HAW HAMBURG

BACHELORTHESIS
Tobias Rawald
[redacted]

Zeitreihenvorhersage mit Neuronalen Netzen

FAKULTÄT DESIGN, MEDIEN UND INFORMATION
Department Medientechnik

FACULTY OF DESIGN, MEDIA AND INFORMATION
Department Media Technology

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Tobias Rawald
[redacted]

Zeitreihenvorhersage mit Neuronalen Netzen

Time series forecasting with neural networks

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Media Systems*
am Department Medientechnik
der Fakultät Design, Medien und Information
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuende Prüferin: Prof. Dr. Larissa Putzar
Zweitgutachterin: Prof. Dr.-Ing. Sabine Schumann

Eingereicht am: 28. Juni 2022

Stichworte

Zeitreihenvorhersage, Neuronale Netze, FFNN, LSTM, Autoencoder, CNN, Energieprognose

Kurzzusammenfassung

In dieser Arbeit wird untersucht, inwiefern Zeitreihen mithilfe von neuronalen Netzen vorhergesagt werden können. Dafür werden die Netz-Architekturen FFNN, LSTM, Autoencoder und CNN verwendet. Die Versuchsdurchführung ist so konzipiert, dass ein Vergleich zwischen den Architekturen anhand von drei Experimenten umgesetzt wird, welche sich in ihrer Komplexität unterscheiden. In den ersten beiden Versuchen werden Modelle darauf trainiert, eine Zahlenreihe und die Sinuskurve fortzuführen, wobei das LSTM-Modell und der Autoencoder die niedrigsten Abweichungen zu den Testdaten aufweisen. Der dritte Versuch basiert auf öffentlichen Energiedaten von Deutschland. Anhand von stündlichen und täglichen Solarprognosen wird festgestellt, dass LSTM-CNN-Modelle die größtenteils besten Ergebnisse erzielen. In der Vorhersage von Solardaten ist ein Trend zur steigenden Energieerzeugung durch Solarstrom zu erkennen.

Keywords

Time Series Forecasting, Neural Networks, FFNN, LSTM, Autoencoder, CNN, Energy Prediction

Abstract

This study analyses whether time series can be predicted by using the neural network architectures FFNN, LSTM, Autoencoder and CNN. The comparison was accomplished by three experiments using different levels of complexity. In the first two experiments, models are trained to continue a series of numbers and the sine curve, with the LSTM model and the Autoencoder archieving the lowest deviations from the test data. The third experiment is based on public energy data from Germany. Using hourly and daily solar forecasts, LSTM-CNN models achieve mostly the best results. It was possible to determine a trend towards an increased energy production by solar power in the future.

Inhaltsverzeichnis

1 Einleitung

Bereits seit Jahren gibt es Berichte darüber, dass sich die Erde zunehmend erwärmt, mit gravierenden Auswirkungen auf unser Leben. Die Sommer werden heißer, das Wetter extremer, das Polareis schmilzt, der Meeresspiegel steigt an, Tierspezies sterben aus, Menschen verlieren ihr Hab und Gut und am Ende sind wir selbst dafür verantwortlich. [28, 33] Ein Beispiel extremer Wetterverhältnisse ist die Hitzewelle an der pazifischen Küste in der Region um Vancouver, Seattle, Portland der USA und Kanada im Juni 2021. Die gemessenen Temperaturen waren so hoch, dass diese nicht mal in Relation zu historischen Messungen gesetzt werden konnten. Statistisch gesehen ist dies ein Ereignis, welches einmal in 1000 Jahren eintritt. Mit Blick auf den globalen Trend sind diese Ereignisse bald keine Seltenheit mehr, weltweit steigt der Peak der durchschnittlichen wärmsten Tage im Jahr immer weiter an. [26]

Mittlerweile gehören Klimagipfel wie die UN-Klimakonferenz seit 1995 zu alljährlichen Veranstaltungen rund um den Globus, um eine wachsende Klimakrise abzuwenden und den Generationen nach uns ein gleichwertiges Leben mit Zugriff auf gleiche Ressourcen zu ermöglichen. Während große Industriestaaten- oder Konzerne eher langsam auf diese besorgniserregende Lage reagieren, gibt es aus der Bevölkerung initiierte Bewegungen, wie die von der Schwedin Greta Thunberg „Fridays for Future“, um auf die Klimasituation und die Folgen für künftige Generationen aufmerksam zu machen. Richtete sich anfangs dieser Aufruf noch an Schüler:innen und Student:innen, welche während der Lehrzeit freitags weltweit für das Klima protestierten, gibt es seit einiger Zeit unter allen Generationen Akzeptanz für diese Bewegung. [31]

Klimaaktivisten, wie z.B. von Greenpeace, kämpfen seit Jahren für mehr Aufmerksamkeit für Natur-, Umwelt- und Klimaschutz, teilweise mit Mitteln am Rande der Legalität. [35] Oft fällt der Begriff Nachhaltigkeit. So stellen neben der Reduzierung von Plastik, einem weltweit verbesserten Ansatz an Ressourcenschonung oder der Stromerzeugung und dessen breiter Umschwung auf erneuerbare Energien ein wichtiges Thema dar. [30]

1.1 Klimaneutralität als Motivation

Der Weltklimarat (IPCC, Intergovernmental Panel on Climate Change) publiziert regelmäßig wissenschaftliche Berichte über die Klimasituation auf der Welt. Erkenntnisse daraus sind, dass die Treibhausgasemissionen maßgeblich für die globale Erderwärmung verantwortlich sind und drastisch gesenkt werden müssen. [21] Auf Grundlage der Forschungen des Weltklimarates wurde im Jahr 2015 schließlich das Pariser Klimaabkommen verabschiedet, bei welchem sich 197 Staaten verpflichtet haben, den Klimawandel und die damit eingehende Erderwärmung bis 2030 auf 1,5 Grad zu begrenzen. [7] Des Weiteren lautet das längerfristige Ziel der EU, die Treibhausgase bis 2050 um 80 - 95 % zu reduzieren [6] oder sogar gänzlich zu neutralisieren. [7]

Angelehnt an das Abkommen wurde von der Bundesregierung im Jahr 2019 ein Klimaschutzgesetz verabschiedet und 2021 aktualisiert. Ziel ist vor allem die Reduzierung der Treibhausgase aus dem Energiesektor, [7] denn dieser ist vor dem Industrie- und dem Verkehrssektor der größte Klimasünder in Deutschland. [6] Gesprochen wird über den Kohleausstieg, allen voran Kohlenstoffdioxid (CO_2), der schädlichste aller Energieträger. [33] Fällt die Kohleförderung als Energieträger weg und soll die Treibhausgasneutralität erreicht werden, braucht Deutschland im Zuge des zunehmenden Energiebedarfs eine Alternative und diese steckt zwingend in den emissionsfreien erneuerbaren Energien. In einem Bericht über die Nachhaltigkeitsstrategie 2021 der Bundesregierung, soll der Anteil aus erneuerbaren Energiequellen am Bruttostromverbrauch bis 2030 auf 65 % ansteigen. [7] Anderen Meinungen zur Folge reicht dies nicht, in Deutschland muss bis 2030 der Anteil erneuerbaren Energien wesentlich höher sein. [20] So wäre Deutschlands CO_2-Budget, der einkalkulierte und festgelegte Ausstoß von CO_2, bei gleichbleibenden Emissionen bereits 2026 verbraucht. [33]

Zwar hat sich der Anteil der Stromerzeugung aus Kohlekraftwerken in Europa zwischen 2015 und 2020 um 50 % reduziert [20] und auch in Deutschland steigt der Anteil grünen Stroms jährlich, doch der Rückgang fossiler Energien stagniert lediglich und nimmt nicht stark genug ab. Grund dafür ist der ebenfalls steigende Energiebedarf. [3, 28]

1.2 Fortschritt durch künstliche Intelligenz

Neben den erneuerbaren Energien gilt noch die Kernfusion als grüner Strom, eine unerschöpfliche Quelle an sauberer Energie. [13] Hierbei wird der Vorgang in der Sonne nachgebildet. Bei dem Prozess werden Kerne von Wasserstoffatomen bei immensem Druck und unter Temperaturen von hunderten Millionen Grad zusammengepresst, um schwerere Helium-Atome zu bilden. Die freigesetzte Energie ist dabei so groß, dass die Verbrennungswärme von 1 Gramm Brennstoff mit 11 Tonnen Kohle zu vergleichen ist. [22]

Die für die Fusion infrage kommenden Wasserstoffsorten sind Deuterium und Tritium. Deuterium ist in nahezu unerschöpftem Vorkommen im Meer vorhanden und Tritium kann innerhalb des Kraftwerkes aus Lithium gewonnen werden, weshalb die Fusionsbrennstoffe als günstig und das gesamte Verfahren in der Theorie als sehr effizient gelten. [22] Die Wissenschaftler müssen anhand bestimmter Parametern einen komplizierten Prozess steuern, welcher unter normalen Umständen nicht möglich wäre, da sich die Atome abstoßen würden. Genau hier fehlt bisher der wissenschaftliche Durchbruch. Die Dauer der Fusion ist noch zu kurz, weshalb selten mehr Energie gewonnen werden kann, als in den gesamten Prozess investiert wird. [13]

Abhilfe schaffen soll künstliche Intelligenz, im folgenden auch KI genannt. DeepMind, die KI-Tochter von der Google-Mutter Alphabet, ist in einer Kooperation mit der Technischen Hochschule Lausanne in der Schweiz, wo eine Deep-Reinforcement-Learning-AI darauf trainiert werden soll, das ultraheiße Plasma in dem Reaktor zu kontrollieren. Reinforcement Learning folgt einem Konzept, basierend auf Belohnungen, wobei ein System unter bestimmten Rahmenbedingungen ein Ziel gesetzt und für dessen Erreichen belohnt wird (bzw. auch Fehlerpunkte erhalten kann). [34] Dabei ist der Weg zum Erreichen des Ziels nicht vorgegeben. Erste Erfolge gibt es im Zusammenhang mit einem virtuellen Reaktor der Hochschule (TCV, Tokamak à Configuration Variable), bei welchem die KI die Hitze des Plasmas erfolgreich und ohne Menscheneinfluss für wenige Sekunden regulieren konnte. [8, 13]

1.3 Ziele der Arbeit

Die Motivation dieser Arbeit ergab sich aus den Überlegungen, dass die umweltfreundliche Energieerzeugung eine der entscheidendsten Fragen unserer Zeit ist und wir womöglich durch Hilfe der künstlichen Intelligenz, die Lösung für dieses Problem schon in unseren Händen halten.

In dieser Arbeit wird untersucht, inwieweit mithilfe neuronaler Netze realistische Zeitreihen vorhergesagt werden können. Anhand dieser Fragestellung werden die Prognosen von verschiedenen Netzstrukturen wie Feedforward Neural Network, Long-Short-Term-Memory, Autoencoder und Convolutional Neural Network miteinander verglichen. Überprüft wird, ob die Vorhersage mit dem Long-Short-Term-Memory-Verfahren zu besseren und realistischeren Ergebnissen führt, als die Vorhersage durch die anderen Netzstrukturen. Anhand dieser Verfahren wird versucht herauszufinden, ob in der Vorhersage von Zeitreihen ein Trend zur vermehrten Stromerzeugung durch erneuerbare Energien in Deutschland zu erkennen ist.

1.4 Struktur der Arbeit

Die strukturelle Aufteilung der Arbeit beginnt mit einem grundlegenden Kapitel 2, das Zeitreihen-, Regressionsanalyse und neuronale Netze erklärt. Darauf aufbauend werden in Kapitel 3 die Architekturen vorgestellt, mit denen der Vergleich dieser Untersuchung durchgeführt wird. Im anschließenden Kapitel 4 wird das methodische Vorgehen der Versuche beschrieben. In Kapitel 5 folgt die Vorhersage für das Fortführen einer Zahlenreihe und einer Sinuskurve. Der Hauptversuch befindet sich in Kapitel 6, in dem Energiewerte über einen unbekannten Zeitraum vorhergesagt werden. Eine Zusammenfassung und die Beantwortung der Forschungsfragen finden sich in Kapitel 7. Abgeschlossen wird die Arbeit mit einem Ausblick in Kapitel 8.

Abbildungen von fremden Quellen sind in der Bildbeschreibung entsprechend markiert. Alle anderen Abbildungen sind Eigendarstellungen.

2 Grundlagen der Zeitreihenanalyse

Zum einen gibt es komplexe Szenarien, wie am Beispiel mit der Kernfusion zu sehen, wo der Einsatz von KI noch nicht gänzlich zum erwünschten Durchbruch geführt hat (Kapitel 1.2). An anderer Stelle gibt es Bereiche, in denen KI-Verfahren erfolgreich eingesetzt und essenziell geworden sind [34].

In diesem Kapitel werden grundlegende, für die Zeitreihenvorhersage mit neuronalen Netzen spezifischen Ansätze aus dem Bereich Machine Learning erläutert. Machine Learning, kurz ML genannt, ist eine Rubrik des KI-Bereichs. Sie ist als Definition für Wissenserzeugung aus Daten zu verstehen, bei welcher ein System aus vorhandenen Daten Zusammenhänge und Gesetzmäßigkeiten lernt und diese gewonnenen Kenntnisse auf ungesehene neue Daten anwenden kann. [14] Ein Beispiel ist die Vorhersage der Stromerzeugung durch Windenergie, die basierend auf bestimmten Variablen wie z.B. Windstärke, Luftdruck, etc. erhoben werden kann.

2.1 Machine Learning in der Praxis

In der Praxis werden Vorhersagen in vielen weiteren Einsatzgebieten eingesetzt. Sei es bei der Prognose eines Aktienverlaufs, des Wetters, von Krankheitsverläufen, der Nachfrage nach spezifischen Produkten oder der frühzeitigen Bekämpfung von Verbrechen, wobei im letzten Fall auch moralische Fragen zur Debatte stehen. [34] Grundlegend kann gesagt werden, dass Systeme für die Vorhersage in allen Bereichen Anwendung finden: Finanzen, Marketing, Sport etc.

Zwar wurden zum Beispiel bereits in den 1950er-Jahren erste Experimente mit neuronalen Netzen durchgeführt, doch erst im Zuge der Digitalisierung und der damit verbundenen Menge an gesammelten Daten, konnten sich Deep Learning Verfahren richtig durchsetzen. [11] Mittlerweile profitiert fast jeder Bereich der Künstlichen Intelligenz von dieser wachsenden Menge an Daten. [14] Doch nicht nur die Quantität, sondern vor allem die

Qualität der Daten ist entscheidend. Gleichzeitig wächst die Gefahr der Datenmanipulation und die daraus resultierenden verfälschten Ergebnisse. Sind in Daten regelmäßig Ausreißer enthalten, die auf z.B. defekte Messgeräte zurückzuführen sind, erkennt das KI-Modell in diesen verfälschten Daten ebenfalls eine Struktur. [14]

Die Sprach- und Text-Verarbeitung sind Bereiche, in welchem die ML-Forschung mit am weitesten fortgeschritten ist. [14, 32] Geprägt wurden diese durch die Entwicklungen in Technologien wie Word-Embedding, rekurrente neuronale Netze und Sequenz-zu-Sequenz-Architekturen. Eine andere Sparte ist die Auswertung von Foto- und Videomaterial. Mit konvolutionalen Netzen können vor allem wiederkehrende Strukturen und Formen in Bildern lokalisiert und analysiert werden. Maßgeblich profitiert haben und erst dadurch ermöglicht wurden, sind z.B. die Gesichtserkennung oder das autonome Fahren. [10, 14]

2.2 Zeitreihen

Ein ähnlich prägender Umbruch findet derzeit bei der Analyse von Zeitreihen statt, mit deren Hilfe diverse Prozesse, zum Teil in der Einleitung dieses Kapitels erwähnt, vorhergesagt werden können. [14] Das Grundprinzip ist es dabei, aus der Historie eines Prozesses ein Verlaufsmuster zu erkennen und damit die weitere Entwicklung basierend nach dem Muster fortzuführen.

Als Menschen vor unserer Zeit versuchten das Wetter vorherzusagen, haben sie sich an relevante und verfügbare Indikatoren, wie z.B. Wolken, Temperatur, Windstärke, das Verhalten von Tieren etc., orientiert. Ihre Prognosen basierten auf Erfahrungen, die sie im Laufe ihres Lebens anhand dieser Indikatoren gesammelt haben. Komplexe Modelle, wie sie heute für Prognosen eingesetzt werden, funktionieren ähnlich. [14] Daten können durch computergestützte Modelle jedoch wesentlich schneller verarbeitet werden als vom Menschen und dabei Zusammenhänge erkennen, die uns verborgen bleiben. [4] Solche Datenmengen werden in Zeitreihen verarbeitet, welche wiederum in kleineren Sequenzen unterteilt werden, was in Kapitel 3 ausführlicher beschrieben wird.

Zeitreihen (engl. Time Series) sind, etwa im Unterschied zu Text- oder Bilddaten, kein spezifischer Datentyp. [14] Im Grunde sind sie Sammlungen von Beobachtungen, die über einen bestimmten Zeitraum aufgenommen werden. Formal betrachtet sind Zeitreihen eine Menge von Werten, die in einem Zeitablauf gemessen und gemäß der Zeit geordnet

sind. [1] Die Zeit gilt dabei als natürliche Ordnung und kann in der Reihenfolge nicht geändert werden. [19] Eine Zeitreihe lässt sich ausdrücken durch:

$$\langle y_1, y_2, y_3, y_4, \dots y_T \rangle$$

Wobei der Index T eine Abbildung der Zeit ist, Quelle [1] Seite 126. Die Komponente Zeit wird genutzt, um herauszufinden, ob spätere Messwerte von früheren Messwerten abhängig sind. Sind in den Daten wiederkehrende Strukturen vorhanden, so bedeutet dies, dass die Reihe einer Art Regelwerk, z.B. einer saisonalen Komponente, unterliegt. [14]

Univariate und multivariate Zeitreihen

Wenn Zeitreihen verwendet werden, wo nur eine abhängige Variable (auch Merkmal oder Feature genannt) betrachtet wird, so wird von einer univariaten Varianzanalyse gesprochen. Zeitreihen, die mehr als einen Faktor enthalten, werden multivariate Varianzanalysen genannt. [1] Eine andere Ausdrucksweise wäre für den Begriff univariat eindimensional und für multivariat mehrdimensional. Die folgende Tabelle 2.1 zeigt den strukturellen Aufbau von univariaten bzw. multivariaten Zeitreihen, wobei der Zeitstempel den Index darstellt.

Tabelle 2.1: Vergleich einer univariaten und multivariaten Datenstruktur.

Univariate Zeitreihe		Multivariate Zeitreihe		
Zeit	Preis (€)	Zeit	Temp. (°C)	Luftfeuchtigkeit (%)
19:00:00	120	2020-04-12	36.2	72
20:00:00	123	2020-04-13	37	77
21:00:00	128	2020-04-14	35.4	75

2.2.1 Datenvorverarbeitung

Im Zuge der Datenvorverarbeitung wird in der Praxis oft von einer 80/20-Teilung gesprochen. [34] Gemeint ist, dass ca. 80 % des Arbeitsaufwandes des Benutzers in die Datenvorverarbeitung fließt und lediglich 20 % in den tatsächlichen Trainingsprozess des Modells, was die Bedeutung der Datenvorverarbeitung hervorhebt. Zur Datenvorverarbeitung gehört z.B. das Bereinigen von lückenhaften Datensätzen [14].

Bevor ML-Verfahren auf Daten angewandt werden können, müssen je nach Verfahren unterschiedliche Schritte durchgeführt werden, um die Daten in Form zu bringen. Bei neuronalen Netzen werden z.B. die Daten in Trainings-, Test- und manchmal auch in Validierungsdatensätze unterteilt. Generell ist das Bereinigen der Daten allerdings ein Schritt, der für alle Bereiche der Datenanalyse wichtig ist. [14] Im folgenden Kapitel werden Methoden vorgestellt, wodurch fehlende Werte in einem Datensatz ersetzt (imputiert) werden können.

2.2.2 Imputation fehlender Werte

Zeitreihen sind anfällig für fehlerhafte oder inkonsistente Daten. Faktoren, wie sich ändernde Rahmenbedienungen, Messverfahren oder verdreckte Sensoren, können Ausreißer in den Daten verursachen und erheblichen Einfluss auf die Aussagekraft der Daten nehmen. [14, 16]

Um die Performance von Algorithmen zu vergleichen, werden in der Literatur gerne vorgefertigte Datensätze verwendet, wie der M3, M4 oder ETTh1. [25] Ist dies nicht gegeben, so können durch mathematische Verfahren Lücken in den Datensätzen imputiert werden. Bei einer Umfrage am Bahnhof zu Alter und Einkommen der befragten Personen werden metrische Variablen erfasst. Dies sind Querschnittsdaten, wo – je nach Menge – fehlende Werte oft entfernt werden können, da sie für das Gesamtergebnis der Datenerhebung weniger entscheidend sind. Technisch gesehen kann in diesem Fall auch häufig der Mittelwert oder Median eingesetzt werden. Zeitreihen reagieren in einem solchen Fall sensibler. [14] Sollen in einer Klimazeitreihe im Januar fehlende Werte durch den Median des Jahres imputieren werden, werden Ausreißer produziert, die den Anlernprozess des Algorithmuses verzerren.

Imputations-Verfahren

Um für das Auftreten von fehlenden Werten in den Versuchsreihen beheben zu können, werden in diesem Abschnitt unterschiedliche Methoden zur Imputation fehlender Werte vorgestellt und getestet. Herangezogen wird ein kleiner Datensatz der Sinuskurve (Quelle des Datensatzes: Sinus.csv, letzter Zugriff 27.05.2022). Um fehlende Werte zu erzeugen, wurden innerhalb der ersten zwei Perioden händisch drei Werte entfernt, wie in Abbildung 2.1 dargestellt ist.

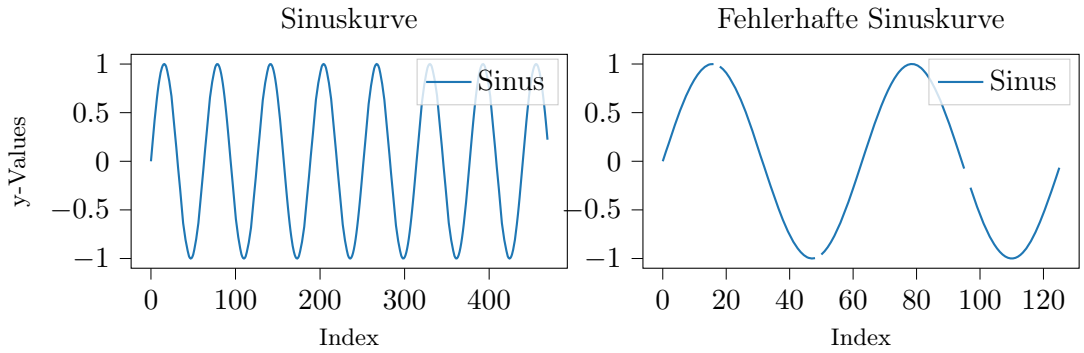


Abbildung 2.1: Links: Ursprüngliche Sinuskurve. Rechts: Per Hand verfälschte Kurve.

Bei den folgenden Imputations-Verfahren wurde sich an die Vorgehensweise in [14] orientiert.

Methode (1) Mean: Bei der ersten Methode wird der Mittelwert des gesamten Datensatzes errechnet und für die fehlenden Werte eingesetzt. Abbildung 2.2 veranschaulicht deutlich, dass es mit dieser Variante zu Ausreißern kommt. Bei Querschnittsdaten würde diese Methode bessere Ergebnisse erzielen.

Methode (2) First Lag: Die zweite Methode nutzt Lags für die Imputation. Lags sind als zeitversetzte Werte einer Reihe zu verstehen, wobei das Lag erster Ordnung den Messwert des vorherigen Intervalls bezeichnet. [14] Das Imputieren durch das erste Lag, ist in Tabelle 2.2 zu sehen. In der 2. Spalte sind Temperaturangaben mit einem fehlenden Wert enthalten. Die 3. Spalte zeigt das Lag erster Ordnung und in der 4. Spalte ist Imputation des fehlenden Wertes durch das Lag erster Ordnung dargestellt.

Tabelle 2.2: Anwendung der Lag-Imputationsmethode für eine univariate Zeitreihe der Temperatur.

Zeitindex	Temp. (°C)	Temp. (°C) 1. Lag	Temp. (°C) Lag **imputiert**
2016-03-05	12.4	-	12.4
2016-03-12	15.1	12.4	15.1
2016-03-19	14.8	15.1	14.8
2016-03-26	**-**	14.8	**14.8**

Das Lag vierter Ordnung wäre der Messwert vier Werte zuvor, usw. In diesem einfacheren Sinus-Beispiel wird der Wert vor einer Lücke in die Lücke selbst eingefügt, so wie in der Tabelle zuvor. Wie in Abbildung 2.2 zu erkennen ist, ist die Kurve im Vergleich zum

Mittelwert-Verfahren mit geringeren Ausreißern sauberer. Anhand des Graphen kann abgeleitet werden, dass die Imputation gleicher aufeinanderfolgender Werte, bei großer Steigung oder Gefälle, deutlicher zu sehen ist, als bei ähnlichen Werten mit kleinerer Differenz.

Methode (3) Rolling Mean: Die Lag-Methode findet auch bei der dritten Methode Verwendung. Die zu imputierenden Werte werden allerdings vorher durch den Rolling Mean berechnet. Dabei wird ein Zeitfenster über die Zeitreihe geschoben und für jedes dieser Fenster ein Mittelwert erstellt. Durch verschiedene Parameter kann die Funktion, z.B. in der Größe des Fensters, verändert werden. Für jeden Wert innerhalb des Fensters wird ein Mittelwert aus den übrigen Werten erstellt. Dieser wird Rolling Mean (rollender Mittelwert) bezeichnet und nach der Lag-Methode in vorhandene Lücken eingefügt. Die so imputierten Werte bilden eine nahezu perfekte und natürlich aussehende Kurve, wie im letzten Graphen der Abbildung 2.2 zu sehen ist.

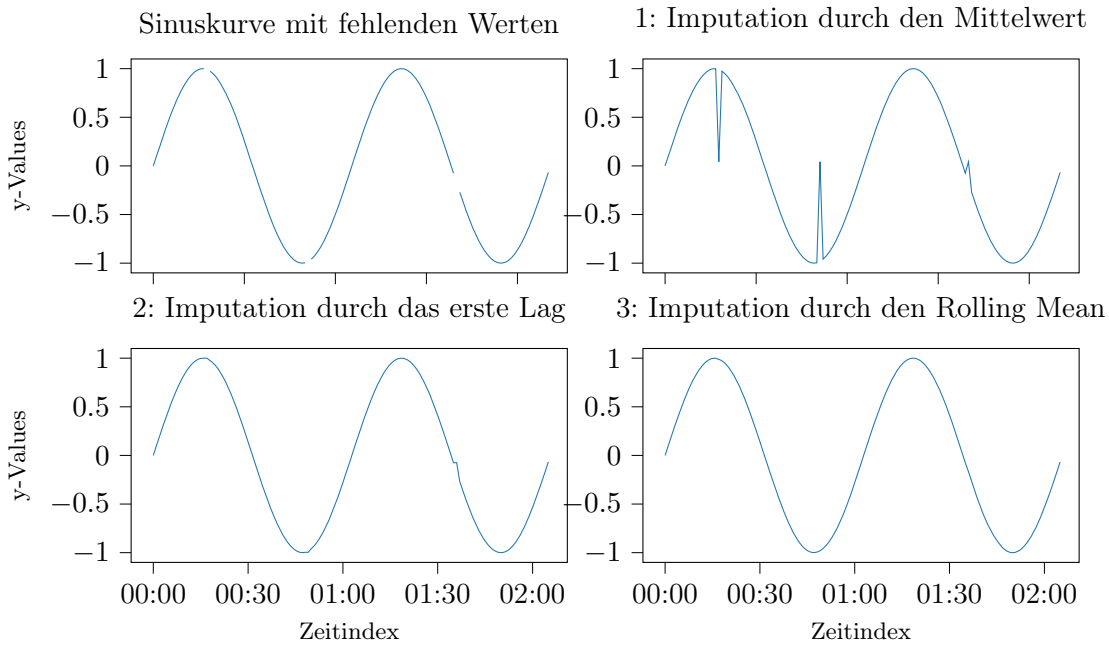


Abbildung 2.2: Darstellung unterschiedlicher Imputations-Verfahren anhand der Sinuskurve, in welcher Werte händisch entfernt wurden.

2.3 Regressionsanalyse

Der Bereich Machine Learning lässt sich in verschiedene Kategorien unterteilen. Bzgl. dieser Arbeit soll ein Algorithmus eine Beziehung zwischen einer Anzahl an Merkmalen (x-Variablen) und einer Zielvariable (y-Variable) lernen, bzw. verstehen. Neuronale Netze, die für diese Aufgabe konstruiert werden, gehören der Kategorie Supervised Learning an. [14] Weitere Kategorien, welche an dieser Stelle nicht weiter behandelt werden, heißen Unsupervised Learning und Reinforcement Learning. [34] Letzteres wurde in der Einleitung unter Kapitel 1.2 bereits aufgeführt.

Ein Grundprinzip neuronaler Netz ist die Regression, welche sich in drei Formen unterscheiden lässt:

- Lineare Regression
- Logistische Regression
- Softmax-Regression

Soll in einer Fragestellung eine Zielvariable geschätzt werden, z.B. die Temperatur an einem bestimmten Tag, so handelt es sich um den Bereich der linearen Regression. Wenn die Aufgabe in die Klassifizierung fällt, also kategorische Variablen wie die Zustände fehlerhaft und fehlerfrei verarbeitet werden, so wird von der logistischen Regression gesprochen. Ist in diesem Falle von mehr als zwei Kategorien die Rede, z.B. Hund, Katze und Maus, so handelt es sich um die Softmax-Regression, einem Verbund aus logistischen Regressionen. [14]

Lineare Regression

Bei den für diese Arbeit interessanten Zeitreihenvorhersagen werden Werte prognostiziert, weshalb die lineare Regression näher vorgestellt wird. Das Beispiel, welches in diesem Abschnitt angewendet wird, stammt aus dem Buch *Machine Learning für Zeitreihen* von Jochen Hirschle (2021). [14] Anhand des Beispiels wird deutlich, weshalb im weiteren Verlauf dieser Arbeit bestimmte Ansätze gewählt wurden.

In diesem Szenario sammelt ein Besitzer eines Cafés hat Daten darüber, wie viele Gäste in einer Stunde sein Café aufsuchen und wie viel Nettoverdienst in diesem Zeitraum

erwirtschaftet wird. Ziel ist es, ein Modell zu erstellen, worin mithilfe bekannter Informationen, der Anzahl an Gästen (x), die unbekannten Informationen, der Nettoverdienst (y), geschätzt wird. Die bekannten Daten sind in einem Streudiagramm zusammengefasst. Jeder Punkt wird durch ein Tupel aus x- und y-Wert beschrieben, wie das linke Diagramm in Abbildung 2.3 veranschaulicht.

Um zu demonstrieren, warum ein solches Modell von Nutzen ist, ist im linken Bereich der Abbildung die Gerade einer Schätzformel eingezeichnet, welche den Zusammenhang der beiden Variablen abbildet. Mithilfe der Geraden kann abgelesen werden, dass die Schwelle zwischen Verdienst und Verlust bei 9 Gästen liegt.

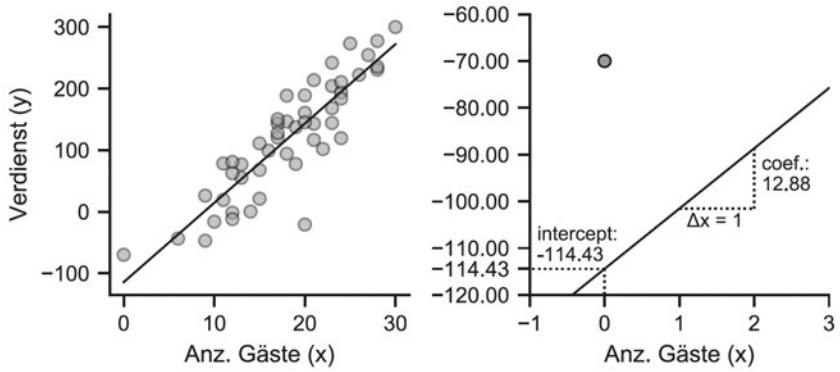


Abbildung 2.3: Links: Modellierung des Zusammenhangs zweier Variablen mithilfe einer linearen Regression. Rechts: Interpretation der Gerade. [14] Seite 39.

Ein einfaches, aber vielseitig einsetzbares mathematisches Modell ist die folgende lineare Regression, Quelle [1] Seite 64:

$$\hat{y} = \alpha_0 + \beta_1 \cdot x_1 \tag{2.1}$$

mit

$\hat{y}$ = Schätzung der abhängigen Variable y $\quad$ α_0 = konstantes Glied

β_1 = Regressionskoeffizient $\quad$ x = unabhängige Variable x

Bei mehreren x-Merkmalen würde jedes x mit einem eigenen Koeffizienten β multipliziert werden.

Die in Gleichung 2.3 eingesetzten Koeffizienten sind mit Blick auf das rechte Bild in Abbildung 2.3 zu erklären. In das Beispiel übertragen, ergibt sich folgende Formel:

$$\hat{y}_{\text{verdienst}} = \alpha + \beta_{\text{gäste}} \cdot \chi_{\text{gäste}} \tag{2.2}$$

$$= -114.43 + 12.88 \cdot \chi_{\text{gäste}} \tag{2.3}$$

Der Koeffizient α entspricht dem Intercept. Er beschreibt die vertikale Lage der Kurve. Durch den Schnittpunkt kann abgeleitet werden, dass der Verdienst, bzw. Verlust bei 0 Gästen -114.43 Euro beträgt. Der Koeffizient β beschreibt die Steigung der Geraden. Wird statt eines Gastes ein zweiter Gast bewirtet, so erhöht sich der Verdienst um durchschnittlich 12.88 Euro. Wenn dementsprechend α und β bekannt sind, kann mit der Formel für jede Gästezahl der geschätzte Verdienst berechnet werden.

Residuen

Um diesen Koeffizienten zu ermitteln, werden im Weiteren die Residuen, die Abstände der Punkte zu der Geraden, betrachtet. Diese sind in Abbildung 2.4 dargestellt.

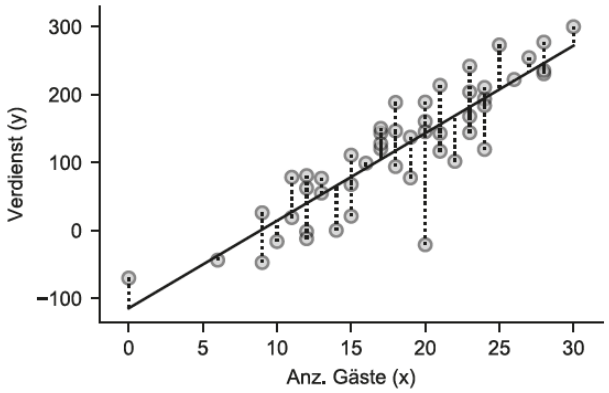


Abbildung 2.4: Modellierung des Zusammenhangs zweier Variablen mithilfe einer linearen Regression. Darstellung der Residuen durch gestrichelte Linien zur Geraden. Quelle [14] Seite 41.

Die Residuen bilden den Schlüssel zur Schätzung der Regressionsgeraden. [1] Sie helfen zum einen die Qualität des Schätzmodells abzuleiten und zum anderen die Koeffizienten

(Gewichte) zu optimieren. Generell gilt, je geringer die Residuen, desto besser ist das Modell. [14]

2.3.1 Verlustfunktionen

Verlustfunktionen (engl. Loss Functions) wie der MAE (engl. Mean Absolute Error) und MSE (auch RMSE bezeichnet, engl. (Root) Mean Squared Error) helfen als Indikatoren weiter, um die Qualität des Modells zu bestimmen. Sie berechnen die Differenzen der Residuen und bilden daraus den Mittelwert [14]. Entscheidend ist der Abstand zu einem Wert von 0, welcher die Qualität des Modells widerspiegelt. Dieses Vorgehen findet auch in neuronalen Netzen statt. [1]

Da Verlustfunktionen (auch Kostenfunktion genannt) wesentliche Bestandteile bei der Optimierung von ML-Modellen sind, [11] soll der folgende Abschnitt einen groben Überblick über gängige Funktionen geben. Es handelt sich jedoch nur um einen kleinen Exkurs, da eine detaillierte Betrachtung den Rahmen dieser Arbeit überschreiten würde. Im vorhergegangenen Café-Beispiel, wurde der Einsatz des MAE und MSE genutzt, um die Aussagekraft eines Modells zu bestimmen. [29] Dies ist die übergeordnete Aufgabe der Verlustfunktionen. Der Verlust (engl. Loss) gibt dabei die Aussagekraft einer Vorhersage y_{pred}, in Relation zum korrekten Wert y_{true} eines Modells an. Verlustfunktionen lassen sich in zwei Kategorien unterteilen:

Klassifikation: Sollen qualitative Werte prognostiziert werden, so wird von einer Klassifikationsfragestellung gesprochen. [16] Eine Beispielfunktionen wäre die Binary Crossentropy (binäre Kreuzentropie), welche bei binären Klassen zwischen echten und prognostizierten Werten durch 0 und 1 unterscheidet. Sind mehr als zwei Werte relevant, wäre z.B. die Sparse Categorical Crossentropy (spärliche kategoriale Kreuzentropie) eine mögliche Alternative.

Regression: Werden quantitative Werte prognostiziert, handelt es sich um eine Regressionsfragestellung (z.B. Temperaturen). [16] Beispielfunktionen wären der MAE oder MSE.

Am MAE und MSE soll beispielhaft erklärt werden, wo ein Unterschied zwischen Verlustfunktionen liegen kann. Der einzige Unterschied bei diesen beiden Funktionen ist,

dass beim MAE der absolute Fehlerwert berechnet und beim MSE dieser Wert quadriert wird. Dementsprechend ähneln sich die Formeln wie folgt, Quelle [27] Seite 36:

$$MAE = \frac{\sum_{t=1}^{n} |y_t - \hat{y}_t|}{n} \tag{2.4}$$

$$MSE = \frac{\sum_{t=1}^{n} (y_t - \hat{y}_t)^2}{n} \tag{2.5}$$

Die Quadrur beim MSE bewirkt, dass starke Abweichungen höher gewichtet werden, der Einfluss geringerer Abweichungen wird hingegen geschwächt. [14] Da diese beiden Funktionen häufig in der Literatur und vergleichbaren Arbeiten wie [14, 36, 37, 38] eingesetzt werden, werden sie auch in dieser Arbeit verwendet. Tabelle 2.3 zeigt eine Übersicht von Aufgabenstellungen und möglichen Verlustfunktionen.

Tabelle 2.3: Wahl der Verlustfunktion und Aufbau der letzten Schicht in Abhängigkeit der Aufgabenstellung. Quelle [14] Seite 157.

Aufgabenstellung	**Letzte Schicht**	**Verlustfunktion**
Regression: Stetige Zielvariable, z.B. Einkommen, Temperatur	Ein Neuron, keine Aktivierung (Identity-Funktion)	Mittlerer quadratischer Fehler (MSE)
Binäre Klassifizierung: z.B. intakt (0) vs. defekt (1)	Ein Neuron, Aktivierung mit Sigmoid (logistische Funktion)	Binäre Kreuzentropie (Binary Crossentropy)
Multinomiale Klassifizierung: Differenzierung mehrerer Klassen: z.B. Rose (0), Tulpe (1), Lilie (2)	Anzahl Neuronen = Anzahl der Klassen, die geschätzt werden, Aktivierung mit Softmax	Kategoriale Kreuzentropie (Categorical Crossentropy)

Überprüfung der Imputations-Verfahren

Im Kapitel *Imputation fehlender Werte* 2.2.2 wurden verschiedene Verfahren getestet, um fehlende Werte in der Sinuskurve zu imputieren. Anhand der Abbildungen 2.2 konnte festgestellt werden, dass das Rolling-Mean-Verfahren das geeignetste Verfahren zur Imputation fehlender Werte ist. Dieses Ergebnis wird im Folgenden durch das Verwenden von Verlustfunktionen validiert. Dabei eingesetzt wurden der MAE und MSE. Die folgende Tabelle 2.4 zeigt anhand der Imputations-Verfahren den Vergleich beider Fehlerwerte. In der ersten Zeile wurde der Verlust über die ersten 125 Werte, einschließlich der

Imputationen, berechnet und in der zweiten Zeile, lediglich über die imputierten Werte selbst. Die Werte zeigen, dass das Rolling-Mean-Verfahren den niedrigsten Verlust hat und somit das geeignetste Verfahren bei diesem Beispiel ist.

Tabelle 2.4: Anwendung von MAE/MSE für den Vergleich der Imputations-Verfahren.

Variante	Loss	Mean	First Lag	Rolling Mean
Ersten 125 Werte	MAE	0.017530	0.000966	0.000086
	MSE	0.015989	0.000081	0.000000
Über imputierten Werte	MAE	0.730431	0.040265	0.003578
	MSE	0.666220	0.003362	0.000016

2.3.2 Gradientenabstieg

Koeffizienten, bzw. Gewichte, werden initial zufällig ausgewählt und durch einen Optimierer schrittweise angepasst, während die Verlustfunktion überprüft, ob der Fehlerwert dabei niedriger wird. Bezogen auf das Café-Beispiel in diesem Kapitel zeigt die linke Grafik in Abbildung 2.5 die schrittweise Annäherung, den Gradientenabstieg, an den tiefsten Punkt (den geringsten Fehler) der Verlustfunktion und somit den optimalen Koeffizienten. Die Größe der Schritte wird durch eine Lernrate angegeben. Durch die Annäherung (insbesondere bei neuronalen Netzen) kann es dabei vorkommen, dass die Verlustfunktionen den optimalen tiefsten Punkt verpassen, quasi überspringen. In diesem Fall wird von einem lokalen Minimum gesprochen, was in der allgemeineren Darstellung in der rechten Abbildung 2.5 gezeigt wird. Der optimale, niedrigste Punkt wird globales Minimum genannt. [34]

Bzgl. der Lernrate ist hinzuzufügen, dass bei kleiner Schrittgröße (z.B. 0.0001) mit höherer Wahrscheinlichkeit das Minimum der Verlustfunktion gefunden wird, die Dauer des Trainingsprozesses dafür dabei steigt. Sie wird in der Praxis oft dynamisch verringert. [11]

Dies ist das Grundprinzip des Modelllernens. Nahezu alle Deep Learning Algorithmen basieren auf den in diesem Café-Beispiel vorkommenden Bestandteile. Benötigt werden ein spezifischer Datensatz, eine Verlustfunktion und ein Optimierungsverfahren, welche zu einem Schätzmodell kombiniert werden. [11]

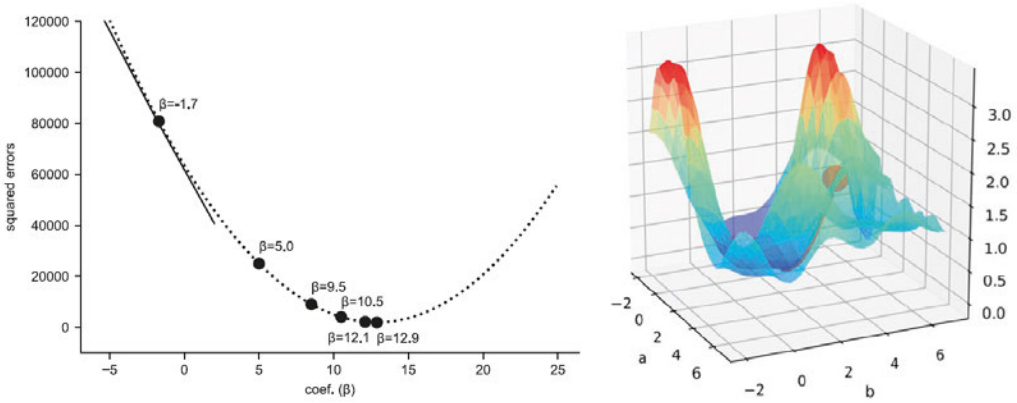


Abbildung 2.5: Links: Iterative Annäherung an den optimalen Koeffizienten aus dem Café-Beispiel. Quelle [14] Seite 43. Rechts: Darstellung des Gradientenabstiegs, bei dem der tiefste Punkt der Fläche dem globalen Minimum der Verlustfunktion entspricht. Quelle [34] Seite 24.

2.4 Künstliche neuronale Netze

Künstliche neuronale Netze, kurz KNN, sind eine vielversprechende Methode zur Analyse von Zeitreihen. [14] Sie sind von biologischen Vernetzungen im Gehirn inspiriert. Den Synapsen im Gehirn nachempfunden können aus diesen Netzstrukturen künstliche Neuronen für Berechnungen genutzt werden, wobei jedes Neuron ein Signal empfängt und dieses ggf. an das nächste Neuron in der nächsten Schicht weiterleitet. Darstellung 2.6 zeigt den schematischen Aufbau eines neuronalen Netzes. Dabei sind Neuronen in Schichten (engl. Layers) angeordnet, sie werden Input-Schicht, Output-Schicht und die dazwischen liegenden versteckte Schichten (engl. Hidden Layer) genannt. [34]

Ein Aufbau mehrschichtiger Netze, von bis zu 1000 hidden layern, wird durch den Begriff Deep Learning beschrieben und kann unter anderem auch aus einer Verflechtung mehrere Netze bestehen. [4, 10] Über Verbindungen gelangen Signale, bzw. Werte, von Neuron zu Neuron. Der Aufbau des dargestellten Netzes, die Architektur, entspricht einem Feedforward-Netz, da die Signale nur in eine Richtung, zum Output Layer, weitergeleitet werden. [11] Dabei wird durch weitere Merkmale Einfluss auf die Signale genommen. So ist jeder Input von einer Gewichtung (engl. Weight) betroffen, welches die Wichtigkeit der Verbindung widerspiegelt. Zusätzlich beeinflusst ein Bias b, in der oberen Grafik durch gelbe Kreise dargestellt, die Berechnung im Neuron. [34] Aufgrund der Regressionsauf-

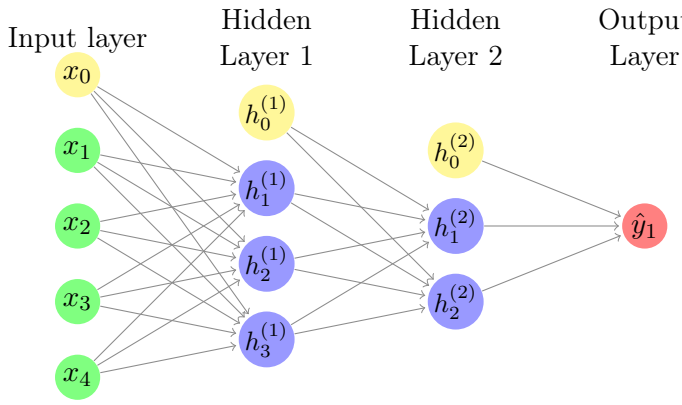
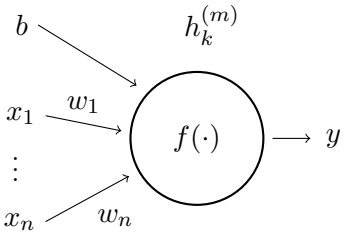


Abbildung 2.6: Schematische Darstellung eines künstlichen neuronalen Netzes mit zwei Hidden Layern: Bei diesem Netz wird von Fully-Connected-Layers gesprochen, da alle Neuronen mit den Neuronen aus der darauffolgenden Schicht verbunden sind. Angelehnt an die Darstellung in [34] auf Seite 19.

gabe befindet sich in den Netzstrukturen ein Neuron in der letzten Output-Schicht, da lediglich ein Prognosewert ausgegeben werden soll.

Aufbau eines Neurons



$$
\begin{aligned}
x_1 \dots x_n &= \text{Inputs} \\
w_1 \dots w_n &= \text{Gewichtungskoeffizienten} \\
b &= \text{Bias} \\
h_k^{(m)} &= \text{Neuron } h \text{ in hidden layer } m \text{ an Position } k \\
y &= \text{Output} \\
f &= \text{Aktivierungsfunktion} \\
n &= \text{Anzahl der Inputwerte}
\end{aligned}
$$

Abbildung 2.7: Darstellung eines Neurons, angelehnt an die Abbildung von Jochen Hirschle. Quelle [14] Seite 151.

Darstellung 2.7 zeigt den inneren Aufbau eines künstlichen Neurons. Die Berechnung darin entspricht dem Prinzip der linearen Regression. [14] Die Eingabewerte x_n stellen in der ersten Schicht den Input aus den Trainingsdaten dar. In den darauffolgenden Schichten sind sie der Output der vorherigen Neuronen und werden mit einem Gewicht verrechnet.

Die Berechnung im Inneren erfolgt durch die Aufsummierung der angepassten Input-Werte. Diese Summe wird mit einem Bias verrechnet, welcher dem Intercept aus der linearen Regression entspricht. Eine Aktivierungsfunktion f transformiert die Eingabe, also die Summe der angepassten Input-Werte und den Bias, auf einen anderen Wertebereich. [34] Der beschriebene Prozess wird durch Formel 2.6 ausgedrückt, angelehnt an die Darstellung von Quelle [14] Seite 151:

$$y = f(b + \sum_{i=1}^{n} w_i x_i) \tag{2.6}$$

2.4.1 Aktivierungsfunktion

Die Sigmoid bzw. logistische Funktion galt lange als Standard der Aktivierungsfunktionen. Hierbei werden Werte auf einen Bereich von 0 und 1 transformiert, wobei sehr große Werte den Wert 1 annehmen und sehr kleine den Wert 0. Dies ist hilfreich bei Wahrscheinlichkeitsberechnungen, da auch hier der Wertebereich zwischen 0 und 1 liegt. Ähnlich zur Sigmoid-Funktion funktioniert der Tangens Hyperbolicus, welcher die Sigmoid-Funktion ab den 2000ern größtenteils abgelöst hat. Der Wertebereich hat eine Spanne von -1 bis 1, wodurch stark negative Werte und Werte um 0 abgebildet werden können. [34]

Mittlerweile hat sich die Rectified Linear Activation (ReL) als Aktivierungsfunktion für tiefe neuronale Netze durchgesetzt. Für Werte kleiner 0 wird 0 zurückgegeben, für Werte größer 0 der Wert selbst. ReL, im Allgemeinen als ReL-Unit (ReLU) bezeichnet, ist eine nicht-lineare Funktion, welche bei negativen Werten eine Steigung von 0 und für positive Werte eine Steigung von 1 annimmt und über den Bias verschoben werden kann. Außerdem wirkt der Algorithmus dem Vanishing-Gradient-Problem entgegen, welches im Trainingsprozess aufgrund des Multiplizierens von niedrigen Zahlen das Finden der optimalen Gradienten erschwert. [34] Weitere Aktivierungsfunktionen wären z.B. Identity und Sign [14].

Für die Netzstrukturen in dieser Arbeit wird daher ReLU als Aktivierungsfunktion eingesetzt.

2.4.2 Training und Prognose

Das Training findet meistens in mehreren Durchläufen, als Epochen bezeichnet, statt. In der Datenvorverarbeitung werden die Daten in Trainings-, Test- und je nach Anwendung auch in Validierungssätze unterteilt. Das neuronale Netz trainiert anhand des Trainingssatzes und gleicht die Ergebnisse parallel mit den Validierungsdaten ab. Nach Abschluss des Trainings wird die Qualität des Modells im Vergleich zu den Testdaten überprüft. [34]

Zu Beginn des Trainings erhalten Neuronen und Bias zufällig initiierte Gewichtungen, welche aus dem Intervall [-1,1] gewählt werden. Nach dem Durchlauf der ersten Epoche wird nun wieder ähnlich wie bei der linearen Regression vorgegangen. [14, 34] Zunächst wird die Abweichung des finalen Outputs y_{pred} mit dem korrekten Trainingswert y_{true} durch eine Verlustfunktion errechnet. Im darauffolgenden Vorgang, der Backpropagation (Ableitung der einzelnen Layers), werden die Fehlerwerte der Verlustfunktion rückwärts durch das Netz gespeist. Dabei passt ein Optimierungsalgorithmus, abhängig der Learning-Rate, die Gewichte in den einzelnen Schichten an (Forward Propagation/Pass). Hierbei ist wieder das Ziel, das globale Minimum der Verlustfunktion zu finden (zur Erinnerung Abbildung 2.5). [34]

Ein weit verbreiteter Optimierungsalgorithmus ist der AdamOptimizer, welcher die Lernrate dynamisch anpasst und den Anlernprozess beschleunigt. [14] Der AdamOptimizer wird auch in dieser Arbeit verwendet wird.

Dieser Prozess wiederholt sich in jeder Epoche. [34] Im Gegensatz zur linearen Regression hat ein neuronales Netz den Vorteil, Aufgabenteile in verschiedenen Schichten und Neuronen zu verarbeiten und ein Problem zu entpacken, z.B. ein nicht-lineares in ein lineares Problem zu übersetzen, mit welcher sich dann eine spätere Schicht befassen kann. [14]

Je nach Größe der Daten, der Komplexität des Netzes, der Learning-Rate, der Anzahl der Epochen, der PC-Hardware und der Batch Size wird die Performance und Dauer des Trainings des neuronalen Netzes beeinflusst. [34] Nach Abschluss des Trainings wird die Aussagekraft des Modells anhand der Testdaten validiert. Die ausgegebenen Schätzwerte $\hat{y}_{\text{pred}}$ werden mit den echten Werten der Testdaten y_{true} verglichen.

3 Vorhersageverfahren

Im Unterkapitel *Zeitreihen* 2.2 wurde die Besonderheit von Zeitreihen vorgestellt. Aufgrund des Zeitstempels innerhalb der Daten können saisonale Eigenschaften wie z.B. Trends erkannt werden. Dieses Kapitel gibt Aufschluss darüber, welche Techniken und Architekturen bei solchen Zeitreihenprognosen eingesetzt werden.

Ein Kernelement ist dabei das Verarbeiten von Sequenzen. [11, 14, 32, 34] Demonstrieren lässt sich das an einer einfachen Aufgabe. Im Beispiel soll eine Raumtemperatur prognostiziert werden. An einem Thermometer wird dabei der Wert 20 °C abgelesen, welcher im einzelnen wenig Aufschluss über die Temperaturentwicklung in dem Raum gibt. Wird jedoch die Temperatur bspw. in einem 5-Minuten-Intervall gemessen und zeigt die Werte 20 °C, 21 °C und 22 °C an, so kann aus diesen Daten wesentlich mehr entnommen und ein Trend vorhergesagt werden, nämlich dass die Temperatur steigt. Diese Herangehensweise soll auch bei neuronalen Netzen angewendet werden. Der gesamte Datensatz wird in Sequenzen unterteilt, welche dem Netz häppchenweise als Input zugeführt werden.

TimeseriesGenerator

Um die in der Einleitung des Kapitels beschriebene Aufgabe zu lösen, kann ein Werkzeug von Keras verwendet werden, dem TimeseriesGenerator. Bei dessen Funktionsaufruf werden mindestens drei Parameter erwartet. Pflichtfelder sind die Angabe der x- und y-Daten, also den Input- und Zielwerten, sowie der Sequenzlänge. Außerdem gibt es die Möglichkeit die Batch Size festzulegen, welche per Default auf 128 gestellt ist. Sie gibt an, aus wie vielen Sequenzen ein Batch (Häppchen) besteht. Die Funktion gibt schließlich ein Array aus, welches alle Batches beinhaltet. Ohne speziellen Anpassungen an dem Datensatz werden die Daten durch den Generator nach der One-Step-Ahead Methode vorbereitet. Wie der Name bereits andeutet, werden die Sequenzen so angeordnet, dass der y-Wert sich einen Schritt nach den x-Werten befindet. Der Aufbau der Sequenzen wird in den Versuchen vorgestellt.

3.1 FFNN

Eine Struktur, welche in den Versuchen eingesetzt wird, wird als Feedforward Neural Network, kurz FFNN, bezeichnet. Dies sind einfache Netzstrukturen, bei welchen Signale nur in eine Richtung, vom Input bis zum Output weitergeleitet werden. [14, 34] Der grundlegende Aufbau ist in Kapitel 2.4 bereits beschrieben worden. Eine schematische Darstellung ist dort ebenfalls in Abbildung 2.6 zu sehen. In dieser Struktur werden ausschließlich dichte Schichten, in Keras Dense-Schicht bezeichnet, verwendet. Anhand der Tabelle 2.3 kann abgelesen werden, dass bei Regressionsaufgaben in der letzten Schicht lediglich ein Neuron eingesetzt wird. Ansonsten variiert die Anzahl an Schichten und Neuronen.

Dass Zeitreihenvorhersagen mit FFNN's durchgeführt und für Vorhersagen eingesetzt werden können, zeigt sich unter anderem in Arbeiten wie [14, 17, 36].

3.2 LSTM

Entsprechend [4, 11], werden für die Verarbeitung sequentieller Daten Netzstrukturen aus der Familie der rekurrenten neuronalen Netze, kurz RNN, eingesetzt. Primär wird hierbei von den Long-Short-Term-Memory Modellen, kurz LSTM, gesprochen. [2, 14, 32, 34] Der Unterschied zu Feedforward-Netzen ist, dass es innerhalb der versteckten Schichten zusätzliche Verbindungen zwischen Neuronen derselben Schicht und Neuronen vorangegangener Schichten gibt, sodass Informationen auch rückwärts durch das Netz gelangen und eine Art Gedächtnis nachgebildet werden. [4, 34]

Abbildung 3.1 zeigt den Aufbau eines LSTM-Moduls, welches in einer Kette aus weiteren Modulen desselben Typs geschaltet ist. Gewöhnliche RNN's neigen dazu, dass Informationen aus weit zurückliegenden Sequenzen nicht mehr berücksichtigt quasi. vergessen werden. [14, 23] Die durch Hochreiter und Schmidhuber [15] eingeführte LSTM-Technik setzt an diesem Punkt an und löst durch eine Art Lang-Kurzzeitgedächtnis dieses Problem. [14, 15, 23]

Der Kern eines LSTM-Moduls besteht aus dem Zell-Status, welcher im Diagramm durch die obere schwarze Linie dargestellt ist. Sie funktioniert als eine Art Laufband, auf welcher die Informationen durch die Module geleitet werden. Über drei Gates, zu erkennen an

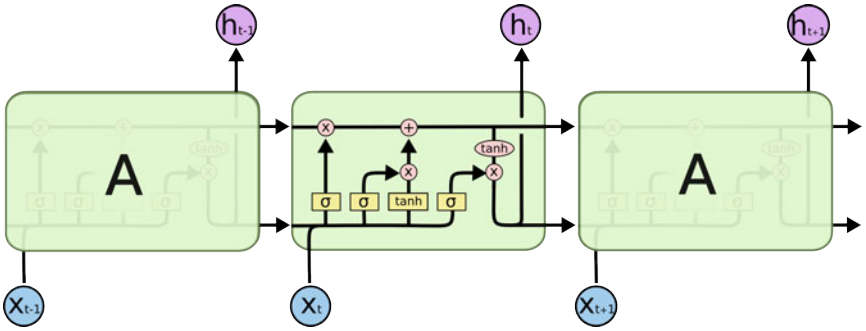


Abbildung 3.1: Aufbau eines sich wiederholenden LSTM-Moduls. Quelle [23].

den gelb-hinterlegten Rechtecken mit dem σ-Symbol, werden Signale dem Informationsfluss hinzugefügt oder aus diesen entfernt. Die Wichtigkeit der Informationen, bestehend aus dem Zustand der vorherigen Zelle h_{t-1} und dem Input X_t, werden hierbei durch Sigmoid-Berechnungen reguliert. Dem Fluss an Daten wird ein Output zwischen 0 und 1 hinzugefügt und aus diesen Werten der neue Zell-Status bestimmt. [23]

Bei den LSTM-Netzen, die in dieser Arbeit verwendet werden, wird an letzter Stelle eine Dense-Schicht mit einem einzelnen Neuron eingesetzt, um die Informationen in einem Ausgabewert zu komprimieren.

3.3 Autoencoder

Ein weiteres genutztes Vorhersageverfahren basiert auf einer Encoder-Decoder-LSTM-Struktur, welche unter anderem in [9] verwendet wird. Diese Struktur wird im Folgenden als Autoencoder bezeichnet. Encoder-Decoder zeichnen sich dadurch aus, dass die Encoder-Funktion die Eingabedaten in eine andere Repräsentation umwandelt, auch Dimensionsreduktion genannt. [14] Bei diesem Schritt sollen durch die Reduzierung der Daten die wichtigen Bestandteile hervorgehoben werden. Eine Decoder-Funktion führt diese neu gewichtete Repräsentation wieder zurück in die Ausgangsform. [11] Nützlich ist diese Vorgehensweise unter anderem bei kategorischen Variablen. [14] Abbildung 3.2 zeigt das Prinzip des Encoder-Decoder-Verfahrens.

Der Aufbau der Autoencoder-Architektur orientiert sich in dieser Arbeit an der Herangehensweise von: A Gentle Introduction to LSTM Autoencoders Quelle [5], bei welcher der Netzstruktur LSTM-Schichten hinzugefügt werden. In dieser Arbeit sind die

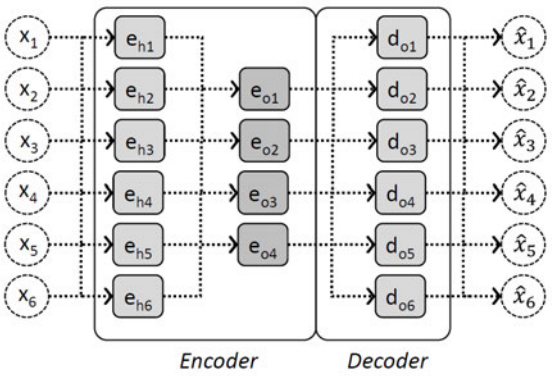


Abbildung 3.2: Aufbau einer Encoder-Decoder-Struktur. Quelle [14] S. 250.

Autoencoder-Modelle so konzipiert, dass am Ende der Netzstruktur eine Sequenz an Schätzwerten ausgegeben wird. Der Mittelwert aus den einzelnen Sequenzwerten bilden dann den tatsächlichen Prognosewert.

3.4 CNN

Die letzte Architektur, welche in dieser Arbeit zum Vergleich von Vorhersagen herangezogen wird, ist ein Convolutional Neural Network, kurz CNN. Zwar werden hauptsächlich Bildanalysen mit ihnen durchgeführt, doch auch Zeitreihen können mit ihnen verarbeitet werden [14].

In der Vorverarbeitung werden Bilder in zweidimensionalen Bildmatrizen, auch Feature Map genannt, abgespeichert, in welchen jedes Pixel durch einen Wert repräsentiert wird. Würden diese Daten im neuronalen Netz verarbeitet werden, müssten Gewichte für jeden Wert aus der Matrix während des Trainings angepasst werden, pro Bild. Neuronen, die in konvolutionalen Schichten Features herausfiltern sollen (z.B. schwarze Pixel oder ähnliche Werte), werden Kernels genannt. Sie besitzen eine Fläche von X mal X Pixel und scannen mit diesem Format schrittweise die Bildmatrix. Für jedes Kernel-Fenster wird ein Schätzwert berechnet, welche zusammengefasst die Responsive Map erzeugen. Im Anschluss an diese Schicht wird die Technik Max Pooling eingesetzt, welche die Responsive Map nach Maximalwerten selektiert und in einer kleineren Matrix abspeichert. So können erste Features erkannt werden, bevor weitere konvolutionale Schichten dieses

Verfahren wiederholen, die wieder neue Features in den Daten erkennen. Abbildung 3.3 visualisiert diesen Aufbau.

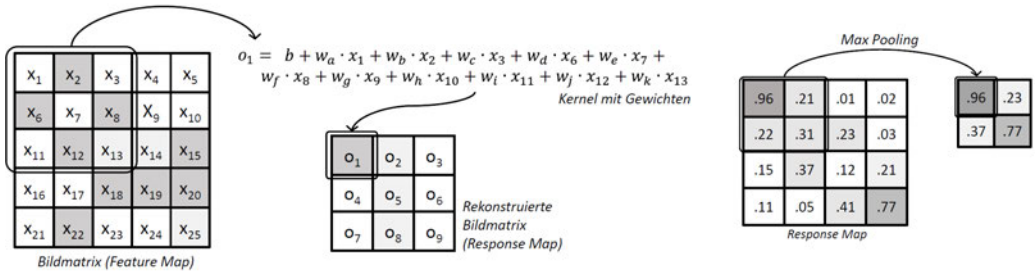


Abbildung 3.3: Funktionsweise der Datenverarbeitung von konvolutionalen Schichten bei zweidimensionalen Daten. Quelle [14] S. 205.

Anstelle von 2D-Bildmatrizen werden bei Zeitreihen eindimensionale Arrays verarbeitet. [11, 14] Keras bietet hierfür eine eigene Schicht an. Abbildung 3.4 zeigt den ähnlichen Aufbau wie bei der Verarbeitung zweidimensionaler Daten. Die Netzstruktur schließt mit einer Dense-Schicht ab, welche wieder nur ein Neuron enthält.

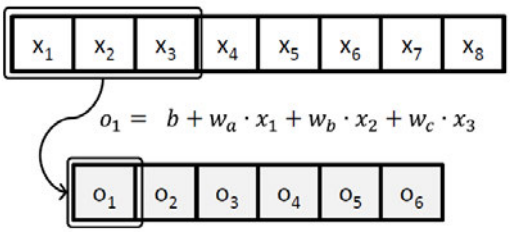


Abbildung 3.4: Funktionsweise der Datenverarbeitung von konvolutionalen Schichten bei eindimensionalen Daten. Quelle [14] S. 207.

4 Konzept

Der Vergleich der Vorhersageverfahren findet anhand drei praktischen Beispielen statt. Die Untersuchung, ob mithilfe neuronaler Netze ein Trend zum vermehrten Zubau von erneuerbaren Energien zu erkennen ist, wird im Hauptexperiment, im dritten Versuch, untersucht. Miteinander verglichen werden die Architekturen FFNN, LSTM, Autoencoder und CNN.

Kapitel 5.1: Im ersten Versuch wird der Verlauf anhand einer linear ansteigenden Zahlenreihe vorhergesagt. Hierbei wird ein univariater Datensatz verwendet.

Kapitel 5.2: Im zweiten Versuch wird der periodische Verlauf einer Sinuskurve mit Verwendung eines multivariaten Datensatzes prognostiziert.

Kapitel 6: Im dritten Versuch wird untersucht, ob durch Vorhersageverfahren ein Trend hin zum vermehrten Zubau erneuerbarer Energien in Deutschland zu erkennen ist. Zum Einsatz kommen univariate und multivariate Datenstrukturen.

4.1 Methodisches Vorgehen

Die Versuche haben grob den gleichen methodischen Aufbau. Zuerst wird der Versuch vorgestellt und das Ziel definiert. Anschließend wird die Vorverarbeitung der Daten beschrieben. Darauf folgt der Abschnitt der Testprognosen und abschließend die Evaluation des Versuches. Hierbei werden anhand des MAE die Testprognosen der Modelle mit den echten Testwerten verglichen.

Aufgrund der Besonderheit des dritten Versuches hinsichtlich der Energieprognose ist Kapitel 6 in mehrere Unterkapitel unterteilt. Zu Beginn werden die Daten genauer vorgestellt, analysiert und eine Erklärung abgegeben, weshalb dieser Versuch in zwei Teilen durchgeführt wird. In Kapitel 6.4 wird das speziellere und für beide Versuchsteile übergreifende Vorgehen vorgestellt. Anschließend erfolgen die beiden Versuchsdurchführungen

in Kapitel 6.5 und 6.6. Hierbei entspricht die Gliederung dem Aufbau zuvor. Zusätzlich gibt es jeweils noch ein Unterkapitel zur Prognose von unbekannten Daten, welche über den Datensatz hinausgehen. Bewusst wird hier (Kapitel 6.5.3 und Kapitel 6.6.3) ein längerer Zeitraum als bei der Testprognose (Kapitel 6.5.1 und Kapitel 6.6.1) gewählt. Hierbei wird untersucht, wie weit die Vorhersagen fortgeführt werden können, bevor die Modellprognosen unrealistische Ergebnisse aufweisen.

In allen Versuchen werden Sequenzen als Dateninput verwendet, welche durch den TimeseriesGenerator generiert werden. Angaben zu Parametern, wie z.B. der Epochenanzahl und den Verlustfunktionen, sind in den jeweiligen Versuchen aufgeführt. Entsprechend der Vorgehensweise in [14] werden die Daten in Trainings- und Testdaten unterteilt, bei Besonderheiten ist es entsprechend angegeben.

4.2 Technische Umgebung

Die technische Umsetzung basiert auf der Programmiersprache Python, da es für sie viele unterstützende Bibliotheken gibt, die zur Datenanalyse und für Aufgabenstellungen im Bereich Machine Learning eingesetzt werden können. Für das Verarbeiten der Daten werden NumPy und speziell Pandas verwendet, welche umfangreiche Operationen bzgl. der Zeitreihenanalyse ermöglichen. Für die Implementierung von neuronalen Netzen wird die Open-Source-Plattform TensorFlow und für die Netzstrukturen die Deep Learning Bibliothek Keras eingesetzt, welche oft gemeinsam genutzt werden. Außerdem wird bei einzelnen Funktionen auf scikit-learn zurückgegriffen. Um Abbildungen zu erstellen, wird zum einen Matplotlib verwendet und zum anderen Tikzplotlib, um Plots in das Latex-Format zu konvertieren. Eine Übersicht der wichtigsten Tools mit entsprechender Version ist in der folgenden Tabelle 4.1 aufgelistet.

Tabelle 4.1: Wichtige Tools mit Angabe der genutzten Version.

Tools	Version	Tools	Version
Python	3.8.5	Keras	2.6.0
NumPy	1.21.2	scikit-learn	1.0.2
Pandas	1.3.4	Matplotlib	3.5.0
TensorFlow	2.6.0	Tikzplotlib	0.10.1

5 Zeitreihenvorhersage

Die Analyse der Architekturen zur Zeitreihenanalyse beginnt mit dem ersten Versuch, der Vorhersage einer Zahlenreihe anhand eines kleinen, univariaten Datensatzes. Darauf folgt der zweite Versuch, die Vorhersage einer Sinuskurve, bei welchem der Datensatz eine multivariate Struktur aufweist und somit komplexer ist. Aufgrund der Komplexität der Daten im dritten Versuch wird dieser in einem eigenen Kapitel 6 untersucht. Verglichen werden die Architekturen FFNN, LSTM, Autoencoder und CNN.

5.1 Vorhersage einer Zahlenreihe

Im ersten Versuch ist es das Ziel, eine Zahlenreihe mit 50 Werten (Frequenz von 1) um erst 10 und dann 20 Werten vorherzusagen. Der verwendete Datensatz, Zahlen von 0 bis 49, wurde durch NumPy erzeugt.

5.1.1 Vorverarbeitung

Vor dem Training werden die Daten zu 80 % in Trainings- und 20 % in Validierungsdaten unterteilt. Die im Anschluss des Trainings vorherzusagenden Zahlen, einmal [50 - 59] und [50 - 69], dienen gleichzeitig als Testdaten und sind im zuvor erzeugten Datensatz nicht mit inbegriffen. Da die Zahlenreihe aus fortlaufenden Zahlen besteht, können die Prognosen parallel mit den echten Werten, den Testdaten, verglichen werden. Die Modelle werden jeweils nur mit einer Spalte an Daten gefüttert, weshalb es sich um einen Versuch mit univariaten Daten handelt. Die folgende Auflistung zeigt die Anordnung der Daten.

$$\begin{aligned} \text{all data} &= [0, 1, 2, 3, 4, 5, \ldots, 45, 46, 47, 48, 49] \\ \text{prediction 1} &= [50, 51, 52, 53, 54, 55, 56, 57, 58, 59] \\ \text{prediction 2} &= [50, 51, 52, 53, 54, \ldots, 65, 66, 67, 68, 69] \end{aligned}$$

Abbildung 5.1 skizziert die allgemeine Vorgehensweise dieses Versuches, welche im Groben auch für die weiteren Versuche zutreffend ist. Die Standardisierung, bei welcher die Daten in einen kleinen Wertebereich transformiert werden, findet bei diesem Versuch nicht statt, da der Datensatz insgesamt sehr klein ist und in den Daten keine großen Werte enthalten sind.

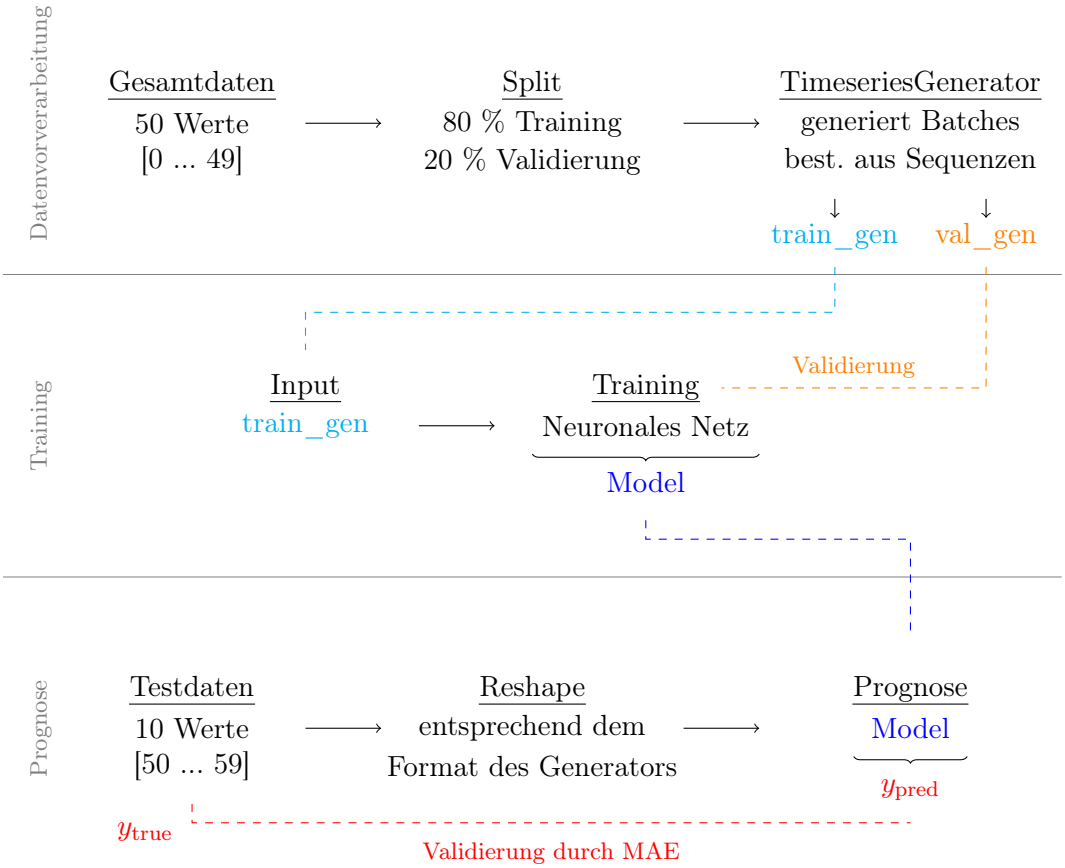


Abbildung 5.1: Darstellung der Vorgehensweise, beginnend bei der Datenvorverarbeitung bis hin zur Überprüfung der Prognoseergebnisse.

Die zu Beginn des Kapitels 3 beschriebene Vorgehensweise, die Daten in Sequenzen zu unterteilen, wird auch in diesem Versuch angewandt. Eingesetzt wird dabei der TimeseriesGenerator, welcher ebenfalls in dem Kapitel beschrieben wurde und die Daten in das gewünschte Format bringt. Für die Fenstergröße, welche angibt, wie weit die Daten in die Vergangenheit reichen, bzw. wie viele Schritte zurückgeblickt werden soll, ist der Wert vier gewählt worden. Die Inputsequenzen bestehen aus der Anzahl an Werten entsprechend der Fenstergröße und haben den Aufbau: $[x_0, x_1, x_2, x_3]$. Das Netz soll an-

hand der Inputsequenzen erkennen, dass diese in einer festgelegten Frequenz vorliegen und dementsprechend eine Prognose auf den darauffolgenden Wert $y = x_3 + Frequenz$ abgeben. Aufgrund der geringen Datenmenge ist für diesen Versuch die Batch Size acht gewählt worden, was bedeutet, dass ein Batch aus acht Inputsequenzen besteht. Tabelle 5.1 zeigt einen Auszug des ersten Batches, welcher vom Generator erstellt wird. Die gesamten Trainings- und Validierungsdaten werden in Batches unterteilt und als Train- und Val-Generator abgespeichert, welche dem neuronalen Netz im nächsten Schritt als Input dienen.

Tabelle 5.1: Vereinfachte Darstellung eines generierten Batches.

x_{0-3}	y
0, 1, 2, 3	4
1, 2, 3, 4	5
...	...
6, 7, 8, 9	10
7, 8, 9, 10	11

Für das Training sind einige Randparameter wichtig. So wird in allen Netzen der MSE als Verlustfunktion und der Adam-Optimierer verwendet, begründet durch Erklärungen in Kapitel 2.3.1 und 2.4.2. Die Trainings haben eine Dauer von 100 Epochen. Dies ist eine zu hohe Epochenanzahl bei diesem kleinen Datensatz, jedoch sollen in erster Linie alle Vergleiche mit denselben Randbedingungen stattfinden. Zudem betrug die Dauer des Trainings nur ca. 10 Sekunden, sodass die Anzahl an Epochen keinen groß zu berücksichtigen Einfluss auf die Trainingsdauer genommen hat.

Nach Abschluss des Trainings wurden die 10 bzw. 20 zu prognostizierenden Werte in dieselbe Form gebracht, wie die vom TimeseriesGenerators generierten Inputsequenzen. Dieser Schritt ist deshalb notwendig, da das Modell während des Trainings mit einem bestimmten Format der Daten antrainiert wurde und dieser Aufbau vom Modell wieder erwartet wird. Das Modell erhält als Input eine Sequenz aus den letzten vier Werten aus dem Datensatz, welche lautet [46, 47, 48, 49] und gibt daraufhin den nächstfolgenden Wert als Prognose zurück. Dieser Schritt wird je nach Anzahl der Vorhersagen wiederholt. Hierbei sei zu beachten, dass die jeweiligen prognostizierten Werte nicht mit in die Inputsequenzen eingefügt werden, sondern separat in einem Dataframe hinsichtlich der Visualisierung gespeichert werden. Als wichtiger Hinweise: Die Vorhersage basiert dementsprechend nicht auf der eigenen Prognose, was im dritten Versuch noch relevant wird.

5.1.2 Testprognosen

Da die Gewichte in den neuronalen Netzen anfangs zufällig initialisiert werden, gleicht ein Modell nach Abschluss des Trainings keinem Zweiten. Es ist wahrscheinlich, dass der Gradientenabstieg in einem lokalen (und nicht in einem globalen) Minimum endet. Je nach Architektur sind hierbei erhebliche Unterschiede beobachtet worden. Demonstriert wird dieser Effekt anhand von LSTM-Modellen. In Bezug auf Grafik 5.2 sind drei aufeinanderfolgende Trainings und somit drei verschiedene Modelle mit einer LSTM-Schicht durchgeführt worden, im folgenden Modell 1, Modell 2 und Modell 3 bezeichnet. Die Randbedingungen waren dieselben und auch der Aufbau des Netzes war identisch. Die Ausgangslage war also jeweils gleich. Anhand der Vorhersage und der Validierung mit den tatsächlichen Werten, kann gesagt werden, dass für dieses Modell ein Trainingsdurchlauf nicht ausreicht, um ein vernünftiges Ergebnis (Modell) zu erhalten. So schwankte der MAE bei der Vorhersage von 10 Werten zwischen 0.11 und 2.7. Bei weiteren Trainings lag dieser zwischendurch sogar bei acht.

Diese Vorgehensweise, das mehrmalige Starten eines Trainings, um die besten Modelle für diesen Vergleich heranziehen zu können, wurde bei allen Architekturen angewendet.

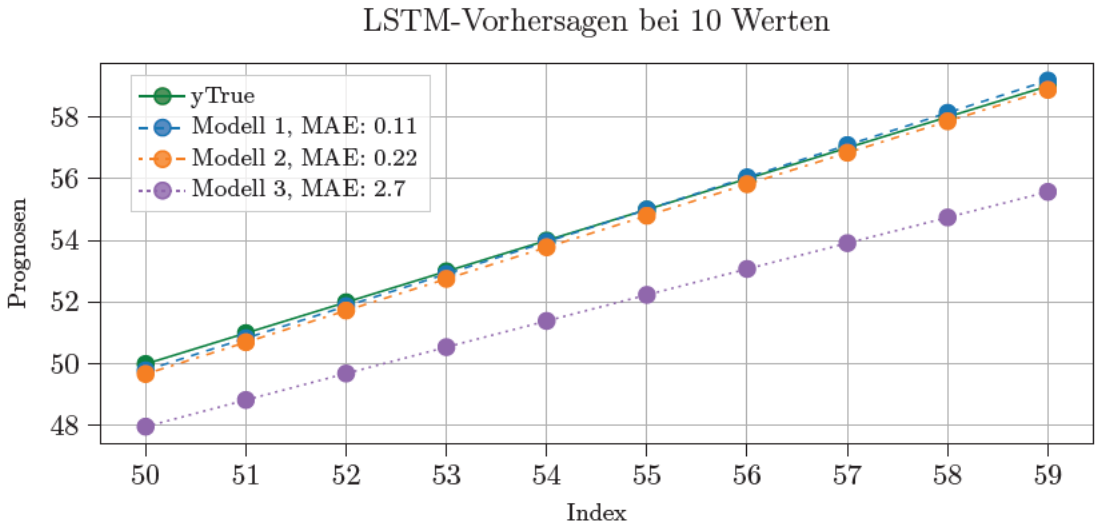


Abbildung 5.2: Vergleich der Vorhersage von 10 Werten anhand von drei LSTM-Modellen mit gleicher Netzstruktur.

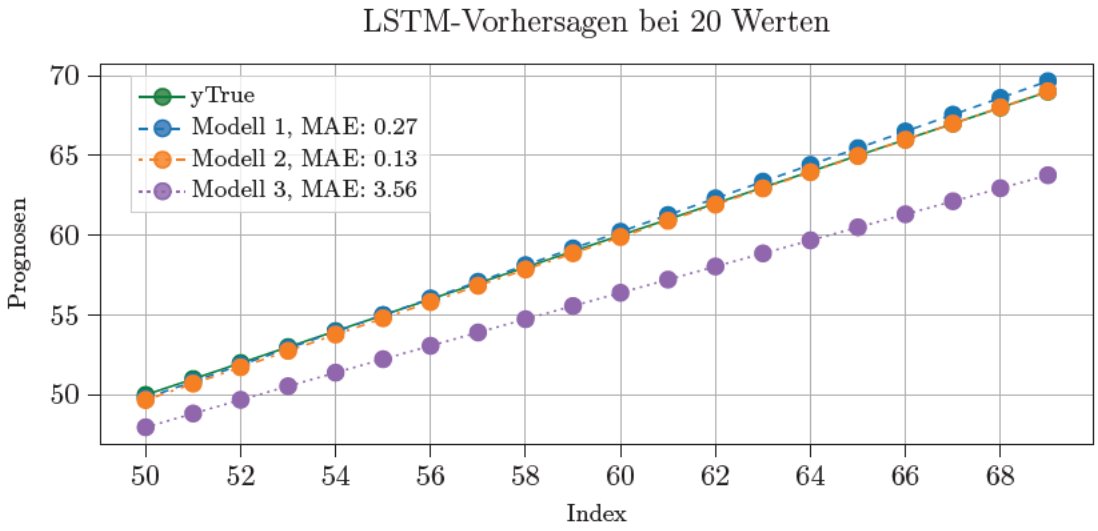


Abbildung 5.3: Vergleich der Vorhersage von 20 Werten anhand von drei LSTM-Modellen mit gleicher Netzstruktur.

Interessant zu sehen ist die Betrachtung des errechneten MAE zwischen 10 und 20 prognostizierten Zahlen. Den Werten aus Grafik 5.2 nach zu urteilen, wurde bei der Verwendung von Modell 1 ein MAE von 0.11 errechnet. Aus dem Graphen kann jedoch zusätzlich interpretiert werden, dass der MAE alleine nicht zwingend ausreicht, um das beste Modell bestimmen zu können. Es kommt auf die Aufgabenstellung an. Bei der Kurve, bestehend aus den prognostizierten 10 Werten, schneidet diese die Gerade (yTrue) und ist somit nah an den echten Werten. Der Verlauf der Kurve spiegelt sich im errechneten MAE wider. Neben der Kurve von Modell 1 verläuft die von Modell 2, welche ebenfalls an die optimale grüne Gerade angrenzt, doch diese nicht schneidet. Modell 3 verläuft gänzlich abgeschlagen und wird nicht weiter beachtet.

Mit Blick auf Abbildung 5.3 ist nun zu erkennen, dass das Modell mit dem niedrigsten MAE, im Gegensatz zu 10 prognostizierten Werten, bei 20 Werten nicht mehr das beste Modell darstellt. Der MAE des 1. Modells liegt nun bei 0.27 und der des 2. Modells bei 0.13. Während sich die Kurve von Modell 1 weiter entfernt, nähert sich die Kurve von Modell 2 weiter der Geraden an. Es ist davon auszugehen, dass nach weiteren 10 prognostizierten Werten (30 insgesamt) der MAE noch weiter sinken wird. Aus den Ergebnissen kann abgeleitet werden, dass je nach Aufgabenstellung (z.B. bei 10 oder 20 Vorhersagen) ein auf die Aufgabe zugeschnittenes Modell für die Vorhersage genutzt werden muss.

Die oben beschriebene Vorgehensweise wurde bei allen Netz-Architekturen angewandt. Verwendet wird das jeweilige Modell, welches bei 20 prognostizierten Werten den niedrigsten MAE-Score erreicht hatte. Zu beachten ist, dass die Größe von 20 Werten die Hälfte des Trainingsdatensatzes ausmacht, welcher zum Antrainieren verwendet wurde.

5.1.3 Evaluation

Abbildung 5.4 zeigt die verschiedenen Architekturen im direkten Vergleich bei 10 prognostizierten Werten und Abbildung 5.5 bei 20 Werten. Auffällig ist, dass die gezeichneten Kurven in den Grafiken alle recht nah beieinander liegen. Dies ist wahrscheinlich auf die weniger komplexe Aufgabenstellung zurückzuführen.

Bei der anschließenden Analyse lassen sich dennoch spannende Ergebnisse beobachten. Spitzenwerte des MAE von unter 0.1, wie vom Autoencoder in Abbildung 5.4, wurden im Zuge weiterer Trainings bei keinem anderen Verfahren erhalten. Dem MAE-Wert als Faktor nach zu urteilen, eignen sich Modelltypen mit LSTM-Strukturen, also dem LSTM- und Autoencoder-Modell, in diesem Prognosebeispiel am meisten. Mit Blick auf Abbildung 5.4 und 5.5, weisen diese Strukturen die niedrigsten MAE-Werte auf.

Aus den ersten beiden Plots in diesem Kapitel, Abbildung 5.2 und 5.3, geht die Vermutung hervor, dass die Steigung der Kurven ein Indikator dafür sein kann, wie gut ein Modell für die Langzeitprognose geeignet ist. Derselbe Rückschluss kann auch nach Betrachtung der Plots 5.4 und 5.5 gezogen werden, da bei größerer (positiver oder negativer) Steigung die Differenz zwischen Kurve und Geraden schneller zunimmt. Im Einzelnen gibt der MAE sicherlich Aufschluss darüber, welches Modell bei entsprechender Aufgabenstellung die beste Wahl ist. Anhand der Ergebnisse können jedoch noch weitere Aspekte betrachtet werden. Nachfolgend wird daher auf die MAE-Werte genauer eingegangen, um im Speziellen zu überprüfen, inwiefern die Differenz der MAE-Werte die Aussagekraft eines Modells beeinflusst. Zunächst werden die Differenzen der unterschiedlichen MAE-Werte berechnet und einander gegenüber gestellt.

An den Differenzwerten ist zu erkennen, dass das Autoencoder-Modell die niedrigste Steigung aufweist. Der MAE, bezogen auf 20 Werte, wird größer, was bedeutet, dass die Kurve des Modells, im Gegensatz zu den anderen Kurven, die Gerade bereits geschnitten hat und im weiteren Verlauf zu noch größeren Fehlerwerten führen wird. Modellkurven werden zwar nicht die Eigenschaft einer Geraden haben, dennoch wurde als Annahme bei 30 prognostizierten Werten die Differenz der ersten beiden Vorhersagen auf den MAE der

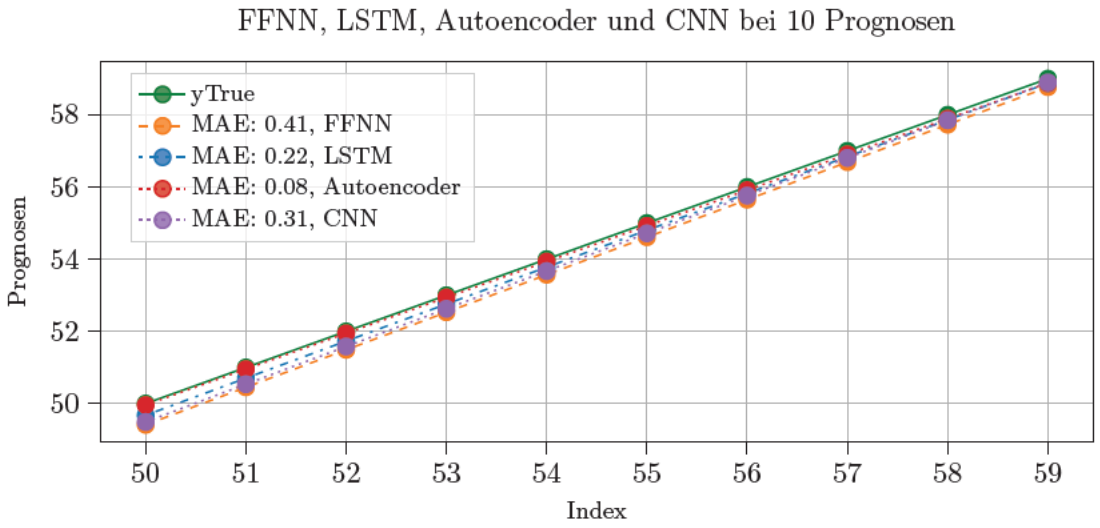


Abbildung 5.4: Vergleich der Vorhersage von 10 Werten anhand verschiedener Modell-Architekturen.

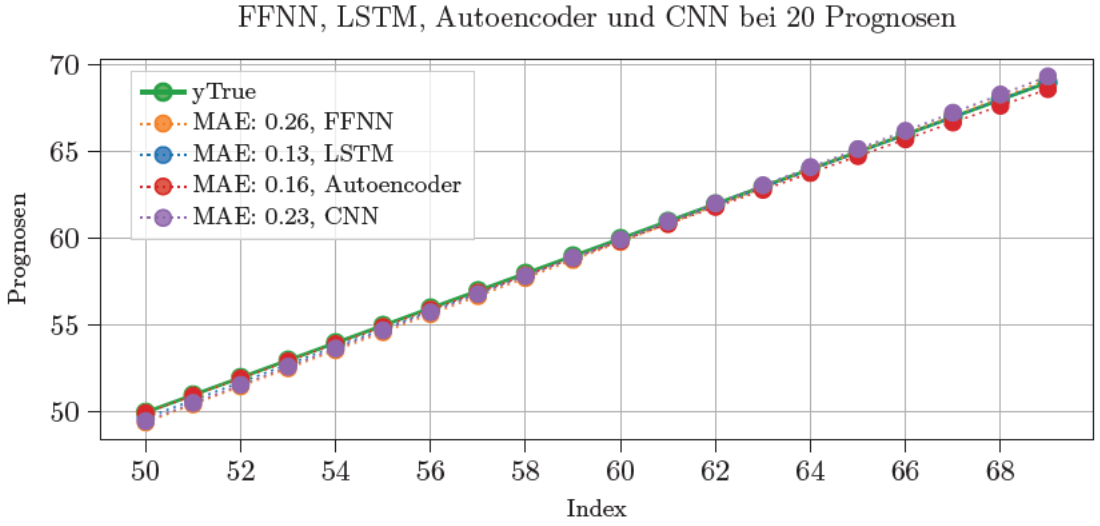


Abbildung 5.5: Vergleich der Vorhersage von 20 Werten anhand verschiedener Modell-Architekturen.

20 Werte hinzuaddiert bzw. abgezogen (beim Autoencoder). Der neue geschätzte MAE-Wert, jeweils abgebildet in der letzten Spalte von Tabelle 5.2, zeigt einen möglichen MAE bei 30 Prognosen des Modells.

Tabelle 5.2: Differenz der MAE-Werte zwischen 10 bzw. 20 und möglichen 30 prognostizierten Werten.

Modeltype	10 Werte	20 Werte	Differenz	MAE bei 30 Werten
FFNN	0.41	0.26	0.15	0.11
LSTM	0.22	0.13	0.09	0.04
Autoencoder	0.08	0.16	0.08	0.24
CNN	0.31	0.23	0.09	0.14

Die MAE-Werte bei 20 Vorhersagen sind mit den geschätzten MAEs bei 30 Werten nicht zu vergleichen. Wieder geht aus diesem Versuch hervor, dass ein Modell für eine bestimmte Aufgabenstellung geeignet sein kann, aber nicht für weitere Vorhersagen allgemein verwendet werden kann.

Stabilität der Trainingsdurchläufe

Während der Trainingsdurchläufe ist die zu Beginn angesprochene Stabilität der Modelle aufgefallen. Zwar wiesen die Netze mit LSTM-Strukturen die geringsten MAE-Werte auf, jedoch sind bei diesen Strukturen Ausreißer ebenfalls am häufigsten aufgetreten. Abbildung 5.6 zeigt den jeweiligen MAE bei zehn aufeinanderfolgenden Trainingsdurchläufen pro Architektur mit demselben Modell. Anhand der Abbildung ist zu erkennen, dass die FFNN- und CNN-Modelle die höchste Stabilität aufweisen und somit aussagekräftiger sind, wenn nur wenige Modelle antrainiert werden. Diese Auffälligkeit wird durch die folgenden Berechnungen verstärkt.

Ähnlich wie zuvor wurden von denselben zehn Modellen (pro Architektur) die Differenzen zwischen dem MAE bei 10 und bei 20 Werten berechnet und davon schließlich der Mittelwert ermittelt. Die Ergebnisse sind in Tabelle 5.3 festgehalten. Zu sehen ist, dass die CNN-Netzstruktur die im Durchschnitt besten Modelle in dieser Versuchsreihe geliefert hat.

Tabelle 5.3: Mittelwerte der errechneten Differenzen zwischen dem MAE bei 10 und bei 20 prognostizierten Werten, von insgesamt zehn Modellen pro Architektur.

	FFNN	LSTM	Autoencoder	CNN
Mittelwert der Differenzen	0.16	0.34	0.28	0.11

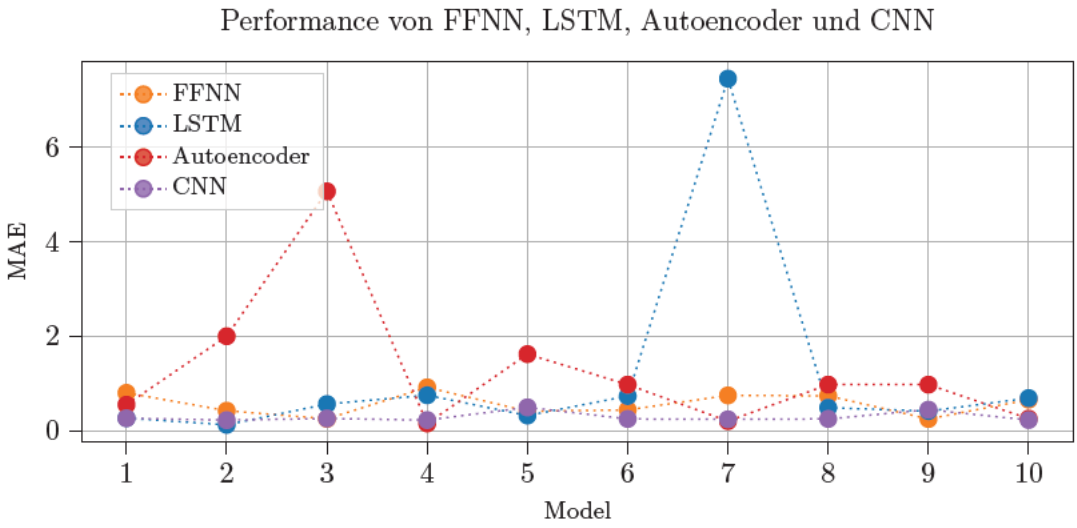


Abbildung 5.6: Vergleich der Modellstabilität zwischen den Modell-Architekturen, anhand der Vorhersage von 20 Werten bei zehn aufeinanderfolgenden Trainingsdurchläufen.

Aus diesem Versuch hat sich ergeben, dass bei dem Fortführen einer Zahlenreihe ein LSTM-Modell bei der Vorhersage von 20 Werten die geringste Abweichung (geringster MAE) gegenüber den Testdaten erzielen konnte. Gleichzeitig sind LSTM-Modelle und Autoencoder, welche ebenfalls auf LSTM-Schichten basieren, die empfindlichsten Modelle gewesen. Bei der Stabilität konnte die CNN-Architektur die im Schnitt besten Ergebnisse erzielen.

5.2 Vorhersage der Sinuskurve

Im zweiten Versuch wird der Verlauf der Sinuskurve prognostiziert. Die Vorgehensweise ist ähnlich gestaltet wie der Aufbau aus dem ersten Versuch (s. Abbildung 5.1). Der genutzte Datensatz (Quelle: Sinus.csv, letzter Zugriff 27.05.2022) besteht aus 470 Zeilen und beinhaltet 21 Spalten. Die ersten 20 Spalten stellen die Sinuskurve mit leichtem Versatz dar. Die letzte Spalte, „target“ genannt, entspricht der Kurve, die von den Modellen vorhergesagt werden soll. Abbildung 5.7 zeigt einen Auszug der Daten. Die Bezeichnungen der Spalten lauten `t0` bis `t19` bzw. die Letzte von ihnen `target`, welches die Zielvariable ist.

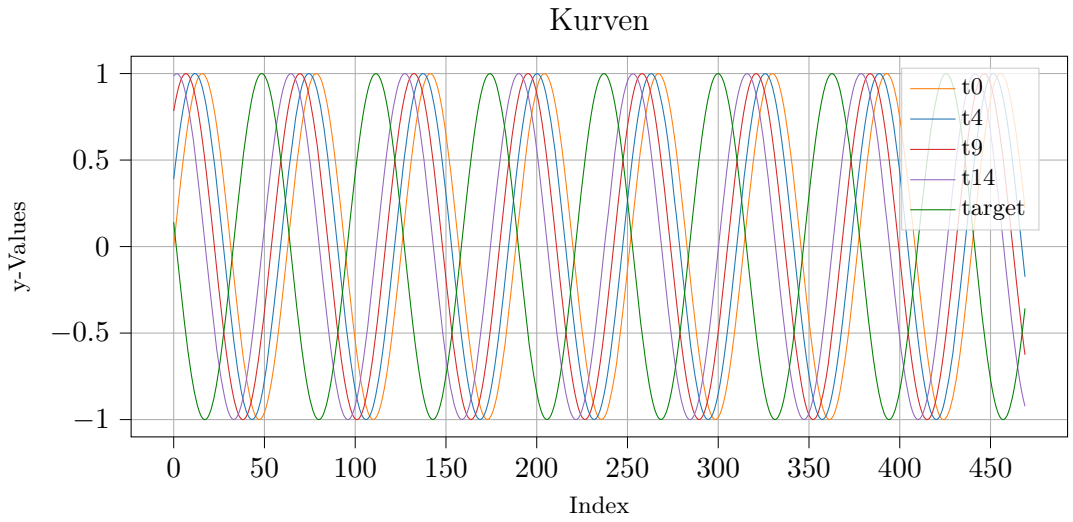


Abbildung 5.7: Darstellung einiger Spalten aus dem Sinus-Datensatz. Neben der Sinuskurve sind leicht versetzte Kurven enthalten.

Gegenüber dem vorherigen Versuch gibt es hierbei zwei wesentliche Unterschiede. Zum einen handelt es sich um eine multivariate Analyse, da der Datensatz aus mehr als einer Spalte besteht und die Modelle mit einer komplexeren Datenstruktur angelernt werden. Zum anderen weisen die Kurven einen periodischen, wiederkehrenden Verlauf auf, weshalb die Prognose nicht im Vergleich zu gänzlich unbekannten Daten durchgeführt wird.

5.2.1 Vorverarbeitung

Da bei ersten Versuchen, mit einem Verhältnis von 80 % Trainings- und 20 % Validierungsdaten, die Ergebnisse nicht zufriedenstellend waren, wurde mit einer Aufteilung von 70 % / 30 % in Trainings- und Validierungsdaten fortgefahren. Aufgrund des periodischen Verlaufs werden die Validierungsdaten neben dem Validieren während des Trainings auch für das Testen der eigentlichen Prognose eingesetzt. Ansonsten sind die Rahmenbedingungen für diesen Versuch wieder identisch zu dem ersten Versuch. Während des Trainings wird der MSE zur Validierung des Verlustwertes und der Adam-Algorithmus für dessen Optimierung verwendet. Wie bei der Prognose einer Zahlenreihe ist auch hier die Erwartung, dass die LSTM- und Autoencoder-Struktur die besseren Prognosen abgeben werden. Aufgrund der unterschiedlichen Modellstabilitäten aus dem ersten Versuch, vorgestellt in Kapitel 5.1.3, wird auch dieser Umstand untersucht.

5.2.2 Testprognosen

Entsprechend der Ergebnisse aus dem ersten Versuch in Abschnitt 5.1.2 wurden zuerst die LSTM-Modelle untersucht und einfache Netzstrukturen mit nur wenigen Neuronen eingesetzt. Das Herantasten an eine geeignete Epochenanzahl und die daraus resultierenden Modellkurven sind in Abbildung 5.8 dargestellt.

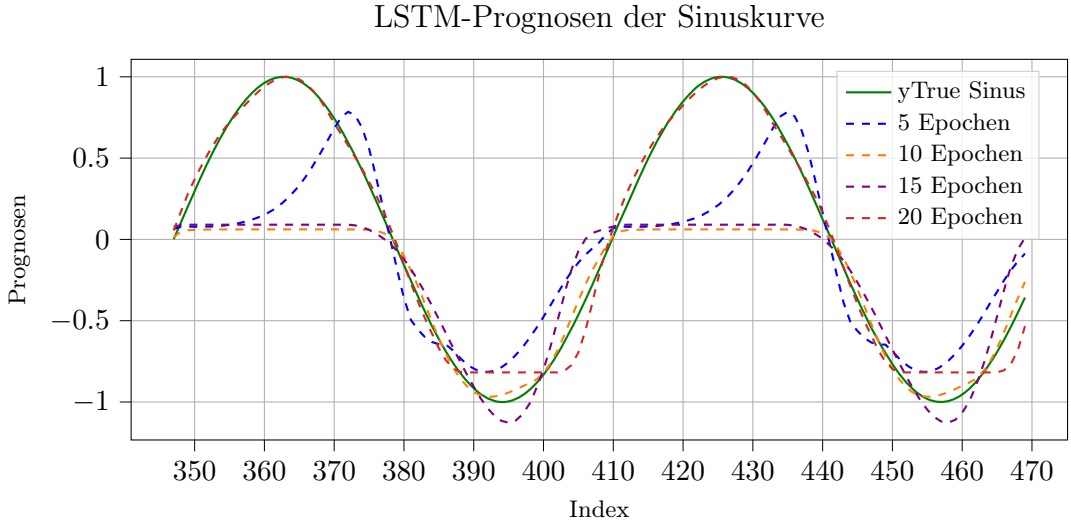


Abbildung 5.8: Prognosen eines LSTM-Modells bei unterschiedlicher Anzahl an Trainingsepochen.

Der Index der x-Achse ergibt sich daraus, dass die prognostizierten 123 Werte aus den Validierungsdaten, bei 347 starten und bei 469 enden. Bei den ersten Trainingsdurchläufen betrug die Epochenanzahl fünf, zehn, 15 und 20. In den folgenden Abbildungen ist die Vergleichskurve wieder in Grün dargestellt. Anhand der Kurvenverläufe ist ein rasanter Anstieg der Prognosequalität bei zunehmender Anzahl an Epochen zu erkennen.

Für den eigentlichen Vergleich der unterschiedlichen Netz-Architekturen sind aufgrund der ersten Ergebnisse für die weiteren Trainings 50 Epochen gewählt worden. Wie im vorangegangenen Versuch wurde der Trainingsprozess eines Modells öfter gestartet, sowie verschiedene Mengen von Schichten und Neuronen getestet, um die Stabilität der Modelle zu überprüfen. Nennenswerte, vergleichbare Ausreißer wie beim LSTM und Autoencoder aus dem Kapitel 5.1, sind hier jedoch nicht aufgetreten. In Grafik 5.9 sind die Prognoseergebnisse der jeweils besten Modelle aufgeführt.

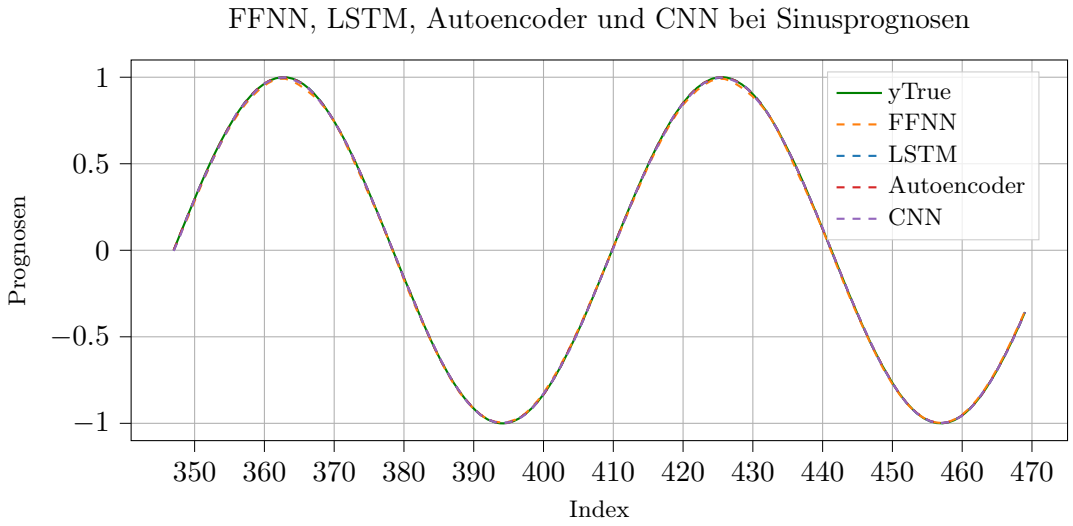


Abbildung 5.9: Vorhersage der Sinuskurve durch verschiedene Modellstrukturen.

5.2.3 Evaluation

Da die Kurven fast vollständig übereinander liegen, ist eine visuelle Bewertung, welches Modell die besten Prognosen ausgegeben hat, nur schwer möglich. Daher wird der MAE zur Überprüfung herangezogen. Wie schon im vorherigen Versuch kann entsprechend Tabelle 5.4 festgestellt werden, dass Architekturen mit beinhaltenden LSTM-Schichten, die niedrigsten Abweichungen gegenüber den Vergleichswerten aufweisen. Mit einem Fehlerwert von 0.0009 schnitt der Autoencoder in diesem Versuch am besten ab.

Tabelle 5.4: MAE zwischen den Vorhersagen und den Testwerten.

	FFNN	LSTM	Autoencoder	CNN
MAE	0.0076	0.0015	0.0009	0.0024

In diesem Versuch konnte festgestellt werden, dass für die Vorhersage von Werten der Sinuskurve die Autoencoder-Architektur aufgrund des geringsten MAEs am besten geeignet ist. Bei der visuellen Auswertung der Kurven war kein Unterschied zwischen den Prognosen der verschiedenen Architekturen zu erkennen.

6 Vorhersage von Energiewerten

Im Vergleich zu den ersten beiden Experimenten ist dieser Versuch wesentlich größer und komplexer. Im Motivationsschreiben zu Beginn dieser Arbeit wurde auf die Klimasituation weltweit und in Deutschland aufmerksam gemacht. Einigkeit bestand darin, dass die Nutzung fossiler Energieträger rapide abnehmen und sich der Anteil erneuerbarer Energien dementsprechend steigern muss. Anhand eines Datensatzes wird untersucht, ob durch Prognosen, durchgeführt mit neuronalen Netzen, ein Trend hinsichtlich des vermehrten Zubaus von erneuerbaren Energien in Deutschland zu erkennen ist. Neben dieser Frage wird geprüft, ob die Modelle überhaupt realistische Prognosen erzielen und wenn ja, ob LSTM-Modelle bessere Ergebnisse liefern im Vergleich zu den restlichen Architekturen FFNN, Autoencoder und CNN.

6.1 Datensatz

Herausgeber der Daten ist die ENTSO-E Transparency Platform, welche in Europa öffentlich zugängliche Daten der Energiegewinnung bereitstellt. Auch andere Arbeiten wie [36] verwenden Daten dieser Plattform. In Kapitel 2.2.2 wurde darauf hingewiesen, dass die Qualität der Daten für ML-Analysen wichtig ist. Deswegen werden bereits im Vorfeld Daten aussortiert und somit nicht alle Energiedaten Deutschlands von der Webseite verwendet. Wird nämlich der Datensatz vor dem Jahr 2015 betrachtet, weisen die Daten insgesamt zu viele Lücken auf, um sie durch Imputations-Verfahren ergänzen zu können. Ab dem Jahr 2015 sind die Daten so gut wie vollständig vorhanden, weshalb der Zeitstempel 01.01.2015 00:00:00 den ersten Wert des Datenfensters markiert. Als abschließenden Wert des Datenfensters wurde der 31.12.2021 23:45:00 ausgewählt, da dieser den letzten Zeitstempel im Jahr 2021 darstellt. Zudem könnten die Energiedaten ab Beginn des Jahres 2022 durch den Einfluss des Krieges in der Ukraine verfälscht werden. Daher werden diese Daten nicht berücksichtigt. Der exportierte Datensatz beinhaltet somit Daten aus

sieben Jahren, von Beginn 2015 bis Ende 2021. Neben dem Index und Zeitstempel besteht der Datensatz aus weiteren 21 Spalten, eine pro Energietyp, mit insgesamt 245.500 Zeilen. Die Einträge liegen in einem 15 Min. Intervall bzw. Frequenz vor. Die Einheit der Daten ist in Megawatt [MW] angegeben. Aufgrund der Formatierung im Datensatz werden die Energiebezeichnungen in Englisch angegeben.

Pandas bietet umfangreiche Möglichkeiten, Daten zu verarbeiten und anzeigen zu lassen. So können z.B. die Mittelwerte aller Spalten (Energieträger) ausgegeben werden, wie in Tabelle 6.1 dargestellt ist. und bereits einer ersten Übersicht der vorhandenen Energieträger liefert. Die Tabelle zeigt außerdem die Aufteilung der Spalten in erneuerbare (Renewable) und fossile (Conventional) Energien. Die Bezeichnung *fossile Energien* wird im Folgenden zusammenfassend für die umweltschädlichen Energieträger verwendet. Neben den 17 aufgelisteten Energiearten wurden weitere vier Spalten aus dem Datensatz entfernt, da die einzelnen Zellen immer den Wert 0 enthielten oder ohne Eintrag (n/e) waren. Diese lauten: Fossil Oil Shale, Fossil Peat, Hydro Pumped Storage (Consumption) und Marine. Zusätzlich wurden zwei Spalten für die erneuerbaren bzw. fossilen Energien hinzugefügt, bei welchen die Einträge auf den aufsummierten Werten der jeweiligen Spalten basieren.

Tabelle 6.1: Aufteilung und Vorstellung der erneuerbaren bzw. fossilen Energieträger.

Renewable	Mean [MW]	Conventional	Mean [MW]
Biomasse	4491	Fossil Brown Coal	13135
Geothermal	19	Fossil Coal derived Gas	232
Hydro Pumped Storage	1137	Fossil Gas	4463
Hydro Run of River	1660	Fossil Hard Coal	7073
Hydro W. Reservoir	108	Fossil Oil	31
Other Renewable	147	Nuclear	8245
Solar	4576	Other	2278
Wind Offhore	2144	Waste	420
Wind Onshore	9795		
Renewable	24080	Conventional	36160

Der genutzte Datensatz besteht somit aus insgesamt 20 Spalten: Dem Zeitstempel, den 17 Energieträgern und zwei Spalten, welche die erneuerbaren und fossilen Energien zusammenfassen. In Tabelle 6.2 ist ein Ausschnitt der Daten aufgelistet. Die Nullen in der

Solar-Spalte kommen dadurch zustande, dass um diese Uhrzeit kein Solarstrom erzeugt werden kann.

Tabelle 6.2: Darstellung einiger Spalten aus dem Datensatz.

Index	Zeitstempel	Fossil Brown Coal	Nuclear	Solar	Wind Onshore
0	2015-01-01 00:00:00	15859.0	10742.0	0	8113.0
1	2015-01-01 00:15:00	15803.0	10585.0	0	8092.0
2	2015-01-01 00:30:00	15649.0	10643.0	0	8161.0
3	2015-01-01 00:45:00	15438.0	10872.0	0	8146.0
4	2015-01-01 01:00:00	15552.0	11089.0	0	8183.0
...	...	...	...	...	...
245495	2021-12-31 22:45:00	3717.0	3677.0	0	26313.0
245496	2021-12-31 23:00:00	3663.0	3397.0	0	26018.0
245497	2021-12-31 23:15:00	3626.0	3207.0	0	25538.0
245498	2021-12-31 23:30:00	3620.0	3162.0	0	25535.0
245499	2021-12-31 23:45:00	3591.0	3132.0	0	25731.0

6.2 Datenanalyse

Mit Blick auf das Ziel, einen möglichen Trend durch Vorhersagen erkennen zu können, stellt sich die Frage, ob es in den Daten nicht bereits einen Trend zum vermehrten Zubau erneuerbarer Energien gibt. Denn nach den Recherchen, zusammengefasst im Einleitungs-Kapitel 1, sollte dies der Fall sein.

Um dem nachzugehen und ein Verständnis für die Daten zu bekommen, werden zunächst die Energieträger mit den höchsten Mittelwerten über den gesamten Zeitraum einander gegenübergestellt. Von den erneuerbaren Energien sind dies Wind OnShore, die Energieerzeugung durch Windkraftanlagen an Land, und Solarstrom. Im Gegensatz dazu stehen Braunkohle und nuklearer Strom aufseiten der fossilen Energien. Der dazugehörige Plot 6.1 zeigt die Mittelwerte der Energieträger pro Monat.

Am auffälligsten ist der periodische Verlauf der Energieerzeugung durch Solarstrom, welcher durch den natürlichen Vier-Jahreszeiten-Zyklus zu erklären ist und die höchsten Peaks jeweils in den Sommermonaten aufweist. Ähnlich verhält sich die Kurve der Wind

OnShore Energieerzeugung, wobei die höchsten Peaks hier zwischen Herbst und Frühling auftraten. Eine weitere Auffälligkeit ist der große Peak der OnShore Kurve Anfang 2020. Dieser ist darauf zurückzuführen, dass es in dieser Zeit eine erhöhte Sturmaktivität über dem Atlantik und Europa gab. [12] Verglichen mit den beiden Kurven der erneuerbaren Energien, verlaufen die Kurven der fossilen Energien etwas konstanter, da natürliche Faktoren wie Wetterverhältnisse keinen großen Einfluss haben.

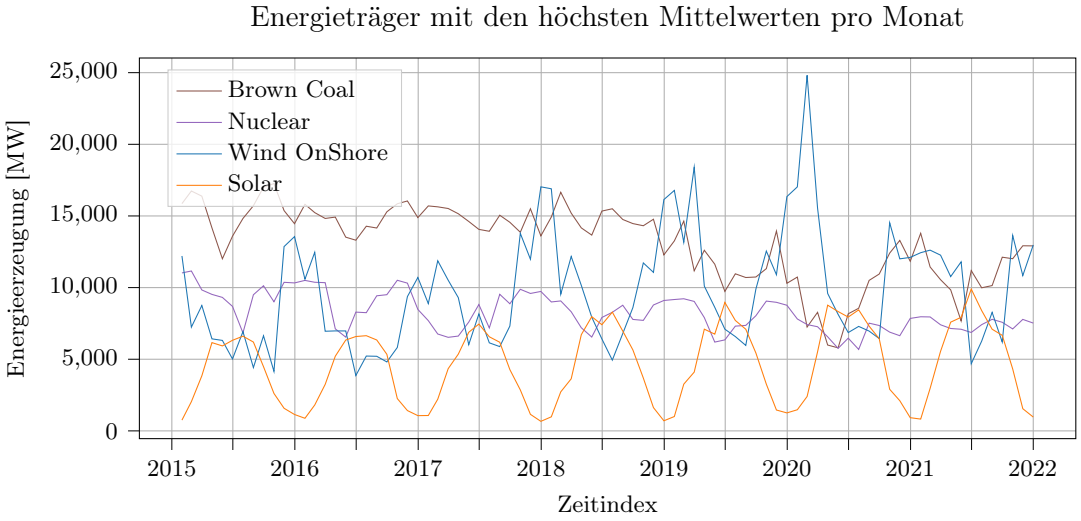


Abbildung 6.1: Darstellung der Energieträger mit den höchsten Mittelwerten zwischen 2015 und 2022.

In den Kurven ist durchaus ein fortlaufender Anstieg von Wind- und Solarstrom zu erkennen genauso wie ein leichter Rückgang von Braunkohle und nuklearem Storm. Bestätigt wird dieser Trend durch das Betrachten der Stromerzeugung pro Jahr. In Abbildung 6.2 sind die Werte für erneuerbare und fossile Energien einander gegenübergestellt.

Ein interessanter Aspekt bei dieser Betrachtung ist, dass die Zu- und Abnahme der Kurven sich über die Jahre von 2015 bis 2020 ebenso verhält. Dieser Trend endet jedoch 2020 und verläuft in die entgegengesetzte Richtung. Bzgl. des Versuchs wird spannend sein, ob dieser Umstand Einfluss auf die Prognoseergebnisse nimmt, denn mit Berücksichtigung der Klimaziele dürfte eine umweltfreundliche Entwicklung erwartet werden.

Für eine vollständige Analyse der Daten müsste jeder Energietyp einzeln oder alle gemeinsam betrachtet werden. Dies würde den Umfang der Arbeit überschreiten.

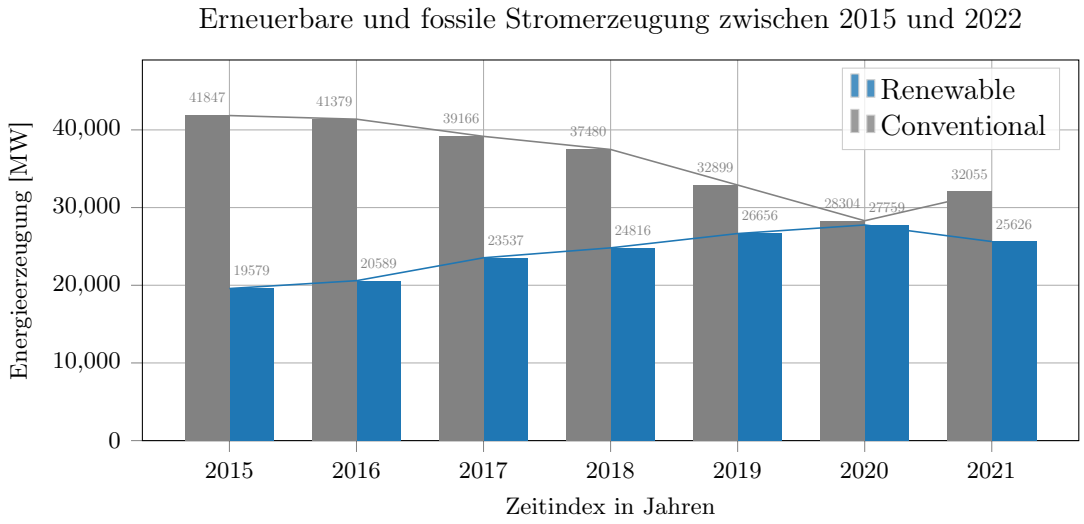


Abbildung 6.2: Mittelwerte der Zu- und Abnahme erneuerbarer und fossiler Energien.

6.3 Vorverarbeitung

Extreme Peaks, wie sie in den Wind OnShore Daten aufgetreten sind, sind durchaus mit Vorsicht zu betrachten. Genau auf solche Ausreißer wurde in Kapitel 2.2.2 hingewiesen. Die Ursache für diese Peaks wurde jedoch mit der außergewöhnlichen Sturmsaison Anfang 2020 erklärt. Weitere Unauffälligkeiten sind in den Daten nicht aufgetreten.

Bereinigung der Daten

In diesem Abschnitt wird auf die wichtigsten Schritte bzgl. des Bereinigens der Daten eingegangen. Bis zu diesem Punkt wurden die Daten bei der Auswahl des Zeitfensters bereits grob bereinigt, um große, fehlerhafte Datenspalten herauszufiltern. Werden mit Matplotlib allerdings Abbildungen geplottet und sind weiterhin noch fehlende Werte enthalten, so werden diese Lücken im Plot automatisch geschlossen bzw. überzeichnet. Um fehlende Werte zu finden, kann wieder auf Pandas zurückgegriffen werden. Nach der Überprüfung wurden in den Daten vier fehlende Werte pro Jahr für jeden Energieträger festgestellt. Pro Spalte betrug diese Anzahl somit 28 und insgesamt 476 Werte. Bei einer Gesamtmenge von 4.173.500 Einträgen aus den Spalten der Energieträger (Anzahl der Zeilen und Spalten miteinander multipliziert) ist die Anzahl der fehlenden Werte jedoch ein sehr geringer Bestandteil. Mithilfe der Methode Rolling Mean, welche in Kapitel

Imputation fehlender Werte am besten abgeschnitten hatte, wurden gleitende Mittelwerte errechnet und in den Lücken eingesetzt.

Die vier fehlenden Werte befinden sich in jedem Jahr jeweils aufeinanderfolgend am 29. März zwischen 2:00 und 2:45 Uhr. Abbildung 6.3 veranschaulicht stellvertretend für die fehlenden Bereiche, wie die Kurve nach Ergänzen der Werte einen natürlichen Verlauf aufweist. Im linken Plot sind vier fehlende Werte der Wind OnShore Spalte zu sehen, gefolgt von desselben Ausschnittes nach der Imputation im rechten Plot.

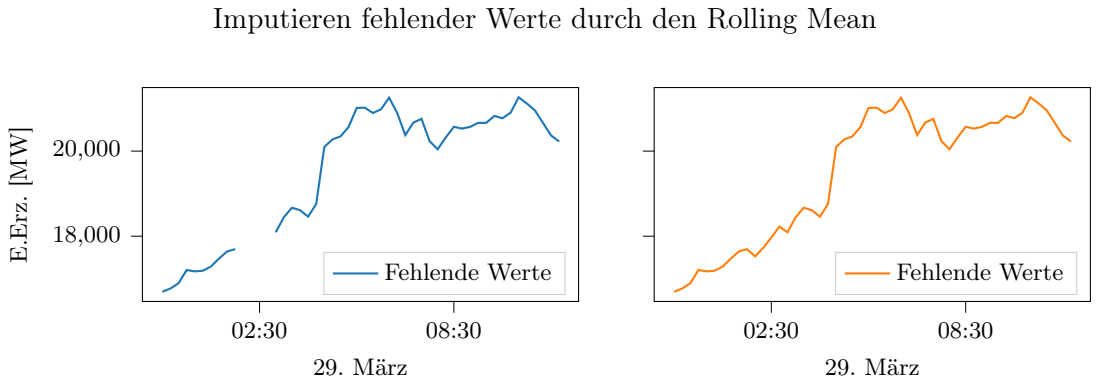


Abbildung 6.3: Beispielhafte Darstellung der Imputation fehlender Werte durch den Rolling Mean (hier für die Wind OnShore Spalte).

Die folgende Auflistung gibt einen Überblick darüber, welche Schritte für das Bereinigen und Sortieren der Daten durchgeführt worden sind:

Vorauswahl: Betrachtung und Wahl des Zeitfensters auf der Plattform ENTSO-E, um große Mängel in den Daten auszuschließen.

Aussortieren: Energieträger bzw. Spalten, die keinen gültigen Eintrag `n/e` aufgewiesen, wurden entfernt.

Ersetzen: Restliche Einträge, die `n/e` enthielten, wurden in eine 0 mit Typ Integer umgeändert.

Zeitstempel: Da jeder Zeitstempel ursprünglich einen Bereich von 15 Min. darstellte, z.B. 01.01.2015 00:00 - 01.01.2015 00:15, wurde der String zerteilt und lediglich der erste Teil davon abgespeichert. In diesem Fall lautet der neue String 01.01.2015 00:00. Diese Datumsangabe wurde anschließend in Pandas Datentyp DateTime umformatiert.

Imputation: Fehlende Werte mit dem Eintrag `NaN` wurden durch den Rolling Mean ersetzt.

Export: Für die weitere Verwendung wurden die bereinigten Dataframes in .csv Dateien abgespeichert.

6.4 Rekursive Vorhersagemethode

Bevor die praktischen Versuche vorgestellt werden, wird in diesem Kapitel näher auf den Versuchsaufbau eingegangen. In den darauf folgenden Kapiteln wird daher auf eine erneute Beschreibung verzichtet.

Für die zukünftigen Prognosen wird ein rekursives Vorgehen verwendet. Kurz und knapp bedeutet das, dass zukünftige Prognosen auf den eigenen Prognosen basieren und sich dieser Prozess je nach Anzahl der Prognoseschritte wiederholt. Wie bereits bei den Versuchen aus Kapitel 5 werden die Modelle mit in Sequenzen verpackten Daten gefüttert, welche zuvor vom TimeseriesGenerator vorbereitet werden. Es gilt wieder, dass die Inputsequenz für ein Modell immer dasselbe Format aufweisen muss, das Format, womit das Modell trainiert wurde. Wenn eine Prognose für einen unbekannten Zeitraum, also nach der letzten Zeile der Testdaten, getätigt wird, müssen die weiteren Inputsequenzen ebenfalls diesem Format entsprechen. Um das zu gewährleisten, werden bzgl. der Prognosen über einen unbekannten Zeitraum die letzten bekannten Daten des Datensatzes in eine Sequenz mit festgelegter Größe gespeichert, welche dann die Startsequenz ist. Anschließend wird für den ersten Datenpunkt nach dem 31. Dezember 2021 eine Prognose abgegeben. Diese wird abgespeichert und der zuletzt bekannten Inputsequenz hinzugefügt, welche wiederum in Form gebracht wird, sodass die Sequenz das richtige Format hat.

In Abbildung 6.4 ist der Prozess rekursiver Vorhersagen skizziert. Dort ist zu dargestellt, wie sich die Sequenz immer weiter mit Werten füllt, bis sie schließlich ganz aus Prognosen besteht. Je nach Größe der Sequenz und Zunahme der darin enthaltenen Prognosen nimmt die Aussagekraft der Prognosen ab, denn in die Zukunft gerichtete Prognosen sind immer mit Fehlern verbunden, welche fortlaufend größer werden. [1] Daraus folgt, je kleiner die gewählte Länge der Sequenz, desto weniger realistische Prognosen können abgegeben werden, bevor der Prognosefehler zu groß wird.

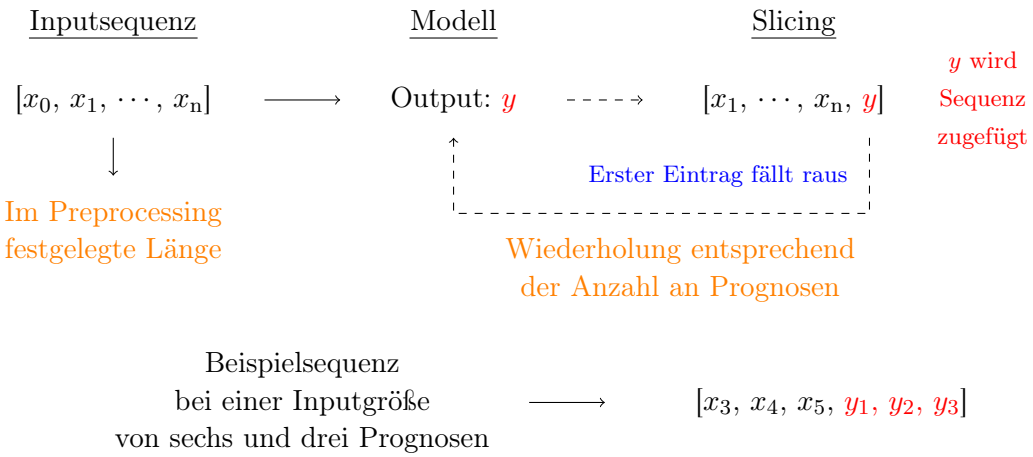


Abbildung 6.4: Prinzip der rekursiven Vorhersage.

Im Laufe der Versuchsdurchführungen stellte sich heraus, dass der Datensatz sehr chaotisch bzw. die zusammengefügten erneuerbaren und fossilen Energien, genauso wie viele andere Energietypen, schwer vorherzusagen und die Ergebnisse oft unbrauchbar waren. Die Wahl der Daten, welche für die weiteren Versuche genutzt wurden, fiel auf die Solar-Spalte, da diese ein periodisch ähnliches, markanteres Muster aufweist. Spannend ist hier zu betrachten, ob die neuronalen Netze das Muster des Tag-Nacht-Wechsels erkennen und wie sie mit unterschiedlichen Peaks durch die Sonne umgehen.

Aufgrund der größeren Menge und um die Trainings zu beschleunigen, werden für beide Versuche die Daten standardisiert.

6.4.1 Datetime Frequenz

Wie eingangs im Kapitel vorgestellt, besteht der Datensatz aus einer Frequenz von 15 Minuten, was 96 Werte pro Tag entspricht. Bei einer Vorhersagedauer von z.B. drei Monaten, bzw. 90 Tagen, wären das 8.640 zu prognostizierten Werte. Damit sich mit Prognosen über einen etwas längeren Zeitraum befasst werden kann und gleichzeitig weniger Werte vorhergesagt werden müssen, wird der Datensatz in eine stündliche und tägliche Frequenz unterteilt. Ziel ist es, dadurch den Prognosefehler zu verringern und somit bessere Ergebnisse zu erzielen. Hierbei ist der Nachteil, dass die große Menge an Daten nicht so wie erwartet genutzt werden kann. Mithilfe von Pandas können zeitbasierte Daten in

unterschiedliche Frequenzen unterteilt werden. Je nach Frequenz werden die Mittelwerte der Zeilen berechnet und zusammen in einem neuen Dataframe abgespeichert.

Stündliche Frequenz

Die Anzahl der Daten bei einer stündlichen Frequenz beträgt nun 61.368 statt den ursprünglichen 245.500 Zeilen. Für diesen Versuch wurde ein Fenster von 168 Stunden als Länge der Inputsequenz gewählt. Das entspricht einer Dauer von umgerechnet sieben Tagen. Abbildung 6.5 zeigt eine Beispielsequenz. Ziel ist es, basierend auf diesem Inputfenster, zunächst 24 Werte, also einen Tag, aus den Testdaten vorherzusagen und entsprechend abzugleichen.

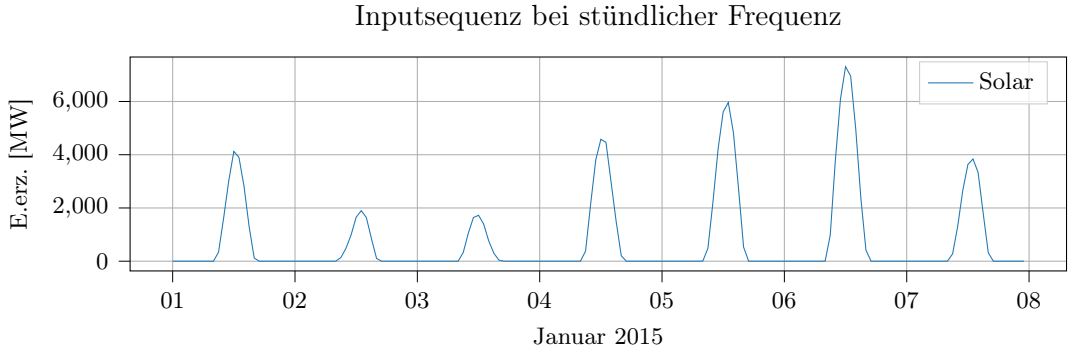


Abbildung 6.5: Darstellung eines Sieben-Tage-Zeitraums bei einer Sequenzlänge von 168.

Tägliche Frequenz

Bei den täglich frequentierten Daten ist es fraglich, ob noch realistische Ergebnisse erzielt werden können, da die Anzahl der Daten weiter um den Faktor 24 verringert wird. Somit verbleiben für Training und Testen lediglich 2.557 Datenpunkte. Damit in der Inputsequenz eine saisonale Komponente enthalten ist, wurde für dessen Größe eine Länge von 365 Werten gewählt. Abbildung 6.6 zeigt eine solche Sequenz.

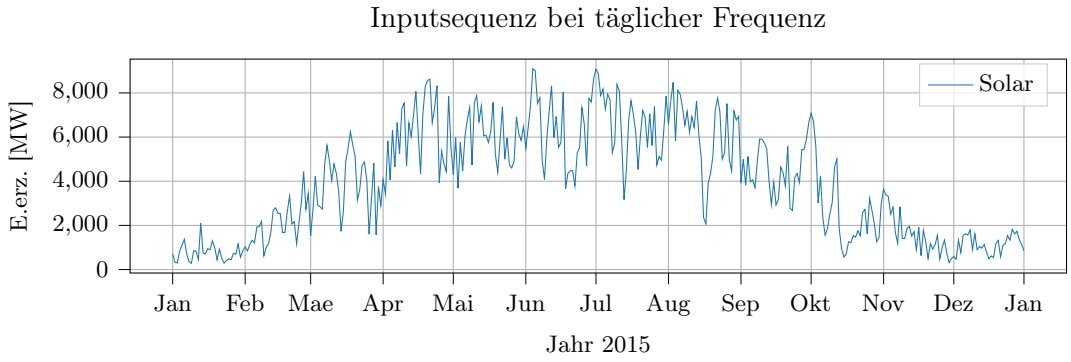


Abbildung 6.6: Darstellung eines 365-Tage-Zeitraums bei einer Sequenzlänge von 365.

6.4.2 Baseline-Modelle

Um zu überprüfen, ob die erstellten Modelle für diese Aufgabenstellung geeignet sind, werden die Ergebnisse mit zwei Baseline-Schätzmodellen verglichen. [14, 32] Diese Modelle orientieren sich anhand naiver Faustformeln und bestehen aus verschobenen Kurven.

Stündliche Frequenz

Bei dem Versuch mit stündlichem Index werden die Daten exakt fünf und drei Tage zuvor für die Baseline-Modelle verwendet. Die Bezeichnung für das Fünf-Tage-Modell lautet B1 und für das Drei-Tage-Modell B2. Je näher diese verschobenen Werte an dem aktuellen Wert liegen, desto niedriger müsste der Fehler sein und desto besser muss die Prognose der neuronalen Netze ausfallen, um diesen Fehlerwert unterbieten zu können [14].

Im Laufe der Untersuchungen wurde festgestellt, dass die Schwankungen der Peaks willkürlich sind. Dies deckt sich mit dem MAE, welcher beim Fünf-Tage Modell teilweise niedriger war, als der des eigentlich näheren Drei-Tage Modells.

Im Plot 6.7 ist ein Ausschnitt der Schätzungen der Baseline-Modelle zu sehen. Die farbigen Kurven stellen deren täglichen Mittelwert dar und werden später mit den tatsächlichen Prognosen verglichen.

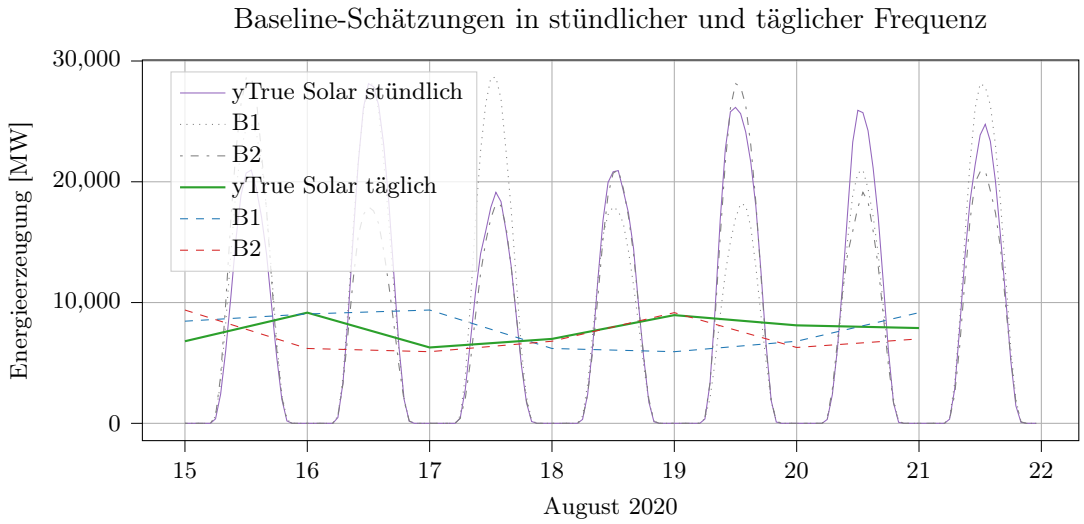


Abbildung 6.7: Darstellung der Baseline-Modelle B1 und B2 verglichen mit einem Auszug aus den Testdaten. Im Plot sind die Ergebnisse in stündlicher und täglicher Frequenz aufgeführt.

Tägliche Frequenz

Wie zuvor werden auch für die tägliche Prognose zwei Baseline-Modelle erstellt. Der Versatz zu dem originalen Wert beträgt sieben und zwei Tage. Die Bezeichnung für das Sieben-Tage-Modell lautet B1 und für das Zwei-Tage-Modell B2. Der Umstand, dass an bestimmten Tagen die Schätzung des weiter in der Vergangenheit liegenden Modells besser ist, als das Modell mit dem niedrigeren Versatz, lässt sich auch hier bestätigen.

In Abbildung 6.8 sind die Schätzwerte der Baseline-Modelle dargestellt. Die farbigen Kurven zeigen den wöchentlichen Mittelwert und werden später ebenfalls mit den Prognosen verglichen.

6.4.3 Univariate und multivariate Analyse

Beide Versuche werden mit der univariaten Methode untersucht, bei welcher nur ein Feature als Input dient. Würde zu Beginn des Trainings mehr als ein Feature festgelegt werden, kann die Prognose nicht für unbekannte Zeitpunkte durchgeführt werden. Da bei der rekursiven Vorhersage die Prognose auf sich selbst basiert, würden die restlichen Features bei der Prognose von unbekannten Daten fehlen. Zudem muss, wie erwähnt, das Format der Inputsequenz immer gleich sein.

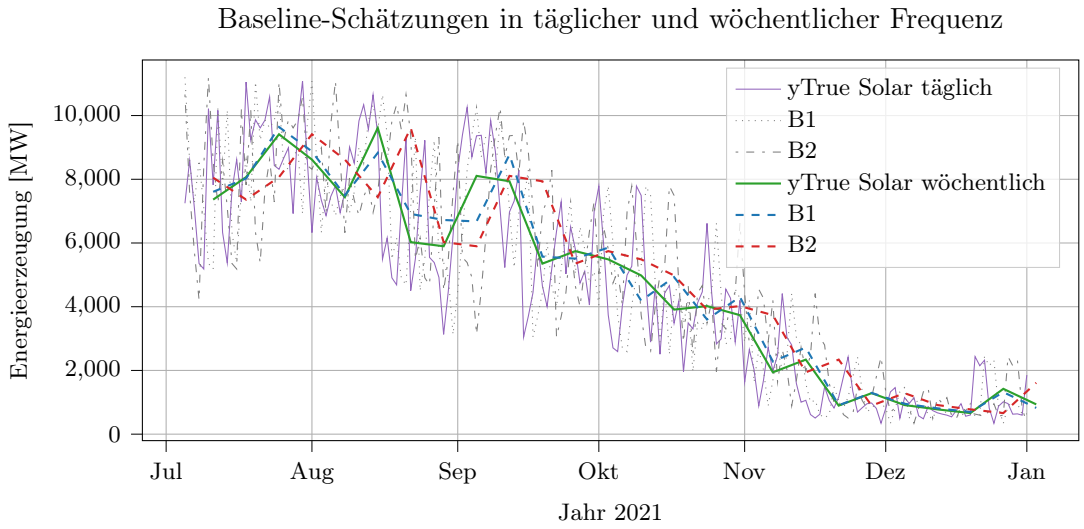


Abbildung 6.8: Darstellung der Baseline-Modelle B1 und B2 verglichen mit einem Auszug aus den Testdaten. Im Plot sind die Ergebnisse in täglicher und wöchentlicher Frequenz aufgeführt.

Eine Möglichkeit, daraus eine multivariate Analyse zu gestalten, ist das Trainieren von mehreren Modellen. Werden z.B. zwei Features verwendet, A wird vorhergesagt und B ist das zweite Feature, so könnte Spalte B im Einzelnen mit der univariaten Methode trainiert und z.B. 30 Tage vorhergesagt werden. Diese Werte werden in den multivariaten Datensatz eingefügt, wodurch die Prognose für maximal 30 Schritte getätigt werden kann. Anschließend würde das Format mit zwei Features als Input nicht mehr stimmen. Aufgrund der chaotischen Daten kommt keine weitere Spalte als die Solarenergie infrage (geprüft wurden Solar, Wind, Nuklear, Braunkohle). Das Testen aller möglichen Kombinationen aus den 19 Spalten war in dieser Arbeit nicht möglich.

Eine weitere Methode die Struktur eines multivariaten Ansatzes aufzubauen, ist auf den Zeitindex zurückzugreifen. Mithilfe von Pandas lassen sich aus dem Zeitstempel weitere Zeitangaben extrahieren und in eigenen Spalten abspeichern, dargestellt in Tabelle 6.3 mit zusätzlichen Spalten für Jahr, Monat, Woche und Tag.

Mithilfe des Zeitstempels wird also ein univariater in einen multivariaten Datensatz transformiert. Im Weiteren werden die zusätzlichen Spalten genauso standardisiert, bzw. verarbeitet, wie mit den Solardaten vorgegangen wurde. Der Vorteil dieser Vorgehensweise ist, dass zum einen der Input für die neuronalen Netze aus mehr Daten besteht. Zum anderen, dass die zusätzlichen Spalten für die Prognose über einen unbekannten Zeitraum,

Tabelle 6.3: Auszug aus dem multivariaten Datensatz für die tägliche Analyse. Darstellung der hinzugefügten Spalten.

	Zeitstempel	Solar	Jahr	Monat	Woche	Tag
0	2015-01-01	718.416667	2015	1	1	1
1	2015-01-02	322.958333	2015	1	1	2
2	2015-01-03	302.291667	2015	1	1	3
3	2015-01-04	836.041667	2015	1	1	4
4	2015-01-05	1109.541667	2015	1	2	5

einfach generiert werden können. Auch hierfür wird Pandas wieder eingesetzt. Über Funktionsparameter kann angegeben werden, in welcher Frequenz und wie viele Zeitschritte erzeugt werden. Tabelle 6.4 zeigt die ersten fünf Zeilen der generierten Datumsangaben. Die „Forecasting“ Spalte ist als Platzhalter für die Prognosen vorgesehen. In jedem Prognoseschritt wird aus der generierten Datumsangabe der entsprechende Wert herausgelesen, skaliert und mit in die Sequenz für den nächsten Prognoseschritt eingefügt. Diese Methode kommt bei der täglichen Vorhersage zum Einsatz.

Tabelle 6.4: Auszug der generierten Datumsangaben, aus welchen die zusätzlichen Features für die Vorhersage von unbekannten Daten extrahiert wurden.

	Zeitstempel	Vorhersage	Jahr	Monat	Woche	Tag
0	2022-01-01	0	2022	1	52	1
1	2022-01-02	1	2022	1	52	2
2	2022-01-03	2	2022	1	1	3
3	2022-01-04	3	2022	1	1	4

6.4.4 Trainingsparameter

In dieser Sektion werden die Randparameter für die Trainings vorgestellt. Die Datensätze werden jeweils zu 80 % in Trainings- und 20 % in Testdaten segmentiert.

Im TimeseriesGenerator werden die unterteilten Daten beim Versuch mit stündlicher Frequenz mit einer Batch Size von 64 und beim täglichen Versuch mit einer Größe von 32 vorbereitet. 32 ist ein typischer Wert, welcher für diese Einstellung genutzt wird. [14]

Der Datensatz mit stündlicher Frequenz ist allerdings 24-mal größer, weshalb zur Beschleunigung des Trainings 64 als Größe festgelegt wurde. Beides sind Werte, die im Mittelfeld der üblichen, genutzten Größen liegen. [14] Dadurch sind Ausreißer bei der Gewichtsanpassung zwischen den Batches nicht zu groß und auf der anderen Seite wird so die Chance gesenkt, in einem lokalen Minimum zu enden.

Während des Trainings wird auf den MSE als Verlustfunktion zur Messung der Modellleistung zurückgegriffen. Zur Validierung mit den Testdaten nach Abschluss des Trainings wird der MAE verwendet. Beide Funktionen und die Begründung dieser Auswahl wurde in Kapitel 2.3.1 genauer erklärt. Des Weiteren wurde der Adam-Algorithmus zum Optimieren der Gewichte verwendet.

Um das Herausfinden einer geeigneten Epochenanzahl zu vereinfachen, werden bei jedem Training sogenannte Callback-Funktionen von Keras verwendet. Hierdurch werden immer die besten Gewichte eines neuronalen Netzes abgespeichert. Dies geschieht auf zwei Wegen, dem EarlyStopping und dem ModelCheckpoint. Zuerst wird angegeben, welcher Parameter während des Trainings beobachtet werden soll, was in diesem Fall der MSE ist. Des Weiteren ist ein Wert anzugeben, der bestimmt, wie viele weitere Epochen vergehen dürfen (z.B. fünf, zehn, 15 etc.), nachdem der minimale MSE erreicht wurde.

Der Trainingsprozess wird dann abgebrochen, wenn sich das Modell im Laufe der angegebenen Epochen nicht verbessert. Anschließend wird das Modell mit den besten Gewichten gespeichert. Ein weiterer Vorteil ist, dass eine beliebig große Anzahl an Epochen von z.B. 1000 eingestellt werden kann und das Training nach z.B. 25 Epochen automatisch abgebrochen wird, falls es sich nicht verbessert hat. Es besteht allerdings die Möglichkeit, dass dieser Abbruch innerhalb eines lokalen Minimums geschieht, obwohl sich das Modell eigentlich noch verbessern könnte.

6.5 Stündliche Vorhersage

In diesem Kapitel wird die Vorhersage von Solarwerten untersucht. Der Versuch wird mit einem univariaten Datensatz durchgeführt und die Prognosen werden mit der rekursiven Strategie erzielt. Wie in den beiden kleineren Versuchen aus Kapitel 5 werden die Prognosen verschiedener Modelle miteinander verglichen.

Architekturen

Zum Einsatz kommen ein FFNN, ein LSTM-Modell, eine Autoencoder-Struktur und ein CNN. Auffällig war, dass die LSTM- und Autoencoder-Modell erst nach dem Hinzufügen von konvolutionalen Schichten zur Netzstruktur passable Ergebnisse geliefert haben.

6.5.1 Testprognosen

In Kapitel 6.4.1 wurde beschrieben, warum für die stündliche Prognose die Sequenzlänge von 168 Werten gewählt wurde. Nach dem Input von umgerechnet sieben Tagen werden jeweils 24 Werte, was einem Tag entspricht, vorhergesagt. In Abbildung 6.9 sind die unterschiedlichen Vorhersagen der Modelle für einen Tag dargestellt. Zu erkennen ist, dass die Modelle in den Abend- bis Morgenstunden ziemlich genau die richtigen Testwerte treffen. Die Modelle konnten sich sozusagen einprägen, dass in regelmäßigem Abstand Werte um 0 herum auftreten. Außerdem ist zu erkennen, dass die Kurven gegen Mittag den höchsten Punkt aufweisen, was mit dem höchsten Punkt der Sonne und einem besseren Winkel für die beschienenen Solarpanels zusammenpasst.

FFNN, LSTM, Autoencoder und CNN bei Testprognosen über einen Tag

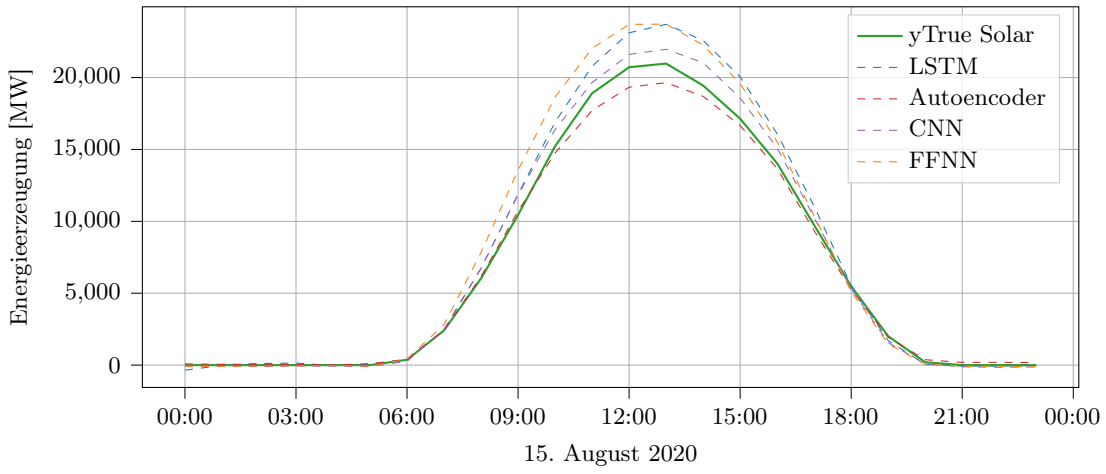


Abbildung 6.9: Vergleich der Prognosen mit den echten Testwerten eines Tages.

Als Beispiel für die Testprognosen werden in diesem Abschnitt sieben Tage aus den Testdaten betrachtet. In Abbildung 6.10 sind diese sieben aufeinanderfolgenden Tage einzeln vorhergesagt und aneinander gereiht worden. Wie im vorherigen Plot sind in den Abend- und Morgenstunden gute Prognosen zu erkennen. Auch der jeweilige Peak ist

mittags gegen 13 Uhr auszumachen. Die Abbildung bestätigt aber auch die Vermutung, dass es für die Modelle schwierig ist, die richtigen Peaks im Vergleich zu den Testdaten zu treffen. Vor allem bei sehr hohen Peaks wie am 16. und dem darauffolgenden sonnenarmen 17. August 2020. Auffällig ist auch, dass das LSTM-Modell im Vergleich mit den anderen Kurven oft die höchsten Werte und der Autoencoder, welcher auch LSTM-Strukturen enthält, oft die niedrigsten Werte prognostiziert hat.

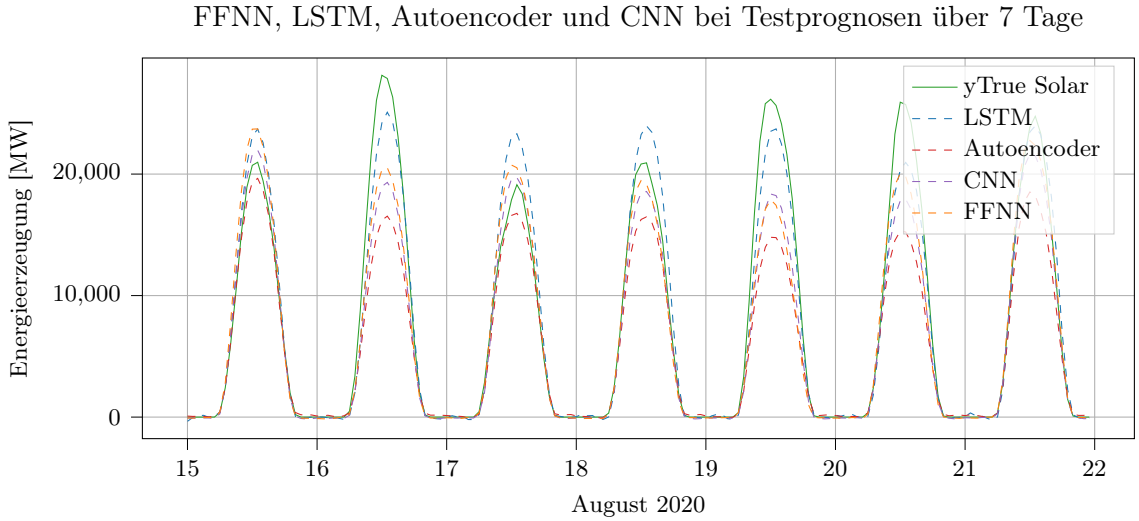


Abbildung 6.10: Gegenüberstellung der Darstellung von aufeinanderfolgenden Tagesprognosen mit den echten Testwerten.

6.5.2 Evaluation

Mit Blick auf den MAE, welcher zur Berechnung des Fehlerwertes zwischen Prognosen und Testdaten herangezogen wurde, lässt sich sagen, dass die naiven Baseline-Schätzungen recht nah an den Modell-Prognosen liegen. In Tabelle 6.5 sind die MAE-Werte für diesen Testausschnitt enthalten. An dieser Stelle die Anmerkung, dass die Werte vom FFNN und CNN abgesehen von den Nachkommastellen identisch sind und dies kein Tippfehler ist. Für diese Versuche wurden mehrere Modelle trainiert, um zu sehen, ob es große Schwankungen bei der Genauigkeit der Prognosen gibt.

Aus der Tabelle geht hervor, dass das LSTM-Modell über diesen Zeitraum die besten Prognosen abgegeben hat. Der Autoencoder, welcher bei den Versuchen aus Kapitel 5 noch mit die niedrigsten MAE-Werte aufwies, schnitt in diesen Versuchen am schlechtesten

ab. Das FFNN und CNN sind etwa in der Mitte einzuordnen, verglichen mit den anderen Prognosen. Wie bereits angedeutet, befinden sich die fehlerhaften Abweichungen der Baseline-Schätzungen genau im selben Fehlerbereich. Lediglich das LSTM-Modell konnte bessere Prognosen abgeben als das theoretisch bessere B2-Modell. Der Autoencoder ist als einzige Architektur sogar schlechter als das B1-Modell.

Tabelle 6.5: Errechnete MAE-Werte zwischen den Prognosen und den Baseline-Schätzungen, verglichen mit den echten Testwerten.

	FFNN	LSTM	Autoencoder	CNN	B1	B2
MAE	1458	1119	2030	1458	1683	1376

In diesem Versuch konnte festgestellt werden, dass das LSTM-Modell, aufgrund der niedrigsten Abweichung (MAE) zwischen den Testprognosen und den Testdaten, für die Prognose von Solardaten (bezogen auf deinen Tag) am besten geeignet war.

6.5.3 Zukünftige Prognose

Bei der Vorhersage von Daten außerhalb des Datensatzes wurden die letzten sieben Tage der Testdaten als Inputsequenz gewählt. Abbildung 6.11 zeigt das Resultat der Vorhersage für die ersten 21 Tage im Januar 2022. Der MAE kann nicht zur Validierung herangezogen werden, da der Zeitraum theoretisch unbekannt ist und die Vorhersage mit keinen Werten, so wie bei den Testprognosen, verglichen werden kann.

Die Eigenschaft der Kurven aus den Testprognosen, dass das LSTM-Modell die höchsten und der Autoencoder die niedrigsten Werte vorhersagt, deckt sich mit der Vorhersage von unbekannten Daten. Mit Blick auf die Kurven in der Abbildung 6.11 verglichen mit den letzten fünf Tagen des Jahres 2021, fällt jedoch auf, dass während die Prognosen vom FFNN und LSTM-Modell hoch bzw. sehr hoch ausfallen, der Autoencoder und CNN die realistischeren Ergebnisse erzielten. Anzumerken ist, dass aufgrund der rekursiven Vorhersage die anfängliche Inputsequenz bei einer Dauer von 21 vorhergesagten Tagen ganze dreimal ersetzt wurde. Von den noch echten Werten aus den letzten fünf Tagen sind dementsprechend keine mehr enthalten. Nach erster Beurteilung deckt sich diese Tatsache mit dem wachsenden Prognosefehler bei allen Architekturen außer beim LSTM-Modell. Hier ist der Fehler stark gestiegen. Dieses Ergebnis deckt sich nicht mit den Erkenntnissen aus den Testprognosen. Auch wenn die Prognosen der Tageshöchststände im Allgemeinen

mit Skepsis zu betrachten sind, sind die Abend- bis Morgenstunden größtenteils korrekt prognostiziert worden.

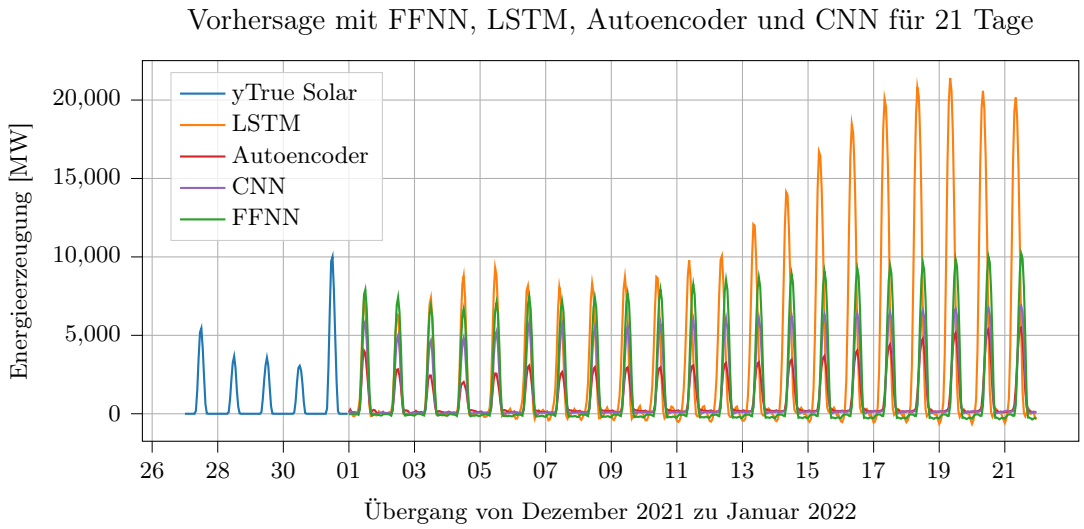


Abbildung 6.11: Darstellung der Vorhersage über einen unbekannten Zeitraum von 21 Tagen.

Aufgrund der widersprüchlichen Ergebnisse zwischen der Prognose von einem Tag im Testzeitraums und der Prognose von Daten außerhalb des Testzeitraums über mehrere Tage (im Versuch wurden 21 Tage verwendet) konnten keine zufriedenstellenden Ergebnisse erzielt werden. Daher wurde dieser Versuch nicht mit der multivariaten Methode durchgeführt.

6.6 Tägliche Vorhersage

Bessere Prognosen sind dafür bei der täglichen Vorhersage zu erwarten: Als gewählte Länge der Inputsequenz wurden hier 365 (Tage) gewählt. Im Vergleich zum stündlichen Versuch (hier waren es 168 Werte), ist die Länge also etwa doppelt so lang, aufgrund der täglichen Frequentierung wird hier jedoch ein wesentlich größerer Zeitraum abgedeckt. Da bei der rekursiven Vorhersage von z.B. 180 Tagen in der Inputsequenz noch knapp die Hälfte an echten Werten aus dem Datensatz enthalten ist, ist ein geringerer Prognosefehler und das bei einem weitreichenderen Prognosezeitraum zu erwarten.

Wie in Kapitel 6.4.3 beschrieben, wird dieser Versuch mit der univariaten und multivariaten Methode durchgeführt. Auftretende Ergebnisse wie MAE-Werte und Abbildungen werden parallel vorgestellt.

Architekturen

Bei diesem Versuch werden die üblichen Architekturen untersucht. Verglichen werden die Prognosen von einem FFNN, einem LSTM-Modell, einer Autoencoder-Struktur und einem CNN. Bei der multivariaten Methode wird das FFNN nicht berücksichtigt, da dieses Modell mit der komplexeren Datenstruktur keine guten Ergebnisse geliefert hat. Beim LSTM- und dem Autoencoder-Modell wurden, wie im vorherigen Versuch auch, dem Netzaufbau konvolutionale Schichten hinzugefügt. Erst dann wurden vernünftige Ergebnisse erzielt.

6.6.1 Testprognosen

Die in diesem Abschnitt dargestellten Testprognosen beziehen sich auf einen Testzeitraum der letzten 180 Tage vom Jahr 2021. In Abbildung 6.12 sind die Testprognosen mit Nutzung der univariaten Methode festgehalten.

Zwei Dinge fallen hierbei auf. Zum einen, dass die echten Solarwerte wesentlich mehr Schwankungen aufweisen als es bei den Modellprognosen der Fall ist. Zum anderen, dass die Kurven eine ähnlich starke Abnahme in Richtung Winter enthalten. Abgesehen vom FFNN, dessen Prognosekurve im Vergleich die größten Schwankungen aufweist, ähneln sich die Kurven der anderen Modelle in ihrem Verlauf. Noch besser zu erkennen ist diese Beobachtung, wenn die wöchentlichen Mittelwerte der Prognosen betrachtet werden (s. Abbildung 6.13). Die Kurven der Modellprognosen ähneln nun mehr der Kurve der echten Solardaten.

Werden diese Kurven mit dem Ergebnis der multivariaten Analyse, dargestellt in Abbildung 6.14, verglichen, so bilden auch diese Kurven einen ähnlich aussehenden Verlauf ab. Zusammenfassend lässt sich anhand dieser Visualisierungen nicht genau sagen, welche Modelle bei welcher Methode die besten Vorhersagen getroffen haben. Deswegen werden die Ergebnisse im nächsten Abschnitt mit dem MAE untersucht.

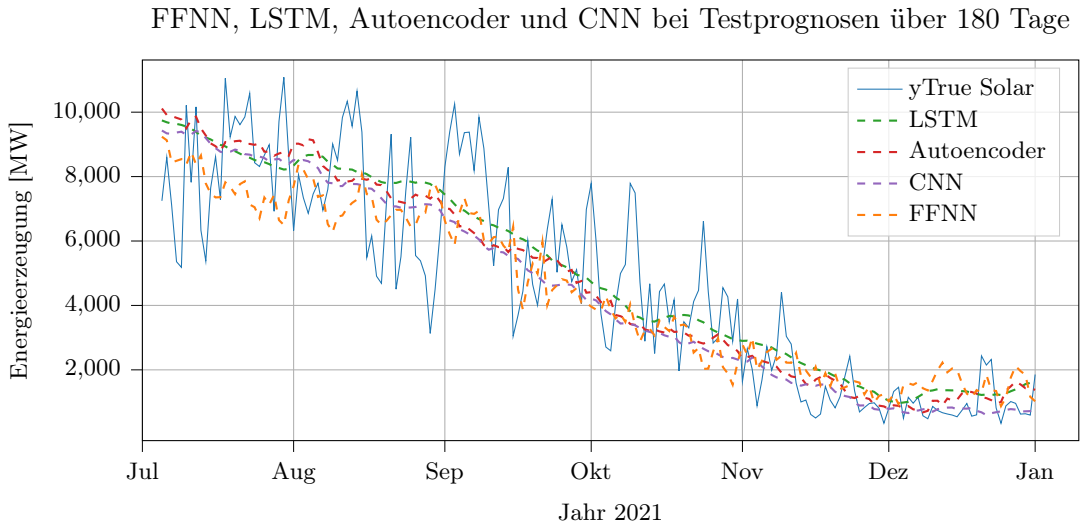


Abbildung 6.12: Univariate Methode: Darstellung des Vergleichs von Testprognosen mit den echten Testwerten in täglicher Frequenz.

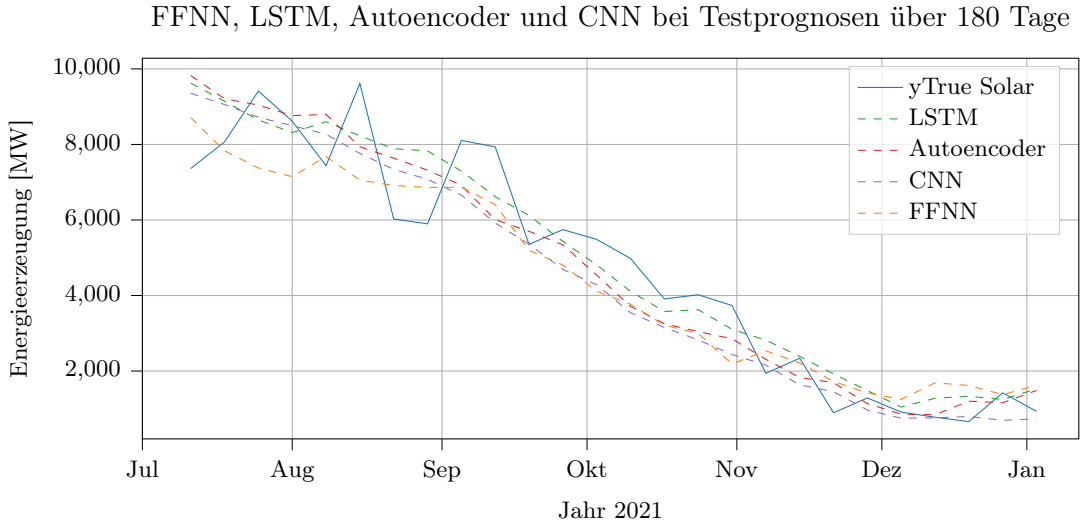


Abbildung 6.13: Univariate Methode: Darstellung des Vergleichs von Testprognosen mit den echten Testwerten in wöchentlicher Frequenz.

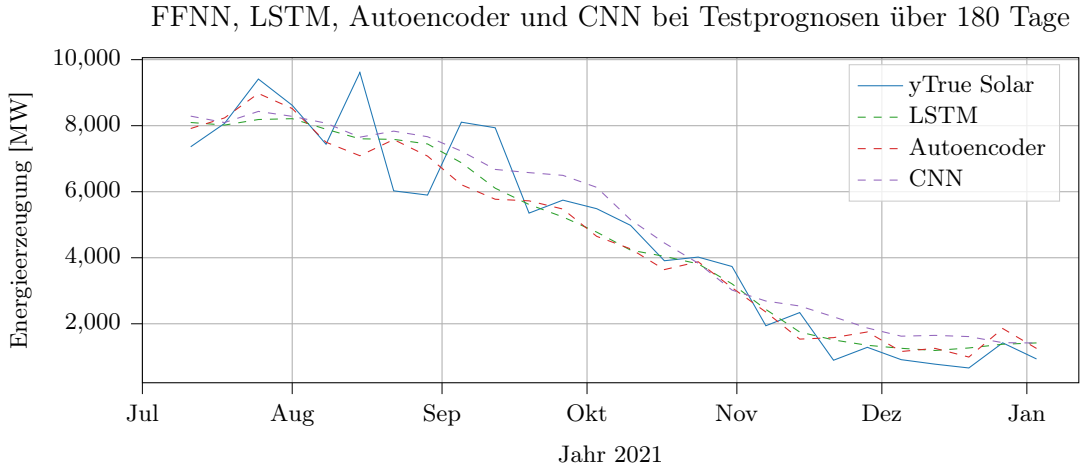


Abbildung 6.14: Multivariate Methode: Darstellung des Vergleichs von Testprognosen mit den echten Testwerten in wöchentlicher Frequenz.

6.6.2 Evaluation

Bei der Betrachtung der MAE-Werte in Tabelle 6.6 fällt auf, dass sowohl bei der univariaten als auch bei der multivariaten Analyse die Modelle bessere Prognosen abgegeben haben als beide Baseline-Modelle. Dieses Ziel wurde also erreicht.

Tabelle 6.6: Errechneter MAE zwischen den Testprognosen und den echten Testwerten.

		FFNN	LSTM	Autoencoder	CNN	B1	B2
Univariate M.	MAE	1391	1244	1263	1233	1697	1395
Multivariate M.	MAE	-	1170	1221	1262	1697	1395

In diesem Versuch konnte bei Verwendung der univariaten Methode die CNN-Struktur die besten Prognosen verglichen mit den Testdaten erzielen. Bei der multivariaten Methode hat das LSTM-Modell den niedrigsten Fehlerwert (MAE). Insgesamt sind die Abweichungen bei den multivariaten Modellen größtenteils niedriger ausgefallen.

6.6.3 Zukünftige Prognose

Die Prognose von unbekannten Daten (Daten außerhalb des Datensatzes) wird wieder durch rekursive Vorhersagen durchgeführt. In Kapitel 6.4.3 wurde erklärt, wie das Auf-

bereiten der Daten für die univariate und multivariate Analyse bei unbekannten Daten durchgeführt wird.

Abbildung 6.15 zeigt bei Nutzung der univariaten Methode die Vorhersageergebnisse der Solardaten über den Zeitraum von einem Jahr. Bis auf den Dezember, Ende 2022, haben die Kurven des LSTM- und des Autoencoder-Modells einen sehr ähnlichen Verlauf. Mit Betrachtung der Solarwerte aus dem Jahr 2021 weisen wieder alle Prognosekurven für das Jahr 2022 einen ruhigeren Verlauf auf. Die Kurve des FFNN ist die Einzige, welche eine ähnliche Eigenschaft aufweist. Aufgrund der unterschiedlichen Höchstwerte der Kurven kann nicht genau gesagt werden, welches Modell die realistischeren Ergebnisse geliefert hat.

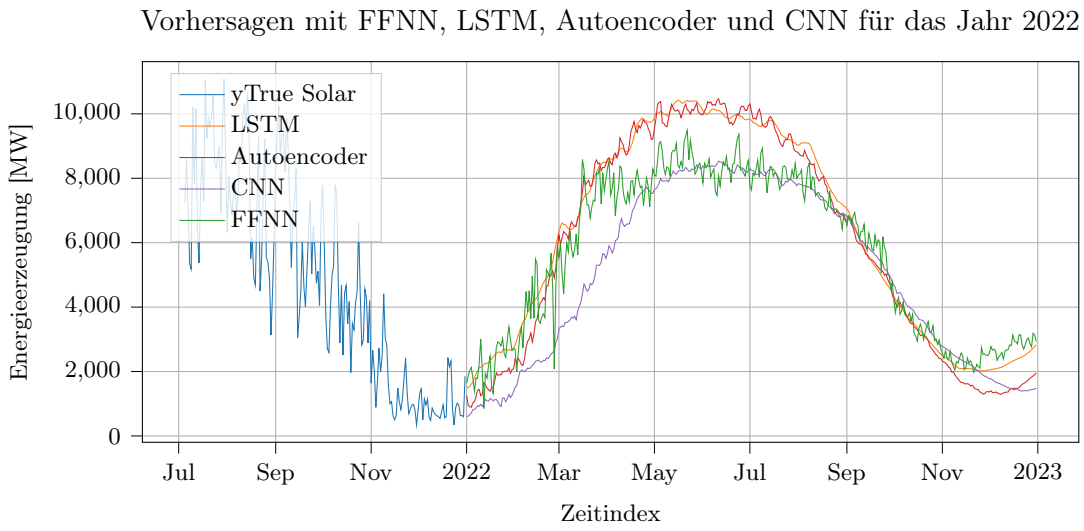


Abbildung 6.15: Univariate Methode: Vergleich der Modelle bei der Vorhersage von unbekannten Daten für das Jahr 2022.

Anders sehen die Kurven von Abbildung 6.16 aus, welche die Prognosen für das Jahr 2022 bei Verwendung der multivariaten Methode zeigt. Die Kurven liegen insgesamt näher beieinander und vor allem beim Höchststand im Juli haben die Modelle ähnlicher Prognosen abgegeben.

Sollen allerdings drei Jahre vorhersagt werden, so weisen die Kurven deutliche Unterschiede auf. Während bei der univariaten Analyse, dargestellt in Abbildung 6.17, für das erste Jahr jeweils zwei Modellkurven ähnliche Peaks in den Sommermonaten aufweisen, zerstreuen sich die Kurven in dem zweiten vorhergesagten Jahr zunehmend. Ab dem

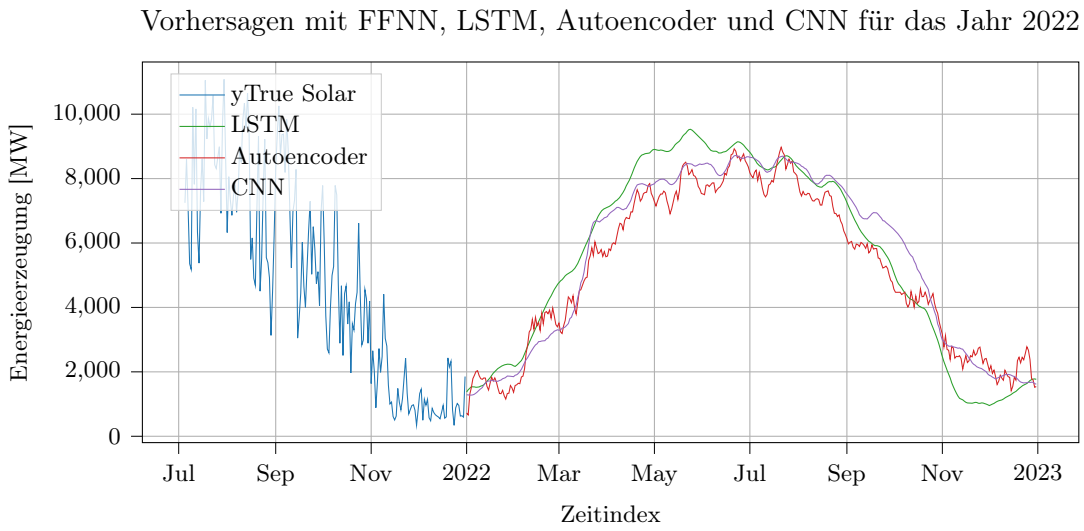


Abbildung 6.16: Multivariate Methode: Vergleich der Modelle bei der Vorhersage von unbekannten Daten für das Jahr 2022.

Übergang zum Jahr 2024 werden die Prognosen vollständig unrealistisch, sodass nicht mal die Winter-Sommer-Zeit richtig dargestellt wird.

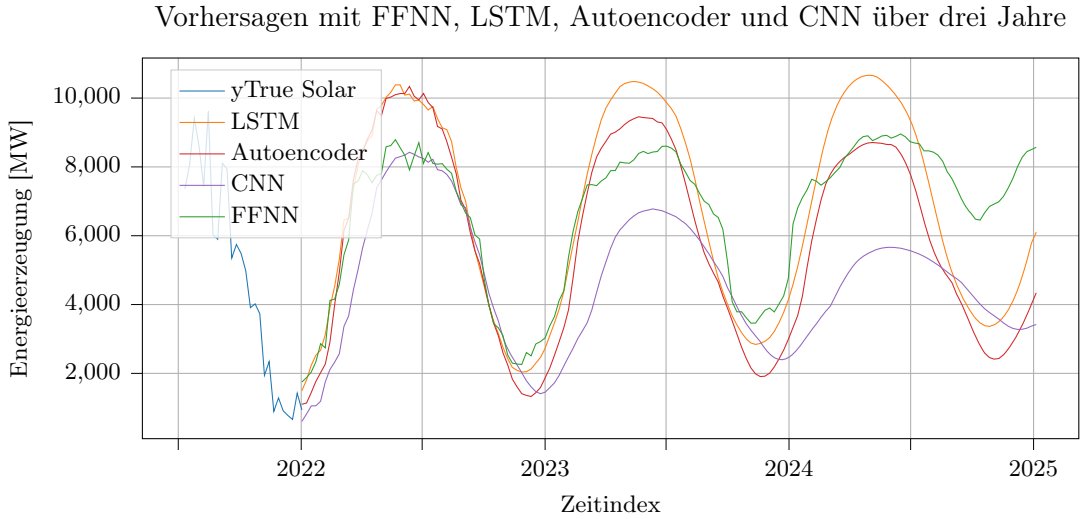


Abbildung 6.17: Univariate Methode: Vorhersage von unbekannten Daten über einen Zeitraum von drei Jahren.

Bessere Vorhersagen lieferte hier die multivariate Methode, welche in Abbildung 6.18 dargestellt ist. Auch wenn in diesem Plot keine Kurve vom FFNN enthalten ist und dies die Visualisierung etwas ruhiger darstellen lässt, gibt die Grafik den Eindruck, dass die Kurven besser verlaufen als bei der univariaten Methode. Verglichen mit dem vorherigen Plot sind vor allem die Prognosewerte, welche um die Jahreswechsel herum liegen, wesentlich realistischer und das bis zum Jahr 2025. Die größte Auffälligkeit bildet hier die Kurve des LSTM-Modells, welche ab 2023 im Vergleich zu den anderen beiden Kurven stark steigt und an den Jahresübergängen erheblich fällt. Auch wurde die Inputsequenz bei der Vorhersage von drei Jahren mittlerweile dreimal gänzlich durch eigene Prognosewerte ersetzt, was den natürlich auftretenden Prognosefehler erhöht.

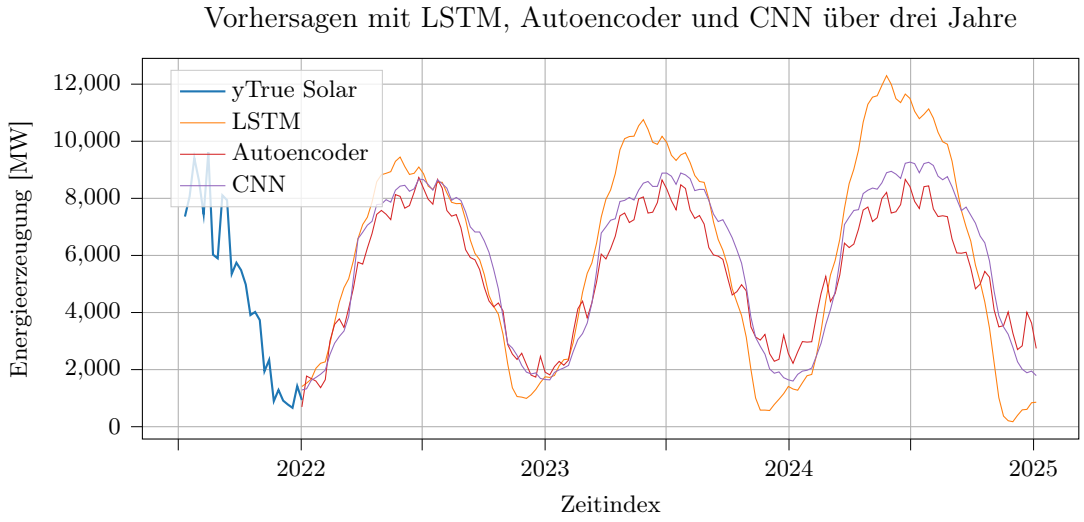


Abbildung 6.18: Multivariate Methode: Vorhersage von unbekannten Daten über einen Zeitraum von drei Jahren.

Zusammenfassend sind bei diesem Versuch zwei Ergebnisse zu unterscheiden. Anhand der Testprognosen konnte bei der univariaten Vorhersage der niedrigste MAE durch ein CNN-Modell erreicht werden, während bei der multivariaten Vorhersage ein LSTM-Modell die niedrigste Abweichung (MAE) erzielte.

Mit Betrachtung der Vorhersage von unbekannten Daten stehen diese Prognosen zum Teil widersprüchlich der Ergebnisse aus den Testprognosen gegenüber. Obwohl bei der univariaten Methode das LSTM nicht den niedrigsten MAE-Wert aufgewiesen hat, sind Vorhersagen für ein bis drei Jahre wesentlich realistischer als die vom FFNN, Autoencoder und CNN. Ähnlich verhält es sich bei der multivariaten Analyse. Hierbei hat das LSTM

den besten MAE-Wert bei den Testprognosen erreicht, doch dessen Prognosekurve über drei unbekannte Jahre wirkt unrealistischer als vom CNN und Autoencoder.

6.7 Vergleich der stündlichen und täglichen Vorhersage

In der Auswertung der Ergebnisse aus den Versuchen in stündlicher und täglicher Frequenz wurde deutlich, dass die Menge an Daten aus dem Datensatz zum größten Teil überhaupt nicht genutzt werden konnte. In Unterkapitel 6.3 wurde der Datensatz beschrieben und die Anzahl der enthaltenen Einträge genannt. Insgesamt betrug diese Zahl 4.173.500 Datenpunkte. Aufgrund der insgesamt chaotischen Daten konnten letztlich jedoch nur die Solardaten verwendet werden.

Eine weitere Reduzierung der Daten kam durch die Frequentierung in stündliche und tägliche Daten zustande. Da die Daten ursprünglich in einer 15 Min. Frequenz vorlagen, wurden die Daten für die stündliche Vorhersage um ein Viertel auf 61.368 Solareinträge verringert. Bei der täglichen Vorhersage reduzierte sich die Menge an Daten sogar weiter um den Faktor 24, sodass bei der univariaten Methode nur noch 2.557 Einträge vorlagen. Durch das Hinzufügen der Spalten von Jahr, Monat, Woche und Tag, welche aus der Datumsangabe extrahiert werden konnten, wurde die Größe des Datensatzes beim multivariaten Versuch immerhin auf 15.342 Einträge erhöht.

6.7.1 Vergleich der Testprognosen

Bei den stündlichen und täglichen Versuchen hatten die genutzten Architekturen FFNN, LSTM, Autoencoder und CNN Schwierigkeiten damit, die Peaks der Solarwerte im Vergleich zu den Testdaten vorherzusagen. Dieses Merkmal ist jedoch auch in anderen Arbeiten aufgetreten ist, wie in [18]. Positiv geht dafür hervor, dass die Modelle den Tag-Nacht-Rhythmus und die damit verbundenen Unterschiede bei der Stromerzeugung durch die Sonne erkennen konnten. Technisch ausgedrückt, konnten die Modelle in den Sequenzen ein wiederkehrendes Muster erkennen. In Abbildung 6.10 ist z.B. gut zu sehen, wie die Modellprognosen der Solarwerte in den nächtlichen Übergängen von Tag zu Tag Werte um die 0 aufweisen.

Werden die Prognoseergebnisse in einem wöchentlichen Rhythmus betrachtet, Abbildung 6.13 und für den täglichen Versuch zusammengefasst in Abbildung 6.14, sehen die

Testprognosen wesentlich besser aus. Aufgrund der wöchentlichen Mittelwerte sind die Kurven etwas geglättet worden, wodurch die Peaks weniger präsent sind und sich die Kurven nun ähneln.

Dass das Vorhersagen der Höchstwerte in den Solardaten durch die Modelle schwierig war, deckt sich mit den errechneten MAE-Werten. Diese lagen recht nah an den einfachen Baseline-Schätzungen. Lediglich bei der multivariaten Methode waren alle MAE-Werte niedriger als die der Baseline-Modelle. Zwischen den Modellen lagen die Abweichungen maximal bei 500 Megawatt. Anhand dessen eine Architektur zu bestimmen, welche die besten Ergebnisse geliefert hat, ist nur bedingt möglich. Zwar hatten die LSTM-Modelle meistens die niedrigsten Fehlerwerte. Aus einer reinen Netzstruktur aus LSTM-Schichten bestanden sie jedoch auch nicht, wie in Unterkapitel 6.5 und 6.6 beschrieben wurde. So mussten den LSTM- und Autoencoder-Modellen konvolutionale Schichten zur Netzstruktur hinzugefügt werden, um gute Ergebnisse zu erzielen. Aufgrund der ähnlichen Ergebnisse und keiner Prognose, die besonders gut abgeschnitten hat, wird bei der zukünftigen Vorhersage darauf geachtet, ob die Vorhersagen generell realistisch aussehen und ob die wiederauftretenden Muster in den Daten zu erkennen sind.

6.7.2 Vergleich der zukünftigen Prognosen

Bei der Vorhersage von Daten innerhalb und außerhalb des Datensatzes hat sich im Laufe der Versuchsdurchführungen eine längere Inputsequenz bewährt. Bei der Verwendung der rekursiven Vorhersagemethode wurden die Inputsequenzen dadurch nicht zu schnell durch neue Prognosewerte ersetzt, was den generellen Prognosefehler reduziert hat.

Beim stündlichen Versuch führte die Vorhersage von Daten außerhalb des Datensatzes insgesamt recht schnell zu unrealistischen Ergebnissen, wie in Abbildung 6.11 dargestellt ist. Zum einen haben alle Kurven einen recht eigenen Verlauf abgebildet, zum anderen steigt besonders die Kurve der LSTM-Prognosen für den Monat Januar unrealistisch stark an, was etwas widersprüchlich zu den zuvor gemessenen niedrigen MAE-Werten steht. Eine Möglichkeit, hier bessere Ergebnisse zu erhalten, wäre die Nutzung einer noch größeren Sequenzlänge als die verwendeten 168 (was sieben Tagen entspricht). Da bei der stündlichen Frequenz die Werte aus einer recht kleinen Intervallgröße besteht, müssten viele Datenpunkte prognostiziert werden, um wenigstens ein paar Tage vorhersagen zu können.

Beides wurde beim Versuch mit täglicher Frequenz umgesetzt. Die Sequenzlänge beträgt hier 365, wodurch ein ganzes Jahr abgedeckt wird. Hinsichtlich des Prognosefehlers können dadurch wesentlich mehr Werte und somit Tage vorhergesagt werden, da die Sequenz über einen längeren Zeitraum noch aus originalen Datenpunkten besteht. Werden z.B. 60 Tage vorhergesagt, besteht der Anteil an Datenpunkten in der Inputsequenz überwiegend aus originalen Datenpunkten.

Auch wenn mit dem FFNN bei der multivariaten Methode kein Ergebnis erzielt wurde, so waren die restlichen Prognoseergebnisse besser als bei den anderen Versuchen. Wird der Zeitraum von einem Jahr betrachtet, konnten überwiegend ähnliche Vorhersagen durch LSTM, Autoencoder und CNN erzielt werden, was in Abbildung 6.18 zu sehen ist.

Die Aufgabe, Zeitreihenvorhersagen und das über einen längeren Zeitraum mithilfe von neuronalen Netzen durchzuführen, wurde größtenteils erfüllt. Inwieweit die prognostizierten Werte der Wahrheit entsprechen, lässt sich jedoch nicht sagen, da keine vergleichbaren Werte herangezogen werden können. Wiederkehrende Muster wie der Tag-Nacht- oder Winter-Sommer-Wechsel konnten in den Prognoseergebnissen von allen Modellen abgebildet werden.

6.7.3 Vergleich mit zurückliegenden Werten und Diskussion

In diesem Abschnitt werden die vorhergesagten drei Jahre der täglichen Prognoseergebnisse in Relation zu den vergangenen Jahren gesetzt.

In den Abbildungen 6.19 und 6.20 sind die monatlichen und jährlichen Mittelwerte der Solardaten enthalten. Trotz erster, mit Skepsis zu betrachtender Ergebnisse, sehen die LSTM-Kurven, unter Berücksichtigung des Klimaziels, in beiden Abbildungen im jährlichen Schnitt am realistischsten aus. Mit Berücksichtigung des Klimawandels ist gemeint, dass aufgrund der politischen Zielsetzung bzgl. erneuerbarer Energien, ein Anstieg dieser in der Vorhersage zu erwarten ist.

In der zweiten Abbildung 6.20, wo die multivariate Methode eingesetzt wurde, weisen allerdings alle jährlichen Kurven einen ähnlich realistischen Verlauf auf. Nur die Jahreskurven zu betrachten, ist jedoch auch nicht zielführend, wie die Jahreskurve des FFNN in Abbildung 6.19 zeigt. Auch wenn eine so starke Zunahme der Energieerzeugung durch Solarstrom schön wäre, die Basis dieses Graphen ergibt sich aus unrealistischen Ergebnissen, wie im selbigen Plot an den Jahresübergängen, vor allem am Ende vom Jahr

2024, zu erkennen ist. In diesem Zeitraum, den Wintermonaten, sind die Prognosen im Vergleich vergangener Jahre unrealistisch hoch. Wird für eine Bewertung wieder das Klimaziel berücksichtigt, weist das LSTM-Modell und das CNN-Modell die realistischeren jährlichen Kurven auf.

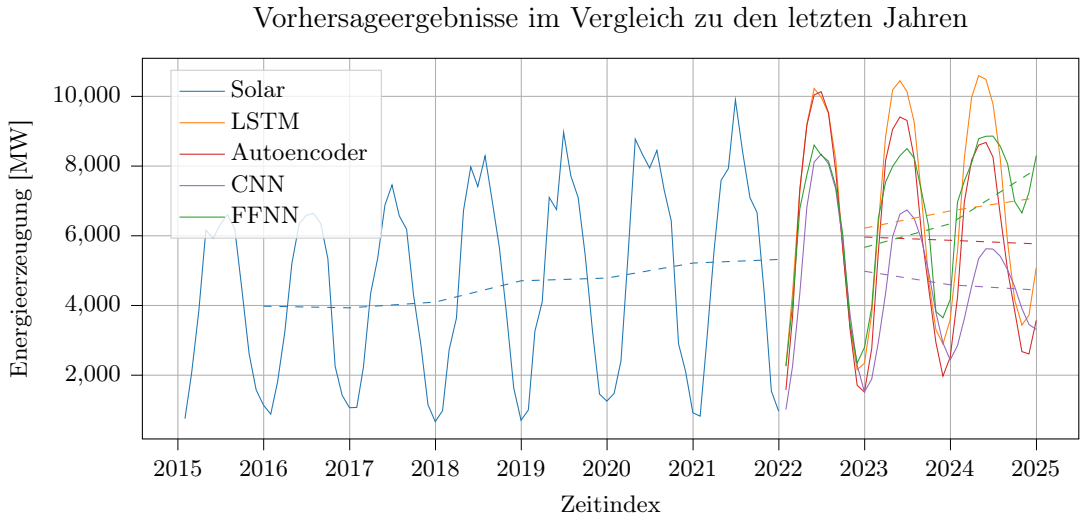


Abbildung 6.19: Univariate Methode: Vorhersage von unbekannten Daten durch FFNN, LSTM, Autoencoder und CNN über einen Zeitraum von drei Jahren in Relation zu den letzten sieben Jahren.

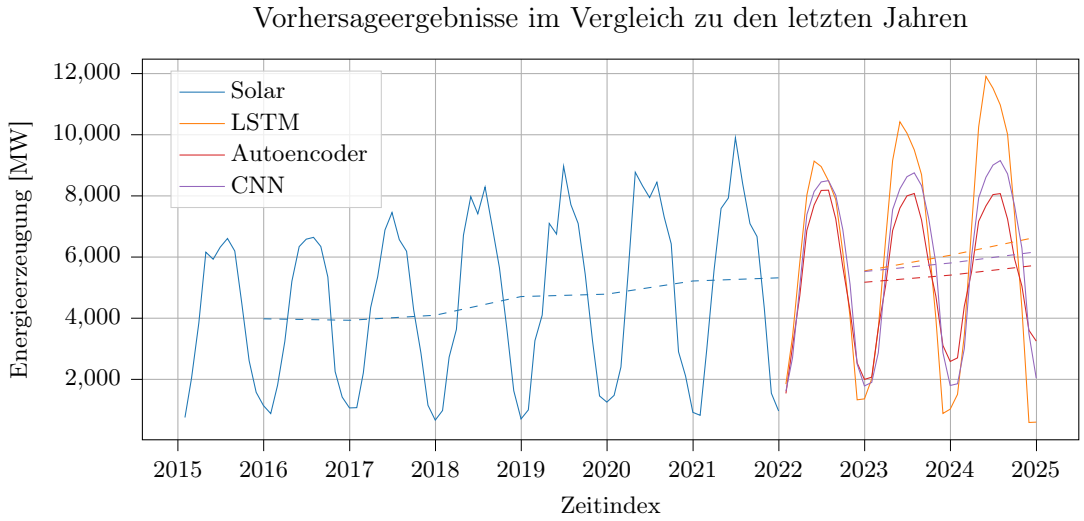


Abbildung 6.20: Multivariate Methode: Vorhersage von unbekannten Daten durch LSTM, Autoencoder und CNN über einen Zeitraum von drei Jahren in Relation zu den letzten sieben Jahren.

Nach ersten Beurteilungen, verfasst in Kapitel 6.1, wirkte der Datensatz noch qualitativ geeignet und bestand aus nahezu lückenlosen sieben Jahren an Daten, von Beginn 2015 bis Ende 2021. Erst durch die alleinige Berücksichtigung der Solardaten und darüber hinaus dem Hinzufügen konvolutionaler Schichten zum Netzaufbau des LSTM und Autoencoder, konnten auch mit diesen Modellen Ergebnisse erzielt werden.

Weshalb die zukünftigen Prognosen zwischen den Modellen beim Versuch mit stündlicher Frequenz im Vergleich zur täglichen Frequenz, einen so deutlichen Unterschied ausmachten, ist schwer zu beurteilen. Nach erster Bewertung ist dieses Ergebnis vermutlich auf die Länge der Inputsequenz im stündlichen Versuch von nur sieben Tagen zurückzuführen, welche im Laufe dieser Prognosen dreimal vollständig durch eigene Prognosewerte ersetzt wurden und den Prognosefehler steigen ließ. Beim Versuch mit täglicher Frequenz wurde eine doppelt so lange Sequenzlänge von einem Jahr bzw. 365 Werten gewählt. Dadurch konnte eine ganze Saison abgebildet werden, welche im Vergleich zu den vielen täglichen Peaks aus dem ersten Ansatz, jetzt nur noch einen Peak pro Jahr enthielt. Bei der Vorhersage von drei Jahren wurde auch bei diesem Versuch die Inputsequenz mehrere Male durch eigene Prognosen ersetzt. Saisonbedingt ist in diesen drei Jahren jeweils nur einen Peak im Sommer vorherzusagen, was zur Folge hat, dass der Prognosefehler als wesentlich geringer einzuschätzen ist, als beim Versuch mit stündlicher Frequenz.

Zusammengefasst haben die Architekturen CNN und eine Struktur aus LSTM- und konvolutionalen Schichten beim stündlichen Versuch anhand der Testprognosen die vergleichsweise besten Ergebnisse geliefert. Diese Ergebnisse stehen etwas im Widerspruch zu den Prognosen über einen unbekannten Zeitraum außerhalb des Datensatzes. Bei der Betrachtung der jährlichen Kurven können bei univariaten Methode realistische Ergebnisse beim LSTM und bei der multivariaten Methode bei allen Strukturen (LSTM, Autoencoder und CNN) beobachtet werden.

7 Zusammenfassung und Fazit

Ziel dieser Arbeit war es, anhand mehrerer Versuche die neuronalen Netz-Architekturen FFNN, LSTM, Autoencoder und CNN bzgl. des Einsatzes zur Zeitreihenvorhersage zu vergleichen. Hierbei sollte überprüft werden, ob LSTM-Modelle bessere und realistischere Ergebnisse liefern als die anderen Architekturen. Des Weiteren wurde untersucht, ob durch die Vorhersage von Energiedaten ein Trend zur vermehrten Stromerzeugung durch erneuerbare Energien in Deutschland zu erkennen ist. Die Ergebnisse dieser Arbeit werden im Folgenden zusammengefasst. Anschließend wird ein Fazit gezogen.

7.1 Zusammenfassung

In Kapitel 2.2 wurde auf die Besonderheit von Zeitreihen, Regressionen und von neuronalen Netzen eingegangen und begründet, weshalb sich für bestimmte Verfahren in dieser Arbeit entschieden wurde. Die verwendeten Architekturen FFNN, LSTM, Autoencoder und CNN wurden in Kapitel 3 vorgestellt. Eine grobe Beschreibung des methodischen Vorgehens und der technischen Umgebung erfolgte in Kapitel 4.

Die Ergebnisse der einfacheren Aufgabenstellungen, die Vorhersage einer Zahlenreihe und der Sinuskurve, wurden in Kapitel 5 untersucht. Die besten Testprognosen konnten im ersten Versuch durch ein LSTM-Modell und im zweiten Versuch durch einen Autoencoder erzielt werden. Die beste Modellstabilität, also gute Prognosen bei aufeinanderfolgenden Trainings, wies hierbei das CNN auf.

Im Hauptexperiment in Kapitel 6, der Vorhersage von Energiewerten, stellte sich nach ersten Versuchsdurchführungen heraus, dass die meisten Energiekurven einen zu chaotischen Verlauf aufwiesen. Dadurch war es schwierig, aussagekräftige Ergebnisse zu erzielen. Aufgrund des periodischen und zugleich saisonalen Verlaufs der Solardaten wurde sich daraufhin nur noch auf diese einzelne Sparte bezogen und die Versuche mit stündlichen

und täglichen Frequenzwerten weitergeführt. Generell konnten die Modelle gute Vorhersagen bzgl. des Tag-Nacht- bzw. Sommer-Winter-Rhythmuses erzielen. Tageshöchstwerte in den Solardaten traten jedoch willkürlich auf, was auch in der Genauigkeit der Modellprognosen zu erkennen war. Für die anschließenden Versuche wurden rekursive Vorhersagen verwendet, bei denen die Prognosen auf den eigenen Prognosen basiert.

Beim stündlichen Versuch wurde basierend auf einer Inputsequenz von sieben Tagen (umgerechnet 168 Werte) ein Tag, also 24 Werte, vorhergesagt. Nach mehreren Stichproben aus den Testdaten erwies sich hierbei aufgrund des niedrigsten MAE die LSTM-Struktur als das geeignetste Modell. Während nur das LSTM-Modell bessere Prognosen als die verwendeten Baseline-Modelle erzielen konnte, erwies sich der Autoencoder am ungenausten. Bei der Prognose von unbekannten Daten, welche außerhalb des Datensatzes lagen, wurden bewusst mehr Datenpunkte vorhergesagt, als bei dem Vergleich der Prognosen mit den Testdaten. Hierbei wurde untersucht, wie weit die Vorhersagen fortgeführt werden konnten, bevor die Modellprognosen unrealistische Ergebnisse aufgewiesen haben. Getestet wurde das Vorhersagen von 21 Tagen. Widersprüchlich war hierbei, dass das zuvor vermeintlich beste Modell (LSTM) Werte vorhergesagt hatte, die den unrealistischsten Kurvenverlauf bildeten, während die Kurve des Autoencoder wesentlich realistischer einzuschätzen war. Da dieses Ergebnis nicht zufriedenstellend war, wurde der Versuch nicht mit der multivariaten Methode durchgeführt.

Bei der Analyse täglicher Frequenzwerte unter Verwendung der univariaten und der multivariaten Methode wurden bessere Ergebnisse erzielt, obwohl die Datenmenge aufgrund der größeren Frequenz gravierend reduziert worden war. Die gewählte Sequenzlänge betrug hierbei 365 und deckte dadurch ein ganzes Jahr ab. Vorhergesagt wurden 180 Werte. Aus den Ergebnissen der univariaten Untersuchung geht hervor, dass das CNN die besten Testprognosen abgegeben hat. Generell bildeten die Vorhersagen der Architekturen zum Teil sehr unterschiedliche Kurven. Hierbei waren jedoch, bis auf das FFNN, alle MAE-Werte niedriger als die der Baseline-Modelle. In der jährlichen Vorhersage über einen außerhalb des Datensatzes liegenden Zeitraum von drei Jahren, wies die Kurve der LSTM-Vorhersage einen realistischeren Verlauf auf als die der anderen Architekturen.

Um im täglichen Versuch aus dem univariaten einen multivariaten Datensatz zu transformieren, wurden neben der Spalte der Solardaten weitere Spalten für Jahr, Monat, Woche und Tag aus dem Zeitstempel extrahiert und dem Datensatz für mehr Komplexität hinzugefügt. Das FFNN konnte wegen einer technischen Limitierung nicht berücksichtigt werden. Den niedrigsten Fehlerwert (MAE) konnte das LSTM-Modell erzielen. Gene-

rell waren alle Testprognosen besser als die der Baseline-Modelle. Bei der multivariaten Analyse war aufgefallen, dass es keine großen Unterschiede zwischen den Prognosen der verschiedenen Architekturen gab. Auch bei der Vorhersage des gleichen unbekannten Zeitraums über drei Jahre wie im univariaten Versuch, wiesen in der Darstellung der jährlichen Kurven alle Modelle (LSTM, Autoencoder, CNN) gute und realistische Ergebnisse auf.

Es ist zu erwähnen, dass den LSTM- und Autoencoder-Modellen bei der Vorhersage von Energiewerten konvolutionale Schichten zur Netzstruktur hinzugefügt werden mussten, um gute Ergebnisse zu erzielen.

7.2 Fazit

Anhand der zusammengefassten Ergebnisse aus den Versuchsdurchführungen, der Vorhersage einer Zahlenreihe, der Sinuskurve sowie stündlicher und täglicher Solardaten, kann folgendes Fazit gebildet werden.

Bei einfacheren Zeitreihen, der Vorhersage einer Zahlenreihe und der Sinuskurve, konnten ein LSTM-Modell und ein Autoencoder die besten Ergebnisse erzielen. Im Versuch Solardaten in stündlicher Frequenz vorherzusagen, konnte ein LSTM-CNN-Modell den niedrigsten Fehlerwert (MAE) erreichen. Bei der Prognose von Solardaten in täglicher Frequenz erzielte unter Verwendung der univariaten Methode ein CNN und unter Verwendung der multivariaten Methode ein LSTM-CNN-Modell die besten Testprognosen. Bei der Vorhersage von Solardaten, die außerhalb des Testzeitraums lagen, erwiesen sich CNN- und LSTM-CNN-Modelle als am besten geeignet. Anhand dieser Ergebnisse kann zusammenfassend festgestellt werden, dass LSTM-CNN-Modelle größtenteils die besseren Prognosen bei der Zeitreihenvorhersage lieferten.

Die Untersuchung der Architekturen, ob durch die Vorhersage mithilfe neuronaler Netze eine vermehrte Stromerzeugung durch erneuerbare Energien über einen längeren Zeitraum zu erkennen ist, kam zu keinem allgemein repräsentativen Ergebnis. Allerdings konnte die Kurve der Solardaten bei Verwendung der multivariaten Methode für ein bis drei Jahre vorhergesagt und in diesen Werten ein Trend zum Anstieg der Erzeugung von Solarstrom erkannt werden.

8 Ausblick

Zeitreihendaten sind der Schlüssel zur Prognose der Zukunft. [14] Das Vorhersagen für solche Zeitreihen basiert dementsprechend auf Daten. Nicht umsonst ist die Datenvorverarbeitung also ein grundlegender Schritt in Machine Learning Anwendungen. Wenn die Datenbasis von guter Qualität ist, können neuronale Netze auch gute Ergebnisse erzielen.

Um die Ergebnisse dieser Arbeit zu erweitern, kann an mehreren Punkten angesetzt werden. Der erste Gedanke gilt hier der Wahl der Daten, die sich in den Versuchsdurchführungen als nicht optimal herausgestellt haben. Ein Ziel dieser Arbeit, mehrere Modellarchitekturen miteinander zu vergleichen, hätte z.B. mit einem sich für Zeitreihenvorhersagen geeigneten Datensatz durchgeführt werden können, wie dem M3, M4 oder ETTh1, welche in vergleichbaren Arbeiten [24, 25] verwendet wurden.

Da es weitere Stellschrauben gibt, die das Training eines neuronalen Netzes beeinflussen wie z.B. Verlustfunktionen, Größe der Batch Size, Optimierer, Anzahl an Schichten und Neuronen, Kombinationen aus verschiedenen Strukturen und vielen weiteren Faktoren, kann der Fokus im Weiteren auf eine tiefergehende Untersuchung einer bestimmten Architektur gelegt werden.

Neben der One-Step-Ahead Methode, nach welcher der TimeseriesGenerator in diesen Versuchsreihen ausgerichtet war, kann ein Horizont für die Analyse eingesetzt werden. [14] Der Horizont beschreibt einen Zeilenabstand bzw. Zeitabstand, zwischen Inputsequenz und dem Wert, welcher vorhergesagt werden soll. Der Vorteil dieser Methode ist, dass wenn für den Horizont der Wert 30 gewählt wurde, 30 Datenpunkte im Anschluss des Datensatzes vorhergesagt werden können bevor Prognosewerte der Inputsequenz hinzugefügt werden. Dies gilt, sofern die rekursive Vorhersagestrategie genutzt wird. Sicher ist es dennoch nicht, dass auf diesem Wege gute Prognosen erzielt werden, wie die Untersuchungen in [18] zeigen. Hier war die fehlerhafte Abweichung (MAE) durch größere

Horizontwerte gestiegen. [18] Abgesehen von der Wahl an Daten oder Trainingsparametern kann demzufolge auch eine andere Vorhersagestrategie für die Untersuchung herangezogen werden. Eine bewährte Methode ist die Sequenz-zu-Sequenz-Vorhersage. [32, 38] Genauso wie bei der rekursiven Variante, werden hierbei Sequenzen zum Antrainieren der Netze verwendet. Die Datenstruktur wird jedoch so vorbereitet, dass ein neuronales Netz nicht nur einen y-Wert ausgibt, sondern eine Sequenz von y-Werten. Übertragen auf das Hauptexperiment in dieser Arbeit, könnten die Daten erst in eine monatliche Frequenz transformiert werden, bevor die Vorhersage des jeweiligen Modells durchgeführt wird. Je nach Größe der Output-Sequenz hätte z.B. eine Länge von sechs ausgereicht, um aus einer Sequenz die Vorhersage für sechs Monate ablesen zu können. Für eine monatliche Frequentierung wäre der in dieser Arbeit genutzte Datensatz jedoch zu klein gewesen und bestünde lediglich aus 84 Solarwerten. Da die meisten existierenden Methoden für die Vorhersage von wenigen Werten entworfen sind, könnte auch ein solcher Sequenz-zu-Sequenz-Ansatz weiter verfolgt werden [15, 38].

Obwohl rekurrente Netze, speziell die LSTM-Modelle seit der Einführung von Hochreiter, in großem Stil im Bereich der Zeitreihenvorhersagen mit neuronalen Netzen verwendet wurden [2, 14, 32, 34], wird in aktuellen Arbeiten auf Encoder-/Decoder-Strukturen gesetzt, wie z.B. dem Autoencoder oder Transformer [9, 18, 38]. Ein weiterer Ansatz wäre, diese Architekturen bei zeitreihenrelevanten Aufgabenstellungen noch genauer zu betrachten.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Backhaus, Klaus ; Erichson, Bernd ; Plinke, Wulff ; Weiber, Rolf: Multivariate Analysemethoden. Springer Gabler, 2018. – ISBN 978-3-662-56654-1

[2] Bandara, Kasun ; Bergmeir, Christoph ; Smyl, Slawek: Forecasting Across Time Series Databases using Recurrent Neural Networks on Groups of Similar Series: A Clustering Approach. 2018. – URL https://arxiv.org/abs/1710.03222

[3] Bauchmüller, Michael: Anteil fossiler Energie stagniert seit zehn Jahren. 2021. – URL https://www.sueddeutsche.de/wissen/erneuerbare-energien-leere-worte-1.5322634. – Zugriffsdatum: 17.04.2022

[4] Bischoff, Manon: Künstliche Intelligenz: Vom Schachspieler zur Superintelligenz? Springer, 2022. – ISBN 978-3-662-62491-3

[5] Brownlee, Jason: A Gentle Introduction to LSTM Autoencoders. 2020. – URL https://machinelearningmastery.com/lstm-autoencoders/. – Zugriffsdatum: 10.06.2022

[6] Bundesregierung: C02 Emissionen. Bundesregierung, 2020. – URL https://www.bundesregierung.de/breg-de/themen/energiewende/co2-kohlenstoffdioxid-oder-kohlendioxid-emission-614692. – Zugriffsdatum: 16.04.2022

[7] Bundesregierung: Deutsche Nachhaltigkeitsstrategie. 2021. – URL https://www.bundesregierung.de/breg-de/aktuelles/nachhaltigkeitsstrategie-2021-1873560. – Zugriffsdatum: 18.04.2022

[8] Degrave, Jonas ; Felici, Federico ; Buchli, Jonas ; Neunert, Michael ; Tracey, Brendan ; Carpanese, Francesco ; Ewalds, Timo ; Hafner, Roland ; Abdolmaleki, Abbas ; Casas, Diego ; Donner, Craig ; Fritz, Leslie ; Galperti, Cristian ; Huber, Andrea ; Keeling, James ; Tsimpoukelli, Maria ; Kay, Jackie ; Merle,

Antoine ; Moret, Jean-Marc ; Riedmiller, Martin: Magnetic control of tokamak plasmas through deep reinforcement learning. In: Nature 602 (2022), 02, S. 414–419

[9] Du, Shengdong ; Li, Tianrui ; Horng, Shi-Jinn: Time Series Forecasting using Sequence-to-Sequence Deep Learning Framework. In: 2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), IEEE, 2018, S. 171–176. – URL https://ieeexplore.ieee.org/document/8701741

[10] Ertel, Wolfgang: Grundkurs Künstliche Intelligenz. Springer Vieweg, 2021. – ISBN 978-3-658-32074-4

[11] Goodfellow, Ian ; Courville, Aaron C. ; Bengio, Yoshua: Deep Learning. Das umfassende Handbuch. mitp, 2018. – ISBN 978-3-95845-701-0

[12] Haeseler, Susanne ; Bissolli, Peter ; Dassler, Jan ; Zins, Volker ; Kreis, Andrea: Orkantief Sabine löst am 09./10. Februar 2020 eine schwere Sturmlage über Europa aus. DWD, 2020. – Forschungsbericht. – URL https://www.dwd.de/DE/leistungen/besondereereignisse/stuerme/20200213_orkantief_sabine_europa.html

[13] Heaven, Will D. ; Stieler, Wolfgang: Unendliche saubere Energie? DeepMind-KI soll Plasma in Fusionsreaktor steuern. heise online, 2022. – URL https://www.heise.de/hintergrund/Unendliche-saubere-Energie-DeepMind-KI-soll-Plasma-in-Fusionsreaktor-steuern-6488777.html. – Zugriffsdatum: 17.05.2022

[14] Hirschle, Jochen: Machine Learning für Zeitreihen. Hanser, 2021. – ISBN 978-3-446-46726-2

[15] Hochreiter, Sepp ; Schmidhuber, Jürgen: Long Short-Term Memory. In: Neural Computation 9 (1997), 11, Nr. 8, S. 1735–1780. – URL https://doi.org/10.1162/neco.1997.9.8.1735

[16] Hude, Marlis von der: Predictive Analytics und Data Mining. Springer Vieweg, 2020. – ISBN 978-3-658-30152-1

[17] Khaldi, Rohaifa ; Chiheb, Raddouane ; Afia, Abdellatif E.: Feedforward and Recurrent Neural Networks for Time Series Forecasting: Comparative Study. In: Proceedings of the International Conference on Learning and Optimization

Algorithms: Theory and Applications. New York, NY, USA : Association for Computing Machinery, 2018 (LOPAL '18). – URL https://doi.org/10.1145/3230905.3230946. – ISBN 9781450353045

[18] Li, Yaguang ; Yu, Rose ; Shahabi, Cyrus ; Liu, Yan: Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. 2017. – URL https://arxiv.org/abs/1707.01926

[19] Lippe, Peter von der: Deskriptive Statistik. Gustav Fischer Verlag, 1993. – ISBN 3-437-40268-4

[20] Litz, Philipp ; Thuy, Nga N. ; Graichen, Dr. P.: Warum Deutschlands neue Klimaziele den Kohleausstieg bis 2030 besiegeln. Agora Energiewende, 2021. – URL https://www.agora-energiewende.de/blog/warum-deutschlands-neue-klimaziele-den-kohleausstieg-bis-2030-besiegeln/. – Zugriffsdatum: 17.05.2022

[21] Masson-Delmotte, Valérie ; Pörtner, Hans-Otto ; Skea, Jim ; Pirani, Anna ; Pidcock, Roz ; Chen, Yang ; Moufouma-Okia, Wilfran ; Connors, Sarah ; Zhou, Xiao ; Zhai, Panmao ; Roberts, Debra ; Shukla, Priyadarshi R. ; Péan, Clotilde ; Matthews, J. B. R. ; Gomis, Melissa I. ; Lonnoy, Elisabeth ; Maycock, Tom ; Tignor, Melinda ; Waterfield, Tim: 1,5 °C Globale Erwärmung. IPCC, 2018. – Forschungsbericht. – URL https://www.de-ipcc.de/256.php

[22] Max-Planck-Institut: Was ist Kernfusion? Max-Planck-Institut für Plasmaphysik, 2022. – URL https://www.ipp.mpg.de/ippcms/de/pr/fusion21/kernfusion/index. – Zugriffsdatum: 17.05.2022

[23] Olah, Christopher: Understanding LSTM Networks. 2015. – URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/. – Zugriffsdatum: 02.05.2022

[24] Oreshkin, Boris N. ; Carpov, Dmitri ; Chapados, Nicolas ; Bengio, Yoshua: N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. 2019. – URL https://arxiv.org/abs/1905.10437

[25] PapersWithCode: Time Series Forecasting. PapersWithCode, 2022. – URL https://paperswithcode.com/task/time-series-forecasting. – Zugriffsdatum: 08.06.2022

[26] Philip, Sjoukje Y. ; Kew, Sarah F. ; Oldenborgh, Geert J. van ; Anslow, Faron S. ; Seneviratne, Sonia I. ; Vautard, Robert ; Coumou, Dim ; Ebi, Kristie L. ; Arrighi, Julie ; Singh, Roop ; Aalst, Maarten van ; Marghidan, Carolina P. ; Wehner, Michael ; Yang, Wenchang ; Li, Sihan ; Schumacher, Dominik L. ; Hauser, Mathias ; Bonnet, Rémy ; Luu, Linh N. ; Lehner, Flavio ; Gillett, Nathan ; Tradowsky, Jordis ; Vecchi, Gabriel A. ; Rodell, Chris ; Stull, Roland B. ; Howard, Rosie ; Otto, Friederike E. L.: Rapid attribution analysis of the extraordinary heatwave on the Pacific Coast of the US and Canada June 2021. In: Earth System Dynamics Discussions 2021 (2021), S. 1–34. – URL https://www.worldweatherattribution.org/western-north-american-extreme-heat-virtually-impossible-without-human-caused-climate-change/

[27] Purwanto, Dr. ; Eswaran, Chikkannan ; Logeswaran, Rajasvaran: A Comparison of ARIMA, Neural Network and Linear Regression Models for the Prediction of Infant Mortality Rate. In: 2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation, IEEE, 2010, S. 34–39

[28] Rüth, Petra v. ; Schönthaler, Konstanze ; Andrian-Werburg, Stefan von ; Buth, Mareike: Monitoringbericht 2019 zur Deutschen Anpassungsstrategie an den Klimawandel. Umweltbundesamt, 2019. – Forschungsbericht. – URL https://www.umweltbundesamt.de/publikationen/umweltbundesamt-2019-monitoringbericht-2019-zur

[29] Sammut, Claude ; Webb, Geoffrey I.: Encyclopedia of Machine Learning and Data Mining. S. 781–781. In: Loss. Boston, MA : Springer US, 2017. – URL https://doi.org/10.1007/978-1-4899-7687-1_499. – ISBN 978-1-4899-7687-1

[30] Santen, Manfred: Der Recycling Mythos 2.0. 2020. – URL https://www.greenpeace.de/engagieren/nachhaltiger-leben/exportschlager-umweltgifte. – Zugriffsdatum: 17.04.2022

[31] Schneider, Gerd ; Toyka-Seid, Christiane: Fridays for Future. 2022. – URL https://www.bpb.de/kurz-knapp/lexika/das-junge-politik-lexikon/320328/fridays-for-future/. – Zugriffsdatum: 17.04.2022

[32] Sutskever, Ilya ; Vinyals, Oriol ; Le, Quoc V.: Sequence to Sequence Learning with Neural Networks. 2014. – URL https://arxiv.org/abs/1409.3215

[33] Umweltrat: Umweltgutachten 2020: Für eine entschlossene Umweltpolitik in Deutschland und Europa. Umweltrat, 2020. – Forschungsbericht. – ISBN 978-3-947370-16-0

[34] Wennker, Phil: Künstliche Intelligenz in der Praxis. Springer Gabler, 2020. – ISBN 978-3-658-30479-9

[35] Wiederschein, Harald: Die spektakulärsten Aktionen von Greenpeace. 2013. – URL https://www.focus.de/wissen/mensch/die-spektakulaersten-aktionen-von-greenpeace-40-jahre-kampf-fuer-die-umwelt_id_2273507.html. – Zugriffsdatum: 17.04.2022

[36] Zaunseder, Elaine ; Müller, Larissa ; Blankenburg, Sven: High Accuracy Forecasting with Limited Input Data: Using FFNNs to Predict Offshore Wind Power Generation. In: Proceedings of the Ninth International Symposium on Information and Communication Technology. New York, NY, USA : Association for Computing Machinery, 2018 (SoICT 2018), S. 61–68. – URL https://doi.org/10.1145/3287921.3287936. – ISBN 978-1-450-36539-0

[37] Zhang, Jie ; Hodge, Bri-Mathias ; Lu, Siyuan ; Hamann, Hendrik F. ; Lehman, Brad ; Simmons, Joseph ; Campos, Edwin ; Banunarayanan, Venkat ; Black, Jon ; Tedesco, John: Baseline and target values for regional and point PV power forecasts: Toward improved solar forecasting. In: Solar Energy 122 (2015), S. 804–819. – URL https://www.sciencedirect.com/science/article/pii/S0038092X1500540X. – ISSN 0038-092X

[38] Zhou, Haoyi ; Zhang, Shanghang ; Peng, Jieqi ; Zhang, Shuai ; Li, Jianxin ; Xiong, Hui ; Zhang, Wancai: Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. 2020. – URL https://arxiv.org/abs/2012.07436

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort Datum Tobias Rawald