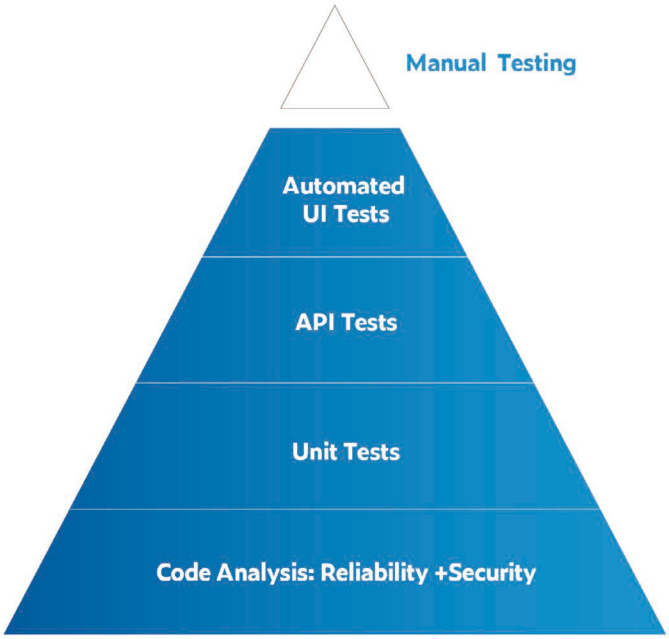
Fördern KI und ML die Übernahme der statischen Analyse?



Die Künstliche Intelligenz bringt Vorteile für jede Ebene des Software-Testings. Alle Bilder © Parasoft

Ziel der statischen Analyse ist das Auffinden von Schwachstellen im Code – das geschieht oft mit branchenüblichen Programmier- und Sicherheitsstandards. Damit die statische Analyse größtmöglichen Nutzen erzielt, muss sie regelmäßig und gründlich erfolgen – einen einfachen Modus gibt es nicht. Allerdings sind Entwickler oft nicht in der Lage, ihre eigenen Codes auf Schwachstellen zu analysieren oder notwendige Korrekturen zur Priorisierung zu ermitteln. Eine wertvolle Hilfe für die Ergebnisse der statischen Analyse ist definitiv deren Automatisierung mit Hilfe von Machine Learning (ML) und KI.

Grundgedanke von Machine Learning

Der Grundgedanke des ML ist, dass KI auf der Grundlage der Beobachtung von Nutzeraktionen lernt. Sie kann darauf trainiert werden, bestimmte Muster zu erkennen und sich darauf anzupassen. In Übereinstimmung mit dieser Methodik werden durch das Identifizieren von Clustern und Verstößen die Ergebnisse der statischen Analyse für die Entwickler verbessert. Während sich die Arten der Analyse und ihre Prioritäten unterscheiden können, sind die Arbeitsweise des SA-Tools und die Anwendung seiner Methodik dieselben. Durch die kontinuierliche Automatisierung ihrer Abläufe können Teams sie besser verwalten und potenzielle Probleme erkennen, bevor diese zu großen Problemen anwachsen.

„Babysteps“

Wegen des Projektumfangs und der Vorgehensweise nehmen viele Entwickler Abstand von der Einführung der statischen Analyse. Oft möchten sie die für sie dringlichsten Themen möglichst alle zugleich angehen und muten sich dabei mehr zu, als sie an diesem Punkt bewältigen können. Dabei lautet das Erfolgsrezept der statischen Analyse, die wichtigsten Probleme in kleinen Schritten anzugehen. Solche ‚Babysteps‘ stellen sicher, dass das Entwicklerteam nicht mit Tausenden von Verstößen überfordert wird. Es geht also darum, Probleme noch vor dem Kompilieren und Ausführen des Codes zu erkennen.

KI kann ihre Stärken auf mehreren Ebenen des Softwaretests ausspielen, etwa bei:

- **UI** (User Interface)-Tests: Verwalten und Warten von flüchtigen automatisierten UI-Tests und Optimieren der Ausführung von manuellen Tests.
- **API** (Application Program Interface)-Tests: Entdecken von API-Nutzungsmustern und automatische Generierung vollständiger Testszenarien.
- **Unit Tests:** Erzielen und Pflegen der Code-Abdeckung, insbesondere bei modifiziertem Code.
- **Code-Analyse** (Zuverlässigkeit und Sicherheit): Bekämpfen von Verstößen in der Codebasis.

Die KI erleichtert nicht nur das Erstellen und Warten automatisierter Tests, sondern kann auch die Testausführung optimieren und die Bereitstellung verwertbarer Ergebnisse maximieren, indem sie die Abläufe auf verschiedene Weise erweitert. Werden Tests automatisiert und Workflows entwickelt, lassen sich mehr Probleme in kürzerer Zeit lösen. Allerdings sollte man das Sortieren den statischen Analysetools überlassen. Durch die Arbeit unter der Aufsicht von statischen Analysetechnologien können Entwickler ihre Fertigkeiten erweitern, um bessere Programmiertechniken zu erlernen und sichereren Code zu schreiben.

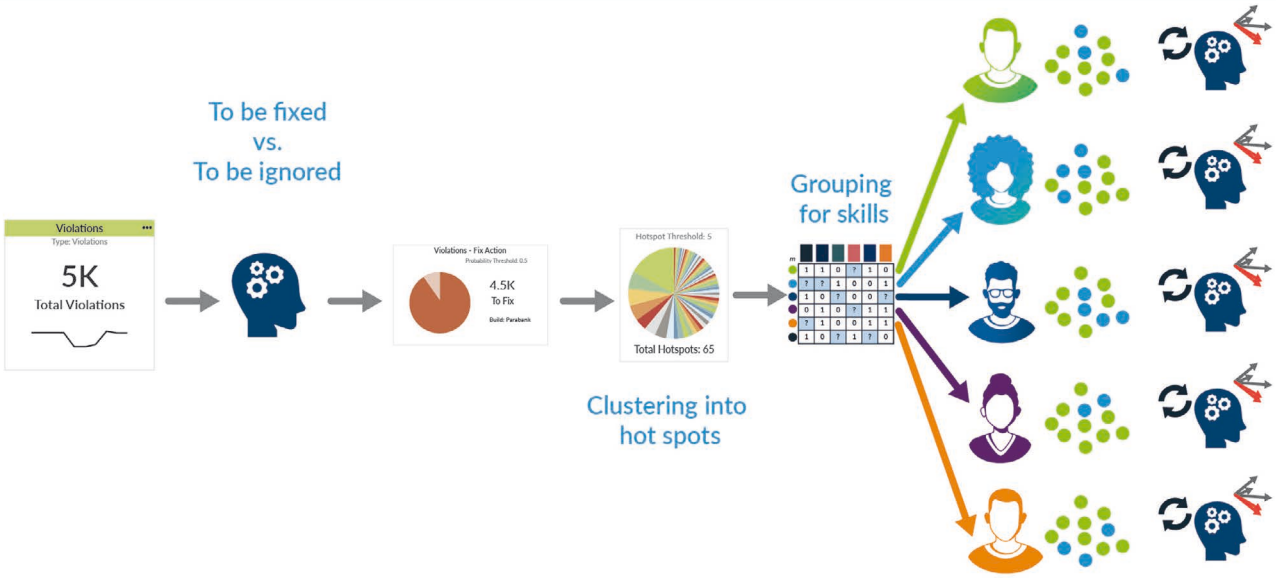
Vorbeugen statt korrigieren

Mithilfe von Techniken zum Vorbeugen und Erkennen kann die Codeanalyse die mit der Qualität des Codes verbundenen Risiken kontrollieren. Auch wenn sie beim Auffinden von Problemen hilft, ist es effizienter, deren Auftreten von vornherein zu verhindern. Genau hier setzt die Automatisierung der statischen Analyse an: Die Integration von KI und ML in die statische Analyse erleichtert den Teams die Übernahme dieser Praxis. Sie erhalten Vorschläge, die Verstöße auf eine Weise zu beheben, die die Effizienz fördert, die Arbeitsabläufe optimiert und die Produktivität und den Erfolg der Entwickler unterstützt.

Clustering-Methodik

Neben dem Clustering von Verstößen auf der Grundlage moderner Klassifizierungsalgorithmen kann das KI-Modell verschiedene neuronale Netze (z. B. code2vec) verwenden, um Code-

Autor:
Igor Kirilenko
VP Development
Parasoft Deutschland GmbH
www.parasoft.com



Mithilfe von AI/ML-Techniken lassen sich die Leistung der Entwickler steigern und zugleich wichtige Code-Probleme beheben

Methoden zu vektorisieren und sie miteinander zu vergleichen, und Verstöße entsprechend der semantischen Bedeutung des sie umgebenden Codes zu gruppieren. So wie die Entwickler zunächst spezifische Verstöße angehen können, erlaubt ihnen das KI-Modell auch, Verletzungen innerhalb ähnlichen Codes anzugehen.

Eine Art „Netflix-Ansatz“

Für eine maximale Produktivität wollen Entwickler oft ähnliche Verstöße gleichzeitig beheben – ein sinnvoller Gedanke, und KI mit ML sollte diese Strategie verbessern. Hier kommt eine Art Netflix-Ansatz ins Spiel: Der Algorithmus einer Streaming-Plattform lernt auf Grundlage von Sehgewohnheiten seiner Benutzer. Das ML für die KI der statischen Analyse funktioniert ähnlich. Auf der Basis früherer Verstöße, die ein Entwickler behoben hat, schlägt ihm das System ähnliche Verstöße vor – es passt sich dem von ihm erstellten Profil an, das auf seiner Historie basiert – analog zur NetFlix-Plattform. Mit diesem Ansatz verbringen Entwickler weniger Zeit mit der Suche nach ähnlichen Verstößen und befassen sich mit solchen, die sie am besten beheben können.

Verfahren des ML für die KI der statischen Analyse

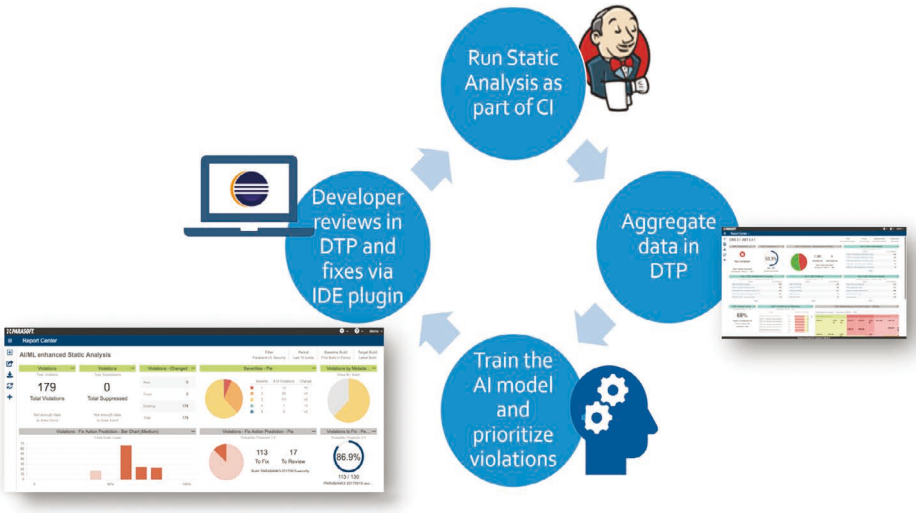
Die Art und Weise, wie KI und ML die statische Analyse beeinflussen, lässt sich wie folgt kategorisieren. Alle diese Methoden arbeiten zusammen, um den Entwicklungsprozess zu verbessern - von der Vereinheitlichung des Quellcodes bis hin zur Identifizierung von Sicherheitsschwachstellen und der Reduzierung von Fehlalarmen.

Identifikation und Rauschunterdrückung

- Klassifiziert wichtige oder kritische Verstöße, um sie früher korrigieren zu können.
- Filtert Rauschen oder unkritische Probleme, um sie später zu beheben.
- Gruppiert Verstöße in semantisch ähnlichen Codes.
- Identifiziert Hotspots oder Ursachen, die mehrere Verstöße auslösen.
- Verringert den Zeitaufwand des Entwicklungsteams für das manuelle Einstufen von Fehlern.

Priorisierung und Gruppierung

- Priorisiert Hotspots nach Anzahl der von ihnen verursachten Verstöße.
- Identifiziert Entwickler, die aufgrund ihrer Fähigkeiten am besten in der Lage sind, Verstöße zu beheben.
- Schlägt den Entwicklern Verstöße auf Basis ihrer Vertrautheit mit dem Code vor.



Die Integration von AI in den Workflow der statischen Analyse bringt mehrere Vorteile.

Fazit

Die technologische Entwicklung und agile Methoden treiben die Softwareentwicklung immer schneller voran. Testansätze müssen mindestens Schritt halten oder sogar einen Schritt voraus sein. Der beste Weg, dies zu erreichen, sind Lösungen, die Automatisierung und maschinelles Lernen für KI einbeziehen. Sie versprechen maximalen Nutzen für die statische Analyse – und Entwickler können dabei noch ihre Fertigkeiten ausbauen. Darum sind Unternehmen und Entwicklungsteams gut beraten, auf Anbieter von innovativen Lösungen mit Erfahrung zu setzen.

Wer schreibt:

Igor Kirilenko zeichnet als VP of Development bei Parasoft für die technische Strategie, Architektur und Entwicklung der Parasoft-Produkte verantwortlich. Er verfügt über mehr als 20 Jahre Erfahrung in der Leitung von Entwicklungsteams, mit einer Spezialisierung auf die Einführung und Förderung der besten agilen Praktiken in Softwareentwicklungsumgebungen. ◄