Automatisierte Optimierung von Metamaterialien im Leichtbau

Untersuchung von Liquid Time-Constant Networks und Long Short-Term Memory Networks

Peter Grohmann,*
Tobias Rosnitschek,
Farzeen Karnoor Althaf und
Stephan Tremmel

Aktuell gewinnen datengetriebene Methoden wie Reduced Order Models (ROMs) und Physics-Informed Machine Learning (PIML) an Bedeutung, da sie physikalische Phänomene effizient modellieren. Ihre Leistungsfähigkeit hängt jedoch von hochwertigen Trainingsdaten ab, die in der Realität häufig nicht verfügbar sind. Diese Arbeit untersucht Long Short-Term Memory (LSTM) und Liquid Time Constant (LTC) Netzwerke zur Abschätzung des Designraums additiv gefertigter Metamaterialstrukturen für optimierte Dämpfungseigenschaften. Trotz begrenzter Datenbasis zeigte das LTC-Netzwerk mit Nadam-Optimierer eine gute Modellgüte für liegend gedruckte Proben, jedoch reduzierte Generalisierbarkeit für stehende Proben. Zudem konnten periodische Buckling-Effekte nicht adäquat abgebildet werden. Die Ergebnisse zeigen das Potenzial von LTC-Netzwerken, weisen aber auch auf bestehende Grenzen hin, die im Kontext der Metamaterialien und Leichtbaustrukturen berücksichtigt werden müssen.

Ausgangslage

Die Entwicklung von Metamaterialien ist ein relativ neues Gebiet in den Ingenieurwissenschaften, begünstigt durch die Freiheiten der additiven Fertigung. Dabei handelt es sich um gezielt aufgebaute Strukturen auf der Meso-, Mikro- und Nanoebene, die in spezifischen räumlichen Konfigurationen angeordnet sind und zu einzigartigen effektiven Eigenschaften auf der Makroebene führen, weshalb sie als Metamaterialien bezeichnet werden. Dieser Ansatz ermöglicht es, Eigenschaften zu erzeugen, die über das hinausgehen, was die einzelnen Bestandteile des Materials bieten. Entsprechend entstehen so große Designräume, die es effizient abzudecken gilt, um Metamaterialien gezielt für spezifische technische Anforderungen zu entwickeln. Soll dieser Ansatz auf neue Produkte übertragen werden, ergeben sich eine Vielzahl von Designvariablen und örtlich unterschiedliche Zielfunktionen. In Bild 1 ist das am Beispiel einer additiv gefertigten, mechanisch beanspruchten Mittelsohle dargestellt, je nach Bereich gelten unterschiedliche Steifigkeiten als ideal, die wiederum abhängig vom Träger des Schuhs und dessen Einsatzzweck sind. Somit muss im Zuge der Produktentwicklung eine Vielzahl von Gitterstrukturen zuerst im Rechner erstellt, vernetzt, mittels Finite-Elemente-Simulation analysiert und schließlich ausgewertet werden; dieser Workflow ist zeitaufwändig und rechenintensiv.

Im konkreten Fall aus Bild 1 betrug die durchschnittliche Vernetzungsdauer etwa 20 min auf einer Workstation mit Intel-Core-i9-10980XE-CPU, 128 GB DDR4. Die quasi-statische Simulation mit Stauchung der Struktur um 70 % unter Ver-

* **Korrespondenzautor**
Peter Grohmann, M. Sc.; Lehrstuhl für Konstruktionslehre und CAD, Universität Bayreuth; Universitätsstr. 30, 95447 Bayreuth; Tel.: +49 (0) 0921 55-7144, E-Mail: peter.grohmann@uni-bayreuth.de

Weitere Autoren
Dr.-Ing. Tobias Rosnitschek; Lehrstuhl für Konstruktionslehre und CAD, Universität Bayreuth
Farzeen Karnoor Althaf, B. Sc.; Lehrstuhl für Konstruktionslehre und CAD, Universität Bayreuth
Prof. Dr.-Ing. Stephan Tremmel; Lehrstuhl für Konstruktionslehre und CAD, Universität Bayreuth

Hinweis
Bei diesem Beitrag handelt es sich um einen von den Advisory-Board-Mitgliedern des ZWF-Sonderheftes wissenschaftlich begutachteten Fachaufsatz (Peer-Review).

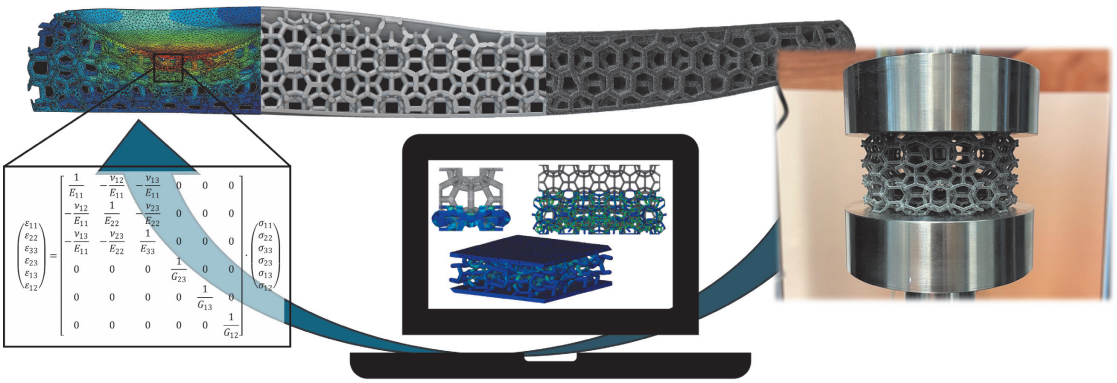

Bild 1. Exemplarische Darstellung eines Workflows zur Auslegung einer personalisierten Mittelsohle mit Auswertung und Anpassung der Gitterstrukturen zur Optimierung der effektiven Steifigkeit basierend auf Stauchversuchen und FE-Simulationen

wendung von 16 Rechenkernen dauerte im Durchschnitt 72 Stunden. Diese Zeitdauer ist im Rahmen einer effizienten Bauteilgestaltung und Produktoptimierung nicht praktikabel, weshalb die Entwicklung geeigneter Metamodelle zur Auswertung des effektiven Strukturverhaltens maßgeblich dazu beitragen kann, die Marktreife von additiv gefertigten Metamaterialien signifikant voranzutreiben. Entsprechend geht dieser Artikel der Frage nach, ob datengetriebene Modelle, in diesem Fall Modelle zur Zeitreihenvorhersage wie Long Short-Term Memory Networks (LSTM) [1] und Liquid Time-Constant Networks (LTC) [2], beides spezielle Formen von Recurrent Neural Networks (RNNs), effektiv eingesetzt werden können, um das nichtlineare Deformationsverhalten von Metamaterialien vorherzusagen und somit den Designprozess zu verbessern, insbesondere zu beschleunigen. Dabei verfolgt der Artikel die folgenden Ziele:

- Analyse der Eignung von LTC-Netzwerken für die Vorhersage langfristiger Deformationsmuster in Metamaterialien bei geringer Datenverfügbarkeit,
- Identifikation von Limitierungen der LTC-Architektur in diesem Kontext sowie
- Kombination mit LSTM-Netzwerken, um festzustellen, ob dadurch eine bessere Leistung erzielt werden kann.

Stand der Technik

Die im Rahmen dieser Arbeit verwendeten Netzwerke unterscheiden sich von den klassischen Feedforward Neural Networks (FNNs), deren Aufbau und Weiterentwicklungen werden im Folgenden kurz dargestellt.

Recurrent Neural Networks (RNNs)
RNNs stellen nach [3] eine Klasse von neuronalen Netzen dar, die speziell zur Erkennung von Mustern und Abhängigkeiten in sequenziellen Daten eingesetzt wird. Im Gegensatz zu Feedforward-Neural Networks (FNNs), die Informationen in nur eine Richtung verarbeiten, ermöglichen RNNs nach [3] und [4] Informationen durch Rückkopplungen in das Netzwerk zurückzuführen und auf diese Weise frühere Eingaben in die Verarbeitung einzubeziehen (Bild 2).

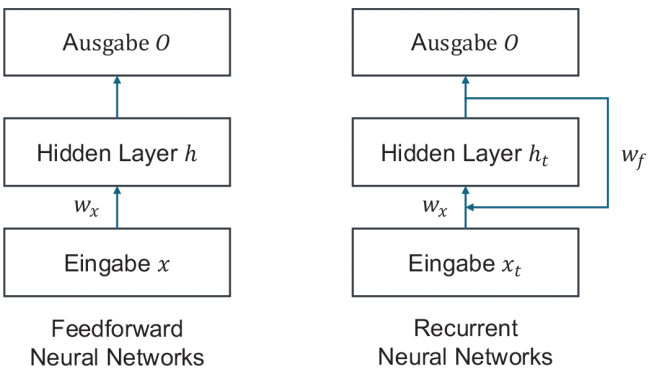

Bild 2. Unterschied zwischen FNNs und RNNs. FNNs verarbeiten Informationen nur vorwärts, während RNNs durch Rückkopplungen frühere Eingaben in die Berechnungen einbeziehen, nach [3] (abgeändert)

Eine Form der RNN-Aktivierung für die Aktualisierung des Hidden Layer (h_t) in einem RNN lässt sich nach [4] durch folgende Gleichung (1) beschreiben:

$$h_t = \sigma(w_f h_{t-1} + w_x x_t + b) \quad (1)$$

Hierbei sind:
h_t: der versteckte Zustand (Hidden Layer) zum Zeitpunkt t,
x_t: die Eingabe zum Zeitpunkt t,
w_f: Gewichtsmatrix für den vorherigen Zustand h_{t-1},
w_x: Gewichtsmatrizen für die Eingabe x_t,
b: der Bias,
σ: eine Aktivierungsfunktion (z. B. tanh, sig oder ReLU).

Zentraler Bestandteil des Trainings von RNNs ist der Backpropagation Through Time Algorithmus (BPTT), welcher eine Erweiterung des Standard-Backpropagation Algorithmus für FNNs ist. Beim BPTT wird der Fehler über alle Zeitschritte zurückverfolgt, um die Gewichte des Netzwerks anzupassen. Die Verlustfunktion über alle Zeitschritte T lautet nach [3]:

$$L(O,Y) = \sum_{t=1}^{T} l_t(O_t, Y_t) \quad (2)$$

wobei l_t die Verlustfunktion (z. B. mittlere quadratische Abweichung oder Kreuzentropie) und die Größen O_t und Y_t die vorhergesagten und Zielwerte zu einem Zeitpunkt t darstellen. Ein wesentliches Problem bei der Anwendung beim BPTT ist das sogenannte „Vanishing Gradient"-Problem. Dabei nimmt der Gradient, der bei der Rückpropagation durch die Zeit berechnet wird, mit zunehmender Anzahl an Zeitschritten ab. Dieser Gradient ist ein Maß dafür, wie stark sich die Feh-

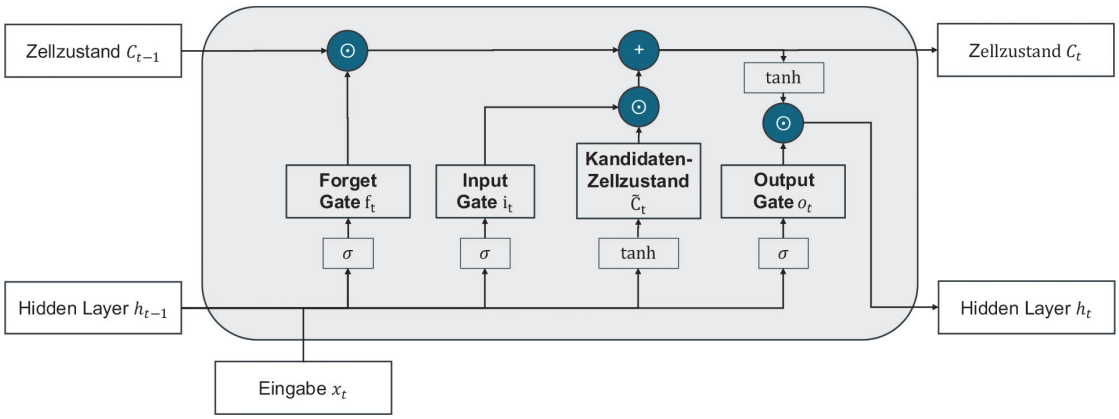

Bild 3. Diagramm einer LSTM-Zelle: Es zeigt den Informationsfluss durch Forget Gate, Input Gate und Output Gate, die den Zellzustand und den Hidden State steuern, um langfristige Abhängigkeiten in sequenziellen Daten zu erfassen, nach [3] (abgeändert)

lerrate des Netzwerks ändert, wenn man dessen Gewichte anpasst, und gibt dadurch die Richtung und Größe der Anpassung der Gewichte vor. Beim Training über viele Zeitschritte multiplizieren sich die Gradienten miteinander. Wenn nun die Gradienten häufig Beiträge kleiner als 1 haben, kann dies dazu führen, dass die Gesamtgradienten über die Zeit gegen Null gehen. Bei Werten größer als 1 kann ein exponentielles Anwachsen der Gradienten erfolgen, was zum „Exploding Gradient"-Problem führt. In beiden Fällen wird das Lernen langfristiger Abhängigkeiten erschwert – entweder, weil die Anpassungen der Gewichte vernachlässigbar werden, oder weil das Training instabil wird. Um dieses Problem zu mildern, wird häufig der Truncated BPTT (TBPTT) eingesetzt, bei dem die Rückpropagation auf eine feste Anzahl von Zeitschritten beschränkt wird, um zu verhindern, dass der Gradient über viele Zeitschritte hinweg zu stark abnimmt oder anwächst [3].

Long Short-Term Memory Networks (LSTMs)

LSTMs wurden 1997 durch Hochreiter und Schmidhuber in [5] als eine erweiterte Form von RNNs, vorgestellt, um das „Vanishing Gradient"-Problem zu verringern. Die Schlüsselkomponente einer LSTM-Zelle ist die sogenannte Speicherzelle. Diese besitzt einen Zellzustand C_t der es ermöglicht, Information über Zeitschritte hinweg zu speichern und zu aktualisieren, wodurch der Gradient über längere Zeiträume zurückgeführt werden kann, ohne dass dieser exponentiell abnimmt [5]. Dabei steuern das Input Gate (i_t) und das Output Gate (o_t) den Informationsfluss in und aus der Zelle heraus. Zudem wurden in [6] LSTMs mit dem Forget Gate (f_t) um die Möglichkeit erweitert, spezifische Teile des Zellzustands zu verwerfen oder für den nächsten Zeitschritt beizubehalten (Bild 3).

Der Zellzustand C_t ergibt sich nach [3] aus dem Zusammenspiel des Forget Gate f_t, Input Gate i_t und des Kandidaten-Zellzustands $\tilde{C}_t$ durch elementweise Multiplikation $\odot$:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{3}$$

Der neue Hidden Layer h_t wird schließlich wie folgt berechnet:

$$h_t = o_t \odot \tanh(C_t) \tag{4}$$

Liquid Time-Constant Networks (LTCs)

LTCs wurden erstmals 2021 von *Hasani et al.* eingeführt. Sie stellen nach [2] eine Klasse von zeitkontinuierlichen RNNs dar. Im Gegensatz zu diskreten RNNs modellieren LTCs die zeitliche Dynamik durch gewöhnliche Differenzialgleichungen (GDGL):

$$\frac{dx(t)}{dt} = f(x(t), I(t), t, \theta) \tag{5}$$

wobei

$x(t)$: der Zustand des Systems zum Zeitpunkt t,
$I(t)$: Eingabevektor zum Zeitpunkt t,
f: eine nichtlineare Funktion, parametrisiert durch die Parameter des Neuronalen Netzwerks Θ (Gewichte und Biases),

Zur numerischen Stabilisierung der GDGL wird in [7] vorgeschlagen, einen Dämpfungsterm hinzuzufügen:

$$\frac{dx(t)}{dt} = -\frac{x(t)}{\tau} + f(x(t), I(t), t, \theta) \tag{6}$$

wobei τ die Zeitkonstante ist.

LTCs erweitern dieses Modell, indem sie die Zeitkonstante dynamisch, basierend auf den aktuellen Eingaben, anpassen:

$$\tau_{sys} = \frac{\tau}{1 + \tau f(x(t), I(t), t, \theta))} \tag{7}$$

Die variable Zeitkonstante und der stabilisierende Term ermöglichen es, Dynamiken und Langzeitabhängigkeiten potenziell besser abzubilden als herkömmliche RNNs [2].

Materialien und Methoden

Die Datengrundlage für die durchgeführte Studie bilden Druckversuche an zylinderförmigen Gitterstruktur-Probekörpern, wie rechts in Bild 1 im gefertigten Zustand zu sehen. Diese Metamaterialien wurden aus der Weaire-Phelan-Einheitszelle [8] mit unterschiedlichen Strebendurchmessern aufgebaut. Gefertigt wurden die Probekörper aus thermoplastischem Polyurethan (TPU) durch selektives Lasersintern (PBF-LB/P). Die Druckversuche wurden bis zu einer Stauchung von 70 Prozent durchgeführt, dabei wurden je vier Belastungs- und Entlastungskurven gemessen, um den Mullins-Effekt (fortschreitende zyklische Entfestigung) zu berücksichtigen. Als Daten für die Modellvorhersage wurde die als technisch relevant erachtete Belastungskurve im vierten Zyklus verwendet. In Summe wurden 40 Versuche durchgeführt, davon flossen 30 in das Modelltraining und in die Modellvalidie-

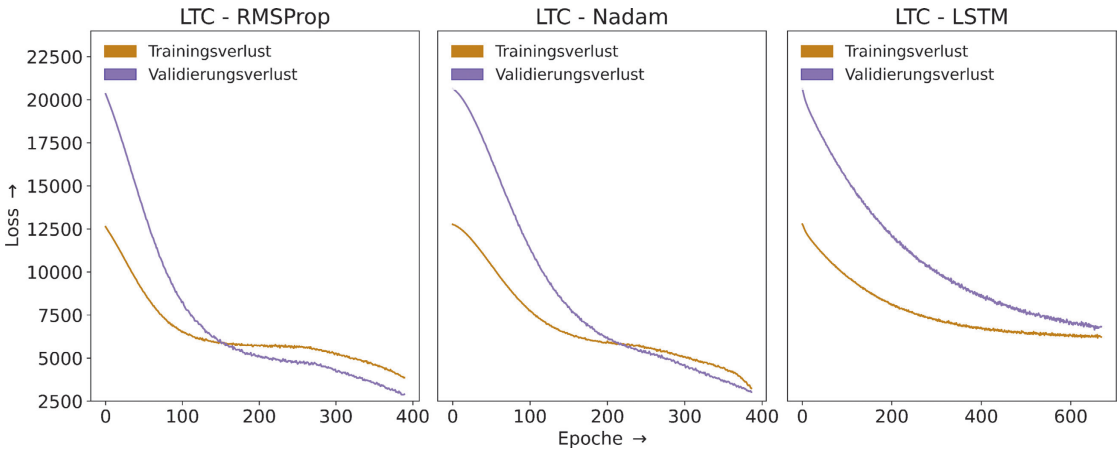

Bild 4. Vergleich des Trainingsverlaufs der drei untersuchten Konfigurationen, über alle 30 Trainingsdatensätze gemittelt

rung ein und 10 wurden als Testdaten verwendet; Jede Zeitserie bestand dabei aus den 507 Zeitschritten der vierten Belastungskurve. Variiert wurden dabei Strebendurchmesser und Druckrichtung, wobei die Testfälle T1 bis T5 liegend und die Testfälle T6 bis T10 stehend gedruckt wurden.

Für die Anwendung datengetriebener Modelle zur Deformationsvorhersage mittels LTC und LSTM wurden drei verschiedene Konfigurationen untersucht, die nachfolgend erläutert werden. Für den Aufbau der Modelle wurde als Grundlage das in [2] angegebene Git-Repository verwendet. Alle Modelle wurden unter Verwendung der Bibliotheken Numpy [9], Pandas [10], Tensorflow [11, 12], Keras [13], Keras-Tuner [14] und Matplotlib [15] aufgebaut. Die Auswertung fand mithilfe der Metriken mittlerer quadratischer Fehler und Bestimmtheitsmaß R^2 statt.

Bei allen Modellen wurden die Orientierung im Druckraum und der Strebendurchmesser als Eingangsdaten, sowie die Kraft-Verschiebungskurve als Ausgangsdaten verwendet. Alle Modelle werden verwendet, um die vollständige Zeitreihe der Kraft-Stauchungskurve vorherzusagen.

Die Daten wurden zu Beginn mit einem One-Hot-Encoder codiert, sodass alle Eingangsdaten als diskrete numerische Werte vorlagen und anschließend über einen Standard-Scaler standardisiert, indem von den einzelnen Datenpunkten der Mittelwert abgezogen und das Ergebnis durch die Standardabweichung geteilt wurde. Da für eine sinnvolle Kreuzvalidierung die verfügbare Datenmenge zu gering war, wurde die L2-Norm zur Regulierung der Gewichte verwendet. Ebenso wurde ein Dropout-Block in jeden Layer eingebaut, um eine Überanpassung an spezifische Neuronen zu verhindern.

Den Grundaufbau aller Modelle stellte ein LTC mit zwei Layern dar, dabei bestand jeder Layer aus 10 LTC-Blöcken und einem Dropout-Block. Dabei wurde für alle Konfigurationen eine Hyperparameterstudie durchgeführt, um eine Vergleichbarkeit der Modelle zu gewährleisten. In der Hyperparameterstudie wurden die Anzahl der Neuronen zwischen 50 und 300 sowie die Dropout-Rate von 0,3 bis 0,7, die L2-Regulierung von 0,0001 bis 0,01 und die Lernrate von 0,0001 bis 0,01 variiert und der so aufgespannte Suchraum per Zufallssuche mit 15 Versuchen abgetastet.

Für die erste Konfiguration LTC-RMSProp wurde der RMSProp-Optimierer [16] verwendet, welcher für Zeitreihendaten konzipiert ist, mit adaptiven Lernraten arbeitet und typischerweise verrauschte Gradienten robust bewältigt. RMSProp passt die Lernrate basierend auf dem gleitenden Durchschnitt der quadratischen Gradienten an, was ihn besser geeignet macht für Probleme, bei denen nicht jede Richtung der Optimierung gleich wichtig ist. Für diese Konfiguration wurde eine Lernrate von 0,001 gewählt und beide Layer wiesen 200 Neuronen auf.

Die zweite Konfiguration LTC-Nadam verwendete den Nadam-Optimierer [17], welcher sich in vielen Fällen dem weit verbreiteten Adam-Optimierer überlegen zeigt und in der Literatur als Alternative zum RMSProp-Optimierer propagiert wird. Für diese Konfiguration wurden eine Lernrate von 0,0005 und 230 Neuronen je Schicht verwendet.

Die dritte Konfiguration LTC-LSTM stellte die Kombination eines LTC-Layer mit 10 Blöcken und eines LSTM-Layer mit ebenfalls 10 Blöcken dar. Als Optimierer wurde RMSProp verwendet. Hinter dieser Konfiguration steckt die Überlegung, dass LTC-Netzwerke zwar geringere Datenmengen als LSTMs benötigen, aber insbesondere für kurzfristige Abhängigkeiten geeignet sind. Die Vorhersage des Deformationsverhaltens kann das Erinnern kumulativer Effekte erfordern. Dafür können LSTM-Netzwerke aufgrund ihrer Spezialisierung auf langfristige Abhängigkeiten die bessere Lösung darstellen. In diesem Fall wurden für den LTC-Layer 200 Neuronen, für den LSTM-Layer 100 Neuronen und als Lernrate 0,001 verwendet.

Alle Modelle wurden für maximal 1000 Epochen trainiert, wobei das Training abgebrochen wurde, wenn für mehr als 15 Iterationen keine Verbesserung detektiert wurde.

Ergebnisse und Diskussion

Der Trainingsverlauf der drei Konfigurationen ist in Bild 4 dargestellt. Dabei ist auffällig, dass bei dem hybriden LTC-LSTM-Modell das Training etwa 200 Epochen länger dauerte als bei den beiden untersuchten (reinen) LTC-Netzwerken.

Ebenso lässt sich aus Bild 4 erkennen, dass bei allen Modellen mit zunehmender Epoche eine Abnahme sowohl des Trainings- als auch des Validierungsver-

Modell	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
LTC-RMSProp	0,69	0,52	-1,78	0,21	0,86	0,85	0,69	0,95	0,44	0,70
LTC-Nadam	0,93	0,96	0,68	0,92	0,89	0,17	0,16	0,54	-0,05	-0,28
LTC-LSTM	0,2	0,26	0,77	-0,03	-0,03	-0,77	-0,8	-0,47	-1,14	-1,36

Tabelle 1. Übersicht über die Modellperformance anhand des Bestimmtheitsmaß R^2

lustes zu beobachten ist. Eine leichte Überanpassung ist jedoch feststellbar, insbesondere bei den Modellen mit den Optimierern RMSProp und Nadam, wo der Validierungsverlust im Verhältnis zum Trainingsverlust in Teilen weniger stark abfällt. Insgesamt erscheint die Überanpassung auf dieser Grundlage jedoch moderat.

Zur Auswertung der Modellperformance ist das Bestimmtheitsmaß R^2 der Modelle für alle 10 Testdatensätze in Tabelle 1 aufgetragen.

Aus den Ergebnissen geht hervor, dass für die liegend gedruckten Proben T1 bis T5 mit dem LTC-Modell und dem Nadam-Optimierer trotz der begrenzten Datenbasis eine gute Modellperformance erreicht werden konnte. Hierbei ist zu beachten, dass das für die Modellbewertung genutzte Bestimmtheitsmaß durch Overfitting und Ausreißer verzerrt werden kann. So nimmt etwa die Modellgüte für stehend gedruckte Proben T6 bis T10 deutlich ab, was wiederum auf eine eingeschränkte Generalisierbarkeit hindeutet. Interessanterweise verhält sich das gleiche Netzwerk mit dem RMSProp-Optimierer teilweise entgegengesetzt – auch wenn die Modellperformance über die Testdaten hinweg teilweise signifikant geringer ausfällt. Die Annahme, dass die Kombination eines LTC-Netzwerks für kurzzeitige Dynamik mit der Fähigkeit des kumulativen Speicherns des LSTM-Netzwerks von Vorteil für die Vorhersage der Kraft-Stauchungskurve ist, konnte anhand der Ergebnisse nicht bestätigt werden. Zudem war keines der untersuchten Modelle in der Lage, die lokalen Instabilitäten und periodischen Beuleffekte der zugrunde gelegten Kraft-Stauchungskurven adäquat zu erfassen. Die Vorhersage ist exemplarisch für die drei Modellkonfigurationen und den Datensatz T1 in Bild 5 dargestellt.

Hier unterscheiden sich die Modelle signifikant in ihrer Fähigkeit, die experimentell ermittelte Kraft-Stauchungskurve für T1 nachzubilden. Das Modell LTC-RMSProp zeigt über weite Teile der Kurve eine relativ stabile Annäherung an die experimentellen Daten. Allerdings überschätzt es systematisch die Kraft im Bereich geringer bis mittlerer Stauchung. LTC-Nadam liefert wiederum unter den Modellen das beste Bestimmtheitsmaß für den Testfall T1. Dabei erfasst es die experimentellen Werte von 0 bis 20 mm Stauchung sehr präzise. Danach nimmt die Diskrepanz zwischen den vorhergesagten und tatsächlichen Werten zu, was darauf hindeutet, dass das Modell die Dynamik bei höheren Stauchungen nicht ausreichend erfassen kann. Beim LTC-LSTM zeigen sich bereits im Anfangsbereich signifikante Abweichungen zu den gemessenen Werten. Es unterschätzt anschließend durchgehend die Kraft. Auch ist anhand der Abbildung zu erkennen, dass keines der Modelle in der Lage ist, die periodischen Beuleffekte abzubilden. Ursache hierfür könnte die begrenzte Anzahl an Samples sein, mit denen die Modelle trainiert wurden. Dadurch könnten nicht alle Variabilitäten ausreichend im Training repräsentiert worden sein, wodurch das spezifische Deformationsverhalten durch die Modelle nicht genügend generalisiert werden konnte. Zum anderen beeinflusst die Wahl des Optimierers das Trainings- und Konvergenzverhalten signifikant. Die vorliegenden Ergebnisse sollten demnach vor dem Hintergrund der begrenzten Datenbasis interpretiert werden und nicht als allgemeingültig betrachtet werden. Die Ergebnisse zeigten, dass für eine zuverlässige Modellierung des komplexen Materialverhaltens auch mit LTCs eine angemessene Datenmenge erforderlich ist. Zukünftige Arbeiten könnten demnach darauf abzielen, den Datensatz zu erweitern und unter Umständen die Modellarchitekturen sowie Hyperparameter weiter zu optimieren, um die Generalisierungsfähigkeit, Modellgüte sowie Aussagekraft der Ergebnisse weiter – und deutlich – zu erhöhen. Dies könnte durch die Integration zusätzlicher experimen-

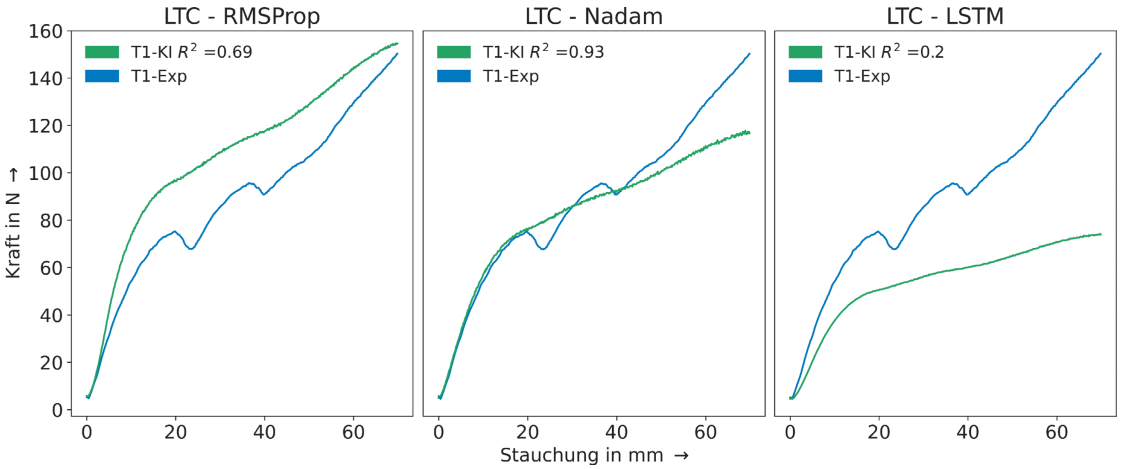

Bild 5. Vergleich der vorhergesagten (grün) mit der gemessenen (blau) Kraft-Stauchungskurve für den Testfall T1 und alle drei untersuchten Modelle

teller Daten oder – aufgrund der in diesem Kontext hohen zeitlichen und ressourcenintensiven Generierung umfangreicher Datensätze – durch den Einsatz synthetischer Datenerweiterungstechniken erreicht werden.

Schlussfolgerungen

In diesem Beitrag wurde der Einsatz von LTCs zur datengetriebenen Vorhersage von Kraft-Stauchungskurven bei Metamaterialien untersucht. Dabei stand das industriell relevante Szenario geringer Datenverfügbarkeit im Mittelpunkt. Die Ergebnisse der Studie zeigten, dass LTCs mit dem Nadam-Optimierer trotz der eingeschränkten Datenbasis in der Lage waren, für liegend gedruckte Proben angemessene Vorhersagen zu liefern. Allerdings wies das Modell bei den stehend gedruckten Proben eine deutlich reduzierte Modellgüte auf, was auf eine noch eingeschränkte Generalisierbarkeit des Modells hindeutet. Auch konnte keines der Modelle die periodischen Beuleffekte zufriedenstellend abbilden, was insgesamt die Herausforderungen der datengetriebenen Vorhersage komplexer mechanischer Systeme bei einem quantitativ begrenzten Datensatz unterstreicht. Das kombinierte LTC-LSTM-Hybridmodell führte nicht zu den erwarteten Verbesserungen. Die Ergebnisse deuten darauf hin, dass sowohl die Datengrundlage als auch die Modellarchitekturen zu optimieren sind, um einerseits die dynamischen Effekte der Kraft-Stauchungskurven adäquat zu erfassen und andererseits die Generalisierbarkeit der Modelle auf Variationen, wie z. B. der Druckrichtung, zu verbessern. Zukünftige Arbeiten könnten sich daher auf die zielgerichtete Erzeugung zusätzlicher experimenteller Daten oder – aufgrund der damit verbundenen Herstellungs- und Versuchsaufwände – auf den Einsatz von Data-Augmentation-Techniken konzentrieren. Darüber hinaus könnten optimierte Trainingsstrategien und die Untersuchung alternativer Modellansätze dazu beitragen, die Abbildung der dynamischen Effekte zu verbessern. Insgesamt zeigen die Ergebnisse der Untersuchungen das Potenzial von LTCs für die datengetriebene Vorhersage des mechanischen Verhaltens von Metamaterialien im Kontext des Leichtbaus, weisen aber zugleich auch auf die damit verbundenen Herausforderungen hin.

Literatur

1. Staudemeyer, R. C.; Morris, E. R.: Understanding LSTM – a Tutorial into Long Short-Term Memory Recurrent Neural Networks. 2024 DOI:10.48550/arXiv.1909.09586
2. Hasani, R.; Lechner, M.; Amini, A. et al.: Liquid Time-constant Networks. In: Proceedings of the AAAI Conference on Artificial Intelligence 35 (2021) 9, S. 7657–7666 DOI:10.1609/aaai.v35i9.16936
3. Schmidt, R. M.: Recurrent Neural Networks (RNNs): A Gentle Introduction and Overview. 2019 DOI:10.48550/arXiv.1912.05911
4. Sherstinsky, A.: Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. Physica D Nonlinear Phenomena 404 (2020) 8 DOI:10.1016/j.physd.2019.132306
5. Hochreiter, S.; Schmidhuber, J.: Long Short-Term Memory. Neural Computation 9 (1997) 8, S. 1735–1780 DOI:10.1162/neco.1997.9.8.1735
6. Gers, F. A.; Schmidhuber, J.; Cummins, F.: Learning to Forget: Continual Prediction with LSTM. Neural Computation 12 (2000) 10, S. 2451–2471 DOI:10.1162/089976600300015015
7. Funahashi, K.; Nakamura, Y.: Approximation of Dynamical Systems by Continuous Time Recurrent Neural Networks. Neural Networks 6 (1993) 6, S. 801–806 DOI:10.1016/S0893-6080(05)80125-X
8. Kusner, R.; Sullivan, J. M.: Comparing the Weaire-Phelan Equal-volume Foam to Kelvin's Foam. Forma 11 (1996) 3, S. 233–242
9. Harris, C. R.; Millman, K. J.; van der Walt, S. J. et al.: Array Programming with NumPy. Nature 585 (2000), S. 357–362 DOI:10.1038/s41586-020-2649-2
10. Reback, R.; McKinney, W.; Brockmendle, J. et al.: pandas-dev/pandas: Pandas 1.0.3. Zenodo, 03/2020. DOI:10.5281/ZENODO.3715232
11. Abadi, M; Agarwal, A.; Barham, P.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015 (https://www.tensorflow.org/s/results?q=Martin%20Abadi [Abgerufen am 23.1.2025])
12. TensorFlow Developers, TensorFlow. Zenodo. 12/2021 DOI:10.5281/ZENODO.5799851
13. Keras. (https://keras.io [Abgerufen am 23.1.2025])
14. KerasTuner. (https://github.com/keras-team/keras-tuner [Abgerufen am 23.1.2025])
15. Caswell, T. A.; Droettboom, M.; Lee, A. et al.: matplotlib/matplotlib: REL: v3.5.1. Zenodo. 2021 DOI:10.5281/ZENODO.5773480
16. Ruder, S.: An Overview of Gradient Descent Optimization Algorithms. 2017 DOI:10.48550/arXiv.1609.04747
17. Dozat, T.: Incorporating Nesterov Momentum into Adam. 2016. (https://api.semanticscholar.org/CorpusID:70293087 [Abgerufen am 23.1.2025])

Die Autoren dieses Beitrags

Peter Grohmann, M.Sc., studierte Applied Research and Engineering Sciences an der Hochschule Ansbach. Seit April 2022 ist er als wissenschaftlicher Mitarbeiter am Lehrstuhl für Konstruktionslehre und CAD tätig.

Dr.-Ing. Tobias Rosnitschek studierte Materialwissenschaften und Werkstofftechnik an der Universität Bayreuth und promovierte 2023 am Lehrstuhl für Konstruktionslehre und CAD. Er leitet seit 2022 die Forschungsgruppe Material und Tribologie am Lehrstuhl für Konstruktionslehre und CAD.

Farzeen Karnoor Althaf absolviert derzeit sein Masterstudium im Elitestudiengang Scientific Computing an der Universität Bayreuth. Er besitzt einen B.Sc. in Maschinenbau und arbeitet als studentische Hilfskraft am Lehrstuhl für Konstruktionslehre und CAD, dort konzentriert er sich auf numerische Methoden, Optimierung und maschinelles Lernen für wissenschaftliche Anwendungen.

Prof. Dr.-Ing. Stephan Tremmel studierte Maschinenbau in der Studienrichtung Rechnerunterstützte Methoden der Produktentwicklung an der Friedrich-Alexander-Universität Erlangen-Nürnberg und promovierte am Lehrstuhl für Konstruktionstechnik. Nach langjähriger Tätigkeit in Forschung und Lehre am Lehrstuhl für Konstruktionstechnik als Abteilungsleiter und stellvertretender Lehrstuhlleiter übernahm er 2021 die Leitung des Lehrstuhls für Konstruktionslehre und CAD an der Universität Bayreuth.

Abstract

Automated Optimization of Metamaterials in Lightweight Engineering. Investigation of Liquid – Time-Constant Networks and Long Short-Term Memory Networks. Currently, data-driven methods such as Reduced Order Models (ROMs) and Physics-Informed Machine Learning (PIML) are gaining significance due to their ability to efficiently model physical phenomena. However, their performance heavily depends on high-quality training data, which is often unavailable in real-world engineering scenarios. This study investigates Long Short-Term Memory (LSTM) and Liquid Time Constant (LTC) networks for estimating the design space of additively manufactured metamaterial structures with optimized damping properties. Despite a limited dataset, the LTC network with the Nadam optimizer demonstrated good model

accuracy for horizontally printed samples but exhibited reduced generalizability for vertically printed samples. Moreover, periodic buckling effects could not be adequately captured. The results highlight the potential of LTC networks while also underscoring existing limitations that must be considered in the context of metamaterials and lightweight structures.

Danksagung

Die Autoren danken dem Bundesministerium für Wirtschaft und Klimapolitik und dem Programm für Technologietransfer und Leichtbau (TTP-LB) für die finanzielle Unterstützung unter dem Förderkennzeichen 03LB3054A. Dieser Beitrag entstand zudem im Kontext des Forschungsprojekts Gate2HPC, welches vom Europäischen Fonds für Regionale Entwicklung und der Oberfrankenstiftung gefördert wird.

Datenverfügbarkeit

Der vollständige Datensatz und Ergebnisse aller Messkurven werden auf Anfrage von den Autoren zur Verfügung gestellt.

Interessenkonflikte

Die Autoren erklären, dass keine Interessenskonflikte bestehen.

Schlüsselwörter

Datengetriebene Modellierung, Liquid Time Constant Networks, Long-Short-Term Memory Networks, Metamaterialien, Leichtbaustrukturen

Keywords

Data-Driven Modelling, Liquid Time Constant Networks, Long Short-Term Memory Networks, Metamaterials, Lightweight Engineering

Bibliography

DOI:10.1515/zwf-2024-0142
ZWF 120 (2025) Special Issue; page 250 – 256

ISSN 0947-0085 · e-ISSN 2511-0896