Technische Hochschule
Brandenburg
University of
Applied Sciences
Fachbereich
Informatik und Medien

Online Deep Learning mit Hedge-Backpropagation für Predictive Maintenance-Anwendungen

Ursina Bisang

Bachelorarbeit • Studiengang ACS • Fachbereich Informatik und Medien • 01.08.2018

Aufgabenstellung

Ziel der Arbeit war es, mit einem neuronalen Netz, das mit dem Hedge-Backpropagation Algorithmus trainiert wurde, eine Vorhersage auf einem Datensatz über Flugzeugmotoren (Turbofan Datensatz) zu treffen. Es sollte getestet werden, ob der Algorithmus sich für Predictive Maintenance eignet. Um dies zu prüfen wird ein Vergleich mit einem neuronalem Netz von einem anderen Typ (Long short-term memory) durgeführt.

Predictive Maintenance

Predictive Maintenance ist eine Wartungsmethode, bei der der aktuelle Zustand des gewarteten Systems mit Sensoren überwacht wird, dadurch werden unerwartete Ausfälle verringert und so Wartungskosten gespart.
Für Predictive Maintenance mit Machine Learning werden Daten benötigt, mit diesen wird ein Modell trainiert, so dass es Fehler und Wartungsbedarf möglichst früh erkennt.

Online Deep Learning

Deep Learning beschreibt das Lernen mit neuronalen Netzen, wobei der Lernschritt nach der Evaluierung von jedem Datum stattfindet, anstatt wie sonst nach der Verarbeitung einer größeren Menge Daten.

Hedge Backpropagation

Hedge Backpropagation ist eine Variante der Backpropagation, mit der neuronale Netze normalerweise lernen. Er wurde 2017 von Sahoo et al [SPHL17] entwickelt und ist für das Online Deep Learning optimiert. Das heißt er löst Probleme, die die Backpropagation mit Online Learning hatte.
In Abbildung 1 ist zu sehen, wie Hedge-Backpropagation funktioniert.
Zunächst werden Daten wie bei der normalen Backpropagation vorwärtspropagiert, also nach vorne durch das Netz geleitet und dabei verarbeitet (blaue Pfeile). Parallel läuft die Vorwärtspropagierung, die für Hedge-Backpropagation typisch ist. Hierbei wird für jede verborgene Schicht ein Ausgang $f^{(l)}$ berechnet. Diese werden mit dem Hedge-Algorithmus zusammengenommen und zum Ausgang F_t für das ganze Netz verrechnet (oragene Pfeile). Die grünen Pfeile stehen für den Rückwärtsverarbeitungs-schritt bei der Hedge-Backpropagation, hier werden alle neuen Werte berechnet und alte Werte aktualisiert.

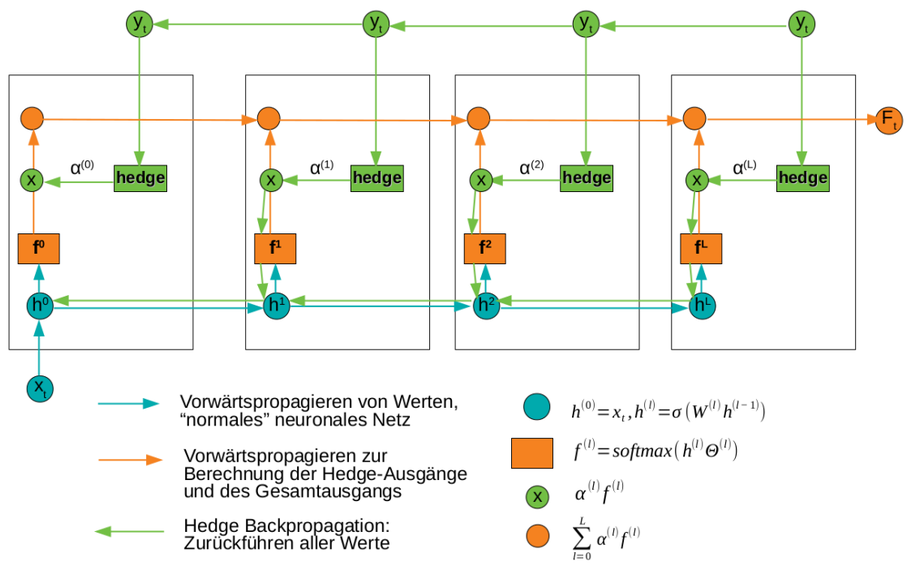

Abb. 1: Schematische Darstellung der Funktionsweise von Hedge-Backpropagation (angelehnt an Abb.1 aus [SPHL17])

Umsetzung

Es wurde ein neuronales Netz mit Hedge-Backpropagation trainiert, dafür wurde eine von Sahoo et al auf GitHub zur Verfügung gestellte Implementierung genutzt.
Im ersten Schritt wurden die genutzten Daten über die Abnutzung von Flugzeugmotoren vorbereitet. Die Daten wurden erst normalisiert, das heißt, die Werte der Daten wurden auf das Intervall zwischen 0 und 1 abgebildet, wobei 0 dem kleinsten vorkommenden Wert entspricht und 1 dem größten.
Dann wurden die Daten gefenstert. Es wurden also Daten über die 30 letzten Durchlaufzyklen zu den aktuellen Daten hinzugefügt, damit das Netz auch Veränderungen über längere Zeiträume erlernen kann.
Dann wurde eine Hyperparameteroptimierung durchgeführt. Das heißt es wurden neuronale Netze mit verschiedenem Aufbau trainiert und ihre Ergebnisse wurden verglichen. Dies war etwas schwierig, da die genutzte Implementierung keine Vorhersagen treffen kann und somit die Trainingsergebnisse nich überprüft werden konnten.
Verglichen wurden Memorynutzung und Zeitaufwand beim Training. Hier zeigt sich, dass Hedge-Backpropagation im Training effizienter mit beiden Ressourcen umgeht (vgl. Abb. 2).

Wert	HBP	LSTM
peak_memory	417.52 MiB	665.28 MiB
increment	58.33 MiB	136.61 MiB
Bestzeit	19.8 s	-
Durchschnitt	-	1min 41s
Standardabweichung	-	28.3 s

Abb. 2: Ergebnisse der Vergleiche von HBP und LSTM

Ergebnis

Die GitHub-Implementierung ist aktuell nicht für Predictive Maintenance geeignet, da sie nicht vorhersagen kann. Sobald sie dies kann, ist sie im Training effizienter als ein LSTM, kann also aus dieser Perspektive empfohlen werden.

Betreuung: Dipl.-Inform. Ingo Boersch, Prof. Dr.-Ing. Sven Buchholz • Technische Hochschule Brandenburg
Prof. Dr. rer. nat Adrian Paschke • Fraunhofer FOKUS

Quellen: [SPHL17] Doyen Sahoo, Quang Pham, Jing Lu, Steven C.H. Hoi (2017): Online Deep Learning: Learning Deep Neural Networks on the Fly, GitHub-Repository mit Code: https://github.com/LIBOL/ODL [29.07.2018].