Test der Coverage von neuronalen Netzen um Fehler im Netzverhalten zu erkennen

Manuel Hirth
Universität Stuttgart
Institut für Automatisierungstechnik
und Softwaresysteme
st174495@stud.uni-stuttgart.de

***Abstract* — Inzwischen gibt es eine Vielzahl von Testverfahren und deren Metriken zur Evaluation von neuronalen Netzen, deren Robustheit und Anfälligkeit für Corner Cases. Ein Ansatz ist das überzeugende und intuitive Kriterium der Neuron Coverage. In Anlehnung an Testverfahren traditioneller Software konnten teils imposante Erfolge verzeichnet werden. Jedoch wird dieser Ansatz durch neue Publikationen, teils grundlegend, in Frage gestellt und zumindest als unvollständig dargestellt. In diesem Paper findet eine Literaturanalyse statt. Somit soll näher auf den Begriff der Neuron Coverage, den zugrunde liegenden Konzepten als auch auf mögliche additive Erweiterungen und Alternativen eingegangen werden. Korrelationen zwischen den Kriterien und Ergebnissen werden betrachtet. Die Beschreibung eines möglichen Usecases, zum Test eines Warnsystems für einen Bagger soll zu einer breiten Diskussion einladen. Der Ausblick in die Zukunft- z.B. GAN-Ansätze- wird versucht. Die Definition von Anforderungen, an ein Testsystem, bleibt schwierig.**

Keywords — Deep Learning, Testverfahren, tiefe künstliche neuronale Netze, Neuronen Abdeckung

I. Einleitung

Die immer weiter voranschreitende Forschung und Entwicklung des maschinellen Lernens (ML), speziell der tiefen künstlichen neuronalen Netze (KNN), erfordert gerade bei sicherheitskritischen Anwendungen robuste Testverfahren. Häufig wird hier das Beispiel des tödlichen Unfalls eines Teslas im Autopilot-Modus mit einem LKW im Juli 2016 genannt. Das nur auf Bilderkennung aus Kamerabildern beruhende System erkannte einen weißen Lastwagen vor dem hellen Horizont nicht und kollidierte mit hoher Geschwindigkeit [1]. Tesla konnte keine Fehler an der Hardware feststellen und die Kollision wurde auf eine Fehlklassifikation der Bilddaten zurückgeführt. Ein weiterer zu bedenkender Umstand ist, dass das System den Fahrer mehrfach aufforderte die Hände ans Steuer zu nehmen [6]. Es sollte zumindest in einer gewissen Übergangsphase ein autonomes System nicht ohne Überwachung laufen, um Probleme mit Corner Cases zum einen zu verhindern und zum anderen weitere Trainingsdaten daraus generieren zu können. Da KNNs Blackbox Systeme sind, bei denen die Entscheidungsfindungen schwer bis unmöglich nachzuvollziehen sind, wird die Erforschung von geeigneten Testverfahren immer bedeutender und ist Gegenstand zahlreicher Studien vergangener Jahre. Besondere Schwierigkeiten bei der Integrität der Entscheidungsfindung bilden Corner Cases, die Wahl geeigneter Metriken zur Beurteilung der Testabdeckung und die Bewertung der Trainings- und Testdatenqualität [1]. Ein Ansatz stellt dabei die Neuron Coverage (NC) dar. In Kapitel II. wird in diese und weitere Grundlagen eingeführt. Es folgt ein Vergleich von Papern in Kapitel III. In Kapitel IV wird der konkrete Anwendungsfall eines Baggerwarnsystems vorgestellt. Schließlich werden in Kapitel V Anforderungen konkretisiert und ein wünschenswertes Szenario erdacht.

II. Grundlagen

Zur Verbesserung des Verständnisses sollen die elementaren Begriffe der Arbeit erläutert werden.

A. Neuronen

Die Neuronen, die ein KNN bilden, sind ein in Anlehnung an das natürliche Vorbild - den Neuronen im Gehirn - entwickeltes mathematisches Modell. Ein Neuron, eines modernen KNNs, besteht hierbei aus gewichteten Eingängen, der Übertragungsfunktion und einer nicht-linearen Aktivierungsfunktion. Darüber hinaus kann ein Bias Wert b aufgeschaltet werden.

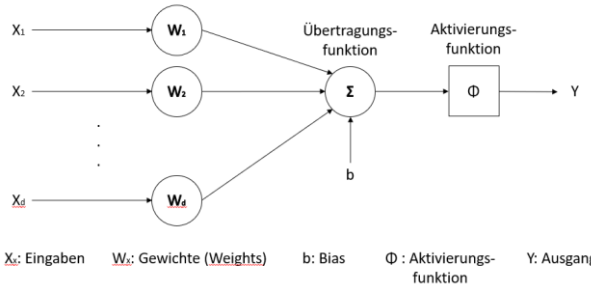

Figure 1: Schema eines künstlichen Neurons

Wie in Fig. 1 zu sehen, werden zunächst die empfangenen Eingänge x mit den Gewichten w multipliziert. So wird der Einfluss der Eingangsvariablen definiert. Nach der anschließenden Summation wird durch die Übertragungsfunktion die Netzeingabe berechnet und anhand einer Aktivierungsfunktion die Ausgabe des Neurons bestimmt. Der Wert y wird über einen oder mehrere Ausgänge zu anderen Neuronen übertragen [7]. Die mathematische Berechnung beruht hierbei auf Vector-/Matrix-/Tensormultiplikationen.
Die einfachste und erste Form eines Neurons ist, das von Frank Rosenblatt veröffentlichte, Perzeptron (1957). Noch heute bildet es die Grundlage für KNNs. Ein Neuron des Perzeptron bildet eine lineare Funktion $y = f(x)$ ab. Somit sind nur lineare Aufgaben realisierbar, dies stellt noch kein mächtiges Werkzeug dar. Es ist wie folgt definiert nach [8]:

$$y = \underline{w}^T * \underline{x} + b$$

$$\underline{x} = [x_1, \dots, x_d] \in \mathbb{R}^d \quad \underline{w} = [w_1, \dots, w_d] \in \mathbb{R}^d \quad b \in \mathbb{R}$$

Der wesentliche Entwicklungsschritt ist das gemeinsame Nutzen von nicht-linearen Aktivierungsfunktionen Φ, sowie die Bildung tiefer KNNs durch Verwendung mehrerer Schichten aus Neuronen. Dabei ist die Kombination beider Schritte von entscheidender Bedeutung, ohne diese wären nur affine Funktionen der Netzwerkeingabe oder monoton steigende nichtlineare Funktionen abbildbar. Mithilfe dieses Konzepts werden tiefe KNNs zu Multilayer Perzeptrons aus Fullyconnected Layers erweitert. Bei diesen ist jedes vorrausgehende mit jedem nachfolgenden Neuron verbunden. Diese

Art wird in Feedforeward (FF) Netzwerken, d.h. Informationen fließen ohne Rückkopplungsverbindungen von Schicht zu Schicht, eingesetzt. Die Funktion eines einzelnen Neurons erweitert sich zu [nach 8]:

$$y = \Phi(\underline{w}^T * \underline{x} + b)$$

Recurrent Neural Networks (RNNs) erweitern das Konzept um Feedback Pfade mit trainierbaren Gewichten. Die Neuronen besitzen einen Speicher, der eine Zeitrekursion ermöglicht. Diese wird in RNNs entfaltet (unfolded Graph) und das RNN anschließend wie ein FF Netzwerk behandelt.
Eine Erweiterung bilden die Long Short-Term Memory (LSTM) Zellen aus fünf Neuronen. Diese speichern, wie ein rekurrentes Neuron, den Zustand s(n) zum Zeitpunkt n. Hier werden drei multiplikative Gates zum Steuern des Schreib-/Rücksetz-/Lesevorganges des Speichers ergänzt [8,11].
Bei den häufig, in der Bildverarbeitung, verwendeten Convolutional Neural Networks (CNN) wird ein modifiziertes Konzept der Neuronen verwendet. Hierbei wird in einigen oder allen Layern die Konvolution/Faltung, statt Matrixmultiplikationen genutzt [10]. Die Neuronen finden sich hier in Kernels, für gewöhnlich der Größe 3x3 oder 5x5, wieder. Daraus folgt eine mehrdimensionale Ausgabe, selber oder verminderter Größe pro Layer.
Die Gewichte der Kernels werden darauf trainiert, bestimmte lokale Muster zu detektieren. Der Prozess ist an das biologische Vorbild des rezeptiven Feldes angelehnt, in dem ein bestimmter (Seh-)Bereich an ein einzelnes Neuron weitergeleitet wird [9,8].

B. Corner Cases

Für gewöhnlich stellen beim Testen von ML-Systemen den Trainingsdaten ähnliche Situationen keine Probleme dar. Problematisch sind Abweichungen und Ausnahmen, die Corner Cases [5]. Ein KNN zeigt hier fehlerhaftes oder nicht erwartetes Verhalten, das unerwünschte oder sogar tödliche Folgen haben kann. Beim Testen der KNNs ist Wissen über solche Corner Cases nachrangig zu betrachten, nötig sind Verständnis der Architektur und Funktionsweise der KNNs, um so spezifische Schwächen zu detektieren [5].

C. Neuronen Abdeckung/Neuron Coverage

Die Metrik NC ist einer der meistverfolgten Ansätze der letzten Jahre zur Erhöhung der Testqualität von KNNs verschiedener Strukturen vgl. [1,4,12,14].
Sie ist von dem traditionellen Testverfahren herkömmlicher Software, wie der Verzweigungs-/Codeabdeckung, inspiriert. Da die Qualität der KNNs jedoch nur in geringem Maße vom programmierten Code abhängt, sind diese Verfahren nicht direkt anwendbar. Bei der NC wird die Intuition der Codeabdeckung auf die Besonderheiten der KNNs angewendet. Statt dem aktivierten Code wird der Anteil der aktivierten Neuronen erfasst [1]. Als Analogie wird die If-Verzweigung genannt, bei der ein fehlerhafter Codeanteil nicht entdeckt werden kann, wenn er in einem Zweig liegt, der nie getestet wurde. Dies bietet zwar Inspiration und Anschauung - es bleibt jedoch zu bedenken, dass die Effektivität bei komplexen Strukturen wie KNNs wage bleibt und die Einzigartigkeit nicht ausreichend berücksichtigt wird. Die Codeabdeckung hat eine belastbare Aussagekraft, da bei vollständiger Abdeckung tatsächlich der gesamte Code getestet wurde. Dies ist für die NC nicht der Fall, die Aussagekraft als Testmetrik bleibt ein Forschungsthema.

Pei et al. [17] definiert die Neuron Coverage zu:

$$NCov(T, \boldsymbol{x}) = \frac{|n|\forall n \in T, out(n,x) > t|}{|N|}$$

$$N = \{n_1, n_2, \dots\}: all\ neurons\ \ T = \{x_1, x_2, \dots\}: test\ inputs$$

Hierbei stellt out(n,x) eine Funktion dar, die den Ausgabewert eines Neurons wiederspiegelt und zwischen 0 und 1 liegt. Der Parameter t legt den Schwellwert fest, ab dem das Neuron als aktiviert angesehen wird. Bei einem trainierten KNN bestehen die Veränderlichen nur in T und t.
Harel-Canada et al. [12] veranschaulicht das Konzept:

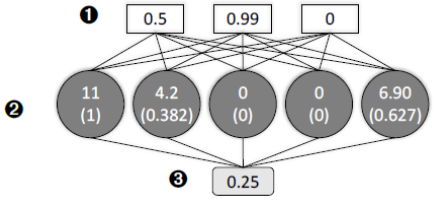

Figure 2: Veranschaulichung der NC (Abbildung aus [12])

Das KNN besitzt drei Eingänge (1), einen Hidden Layer mit 5 Neuronen (2) und eine Ausgangsschicht mit 1 Neuron (3). Dabei ist auch der Wert von out(n,x), der mit t verglichen wird, zu erkennen (bei 2 in Klammer). Verschiedene Veröffentlichungen arbeiten mit unterschiedlichen Schwellwerten vgl. [1,2,17]. Der optimale Schwellwert und ob dieser existiert, ist nicht abschließend geklärt. Beim Vergleich mit dem Schwellwert, zur Berechnung der NC, muss der Netzwerk- und damit der Neuronen Typ berücksichtigt werden. Bei Fullyconnected Layern ist der Vergleich direkt mit dem Ausgabewert jedes Neurons möglich. Bei CNNs liegt eine mehrdimensionale Ausgabe von Layer zu Layer vor, daher wird ein Durchschnitt der Werte berechnet um eine einzige Zahl zu erhalten und diese verglichen. Bei RNNs/LSTMs wird der unfolded Graph betrachtet, in diesem wird jedes entstandene Neuron separat betrachtet und verglichen.
Eine Vorstellung des Vorteils der Erhöhung der NC ist die Idee, dass jedes Neuron unabhängig etwas Nützliches extrahiert. Aus der Definitionsformel lässt sich folgern, dass ein geeigneter Testdatensatz möglichst die volle Entscheidungsvielfallt und damit eine Neuronen Aktivierung aller Neuronen des KNNs abdecken soll. Wenn jedes Neuron von zumindest einem Fall/Bild aktiviert wird, ist es wahrscheinlicher ein fehlerhaftes oder unerwartetes Verhalten in der vorhandenen und bisher nicht aktivierten Netzstruktur und bei den meist nicht aktivierten Neuronen aufzudecken. Diese Fälle sind nicht mit einer bloßen Erhöhung der Datenmenge des Testdatensatzes zu erreichen, sie erfordern speziell erzeugte Testcases. Diese sollen das KNN auf unvorhersehbare Situationen der Realität vorbereiten.
Eine andere Erklärung als die Analogie zu Codeabdeckung könnte die in KNN forcierte sparsity sein. Aufgrund der häufigen 0-Einträge liegt es nahe, dass die NC bei den trainierten KNNs eher gering sein wird.

D. Weitere Ansätze zur Verbesserung der Testqualität

Zur Erhöhung der Testqualität sollten weitere Möglichkeiten in Betracht gezogen werden. Die klassischen Ansätze der Explainable AI, sind Gradient-Weighted Class Activation Mapping (Grad-CAM) und t-Distributed Stochastic Neighbor Embedding (t-SNE), die auf Visualisierung beruhen. Grad-

CAM markiert bedeutende Bereiche oder Pixel des Eingabebildes für eine bestimmte Zielklasse und ist anwendbar auf alle KNNs ohne Retraining. t-SNE visualisiert die Ähnlichkeit hochdimensionaler Daten in einem niederdimensionalen Raum, z.B. dreidimensional und damit anschaulich [8]. Diese können zwar einen Einblick in die Entscheidungsfindung bieten und durchaus zur Erhöhung der Qualität beitragen, jedoch nicht in automatisierten Testsuiten Verwendung finden.
Eine weitere klassische Herangehensweise sind Adversarial Attacks/Examples. Hier können schon geringfügige Eingangsstörungen zu einem Zustand führen, der nicht abgedeckt ist, daraus folgen unvorhersehbare Ergebnisse [20]. Dies ist ein offenes Forschungsthema, bei dem erste Ansätze existieren, aber noch keine endgültigen Lösungen. Arbeiten, wie [21], beschäftigen sich mit der Erhöhung der Entscheidungsqualität durch diesen Ansatz. Auch im Zusammenhang mit NC werden Adversrial Inputs gesucht. Der klassische Ansatz wird als zu unsystematisch bezeichnet [1].
Harel-Canada et al. [12] empfiehlt drei weitere Kriterien. Zum Ersten Fehlererkennung, diese wird definiert zu:

$$ASR(T) = 1 - pert\_acc$$

Dabei gibt ASR die Attack Success Rate an. Dies entspricht zugleich der Detection Default Rate (DDR). T ist die Menge der Testeingaben und pert_acc die Klassifikationsgenauigkeit. Eine solche Metrik ermöglicht Korrelationsanalysen mit anderen Methoden und Metriken. Die DDR stellt eine robuste Metrik dar. Sie gibt, unter der Voraussetzung sinnvoller Testfälle, einen realen Einblick in die Qualität des KNNs.
Zweitens die Natürlichkeit der Test-Inputs. Diese Restriktion beinhalten mehrere Arbeiten. Besonders bei sicherheitskritischen Systemen ist es sinnvoll auf realitätsnahe Testfälle, der Praxis, zu achten. Autos auf dem Kopf machen keinen Sinn. Es existieren zwei aus der GAN-Forschung bekannte Möglichkeiten die Natürlichkeit zu beurteilen. Der Inception Score (IS) beruht auf der Eindeutigkeit der Klassifizierung und der Tatsache, dass alle vorkommenden Klassen gleichmäßig repräsentiert sind. Die Frèchet Inception Distance (FID) ermöglicht eine Beurteilung der Ähnlichkeit zwischen Datensätzen. Oft hilft die Bewertung mit menschlicher Intuition, z.B. bei Bildern. Die FID korreliert mit dieser [12].
Die Unparteilichkeit der Ausgabe ist der dritte Punkt. Diese ergibt sich durch den Bias eines Models. Der Bias gibt an, wie gut ein Modell und dessen Vorhersagen zu einem Datensatz passen. In der Praxis bietet sich häufig die Messung des Trainingsfehlers mit Hilfe des mittleren quadratischen Fehlers (MSE) an. Auf den Fall der KNNs angewendet bedeutet das, dass ein Testdatensatz zu vielfältigem Ausgabeverhalten führen soll und es von Bedeutung ist, dass bestimmte Klassen nicht bevorzugt werden. Voraussetzung ist ein Datensatz mit gleichmäßig repräsentierten Klassen, um sicherzustellen, dass der Fehler im Modell liegt. Kommt das Modell bei einem solchen Datensatz zu häufig zu wenigen bestimmten Klassen, ist die Unparteilichkeit der Ausgabe verletzt [12].

III. Vergleich der Testansätze

Hier werden Arbeiten zur NC vorgestellt und analysiert. Ein Kontext der Literatur soll gefunden werden.

A. DeepXplore

DeepXplore [17] legt den Grundstein aller nachfolgenden Arbeiten zum Thema NC. Mit dem Ziel mehr Interna des KNNs zu nutzen, führte es die in Kapitel IIC beschriebene NC, sowie einen Algorithmus zur Erhöhung dieser ein. Zunächst wird ein breiter Einblick in die Grundlagen von KNNs und die Konzepte gegeben. Es wird gezeigt, dass ein kleiner zufällig ausgewählter Datensatz 100% der Codeabdeckung erzeugt, während die NC bei lediglich 34% liegt. Damit soll verdeutlicht werden, dass kein Nutzen in der Verwendung traditioneller Software-Testverfahren besteht. Je zwei für den gleichen Anwendungsfall trainierte KNNs werden verglichen. Verwendet wurden bekannte Datensätze wie MNIST und Netzstrukturen wie LeNet. Erzeugte Bilder sollen sowohl die NC erhöhen als auch zu unterschiedlichen Klassifikationen der beiden KNNs führen. Der Algorithmus beruht auf Optimierungsverfahren mit Nebenbedingungen und enthält 7086 Zeilen Python Code. Es handelt sich um Whitebox Testing, ist also besonders gut geeignet Fehlerursachen zu lokalisieren/identifizieren. Das Ergebnis der Studie zeigt die Effektivität zur Ergründung fehlerhafter Strukturen und Corner Cases.

B. DeepTest

DeepTest [1] baut auf der Idee der NC auf. Es kommen alle in Kapitel IIA beschriebenen Neuronen-/KNN-Typen zum Einsatz. Auf Grundlage von drei trainierten KNNs der Udacity Challenge für autonome Fahrzeuge werden synthetische Bilder zum Testen generiert und die Ergebnisse anhand des ausgegebenen Lenkwinkels verifiziert. Die Arbeit wird hier deutlich konkreter. So sollen Transformationen angewendet werden, die natürliche Umgebungsveränderungen, wie Sonnenlicht, Regen, Nebel etc. abbilden. Es wird sowohl Blurring als auch das Hinzufügen dieser Effekte angewendet. Die Transformationen werden in den Papern als metamorphic, -deutsch metamorph - bezeichnet, das bedeutet sie ändern den zu erwartenden Lenkwinkel im Vergleich zum Originalbild nicht. Dies erleichtert die Verifikation. Darüber hinaus werden konkret die affinen Transformationen, die in Fig. 3 zusammengefasst sind, eingesetzt.

Affine Transform	Example	Transformation Matrix	Parameters
Translation		$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$	t_x: displacement along x axis t_y: displacement along y axis
Scale		$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \end{bmatrix}$	s_x: scale factor along x axis s_y: scale factor along y axis
Shear		$\begin{bmatrix} 1 & s_x & 0 \\ s_y & 1 & 0 \end{bmatrix}$	s_x: shear factor along x axis s_y: shear factor along y axis
Rotation		$\begin{bmatrix} \cos q & -\sin q & 0 \\ \sin q & \cos q & 0 \end{bmatrix}$	q: the angle of rotation

Figure 3: Affine Transformationen DeepTest (Abbildung aus [1])

Die Transformationen werden auch in Bildbearbeitungssoftware realer Kameras verwendet. Sie werden für gewöhnlich über eine 2x3 Matrix repräsentiert und als Filter oder Kernels bezeichnet. Hier werden Translation eine Verschiebung, Scale eine Bildgrößenänderung, Shear eine Verzerrung und Rotation eine Drehung des Bilds, verwendet. Die mathematische Struktur der jeweiligen Matrizen kann Fig. 3 entnommen werden. Die Bildtransformationen werden über eine Faltung des Bildes, Pixel für Pixel, mit dem Filter in der Spatial Domain realisiert. DeepTest kann zuvor nicht-aktivierte Neuronen erkennen und automatisch detektieren, ob das KNN ein fehlerhaftes Verhalten erzeugt. Es wird ein Suchalgorithmus verwendet, der wiederholt Transformationen auf ein einzelnes Bild anwendet, um die Zahl der aktivierten Neuronen zu

erhöhen. Da es sich bei der Faltung um einen linearen Operator handelt, ist es möglich die Faltung jedes Bildes mit einzelnen Filtern durchzuführen oder zunächst die Filter zu falten, um eine gemeinsame Transformation des Bildes zu berechnen. Von hoher Bedeutung ist die Verfolgung des Effekts der Transformationskombinationen. Erst bei einem möglichst hohen Wert wird zum nächsten Testbild übergegangen. Auch aus den, im Verlauf des Suchalgorithmus gefundenen, besonders effektiven Transformationskombinationen, soll gelernt werden. Diese Art der Software-Tests wird als Greybox-Testing bezeichnet, eine Kombination aus dem bekannten White- und Blackbox-Testing bei dem im Test, Interna der Software nicht betrachtet werden. Für diese Art des Testens ist kein Zugriff auf den Quellcode bzw. die Netzstruktur von Nöten, eine Grundlegende Idee von Algorithmen, Architekturen, internen Zuständen ist ausreichend. DeepTest sowie die verwendeten KNNs sind auf GitHub zur Verfügung gestellt.
Als große Herausforderung wird die manuelle detaillierte Spezifikation solcher Suchalgorithmen gesehen, hier kommen wieder die metamorphen Transformationen ins Spiel, die die Interpretation des Lenkwinkels verbessern. Da es keinen eindeutig richtigen Lenkwinkel gibt, werden die Anforderungen an die Übereinstimmung etwas gelockert. Ein solcher Prozess wird als Erzeugung eines Testorakels bezeichnet. Im Zusammenhang der Testung von KNNs wird auch häufig die Ground Truth direkt als Testorakel angewendet.
Die Ergebnisse laut Studie lassen sich wie folgt zusammenfassen. Systematische Tests anhand der NC sind möglich, da diese mit der Input-Output Diversität korreliert sind. Unterschiedliche Transformationen und -kombinationen aktivieren spezifische Neuronen Sets. Eine systematische Kombination der Transformationen kann die NC bis zu 100% steigern. Mit dem Testsystem lassen sich mehr als 1000 fehlerhafte Strukturen selbst bei diesen Modellen finden. Retraining mit einem mit DeepTest erzeugten Datensatz kann die Genauigkeit der KNNs signifikant verbessern, im besten untersuchten Fall bis zu 46%. Die Autoren weisen darauf hin, dass die erzeugten Transformationen keinen Anspruch auf Vollständigkeit haben und es in der Realität eine Vielzahl von Situationen geben kann, die nicht vorhersehbar sind.

C. Erweiterung von DeepTest

In [22] wird DeepTest weiter untersucht und verwendet, um den Lenkwinkel zu bestimmen. Die Autoren kritisieren, dass wenn auch möglicherweise unbeabsichtigt, die Suchmethode auf Basis des Gradient Descent eine hohe NC auf Kosten übertransformierter Bilder liefert. Sie liefern Beispiele, bei denen das wiederholte Transformieren zu Verzerrungen bis hin zur Unkenntlichkeit führt. Die metamorphe Struktur der Transformationen geht verloren und es werden Zweifel erhoben welchen nutzen solche Bilder in den Experimenten bringen. Um dies zu vermeiden, wurden nur noch Skalierung, Helligkeit, Kontrast und Unschärfe als Transformationen verwendet. Der Einfluss wurde untersucht. Das Ergebnis ist, dass eine Änderung des Kontrasts in Kombination mit dem vertikalen Drehen oder einer Änderung der Helligkeit
die größte Steigerung der NC bewirken.

D. Aussagekraft der NC

Die vielversprechenden Ergebnisse der NC lösten weitere Forschung auf dem Gebiet aus. So stellen [12,23,24,25] die Aussagekraft der NC in Frage.
[12] hatte zum Ziel einen verbesserten Algorithmus zu entwickeln, der die in Kapitel IID erläuterten Kriterien der Default Detection, Natürlichkeit und Unparteilichkeit mit einbezieht. Entgegen den Erwartungen ergab sich der Schluss, dass eine bloße Erhöhung der NC diese Metriken konterkariert. Darüber hinaus wurden DeepXplore und DeepTest einer Analyse unterzogen. Diese kam zum Ergebnis, dass bei DeepXplore nicht eine einzige signifikante Korrelation zwischen der Erhöhung der NC und den neuen Kriterien besteht. Bei DeepTest ließ sich eine solche Korrelation lediglich bei der Unparteilichkeit finden. Ein Ansatz zur Erklärung besteht darin, dass der Basisgedanke der NC, dass jedes Neuron individuell bestimmte Merkmale extrahiert, die Komplexität der Zusammenarbeit mit anderen Neuronen nicht ausreichend berücksichtigt. Es werden nachvollziehbare Zweifel erhoben, ob ein Neuron die richtige semantische Einheit zum Verständnis von KNNs und für die Aussagekraft einer Testmetrik ist.
[23] nutzt DeepXplore, um die Aussagekraft der NC bei LeNet KNNs zu untersuchen. Es soll ein neues Coverage Kriterium gesucht werden, das sich noch stärker an dem Testen herkömmlicher Software orientiert und wie die Code Coverage möglichst alle Teile testet. So soll in diesem der Effekt der Neuronen eines Layers untereinander und zum nächsten Layer Berücksichtigung finden. Hierzu wird je ein Triple, dass die verbundenen Neuronen enthält untersucht. Das gezeigte Ergebnis illustriert, dass sich die NC mit 550 Testinputs bei allen LeNets auf über 98% erhöht. Das neue Kriterium jedoch nur zu maximal 11,6% erfüllt wird. Die Autoren weisen darauf hin, dass die Skalierbarkeit auf große reale KNNs weiterer Untersuchung bedarf.
[24] beschäftigt sich mit der Aussagekraft von Kriterien wie der NC. Dazu wurden 100 State-of-the-Art KNNs untersucht. Die Ergebnisse legen nahe, dass alle bisherigen Testkriterien, wie die NC, keine signifikante Korrelation mit einer Verbesserung der Robustheit von KNNs, in sicherheitskritischen Anwendungen, aufweisen. Die Autoren verweisen darauf, dass zukünftige Arbeiten zur Erforschung von Kriterien, diese Korrelation erreichen sollten und die Ergebnisse der Studie als Benchmark zur Evaluation verwendbar sind.
[25] geht noch einen Schritt weiter und stellt die These in den Raum, dass bisherige Kriterien wie die NC irreführend sein könnten. Die Arbeit konzentriert sich darauf die Aussagekraft bisheriger Kriterien zu widerlegen. Es wird grundsätzlich die Frage gestellt, wie ein nützliches Kriterium aussehen könnte. Nach Meinung der Autoren sollte ein solches auf einem grundlegenden tiefen Verständnis der Zusammenhänge zwischen falsch klassifizierten realen Inputs, Netzwerkstrukturen und der Struktur von Adversarial Examples beruhen.

E. Fazit des Analyse

DeepXplore erläutert die theoretischen Grundlagen ausführlich. Es werden positive Ergebnisse beschrieben. Es ist der erste systematische Ansatz, Fehler nicht nur zu finden, sondern auch durch Nachvollziehbarkeit zu lösen. DeepTest erweitert die Ideen von DeepXplore und versucht durch Erzeugung realistischer Transformationen die Tests zu systematisieren. Es lässt sich direkt anwenden und benötigt keine detaillierten Spezifikationen, wird jedoch von den Autoren selbst nur als erster Schritt auf dem Weg hin zu robusten KNNs betrachtet. [22] weist darauf hin, dass eine blinde Verwendung von DeepTest, wie auch von anderen Tools nicht zu

empfehlen ist. Sinnvolle Anwendungsszenarien werden eingegrenzt. Dies geschieht auf Kosten der Automatisierung und direkten Verwendbarkeit von DeepTest.
Die unterschiedlichen Ergebnisse zur Nützlichkeit der NC als Metrik lassen sich mit den unterschiedlichen Bewertungskriterien erklären, die der NC nicht einen prinzipiellen Nutzen absprechen, sondern einen ganzheitlicheren Ansatz wie in [23] anstreben. Auch DeepXplore und DeepTest finden eine hohe Anzahl von fehlerhaften Ausgaben und Strukturen. Andere Ansätze sollten ebenfalls in Betracht gezogen werden.

IV. USECASE

Aufgrund des von der Politik angestrebten Baubooms wird sich der Fachkräftemangel im Bausektor weiter verschärfen. Um die Folgen einzudämmen, wird das Thema der autonomen Baustellenfahrzeuge größere Bedeutung bekommen.
Auf solchen Baustellen werden Bagger benötigt, daher könnten autonome Bagger ein großes Marktpotenzial besitzen. Sie können auch im Bergbau und bei sonstiger Ressourcenförderung wie Goldabbau, da finanzstarke Sektoren, Absatz finden. Ein Vorteil wäre, dass Aufgaben, die ein Risiko für den Baggerführer darstellen, wie Arbeiten an Abhängen, nur noch den Bagger und nicht das Leben des Fahrers gefährden.
Um eine schnelle Platzierung am Markt, mit einem System, das eine gewisse Marktreife besitzt zu gewährleisten, soll der verkürzte Weg über ein Assistenzsystem zur Bedienung der Baggerschaufel gegangen werden. So kann sichergestellt werden, dass die Entwicklung nicht zu lange dauert. Es kann bereits Absatz erzielt werden und die Nachfrage sowie Kundenwünsche können evaluiert werden.
Mithilfe des Assistenzsystems ist der Ausschluss von zwei Fällen von Bedeutung. Es dürfen keine Menschen verletzt und keine Infrastruktur wie Gas- oder Stromleitungen beschädigt werden. Die vorläufige Begrenzung auf ein solches System sichert eine schnellere Realisierbarkeit.
Da Bagger eine teure und langfristige Anschaffung sind, ist hier die Verwendung vertrauenswürdiger und kostspieliger Sensoren, wie LiDAR, als Ergänzung zu Radar und Kameras denkbar. Sowie Ultraschall zur Detektion von Leitungen. Dies würde redundante Systeme, die von KNNs bewertet werden ermöglichen und den Sicherheitsanforderungen weiter Rechnung tragen. Für diese KNNs benötigt es ein robustes und standardisiertes Testverfahren.

A. Marktanalyse

Das Thema gewinnt in den letzten 2 Jahre stetig an Bedeutung. Junge und etablierte Unternehmen wie Baidu und CAT arbeiten daran, erste Piloten zu entwickeln. So stellte Baidu 2021 einen fahrerlosen Bagger vor. Der Hersteller Sandvik baut seit ca. 20 Jahren autonome Bergbaumaschinen, die Daten bezieht der Bagger, da unter Tage kein GPS verfügbar ist, aus Lasern, Karten und einprogrammierten Wegen. Dieser Ansatz zeigt großen Erfolg wie der Test in einem Glaslabyrinth 2018 und über 2 Millionen Betriebsstunden zeigen. Lernende Algorithmen und KNNs kommen nicht zum Einsatz, dies entspricht nicht der gewünschten Anwendung. Erste Schritte der Digitalisierung, wie die Ausstattung mit Touchscreen und Joystick, zur Steuerung, sind Standard [3,18].
Baidu veröffentlichte [13] eine der seltenen KNN-basierten Arbeiten. Das auf Bildverarbeitung setzende System verwendet einen speziell erstellten Datensatz. Es werden Object Detection, Bewegungs- und Positionsschätzung evaluiert. Die Überlegenheit, der KNNs, gegenüber bisherigen Ansätzen mit Klassifikatoren, wie z.B, der Support Vector Machine (SVM), wird gezeigt. Der Prototyp des Baidu-Baggers beruht auf der Arbeit. Eine Produktionsline besteht bisher nicht.

V. ANFORDERUNGSANALYSE

In diesem Kapitel sollen mögliche Anforderungen für das Testen von KNNs zur Sensordatenauswertung der Baggerschaufel vertieft werden. Die Sicherheit muss zu einem hohen Maß gewährleistet sein. Dies ist nötig, um eine marktreife Anwendung ins Auge fassen zu können. Neben wünschenswerten Faktoren, die weitere Forschung bedürfen, sollen speziell die Ergebnisse der Arbeiten [12,23,24,25,26] und Lösungsansätze aus [2,12,23] miteinbezogen werden.

A. Anforderungen

Da das zu testende Assistenzsystem verschiedene Sensordaten verarbeitet, findet darin eine Sensordatenfusion oder eine Kombination der Klassifizierungsergebnisse statt. Dies erhöht die Sicherheit, könnte sich im Test aber als Schwierigkeit herausstellen. Die Umsetzbarkeit muss geprüft werden. Automatisierte Tests, bei denen nur ein gewisser Datensatz vorgegeben wird, wären wünschenswert. Bei neuartigen Systemen könnte Datenmangel herrschen. Eine robuste Testmetrik zur Bewertung der KNNs sollte angestrebt werden.

B. Forschungsfragen und Resultate

FF1: Existieren Arbeiten zur Sensordatenfusion oder zur Verwendung von KNNs mit LiDAR-/Ultraschallsensoren?
[27] untersucht den Einsatz von KNNs an realen Sensoren. Die Verwendung der Bildgebung per LiDAR bringt Vorteile, da eine exakte Entfernungs- und Geschwindigkeitsbestimmung möglich ist. In der Arbeit wurde ein KNN trainiert, um die Gravitationsrichtung aus den Sensordaten abzuleiten. Die Ergebnisse zeigten LiDAR-Sensoren sind Kameras überlegen und die Daten mit KNNs auswertbar. [28] beschäftigt sich mit dem Trainieren von KNNs zur Auswertung von Ultraschallsensoren. Die Ergebnisse zeigen eine Klassifikationsgenauigkeit von über 90% und empfehlen die Verwendung von CNNs. Zur Sensordatenfusion existieren Arbeiten wie [29], die diese mit CNNs umsetzen. Es wird die Überlegenheit einer CNN basierten Fusion gegenüber anderen Methoden deutlich herausgestellt.
E1: KNNs zur Auswertung von LiDAR- und Ultraschallsensoren existieren, auch die Sensordatenfusion lässt sich im KNN implementieren. Dies ist Grundvoraussetzung, um sie in den KNN-Tests mit einzubeziehen.
FF2: Wie können automatisierte Testsysteme für KNNs gebildet werden? Was könnte nötig sein, zur Realisierung?
Die meisten Arbeiten zum Thema basieren auf den Konzepten von DeepXplore und DeepTest oder lassen sich von den enthaltenen Ideen inspirieren. So werden die Konzepte der metamorphen Transformationen sowie die Suche nach geeigneten Testorakeln häufig aufgegriffen.
Neuste Publikationen wie [30, 31] beschäftigen sich damit, die benötigten Testdaten mit weiteren generativen KNNs wie Variational AutoEncoders (VAE) oder Generative Adversarial Networks (GAN) zu erzeugen. Diese können perfekt realistische, aber synthetische Daten generieren.
Das fundamental Neue an diesem Ansatz ist, dass generative Modelle nicht die Transformationen, wie in Kapitel IIIB be-

schrieben, beinhalten. Statt einen Datensatz mit Veränderungen zu erweitern, werden neue möglichst natürliche Daten erzeugt, die die Varianz der Realität abbilden sollen. So werden nicht nur einzelne Pixel der Bilder verändert. Teure, gelabelte Datensätze werden nur noch zum Training der generativen Modelle benötigt. Es kann derselbe Datensatz wie zum Training des zu testenden KNNs verwendet werden, ohne die nötige Varianz zu konterkarieren. Conditional GANs (cGANs) ermöglichen eine Beeinflussung der Erzeugung. Dies geschieht, indem dem GAN ein Bild als Seed-Input übergeben wird und dieses anhand der gelernten Features ein neues Bild erzeugt. Dieser Prozess wird als Image-to-Image-Translation bezeichnet [8,31]. Ein bedeutendes Ergebnis der Forschung zur Testung mit GANs ist, dass gezeigt werden kann, dass Fehler detektiert werden können, die von Algorithmen wie DeepTest nicht aufgedeckt werden [31,32]. Dies wird auf die Erzeugung der Bilder mit High-Level-Features zurückgeführt. Die Veränderung von High-Level-Features wie Position, Farbe und Struktur der Objekte birgt Vorteile gegenüber Low-Level-Features wie den individuellen Pixelwerten. Die Autoren verweisen darauf, nicht den Anspruch erfüllen zu können, alle Fehler aufzuspüren. Es soll ein Puzzlestück auf dem Weg zu ausreichender Testgenauigkeit sein.
[30] verfolgt einen ähnlichen Ansatz unter Zuhilfenahme von VAEs. Da diese GANs und vor allem cGANs in der Regel unterlegen sind, soll hier nur auf einen Punkt eingegangen werden. Die Arbeit bemängelt, dass bisherige Ansätze zu wenig auf valide Testdaten achten. Dies reduziert die Aussagekraft der Ergebnisse. Als Invalid-Inputs werden solche Daten bezeichnet, die entweder nicht mehr die unterliegende Struktur der Aufgabe erfüllen oder die metamorphen Transformationen, also die Transformationen, die nicht mehr das bisherige Label erfüllen, verletzen. Die Invalid-Inputs bergen drei Probleme. Sie erhöhen den Aufwand ohne entsprechenden Nutzen. Die Entwickler können falsche Schlüsse aus den Ergebnissen ziehen. Sowie die Erhöhung z.B. der NC bringt keine Verbesserung bei realen Corner Cases. So kann der Einsatz von VAEs die Erzeugung unbrauchbarer Testfälle verhindern. Die Arbeit beschäftigt sich unter anderem explizit mit der Testcase-Erzeugung von DeepTest und DeepXplore. Sie stellt in mit diesen Algorithmen erzeugten Testdatensätzen ebenfalls eine im Verhältnis zur Gesamtzahl der Bilder hohe Anzahl von Invalid-Inputs fest.
E2: Automatische Tests basierend auf GANs sollten bei der Entwicklung von KNN-Tests der nächsten Generation als Teil der Lösung miteinbezogen werden.
FF3: Wie können geeignete Metriken zur Quantifizierung der Ergebnisfindung von KNNs aussehen?
[2] enthält eine Ergänzung zur Betrachtung einzelner Neuronen über die NC. Die Wirkung auf Layerebene soll mit der Top-k Neuron Coverage miteinbezogen werden. Die k aktiviertesten Neuronen eines Layers werden ausgewertet. Der Gedanke dahinter ist, dass die Neuronen desselben Layers häufig ähnliche Funktionalitäten erfüllen. Dies dient als Indikator um die Grundfunktionalität eines KNNs zu charakterisieren. Um ein KNN gründlich zu testen, sollte ein Datensatz mehr top-aktivierte Neuronen erzeugen.
Die in IIIC beschriebenen Metriken sind eine weitere Inspirationsquelle auf dem Weg zur Generalisierung von Tests. Die Default Detection Rate ist hier als besonders bedeutsam herauszuheben, da die Anzahlt der Fehlklassifikationen fundamental zur Beurteilung eines KNNs bleiben wird.
[23] stellt die Frage, warum es grundsätzlich nötig ist, bessere Kriterien zu finden. Denn bei der NC wird bemängelt, dass es relativ problemlos möglich ist, mit einer kleinen Zahl von Transformationen und Inputs auf Werte nahe 100% zu gelangen. Dies wird der Komplexität der KNNs nicht gerecht und wird daher wohl nicht aussagekräftig genug sein.
E3: Die Testmetrik zur Angabe der Qualität eines KNNs ist noch nicht gefunden. Bis dies in der Zukunft möglicherweise der Fall sein wird, werden Einzelfallbetrachtungen der Aufgabe empfohlen.

C. Schlussfolgerung

Nahezu alle datenerzeugenden Systeme können mit KNNs ausgewertet und diese somit getestet werden. Die Erzeugung von Datensätzen könnte äußerst kostspielig werden. Eine Kosten-Nutzen-Abwägung ist zu treffen.
GANs sind wohl einer der vielversprechendsten Ansätze zur Bildung von automatisierten Tests, ob sie des großen Rätselts Lösung darstellen, wird sich zeigen.
Die Algorithmen auf dem Gebiet der NC haben ihren Beitrag zu Verbesserung der Testqualität geleistet und viele Fehler detektiert. Es stellt sich heraus, dass sie allein, wohl keine geeignete Metrik darstellt. Der Ansatz wird in den neusten Publikationen nicht mehr in dieser exklusiven Form verfolgt.

D. Offene Fragen

Eine große Herausforderung und Forschungsarbeit liegt im Zusammenfügen der ermittelten Ergebnisse. Eine (schnelle) Umsetzung ist keineswegs garantiert.
Der Ansatz der generativen Modelle zum Test steckt noch in den Kinderschuhen und bedarf ebenfalls weiterer Forschung. Fraglich bleibt, ob es in Zukunft unter dem Aspekt der GAN-basierten Tests noch Metriken wie NC bedarf. Ob der GAN-Ansatz zum Aufspüren des Fehlverhaltens durch Corner Cases imstande ist, wird noch erforscht. Es wäre denkbar, dass diese Tests so umfangreich, realistisch sowie Ergebnisse so aussagekräftig sein werden, dass eine indirekte Metrik überflüssig wird. Zu bedenken bleibt hierbei, dass eine gewisse numerische Angabe gerade Nicht-Experten wichtig sein kann und diese zur gesellschaftlichen Akzeptanz beiträgt. Da dies von großer Bedeutung ist, sollte der Ansatz zur weiteren Suche von Test Metriken, auch bei großem Erfolg anderer Ansätze, nicht komplett verworfen werden. Der Ansatz der NC könnte durch diese Metriken ergänzt werden. Hierbei könnte der Kombination und Wichtung zentrale Bedeutung zukommen. Die größte Frage wird bleiben, ob es möglich ist, eine allumfassende Teststrategie zu finden. Ein Scheitern an der Realität in einem in der Anwendung befindlichen System kann möglicherweise nie ausgeschlossen werden. Ob dies für das Assistenzsystem von Nöten ist, steht auf einem anderen Blatt, denn eine Verbesserung der jetzigen Situation wäre nahezu garantiert. Da es sich um eine Unterstützung handelt, fällt die menschliche Komponente zumindest zunächst nicht weg. Es wäre ein Fortschritt.

VI. DANKSAGUNG

Besonderen Dank möchte ich Herrn M.Sc. Hannes Vietz, meinem Mentor, für seine hervorragende Unterstützung und die interessanten Diskussionen aussprechen. Großer Dank gilt auch Herrn Dr. Eric Heintze für spannende Inspirationen und Hilfestellungen. Weiterer Dank gilt Prof. Dr. Dr. Weyrich für die Gelegenheit, diese Arbeit anfertigen zu dürfen.

[1] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. DeepTest: Automated Testing of Deep-neural-network-driven Autonomous Cars. In Proceedings of the 40th International Conference on Software Engineering (Gothenburg, Sweden) (ICSE '18). ACM, New York, NY, USA, 303ś314. https://doi.org/10.1145/3180155.3180220

[2] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: Multi-Granularity Testing Criteria for Deep Learning Systems. In Proceedings of the 2018 33rd ACM/IEEE International Conference on Automated Software Engineering (ASE '18), September 3–7, 2018, Montpellier, France. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3238147. 3238202

[3] Marcel Sommer. Scherben nach autonomer Baggerfahrt durch Glaslabyrinth. https://www.auto-motor-und-sport.de/news/sandvik-autonomer-bagger/

[4] Seokhyun Lee, Sooyoung Cha, Dain Lee, and Hakjoo Oh. 2020. Effective White-Box Testing of Deep Neural Networks with Adaptive Neuron-Selection Strategy. In Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '20), July 18–22, 2020, Virtual Event, USA. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3395363.3397346

[5] M. HOFFMANN, A. POHL, P. PRILL UND M. MLYNARSKI. 2020. CORNER CASES UND IHRE TÜCKEN »Die Gefahren lauern an allen Ecken«. Sonderdruck German Testing Magazin 01.2020. 24-27.

[6] Uli Baumann, Andreas Of-Allinger. TÖDLICHER TESLA-UNFALL MIT LASTWAGEN Autopilot hat Fahrer gewarnt. Auto Motor Sport 20.06.2017.

[7] Prof. Dr.-Ing. Dr.h.c. Michael Weyrich. Skript Automatisierungs-technik 2. Studentenversion WS 20/21.

[8] Prof. Dr.-Ing. Bin Yang. Skript Deep Learning. SS 21

[9] Hamed Habibi Aghdam, Elnaz Jahani Heravi „Guide to Convolutional Neural Networks, A Practical Application to Traffic-Sign Detection and Classification", eBook ISBN978-3-319-57550-6, ISBN : 978-3-319-57549-0, Springer International Publishing AG 2017, https://doi.org/10.1007/978-3-319-57550-6 85-103

[10] Jakob Hoydis, Sebastian Cammerer, Sebastian Dörner, Stephan ten Brink. Deep Learning Applications in Communications. Lecture 5: ConvNets { Modulation Classication}. SS21

[11] Jakob Hoydis, Sebastian Cammerer, Sebastian Dörner, Stephan ten Brink, Deep Learning Applications in Communications Lecture 09: Recurrent Neural Networks – Decoding Convolutional Codes, July 9. 2020

[12] Fabrice Harel-Canada, Lingxiao Wang, Muhammad Ali Gulzar, Quanquan Gu, and Miryung Kim. 2020. Is Neuron Coverage a Meaningful Measure for Testing Deep Neural Networks?. In Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '20), November 8ś13, 2020, Virtual Event, USA. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3368089. 3409754

[13] Vision-based Excavator Activity Analysis and Safety Monitoring System. 2021. Sibo Zhang, Liangjun Zhang. Baido Research. arXiv:2110.03083v2

[14] Miller Trujillo, Mario Linares-Vásquez, Camilo Escobar-Velásquez, Ivana Dusparic, Nicolás Cardozo. 2020. Does Neuron Coverage Matter for Deep Reinforcement Learning? A Preliminary Study. ACM ISBN 978-1-4503-7963-2/20/05. https://doi.org/10.1145/3387940.3391462

[15] Mnih, V., Kavukcuoglu, K., Silver, D. et al. Human-level control through deep reinforcement learning. Nature 518, 529–533 (2015). https://doi.org/10.1038/nature14236

[16] Junhwi Kim, Minhyuk Kwon, and Shin Yoo. 2018. Generating Test Input with Deep Reinforcement Learning. In Proceedings of ACM Conference (Conference'17). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/ nnnnnnn.nnnnnnn

[17] Kexin Pei, Yinzhi Cao, Junfeng Yang, Suman Jana. 2017. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In Proceedings of ACM Symposium on Operating Systems Principles (SOSP '17). ACM, New York, NY, USA, 18 pages. https: //doi.org/10.1145/3132747.3132785

[18] Henrik Bork / Ute Drescher. Baidu stellt autonomen Bagger vor. https://www.konstruktionspraxis.vogel.de/baidu-stellt-autonomen-bagger-vor-a-1039590/

[19] Wei Ma, Mike Papadakis, Anestis Tsakmalis, Maxime Cordy, and Yves Le Traon. . Test Selection for Deep Learning Systems. 1, 1 (May), 17 pages. arXiv:1904.13195v1

[20] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. 2017. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*. 854–863. arXiv:1704.08847v2

[21] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In Proceedings of the 3rd International Conference on Learning Representations. http://arxiv.org/abs/1412.6572

[22] Toohey, J. , Raunak, M. and Binkley, D. (2021), From Neuron Coverage to Steering Angle: Testing Autonomous Vehicles Effectively, Special Issue on Safety, Security, and Reliability of Autonomous Vehicle Software, [online], https://doi.org/10.1109/MC.2021.3079921, https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=932505 (Accessed December 26, 2021)

[23] Jasmine Sekhon, Cody Fleming, 2019. Towards Improved Testing For Deep Learning. https://arxiv.org/abs/1902.06320v1

[24] Yizhen Dong, Peixin Zhang, Jingyi Wang, Shuang Liu, Jun Sun, Ting Dai, XinyuWang, Jianye Hao, LiWang, and Jin Song Dong. 2019. There is Limited Correlation between Coverage and Robustness for Deep Neural Networks. In Proceedings of ACM Conference (Conference'17). ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[25] Yizhen Dong, Peixin Zhang, Jingyi Wang, Shuang Liu, Jun Sun, Jianye Hao, Xinyu Wang, Li Wang, Jin Song Dong, Dai Ting. 2019. There is Limited Correlation between Coverage and Robustness for Deep Neural Networks. https://arxiv.org/abs/1911.05904v1

[26] Jing Yu, Yao Fu, Yanan Zheng, Wang Zheng, and Xiaojun Ye. 2019. Test4Deep: an Effective White-Box Testing for Deep Neural Networks. https://doi.org/10.1109/CSE/EUC.2019.00013

[27] EKF-Based Real-Time Self-Attitude Estimation With Camera DNN Learning Landscape Regularities. 2021. Ryota Ozaki and Yoji Kuroda. https://doi.org/10.1109/LRA.2021.3060442

[28] Fault Identification Based on PD Ultrasonic Signal Using RNN, DNN and CNN. 2018. Qinqin Zhang; Jun Lin; Hui Song; Gehao Sheng. DOI: 10.1109/CMD.2018.8535878

[29] An Adaptive Multi-Sensor Data Fusion Method Based on Deep Convolutional Neural Networks for Fault Diagnosis of Planetary Gearbox. 2017. Luyang Jing, Taiyong Wang, Ming Zhao and Peng Wang. https://doi.org/10.3390/s17020414

[30] Distribution-Aware Testing of Neural Networks Using Generative Models. 2021. Yuanyuan Yuan, Shuai Wang, Mingyue Jiang, Tsong Yueh Chen. DOI: 10.1109/ICSE43902.2021.00032

[31] Exposing Previously Undetectable Faults in Deep Neural Networks. 2021. Isaac Dunn, Hadrien Pouget, Daniel Kroening, Tom Melham. arXiv:2106.00576v1