

Intelligente Datenanalyse mit Python

Darstellung und Vorhersage von Kryptowährungen mithilfe von LSTM-Netzen

von

Jonas Birkle

und

Florian Zitz

Prüfer

Prof. Dr. Sebastian Dörn

Fakultät Industrial Technologies

Tuttlingen, den 04.02.2021

1 Einleitung und Problemstellung

Seit einigen Jahren erfreuen sich sogenannte Kryptowährungen steigender Beliebtheit. Trotz eines teils sehr starken Wertanstiegs, zum Beispiel des Bitcoins mit über 250% seit Februar 2020 [3], sind Kryptowährungen vor allem durch hohe Preisschwankungen in kurzen Zeiträumen ein hoch spekulatives Geldanlagegebiet. Deutlich wurde dies Anfang 2020, als der Preis von kurzzeitig über 40.000\$ quasi über Nacht um knappe 10.000\$ abfiel. Umso besser wäre es also in die Zukunft schauen zu können, um auf drohende Preisstürze frühzeitig reagieren zu können. Eine Möglichkeit bieten hierbei die in den letzten Jahren immer weiter erforschten und verbreiteten Künstlichen Intelligenzen. Rekurrente neuronale Netze sind in der Lage Zeitintervalle zu verarbeiten und können sich dabei an vergangenes Erinnern. Vor allem die in dieser Arbeit verwendeten LSTM (Long Short Term Memory) Netzwerke bieten sich für die Verarbeitung großer Zeitreihen an.

Das Ziel dieser Anwendung ist es, das Risiko beim Handel mit Kryptowährungen zu minimieren indem mithilfe von LSTM Netzen zuverlässige Voraussagen getroffen werden können. Die sehr komplexen Zusammenhänge bei Prognosen im Finanzsektor sollen hiermit auch Laien etwas greifbarer gemacht werden.



2 Benutzeroberfläche

Die Benutzeroberfläche wurde mithilfe des Python Moduls „Kivy“ [1] erstellt. Kivy bietet die Möglichkeit plattformübergreifende Applikationen zu erstellen. Der Grundaufbau der Oberfläche erfolgt zunächst hauptsächlich in der Kv Programmiersprache [2]. Die so erstellten Widgets werden dann durch ein Python Skript mit den notwendigen Funktionen versehen.

2.1 Menü

Das erste was der Nutzer beim Start der Applikation zu sehen bekommt ist das Hauptmenu (Abbildung 1 links). Falls die App seit einiger Zeit nicht genutzt wurde, sollten zunächst die Datenbankeinträge aktualisiert werden („Update Database“). Sofern die Datenbank auf dem neuesten Stand ist können nach dem Wechsel auf die zweite Seite des Menus (Abbildung 1 rechts) über den Button „Select Coins“ bis zu drei Währungen zur Anzeige ausgewählt werden. In der aktuellen Version stehen die Währungen Bitcoin („btc“), Ethereum („eth“) und Litecoin („ltc“) zur Verfügung.

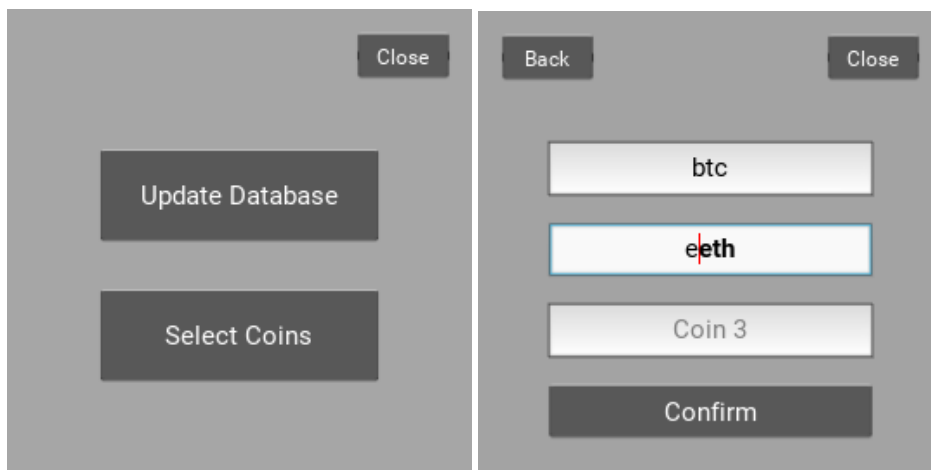


Abbildung 1: Die beiden Menü Bildschirme, rechts: Textfelder mit Autovervollständigung

Bei der Eingabe der Währungen wird der Nutzer durch die automatische Vervollständigung seiner Eingabe unterstützt. Sobald die eingetragenen Coins mit der Betätigung des „Confirm“ Knopfes bestätigt werden beginnt das Programm mit der Abfrage der benötigten Daten aus der verbundenen Datenbank. Sobald alle Daten abgerufen und alle Rechnungen abgeschlossen sind wird in der Datenansicht das Ergebnis präsentiert.

2.2 Datenansicht

Die abgerufenen Daten werden dem Nutzer in einem übersichtlichen Fenster als Graph dargestellt. Ist nur eine Währung ausgewählt wird der Preisverlauf dargestellt, bei zwei oder mehr Währungen werden jeweils die prozentualen Veränderungen im Vergleich zum ersten Wert im Beobachtungsintervall dargestellt. Detailliertere Informationen zur bisherigen Performance der Coins kann der

2

Nutzer der Legende auf der rechten Seite des Fensters entnehmen. Die Kürzel der einzelnen Währungen sind jeweils in der Farbe des zugehörigen Graphen eingefärbt. Die Übersicht stellt neben dem aktuellen Verkaufswert („Value“) und dessen prozentualen Vergleich mit dem Verkaufswert zum Start und den Maximal und Minimalwerten im Verlauf des Beobachtungsintervalls auch eine Vorhersage über die zukünftige Entwicklung der Preise bereit. Dieser Wert wird mithilfe einer Künstlichen Intelligenz ermittelt.



Abbildung 2: Übersicht der ausgewählten Coins

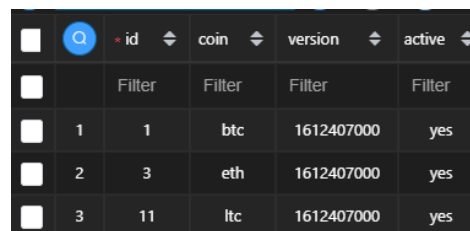
Zur besseren Einordnung dieser Werte wird zudem auch noch die prozentuale Veränderung zwischen Vorhersage und aktuellem Verkaufswert angezeigt. Vorhersagen sind, in der Tagesansicht, für 6 Stunden und, in den anderen Ansichten, für einen Tag möglich. Das Beobachtungsintervall lässt sich über die vier Buttons unmittelbar über dem Graphen einstellen, rechts daneben befinden sich ein Button zur Rückkehr auf den Startbildschirm und ein Button zur schnellen Aktualisierung der Werte der ausgewählten Coins.

2.3 Konsole

Zur Übersicht über die zeitaufwendigen Aufgaben im Hintergrund der Anwendung werden mithilfe der Konsole Fortschrittsbalken ausgegeben, an denen der Nutzer die Funktion der App überwachen kann. Auch wenn das User Interface während der Abfragen und Rechnungen nicht

3 Datenbank

Für die persistente Speicherung der Daten wird durch eine MySQL Datenbank ermöglicht. Vor jeder Berechnung der Graphen oder eines neuronalen Netztes werden die Daten auf den neusten Stand gebracht. Hierzu werden die fehlenden Daten über die Coinbase Pro API heruntergeladen und danach sortiert in die Datenbank

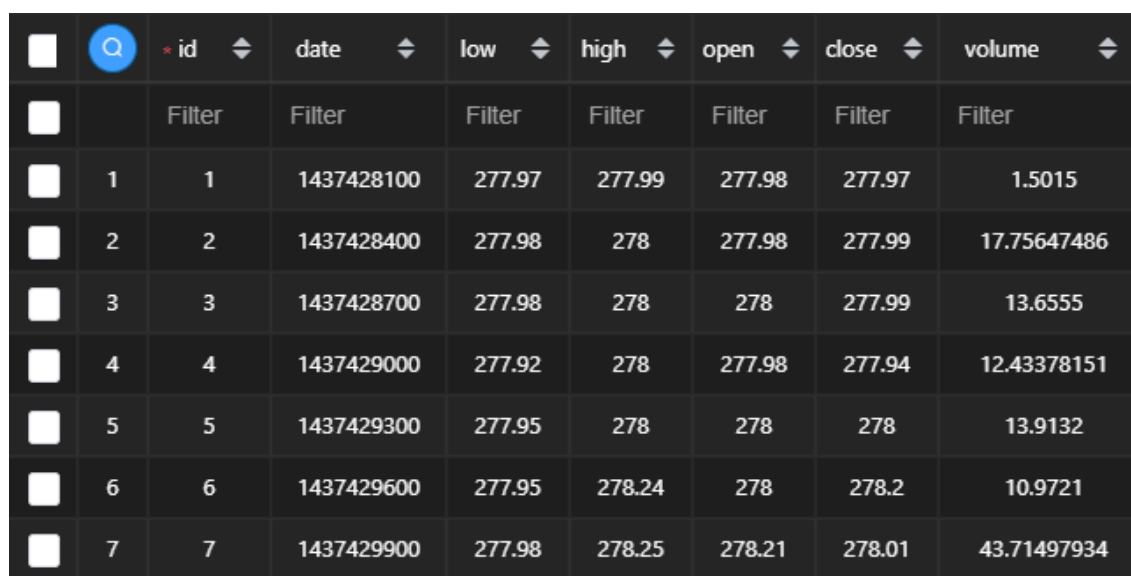


	id	coin	version	active
	Filter	Filter	Filter	Filter
	1	btc	1612407000	yes
	2	eth	1612407000	yes
	3	ltc	1612407000	yes

Abbildung 3: Versionentabelle

eingepflegt. Um die unterschiedlichen Versionen der Daten gut miteinander vergleichen zu können wird in der Tabelle „versions“ immer die Zeit des letzten gesammelten Datensatzes abgelegt. *Abbildung 3* zeigt diese Tabelle. Die Spalte „active“ gibt an, ob eine Kryptowährung schon von Programm verwendet werden kann, oder ob sie sich noch im Aufbau befindet. Für jede Kryptowährung werden die typischen Werte „low“, „high“, „open“, „close“ und „volume“ erfasst und wie in *Abbildung 4* in eine separate Tabelle abgespeichert. Von dort können die Daten von allen Methoden abgerufen werden und mit Hilfe der Klasse „database_interaction“ nach Bedarf transformiert werden.

Bei der Entwicklung wurde darauf geachtet eine portable Version von MySQL zu verwenden, so dass der Endbenutzer keine aufwendige und fehleranfällige Installation auf sich nehmen muss. Alle bedeutenden Konfigurationsdateien und Variablen werden bei der Initialisierung einer Datenbankverbindung automatisch überprüft und falls nötig angepasst. Zusätzlich ist die Datenbank gegen mehrfaches starten abgesichert und garantiert somit die bestmögliche Performance.



	id	date	low	high	open	close	volume
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	1	1437428100	277.97	277.99	277.98	277.97	1.5015
	2	1437428400	277.98	278	277.98	277.99	17.75647486
	3	1437428700	277.98	278	278	277.99	13.6555
	4	1437429000	277.92	278	277.98	277.94	12.43378151
	5	1437429300	277.95	278	278	278	13.9132
	6	1437429600	277.95	278.24	278	278.2	10.9721
	7	1437429900	277.98	278.25	278.21	278.01	43.71497934

Abbildung 4: Datenbanktabelle für eine Kryptowährung

4 LSTM-Netze

Die Vorhersage von in der Zukunft liegenden Werten einer Zeitreihe ist keine einfache Angelegenheit. Mithilfe von rekurrenten neuronalen Netzen kann diese Aufgabe jedoch trotzdem zuverlässig erledigt werden. In diesem Fall finden, zur Vorhersage von in Zukunft liegenden Werten von Kryptowährungen, LSTM-Netze Anwendung.

4.1 Vorverarbeitung

Die Vorverarbeitung der vorhandenen Daten ist ein wichtiger Schritt, um sinnvolle Ergebnisse erzielen zu können. Dieser Prozess wird in der Klasse „Preprocessing“ abgebildet. Zunächst müssen die Werte, der einzelnen Kryptowährungen zu einem großen Datensatz zusammengefügt werden. Der Zeitwert wird zum Index des Datensatzes, um die einzelnen Datenreihen ordnen zu können. In *Abbildung 5* ist ein beispielhaftes Ergebnis dieses Prozesses zu sehen.

time	btc_low	btc_high	btc_open	btc_close	btc_volume	eth_low	eth_high	eth_open	eth_close	eth_volume
1471407600	577.74	578	577.91	577.74	9.38775	11.05	11.06	11.06	11.06	170.13819
1471407900	577.66	578	577.73	577.82	8.46892992	11.05	11.05	11.05	11.05	20.07652
1471408200	577.89	578	577.89	578	4.08380846	11.05	11.09	11.05	11.09	301.86590506
1471408500	577.99	578	577.99	577.99	2.74032	11.09	11.09	11.09	11.09	1.00941
1471408800	577.5	578	578	577.99	7.39036067	11.09	11.11	11.09	11.11	3.70079349

Abbildung 5: Rohdaten aus der Datenbank

Anschließend werden die vorherzusagenden Daten erstellt. Hierfür wird die gewünschte Zeile um die Anzahl der in Zukunft liegenden Schritte nach oben verschoben und als weitere Spalte in den Datensatz eingefügt (siehe *Abbildung 6*).

Im folgen Schritt werden die Daten in die Trainings-, Validierungs- und Testdaten aufgeteilt. Die Testdaten machen hier

ltc_close	ltc_volume	future
3.7	5	580.88
3.7	5	580.99
3.7	5	580.91
3.7	5	581.08
3.7	5	580.86

Abbildung 6: Zukunftsdaten

20% des gesamten Datensatzes aus. Die übrig gebliebenen 80% werden in 95% Testdaten und 5% Validationsdaten aufgeteilt. Jede dieser Gruppen muss nun skaliert werden. Hierfür wurde der Zielwertebereich von null bis eins gewählt. Mithilfe eines MinMaxScalers kann diese Anpassung durchgeführt werden. *Abbildung 7* zeigt die normalisierten Daten.

time	btc_low	btc_high	btc_open	btc_close	btc_volume	eth_low	eth_high	eth_open	eth_close	eth_volume
1471407600	0.029042	0.000455	0.029049	0.000464	0.002596	0.007719	0.003579	0.003621	0.003628	0.001328
1471407900	0.029038	0.000455	0.029040	0.000468	0.002342	0.007719	0.003572	0.003614	0.003621	0.000157
1471408200	0.029050	0.000455	0.029048	0.000478	0.001129	0.007719	0.003600	0.003614	0.003649	0.002356
1471408500	0.029055	0.000455	0.029053	0.000477	0.000758	0.007748	0.003600	0.003642	0.003649	0.000008
1471408800	0.029030	0.000455	0.029054	0.000477	0.002044	0.007748	0.003614	0.003642	0.003663	0.000029

Abbildung 7: Normalisierte Daten

Ziel der Vorverarbeitung ist es eine große Liste mit Sequenzen der Daten und deren Zukunftswert zu erhalten, mit der das neuronale Netz gefüttert wird. Die Sequenzierung erfolgt durch eine deque, die nur eine festgelegte Anzahl an Elementen zulässt. Immer wenn ein zusätzlicher Wert hinzugefügt wird und die deque bereits voll ist, wird der älteste Wert entfernt. In diesem Fall wurde eine Sequenzlänge von 60 Zeitpunkten zum Training des neuronalen Netzes gewählt. Der entsprechende Zukunftswert kann einfach der im ersten Schritt eingefügten Spalte entnommen werden. Den Abschluss der Vorverarbeitung bildet die Ausgleichung. Um einem neuronalen Netz keine Trendfunktionen anzutrainieren sollten genauso viele Sequenzen, in denen der Zukunftswert größer ist, vorhanden sein wie Sequenzen, in denen er niedriger ist. Nach diesem Schritt ist die Vorverarbeitung der Daten abgeschlossen und das neue Netz kann trainiert werden.

4.2 Model

Wie bereits erwähnt werden zur Vorhersage der Kryptowährungen LSTM-Netze verwendet. Ziel ist es für die Tagesansicht einen Wert, der sechs Stunden in der Zukunft liegt und für alle anderen Ansichten einen Wert, der einen Tag in der Zukunft liegt vorherzusagen. Es fallen also pro Kryptowährung zwei neuronale Netze an, da diese darauf ausgelegt sind nur einen Wert vorherzusagen. *Abbildung 8* kann der Aufbau der neuronalen Netze entnommen werden. Auf jede der beiden LSTM-Schichten fol-

Model: "sequential"		
Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 60, 128)	73728
dropout (Dropout)	(None, 60, 128)	0
batch_normalization (Batch Normalization)	(None, 60, 128)	512
lstm_1 (LSTM)	(None, 128)	131584
dropout_1 (Dropout)	(None, 128)	0
batch_normalization_1 (Batch Normalization)	(None, 128)	512
dense (Dense)	(None, 32)	4128
dropout_2 (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 1)	33
Total params: 210,497		
Trainable params: 209,985		
Non-trainable params: 512		

Abbildung 8: Zusammenfassung eines Netzes

gen jeweils ein Dropout und eine BatchNormalization, um die Daten für die nächste Schicht zu verbessern. Im Anschluss folgt noch eine Dense-Schicht, die ihre Werte an die Ausgabeschicht weiterleitet. Als Optimierer wurde der Adam, mit einer Lernrate von 0.01 und einem Decay von $1e-6$, gewählt. Der „MeanSquaredError“ stellt die Verlustfunktion dar. Als zusätzliche Metrik wurde außerdem noch der „MeanAbsoluteError“ erhoben. Um eine gute Visualisierung zu gewährleisten, werden beim Training die Daten des Netzes an ein Tensorboard weitergeleitet. Nach jeder trainierten Epoche wird das aktuelle Netz zwischengespeichert, um später, im Falle eines Problems, auf den Checkpoint zurückspringen zu können. Aus diesen Checkpoints wird am Ende des Trainings das beste ausgewählt und an einem anderen Ort gespeichert, wo es zum Vorhersagen aufgerufen werden kann.

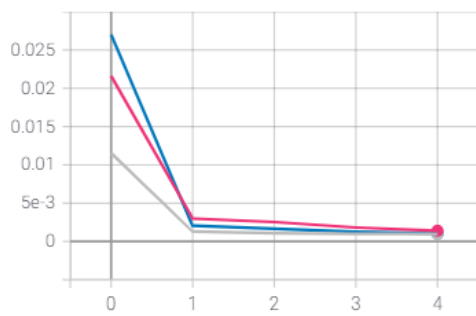


Abbildung 9: Trainingsdaten_6h_mse

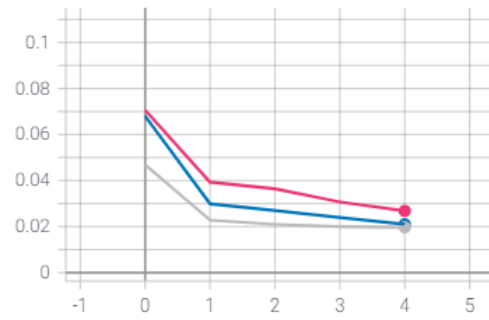


Abbildung 10: Trainingsdaten_6h_mae

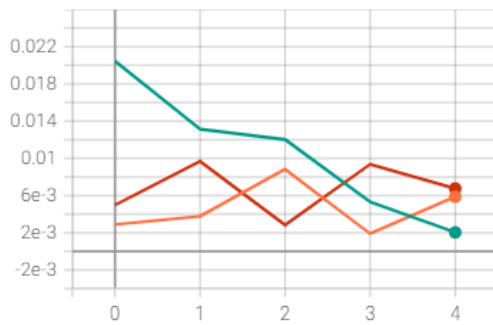


Abbildung 11: Validationsdaten_6h_mse



Abbildung 12: Validationsdaten_6h_mae

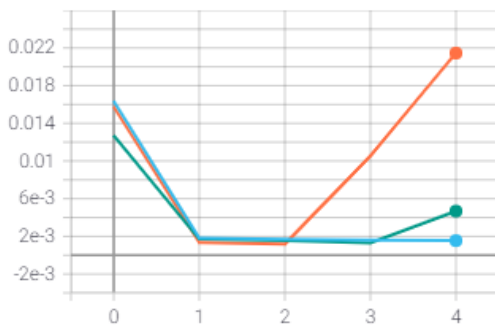


Abbildung 13: Trainingsdaten_1d_mse

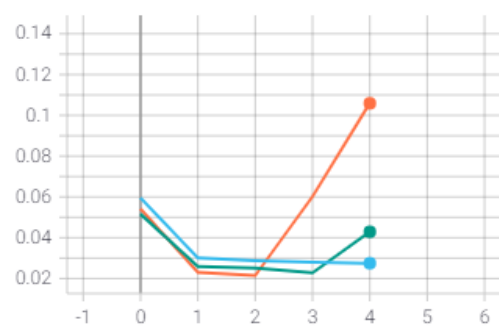


Abbildung 14: Trainingsdaten_1d_mae



Abbildung 15: Validationsdaten_1d_mse

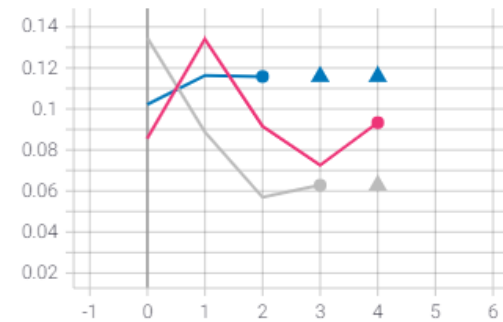


Abbildung 16: Validationsdaten_1d_mae

X_Achse = Epochen

1. Zeile: rot = btc, grau = eth, blau = ltc; 2. Zeile: grün = btc, orange = eth, braun = ltc;

3. Zeile: blau = btc, grün = eth, orange = ltc; 4. Zeile: rosa = btc, grau = eth, blau = ltc

4.3 Training

Für das Trainieren der LSTM-Netze sind die Werte „batch_size“ und „epochs“ besonders relevant. Sämtliche Netze wurden mit einer batch_size von 64 und mit 5 Epochen trainiert. In den *Abbildung 9* - *Abbildung 15*: Validationsdaten_1d_mse

Abbildung 16: Validationsdaten_1d_mae

kann das Verhalten der Loss Funktion und des MeanAbsoluteErrors über die Epochen hinweg nachvollzogen werden. *Abbildung 18* zeigt die Ergebnisse der Evaluierung der Netze.

```
#trains the model
history = model.fit(
    train_x, train_y,
    batch_size=self.batch_size,
    epochs=self.epochs,
    validation_data=(validate_x, validate_y),
    callbacks=[tensorboard, checkpoint],
    verbose=1,
)
```

Abbildung 17: Model trainieren Code

Ergebnisse Test Daten:						
	btc_72	btc_288	eth_72	eth_288	ltc_72	ltc_288
loss	0.00041930758743546903	0.0020185387693345547	0.0012018809793516994	nan	0.0017264005728065968	nan
mse	0.015028763562440872	0.03212708234786987	0.021644040942192078	nan	0.022016357630491257	nan

Abbildung 18: Ergebnisse der Testdatenauswertung

4.4 Automatische Hinzufügung von weiteren Währungen

Falls eine Kryptowährung nicht im Programm vorhanden ist kann diese vollautomatisch hinzugefügt werden. Hier muss zuerst die neue Währung in der Datei „coins.conf“ hinterlegt werden. Anschließend muss aus dem Modul „activate_coins“ die Methode „activateCoins“ aufgerufen werden. Zuerst werden dann alle benötigten Zeitdaten aus der Coinbase API in die Datenbank geladen. Dies dauert pro Währung ca. 15 Minuten. Danach werden neue neuronale Netze trainiert. Dies passiert für jede Währung, da auch die bereits gespeicherten Währungen von den neuen Zeitdaten profitieren sollen. Pro neuronales Netz kann hier eine Zeit von bis zu zwei Stunden anfallen. Während dem Erstellungsvorgang sind die neuen Währungen in der Datenbank als nicht aktiv gekennzeichnet. Dies verhindert, dass nicht vollständige Daten den Programmablauf behindern. Die Anwendung kann also während dem Hinzufügen normal verwendet werden.

5 Evaluation der verwendeten Netze

Zur Evaluation der verwendeten Netze stehen unterschiedliche Methoden zur Verfügung. Es können die vorhergesagten Werte quantitativ und qualitativ mit der realen Wertentwicklung verglichen werden. Zudem kann in einer Simulation die Wertentwicklung eines Startkapitals unter Berücksichtigung eines bestimmten Anlageverhaltens veranschaulicht werden.

5.1 Abweichung zwischen vorhergesagten und realen Werten

Die Abweichung der Vorhersage von der tatsächlichen Wertentwicklung gibt Auskunft über die Qualität der verwendeten Netze.

Die Qualität kann sowohl quantitativ über Mittelwert und Standardabweichung als auch qualitativ über das Verhältnis zwischen Vorhersagen in die richtige und falsche Wertentwicklungsrichtung bewertet werden.

Obwohl die verwendeten Netze denselben Aufbau haben zeigen sich Unterschiede in der Qualität ihrer Vorhersagen. Dies lässt sich durch das individuelle Training der einzelnen Netze erklären.

Tabelle 1: Übersicht über die Bewertungsfaktoren

Währung	BTC		ETH		LTC	
Vorhersagezeitraum	6h	1d	6h	1d	6h	1d
Mittelwert der Abweichung [%]	3,31	5,99	8,56	9,96	15,67	10,93
Standardabweichung [%]	2,89	5,12	3,08	4,6	5,97	7,16
Richtungsvorhersage richtig / falsch	21/19	2/8	14/25	5/5	25/15	7/3

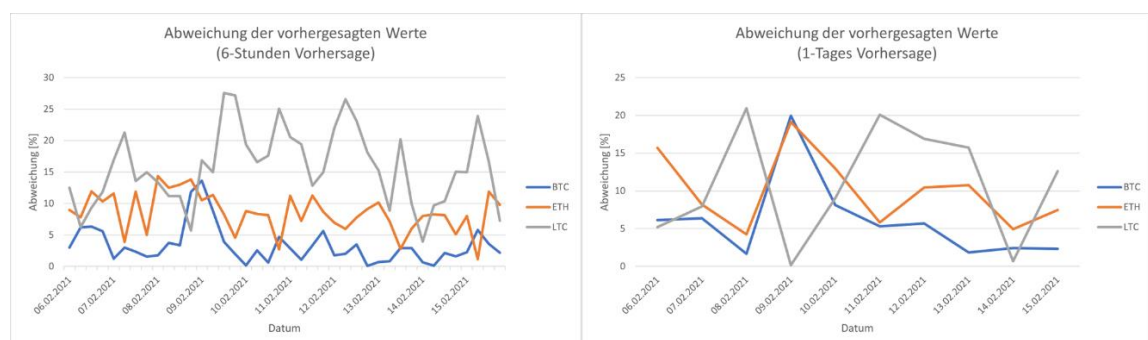


Abbildung 19: Abweichungen der Vorhersagen von der tatsächlichen Wertentwicklung

5.2 Anlagesimulation mit einem Startkapital von 4000\$

Die Simulation beginnt mit einem Startkapital von 4000\$ die zu Beginn jeweils zu 2000\$ in der jeweiligen Kryptowährung und zu 2000\$ in einem Guthaben vorliegen. Hiernach wird in einem

Abstand von 6h / 1d die nächste Vorhersage abgefragt. Liegt der vorhergesagte Wert über dem aktuellen Wert, wird für 50\$ von der Kryptowährung zugekauft. Ist die Wertedifferenz entgegengesetzt wird für 50\$ von der Kryptowährung verkauft. Die Werte der einzelnen Datenpunkte sind jeweils die Summe aus dem aktuellen Guthaben und dem Verkaufswert des Guthabens der Kryptowährung.

Allgemein ist durch die Kombination aus den verwendeten Netzen und der genannten Strategie eine Steigerung des Gesamtkapitals erkennbar. Die Kapitalentwicklungen sind in *Abbildung 20* dargestellt. Die Anlagesimulation eignet sich in diesem Fall nur bedingt zur Bewertung der Netze, da während des Zeitraums eine permanente Werteerhöhung der Währungen vorhanden war.

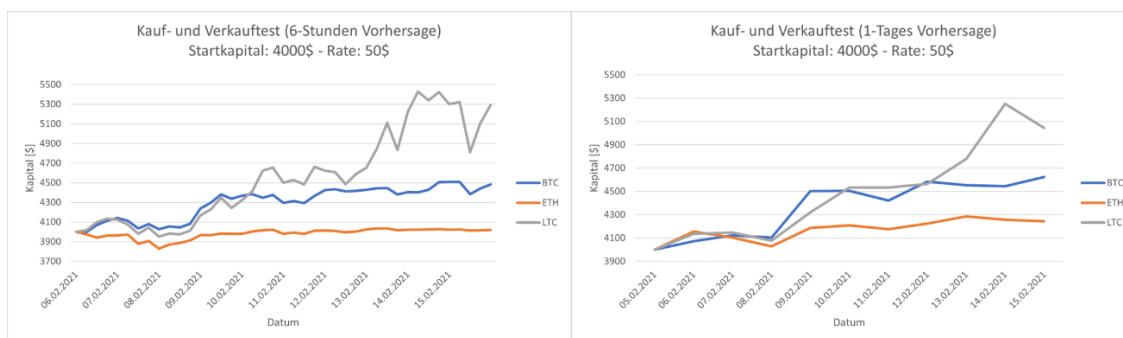


Abbildung 20: Simulation des Anlegens mit einem Startkapital von 4000\$

6 Ausblick

Die erstellte Anwendung bietet grundsätzlich viel Erweiterungspotenzial, das bei einer Umsetzung auch maßgeblich zu einer besseren Benutzerfreundlichkeit führen kann. Selbstverständlich sollte in naher Zukunft eine Usability Evaluation stattfinden, um die grafische Oberfläche angemessen an den Nutzer anzupassen. Momentan ist die Applikation davon gezeichnet, dass beim neu Laden lange Wartezeiten vorhanden sind. Dies könnte durch eine Umsetzung als Server-Client Anwendung beseitigt werden. Die Hauptursachen für die langen Ladezeiten sind sowohl die Datenzusammenstellung für die verschiedenen Ansichten als auch die Vorhersagung der Zukunftswerte. Ein Server mit einer Datenbank, der diese benötigten Werte ständig auf dem Laufenden hält und diese bei Bedarf direkt an den Client sendet würde die Ladezeiten verkürzen. Es müssten nicht immer alle Werte neu berechnet werden.

In der aktuellen Versionen sind nur Vorhersagen für sechs Stunden (in der Tagesansicht) und für einen Tag (in allen anderen Ansichten) möglich. Durch die Erstellung von weiteren neuronalen Netzen könnten auch noch individuelle Vorhersagen für die Monats- und Jahresansicht getroffen werden. Dadurch wäre außerdem ein Vergleich der neuronalen Netze mit anderen Methoden, die Finanzdaten weit in die Zukunft vorhersagen, möglich.

Ein großes Problem bei der Entwicklung dieser Applikation stellte die eingeschränkte Rechnerkapazität dar. Bereits bei der Berechnung einer Monatsvorhersage konnten RAM und CPU nicht mehr mithalten. Für ein genaueres und besseres Training der neuronalen Netze sollte also in Zukunft auf ein leistungsstärkeres System zurückgegriffen werden. Es ist möglich, dass dadurch deutlich bessere Ergebnisse erzielt werden können.

Ebenfalls findet sich weiteres Entwicklungspotential in der Qualität der Neuronalen Netze. Hier gilt es die Netzstruktur weiter zu optimieren, um genauere Vorhersagen treffen zu können.

Allgemein ist die Vorhersage der Werte von Kryptowährungen schwer, da diese stark fluktuieren.

Quellen

[1] <https://kivy.org/#home> (04.02.2021)

[2] <https://kivy.org/doc/stable/guide/lang.html> (04.02.2021)

[3] <https://www.coinbase.com/price/bitcoin> (04.02.2021)

Python Icon: https://external-content.duckduckgo.com/iu/?u=http%3A%2F%2Flogos-download.com%2Fwp-content%2Fuploads%2F2016%2F10%2FPython_logo_icon.png&f=1&nofb=1

Tensorflow Icon: https://external-cotent.duckduckgo.com/iu/?u=https%3A%2F%2Ftse1.mm.bing.net%2Fth%3Fid%3DOIP.uOz__WYQlcOzCPuH5xas8wHaH7%26pid%3DApi&f=1

Coinbase Icon: https://external-content.duckduckgo.com/iu/?u=http%3A%2F%2Fih5.ggpht.com%2FyJouMLD-_TznAWx5ICIEOtEKe-Ovm_OX4gJ7BVBV6Ld_2Zz983jnjaRKs5jr4F86GQ%3Dw300&f=1&nofb=1

MySQL Icon: https://external-content.duckduckgo.com/iu/?u=http%3A%2F%2Fsqlbackupand-ftp.com%2Fblog%2Fwp-content%2Fuploads%2F2015%2F01%2Fmysql-logo_2800x2800_pixels1-1024x1024.png&f=1&nofb=1