Titel der Abschlussarbeit

Bachelorarbeit
von
Marius Wodtke

an der Fakultät für Informatik

In dem Studiengang
Informatik (B.Sc.)

eingereicht am 01.01.01 beim
Institut für Angewandte Informatik
und Formale Beschreibungsverfahren
des Karlsruher Instituts für Technologie

Referent: Prof. Dr. Hartmut Schmeck
Betreuer: Kaibin Bao

KIT – Universität des Landes Baden-Württemberg und
nationales Forschungszentrum in der Helmholtz-Gemeinschaft

Eidesstattliche Erklärung

Ich versichere hiermit wahrheitsgemäß, die Arbeit und alle Teile daraus selbständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderung entnommen wurde.

Karlsruhe, den DATUM *NAME*

Inhaltsverzeichnis

Abkürzungsverzeichnis

NILM	Non-intrusive load monitoring
KNN	Künstliches Neuronale Netz
ESHL	Energy Smart Home Lab
MLP	Multi Layer Perceptron

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Energie zu sparen ist seit Langem ein Ziel der Umweltpolitik, dennoch steigt der Energieverbrauch in Industrieländern kontinuierlich und es werden nach wie vor Wege gesucht, diesen zu senken. In den USA verursachen private Haushalte 40% des CO2-Ausstoßes [VBG08] und stehen deshalb im Fokus vieler Programme zum Energiesparen.
Je nach Studie besteht für die Haushalte ein Einsparungspotential von bis zu 20% [AGSA13] für dessen Energieverbrauch, Fischer [Fis08] untersucht mehrere dieser Studien und beschreibt ein durchschnittliches Einsparungspotential von 5-12%. Um dieses Einsparungspotential auszunutzen, muss der Nutzer regelmäßig und möglichst genau über seinen Verbrauch aufgeklärt werden.

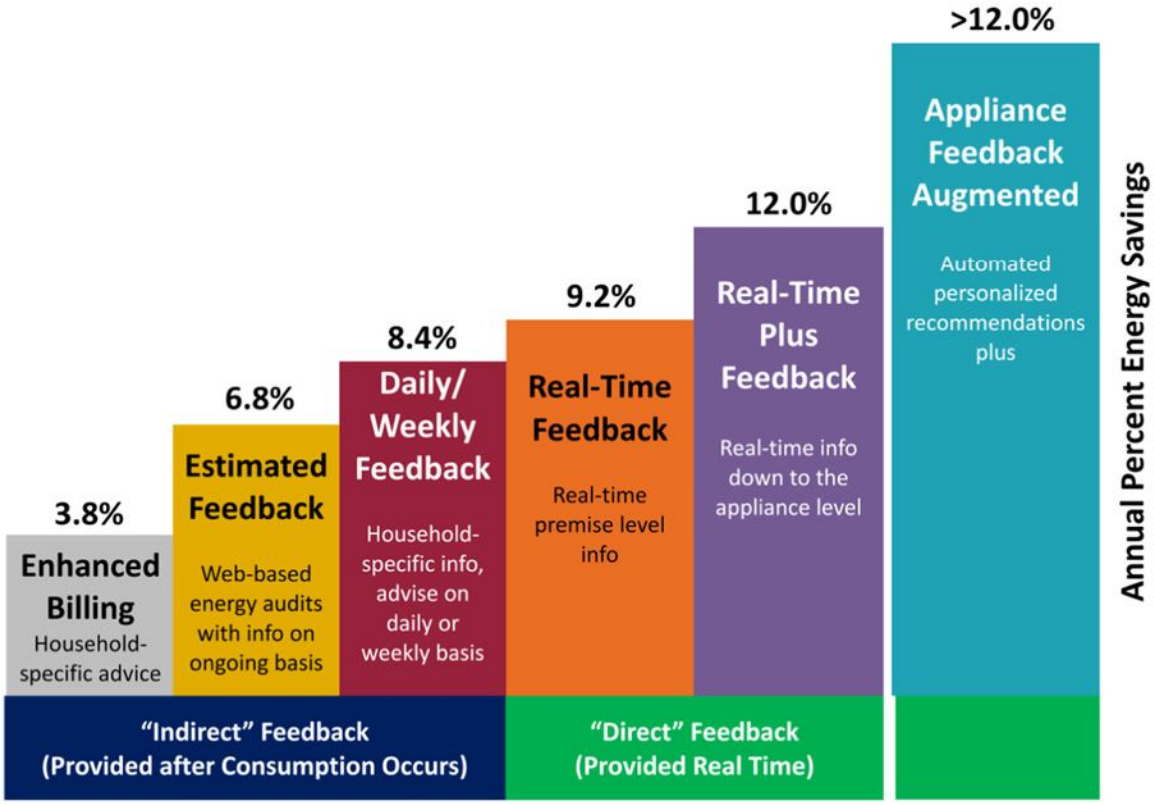

Abbildung 1: Grafik zu Einsparpotetialen je nach getroffener Aufklärungsmaßnahme aus [AGSA13]

Er kann sich oft nicht mit seinem Verbrauch identifizieren, weil dieser intransparent und durch die langen Rechnungsintervalle nicht präsent ist.
Damit der Nutzer anfängt sich selbst zu kontrollieren, muss er begreifen, dass sich sein Verhalten auf seinen Verbrauch auswirkt und er ihn auch durch die gezielte Veränderungen

seines Handelns senken kann [Fis08].
Non-intrusive load monitoring (NILM) bietet nun die Chance dem Nutzer eine regelmäßige, detaillierte Rückmeldung zu seinem Energiekonsum zu geben, Kolter und Matthew [KJ11] beschreibt NILM als die Aufgabe aus einem, für den gesamten Haushalt messenden Stromzähler, Rückschlüsse über die elektrische Last einzelner Geräte zu ziehen. Ein solches System wird dem Konsumenten helfen, verschwenderische Verhaltensweisen und Geräte zu identifizieren ohne den Haushalt mit vielen digitalen Zählern für die individuellen Geräte ausrüsten zu müssen, wie es derzeit z.B. mit sogenannten *Energiekostenmessgeräten* auf Steckerbasis praktiziert wird.

1.1 Fragestellung

Viele Systeme zur Disagregation oder zu verschiedenen Vorhersage-Aufgaben benutzen Ansätze des Maschinellen Lernens. Sie benötigen annotierte (gelabelte) Trainingsdaten, um robuste Modelle zu erzeugen. Annotieren bedeutet in diesem Zusammenhang, die Daten mit Metadaten wie etwa dem Gerätezustand zu versehen. Mehr Trainingsdaten führen in der Regel zu besseren Modellen, die Annotation ist allerdings sehr aufwändig, weil sie manuell erfolgen muss. Der Ansatzpunkt dieser Arbeit ist, das Problem der Disagregation auf ein Gerät zu reduzieren und so zu vereinfachen. Nun kann man ein robustes Modell mit nur wenigen Trainingsdaten erstellen und mit diesem dann eine große Menge an Daten schnell annotieren. Diese gelabelten Daten sollen dann wiederum als Trainingsdaten für schwierigere Aufgaben verwendet werden. Aufgabe dieser Bachelorarbeit ist es ein System zu entwickeln, welches in der Lage ist die Zustände von ausgewählten Geräten in einem disaggregierten Lastgang zu klassifizieren und so eine Segmentierung für diese Geräte zu erstellen. Dies beinhaltet sowohl die nötige Vorverarbeitung der Daten sowie eventuelle nachträgliche Formatierungen. Die eigentliche Klassifikation soll mit Künstlichen Neuronalen Netzen (KNN) stattfinden, dabei sollte eine möglichst hohe Akkurarität erreicht werden, da Systeme, die mit den resultierenden Daten trainiert werden keine Möglichkeit haben Fehler, die bereits in den Trainingsdaten sind, zu korrigieren. Außerdem soll versucht werden aus den klassifizierten Daten wieder einen Lastgang zu erstellen. Dies soll die Mächtigkeit der Segmentierung der Gerätezustände untersuchen und feststellen ob man zuverlässig den Lastgang des Geräts vorhersagen kann, wenn es seine Zustandsfolge im Voraus bekannt gibt.

1.2 Weiterer Aufbau

Im folgenden Kapitel 2 werden die verwendeten, disaggregierten Energiedatensätze beschrieben. Insbesondere werden die gemessenen Werte und die daraus berechenbaren

Werte untersucht. Kapitel 3 gibt zunächst einen Überblick über die aktuelle Forschung im Bereich Vorverarbeitung der Daten und Feature-Auswahl, anschließend werden die für diese Arbeit verwendeten Vorverarbeitungsschritte und Features erläutert. In Kapitel 4 soll schließlich die eigentliche Klassifizierung beschrieben werden, hier werden insbesondere das verwendete Netz und die Trainingsmethoden besprochen. Auch hier wird es einen kurzen Überblick über die aktuelle Forschung geben. In Kapitel 5 wird die Generierung eines Lastprofils aus dem Zustandsprofil erläutert. Es wird ein Ansatz aus der aktuellen Forschung beschrieben und auf dieses Problem adaptiert. In Kapitel 6 findet die Evaluation statt, hier werden verschiedene Klassifikationsaufgaben ausgeführt und ausgewertet. Am Schluss steht das Fazit in Kapitel 7, in dem die Ergebnisse zusammengefasst und mit ursprüngliche Fragestellung verglichen werden.

2 Datensatz

Die verwendeten Daten stammen aus dem *KIT Energy Smart Home Lab (ESHL)*. Das ESHL ist ein $60m^2$ Apartment mit intelligenten Haushaltsgeräten und soll einen zukünftigen Haushalt simulieren. Die Daten des ESHL wurden über fast 3 Jahre zwischen dem 22.08.2011 und dem 31.07.2014 aufgezeichnet, weisen aber einige Lücken auf. Für die Messungen wurden mehrere Stromzähler verwendet, bei Großverbrauchern wie etwa dem Boiler wurden dabei alle drei Phasen gemessen, sie sind durch die Anschlussnummer des Stromzählers in den Daten identifizierbar. Bei den restlichen Geräten wurde nur je eine Phase gemessen, das heißt, dass 3 unterschiedliche Geräte an einem Stromzähler angeschlossen sind. Sie sind durch die Nummer des Stromzälers in Kombination mit Nummer des verwendeten Anschlusses (Port) identifizierbar. Wie viele Phasen zur Messung eines Geräts verwendet wurden lässt sich dem Kommunikationsgateway (Controller) entnehmen. Tabelle 1 zeigt die beschriebene Zuordnung von Geräten zu Stromzähler und Anschluss.

UUID	Name	Klassifikation	Controller	Meter	Port
...-5602c0a80114	DRYER	APPLIANCE	1	3	0
...-5604c0a80114	WASHINGMACHINE	APPLIANCE	1	7	2

Tabelle 1: Beispiele für Gerätezuordnung, die Waschmaschine ist an (Smart-)Meter 7 Port 2 angeschlossen.

2.1 Gemessene Daten

Die Daten des *KIT Energy Smart Home Lab* besitzen folgendes Format (Tabelle 2).

Unixtime	UUID	Controller	Meter	Port	Spannung	Stromstärke	Wirkleistung	Zählerstand
1382291846	...-5604c0a80114	1	7	2	218.5	8.96	1956	145550
1315269463	...-5602c0a80114	1	3	0	223.8	13.16	2950	38300
1315269465	...-5602c0a80114	1	3	0	223.9	12.5	2796	38300

Tabelle 2: Format ESHL Datensatz.

Controller, Meter und Port werden, wie oben beschrieben, verwendet, um einen Datenpunkt einen eindeutigen Gerät zuzuordnen. Hier ist zu beachten, sich diese Werte immer nur gemeinsam verwendet werden dürfen, nur so ist eine eindeutige Identifizierung eines Gerätes möglich. Die Spalte mit der UUID-Zuordnung wird ignoriert, denn sie enthält ebenfalls

die Information um welches Gerät es sich handelt und ist somit redundant zu Controller, Meter und Port.
Ein Datenpunkt enthält die gemessene Spannung, die Stromstärke und die Wirkleistung. Es wird zusätzlich die aggregierte Wirkleistung (Zählerstand) gemessen, welche aber zunächst ignoriert wird, da sie sich aus der Wirkleistung berechnen lässt.
Eine Besonderheit stellt das Aufzeichnungsintervall dar. Die Stromzähler erzeugen jede Sekunde einen neuen Datenpunkt für jeden Anschluss, dieser wird aber nur gespeichert, wenn sich die Wirkleistung im Vergleich zum letzten gespeicherten Punkt um 5 Watt Wirkleistung unterscheidet. So werden insbesondere lange *Off* Phasen auf einen Datenpunkt komprimiert, es gehen aber auch kleinere Schwankungen in der Wirkleistung von sehr verbrauchsarmen Geräten verloren.

2.2 Berechnete Daten

Die Messung von Spannung und Stromstärke erlaubt es eine Vielzahl von Energiewerten zu berechnen, in diesem Abschnitt werden die Schein- und Blindleistung vorgestellt, das Kapitel zur Vorverarbeitung wird zusätzlich eine Form der Normalisierung vorstellen. Die Scheinleistung S ist das Produkt aus Spannung U und Stromstärke I und ist die gesamte, aus dem Netz gezogene Leistung:
$S = U * I$

Zur Berechnung der Blindleistung Q werden Schein- und Wirkleistung P genutzt, die Blindleistung ist dabei Leistung, die aus dem Netz gezogen wird ohne tatsächlich genutzt zu werden:
$Q = \sqrt{S^2 - P^2}$

2.3 Die Waschmaschine

Die Waschmaschine ist ein besonders interessantes Gerät für die Klassifikation, denn sie hat nicht nur einen *On-* und einen *Off-Zustand*, sondern einen Motor, der mit verschiedenen Drehzahlen laufen kann, ein Heizelement und eine Pumpe. Für die Waschmaschine existieren außerdem einige annotierte Waschgänge (Tabelle 3), die als Trainingsdaten verwendet werden können.

Die Waschgänge sind mit sechs verschiedenen Klassen Sigma 0 bis Sigma 5 annotiert, Sigma 0 ist der *Off-Zustand*. Sigma 1 ist ein normaler Schleudervorgang, hier sind Phasen mit 200-350 Watt und dazwischen kurze Pausen mit ca. 4 Watt Verbrauch typisch. Sigma

Profil	Startzeit (Unix time stamp)	Endzeit (Unix time stamp)	Dauer (Sekunden)	Datum
Profil 0	1382290611	1382296244	5633	20.10.2013
Profil 1	1382363246	1382369468	6222	21.10.2013
Profil 4	1382900551	1382905960	5409	27.10.2013
Profil 5	1383173215	1383180425	7210	30.10.2013
Profil 8	1384066710	1384072260	5550	10.11.2013

Tabelle 3: Übersicht über die annotierten Waschgänge, die als Trainingsdaten verwendet werden.

2 ist ein Heizvorgang, dieser kennzeichnet sich durch einen Verbrauch über 1700 Watt. Sigma 3 und Sigma 4 sind Pumpvorgänge und werden durch Verbrauchspitzen von wenigen Sekunden charakterisiert. Unterschieden werden sie anhand der Höhe der Spitze, ein Vorgang Sigma 4 folgt immer auf einen Vorgang Sigma 3 mit einem Abstand von 50 bis 80 Sekunden. Sigma 5 ist das Ausschleudern, es ist ebenfalls ein Schleudervorgang, der aber im Gegensatz zu Sigma 1 ohne Pausen ausgeführt wird.

Grafik 2 veranschaulicht die Wertebereiche und Lage der Klassen innerhalb eines Waschgangs. Die Klassen sind dabei wie in der Legende beschrieben farbig markiert, da die dunkelblau gefärbte Klasse 0 typischerweise einen Verbrauch von 0 Watt aufweist verläuft der Graph zu diesen Zeitpunkten auf der x-Achse. Die rot gekennzeichneten Abschnitte der Heizvorgänge sind in der ersten Hälfte des Waschgangs zu finden, hier wird das Wasser zunächst aufgewärmt und muss dann mehrfach nachgeheizt werden um die Temperatur zu halten. Die hellblau und lila markierten Abschnitte der Pumpvorgänge befinden sich in der 2. Hälfte des Waschgangs. Sie sind sehr kurz und folgen in regelmäßigen Abständen aufeinander. Der Ausschleudervorgang von Klasse 5 ist nahe dem Ende des Waschgangs zu finden, er ist gelb gekennzeichnet.

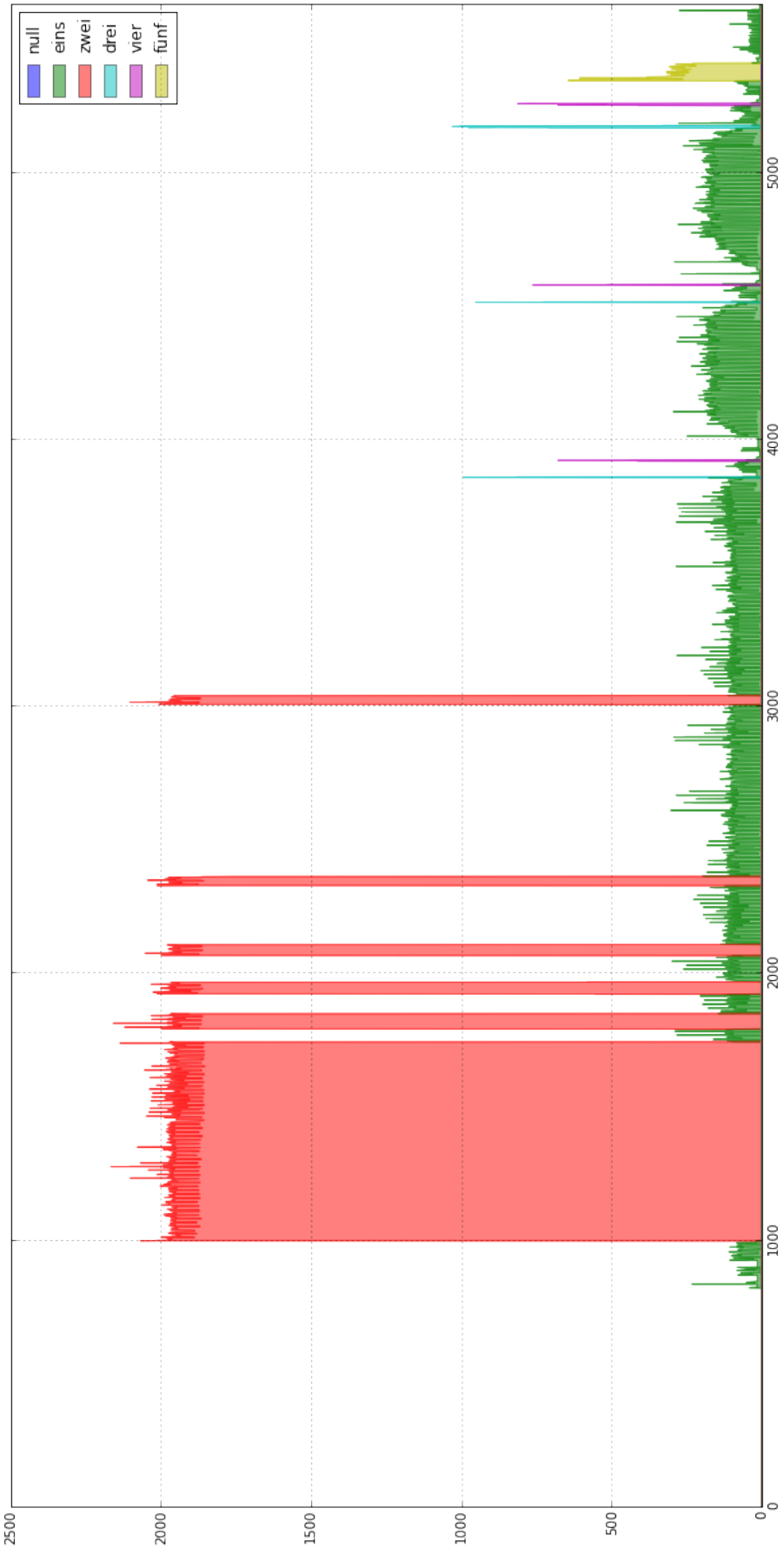

Abbildung 2: Ein typischer Waschgang, je nach Klasse wird die Wirkleistung in unterschiedlichen Farben dargestellt

3 Vorverarbeitung

Verfahren des maschinellen Lernens sind Daten-getriebene Verfahren. Die Qualität des Ergebnisses der Klassifizierung ist also in besonderem Maße abhängig von der Qualität und Quantität der Daten. Die Vorverarbeitung soll durch mögliche Filterungen und Normalisierungen die Qualität des Datensatzes sichern und durch die Auswahl und Berechnung bestimmter Merkmale, den sogenannten Features, dem von dieser Arbeit verwendeten neuronalen Netz die Unterscheidung der verschiedenen Klassen möglichst einfach machen. An der Klassifizierung beziehungsweise der Vorhersage von Energiedaten wird bereits seit über 20 Jahren geforscht und es wurden viele verschiedene Vorverarbeitungsschritte und aussagekräftige Features untersucht. Deshalb wird im Folgenden zunächst eine Übersicht über die bisherige Forschung im Bereich der Vorverarbeitung gegeben und anschließend untersucht, welche Features für die Unterscheidung verschiedener geräteinterner Zustände in Frage kommen.

3.1 Stand der Technik

Die Wirkleistung ist das ursprünglichste Merkmal um die Zustände von Geräten zu unterscheiden. Sie gibt die elektrische Leistung an, die von einem Gerät in andere Leistungen (z.B. Wärme oder Bewegung) umgewandelt werden kann. Oft wird die Wirkleistung durch gröbere Quantisierung oder Normalisierung geglättet, um die Erkennung der Klassen zu vereinfachen. [Har92] verwendet hierzu folgende Formel:

$$P_{norm}(t) = (\frac{U_{nenn}}{U(t)})^2 * P(t)$$

mit Wirkleistung $P(t)$ und Spannung $U(t)$, die Nennspannung U_{nenn} beträgt in Europa 230 Volt.

Die Normalisierung bietet den Vorteil, dass sie nicht verlustbehaftet ist und wird deshalb häufig verwendet, wenn die Spannung gemessen wurde.
Mit der Wirkleistung allein ist man jedoch nicht in der Lage Geräte mit sehr ähnlichem Verbrauch zu unterscheiden, ein Beispiel wären hier ein 2kW Motor und ein 2kW Heizelement. Diese lassen sich mit Hilfe der Blindleistung, welche nicht in andere Leistungen umgewandelt wird, unterscheiden, weil der Motor als induktiver Verbraucher mehr Blindleistung als das Heizelement aus dem Stromnetz zieht. Nachteil hier ist jedoch, dass die Blindleistung (und auch die Spannung) zusätzlich gemessen werden muss, dies kostet gegebenenfalls Geld [ZR11].
Vergleicht man jedoch eine 60W Glühbirne und einen Laptop mit einem 60W Netzteil, dann stellt man fest, dass diese sich auch unter Hinzunahme der Blindleistung kaum

unterscheiden lassen [LLC+03]. Hier hilft die Betrachtung von Micro Level Features. Dabei werden die Wellenformen und die harmonischen Komponenten des Frequenzspektrums verwendet, um Verbraucher zu unterscheiden. Laughman et al. [LLC+03] zeigen z.B., dass sich Glühbirne und Netzteil in der 3. harmonischen Komponente deutlich unterscheiden. Zusätzlich verändern viele Geräte die Wellenform des Stroms, so dass in diesem Bereich weiteres Potential für die Unterscheidung verschiedener Geräte vorliegt [LNKC10].
Micro Level Features bieten sehr gute Unterscheidungsmöglichkeiten, benötigen aber auch ein sehr hochfrequent aufgelöstes Signal, üblich sind mehrere kHz. Anderson et al. [AOB+12] bietet einen Datensatz mit 12kHz Auflösung.
Messgeräte für eine solche Auflösung sind wesentlich teurer als die häufig für Wirk- und Blindleistung verwendeten Geräte mit einer Auflösung von 1Hz und sind im Gegensatz zu Letzteren oft nicht in normalen Haushalten mit intelligentem Stromzähler vorhanden [ZR11].
Da sie in der Praxis häufiger vorhanden sind, wird in dieser Arbeit mit den Macro Level Features mit einer Auflösung von 1Hz gearbeitet.

3.2 Extrahieren der Daten

Da die Trainingsdaten leider nur die Wirkleistung zum aktuellen Zeitpunkt angeben, müssen diese erneut aus dem gesamten Datensatz extrahiert werden. Zunächst wird der Datensatz mit Meter = 7 und Port = 2 gefiltert und so die Waschmaschine ausgewählt. Anschließend werden die Startzeiten der einzelnen Waschgänge über ihre Heizphase gesucht und mit Hilfe der Länge des Waschgangs der zu filternde Abschnitt für jedes Profil bestimmt. Hat man diese nun herausgefiltert, hat man neben der Wirkleistung zusätzlich Spannung, Stromstärke und aggregierte Wirkleistung zur Verfügung. Dies ist wichtig um z.B. die normalisierte Wirkleistung zu berechnen.

3.3 Zeitreihen

Eine fehlerfreie Klassifizierung nur mit Hilfe der Wirkleistung zum aktuellen Zeitpunkt ist wegen der Überlappung der Wertebereiche der einzelnen Klassen ausgeschlossen. Dies fällt insbesondere bei Klasse 3 und 4 auf, ihre Flanken fallen oft in den selben Wertebereich, nur ihre Spitzen sind klar unterschiedlich.
Eine Möglichkeit diese mit einzubeziehen ist die zusätzliche Verwendung von vergangenen und zukünftigen Werten für die Wirkleistung. Diese Arbeit verwendet zunächst eine Umgebung von 10 Sekunden, also die Werte der Wirkleistung von $t - 10$ bis $t + 10$ für einen in Sekunden angegeben Zeitpunkt t.
Zu beachten ist die spezielle Komprimierung der Daten des ESHL. Zur einfacheren Verar-

beitung werden diese zunächst wieder auf sekündliche Daten erweitert und anschließend die Werte der Wirkleistung der vorherigen und nächsten 10 Sekunden angehängt. Dies erlaubt eine Verarbeitung auch außerhalb der zeitlichen Reihenfolge mit nur einer einzigen Zeile dieses neuen Datensatzes.
Nachteil ist die nötige Kenntnis zukünftiger Werte, dies verhindert eine sofortige Klassifizierung. Nutzt man den Wert $t + 10$ hängt der Klassifikator der Gegenwart immer mindestens 10 Sekunden nach. Es wird davon ausgegangen, dass sehr große Datensätze klassifiziert werden und deshalb durch den Verlust der letzten 10 Sekunden nur ein Bruchteil des ursprünglichen Datensatzes gegenüber dem resultierenden, annotierten Datensatz verloren geht.

3.4 Spezielle Features

Neben der Möglichkeit auf die Fähigkeit der Generalisierung des KNNs zu vertrauen und diesem einfach die Zeitreihe zu präsentieren, besteht auch die Möglichkeit ihm neben der Wirkleistung zum aktuellen Zeitpunkt noch weitere, speziell für die Waschmaschine berechnete Features als Eingabedaten zur Verfügung zu stellen.
Für die Unterscheidung der Pumpvorgänge von Klasse 3 und 4 lässt sich die Verbrauchspitze in den umgebenden 10 Sekunden berechnen um diese klar trennen zu können. Um Klasse 4 von Klasse 5 zu trennen kann der Durchschnitt der folgenden Sekunden betrachtet werden, dieser sollte bei einem Punkt der Klasse 5 deutlich höher liegen als bei Klasse 4. Die dritte schwierige Unterscheidung ist die von Klasse 1 und 5, hier hilft es das Minimum der vorher gehenden Werte zu betrachten, Klasse 1 weist hier durch die Pausen Werte von ca. 4 Watt auf, während Klasse 5 Werte über 200 Watt haben sollte. Alle anderen Klassenunterscheidungen sollten sich mit Hilfe der Wirkleistung treffen lassen.
Zusätzlich zur Verwendung von Zukunftswerten weist diese Auswahl an Features den Nachteil auf, nur für diese Waschmaschine explizit zu sein, für andere Geräte müssten gegebenenfalls neue Features gefunden werden, dies erhöht den zeitlichen Aufwand, der zur Klassifizierung eines kompletten Datensatzes nötig ist, enorm.

4 Klassifizierung

Die Aufgabe der Klassifikation übernimmt in dieser Arbeit ein künstliches Neuronales Netz (KNN). KNNs sind in Struktur und Arbeitsweise vom menschlichen Nervensystem inspiriert [KBK+11] und versuchen intelligente Entscheidungen zu treffen. Zunächst wurde nur ein einzelnes Perzeptron verwendet, welches eine Nervenzelle simulieren sollte. Das Perzeptron besitzt einen oder mehrere Eingänge, an die unterschiedlich gewichtete Signale angelegt werden, und mindestens einen Ausgang, der aktiv wird, falls der interne Schwellwert überschritten wird. Dies ist stark an die Natur angelehnt, die Eingänge entsprechen den Dendriten einer Nervenzelle und die Ausgänge den Endknöpfchen[KBK+11].
Ein einzelnes Perzeptron ist aber in seiner Mächtigkeit sehr eingeschränkt, da es nur linear separierbare Funktionen korrekt darstellen kann. Dieses Problem löst die Zusammenfassung mehrerer Perzeptren zu einem Netz, solche Netze zu trainieren wurde aber erst durch den von [Wer74] beschriebenen error-backpropagation Algorithmus möglich. Seitdem wurden verschiedene Typen von Netzen und Formen von Perzeptren entwickelt, um unterschiedlichen Aufgaben gerecht zu werden. Diese Arbeit verwendet ein mehrschichtiges Perzeptron (Multilayer Perceptron MLP) in einem vorwärts gerichteten Netz (feed forward), dabei werden für die Klassifikation sigmoide Aktivierungsfunktionen verwendet. Die verwendeten Technologien werden im folgenden Abschnitt genauer beschrieben.

4.1 Stand der Technik

[LNKC10] und [SNL06] legen die Verwendung von MLPs gegenüber Radiale-Basisfunktionen-Netzen (Radial Basefunction Net RBF) nahe, da sie mit diesen deutlich bessere Ergebnisse erzielen. Bei einem MLP handelt es sich um ein streng geschichtetes KNN. Das bedeutet, dass keine Verbindungen innerhalb der Schichten bestehen und die Eingänge einer Schicht nur mit den Ausgängen der vorherigen Schicht verbunden sind [KBK+11], siehe dazu auch Abbildung 3. Ein MLP besteht immer aus mindestens zwei Schichten, nämlich der Eingabe- und der Ausgabeschicht, die Anzahl der versteckten Schichten dazwischen ist beliebig.
Die in Abbildung 4 gezeigte Aktivierungsfunktion ist beispielhaft für die Aktivierung der Ausgänge eines Perzeptrons im Netz, die Eingaben werden mit ihren jeweiligen Gewichten multipliziert und anschließend addiert, ist der resultierende Wert größer als der Schwellwert Θ wird der Ausgang aktiv. Die Nutzung der Sinus-Funktion führt hierbei jedoch dazu, dass es keinen Sprung von 0 auf 1 gibt, sondern einen stetigen Anstieg. Für das Training mit dem error-backpropagation Algorithmus ist jedoch die Differenzierbarkeit notwendig, welche die Sprungfunktion nicht besitzt [KBK+11].
Der error-backpropagation Algorithmus berechnet den Fehler zwischen gewünschter und tatsächlicher Ausgabe des Netzes im Training und gibt anschließend das Fehlersignal

rückwärts durch das Netz, dabei passen die Perzeptren die Eingabegewichte an, um den Fehler zu minimieren [KBK+11]. So wird der Trainingsdatensatz mehrfach durchlaufen und der Fehler immer weiter verringert, maßgebender Faktor für die Verringerung des Fehlers ist die Lernrate. Sie gibt an, wie stark die Gewichte in jedem Schritt angepasst werden. Zusätzlich verwenden wir das Momentterm-Verfahren aus [RHW88], welches die Lernrate erhöht, solange sich der Fehler verringert und die Lernrate gemindert wenn der Algorithmus um ein Minimum oszilliert und sich der Fehler nicht mehr verringert. Dies garantiert jedoch nicht, dass das globale Minimum für den gemachten Fehler gefunden wird.
[Pin99] zeigt, dass man jede stetige Funktion mit einem MLP mit nur einer versteckten Schicht mit beliebig geringem Fehler approximieren kann. Das bedeutet aber nur, dass eine Netzstruktur gefunden werden kann, mit der jede beliebige Fehlerschranke unterschritten wird. Laut [KBK+11] ist es aber in der Praxis häufig von Vorteil mehrere versteckte Schichten einzubauen um schneller an eine gute Approximation zu kommen.
Die Ausgabeschicht wird im 1-zu-n Schema modelliert. Das bedeutet, dass die sechs Klassen durch sechs Knoten in der Ausgabeschicht modelliert werden, als Ausgabe wird dann der Knoten mit der höchsten Ausgabe gewählt. Im Optimalfall hat einer der sechs Knoten als Ausgabe den Wert 1 und die anderen Knoten den Wert 0, durch die Verwendung der Sinus-Funktion ist dies jedoch nicht immer der Fall.

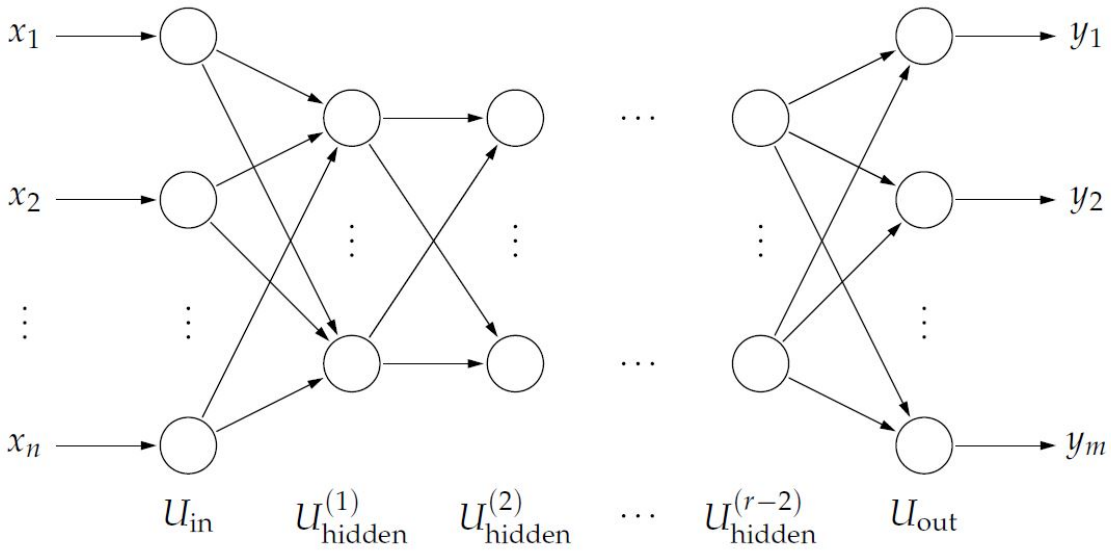

Abbildung 3: Grafik zum allgemeinen Aufbau eines aus r Schichten bestehenden MLPs aus [KBK+11], x_a bezeichnen die Eingaben, y_b die Ausgaben und U_c die verschiedenen Schichten des Netzes

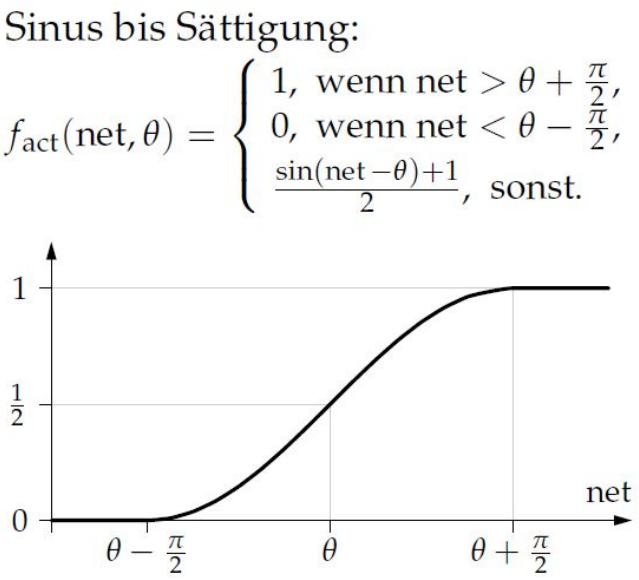

Abbildung 4: Beispiel einer Aktivierungsfunktion aus [KBK+11], Θ ist der Schwellwert für die Aktivierung des Ausgangs und net die Eingaben multipliziert mit den Eingabegewichten

4.2 Weka

Weka ist ein bekanntes Framework für maschinelles Lernen. Die in Java geschriebene Software wurde von der *University of Waikato, New Zealand* entwickelt und unter *GNU General Public License* veröffentlicht[1].
Weka bietet viele, für diese Arbeit wichtige Funktionen an. Insbesondere die integrierten Filter und die automatische Interpretation von Instanzen zur Klassifikation ersparen viel Arbeit. Weka implementiert eine Vielzahl von Klassifikatoren, darunter auch das Multilayer Perceptron (MLP). Außerdem bietet es einige Methoden zur Evaluation der erstellten Modelle, wie zum Beispiel die k-fold-Evaluierung. Dabei ist es jedoch nicht auf die massiv parallele Verarbeitung moderner Prozessoren ausgelegt und skaliert nicht horizontal.
In dieser Arbeit wird Wekas Implementierung des MLP benutzt. Der Algorithmus bietet sich wegen der einfachen Verwendung bei gleichzeitiger Möglichkeit zu parametrieren an. Die bereits implementierten Funktionen zur Filterung und Evaluation verringern den Implementierungsaufwand für die hier erstellte Software enorm.

4.3 Parameter

Ein MLP hat einige wichtige Parameter, die zur Genauigkeit der Klassifikation beitragen. Dieses Kapitel soll aber lediglich auf die Parameter hinweisen, die besten Werte für jeden

[1] http://www.cs.waikato.ac.nz/ml/weka/ Abgerufen am 23.07.2015

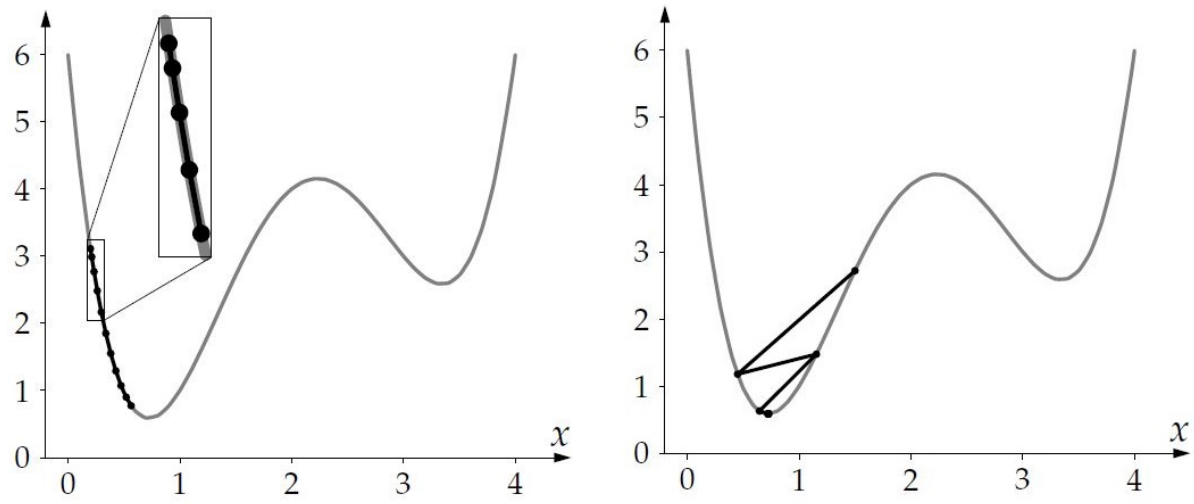

Abbildung 5: Beispiel Momentterm-Verfahren aus [KBK+11], links sieht man die Erhöhung einer für den Abschnitt zu niedrig gewählten Lernrate, rechts die Verringerung der Lernrate wegen Oszillationen

Parameter werden in Kapitel 6 ermittelt. Mit dem ersten wichtigen Parameter hat sich bereits das vorherige Kapitel beschäftigt: Die Features. Diese spielen in der Klassifikation eine zentrale Rolle und wurden daher gesondert behandelt.
Die Anzahl der Epochen N gibt an, wie oft das Netz die Trainingsinstanzen während des Trainings durchläuft. Bei jedem Durchlauf wird der vom Netz verursachte Fehler mit dem error-backpropagation Algorithmus minimiert. Ist die Anzahl der Epochen zu gering, kann der Fehler nicht ausreichend minimiert werden, siehe dazu 5 das Beispiel auf der linken Seite, 11 Perioden reichen hier nicht, um das Minimum zu erreichen.
Die Geschwindigkeit, mit der die Gewichte innerhalb des Netzes angepasst und somit der Fehler minimiert wird, wird durch die Lernrate L bestimmt. Eine höhere Lernrate führt dazu, dass das Netz schneller in Richtung des lokalen Minimums konvergiert, kann aber auch dazu führen, dass es das Minimum überspringt und es nie genau erreicht. Dies kann aber durchaus erwünscht sein, wenn durch die hohe Konvergenzgeschwindigkeit nicht optimale Minima $\tilde{A}\frac{1}{4}$bersprungen werden. Außerdem benötigt eine niedrige Lernrate im Allgemeinen mehr Epochen, um das lokale Minimum zu erreichen.
Das Momentum M ist hilfreich, um den Trade-off zwischen Laufzeit des Trainings und Güte des Ergebnisses zu verbessern. Eine niedrige Lernrate verbessert im Allgemeinen die Annäherung an das lokale Minimum, verlängert aber auch die Trainingszeit, weil sie mehr Schritte zur ausreichenden Minimierung des Fehlers benötigt. Durch die Erhöhung und Verringerung der Lernrate kann das Momentum helfen eine gute Lösung in weniger Epochen zu erzielen.

Auch die Struktur des Netzes H kann einen positiven Einfluss auf die Klassifikation haben. Dabei ist die Anzahl der Knoten in Eingabe- und Ausgabeschicht (Input-/Outputlayer) festgeschrieben, die Eingabeschicht hat so viele Knoten wie es Features gibt und die Ausgabeschicht so viele Knoten wie es Klassen gibt. Die versteckten Schichten (Hiddenlayer) dazwischen sind aber variabel, hier ist es üblich mit einer Schicht zu beginnen und diesem eine Knotenanzahl zwischen den Knotenzahlen der Eingabe- und Ausgabeschicht zuzuweisen. Danach kann man testen, ob sich das Hinzufügen weiterer verdeckter Schichten positiv auf die Fehlerrate auswirkt [KBK+11].

5 Generierung

Bei der Generierung lautet die Aufgabe, aus den Klassenzuordnungen eines Zeitabschnitts wieder einen Lastgang zu erstellen. Es ist also die gegenteilige Aufgabe zur Klassifikation. Statt aus einem Lastgang ein Zustandsprofil zu erstellen wird aus einem Zustandsprofil ein Lastgang erstellt. Zusammen mit der Klassifikation und einer approximierten Zustandsfolge können so Vorhersagen über den Verbrauch der Geräte getroffen werden und man ist in der Lage die Nutzung der Geräte planen.
Die Planung der Gerätenutzung ist insbesondere in Verbindung mit variablen Energiepreisen wie zum Beispiel dem *Dynamic MeRegio tarif*[2] interessant. Diese Form der Approximation eines Lastgangs greift im Gegensatz zu vorherigen Arbeiten auf die Zustandsfolge zu. Dies sollte zu einer besseren Lösung führen, bringt aber auch das Problem mit sich, dass die Zustandsfolge im Allgemeinen nicht bekannt ist, weil auch die intelligente Waschmaschine nicht mitteilt, welche Zustandsfolge sie einnehmen wird. Das ist aber eine wichtige Information, die dadurch nicht einfach zu erhalten ist. Wir nehmen deshalb an, dass wir bereits eine korrekte Zustandsfolge besitzen. Woher diese kommt lässt diese Arbeit offen, sie könnte z.B. aus einem Algorithmus stammen, der sie vorhersagt oder einer hypothetischen Waschmaschine, die mitteilt, welche Zustandsfolge sie annehmen wird. Dazu reicht es theoretisch, wenn sie mitteilt welches Programm gestartet werden soll. Die Zustandsfolge kann dann in einem einmaligen Test mit der Klassifikation erstellt werden.

5.1 Stand der Technik

Das Konzept des L/H-converter aus [HIT08] ist ein sehr allgemeines Konzept zum simulieren eines hoch aufgelösten Signals aus einem niedrig aufgelösten. Das Zustandsprofil stellt dabei das niedrig aufgelöste Signal dar, der Wertebereich umfasst die Werte 0,1,2,3,4,5, während die Wirkleistung als hoch aufgelöstes Signal der kompletten Wertebereich von 0 bis 2200 Watt abdeckt. [HIT08] gibt für den L/H-converter folgende Formel an.

$$x_{high} = R^{L/H} x_{low}$$

$R^{L/H}$ stellt dabei die Funktion zur Simulation dar, diese muss in der Implementierung spezifiziert werden. Abbildung 6 zeigt noch einmal die Einordnung des L/H-converter in einem System mit verschiedenen Auflösungen, wobei das Low Resolved System in unserem Fall das Zustandsprofil und das High Resolved System den Lastgang repräsentiert.

[2] http://ahk.de/fileadmin/ahk_norwegen/Dokumente/Presentasjoner/SmardGrid2012/Session_2/Frey_-_EnBW.pdf Abgerufen am 17.06.2015

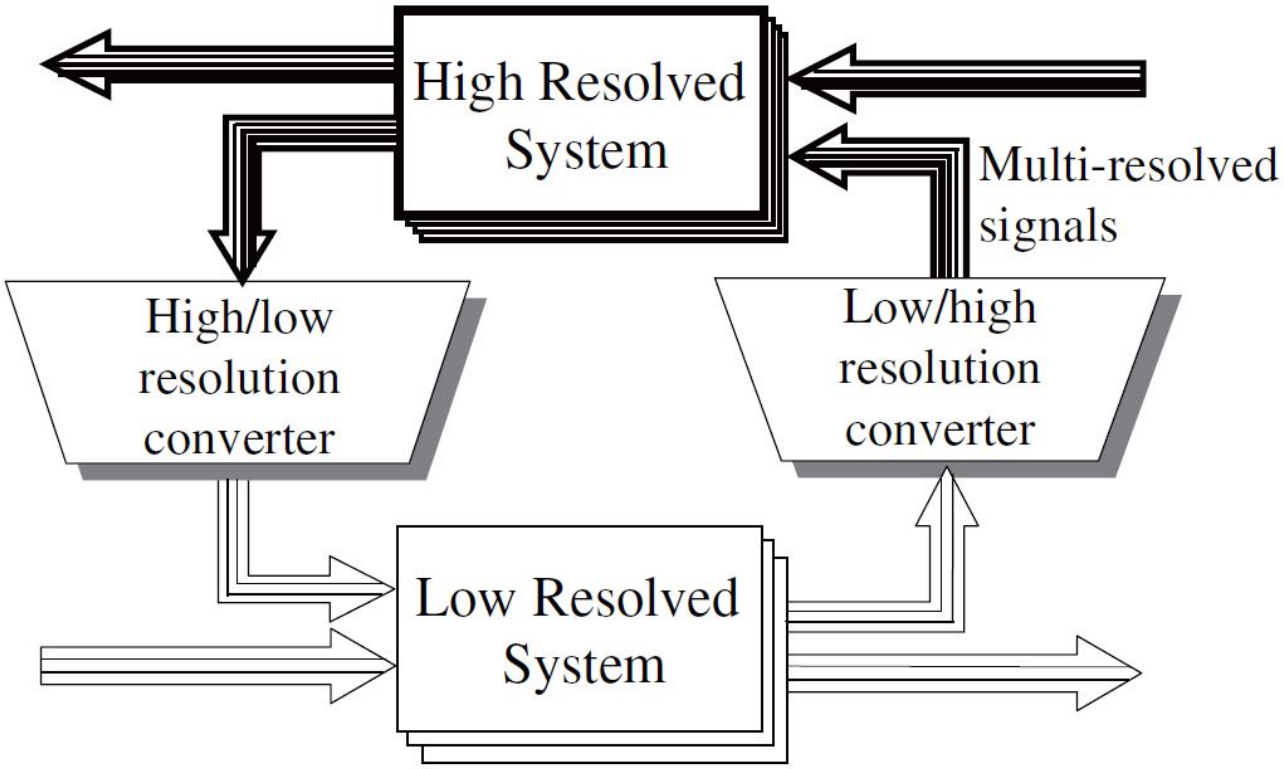

Abbildung 6: Schematische Darstellung eines Systems mit verschiedenen Auflösungen und den Konvertern, die die Auflösung im Informationsfluss anpassen aus [HIT08].

5.2 Vorgehensweise

Die Erstellung eines Lastgangs nur aufgrund des klassifizierten Zustands der Maschine ist nicht besonders sinnvoll, denn das neuronale Netz kann nach dem Training nur die Mittelwerte für die Wirkleistung der einzelnen Klassen angeben und es entsteht ein unrealistischer, extrem glatter Lastgang. Die Verwendung einer Zeitreihe als zusätzliches Feature schafft einen wesentlich realistischeren Lastgang. Man kann sich hier zu Nutze machen, dass man vor Beginn des Programms implizit annehmen kann, dass sich die Maschine im *Off-Zustand* befindet. Man nimmt also für den ersten zu simulierenden Wert des Waschgangs die vorherige Wirkleistung als 0 an und kann dann iterativ immer einen neuen Wert berechnen. Für die Simulation stehen dem Netz die Werte der Wirkleistung von $t-10$ bis $t-1$ und die Klasse von $t+0$ zur Verfügung.

$$t+0 = F(t-1, ..., t-10, K(t+0))$$

Beispiel:

$$1994 \stackrel{!}{=} F(82, 82, 82, 18, 18, 18, 18, 18, 18, 2068, 2)$$

Die Funktion F ist die Simulationsfunktion, in diesem Falls also das neuronale Netz. $K(t+0)$ ist die Klassifikation des Zeitpunktes t+0. Die Rekursion von F endet, wenn der Beginn des Waschgangs erreicht ist, dann gilt
$t+0 = F(t-1, ..., t-10, K(t+0)) = F(0, ..., 0, K(t+0))$ $für$ $t = 0$. Bezogen auf die in 5.1 gegebene Formel für die Simulation gilt: $t+0 \mathrel{\widehat{=}} x_{high}$, $K(t+0) \mathrel{\widehat{=}} x_{low}$ und $F \mathrel{\widehat{=}} R^{L/H}$.

6 Evaluation

Die Evaluation soll klären, wie genau die neuronalen Netze die Aufgaben der Klassifizierung und Generierung ausführen können. Um eine möglichst gute Lösung zu bekommen wird in der Evaluation untersucht, welche Features und Parameter sich als vorteilhaft erweisen und wann Features und Verarbeitungsschritte überflüssig sind. L, M, N und H bezeichnen die in Kapitel 4 erläuterten Features. L ist die Lernrate, M das Momentum, N die Anzahl der Trainingsepochen und H beschreibt die Anzahl der Knoten in der verdeckten Schicht. Hat ein Netz mehrere verdeckte Schichten, werden diese durch Kommata getrennt.

6.1 Klassifizierung

Bei der Klassifizierung handelt es sich um eine Aufgabe bei der ein Lastgang in ein Zustandsprofil umgewandelt werden soll. Eingabe sind Merkmale des Lastgangs und Ausgabe ist eine Klassenzuordnung für jeden Datenpunkt.
Als Maß für die Güte der Klassifizierung dient die Genauigkeit und damit die Frage, bei wie vielen Datenpunkten (Instanzen) die richtige Klasse klassifiziert wurde. Eine Gewichtung der Fehlertypen findet zunächst nicht statt.

Zu Beginn beschäftigen wir uns mit der Auswahl des richtigen Zeitfensters. Es zeigt sich schnell, dass selbst dieser begrenzte Featureraum bereits mehrere lokale Optima aufweist, siehe dazu Tabelle 4. Die besten Ergebnisse liefert der Zeitraum von 3 Sekunden um den Zeitpunkt t, für diesen werden weitere Tests durchgeführt.

Zunächst variieren wir die Epochen N. Festzustellen ist, dass eine Erhöhung der Epochen über 1000 keine Verbesserung der Genauigkeit mehr mit sich bringt. Die Normalisierung der Wirkleistung verschlechtert das Ergebnis. Dies ist vermutlich auf die vorherige Quantisierung der Daten zurückzuführen, diese sorgt dafür, dass die Normalisierung die Werte eher streut statt glättet.
Das Hinzufügen eines zweiten Hiddenlayer führt zu einer Verbesserung der Klassifikationsgenauigkeit. Die Veränderung von Lernrate und Momentum, beziehungsweise das Hinzufügen eines dritten Hiddenlayer bringen jedoch keine weiteren Verbesserungen. Die Ergebnisse für die Tests mit verschiedenen Parametern sind in Tabelle 6 zusammengefasst.

Die speziell für die Waschmaschine gesuchten Features schneiden schlechter als die Zeitreihen ab (Tabelle 5). Dies soll kein großer Rückschlag für diese Arbeit sein, die Zeitreihen sind leichter zu berechnen und besitzen allgemeineren Charakter. Letztere Eigenschaft

scheint nicht nur praktisch, sondern auch der Klassifikation dienlich zu sein.

Ein genauerer Blick auf die Konfusionsmatrix 7 des Netzes mit dem besten Ergebnis verrät, dass die meisten Verwechselungen zwischen Klasse 3, 4 und 5 auftreten. Das Netz hat hier offenbar Probleme die Unterschiede in den Verbrauchsspitzen zu erkennen. Diese Art von Fehlern umfasst 62 Instanzen und somit knapp 2/3 der Fehlklassifikationen.
Die 22 Fehler bei denen ein *Off-Zustand* als Klasse 1 klassifiziert wurden sind im Prinzip *off-by-one-Fehler*. Das Netz erkennt nicht die semantische Bedeutung des Features t+0 und klassifiziert auch solche Zeitpunkte als *On-Zustand*, bei denen die Werte t+1,2,3 bzw. t-1,2,3 größer Null sind, obwohl t+0 gleich Null ist. Die restlichen Fehlklassifikationen sind ähnlich zu betrachten, insgesamt machen die *off-by-one-Fehler* mit 33 Instanzen gut 1/3 der Fehlklassifikationen aus.

Features	Korrekt klassifizierte Instanzen	Inkorrekt klassifizierte Instanzen	Genauigkeit
t-10 bis t+10	29263	740	97.5336%
t-9 bis t+9	29551	452	98.4935%
t-8 bis t+8	29652	351	98.8301%
t-7 bis t+7	29612	391	98.6968%
t-6 bis t+6	29372	631	97.8969%
t-5 bis t+5	29249	754	97.4869%
t-4 bis t+4	29802	201	99.3301%
t-3 bis t+3	29876	127	99.5767%
t-2 bis t+2	29093	910	96.967%
t-1 bis t+1	26526	3477	88.4112%
t+0	29508	495	98.3502%

Tabelle 4: Genauigkeit des Netzes bezüglich verschiedener Längen von Zeitreihen, Netz mit Weka Standardkonfiguration (-L 0.3 -M 0.2 -N 500 -H (# Features+# Klassen)/2)

L	M	N	H	Korrekt klassifizierte Instanzen	Inkorrekt klassifizierte Instanzen	Genauigkeit
0.3	0.2	500	5	29535	486	98.3811%
0.3	0.2	1000	5	29853	168	99.4404%
0.3	0.2	1500	5	29535	486	98.3811%

Tabelle 5: Genauigkeit des Netzes bezüglich verschiedener Parameter des Netzes, es werden die speziellen Features verwendet

L	M	N	H	Normalisiert	Korrekt klassifizierte Instanzen	Inkorrekt klassifizierte Instanzen	Genauigkeit
0.3	0.2	500	6	nein	29876	127	99.5767%
0.3	0.2	500	6	ja	29870	133	99.5567%
0.3	0.2	1000	6	nein	29888	115	99.6167%
0.3	0.2	1000	6	ja	29882	121	99.5967%
0.3	0.2	1500	6	nein	29883	120	99.6%
0.3	0.2	1500	6	ja	29870	133	99.5567%
0.3	0.2	1000	13,6	nein	29908	95	99.6834%
0.2	0.2	1000	13,6	nein	29755	248	99.1734%
0.4	0.2	1000	13,6	nein	29543	460	98.4668%
0.3	0.1	1000	13,6	nein	29886	117	99.61%
0.3	0.3	1000	13,6	nein	29540	463	98.4568%
0.3	0.2	1000	7,13,6	nein	28368	1635	94.5505%

Tabelle 6: Genauigkeit des Netzes bezüglich verschiedener Parameter des Netzes, als Zeitreihe wird der Zeitraum t-3 bis t+3 verwendet, zusätzlich wird gegen die normalisierte Wirkleistung getestet

a	b	c	d	e	f	← classified as
3670	22	0	0	0	0	a = 0
0	20984	0	0	0	2	b = 1
0	2	4762	0	0	0	c = 2
0	3	0	39	15	0	d = 3
0	2	0	18	52	12	e = 4
0	2	0	0	17	401	f = 5

Tabelle 7: Konfusionsmatrix des Netzes mit den Features t-3 bis t+3 und den Parametern -L 0.3 -M 0.2 -N 1000 -H 13,6

6.2 Generierung

Die Aufgabe der Generierung erstellt aus einem Zustandsprofil wieder einen Lastgang, dazu darf die Klasse des zu prognostizierenden Datenpunktes und die ihm vorhergegangenen Datenpunkte verwendet werden. Als Anfangswert kann man Werte des *Off-Zustands* annehmen und dann iterativ neue Datenpunkte berechnen beziehungsweise prognostizieren. Für die Evaluation werden die vorhergegangenen Datenpunkte als gegeben betrachtet.

Zu Beginn steht auch hier wieder die Auswahl des richtigen Zeitfensters. Tabelle 8 zeigt die Ergebnisse bezüglich der Standardkonfiguration von Weka, die Verwendung der letzten 7 Sekunden stellt das Optimum im Suchbereich dar, für dieses werden weitere Tests durchgeführt.

Bei der Variierung der Parameter wird mit der Anzahl der Epochen und der Breite des Hiddenlayers begonnen. Es zeigt sich, dass ein schmaler Hiddenlayer besser die Charakteristik des Lastgangs erkennen kann, als ein breiter. Beim anschließenden variieren von Lernrate und Momentum zeigt sich außerdem, dass für beide Parameter niedrige Werte das Ergebnis verbessern, dies gilt insbesondere für die Kombination von niedriger Lernrate mit niedrigem Momentum. Die Ergebinsse finden Sie in Tablle 9. Danach werden noch einmal die Epochen erhöht um festzustellen, ob sich dadurch das Ergebnis noch weiter verbessern lässt, dies ist jedoch schon nach 2000 Epochen nicht mehr der Fall.

Abbildung 7 zeigt in blau einen echten Waschgang und in grün einen, mit der Kenntnis vergangener Werte, Vorhergesagten. Gut zu erkennen ist die Tendenz des Netzes den Lastgang zu glätten, Spitzen und Schluchten im Verbrauch weichen im prognostizierten Verlauf weniger stark vom Mittelwert der Klasse ab.

Features	Root Mean Squared Error	Root Relative Squared Error
t-10 bis t-1	87.7099	12.8235%
t-9 bis t-1	91.2275	13.3378%
t-8 bis t-1	91.0262	13.3084%
t-7 bis t-1	84.3952	12.3389%
t-6 bis t-1	99.7641	14.5859%
t-5 bis t-1	94.7697	13.8557%
t-4 bis t-1	93.1525	13.6193%
t-3 bis t-1	84.6953	12.3828%
t-2 bis t-1	94.5766	13.8275%
t-1	91.6551	13.4004%

Tabelle 8: Genauigkeit der Generierung bezüglich verschiedener Längen von Zeitreihen, die Klasse des zu klassifizierenden Datenpunktes ist ebenfalls gegeben. Netz mit Weka Standardkonfiguration (-L 0.3 -M 0.2 -N 500 -H (# Features+# Klassen)/2)

L	M	N	H	Root Mean Squared Error	Root Relative Squared Error
0.3	0.2	500	7	84.3952	12.3389%
0.3	0.2	500	14	88.0205	12.869%
0.3	0.2	1000	7	81.7189	11.9476%
0.3	0.2	1000	14	88.9724	13.0081%
0.3	0.2	1500	7	81.0541	11.8504%
0.3	0.3	1500	7	83.2135	12.1662%
0.3	0.1	1500	7	79.5297	11.6276%
0.1	0.2	1500	7	72.1011	10.5415%
0.1	0.1	1500	7	71.9262	10.5159%
0.05	0.1	1500	7	68.3153	9.988%
0.01	0.1	1500	7	66.8258	9.7702%
0.005	0.1	1500	7	66.6262	9.741%
0.3	0.2	2000	7	81.096	11.6639%
0.1	0.2	2000	7	72.1756	10.5524%
0.01	0.1	2000	7	66.9218	9.7843%
0.01	0.1	2500	7	67.0246	9.7993%
0.01	0.1	3000	7	67.113	9.8122%
0.01	0.1	3000	14	66.8768	9.7777%
0.01	0.1	4000	14	66.9756	9.7921%

Tabelle 9: Genauigkeit der Generierung bezüglich verschiedener Parameter des Netzes, als Zeitreihe wird der Zeitraum t-7 bis t-1 und die Klasse des Datenpunktes verwendet

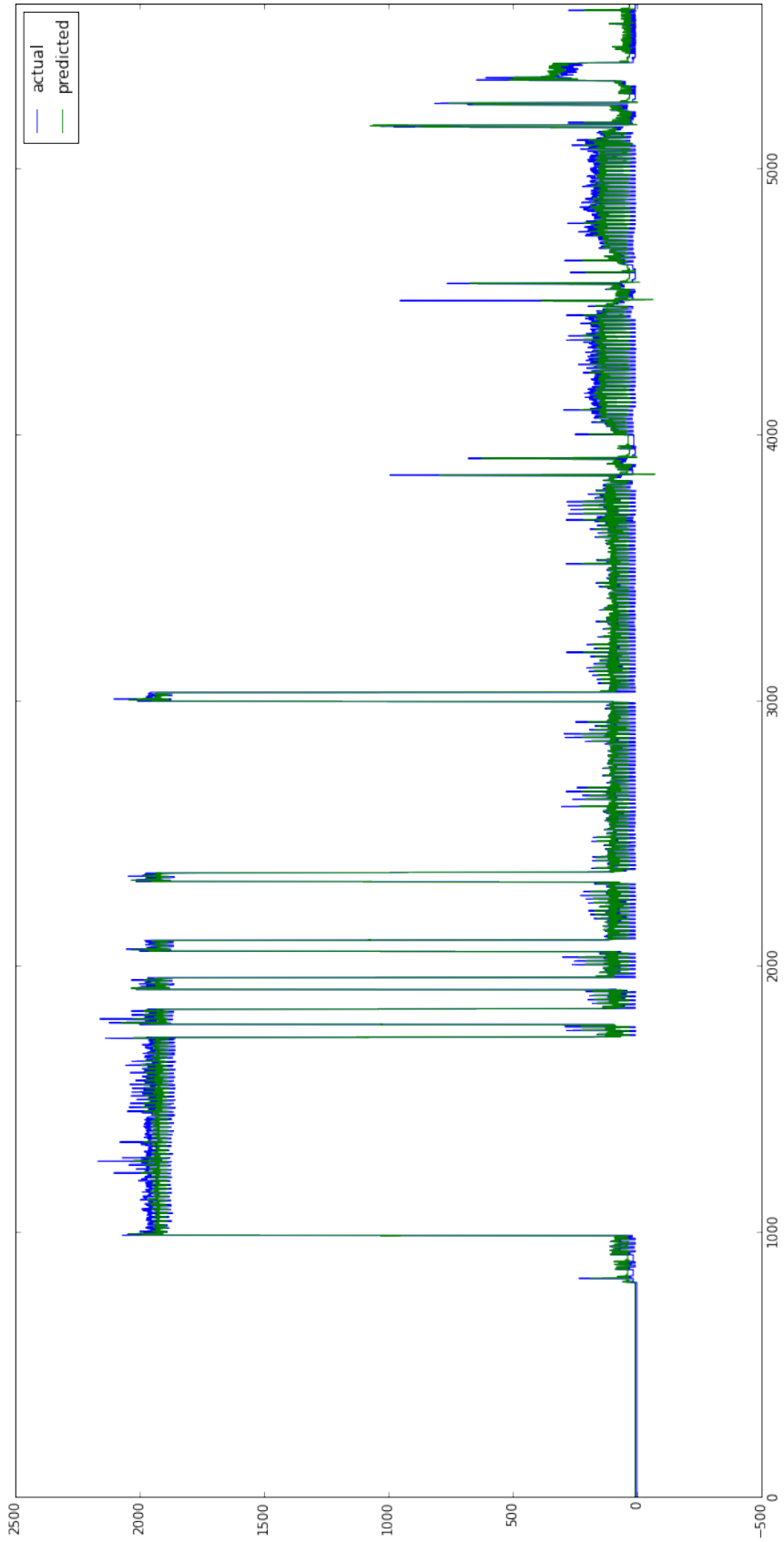

Abbildung 7: Ein typischer Waschgang, blau zeigt den tatsächlichen Lastgang, grün den von Netz vorhergesagten. Als Grundlage für die Vorhersage dienen die Werte t-7 bis t-1 und die Klasse von t+0, das Netz wurde mit den Parametern -L 0.01 -M 0.1 -N 4000 -H 14 erstellt

7 Fazit

A Anhang

Literatur

[AGSA13] Armel, K Carrie, Abhay Gupta, Gireesh Shrimali und Adrian Albert: *Is disaggregation the holy grail of energy efficiency? The case of electricity.* Energy Policy, 52:213–234, 2013.

[AOB+12] Anderson, Kyle, Adrian Ocneanu, Diego Benitez, Derrick Carlson, Anthony Rowe und Mario Berges: *BLUED: A fully labeled public dataset for event-based non-intrusive load monitoring research.* In: *Proceedings of the 2nd KDD workshop on data mining applications in sustainability (SustKDD)*, Seiten 1–5, 2012.

[Fis08] Fischer, Corinna: *Feedback on household electricity consumption: a tool for saving energy?* Energy efficiency, 1(1):79–104, 2008.

[Har92] Hart, George William: *Nonintrusive appliance load monitoring.* Proceedings of the IEEE, 80(12):1870–1891, 1992.

[HIT08] Hara, Shinji, Jun-ichi Imura und Koji Tsumura: *Multi-resolved dynamical system theory for large scale complex systems.* In: *SICE Annual Conference, 2008*, Seiten 3115–3118. IEEE, 2008.

[KBK+11] Kruse, R, C Borgelt, F Klawonn, C Moewes, G Russ und M Steinbrecher: *Computational Intelligence: Eine methodische Einführung in Künstliche Neuronale Netze.* Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze Vieweg+ Teubner, Wiesbaden, 2011.

[KJ11] Kolter, J Zico und Matthew J Johnson: *REDD: A public data set for energy disaggregation research.* In: *Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA*, Band 25, Seiten 59–62. Citeseer, 2011.

[LLC+03] Laughman, Christopher, Kwangduk Lee, Robert Cox, Steven Shaw, Steven Leeb, Les Norford und Peter Armstrong: *Power signature analysis.* Power and Energy Magazine, IEEE, 1(2):56–63, 2003.

[LNKC10] Liang, Jian, Simon KK Ng, Gail Kendall und John WM Cheng: *Load signature study - Part I: Basic concept, structure, and methodology.* Power Delivery, IEEE Transactions on, 25(2):551–560, 2010.

[Pin99] Pinkus, Allan: *Approximation theory of the MLP model in neural networks.* Acta Numerica, 8:143–195, 1999.

[RHW88] Rumelhart, David E, Geoffrey E Hinton und Ronald J Williams: *Learning representations by back-propagating errors.* Cognitive modeling, 5:3, 1988.

[SNL06] Srinivasan, D, WS Ng und AC Liew: *Neural-network-based signature recognition for harmonic source identification.* Power Delivery, IEEE Transactions on, 21(1):398–405, 2006.

[VBG08] Vandenbergh, Michael P, Jack Barkenbus und Jonathan M Gilligan: *Individual carbon emissions: The low-hanging fruit.* UCLA Law Review, 55:08–36, 2008.

[Wer74] Werbos, Paul: *Beyond regression: New tools for prediction and analysis in the behavioral sciences.* Dissertation, Havard University, 1974.

[ZR11] Zeifman, Michael und Kurt Roth: *Nonintrusive appliance load monitoring: Review and outlook.* IEEE Transactions on Consumer Electronics, Seiten 76–84, 2011.