Anwendung von KI und ML zur Messdatenverarbeitung aus Energieversorgungskomponenten

Bachelorarbeit

Sergej Lamert - 727245
Fakultät angewandte Informatik

4. Februar 2021

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Buck Wandler sind heutzutage in aller Munde, sie stellen beispielsweise Spannung in Notebook Prozessoren und Ladegeräten zu verfügung, Regeln den Strom an Stepper-Motoren. Durch die weitläufigen Anwendungsfälle, ist es besonders in der heutigen Zeit, in der Künstliche Intelligenz und Machine Learning Algorithmen immer wichtiger werden, zu evaluieren, inwieweit sich solche Wandler für Machine Learning Anwendungen eignen, wenn Daten über Strom, Spannung, Leistung und Temperatur bekannt sind. Deshalb ist das Thema dieser Arbeit die Analyse eines Hybrid-Buck-Wandlers. Im Folgenden wird solch ein Wandler hinsichtlich seiner Anwendbarkeit für Themen im Bereich Maschine Learning und Zustandsüberwachung analysiert werden. Die Analyse erfolgt dabei in mehreren Schritten. Zuerst werden Daten erhoben, während der Wandler an unterschiedliche statische Lasten angeschlossen ist, um einen Überblick über das Verhalten des Wandlers zu gewinnen. Anschließend wird der Wandler an eine dynamische Last angeschlossen und es werden periodische Signale gemessen. Die gemessenen Daten werden dann in Echtzeit mit bereits bekannten Daten verglichen und ebenfalls gegen andere, parallel laufende Wandler, abgestimmt. Das Letztendliche Ziel dieser Arbeit ist es, zu evaluieren, wie präzise und zuverlässig die Daten sind und inwieweit diese sich für Anwendungen in den am Anfang genannten Bereichen eignen. Da der vorliegende Wandler über einen Mikrocontroller mit I2C Schnittstelle verfügt, aus welchem Strom, Spannungs und Temperaturdaten erhoben werden können, wird das Erfassen der Daten durch das vorhandene Protokoll stark vereinfacht und sehr portierbar gemacht, was es ermöglicht die Daten aus beliebigen Platformen (Win 10, Linux, etc.) oder einfach durch andere Mikrocontroller (z.B. Raspberry Pi) auszulesen.

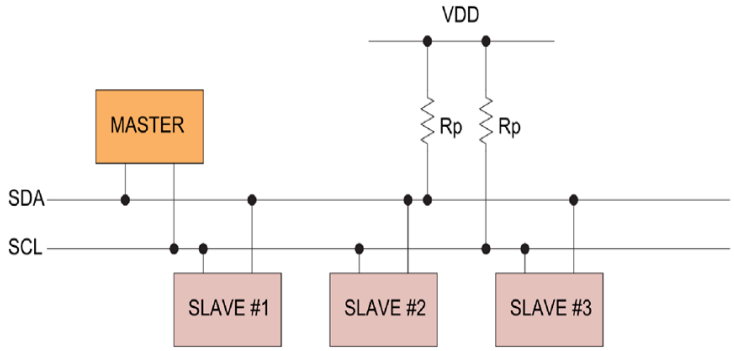

Abbildung 1: I2C Bus Beispiel Quelle: https://www.analog.com/en/technical-articles/i2c-primer-what-is-i2c-part-1.html

2 Theoretischer Hintergrund

Dadurch bedingt, das diese Arbeit sich an vielen Konzepten aus der Signaltheorie, Elektrotechnik und Informatik bedient, ist es sinnvoll vorab einige Begriffe und Konzepte zu definieren. Aus diesem Grund, werden im Folgenden das I²C Protokoll, verwendete Software, sowie die genutzte Hardware näher erläutert werden.

2.1 Hybrid-Buck-Wandler

Der Buck-Wandler nimmt als Eingangsspannung 15 - 40 Volt entgegen und reduziert diese auf festgelegte 12 V Ausgangsspannung. Der maximale Strom ist auf 10 A limitiert. Sollte der Strom das Maximum übersteigen, so wird die Spannung gedrosselt. Des Weiteren besitzt der Wandler eine I²C Schnittstelle, aus der Ausgangsspannung, Eingangsspannung, Ausgangsstrom und Temperatur des Boards gemessen und ausgelesen werden können. Die Datenleitungen des I²C Busses sind bereits auf dem Wandler selbst mit Pull-Up Widerständen ausgestattet.

2.2 I2C

I2C steht für Inter-Integrated Circuit und wurde von Philips Semiconductors 1982 entwickelt. Es handelt sich dabei um einen seriellen Datenbus im Master-Slave Stil. Dieses Protokoll setzt auf zwei Leitungen für den Datenaustausch, davon ist ein Kanal explizit für die Daten vorgesehen und der andere Kanal für den Takt. Der Takt beträgt im klassischen 100 KHz oder 400 KHz.

Es ist zu erkennen, dass es eine Master -und mehrere Slave Komponenten gibt. Der Master gibt allen Slaves vor, was sie zu senden haben, und

wie schnell sie es tun sollen. Des Weiteren ist zu erkennen, das die beiden Leitungen SDA und SCL an einem Pull-Up Widerstand angeschlossen sind. Dieser dient dazu, die Leitungen, wenn weder Master noch Slave sendet, auf die Versorgungsspannung zu schalten, sodass keine undefinierten Zustände und die Leitung im unbenutzten Zustand eine logische 1 besitzt. Die Kommunikation erfolgt durch Adressen, so besitzt z. B. jeder Sensor eine I2C Slave Adresse, über die ein Master auf diese zugreifen kann. Das auslesen eines Slaves erfolgt per Registeradressen, so hat beispielsweise ein beliebiger I^2C fähiger Sensor ein Register, in dem bestimmte Werte gespeichert sind. In dieser Arbeit handelt es sich bei dem I2C Slave um einen Mikrocontroller auf dem Buck-Wandler. Dieser Slave ist in der Lage Daten zum Strom der Eingangs -und Ausgangsspannung und Temperatur auszugeben. Der Master ist dabei entweder ein PC, welcher per PicKit (I2C auf USB) damit kommuniziert, oder ein Raspberry Pi, welcher direkt per I2C mit dem Netzteil verbunden werden kann.

2.3 Verwendeten Platformen, Programmiersprachen und Bibliotheken

Für die statischen Tests wurde Windows 10 als Platform und C als Programmiersprache gewählt, da allein diese mit dem PICkit kompatibel sind.

Da sich herausgestellt hat, dass ein PicKit eine zu geringe Abtastrate ermöglicht, wurde für die dynamischen Tests ein Raspberry Pi 4 (4 GB RAM) mit der Programmiersprache Python für Datenerfassung- und Verarbeitung verwendet. Des Weiteren wurde für das Erstellen von Neuronalen Netzen die library "Tensorflow"verwendet, wobei Keras als High-Level API dient. Die graphische Visualisierung der Daten erfolgt ausschließlich per Pyplot library.

2.4 Neuronale Netze

Zur durchführung einiger simpler Testfälle unter den Punkten 3.2.1 und 3.2.2 ist es notwendig, eines der Grundlegenden Konzepte von Machine Learning zu nennen: das Neuronale Netz. Neuronale Netze gehören zur maschine learning Kategorie des ßupervised learning". Beim supervised learning wird ein neuronales Netz anhand von Eingabedaten (Input) und bereits definierten Lösungen (Output) daraufhin optimiert, bei gegebenen Input und Output Daten eine passende Lösungsstrategie zu finden.

In Abb. 2 erkennt man eine Abbildung des neuronalen Netzes. Dieses ist aufgeteilt in mehrere Schichten (layers). Das Input layer nimmt die Daten auf, welche verarbeitet werden müssen. Im hidden layer werden die Daten durch mathematische Operationen verarbeitet, um im Output layer Aussagen über die Input Daten machen zu können. Des Weiteren erkennt man, das es mehrere Knoten gibt, welche untereinander vollvermascht sind. Diese

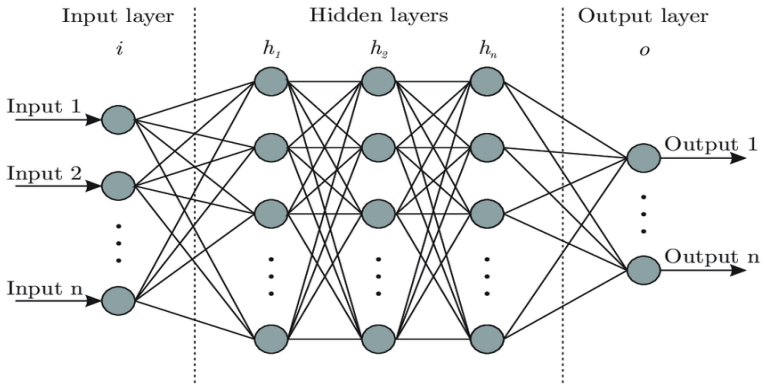

Abbildung 2: Beispiel eines Neuronalen Netzes: https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o$_fig1_321259051$

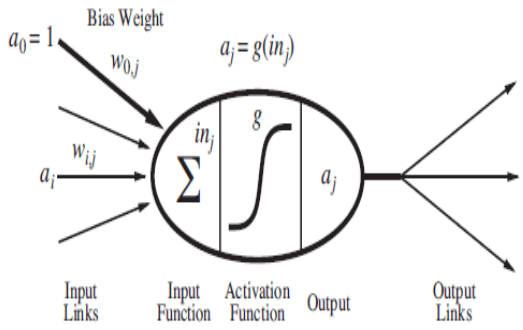

Abbildung 3: Konzept eines Neurons:

werden Neuronen genannt.

In Abb. 3 erkennt man die Struktur eines Neurons. Diese beinhaltet mehrere Verarbeitungsschritte. Zuerst werden die Inputs des Neurons mit seinen Gewichtungen (weights) durch Multiplikation verrechnet und anschließend aufsummiert. Darauffolgend wird auf das Ergebnis eine Aktivierungsfunktion angewandt, um zu bestimmen, wie äktiv"das Neuron ist. Es existiert eine Mehrzahl verschiedener Aktivierungsfunktionen, von denen zwei im nachfolgenden erläutert werden. Das Ergebnis ist dann der Output Wert des Neurons oder auch sein Zustand Diese Verarbeitungsschritte können auch per Matrizen-Schreibweise dargestellt werden:

$$\mathrm{A}\left(\begin{bmatrix} w_{10} w_{11} \\ w_{20} w_{21} \\ w_{30} w_{31} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

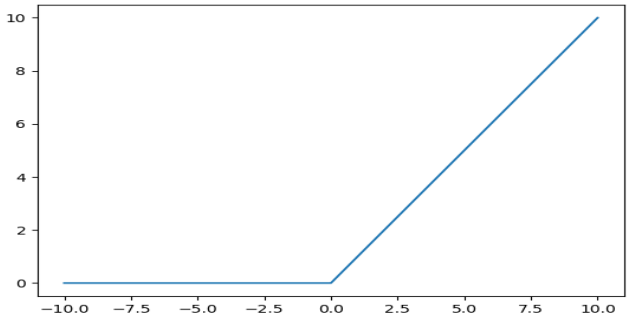

Abbildung 4: Rectified Linear Unit

In Abb. 2 erkennt man ein Diagram eines Neuronalen netzes. Die Kreise re die Neuronen, die Kante stellen Verbindungen unter den Neuronen dar. Es ist zu beachten, das in diesem Model die Neuronen von einer Schicht N vollvermascht mit den Neuronen einer Schicht N+1 sind. Des Weiteren ist zu erkennen, dass der dargestellte Graph drei Schichten (engl. Layers) besitzt. Der Input Layer dient dabei als Ausgangspunkt, an dieser Stelle wird in jedes Neuron ein numerischer Wert eingespeist. Alle Werte im Input Layer zusammengenommen ergeben einen Vektor V.

Abschließend erfolgt eine weitere Matrixmultiplikation des Hidden Layers und seinen Weights, um das Output Layer zu generieren. Das Output Layer gibt, wie der Name schon sagt, die Ausgangswerte der Berechnung. Bei Klassifizierungsaufgaben sind die Werte im Output Layer Warscheinlichkeiten für die jeweiligen, zu klassifizierenden Klassen. Die Anzahl der Neuronen im Output Layer ist gleich der Anzahl der Klassen.

Die Formel für die Sigmoid Funktion:

$$s(z) = \frac{1}{1 + e^{-z}} \tag{1}$$

In Abb. 4 ist eine der beliebtesten Aktivierungsfunktionen zu sehen, die ReLu Funktion. Wenn der Input der Funktion ¡= 0 ist, dann ist der Output immer 0. Für Werte ¿ 0, ist der Output immer der selbe Wert

$$y = \begin{cases} 0 & \text{if } x <= 0 \\ x & \text{if } x > 0 \end{cases}$$

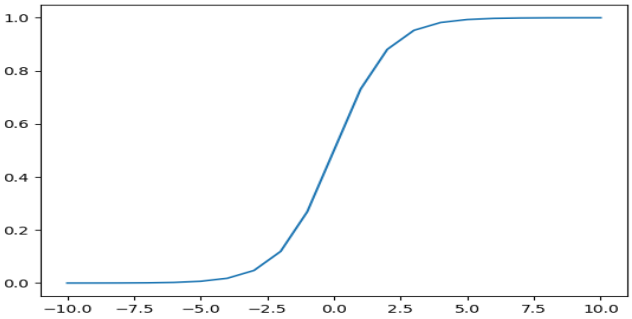

Abbildung 5: Sigmoid Funktion

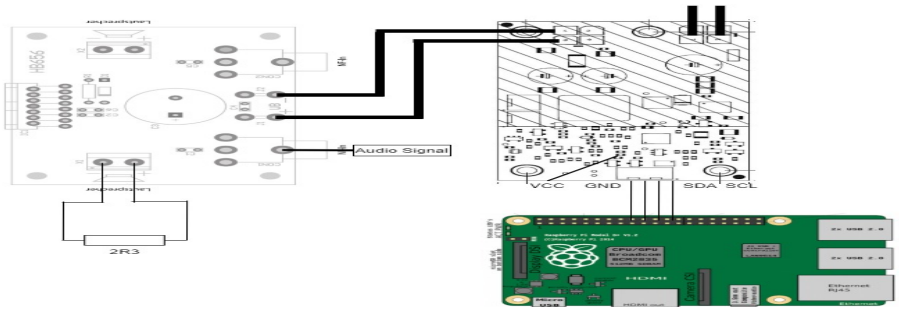

Abbildung 6: Versuchsaufbau der dynamischen Datenerfassung

3 Methodik

3.1 Statische Datenerfassung

Die statische Datenerfassung dient dem Zweck, die Funktionalität der Wandler unter einer konstanten Last zu überprüfen. Es wurden zwei simple Testfälle aufgebaut. Bei beiden handelt es sich um Parallelschaltungen von 4,7 Ohm Widerständen. Der erste Fall schaltet zwei davon parallel, der zweite drei. Durch das Anwenden des Ohmschen Gesetzes, kann ein theoretischer Stromfluss und somit auch die theoretische Leistungsaufnahme berechnet werden. Der Wandler wird durch ein Netzteil mit 24 Volt Ausgangsspannung betrieben. Diese 24 Volt werden von dem Wandler auf 12 V herabgesetzt. Somit kann der Strom berechnet werden, der durch die Widerstände fließt:

$$\frac{U}{R} = I \tag{2}$$

$$\frac{12V}{4,7Ohm} * 2 = 5,10A \frac{12V}{4,7Ohm} * 3 = 7,65A \tag{3}$$

Da es sich in diesen Testfällen um statische Lasten handelt, sind keine hohen Abtastraten erforderlich, deshalb wird zur Datenerfassung ein PicKit mitsamt I²C Schnittstelle verwendet und kein Raspberry Pi.

Des Weiteren ist zu erwähnen, dass mehrere Wandler getestet wurden, einer davon mit welcher fehler. Bei der Messung der Daten, zeigt sich, dass sowohl Temperatur als auch Ausgangsstrom, dieses Wandlers unter gleicher Last höher waren als die der anderen, was man in Abb. X erkennen kann

3.2 Dynamische Datenerfassung

Das Ziel der dynamischen Datenerfassung ist es, zu evaluieren, wie der Wandler sich unter Lasten verhält, die sich mit der Zeit ändern. Sowie mögliche Abweichungen zwischen den einzelnen Wandlern selbst. Der Testaufbau beinhaltet den Hybridwandler, einen Stereoverstärker, welcher per

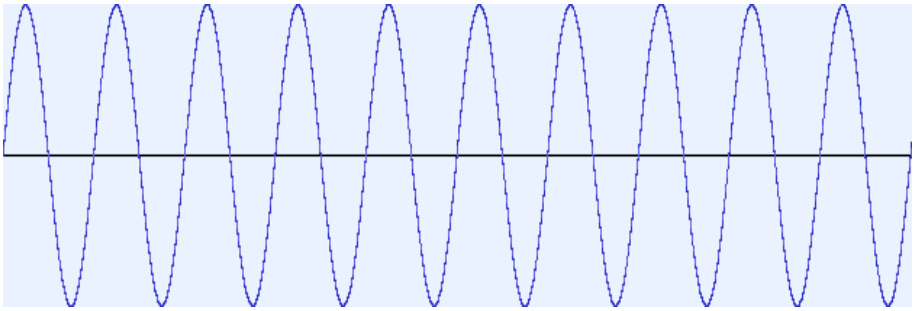

Abbildung 7: In Audacity generiertes 10 Hz Sinussignal

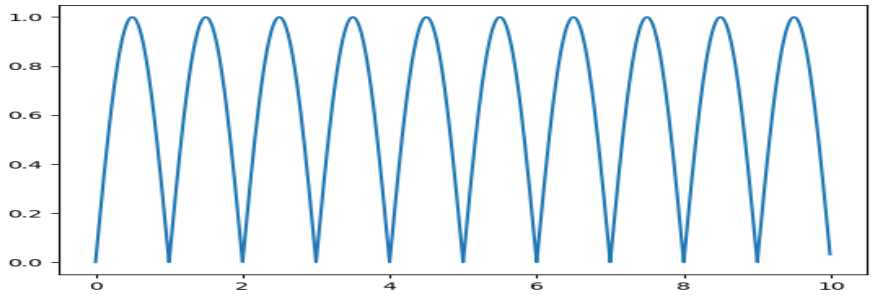

Abbildung 8: Effektive Frequenz eines 10 Hz Signals ist 20 Hz, weil, weil es für den Verstärker egal ist ob + oder - Signal

Wandler mit Strom versorgt wird und in den ein Audiosignal eingespeist wird. In Abb. 5 ist die Schaltung des Testaufbaus abgebildet. Es ist zu erkennen, das am Eingang des Buck-Wandlers 24 V eingespeist werden und das die am Ausgang liegenden 12 V an den Stereoverstärker angeschlossen sind. Des Weiteren ist ersichtlich, dass nur ein Kanal des Stereoverstärkers genutzt wird. Dieser Kanal wird mit einem Audiosignal aus einem externen Rechner versorgt. Als Signale werden hierbei Sinussignale mit variierender Frequenz verwendet, welche mit dem Programm Audacity generiert werden (https://www.audacityteam.org/). Der erste Testschritt ist dabei die Überprüfung, wie hoch die maximale Frequenz eines Signals sein kann, sodass interpretierbare Daten erzeugt werden können. Die maximale Frequenz kann durch praktische Tests empirisch bestimmt werden. In Abb. 4 ist das generierte Signal zu sehen, welches in den Verstärker eingespeist wird. Dabei handelt es sich in diesem Beispiel um ein 10 Hz Sinussignal.

Das Messen der Daten erfolgt ebenfalls per I^2C, jedoch auf einem Raspberry Pi, da dieser im Vergleich zum PicKit höhere Datenraten ermöglicht. Das Verarbeiten der Daten erfolgt per Python.

Die ermittelte Abtastrate des Wandler, welche bei einer I^2C Taktrate von 70 KHz erfolgte, beträgt im durchschnitt hier den durchschnitt berechnen

In Abb. 6 ist der ungefilterte Leistungsverlauf des Wandler dargestellt.

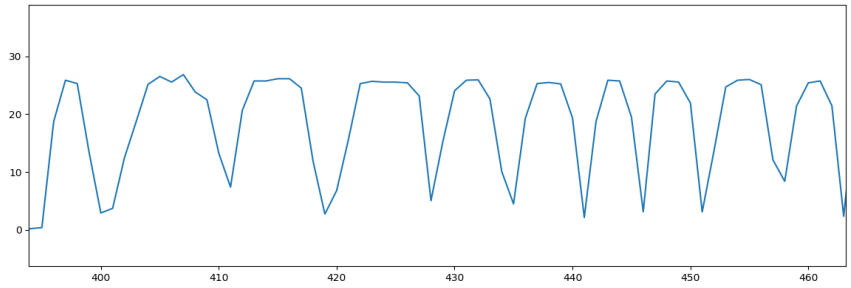

Abbildung 9: Tatsächlich gemessene Leistung des Wandlers

Es ist ersichtlich, dass das 10 Hz Sinussignal in Abb. 5, dem gemessenen Sinussignal, entspricht. Der unsaubere Verlauf der Leistung ist auf Rauschen innerhalb der Hardware und auf die Abtastrate zurückzuführen.

$$f_{\text{nyquist}} = 1/2 * f_{\text{abtastrate}} \tag{4}$$

Experimentelle Werte zeigen, dass die maximale Abtastfrequenz bei ca. 100 Hz liegt, somit ist die Nyquistfrequenz des Signals 50Hz.

Die linken Wert repräsentieren die Zeitstellen in Sekunden, die Rechten sind die Signaldaten, welche dann per Programm zu einer Kurve interpoliert werden.

3.2.1 Echtzeitmonitoring

Das Ziel des Echtzeitmonitoring ist es, in laufenden Betrieb Aussagen darüber zu treffen welche Last gerade an dem Wandler angeschlossen ist. Eine Last ist in diesem Fall ein Audiosignal, welches durch einen Stereo Verstärker, welcher an den Wandler angeschlossen ist, verstärkt wird. Zur Klassifizierung, welche Last, bzw. welches Signal gerade am Stereoverstärker angeschlossen ist, wird ein neuronales Netz verwendet, welches per Tensorflow API generiert wird. Die gemessenen Signale werden dann vom Raspberry Pi, zwecks Verminderung der Rechenleistung, per Socket an einen Rechner geschickt, welcher dann die Daten per Neuronales Netz interpretiert und eine entsprechende Klassifizierung berechnet. Als Testfälle wurden Sinussignal mit verschiedenen Frequenzen verwendet, um die allgemeine Anwendbarkeit von Deep-Learning Algorithmen zu verifizieren.

3.2.2 Vergleichsmonitoring mehrerer Wandler im laufenden Betrieb

Das Ziel des Vergleichsmonitoring ist es, eine Software Struktur aufzubauen, welche es ermöglicht, mehrere Wandler gleichzeitig auf einem Raspberry Pi per I^2C zu betreiben und die Daten in Echtzeit zu visualisieren und in

Textdateien abzuspeichern. Dadurch, dass die verwendeten Wandler unterschiedlich Adressen besitzen, müssen keine Ergänzungen an der Schaltung durchgeführt werden, es reicht einzig die Datenleitungen (SDA, SCL) zusammenzuschalten. Während der Messung der Wandler, werden die Daten von jedem Messpunkt in Echtzeit auf ein Koordinatensystem per PyPlot übertragen.

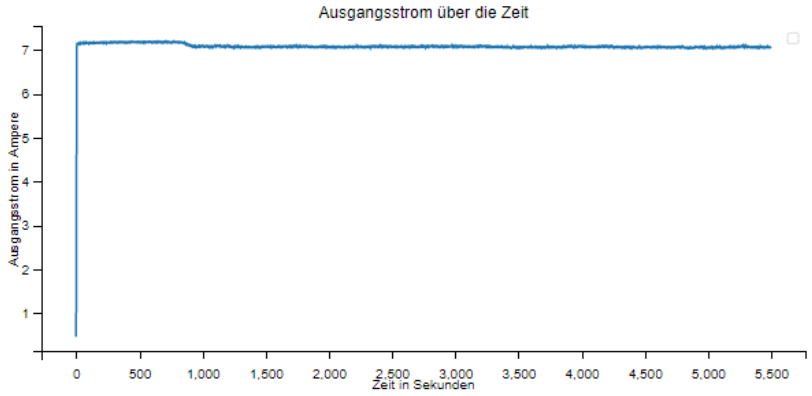

Abbildung 10: Ausgangsstrom über die Zeit mit Filter

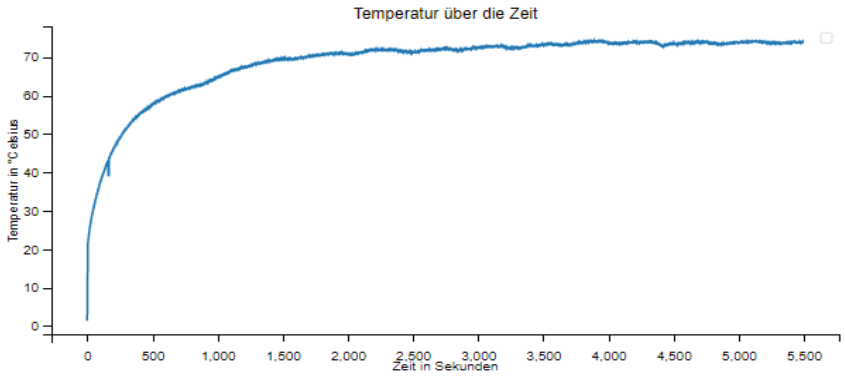

Abbildung 11: Temperatur über die Zeit

4 Ergebnisse

In Abb. x ist der Ausgangsstrom geplotet gegen die Zeit zu sehen. Der erste Graph ist eine darstellung der Daten ohne Frequenz filter, der zweite mit. Der Frequenzfilter ist für die visualisierung notwendig, da der Wandler eine Standardabweichung von mehreren Milliampere besitzt. Für weiteres wird nun immer eine Visualisierung mit Frequenzfilter verwendet insofern nicht anders genannt. Es ist in Abb. x zu erkennen, das der Strom nach einer Gewissen Zeit absackt, es liegt nahe, dass dies an der steigenden Temperatur assoziiert ist,

Bei der Temperatur ist erkenntlich, das diese anfangs Rasant ansteigt und nach etwa 1600 Sekunden auf eine Endtemperatur konvergiert. Es ist bei T

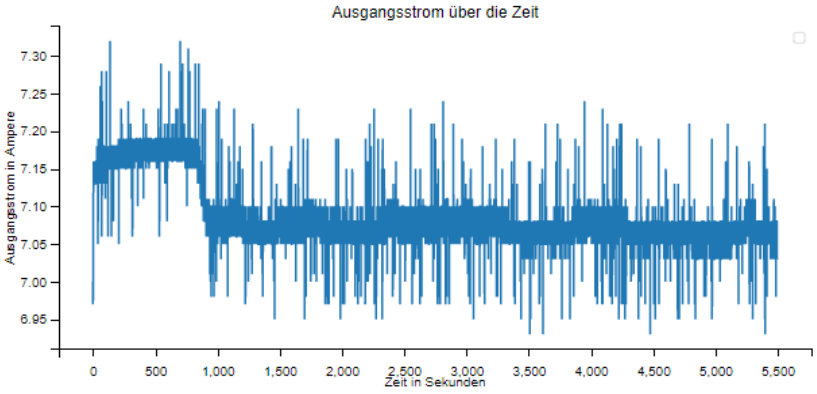

Abbildung 12: Ausgangsstrom über die Zeit

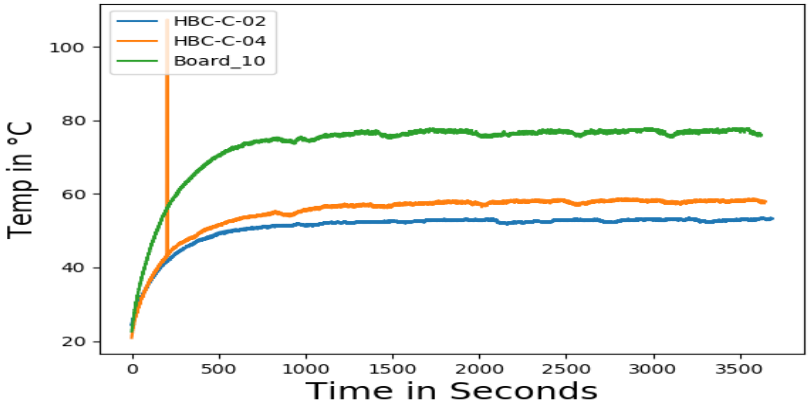

Abbildung 13: Temperaturvergleich mehrerer Wandler

= 250 s erkenntlich, das der Graph einen ausreiser hat, dies hat zwei ursachen: die 1. und fundamentalste ist, dass der I2C Mikrocontroller auf dem Board in intervallen den Wert -20.51 ausgibt, dadurch, dass dies dann entsprechend ebenfalls frequenzgefiltert wird, ergibt sich so eine kleine auslenkung. hlEs steht außer Frage, dass gelegentliche Sprünge zu negativen Werten, durchaus kritisch für das System sein können, wenn die Daten weiterverarbeitet und abhängig von diesen Daten Aktionen durchgeführt werden sollen. Daher werden Daten, während sie verarbeitet werden, dahingehen überprüft, ob solche Fehlerfälle auftreten, und sofort beseitigt. Das beseitigen kann dabei mit mehreren Methoden erfolgen :

Abb. 11 zeigt den gefilterten Stromverlauf der selben Wandler sowie der selben Lasten, nämlich 2 parallel geschaltene Widerstände wie in Abb. 10.

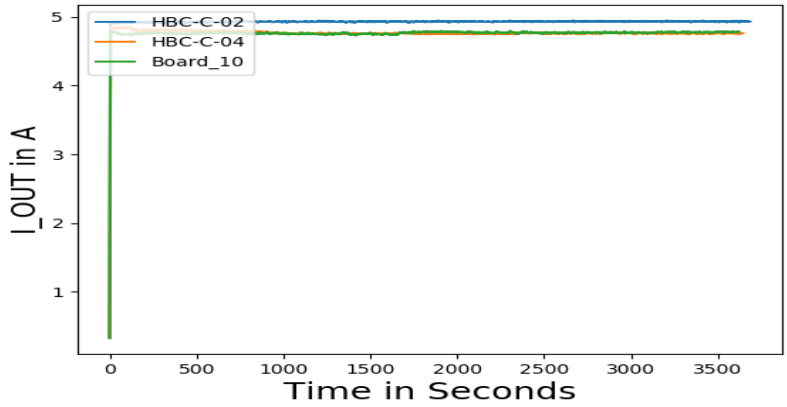

Abbildung 14: Stromvergleich mehrerer Wandler

Es ist zu erkennen, dass der Wandler "HB-C-02ëinen höheren Strom ausgibt als die anderen beiden. Dies ist auch erkenntlich, sobald die Mittelwerte angeschaut werden: 4.933414 für HB-C-02, 4.774112 für HB-C-04 und 4.769322 für Board10. Des Weiteren ist ersichtlich, dass HB-C-04 und Board10 relativ ähnliche Ströme liefern. Ebenfalls nennenswert ist die Standardabweichung der Wandler, diese beträgt bei HB-C-02 0.0165 A, bei HB-C-04 0.0274 A und bei Board10 0.0218 A. Es zeigt sich nun, dass der Wandler HB-C-02 tendenziell besser Arbeitet als die anderen beiden, da diese geringere Schwankungen hat und liefert, bzw. liefert die an dem theoretischen Stromfluss liegen
In Abb. 10 ist ein Temperaturvergleich von mehreren Wandler über die Zeit dargestellt. Das offensichtlichste ist der Wandler mit der Bezeichnung "Board10", dieser hat nämlich eine über 20 Grad Celsius höhere Einschwingtemperatur. Dies ist dadurch erklärbar, dass das PWM Signal des Wandler keinen stati Zustand erreicht. Ein weiterer Aspekt, der sofort ersichtlich ist, ist der extreme Ausschlag beim Wandler mit der Bezeichnung "HB-C-04", bei diesem springt die Temperatur für einen kurzen Moment auf über 100 °C. Dies ist durch Fehlerhafte Sensordaten erklärbar und kann entweder nachträglich oder zur Laufzeit korrigiert werden. Eine dritte Beobachtung die gemacht werden kann, ist die, das die Wandler "HB-C-02ünd "HB-C-04"relativ ähnliche Temperaturverläufe haben falls mehr Wandler man die Standardabweichung und den Mittelwert bestimmen.
Abb. 13 zeigt die Genauigkeit des Neuronalen Netzes in der Vorhersage der verschiedenen Frequenzen anhand der Trainingsdaten. Es ist aus dem Graphen ersichtlich, dass die Genauigkeit über 1000 Trainingsepochen 99% beträgt.

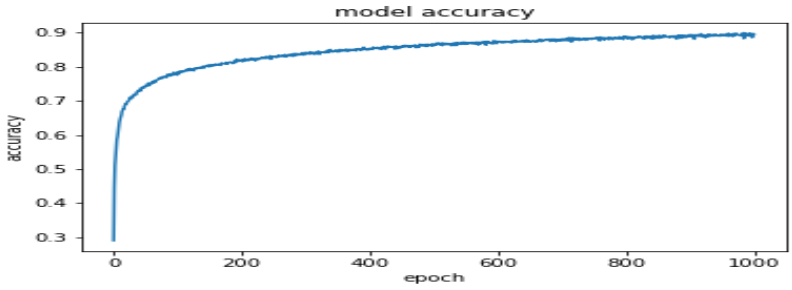

Abbildung 15: Genauigkeit der Anwendbarkeitstests

5 Literaturverzeichnis