A Survey of Software Quality for Machine Learning Applications

Satoshi Masuda, Kohichi Ono, Toshiaki Yasue, Nobuhiro Hosokawa
IBM Research - Tokyo
Chuou-ku, Tokyo, Japan 103-8510
Email: {smasuda, onono, yasue, carvin}@jp.ibm.com

***Abstract*—Machine learning (ML) is now widespread. Traditional software engineering can be applied to the development ML applications. However, we have to consider specific problems with ML applications in therms of their quality. In this paper, we present a survey of software quality for ML applications to consider the quality of ML applications as an emerging discussion. From this survey, we raised problems with ML applications and discovered software engineering approaches and software testing research areas to solve these problems. We classified survey targets into Academic Conferences, Magazines, and Communities. We targeted 16 academic conferences on artificial intelligence and software engineering, including 78 papers. We targeted 5 Magazines, including 22 papers. The results indicated key areas, such as *deep learning*, *fault localization*, and *prediction*, to be researched with software engineering and testing.**

***Index Terms*—Machine Learning, Software Quality, Software Engineering and Testing**

I. Introduction

Machine learning (ML) applications, such as face recognition, question answering, and sales analysis, are now widespread. Many ML services are available from APIs on the cloud and are called ML-as-a-services (MLaaS). ML applications often consist of such MLaaS. When we develop ML applications based on requirements, traditional software engineering can be applied. However, we have to consider specific problems with ML applications in terms of their quality. The problems are that training data determines the logic of ML applications, and the results of ML application to unknown data cannot be verified in terms of correctness. Hence, new software engineering approaches are required to solve the problems. In this paper, we present a survey of software quality for ML applications to consider the quality of ML applications. From the survey, we raised problems with ML applications and discovered software engineering approaches and software testing research areas to solve these problems.

We classified survey targets into Academic Conferences, Magazines, and Communities. We targeted 16 academic conferences on artificial intelligence and software engineering, including 78 papers. We targeted 5 Magazines, including 22 papers.

II. Procedure of survey

We present the procedure of our survey in this section.

A. Survey objective

The main objectives of the survey is to discover techniques to evaluate and improve the software quality of ML applications, and discuss future software testing research area. The sub objective of the survey is to find solutions to software testing problems for ML applications. Figure 1 shows an example of an ML application and problems in software quality for that ML applications from industry experiences. We focus the following problems:

- *How to verify the answer that MLaaS return to unknown data.*
- *How to verify insufficient or biased data because problems because data determine the logic of ML applications.*
- *How to verify the quality of end-to-end systems.*
- *How to notify users confidence of answer correctness from the system.*

These problems have not been solved by traditional software engineering or testing, and are candidates of research area for research in software testing.

B. Survey targets

The 16 targeted academic conferences on artificial intelligence and software engineering are as follows:

- AAAI: Association for the Advancement of Artificial Intelligence
- ASE: IEEE/ACM International Conference on Automated Software Engineering
- CVPR: IEEE Conference on Computer Vision and Pattern Recognition

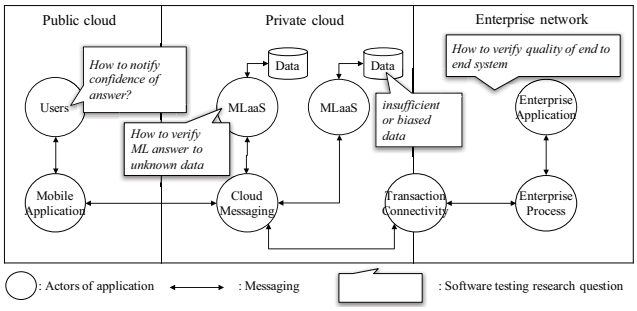

Fig. 1. ML Application and software testing research questions

0-7695-6432-1/18/$31.00 ©2018 IEEE
DOI 10.1109/ICSTW.2018.00061

- FAS*W: Foundations and Applications of Self* Systems, IEEE International Workshops on
- ICML: International Conference on Machine Learning
- ICST: IEEE International Conference on Software Testing, Verification and Validation
- ICSTW: Workshop on IEEE International Conference on Software Testing, Verification and Validation
- IEEE Transaction: IEEE Transactions on Software Engineering
- IJCAI: International Joint Conference on Artificial Intelligence
- ISSTA: ACM SIGSOFT International Symposium on Software Testing and Analysis
- ITAW: Information Theory and Applications Workshop
- KDD: ACM SIGKDD Conference on Knowledge Discovery and Data Mining
- MELECON: Mediterranean Electrotechnical Conference
- NIPS: Annual Conference on Neural Information Processing Systems
- PUC: Personal and Ubiquitous Computing
- SOSP: ACM Symposium on Operating Systems Principles

The five targeted magazines are as follows:

- AI Magazine
- IEEE Spectrum
- MIT news - Artificial intelligence
- Analytics Magazine
- OpenAI

The two targeted communities are the developer communities of software engineers.

C. Survey Steps

We executed the survey in the following steps:

1) Search articles by retrieving key words: *quality, verification, engineering, design* and other words related to software engineering.
2) Read articles.
3) Tag one key word to articles to classify them. The tags are based on the *Machine Learning Dictionary* [1] and *Artificial Intelligence Vocabulary* [2]. If we could not find an appropriate word in the vocabulary, we created a tag from the contents of the article. Some academic papers in the articles have key words in their contents; however, the key words indicate so extensive knowledge that we could not classify the articles.
4) Make comments and evaluate. We evaluated the articles as representative papers **R** or subsidiary papers **S**.

III. Survey Results

We present the survey results in this section and summarize the methods and techniques that described in the articles.

A. Results

Table I lists the conferences and the number of articles. The total number of articles was 101 from the search results. This number was too small with respect to articles in each conferences. For instance, about 700 papers were accepted at AAAI2017. This small number indicates that the topics of *quality, verification, engineering, and design* are not yet major topics. Table II lists tags and the number of tags for each evaluations sorted by the number of evaluations. *Deep learning* was the top-ranked tag. Deep learning has been the most important technique in ML, so there were many articles related to the search words *quality, verification, engineering* and *design*. The second-ranked tag was *Fault localization*, and the third was *Prediction*. Fault localization is a software engineering term, so it indicates that software engineering may shift to ML.

B. Software Quality methods and techniques for ML applications

We summarize the software-quality methods and techniques for ML applications from the top 7 ranked tags in the survey results.

1) Deep Learning: Pei et al. discussed how to find a "corner case" (case rarely occurs, a troublesome case) in a deep learning system and a test method using it [80]. Hsu mentioned the method in the article "A New Way to Find Bugs in Self-Driving AI Could Save Lives" [96]. Bau et al. proposed a technique for interpreting what internal firing of a deep learning model corresponds to using semantic segmentation [28]. Schulam et al. proposed a method of generating counterfactual predictions [73]. The method targeted a circumstances that if prediction can be executed by using experience-based data the predictions have good results, but experience-based data cannot be always used due to ethical problems. Zhang et al., Sangkloy et al. proposed verification method of visual pattern in deep learning [14], [19]. Kokkinos et al. discussed an innovative method of deep learning. They proposed a method that train in an end-to-end manner a convolutional neural network (CNN) that jointly handles low-, mid-, and high-level vision tasks in a unified architecture [22].

2) Fault Localization: Wong et al. surveyed software fault localization [44]. Software fault localization, the act of identifying the locations of faults in a program, is widely recognized as one of the most tedious, time consuming, and expensive activities in program debugging. Fault localization is challenging for ML applications [59]. Le et al. proposed a new fault localization approach that employs a learning-to-rank strategy. The approach includes ranking methods based on their likelihood of being a root cause of a failure. Sun et al. investigated several coverage-based statistical fault localization metrics and reported on an empirical study they conducted to assess the relative importance of those metrics elements [37].

3) Prediction: Prediction and inference are important functions of applications with ML applications. As software systems increase in complexity and operate with less human supervision, it becomes more difficult to use traditional techniques to detect when software is not behaving as intended [62]. For instance, predicting human behavior in front of software systems helps better design software. Katz et al. proposed techniques to model the behavior of executing programs

TABLE I
NUMBER OF ARTICLES

Category	Name	Number of articles	References
Academic conferences	AAAI2016	3	[3], [4], [5]
	AAAI2017	4	[6], [7], [8], [9]
	ASE2015	1	[10]
	ASE2016	3	[11], [12], [13]
	CVPR017	17	[14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]
	FAS*W2016	1	[31]
	ICML2015	1	[32]
	ICML2017	1	[33]
	ICST2015	3	[34], [35], [36]
	ICST2016	3	[37], [38], [39]
	ICST2017	3	[40], [41], [42]
	ICSTW2011	1	[43]
	IEEE Transactions	1	[44]
	IJCAI2016	4	[45], [46], [47], [48]
	IJCAI2017	6	[49], [50], [51], [52], [53], [54]
	ISSTA2013	3	[55], [56], [57]
	ISSTA2016	2	[58], [59]
	ISSTA2017	4	[60], [61], [62], [63]
	ITAW2017	1	[64]
	KDD2016	2	[65], [66]
	KDD2017	2	[67], [68]
	MELECON16	1	[69]
	NIPS2015	1	[70]
	NIPS2017	8	[71], [72], [73], [74], [75], [76], [77], [78]
	PUC2016	1	[79]
	SOSP2017	1	[80]
Community	Commercial	1	[81]
	Developer	1	[82]
Magazines	AI Magazine	9	[83], [84], [85], [86], [87], [88], [89], [90], [91]
	Business Insider	1	[92]
	IEEE Spectram	5	[93], [94], [95], [96], [97]
	Open AI	6	[98], [99], [100], [101], [102], [103]
Total		101	

TABLE II
NUMBER OF TAGS

Tags	**R** [1]	**S** [2]	Sub Total
Deep learning	3	6	9
Fault localization	3		3
Prediction	3		3
MLaaS	2	3	5
Multi agent	2	2	4
Search-Based	2	2	4
Model checking	2	1	3
AI business	1	7	8
Genetic algorithm	1	1	2
Robot	1	1	2
Technical debt	1		1
Democratizing AI	1		1
AI ethics	1		1
Continuous integration	1		1
Adversarial	1		1
AI for complex	1		1
Lifelong Learning Machines	1		1
Graph quality	1		1
Testing		7	7
Neural network		7	7
Safety		7	7
Software development		6	6
Software engineering		4	4
AI overview		3	3
Testing automation		2	2
Symbolic execution		1	1
Static analysis		1	1
Timed failure		1	1
Auto encoder		1	1
Pairwise		1	1
Algorithm design		1	1
Security		1	1
Training data		1	1
Speaker verification		1	1
Linear dynamic logic		1	1
3d model		1	1
Mechanism design		1	1
Label data		1	1
Metrics		1	1
Total	28	73	101

1. **R**: Representative papers, 2. **S**: Subsidiary papers

using low-level signals collected during executions [62]. The models provide a basis for predicting whether an execution of a program or program unit under test represents intended behavior. Hotzkow et al. presented automatically inferring user expectations from the semantic contexts over multiple applications [63]. Once the user expectations are established, this knowledge can be used as an oracle

4) MLaaS: Ribeiro et al. defined MLaaS as a ready-to-use ML functions. They proposed an architecture to create a flexible and scalable MLaaS [32]. VanDerHerten et al. developed a proof of concept for adaptive modeling and sampling methodologies for Internet-of-Things applications with MLaaS [69]. Assem et al. developed a methodology to build an application with train, compare, decide, and change approach with MLaaS [79]. This is more practical than theoretical; however, it seems to be the most helpful reference in the real world. For concrete application, Bosse et al. developed distributed ML with self-organizing mobile agents for earthquake monitoring [31].

5) Multi Agent: Multi agent is a key approach to adapt software engineering into ML applications such as autonomous vehicles and robots. Verifying a data-aware multi-agent system (DAMAS) is challenging because of the infinite state models generated by their infinite-domain variables. Belardinelli et al. proposed a method of parameterized DAMAS (P-DAMAS) as a system with an unbounded number of homogenous agents, each assumed to be data-aware, i.e., endowed with possibly infinite domains and interacting with an environment composed of partially shared data [53]. Wooldridge et al. presented formal models through which rational verification can be studied, and surveyed the complexity of key decision problems in multi-agent systems [4].

6) Search-Based: Search-Based Software Engineering is the name given to a body of work in which search-based optimization is applied to software engineering [34]. McMinn

TABLE III
CORRESPONDENCE OF PROBLEMS AND TAGS

Problems	Tags						
	Deep learning	Fault localization	Prediction	MLaaS	Multi agent	Search-Based	Model checking
How to verify the answer that MLaaS return to unknown data	[80]		[62]				[40]
How to verify insufficient or biased data because problems because data determine the logic of ML applications.	[73]				[53]		
How to verify the quality of end-to-end systems.		[59]		[79]	[4]	[42]	[47]
How to notify users confidence of answer correctness from the system.			[63], [66]				

et al. presented Search-Based Software Testing, which is the use of a meta-heuristic optimizing search technique, such as a Genetic Algorithm, to automate or partially automate a testing task; for example, the automatic generation of test data [43]. Gay2017 proposed assessed search-based generation of test suites that detect real faults [42].

7) Model Checking: Model checking is a traditional technique of software engineering. However, describing applications models with ML allows the application model checking to verify the applications. Alechina et al. proposed a method for verifying existence of resource-bounded coalition uniform strategies to be able to automatically verify properties of such systems using model-checking [47]. Tappler et al. presented learning-based approach to detecting failures by model-based testing [40].

IV. DISCUSSION

Table III shows articles that have correspondence between problems and top seven tags. For instance, Pei et al. [80] discussed a corner-case in a deep learning system and a white-box testing method using the corner-case. The corner-case is an unknown data for ML applications; hence, their paper corresponds to the problem. We put corresponding papers to each problems in the same manner. As summarized in Section 3, we found several techniques or methods of software quality for ML applications. Some of these techniques corresponded to problems that we raised. However, the problems have not been completely solved. Each papers covered some of the problems. For instance, Pei et al. targeted deep learning systems but did not target other ML models such as *k*-means, decision-tree, and support vector machine. ML applications consist of various ML models; therefore, techniques of software quality or testing are required for verifying of ML applications.

V. CONCLUSION

ML applications such as face recognition, question answering, and sales analysis are now widespread. New software engineering approaches are required to solve the problems. We presented a survey of software quality for ML applications to consider the quality of ML applications. From our survey determined problems with ML applications and discovered software engineering approaches and software testing research areas to solve these problems.

REFERENCES

[1] B. Wilson, "The Machine Learning Dictionary," 2012. [Online]. Available: http://www.cse.unsw.edu.au/ billw/mldict.html

[2] G. S. Novak Jr., "Artificial Intelligence Vocabulary," 2005. [Online]. Available: https://www.cs.utexas.edu/users/novak/aivocab.html

[3] B. Zarrieß and J. Claßen, "Decidable Verification of Golog Programs over Non-Local Effect Actions," pp. 1109–1115.

[4] M. Wooldridge, J. Gutierrez, P. Harrenstein, E. Marchioni, G. Perelli, and A. Toumi, "Rational Verification: From Model Checking to Equilibrium Checking," *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, pp. 4184–4190, 2016.

[5] B. Bittner, M. Bozzano, A. Cimatti, and G. Zampedri, "Automated Verification and Tightening of Failure Propagation Models," *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, pp. 907–913, 2016.

[6] M. Witbrock, "AI for Complex Situations: Beyond Uniform Problem Solving," 2017.

[7] V. Vanhoucke and G. B. Robotics, " ' OK Google , fold my laundry s ' il te plaît '."

[8] T. Sunahase, Y. Baba, and H. Kashima, "Pairwise HITS : Quality Estimation from Pairwise Comparisons in Creator-Evaluator Crowdsourcing Process," *Proceedings of the 31th Conference on Artificial Intelligence (AAAI 2017)*, no. Kleinberg, pp. 977–983, 2017.

[9] J. Goldsmith and E. Burton, "Why teaching ethics to AI practitioners is important," *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pp. 110–114, 2017.

[10] A. N. Lam, A. T. Nguyen, H. A. Nguyen, and T. N. Nguyen, "Combining deep learning with information retrieval to localize buggy files for bug reports," *Proceedings - 2015 30th IEEE/ACM International Conference on Automated Software Engineering, ASE 2015*, pp. 476–481, 2016.

[11] M. White, M. Tufano, C. Vendome, and D. Poshyvanyk, "Deep learning code fragments for code clone detection," *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering - ASE 2016*, pp. 87–98, 2016.

[12] X. Li, Y. Liang, H. Qian, Y.-Q. Hu, L. Bu, Y. Yu, X. Chen, and X. Li, "Symbolic Execution of Complex Program Driven by Machine Learning Based Constraint Solving," *Ase*, pp. 554–559, 2016.

[13] N. Li, Y. Lei, H. R. Khan, J. Liu, and Y. Guo, "Applying combinatorial test data generation to big data applications," *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering - ASE 2016*, pp. 637–647, 2016.

[14] W. Zhang, X. Cao, R. Wang, Y. Guo, and Z. Chen, "Binarized Mode Seeking for Scalable Visual Pattern Discovery," *Cvpr2017*, pp. 3864–3872, 2017.

[15] J. Wu, J. B. Tenenbaum, and P. Kohli, "Neural Scene De-rendering," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 699–707, 2017.

[16] W. Treible, P. Saponaro, S. Sorensen, A. Kolagunda, M. O. Neal, B. Phelan, K. Sherbondy, and C. Kambhamettu, "CATS : A Color and Thermal Stereo Benchmark," *Cvpr*, pp. 134–142, 2017.

[17] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from Simulated and Unsupervised Images through Adversarial Training," pp. 2242–2251, 2016.

[18] K. Sasaki, S. Iizuka, E. Simo-serra, and H. Ishikawa, "Joint Gap Detection and Inpainting of Line Drawings," *Cvpr*, pp. 5768–5776, 2017.

[19] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling Deep Image Synthesis with Sketch and Color," pp. 6836–6845, 2016.

[20] K. Nakamura, S. Yeung, A. Alahi, and L. Fei-Fei, "Jointly Learning Energy Expenditures and Activities using Egocentric Multimodal Signals," *Cvpr*, pp. 6817–6826, 2017. [Online]. Available: http://vision.stanford.edu/pdf/nakamura2017cvpr.pdf

[21] H. Le, T.-j. Chin, and D. Suter, "An Exact Penalty Method for Locally Convergent Maximum Consensus," *Cvpr2017*, pp. 1888–1896, 1888.

[22] I. Kokkinos, "UberNet: Training a 'Universal' Convolutional Neural Network for Low-, Mid-, and High-Level Vision using Diverse Datasets and Limited Memory," 2016.

[23] W. Kehl, F. Tombari, S. Ilic, and N. Navab, "Real-Time 3D Model Tracking in Color and Depth on a Single CPU Core," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 465–473, 2017.

[24] D. He, X. Yang, C. Liang, Z. Zhou, D. Kifer, C. L. Giles, and A. Ororbia, "Multi-scale FCN with Instance Aware Segmentation for Arbitrary Oriented Word Spotting In The Wild," *Cvpr 2017*, no. 1, 2017.

[25] C. Fan, J. Lee, M. Xu, K. K. Singh, Y. J. Lee, D. J. Crandall, and M. S. Ryoo, "Identifying First-person Camera Wearers in Third-person Videos," no. 1, pp. 4734–4742, 2017.

[26] Y. Chen and Y.-j. Liu, "Learning to Rank Retargeted Images," *Learning*, pp. 3994–4002, 2017.

[27] L. Castrejn, K. Kundu, R. Urtasun, and S. Fidler, "Annotating object instances with a polygon-rnn," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 4485–4493.

[28] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 3319–3327.

[29] X. Alameda-Pineda, A. Pilzer, D. Xu, N. Sebe, and E. Ricci, "Viraliency: Pooling local virality," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 484–492.

[30] V. K. Adhikarla, M. Vinkler, D. Sumin, R. K. Mantiuk, K. Myszkowski, H. P. Seidel, and P. Didyk, "Towards a quality metric for dense light fields," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 3720–3729.

[31] S. Bosse, "Distributed machine learning with self-organizing mobile agents for earthquake monitoring," *Proceedings - IEEE 1st International Workshops on Foundations and Applications of Self-Systems, FAS-W 2016*, pp. 126–132, 2016.

[32] M. Ribeiro, K. Grolinger, and M. A. Capretz, "MLaaS: Machine Learning as a Service," *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, no. c, pp. 896–902, 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7424435/

[33] B. Chen, D. Kumor, and E. Bareinboim, "Identification and Model Testing in Linear Structural Equation Models using Auxiliary Variables," *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, pp. 757–766, 2017. [Online]. Available: http://proceedings.mlr.press/v70/chen17f.html

[34] M. Harman, P. McMinn, J. De Souza, and S. Yoo, "Search based software engineering: Techniques, taxonomy, tutorial," *Search*, vol. 2012, pp. 1–59, 2011. [Online]. Available: http://discovery.ucl.ac.uk/1340709/

[35] N. Erman, V. Tufvesson, M. Borg, P. Runeson, and A. Ardö, "Navigating information overload caused by automated testing - A clustering approach in multi-branch development," *2015 IEEE 8th International Conference on Software Testing, Verification and Validation, ICST 2015 - Proceedings*, 2015.

[36] R. Carbone, L. Compagna, A. Panichella, and S. E. Ponta, "Security threat identification and testing," *2015 IEEE 8th International Conference on Software Testing, Verification and Validation, ICST 2015 - Proceedings*, 2015.

[37] S. F. Sun and A. Podgurski, "Properties of Effective Metrics for Coverage-Based Statistical Fault Localization," *Proceedings - 2016 IEEE International Conference on Software Testing, Verification and Validation, ICST 2016*, pp. 124–134, 2016.

[38] K. Moran, M. Linares-Vasquez, C. Bernal-Cardenas, C. Vendome, and D. Poshyvanyk, "Automatically Discovering, Reporting and Reproducing Android Application Crashes," *Proceedings - 2016 IEEE International Conference on Software Testing, Verification and Validation, ICST 2016*, pp. 33–44, 2016.

[39] B. Marculescu, R. Feldt, and R. Torkar, "Using Exploration Focused Techniques to Augment Search-Based Software Testing: An Experimental Evaluation," *Proceedings - 2016 IEEE International Conference on Software Testing, Verification and Validation, ICST 2016*, pp. 69–79, 2016.

[40] M. Tappler, B. K. Aichernig, and R. Bloem, "Model-Based Testing IoT Communication via Active Automata Learning," *Proceedings - 10th IEEE International Conference on Software Testing, Verification and Validation, ICST 2017*, pp. 276–287, 2017.

[41] D. Pradhan, S. Wang, S. Ali, T. Yue, and M. Liaaen, "CBGA-ES: A Cluster-Based Genetic Algorithm with Elitist Selection for Supporting Multi-Objective Test Optimization," *Proceedings - 10th IEEE International Conference on Software Testing, Verification and Validation, ICST 2017*, pp. 367–378, 2017.

[42] G. Gay, "The Fitness Function for the Job: Search-Based Generation of Test Suites That Detect Real Faults," *Proceedings - 10th IEEE International Conference on Software Testing, Verification and Validation, ICST 2017*, pp. 345–355, 2017.

[43] P. McMinn, "Search-Based Software Testing: Past, Present and Future," *2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops*, pp. 153–163, 2011. [Online]. Available: http://ieeexplore.ieee.org/document/5954405/

[44] W. E. Wong, R. Gao, Y. Li, R. Abreu, and F. Wotawa, "A Survey on Software Fault Localization," *IEEE Transactions on Software Engineering*, vol. 42, no. 8, pp. 707–740, 2016.

[45] H. Narasimhan, S. Agarwal, and D. C. Parkes, "Automated mechanism design without money via machine learning," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2016-Janua, pp. 433–439, 2016.

[46] Y. F. Li, S. B. Wang, and Z. H. Zhou, "Graph quality judgement: A large margin expedition," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2016-Janua, pp. 1725–1731, 2016.

[47] N. Alechina, M. Dastani, and B. Logan, "Verifying existence of resource-bounded coalition uniform strategies," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2016-Janua, pp. 24–30, 2016.

[48] S. Adriaensen and A. Nowé, "Towards a white box approach to automated algorithm design," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2016-Janua, pp. 554–560, 2016.

[49] V. Noroozi, L. Zheng, S. Bahaadini, S. Xie, and P. S. Yu, "SEVEN: Deep SEmi-supervised verification networks," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 2571–2577, 2017.

[50] P. Kouvaros and A. Lomuscio, "Verifying fault-tolerance in parameterised multi-agent systems," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 288–294, 2017.

[51] J. Kong and A. Lomuscio, "Model checking multi-agent systems against LDLK specifications," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 1138–1144, 2017.

[52] N. Gorogiannis and F. Raimondi, "A Novel Symbolic Approach to Verifying Epistemic Properties of Programs ," pp. 206–212, 2009.

[53] F. Belardinelli, L. Ibisc, and I. Toulouse, "Parameterised Verification of Data-aware Multi-agent Systems," pp. 98–104, 2016.

[54] F. Belardinelli, L. Ibisc, A. Murano, and S. Rubin, "Verification of Broadcasting Multi-Agent Systems against an Epistemic Strategy Logic Imperial College London," pp. 91–97, 2014.

[55] O. Tripp, O. Weisman, and L. Guy, "Finding Your Way in the Testing Jungle: A Learning Approach to Web Security Testing," *Proceedings of the 2013 International Symposium on Software Testing and Analysis*, pp. 347–357, 2013. [Online]. Available: http://doi.acm.org/10.1145/2483760.2483776

[56] F. M. Kifetew, A. Panichella, A. De Lucia, R. Oliveto, and P. Tonella, "Orthogonal exploration of the search space in evolutionary test case generation," *Proceedings of the 2013 International Symposium on Software Testing and Analysis - ISSTA 2013*, p. 257, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2483760.2483789

[57] F. Howar, D. Giannakopoulou, and Z. Rakamarić, "Hybrid learning: interface generation through static, dynamic, and symbolic analysis," *Proceedings of the 2013 International Symposium on Software Testing and Analysis - ISSTA 2013*, p. 268, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2483760.2483783

[58] I. Medeiros, N. Neves, and M. Correia, "DEKANT: A Static Analysis Tool that Learns to Detect Web Application Vulnerabilities," *Proceedings of the 14th ACM conference on Computer and communications security CCS 07*, p. 529, 2007. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1315245.1315311

[59] T.-D. B. Le, D. Lo, C. Le Goues, and L. Grunske, "A learning-to-rank based fault localization approach using likely invariants," *Proceedings of the 25th International Symposium on Software Testing and*

Analysis - ISSTA 2016, pp. 177–188, 2016. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2931037.2931049

[60] H. Spieker, A. Gotlieb, D. Marijan, and M. Mossige, "Reinforcement Learning for Automatic Test Case Prioritization and Selection in Continuous Integration," *Proceedings of 26th International Symposium on Software Testing and Analysis (ISSTA'17)*, pp. 12—-22, 2017.

[61] M. Santolucito, "Version Space Learning for Verification on Temporal Differentials," *Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 428–431, 2017. [Online]. Available: http://doi.acm.org/10.1145/3092703.3098238

[62] D. S. Katz, "Understanding Intended Behavior using Models of Low-Level Signals," pp. 424–427.

[63] J. Hotzkow, "Automatically Inferring and Enforcing User Expectations," *Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis - ISSTA 2017*, no. July, pp. 420–423, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3092703.3098236

[64] K. R. Varshney, "Engineering safety in machine learning," *2016 Information Theory and Applications Workshop, ITA 2016*, 2017.

[65] G. I. Webb and F. Petitjean, "A Multiple Test Correction for Streams and Cascades of Statistical Hypothesis Tests," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pp. 1255–1264, 2016. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2939672.2939775

[66] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 1135–1144. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939778

[67] R. Parekh, "Designing AI at Scale to Power Everyday Life," *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17*, pp. 27–27, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3097983.3105815

[68] D. Baylor and E. Breck, "TFX: A TensorFlow-Based Production-Scale Machine Learning Platform," *Kdd*, pp. 1387–1395, 2017.

[69] J. Van Der Herten, I. Couckuyt, D. Deschrijver, P. Demeester, and T. Dhaene, "Adaptive modeling and sampling methodologies for Internet of Things applications," *Proceedings of the 18th Mediterranean Electrotechnical Conference: Intelligent and Efficient Technologies and Services for the Citizen, MELECON 2016*, no. April, pp. 18–20, 2016.

[70] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, and D. Dennison, "Hidden Technical Debt in Machine Learning Systems," *Nips*, pp. 2494–2502, 2015. [Online]. Available: http://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.pdf

[71] F. Yang, A. Ramdas, K. Jamieson, and M. J. Wainwright, "A framework for Multi-A(rmed)/B(andit) testing with online FDR control," no. 3, 2017. [Online]. Available: http://arxiv.org/abs/1706.05378

[72] R. Sen, A. T. Suresh, K. Shanmugam, A. G. Dimakis, and S. Shakkottai, "Model-Powered Conditional Independence Test," no. Nips, pp. 1–11, 2017.

[73] P. Schulam and S. Saria, "Reliable Decision Support using Counterfactual Models," no. Nips, 2017. [Online]. Available: http://arxiv.org/abs/1703.10651

[74] J. Z. Liu and B. Coull, "Robust Hypothesis Test for Nonlinear Effect with Gaussian Processes," *Nips2017*, vol. 2, no. Nips, pp. 1–10, 2017. [Online]. Available: http://arxiv.org/abs/1710.01406

[75] H. C. L. Law, C. Yau, and D. Sejdinovic, "Testing and Learning on Distributions with Symmetric Noise Invariance," no. Mmd, 2017. [Online]. Available: http://arxiv.org/abs/1703.07596

[76] W. Jitkrittum, W. Xu, Z. Szabo, K. Fukumizu, and A. Gretton, "A Linear-Time Kernel Goodness-of-Fit Test," no. Nips, pp. 0–3, 2017. [Online]. Available: http://arxiv.org/abs/1705.07673

[77] Z. Daniel Guo, P. S. Thomas, and E. Brunskill, "Using Options and Covariance Testing for Long Horizon Off-Policy Policy Evaluation," *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, no. Nips, 2017. [Online]. Available: http://papers.nips.cc/paper/6843-using-options-and-covariance-testing-for-long-horizon-off-policy-policy-evaluation.pdf

[78] F. Cecchi and N. Hegde, "Adaptive Active Hypothesis Testing under Limited Information," no. d, pp. 1–9, 2017.

[79] H. Assem, L. Xu, T. S. Buda, and D. O'Sullivan, "Machine learning as a service for enabling Internet of Things and People," *Personal and Ubiquitous Computing*, vol. 20, no. 6, pp. 899–914, 2016.

[80] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *Proceedings of the 26th Symposium on Operating Systems Principles*, ser. SOSP '17. New York, NY, USA: ACM, 2017, pp. 1–18. [Online]. Available: http://doi.acm.org/10.1145/3132747.3132785

[81] Amazon, "Amazon Alexa." [Online]. Available: https://developer.amazon.com/alexa

[82] IBM, "With Watson Program," 2018. [Online]. Available: https://www.ibm.com/watson/with-watson/

[83] R. G. Smith, "On the development of commercial expert systems," *AI Magazine*, vol. 5, no. 3, pp. 61–73, 1984. [Online]. Available: https://www.aaai.org/ojs/index.php/aimagazine/article/view/449

[84] S. Sievers, M. Ortlieb, and M. Helmert, "Efficient Implementation of Pattern Database Heuristics for Classical Planning," *Symposium on Combinatorial Search*, pp. 105–111, 2012.

[85] A. Ramaswamy, B. Monsuez, and A. Tapus, "AI Dimensions in Software Development for Human-Robot Interaction Systems," pp. 128–130, 2014.

[86] D. S. Prerau, "Knowledge Acquisition in the Development of a Large Expert System," *AI Magazine*, vol. 8, no. 2, pp. 43–51, 1987.

[87] G. Peter, "Knowledge-Based Software Engineering Conference," 1992.

[88] M. R. Lowry, "Software Engineering in the Twenty-First Century," *AI Magazine*, vol. 13, no. 3, pp. 71–87, 1992.

[89] S. Giroux and N. Bier, "Cognitive Assistance to Meal Preparation: Design, Implementation, and Assessment in a Living Lab," *2015 AAAI Spring ...*, pp. 01–25, 2015. [Online]. Available: https://www.aaai.org/ocs/index.php/SSS/SSS15/paper/view/10329/10013

[90] M. A. Cohen, F. E. Ritter, and S. R. Haynes, "Applying Software Engineering to Agent Development," pp. 25–44, 2010.

[91] S. Burton, K. Swanson, and L. Leonard, "Quality and Knowledge in Software Engineering," *AI Magazine*, vol. 14, no. 4, pp. 43–50, 1993.

[92] R. Hollander, "Alibaba, Baidu, and Tencent, China's powerhouses, focus on AI to surpass the US - Business Insider," 2017. [Online]. Available: http://www.businessinsider.com/alibaba-baidu-and-tencent-chinas-powerhouses-focus-on-ai-to-surpass-the-us-2017-11

[93] A. Nordrum, "Automatic Speaker Verification Systems Can Be Fooled by Disguising Your Voice - IEEE Spectrum," 2017. [Online]. Available: https://spectrum.ieee.org/tech-talk/telecom/security/automatic-speaker-verification-systems-can-be-fooled-by-disguising-your-voice

[94] S. K. Moore, "DARPA Seeking AI That Learns All the Time - IEEE Spectrum," 2017. [Online]. Available: https://spectrum.ieee.org/cars-that-think/robotics/artificial-intelligence/darpa-seeking-ai-that-can-learn-all-the-time

[95] J. Hsu, "Deep Learning AI for NASA Powers Earth Robots - IEEE Spectrum," 2017. [Online]. Available: https://spectrum.ieee.org/automaton/robotics/artificial-intelligence/ai-startup-neurala-deep-learning-for-nasa-powers-earth-robots

[96] ——, "A New Way to Find Bugs in Self-Driving AI Could Save Lives - IEEE Spectrum," 2017. [Online]. Available: https://spectrum.ieee.org/tech-talk/robotics/artificial-intelligence/better-bug-hunts-in-selfdriving-car-ai-could-save-lives

[97] E. Ackerman, "After Mastering Singapore's Streets, NuTonomy's Robo-taxis Are Poised to Take on New Cities - IEEE Spectrum," 2016. [Online]. Available: https://spectrum.ieee.org/transportation/self-driving/after-mastering-singapores-streets-nutonomys-robotaxis-are-poised-to-take-on-new-cities

[98] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data," no. 2015, pp. 1–16, 2016. [Online]. Available: http://arxiv.org/abs/1610.05755

[99] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial Training Methods for Semi-Supervised Text Classification," pp. 1–11, 2016. [Online]. Available: http://arxiv.org/abs/1605.07725

[100] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial Attacks on Neural Network Policies," 2017. [Online]. Available: http://arxiv.org/abs/1702.02284

[101] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," 2017. [Online]. Available: http://arxiv.org/abs/1706.03741

[102] V. Cheung, J. Schneider, I. Sutskever, and G. Brockman, "Infrastructure for Deep Learning," pp. 1–11, 2016. [Online]. Available: https://openai.com/blog/infrastructure-for-deep-learning/

[103] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete Problems in AI Safety," pp. 1–29, 2016.