically aligned design [2]) before technical details. ML itself can never take responsibility for safety. I believe we need to consider, somehow, keeping developers and/or operators within the loop of online learning in order to clarify the responsibility for the safety of the system.

Ernst: Does ML lead to new security threads? How difficult will it be to protect an ML operated system?

Heiser: Indeed, they do. There is a whole field of adversarial ML, which is about fooling ML algorithms. Examples include pictures or even 3-D models that to us look like innocent objects but get recognized as guns with very high probability. Or things we easily recognize as STOP signs to which the ML is blind. And, of course, there's the inherent complexity of ML systems, which means there is a high likelihood of critical faults. As always, complexity is the arch-enemy of security.

Keutzer: Although their accuracy is generally superior to prior ML-based methods, Deep learning-based computer vision systems are currently vulnerable to adversarial attacks. However, the more successful adversarial attacks that I'm familiar with require access to the original model, or at least its associated probability distributions, and the ability to probe internal responses in order to craft the attack. There is lots of research on developing adversarial attacks and there is lots of research on making Deep Neural Nets more robust against adversarial attacks. Even in an adversarial environment, simple precautions like not exposing the Deep Neural Net model for experimentation will surely help. Moreover, we are used to craft traffic environments to accommodate, not accentuate human weaknesses. For example, color-blind humans navigate through traffic lights by relying on the relative positioning of the light. Similarly, we can provide multiple cues to Deep Neural Nets to facilitate the robust behavior. In other words, all of these adversarial attacks presume a kind of adversarial behavior. There are lots of easier ways to sabotage a stop light than to craft it to present an adversarial attack to some of the autonomous vehicles that might encounter it.

Eles: Unfortunately, this is not my niche.

Tohdo: I am not so familiar with cybersecurity. ML technologies may introduce new kinds of vulnerabilities, but I do not know whether they lead to new threats or not.

Hu: ML for an embedded system, where is the hype and where is the lasting impact?

Eles: I think that ML combined with the computational power available today provides a real potential for very important applications. We already see it and this is undeniable. Yes, there is much talk about it in the media and it excites the imagination of many people with scenarios beyond what we can achieve today. But that does not change the fact that it opens up for applications of huge impact. Now, if we restrict to "embedded systems," there are some applications with specific demands. And here is the one we always bring up as an example, autonomous driving. There is, of course, much hype behind it, sometimes creating the impression that full autonomous driving is just around the corner. It's not and the need for systematic verification is just one of the many challenges ahead. Nevertheless, while some hype is here, the real impact is already visible.

Heiser: ML will definitely have a profound impact on the embedded space, although it will be in combination with the networking of embedded devices [Internet of Thing (IoT)]. This will enable much innovation and change the ways in which we interact with such systems. As with many new technology trends, this will also create a serious risk. This does not only include the obvious risks of critical systems failing, a risk that is amplified by the increasing dependence on the technology of just anything we do in daily life. There is also the risk that insecure systems could be leveraged into large-scale attacks with massive economic cost. And there is the much-ignored risk to privacy, with intelligent IoT devices providing the infrastructure for mass surveillance, both by governments as well as organized crime.

Keutzer: I believe that time will show that the application of deep learning to embedded systems is very much underhyped and its value is just beginning to be realized. In the next decade, we will begin to see more intelligence appearing in our cars, homes, and workplaces.The source of that intelligence will be the embodiment of deep learning in embedded devices. In time, we will be surprised when everyday devices do not show some intelligence.Advances will not be the result of some astonishing breakthrough in strong AI. Advances will be due to the disciplined and systematic engineering of a very modest amount of intelligence; however, we don't need that much additional intelligence to be surprised. Our kitchens will serve us up surprisingly good new desserts not because the kitchen has some amazing robot chef with strong AI. Instead, our kitchens will learn our likes and dislikes and will draw on a large database of recipes to craft something we particularly like. Our cars will not only safely drive us along routes but also they will find novel routes by being strongly connected to other vehicles and up to date information. Meetings in our workplaces will be dynamically reconfigured because our offices will know who is in today and who had an unexpected visit to the dentist.

Kim: I believe there are both hypes and lasting impacts in today's ML phenomenon. The most obvious hype I have seen is that ML is the solution for all. It reminds me of the general-purpose graphics processing unit (GPGPU) hype that GPU will replace CPU for all, which I believe is not true. However, there is definitely the lasting impact. I have no doubt that ML provides very useful and effective tools for many engineering problems. I can give the same GPGPU analog. Though GPUs may have not replaced CPU, it has defined a new computing platform, which has eventually led to today's deep learning era. I believe the key is to understand that ML is a tool, not a goal. We need to know both the potential and limitation of ML clearly and apply ML to right problems. ML will open up new opportunities to solve very difficult problems. In such a sense, there are plenty of ML applications in the embedded system such as cars, robots, IoTs, and home appliances. ML will allow us to make products what we dreamed of, but did not have technologies to do. Thanks to ML, self-driving car, human-like robots, and more will be realized in the "near" future.

Tohdo: In general, people might expect new technologies such as AI including ML to bring new innovative values. I do not have any idea whether ML is hype or lasting impact with respect to such new values. But if talking on the established values of embedded systems such as safety, I definitely consider ML brings a lasting impact on the way how to develop and maintain systems. ML offers the alternative approach to system development by replacing some of the requirement management and system design using the training data set. This new approach to embedded system development will provide additional flexibility and will require additional efforts, as I mentioned in the panel session in Seoul.

Ernst: What is missing in the current research/ practice on ML for embedded systems and what is needed to help make ML have a lasting impact on embedded systems?

Tohdo: These few years, many research ideas that cover important topics of ML technologies to apply to the embedded systems are offered in academia. Although many of them are not yet mature enough to solve problems completely, I do not find any missing topic. On the other hand, we have lots of open issues around embedding ML in practice. These issues are not only technological ones but highly depending on application-specific demands. We have discussed a lot on automated (autonomous) driving as an example of critical systems, but the discussion points might much differ if we chose financial applications as examples. So in my opinion, it would be important to bridge the research community and the industries to get quick feedback for each other.

Eles: There are certainly several items still missing, even if we already see the impact. At least in the context of a certain category of embedded systems, we are missing the systematic verification and validation procedures that also can support certification, internal to the company or by external authorities. This is an extremely exciting research topic!

Heiser: I agree that correctness guarantees of the sort are important. However, that cannot mean completely deterministic behavior, which is not achievable with the human control either. Rather we need guarantees of a higher level notion of correctness of the sort that the system will detect a dangerous state or trajectory and will probably act to return to a safe state, even if the individual state changes are not deterministic. Such a higher level notion of safety is hard to formalize and even harder to prove. This is where I see the biggest hole in the safety verification story right now.

Keutzer: I think the most promising use of formal methods for deep learning-based computer vision is in quantitatively improving the accuracy of these systems. There are a few related ways of doing this, such as counterexample driven methods or adversarial methods. Another approach toward the same goal is to use robust optimization approaches to make the results of deep learning-based systems more generalizable to new environments. These approaches are sure to improve the robustness of deep learning-based systems which will, in turn, have a practical impact on the resulting system. Beyond that, assume/guarantee methods can be applied at a higher level to verify the system-level behavior.

Kim: A very good last question as a summary, and I can also give the answer as a summary of the previous answers. From a technical perspective, what is the most important, and though may not be completely missing but need to be improved a lot, is the “guarantee” of its behavior. Embedded systems are mission critical in many cases, where it is not easy to deploy such a system with a lot of uncertainty. The ML verification methodology will be important for it and data for ML training will play an important role. From the social perspective, we need to put our efforts to establish a social safety net that can embrace the imperfectness of ML. I do not believe we can make ML 100% perfect even with all future technical advances. So, without a proper social agreement for its imperfectness, ML will not be business successful, and money is a key factor for the lasting impact.

Ernst: Any final comments?

Keutzer: There seems to be an implicit premise in this discussion that: 1) we understand conventional software systems that do not use ML; these systems are amenable to formal verification, and many portions of deployed software systems are in fact formally verified, but that 2) we do not understand how ML methods work, and therefore, we can never formally verify them. To the first point on conventional software systems, anyone who has had responsibility for delivering a significant (e.g., 200,000–1,000,000 lines of code or more) software system knows how little we really understand about such systems. The legacy code for significant subsystems tends to live on forever precisely because no one knows how it works anymore, although it seems to function in the operation. Formal verification, if used at all in conventional software systems, is probably applied to only an abstraction of the system, or to small modules. So, with regard to ML systems, and deep learning, in particular, the situation is not any worse and is, in many ways, better. A Deep Neural Net describable in less than 1000 lines may replace a conventional software subsystem of 25,000 lines or more. While we do not understand precisely how the Deep Neural Net, after training, will perform, we do understand the basic principles of its operation. Moreover, since its design is linked to such a small initial description, it is easy to modify its design. Moreover, this style of design naturally leads itself to modularity and this, in turn, gives the potential to apply formal methods at the system level. Thus, in many ways, I feel that Deep Learning systems are more amenable to apply formal techniques, not less so. In any case, the real verification approach for most software systems is extensive testing and there's nothing about an ML system that impedes that.

Eles: For me, the most interesting aspect of this discussion is exactly about that kind of embedded software which is related to critical functionality and is supposed to be verified such that some quantifiable notion of trust is provided. It might be with formal verification or testing based on some meaningful metrics. Thus, my conclusion is that, in this particular context, much work is still to be done in order to integrate ML-based modules into such safety-critical embedded applications. I see this as an extremely exciting challenge for both research and industry.

Ernst: This was a long panel discussion on a very relevant topic. We are obviously only at the beginning of a highly dynamic development with a strong impact on engineering practice and with many new research opportunities for scientists with a background in embedded and cyber-physical systems.On behalf of my co-moderator Sharon Hu and myself, I would like to thank the roundtable participants for their active and partly controversial discussion and for sharing their insights with us.

References

[1] R. Salay, R. Queiroz, and K. Czarnecki, "An analysis of ISO 26262: Using machine learning safely in automotive software," arXiv preprint. [Online]. Available: https://arxiv.org/pdf/1709.02435.pdf

[2] [Online]. Available: https://ethicsinaction.ieee.org/