AI-Blueprint for Deep Neural Networks

Ernest Wozniak[1], Henrik Putzer[1,2], Carmen Cârlan[1]

[1]fortiss GmbH, Guerickestr. 25, 80807 Munich, Germany, lastname@fortiss.org

[2]cogitron GmbH, Stefaniweg 4, 85652 Pliening, Germany, henrik.putzer@cogitron.de

Abstract

Development of trustworthy (e.g., safety and/or security critical) hardware/software-based systems needs to rely on well-defined process models. However, engineering trustworthy systems implemented with artificial intelligence (AI) is still poorly discussed. This is, to large extend, due to the standpoint in which AI is a technique applied within software engineering. This work follows a different viewpoint in which AI represents a 3rd kind technology (next to software and hardware), with close connections to software. Consequently, the contribution of this paper is the presentation of a process model, tailored to AI engineering. Its objective is to support the development of trustworthy systems, for which parts of their safety and/or security critical functionality are implemented with AI. As such, it considers methods and metrics at different AI development phases that shall be used to achieve higher confidence in the satisfaction of trustworthiness properties of a developed system.

Introduction

A common deficiency of safety standards like ISO 26262 [ISO 26262:2018], in the automotive domain, is that they do not account explicitly for Artificial Intelligence (AI) technology [Putzer, H.J.]. However, opposed to older rumors – safety standards do not prohibit the use of AI, they just do not provide any guidelines on how to use this technology. Actually, lately, AI has reached high attention among the automotive, healthcare, or defense industry. This is due to the capabilities of AI during design (shorter time to market; implementation of implicit requirements) and during operation (improved performance). To achieve the vision in which AI not only supports, but also provides safety and/or security critical functionality, systems must be assured by evidence for trustworthy behavior of AI components.

There is a tendency that AI is regarded as software. It is suggested that the application of existing standards in their current form is adequate and that the reuse of already available processes, methods and metrics specific to software components can be used for AI.

This work advocates a different approach, where AI is considered as a 3rd kind technology (next to software (SW) and hardware (HW)), which requires its own process model to ensure trustworthiness. This is because AI, in particular Machine Learning (ML) is a new, data-driven technology, which requires a specific engineering process with specific tailored methods for assuring trustworthiness. Such a structured engineering process will be introduced by this paper, while also discussing the integration of this process into the overall system lifecycle, as presented in the VDE-AR-E 2842-61 standard [VDE-AR-E 2842-61:2020].

This paper is structured as follows. Foundations section presents the VDE-AR-E 2842-61 standard which states the main context for this work. Next section presents a generic process model called *AI-Blueprint*, upon which, process models tailored to specific AI techniques (e.g. Deep Neural Networks - DNNs, Reinforcement Learning) can be built. After that, a specific instance of AI-Blueprint for DNNs is provided, with the follow-up section showing its application for a development of CNN (Convolutional Neural Network)-based pedestrians' detection component. Finally, related work is presented, i.e. process models for the development of AI components, especially in the context of the development of safety and security critical systems. In the last section, we draw some conclusions and discuss future work.

Foundations

Fig. 1 presents the reference lifecycle defined in VDE-AR-E 2842-61 standard. The standard will consist of six parts where the three of them are already published. The reference lifecycle can be used as a reference for a process model that supports the development and assurance of a concrete trustworthy system. Trustworthiness is considered as a more generic concept that combines a user defined and potentially project specific set of aspects. These aspects include but are not limited to (functional) safety, security, privacy, usability, ethical and legal compliance, reliability, availability,

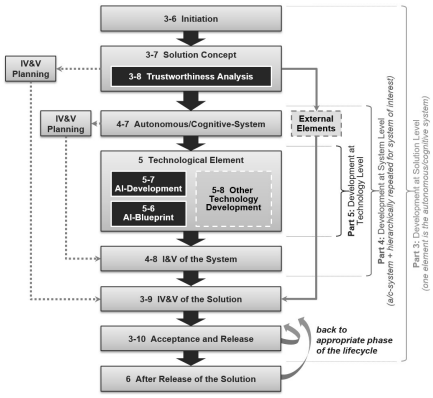

Fig. 1: Reference Lifecycle of VDE-AR-E 2842-61

maintainability, and (intended) functionality. The reference lifecycle defines the logical flow of assurance activities and is inspired by the structure of the ISO 26262 safety lifecycle. However, it is domain-independent. Detailed description of the phases can be found in [Putzer, H.J.]. In this work we focus only on the development phase dedicated to the Technological Element (see Fig. 1). The scope of this phase is to provide guidance for the implementation of elements based on a single technology (e.g. SW, HW, and especially AI).

With a suitable process interface, when using the VDE-AR-E 2842-61 standard, these process models can be borrowed from other suitable standards. For example, in automotive, SW or HW based components can be developed following the V models defined in ISO 26262. However, there is no standardized process model that can be used for AI components. Consequently, the following two sections present a concept of AI-Blueprint, which acts as a template for constructing process models for specific AI technologies, such as the one presented later, i.e. a process model for DNNs.

The AI-Blueprint

The development of AI components does not fit into existing process models (e.g. like for classical software) due to the specific nature of the AI data-driven implementation. Even inside the field of AI, different methodologies and solution concepts can have very specific requirements towards the underlying process model. This urges for a new approach in which specific characteristics of certain AI technology are targeted by specific process model.

In this paper, we introduce the concept of *AI-Blueprint*. It is a template process that shall be refined for a specific AI technique. The AI-Blueprint is characterized by Input and Output Interfaces, Structure (development phases) and Qualifications (e.g. used for the first time, or proven, i.e. used with success in many projects). The execution of an instance of the AI-Blueprint provides an AI element characterized by a predefined quality level, including guarantees to meet defined trustworthiness requirements.

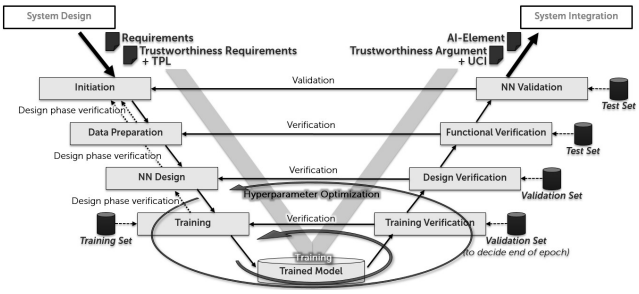

Fig. 2: AI-blueprint for DNN

The development phase at system level provides as inputs to the AI-Blueprint the system and trustworthiness requirements, together with the desired Trustworthiness Performance Level (TPL). TPL is a risk classification scheme similar to the Automotive Safety Integrity Level (ASIL) with the exception that it concerns trustworthiness, not only safety. It appoints selection of qualitative methods and metrics (M&M-s), in a systematic approach, to achieve certain TPL level.

The AI-Blueprint outputs the AI element and the value of UCI (Uncertainty-related Confidence Indicator), which are provided back to the development at system level. UCI is a quantitative indicator that describes the achieved confidence in the trustworthiness of AI component, which can be combined (in a statistically principled way) to calculate failure rate at the system level [Zhao, X.]. It represents a quantitative guarantee that a component can deliver as part of the trustworthiness contract established with the rest of a system. Desired value of UCI is defined via assigned TPL. UCI conceptually can be compared to the idea of λ expressing random hardware failures (ISO 26262 part 5).

AI-Blueprint for DNN

In this section, we instantiate the concept of AI-Blueprint for a certain technology, namely DNN. For each phase in the process model, we discuss its objectives to be achieved and the methods and metrics (M&M-s) that can be used to ensure higher trustworthiness. Significant area of research is

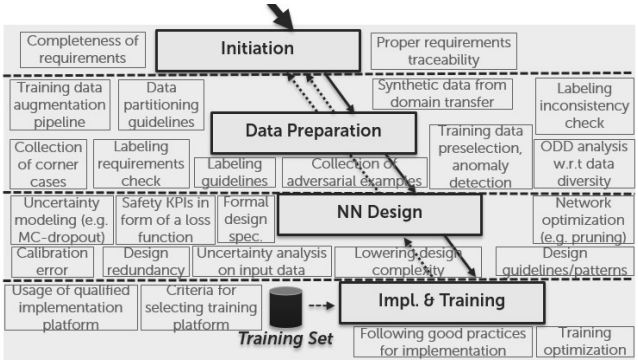

Fig. 3: Left DNN Blueprint Branch M&M-s for Trustworthiness Assurance

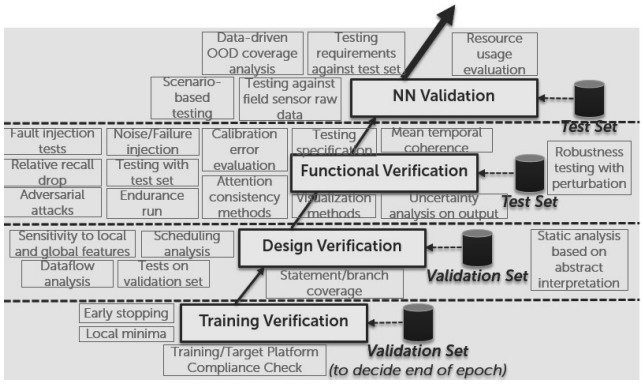

Fig. 4: Right DNN Blueprint Branch M&M-s for Trustworthiness Assurance

devoted to the identification of methods (apart from standard testing on validation and test set), and metrics that can be used to reason about the trustworthiness of DNNs. Their usage (or not) is part of the systematic approach to achieve certain TPL level. Additionally, a subset of them will contribute to the estimation of UCI. This however requires provision of a "bridge" between M&M-s and estimates of UCI, similarly as it could possibly be done for SW (see [Rushby, J]). Fig. 3 and Fig. 4 present examples of M&M-s grouped along the development phases. Still an open problem are the requirements imposed on their usage, depending on the assigned TPL. For example, ISO 26262 part 6 provides corresponding set of methods to assure confidence in the fulfillment of assurance objectives for a SW component. These methods (e.g. usage of strongly typed programming languages, formal verification, etc.) in the context of a particular ASIL level (A, B, C or D) are highly recommended, recommended, or have no recommendation for/against their usage. Similar set of rules shall be also worked out for this DNN blueprint. This is currently left for a future work.

Initiation Phase

During this phase, the team that will develop the DNN component is assembled. Then, all requirements allocated to DNN are collected and harmonized. These are product related and trustworthiness requirements specified during the system-level development phase. Further, the acceptance criteria for DNN are defined. M&M-s that can be used at this phase refer to requirements engineering.

Data Preparation Phase

The first objective of this phase is to derive data-related requirements from system-level requirements. The second objective of this phase is to gather proper data (accordingly to the requirements) and to group it into training, validation, and test sets. Validation set is used during the training, but with a purpose to assess the model convergence after each epoch. This set can be further used during the design phase to verify a design. The set of possible M&M-s that shall increase confidence in trustworthiness of an AI element (i.e. DNN) and which regards data preparation has been intensively researched. For instance, data shall account for corner cases or adversarial examples. Specific use-cases may be obtained through synthetic data generated using for instance Generative Adversarial Networks (GANs) [Esteban, C]. Further, Variational Autoencoders (VAE) may be applied to see whether data falls within the ODD (Operational Design Domain) [Vernekar, S.]. M&M-s can also be used to improve the quality of data labeling. For example, labeling inaccuracies may be circumvented by providing datasets labeled by different teams and/or technologies.

NN Design Phase

This is a phase that outputs as a main artefact the DNN design. The main difference between DNN design and DNN model is that the later contains also information about trained weights. Consequently, the main objectives of this phase are the specification of design-related requirements and the development of a model that satisfies them. The M&M-s at this phase shall contribute to the higher trustworthiness of AI element by making the design robust to failures (e.g. usage of redundancy) or noisy data (e.g. uncertainty calculation with MC-dropout), contributing to generalization property (e.g. design guidelines to select appropriate activation function, etc.), and other non-functional properties which impact trustworthiness.

Implementation & Training Phase

This phase considers as an input the DNN design and the training dataset. In order to assure higher confidence in the DNN training, the NN developer shall follow good practices for coding (e.g. for high-criticality functionality only strongly typed languages may have to be permitted). Other M&M-s can be used to optimize the training. These include but are not limited to: cropping, subsampling, scaling, etc. Higher trust can also be achieved by defining and following criteria for the training platform (e.g. level of coherence with a final execution platform). The output artefact of this phase is a DNN model, which is the main input to the verification and validation phases.

Training Verification Phase

This phase is part of the training procedure with the main purpose of controlling and verifying it. Based on the validation dataset and predefined validation metrics, the evolving DNN model is verified after each epoch. A negative outcome of the verification may require changes in the training (e.g. resignation from subsampling), or may require changes of hyperparameters, defined during the design phase.

Design Verification

This phase aims at the verification of DNN by investigating possible problems related to the NN design. The NN developer shall evaluate the result of the training, using validation dataset, and assess possible problems that may have arisen due to bad design decisions (e.g., inappropriate activation function). The NN developer may request follow-up iterations over the epochs or, if needed, the redesign of NN hyperparameters (return to Design Phase).

NN Verification Phase

The IV&V (Independent Verification and Validation) engineer shall execute a set of tests, in order to judge on the success or failure of the NN generalization, brittleness, robustness or efficiency. The judgement should be primarily based on the measured accuracy of NN over the test data set and/or identified and/or generated adversarial examples and/or corner cases. Tests may also involve faults injection or endurance tests to measure sustainability of an NN.

NN Validation, Deployment, and Release

During this final phase the IV&V engineer shall assess whether the AI element complies with all product and trustworthiness requirements. The NN shall be then integrated with hardware and/or software libraries in order to be deployed in the overall system. The resulting AI element shall be validated while running on the target platform. The final objective of this phase is to calculate the UCI in order to assess compliance of the AI element with an initially assigned TPL. If the obtained UCI value does not correspond to the TPL level, redesign decisions either at the AI level (e.g., design changes, collection of additional data) or at the system level (e.g., introduction of a redundant element to decrease assigned TPL level) shall be planned and executed.

Practical Example

The objective of this section is to showcase the traversal over the proposed DNN process model for developing a Convolutional Neural Network (CNN) for pedestrian detection. The overall context for this use-case, i.e., the System of Interest (SoI) is a pedestrian collision avoidance system. This system entails several components, SW or HW based, among which there is the AI component with the main requirement to detect pedestrians (i.e., 2D bounding box detection of pedestrians) based on the analysis of video data acquired from a single camera.

Input from the System-level

From the system-level requirements, we derive AI functional requirements, such as: **AIR01**: *relevant (defined via reachability zone) pedestrians (any person who is afoot or who is using a wheelchair or a means of conveyance propelled by human power other than a bicycle) are properly detected* and **AIR02**: *ODD is defined through the European roads.* Further, we also derive trustworthiness AI requirements, purposed mainly to counteract identified hazards, e.g.: **AITR01**: *the DNN shall output for each detected relevant pedestrian a bounding box with accurately estimated size and position accuracy in the velocity dependent detection zone, in all situations the Ego Vehicle may encounter, while being in the ODD;* **AITR02**: *pedestrians occluded up to 95% shall be properly identified;* and **AITR03**: *the DNN component shall not output false positives in the detection zone more than once in a sequence of 5 video frames.* Next to these requirements, there is also a value of the TPL, which is assigned at the system level. On a scale from A to D (highest trustworthiness criticality level), in case there are no redundant components to pedestrians detection component, the assigned TPL value is D due to the high criticality of functionality that it provides.

Example of Activities to fulfill Phases Objectives

This subsection presents examples of activities that can be executed over the different phases of the process model for CNN in order to meet the objectives of the phases and provide as a final outcome an AI element together with a trustworthiness guarantee expressed by the value of UCI.

During the **Initiation Phase** system-level requirements are refined. In the example of pedestrian recognition, examples of the refined product requirements are: **AIR01 → AIR01.01**: *pedestrians of min. width (20 pixels) and min. height (20 pixels) shall be classified;* **AIR02 → AIR02.01**: *ODD shall consider right-lane and left-lane traffic.*

Further, we refine the AI trustworthiness requirements as follows: **AITR01 → AITR01.01**: *the value of mean average precision shall be greater or equal to 97.9%*; and **AITR02 → AITR02.01**: *the data samples shall include examples with a sufficient range of levels of occlusion giving partial view of pedestrians at crossings*.

Next, to fulfill further objectives of this phase, the team developing the AI element has to be assembled and, if necessary, the DNN blueprint needs to be adjusted to reflect further identified needs expressed by the team.

The first activity during the **Data Preparation Phase** is to look over the requirements and process those that impact the data gathering and labeling activities. For instance, **AITR02** has an implicit impact on the data because, to properly train and test the model, data shall contain pedestrians with different levels of occlusion (up to 95%). There could also exist data related concerns explicitly expressed, such as **AIR03**: *the data samples shall include a sufficient range of examples reflecting the effects of identified system failure modes*, **AIR04**: t*he format of each data sample shall be representative of that which is captured using sensors deployed on the*

ego vehicle or **AIR05**: *the data samples shall include sufficient range of pedestrians within the scope of the ODD.* The data shall be then gathered, labeled and properly stored, according to the identified requirements.

The **Design Phase** shall first analyze requirements which may have implications on a CNN design. Examples of such requirements are: **AIR06**: *the DNN shall be robust against all types of foreseeable noise*, **AIR07**: *a diagnosis function shall exist in order to detect distributional shift in the environment (out of ODD detection)* or **AIR08**: *plausibility checks of detected bounding boxes are necessary (e.g. pedestrians usually do not fly)*. The designer shall then specify the CNN in terms of the number of convolutional layers, kernel size, number of fully connected layers, neurons in each layer, loss function, and other hyperparameters, to provide a design that will best serve the intended purpose. The analyzed requirements shall also steer design activities. For instance, **AIR06** requests robustness to various types of noise which can occur in the input data. The presence of noise may also be problematic even during the training as it may lead to overfitting. Further, **AIR07** may be handled by either introducing additional component (in such case the solution would affect system level) based on VAE, which can identify distributional shift, or the CNN itself may use MC-dropout to calculate uncertainties, where high uncertainty may result from out of ODD input. **AIR08** may be accommodated by design through additional knowledge injected into the CNN (neural symbolic integration) that rejects labels that based on human knowledge make no sense.

The first activity of the **Implementation and Training Phase** is to provide a code for the CNN. The coding activity can follow standard SW development practices recommended in ISO 26262 part 6. However, certain differences such as tooling, libraries or available programming languages (Python or preferably C++) create additional challenges to this activity. Next, the training platform needs to be selected with justification and training related parameters shall be provided, e.g. batch size = 1024, number of epochs = 4, number of iterations = 10, learning rate = 0.001 or decay factor = 0.9. Then, according to predefined parameters the training shall be assessed during the **Training Verification Phase**. The parameters may be tuned if necessary (e.g. model does not converge) or early stopping triggered if the model has learned to extract all the meaningful relationships from the data before starting to model the noise.

Having the trained model, verification and validation activities can be performed. Verification shall primarily investigate key trustworthiness concerns which are specific to CNNs. These are robustness, brittleness, efficiency, ODD coverage, distributional shift, unknown behavior in rare situations (corner cases or adversarial examples). Verification starts at **Design Verification Phase** in which performed activities shall encounter possible problems regarding mentioned properties in relation to the design. For instance, one could perform design investigation to analyze neurons activation. This contributes to the explainability of how pedestrians are identified and may allow to prune those neurons, which do not play any role in the decision process. Pruning may also be used to limit the number of neurons, to possibly eliminate problems of overfitting. M&M-s that shall be applied can be classified as white-box because they refer to internal characteristics of CNNs. **Functional Verification Phase** activities shall also target verification of key trustworthiness concerns, however more from the grey/black-box perspective. Here not only CNN itself is verified, but also the data. For example, if the CNN does not detect pedestrians on a wheelchair, most likely the CNN was never fed with such training examples. Explainability could be further enhanced by using attention based methods. For example, heat maps (grey-box method) may reveal those features from the image which are used to identify pedestrians. These may be different body parts, or maybe just vertical lines. The result highly depends on the level of features annotation performed during the labeling. If it is not detailed enough, the NN tester may request extended feature annotation to be performed at the data preparation phase.

The validation is executed in the last phase, i.e., **NN Validation, Deployment, and Release.** Its main activity centers at the validation of the requirements provided as an input to the initiation phase, and their refined versions elaborated at that phase. For instance, to validate **AIR01.01** one has to identify input images within the test set on which there are pedestrians with height and width close to 20 pixels and see whether these are properly detected. In case they are not, either requirements shall be changed or the training data shall be checked to identify whether enough samples was there to train the model for this requirement.

Output provided to the System-level

The artefact output by the DNN process model is a trained and verified CNN model for pedestrians' detection. Next to it, the UCI value is computed, which accounts for M&M-s being used throughout the blueprint and their efficiency in minimizing risks of possible hazards which may occur.

Related Work

The fact that there is a need for a dedicated process model for the development of AI components within safety and/or security critical systems has been underlined more than 20 years ago by Rodvold, who proposed a formal development methodology and a validation technique for AI trustworthiness assurance [Rodvold, D.M]. While the phases in the process model proposed by Rodvold resemble the phases of our proposed AI-Blueprint, Rodvold does not discuss the met-

rics and corresponding methods that can be used for the implementation and the verification of the considered trustworthiness requirements.

Microsoft presents a nine-stage ML workflow for the development of AI-based applications [Amershi, S]. The workflow is claimed to be used by multiple teams inside Microsoft, for diverse applications, and to have been integrated into overall, preexisting agile software engineering processes. Amershi et. al. categorize the workflow stages as data-oriented (e.g., collection, cleaning, and labeling) and model-oriented (e.g., model requirements, feature engineering, training, evaluation, deployment, and monitoring). While these stages are similar to the ones in our proposed AI-Blueprint, the Microsoft workflow does not consider activities specific for trustworthiness assurance, as their workflow is only intended to be used for the implementation of non-critical functionality.

Ashmore et. al. present a process model for ML components in critical systems, consisting of four phases: Data Management, Model Learning, Model Verification and Model Deployment [Ashmore, R]. For each phase in the model, they define the assurance-related desiderata (i.e., objectives) and they discuss how state-of-the-art methods may contribute to the achievement of those desiderata. The work presented in this paper is complementary to the work of Ashmore et. al.. First, AI-Blueprint for DNN elaborates more on the validation and verification of AI components, having separate phases for design verification, verification of functional requirements and validation of the AI component w.r.t. trustworthiness requirements. Second, instead of proposing a general AI process model, we advocate the need for both a higher-level template for process models guiding the development of AI components (i.e., AI-Blueprint), and more concrete process models for particular AI technologies (e.g., AI-Blueprint for DNNs). Third, we discuss the process models in the context of the overall system lifecycle.

Toreini et. al. examine the qualities technologies should have to support trust in AI-based systems, but from the perspective of social sciences [Toreini, E]. They present an interesting, but abstract machine learning pipeline, whose phases could be aligned to those in the AI-Blueprint for DNNs. Nevertheless, they do not offer detailed description of each of the phases, neither they deliberate on specific M&M-s and how they shall be used to increase the confidence in trustworthy solution.

Conclusions and Future Work

This paper presented the concept of AI-Blueprint and an example of how to use this blueprint for tailoring a process model for a certain AI technology (i.e., DNNs), with the scope of supporting trustworthiness assurance. We also discussed how the proposed AI-Blueprint fits in an overall framework for the development of trustworthy autonomous/cognitive systems, regulated in the upcoming VDE-AR-E 2842-61 standard. This work however is still in its early phases. As a future work, recommendations for specific metrics and methods, advertised along the DNN blueprint, should be established, based on the TPL levels assigned to AI element. Also, current research status regarding feasibility or performance of these methods should be investigated, to eliminate those which cannot be used while developing industry size DNNs. Next, further research on how to calculate the value for the newly introduced UCI concept is necessary. Finally, having the concept of AI-Blueprint, new blueprints, such as for reinforcement learning, neural symbolic integration could be derived.

References

Putzer, H.J. and Wozniak, E., 2020. A Structured Approach to Trustworthy Autonomous/Cognitive Systems. arXiv preprint arXiv:2002.08210.

ISO 26262 Road vehicles – Functional safety, 2018

VDE-AR-E 2842-61 - Design and Trustworthiness of autonomous/cognitive systems, 2020

Esteban, C., Hyland, S.L. and Rätsch, G., 2017. Real-valued (medical) time series generation with recurrent conditional gans. arXiv preprint arXiv:1706.02633.

Vernekar, S., Gaurav, A., Abdelzad, V., Denouden, T., Salay, R. and Czarnecki, K., 2019. Out-of-distribution detection in classifiers via generation. arXiv preprint arXiv:1910.04241.

Rodvold, D.M., 1999, July. A software development process model for artificial neural networks in critical applications. In IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339) (Vol. 5, pp. 3317-3322). IEEE.

Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B. and Zimmermann, T., 2019, May. Software engineering for machine learning: A case study. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP) (pp. 291-300). IEEE.

Ashmore, R., Calinescu, R. and Paterson, C., 2019. Assuring the machine learning lifecycle: Desiderata, methods, and challenges. arXiv preprint arXiv:1905.04223.

Toreini, E., Aitken, M., Coopamootoo, K., Elliott, K., Zelaya, C.G. and van Moorsel, A., 2020, January. The relationship between trust in AI and trustworthy machine learning technologies. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (pp. 272-283).

Rushby, J., 2009, November. Software verification and system assurance. In 2009 Seventh IEEE International Conference on Software Engineering and Formal Methods (pp. 3-10). IEEE.

Zhao, X., Robu, V., Flynn, D., Salako, K. and Strigini, L., 2019, October. Assessing the safety and reliability of autonomous vehicles from road testing. In 2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE) (pp. 13-23). IEEE.