

BACHELORARBEIT

VORHERSAGE VON AKTIENKURSEN MITTELS TIEFEN LSTM-NETZWERKEN

Eingereicht an der NTB Interstaatliche Hochschule für
Technik Buchs

Autoren: Silvio Jäger
Joel Erzinger

Dozenten: Prof. Dr. Christoph Würsch
Prof. Dr. Klaus Frick

Abgabetermin: 09. August 2019

Code: https://github.com/silviojaeger/ba_machine_learning_2019

ABSTRACT

Vorhersagen von Zeitreihen mit LSTM Zellen feiern seit etwa 2016 bedeutende Erfolge. Diese Arbeit beschäftigt sich mit der These, Aktienkurse mittels tiefen neuronalen Netzwerken mit LSTM Zellen vorhersagen zu können. Dabei werden verschiedene Features und Netzwerkstrukturen getestet und die besten Ergebnisse vorgestellt. Die Arbeit behandelt sowohl das Beschriften und Vorverarbeiten der Daten als auch die Grid Searches zur Findung der optimalen Parametrisierung der neuronalen Netzwerke.

Die Vorhersage der Aktienkurse wird mittels Regressions- und Klassifizierungs-Modell implementiert und evaluiert. Nebst den reinen Kursdaten werden auch verschiedene Indikatoren berechnet und als Input für das LSTM Netzwerk definiert. Mit diesen verschiedenen Inputs und Konfigurationen des Netzwerks werden Grid Searches durchgeführt. Die erhaltenen Ergebnisse werden mittels Kreuzevaluation ausgewertet und präsentiert. Die vorhergehende Datenverarbeitung mittels Sperman-Korrelation bietet zudem eine detaillierte Übersicht der Zusammenhänge der globalen Finanzmärkte aus mathematischer Sicht. Neben der Aufarbeitung der Kernfrage bietet diese Arbeit modulare Anleitungen und Erfahrungsberichte als Inspiration für zukünftige Arbeiten. Die wichtigsten verwendeten Technologien werden im Anhang präsentiert.

Obwohl die Vorhersage diverser Aktienkurse nicht signifikant besser als der Zufall ausgefallen ist, wurden Modelle gefunden, die die Richtung des Kurses in über 70 Prozent der Fälle korrekt vorhersagen konnten. Dies zeigt auf, dass durchaus Potential besteht, LSTM Netzwerke als Indikatoren für Kaufentscheidungen im Finanzmarkt einzusetzen.

INHALT

1	Selbständigkeitserklärung	5
2	Glossar	6
3	Einleitung.....	7
3.1	Ziel	7
3.2	Vorgehen	7
3.3	Gliederung	7
3.4	Umfang der Arbeit	8
4	Beschaffung aktueller Kursdaten	9
4.1	Die Börse	9
4.2	Datenquellen.....	9
5	Korrelation	10
5.1	Correlation Map	10
6	Vorverarbeitung.....	12
6.1	Vorgehen	12
6.2	Datenqualität.....	12
6.3	Indikatoren.....	13
6.4	Slicing	14
7	Architektur	16
7.1	Entwicklungsumgebung.....	16
7.2	Gliederung	16
7.2.1	Programmstruktur	17
7.2.2	Grid Search / Config	17
7.2.3	Collector.....	19
7.2.4	Hilfsbibliotheken.....	20
7.2.5	Auswertungsfunktionen	20
8	LSTM Deep Learning	21
8.1	wiso LSTM.....	21
8.2	Überlegungen zur Strategie.....	22
8.3	Modelle / Strategien.....	22
8.3.1	Regression (Basismodell).....	22
8.3.2	Classification.....	29
8.3.3	Kombination - Custom Loss	32
9	Ergebnisse der Arbeit.....	34
9.1	Grid Search mit dem Custom Loss Modell	34
9.1.1	Korrelation und Datenauswahl	34
9.1.2	Grid Search Parametrisierung	34
9.1.3	Auswertung.....	35

9.1.4	Cross Validation.....	38
9.2	Grid Search mit dem Custom Loss Modell und unterschiedlicher Kursskalierungen	39
9.2.1	Parametrisierung	39
9.2.2	Auswertung.....	40
9.3	Grid Search mit dem Custom Loss – Neuer Ansatz.....	42
9.3.1	Korrelation - Datenauswahl	42
9.3.2	Parametrisierung	42
9.3.3	Resultate.....	44
9.4	Grid Search mit der Classification Methode	45
9.4.1	Grid Search Parametrisierung	45
9.4.2	Auswertung.....	46
9.4.3	Debugging mit Sinus	47
10	Fazit.....	50
10.1	Anwendungsbereich der Resultate dieser Arbeit	50
10.2	kommentar der Autoren	50
11	Mögliches weiteres Vorgehen	51
11.1	Weitere Indikatoren	51
11.2	Kurzfristige Vorhersage	51
11.3	Neuer Ansatz - Gaussian Process Regression (GPR).....	51
11.3.1	Realisierung.....	52
12	Tipps für zukünftige Arbeiten.....	53
12.1	Grösste Probleme.....	53
12.1.1	Datenquelle.....	53
12.1.2	Keras	53
12.1.3	Machine Learning allgemein.....	53
12.2	Empfohlenes Vorgehen bei der Entwicklung mit ML.....	53
12.2.1	Das Netzwerk lernt auswendig, der Validation Loss steigt	53
12.2.2	Der Validation Loss stagniert nach wenigen Epochen	54
12.2.3	Das Netzwerk braucht unerwartet lange für das Training	54
13	Anhang	55
14	Abbildungsverzeichniss	56
15	Literaturverzeichnis	58

1 SELBSTÄNDIGKEITSERKLÄRUNG

Hiermit wird bestätigt, dass die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate und gedankliche Übernahmen kenntlich gemacht wurden.



St. Gallen, 09.08.2019

Joel Erzinger

Silvio Jäger

2 GLOSSAR

API	Kommunikationsschnittstelle für eine Software
Block/Sequenz	Zeitspanne des Inputs für die Vorhersage eines Wertes
Broker	Anbieter von Handelsplätzen, in welchen gehandelt wird
Dataframe	Datentabelle in digitaler Form als Python Pandas Objekt
Dropout	Wert, der bestimmt, wie viele Zellen im neuronalen Netz ausgeschaltet werden
Epoche	Ein einziger kompletter Trainingsdurchlauf über alle Testdaten
Features	Input Daten eines Modelles
Forex	Markt für Währungen
Grid Search	Kombination aller möglichen Parametrisierungen zur Findung der optimalen Einstellungen
Keras	Frontend Bibliothek für Tensorflow
Kreuzevaluation	Validierung eines Netzwerkes mit neuen Daten
Layer	Schicht von Neuralen Zellen im Netzwerk
Long Short Term Memory (LSTM)	Neuronale Zelle mit Kurz- und Langzeitgedächtnis
Loss	Fehler zwischen Wahrheit und Vorhersage eines Models
Moving-Average	Mathematische Glättung eines Kurses/Signals
Node	Zelle in einem neuronalen Netzwerk
Optimizer	Funktion zur Anpassung der Gewichte in einem neuronalen Netzwerk
Overfitting	Überanpassung eines Models an den Trainings Datensatz
Random Walk Theorie	Theorie, die besagt, dass der Verlauf eines Aktienkurses völlig zufällig ist und nicht anhand vergangener Kursdaten vorhergesagt werden kann.
Regression	lineares Modell
Slack	Messenger (Kommunikationsprogramm), unterstützt Textnachrichten, Bilder und Gruppenchats
Tensorflow	Machine Learning Framework von Google
Tick-Bereich	Kleinste mögliche Auflösung der Daten

3 EINLEITUNG

3.1 ZIEL

Die "random walk theory" besagt, dass der Verlauf eines Aktienkurses völlig zufällig ist, und dieser nicht anhand vergangener Kursdaten vorhergesagt werden kann. Da es jedoch diverse Investoren gibt, die regelmäßig Gewinne an der Börse erzielen können, ist diese Theorie sehr umstritten. Eine Vorhersage, die nur knapp besser ist als der Zufall, kann bereits zu einer profitablen Anlagestrategie führen.

Die Mathematik bietet uns heute diverse Instrumente, um Signale zu analysieren und ihren weiteren Verlauf vorherzusagen. Die Rangkorrelation nach Spearman und das maschinelle Lernen mittels neuronaler Netzwerke bieten sich dafür an.

In dieser Arbeit wird versucht, anhand von korrelierenden Aktienkursen und maschinellem Lernen eine möglichst genaue Vorhersage für verschiedene Aktienkurse zu machen.

3.2 VORGEHEN

Zu Beginn wurden historische Aktien-, Rohstoff- und Währungskurse gesammelt und deren Korrelation mittels verschiedener Korrelationstechniken berechnet.

Anschliessend wurden anhand der Ergebnisse der Korrelation, Portfolios von drei bis zwanzig Kursen zusammengestellt. Mit einem selbst programmierten Tool wurden verschiedene neuronale Netzwerke anhand der Kurse im Portfolio trainiert und versucht, einen Kurs vorherzusagen.

Es wurde darauf geachtet, das Programm so modular wie möglich zu gestalten, um mit passenden Grid Searches verschiedene Konfigurationen zu testen. So ist es möglich, verschiedene Modelle und Eingaben zu testen, ohne Änderungen am Code vornehmen zu müssen.

3.3 GLIEDERUNG

Abschnitt	Inhalt
Komponenten der Arbeit Kapitel 3.4 bis 0	Beschreibt den Aufbau, sowie das Vorgehen
Ergebnisse der Forschung Kapitel 9	Durchgeführte Tests und deren Ergebnisse

Struktur dieser Dokumentation

3.4 UMFANG DER ARBEIT

Variation des Inputs

- Europäische und amerikanische Aktienkurse
- Forex sowie Rohstoffe
- Diverse technische Indikatoren
- Variation der Zeiträume

Modelle

- LSTM mit unterschiedlicher Anzahl Layern und Nodes
- Unterschiedlicher Dropout, auch zwischen den verschiedenen Layern
- Verschiedene Loss-Funktionen
- Klassifikation, Winkelberechnung, Absolutwert-Vorhersage
- Verschiedene Optimizer mit starkem Fokus auf den Adam-Optimizer

Evaluation

- Nach tiefstem Loss
- Nach bester Trefferquote
- Kreuzevaluation

4 BESCHAFFUNG AKTUELLER KURSDATEN

4.1 DIE BÖRSE

An der Börse kann nicht nur ausschliesslich mit Aktien von Unternehmen gehandelt werden, sondern es besteht auch die Möglichkeit, Wertpapiere für Rohstoffe und Währungen zu kaufen. Viele Broker erlauben nicht nur den Kauf (Buy), sondern auch den Verkauf (Sell) von Wertpapieren. Dies bietet dem Investor die Option, auch auf fallende Kurse zu setzen.

Der Markt für Währungen nennt sich Forex. Hier werden Währungen in sogenannten Pairs gehandelt. Das Verhältnis der beiden Währungen zueinander bestimmt den Wechselkurs. Beispiele für Forex Kurse sind *EUR/USD* und *EUR/CHF*. Diese Arbeit beschäftigt sich auch mit diesem Markt.

4.2 DATENQUELLEN

Bereits das Beschaffen qualitativ hochwertiger Kursdaten hat sich als aufwendig herausgestellt. Es gibt diverse kostengünstige APIs welche historische Kursdaten bereitstellen. Leider stellt keine der getesteten API's Kursdaten im Tick-Bereich zur Verfügung. Fündig wird man dennoch bei der Firma Dukascopy SA (Vergleich der Anbieter 4.2). Über ein eigenes Programm der Dukascopy Bank namens JForex können diverse Kursdaten, darunter Aktien-, Währungs- und Rohstoffkurse, heruntergeladen werden. Je nach Kurs beinhalten die Daten 3-10 Jahre im Intervall von jeweils einer Minute. Für jede Minute ist der Open-, Close-, High- und Low-Preis sowie das Volumen bekannt.

Eine Übersicht der recherchierten Quellen bietet folgende Tabelle:

Anbieter	Quandl	Alphavantage	WorldTrading Data	Dukascopy
Historische Tageskurse	JA	JA	JA	JA
Historische Minutenkurse	NEIN	NEIN	NEIN	JA
Echtzeitkurse	NEIN	JA	JA	JA
Swiss Stock Daten verfügbar	NEIN	NEIN	JA	NEIN
Frequenz	-	echtzeit	15 min	echtzeit
API vorhanden	JA	JA	JA	JForex
Preis	gratis	gratis	~ 30.- CHF/Monat	gratis

Vergleichstabelle unterschiedlicher Datenquellen

In dieser Arbeit werden die Daten von Dukascopy und World Trading Data verwendet.

5 KORRELATION

Für die Vorhersage der Aktienkurse muss vor dem Aufbau eines Netzwerks zuerst die Auswahl der Daten getroffen werden. Einerseits währen alle verfügbaren Kurse zu viele Features, um in einem Netzwerk als Input zu dienen (gemäss der Hardware, welche dem Projekt zur Verfügung gestellt wurde). Andererseits wird so vermieden, dem Netzwerk unnötige Informationen bereitzustellen. So kann der Trainingsvorgang optimiert werden. (Géron, Hands-On Machine Learning with Scikit-Learn and TensorFlow, 2017)

Zum Vergleich zweier Zeitreihen gibt es verschiedene Verfahren. Die Library Pandas hat bereits diverse Funktionen eingebaut: pearson, kendall, spearman (pandas.pydata.org, 2019). Interessant ist auch zu erfahren, welche Auswirkungen eine Verschiebung der Kurse auf die Korrelation in der Zeitachse hat. Im Rahmen der Projektarbeit wurde daher ein Programm geschrieben, welches ermöglicht, beliebig viele Kurse inklusive Verschiebung der Zeitachse miteinander zu korrelieren. Da sich die Kurse immer unterschiedlich stark verändern, empfiehlt es sich eine Spearman Korrelation einzusetzen.

Als Resultat erhält man von jedem Kurs eine Tabelle mit allen anderen Kursen, deren Korrelations-Index und Verschiebung.

Vergleichskurs	Index	Verschiebung in [10min]
CAD/CHF	0.8160	0
GBP/CHF	0.7852	0
Vale S.A.	0.7289	91
...

Kreuzkorrelation mit EUR/CHF, spearman korrelation, max. 24h verschoben, 10min Skalierung

Zu erkennen sind hohe Korrelation (>0.5) mit Kursen, welche die gleiche Währung enthalten. Diese Informationen geben aufgrund ihrer optimalen Verschiebung von Null Ticks ohnehin nur wenige Informationen über den Zielkurs EUR/CHF bekannt.

Anhand der Tabellen der Kreuzkorrelationen konnten verschiedene Kurspakete für das neuronale Netz geschnürt werden.

5.1 CORRELATION MAP

Die gesamte Korrelation lässt sich auf einer 2D-Karte visualisieren, indem man die Indizes als anziehende Kraft zwischen zwei Kursen verwendet. Aus Leistungsgründen und zur besseren Übersicht ist die Anzahl Verbindungen pro Kurs beschränkt.

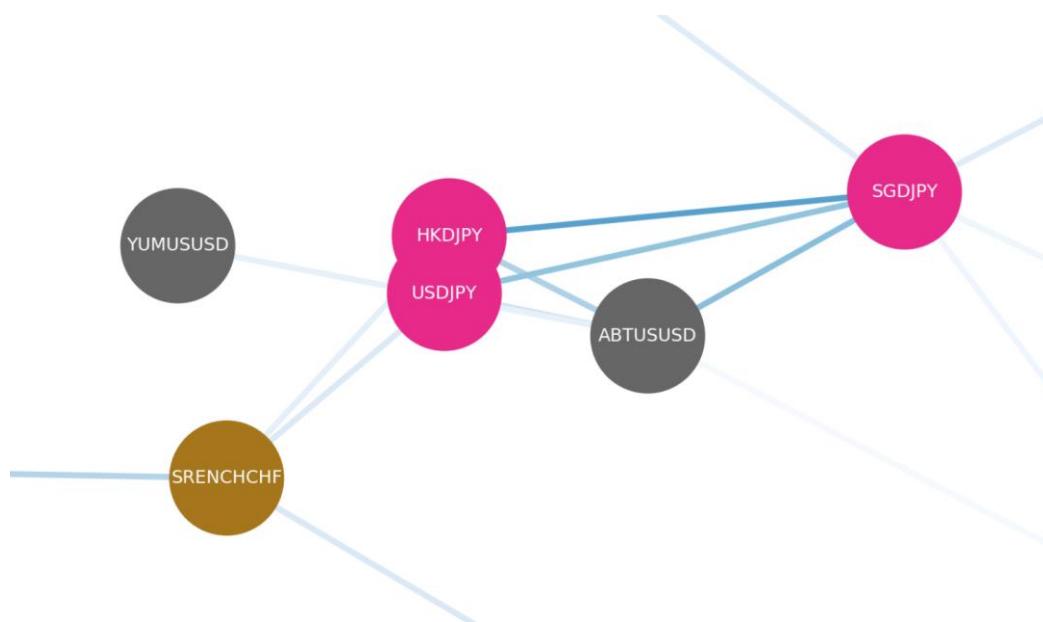


Abbildung 1: Auszug aus der Korrelationskarte aller Kurse (unvollständig)

Je dunkler die blaue Linie, desto grosser ist der Korrelationsindex nach Spearman und desto näher liegen die Kurse beieinander. Es handelt sich um ein Spring Layout, in welchem der Korrelationsindex gleichzeitig die Anziehungskraft darstellt. Die Verschiebung der Kurse durch die Kreuzkorrelation ist nicht visualisiert.

Hier ein weiteres Beispiel aus den Kursen des Schweizer Aktienmarktes:

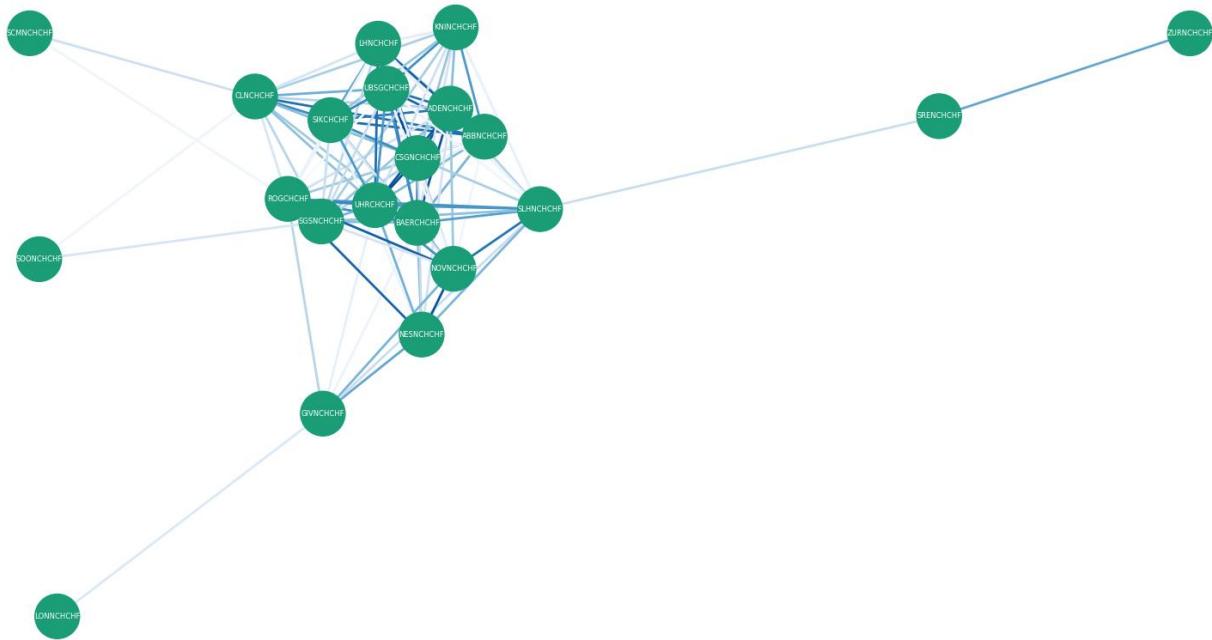


Abbildung 2: Korrelationskarte der Schweizer Kurse von Dukascopy (vollständig, rein quantitativ)

Die Korrelationskarte zeigt wie eng die Schweizer Unternehmen einander folgen. Es entstehen keine Cluster-Strukturen, welche die Auswahl erleichtern. Aus diesem Grund hat man sich bei der Auswahl der Kurse auf die Korrelationstabellen fokussiert und die Korrelationskarte nicht weiter verwendet.

6 VORVERARBEITUNG

6.1 VORGEHEN

Die Vorverarbeitungsschritte wurden während der Arbeit stetig erweitert und überarbeitet. Gestartet wurde aufgrund der verfügbaren Datenmenge mit Forex Daten (siehe Kapitel 6.2).

Der zweite grosse Schritt sind technische Indikatoren, die dem Netzwerk helfen sollen, den Kursverlauf besser zu kategorisieren. Auch diese wurden im Laufe der Arbeit stetig erweitert.

Zuletzt kann der fertige Datensatz mittels Slicing Funktionen in ein, für Keras lesbares Format, gebracht werden. Je nach Art der Prognose des Kurses wird dabei anders vorgegangen.

6.2 DATENQUALITÄT

Das grösste Problem ist grundsätzlich die zeitliche Verschiebung der Öffnungszeiten der verschiedenen Börsen weltweit. So lassen sich Kurse unterschiedlicher Börsen nur mit zeitlicher Verschiebung vergleichen.

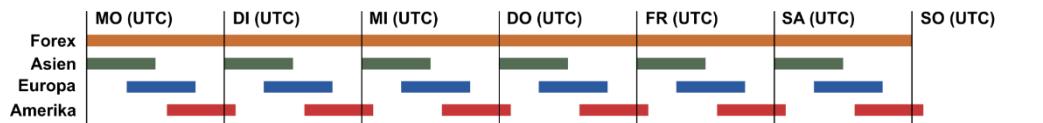


Abbildung 3: Handelszeiten (quantitative Grafik)

Die Schweizer Börse hat beispielsweise folgende Öffnungszeiten.

Central Limit Order Book (CLOB)		
Securities	Opening	Close
Bonds - CHF Swiss Confederation and Swiss Pfandbriefe	08:30	17:00
Shares and Investment Funds	09:00	17:30 *
Rights and Options	09:15	17:15
Bonds - CHF (except for Bonds - CHF Swiss Confederation and Swiss Pfandbriefe)	09:30	17:00

* In the trading segments Shares and Investment Funds an auction is conducted from 17.20 to 17.30.

Quote Driven Market (QDM)		
Securities	Opening	Close
Bonds - Non CHF	08:30	17:00
ETF on Bonds of the Swiss Confederation	09:00	17:00
ETF, ETSF and ETP (except for ETF on Bonds of the Swiss Confederation)	09:00	17:30 **
Sponsored Funds	09:15	17:30 **
Structured Products	09:15	17:15

** In the trading segments ETFs, ETPs and Sponsored Funds an auction is conducted from 17.30 to 17.35

Abbildung 4: Öffnungszeiten der Schweizer Börse (www.six-group.com, 2019)

Bei ca. 8h Öffnungszeiten hat man also Glück, wenn sich die Handelszeiten überhaupt überschneiden. Handelt man mit europäischen Kursen, können nur begrenzte Handelszeiten der anderen Kurse verwendet werden.

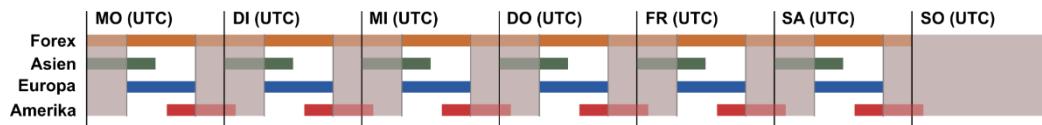


Abbildung 5: Gültiger Handelszeitraum europäischer Kurse (weisse Flächen sind gültig, rein quantitative Grafik)

Das Mischen interkontinentaler Portfolios empfiehlt sich daher nur bedingt für eine Vorhersage.

6.3 INDIKATOREN

Indikatoren können direkt im Grid Search mittels Namensergänzung eingesetzt werden. In einer Test-Unit wird dies wie folgt gehandhabt:

[NameSymbol]-[Column]_[Indicator][Parameter]

So berechnet die Angabe "SIEDEEUR-Open_MA12" den Moving Average über 12 Samples der jeweiligen Eröffnungswerte (Open) des Aktienkurses von Siemens.

Folgende Indikatoren sind implementiert und können in den Tests verwendet werden:

Indikator	Beschreibung
MA	Moving Average
SMA	Shifted Moving Average (Eigenkreation) Berechnet den Moving Average und verschiebt diesen um die Hälfte der Samples in die negative Zeitachse. So erhält man den Moving Average ohne Zeitverzögerung zu den Originaldaten. Dies verstösst eigentlich gegen die Regeln, da mit dieser Operation mit der Verschiebung in die Zukunft geschaut wird. Dies lässt sich jedoch trotzdem umsetzen, wenn die Prediction-Length, also die Vorhersageweite grösser gesetzt wird als der Shift in dieser Operation.
EMA	Exponential Moving Average.
BB	Bollinger Bands

Verwendete Technische Kursindikatoren. Weitere Informationen über technische Indikatoren bietet folgende Quelle: (www.tradingtechnologies.com, 2019)

6.4 SLICING

Die Daten liegen nach der Vorverarbeitung in einem einzigen Dataframe. Keras/Tensorflow erwartet jedoch eine Datentabelle (Numpy Array) mit passender Lösungstabelle zum Input. Folgendes Beispiel zeigt die Zerlegung eines Inputs mit den entsprechenden Eigenschaften:

- **2 Features**
- Eine Blocksize von **5**
- Die Prediction-Length ist auf **3** festgelegt
- Eine Batchsize von **24**
- Mit einem Training von **3** oder mehr Epochen

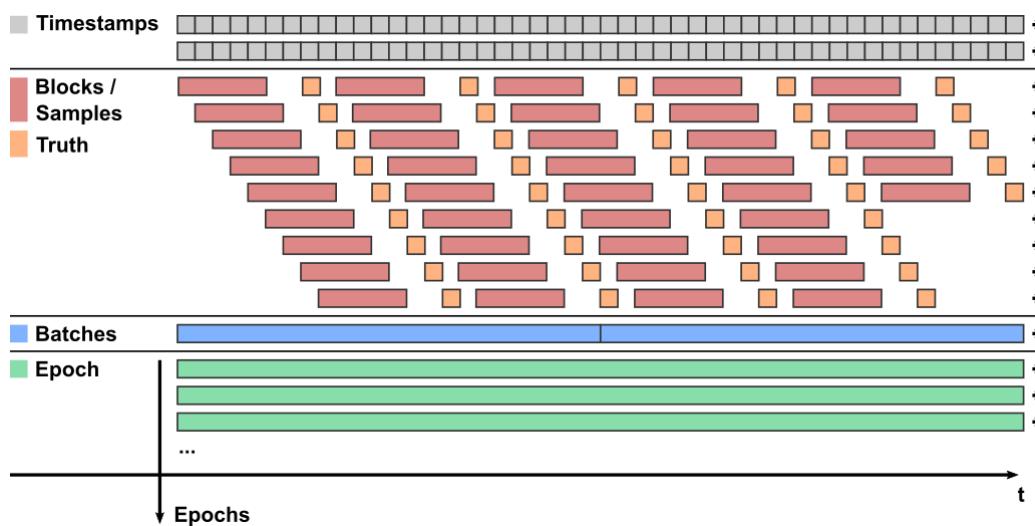


Abbildung 6: Slicing Beispiel, visualisiert die Verpackung der Daten (von oben nach unten gelesen). Die Datenmenge visualisiert die horizontale Achse (von links nach rechts gelesen).

Die Batches und Epochen werden von Keras anschliessend selber verwaltet. Das Datenframe wird wie folgt in Blocks aufgeteilt:

Dimension	Inhalt
1	Block
2	Samples / Sequence
3	Zeitstempel
4	Kurswerte / Indikatorwerte

Übersicht der Datenstruktur zu den Input-Daten des neuralen Netzwerkes. (X-Anteil)

Die Lösung der Input-Daten, die korrekte Vorhersage, wird mit folgender Datenstruktur abgeglichen. Diese dient dem Netzwerk zur Evaluation.

Dimension	Inhalt
1	Samples / Sequences
2	Zeitstempel
3	Truth und der vorherige Datenwert (wird benötigt für die Custom Loss Funktion) (siehe 8.3.3)

Übersicht der Datenstruktur zur Lösung der Input-Daten (Y-Anteil).

Für die Vorhersage mit der Methode Classification wird der Inhalt der Lösung leicht angepasst, bleibt in der Grundstruktur aber gleich:

Dimension	Inhalt
1	Samples / Sequence
2	Zeitstempel
3	Alle Klassen (siehe 8.3.2)

Übersicht der Datenstruktur zur Lösung der Input-Daten für die Classification. (Y-Anteil)

7 ARCHITEKTUR

7.1 ENTWICKLUNGSUMGEBUNG

Aus der Aufgabenstellung geht bereits hervor, dass Python als Programmiersprache verwendet wird. Um einen angenehmen Workflow zu erreichen, bietet sich das kostenlose Tool Visual Studio Code an. Für die Code-Verwaltung wird Git auf der Plattform Gitlab eingesetzt.

Um ein angenehmes Arbeiten im Team zu ermöglichen wird zusätzlich Slack verwendet. (siehe Anhang Slack)

Als Machine Learning Framework wurde Tensorflow mit einem Keras Frontend eingesetzt. Wir haben uns für Keras entschieden, da es den Aufbau der Neuronalen Netze gut abstrahiert.

Für das Training der Netze auf dem Cluster der Hochschule für Technik Rapperswil wurde zusätzlich Docker eingesetzt. (Siehe Anhang Docker)



7.2 GLIEDERUNG

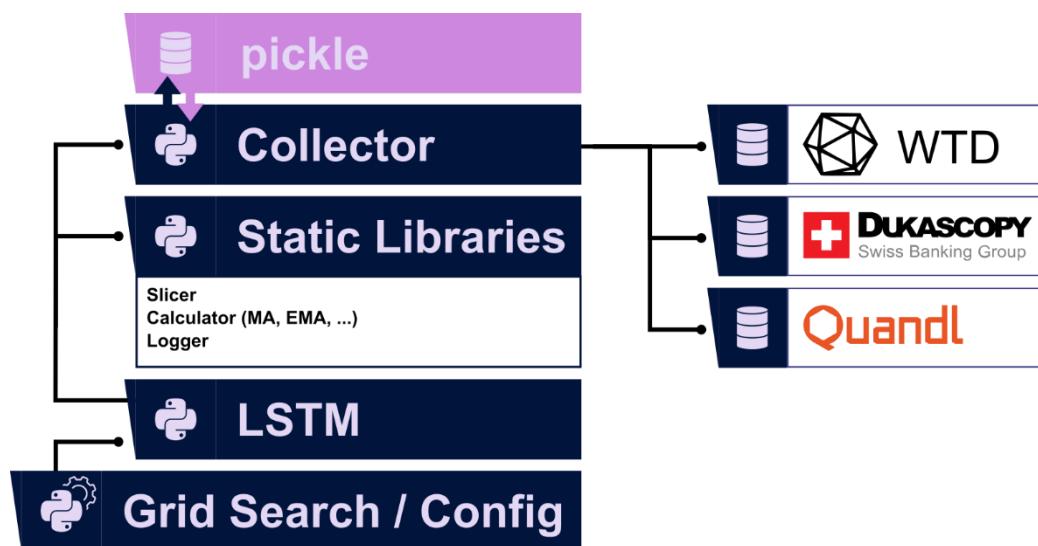


Abbildung 7: Gliederung Softwarearchitektur. Gestartet wird mit einer Konfiguration (Grid Search / Config). Die Datei LSTM ist der Knotenpunkt, welcher das Netzwerk aufbaut und Hilfsbibliotheken verwendet (mitte). Die Datenquellen (rechts) werden via Collector geladen.

7.2.1 PROGRAMMSTRUKTUR

Die Programmstruktur ist so modular wie möglich aufgebaut, dass in möglichst kurzer Zeit Berechnungen durch andere ersetzt werden können oder unterschiedliche Vorverarbeitungsschritte in einem einzigen Grid Search angesprochen werden können.

Gestartet wird immer von einem konfigurierten Grid Search aus. Dieser stellt alle möglichen Konfigurationen der gewählten Parameter zusammen. Anschliessend wird mit einer dieser Konfiguration der LSTM Builder angestoßen, welcher das Netzwerk baut, die Daten verarbeitet und trainiert. Am Ende jedes Trainings wird ein Report erstellt, welcher im Slack gepostet wird. Zusätzlich werden alle wichtigen Objekte (Model, Dataframe), etc. in einem testspezifischen Ordner gespeichert. So kann der Test zu einem späteren Zeitpunkt wieder geladen und analysiert werden.

7.2.2 GRID SEARCH / CONFIG

Von hier aus lässt sich eine Testunit steuern. Zu jedem Parameter lassen sich beliebig viele Optionen eintragen. Beim Start der Testunits wird aus jeder möglichen Parameteroption jeweils ein Test mit Abschlussreport durchgeführt.

Folgende Parameter stehen dabei zur Verfügung.

Parameter	Format	Beschreibung
selectedSymbols	String Array	Input des Netzwerkes (siehe 6.3)
startdate	String (Pandas Date)	Startdatum der Kurse, resp. des Trainings
scale	String (Pandas Time)	Datenskalierung, bestimmt die Auflösung der Datenpunkte
ctivation	String (tanh, sigmoid, relu)	Aktivierungsfunktion des abschliessenden Dense-Layers
trainsetSize	Float	Grösse des Trainings
predictionLength	Integer	Wie weit die Vorhersage in Zeiteinheiten geht (Parameter scale)
blockSize	Integer	Grösse eines Blocks
numNodes	Integer Array	Aufbau des Netzwerkes (siehe 7.2.2.1)
dropout	Float	Dropout nach jedem Layer
useGPU	Boolean	Baut das neuronale Netz mit CuDNNLSTM Zellen welche für das Training mittels GPU optimiert sind
shuffle	Boolean	Mischen der Batches vor jeder Epoche
debug	Boolean	Bietet mehr Informationen zum aktuellen Test. Ausserdem fällt der Report in Slack weg
showPlots	Boolean	Wenn <debug> aktiv ist, werden Plots zu den Inputdaten erstellt

reloadData	Boolean	Keine zwischengespeicherten Kursdaten verwenden (Nur im Zusammenhang mit WTD als Quelle)
name	String	Name des Tests und somit des Testverzeichnisses mit Abschlussinformationen
specialTimeWindows	Integer Array	Datensatz aus der Vergangenheit ($Wert * blocksize * scale$)

Übersicht der Konfigurationsmöglichkeiten eines Netzwerkes.

Mit folgenden Parametern lässt sich der Adam Optimizer konfigurieren.

Parameter	Format	Beschreibung
adamLR	Float	Adams Learning Rate zum Startzeitpunkt
adamBeta_1	Float	Adam Beta1 Parameter
adamBeta_2	Float	Adam Beta 2 Parameter
adamEpsilon	Float	Adam Epsilon zur Verhinderung von Nulldivisionen
adamDecay	Float	Adam Decay zur dynamischen Verkleinerung der Learning Rate
adamAmsgrad	Boolean	Alternativer Optimizer Amsgrad

Übersicht der Konfigurationsmöglichkeiten des gewählten Optimizers eines Netzwerkes.

Eine genauere Erklärung zu den Adam Parametern kann dem Anhang "Adam Optimizer" entnommen werden.

7.2.2.1 NUMNODES, DROPOUT UND USEGPU

Die Variabel numNodes bestimmt die Anzahl Nodes der verschiedenen Layer des LSTM-Netzwerks. Die Layer werden automatisch anhand der angegebenen Array generiert.

So wird beispielsweise folgender Array angegeben:

```
1. 'numNodes'      : [
2.   [1200, 1200]
3. ],
```

Dies generiert folgendes LSTM Netzwerk:

1.	Layer (type)	Output Shape	Param #
=====			
4.	cu_dnnlstm_1 (CuDNNLSTM)	(None, 196, 1200)	5798400
=====			
6.	dropout_1 (Dropout)	(None, 196, 1200)	0
7.	=====		
8.	cu_dnnlstm_2 (CuDNNLSTM)	(None, 1200)	11529600
9.	=====		
10.	dropout_2 (Dropout)	(None, 1200)	0
11.	=====		
12.	dense_1 (Dense)	(None, 2)	2402
13.	=====		
14.	Total params:	17,330,402	
15.	Trainable params:	17,330,402	
16.	Non-trainable params:	0	
17.	=====		

Die Anzahl der zu trainierenden Parameter hängt von den Input Daten ab. Die Dropout Layer können mittels Parameter dropout verändert werden. Wie im Code ersichtlich werden CuDNNLSTM Zellen verwendet. Hierbei handelt es sich um LSTM Zellen welche für das Training mit Grafikkarten optimiert sind. Dies kann mittels Parameter useGPU geändert werden.

7.2.3 COLLECTOR

Die Klasse Collector wird vom LSTM Builder gestartet, mit der spezifischen Anfrage nach bestimmten Kursdaten. Für jede Datenquellen gibt es einen eigenen Collector aufgrund der Eigenheiten jeder Quelle.

- Collector Duka – Dieser bezieht die Daten aus CSV Dateien, welche von der Handelsplattform Dukascopy bezogen werden.
- Collector (WTD) – Dieser arbeitet aktiv mit der API von WTD und speichert Abfragen in Pickle Dateien.

Beide sind von Modulen entkoppelt und können auch für andere Software Projekte verwendet werden. Die Collector sind unter anderem auch dafür verantwortlich, dass die Daten in einer guten Qualität gespeichert werden. Also ohne Nullwerte, mit dem Fokus auf den Hauptkurs, welcher vorhergesagt werden soll.

7.2.4 HILFSBIBLIOTHEKEN

Um den LSTM Builder möglichst kompakt zu halten wurde eine Ansammlung an Bibliotheken mit statischen Funktionen erstellt, welche die Arbeit mit den Daten erleichtert.

Klasse/Datei	Beschreibung
Calculator	<ul style="list-style-type: none">• Berechnungsfunktionen für Technische Indikatoren• Auswertungsalgorithmen• Custom-Loss-Berechnungen
Logger	<ul style="list-style-type: none">• Plot Funktionen inkl. Heatmaps
Slicer	<ul style="list-style-type: none">• Konvertierung des Dataframes zum Input Array (Numpy)• Zuschneiden des Dataframes, zum Entfernen unvollständiger Datensätze am Anfang und Ende.
SendSlack	<ul style="list-style-type: none">• Slack API Ansteuerung

Erstellte Hilfsbibliotheken in Python-Dateien.

7.2.5 AUSWERTUNGSFUNKTIONEN

Datei	Beschreibung
Evaluate	Lädt ein trainiertes Modell zur beliebigen Weiterverarbeitung.

Erstellte Auswertungsfunktionen in Python-Dateien.

8 LSTM DEEP LEARNING

8.1 WISO LSTM

Das maschinelle Lernen wurde bereits in den fünfziger Jahren entwickelt. Durch den technischen Fortschritt der Computer haben in den letzten Jahren vor allem die neuronalen Netze an Aufmerksamkeit gewonnen. Besonders das Training von RNNs (Recurrent Neural Network) ist sehr rechenintensiv. Mit aktueller Hardware sind jedoch auch diese aufwändigen Rechenoperationen problemlos zu bewältigen.

Im Jahr 1997 wurde zum ersten Mal die sogenannte LSTM Zelle vorgeschlagen und bis heute stets verbessert (Géron, Machine Learning mit Scikit-Learn & Tensorflow, 2018). Die LSTM Zelle verwaltet zwei Zustandsvektoren. Man kann sich diese Zustandsvektoren als Kurzzeitgedächtnis und Langzeitgedächtnis vorstellen.

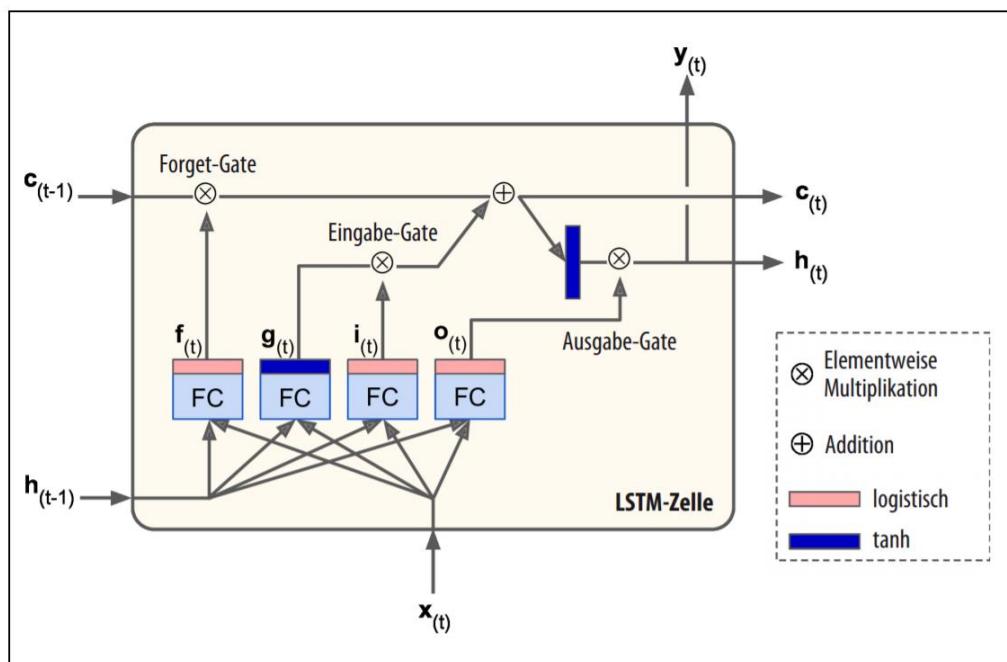


Abbildung 8: Aufbau einer LSTM Zelle, $x_{(t)}$ =Eingabevektor, $y_{(t)}$ =Ausgabevektor, $h_{(t)}$ =Kurzzeitgedächtnis, $c_{(t)}$ =Langzeitgedächtnis, t ist jeweils der aktuelle Zeitschritt, (Géron, Machine Learning mit Scikit-Learn & Tensorflow, 2018)

Die Idee der LSTM Zelle ist es, grobe Strukturen in den Daten zu erkennen und diese im Langzeitgedächtnis zu speichern. Diese gespeicherte Information wird dann an die nächste Zelle weitergegeben. So wird das Lang- und Kurzzeitgedächtnis kombiniert, um eine möglichst präzise Vorhersage machen zu können. Aus diesem Grund eignen sich LSTM Zellen vor allem für die Vorhersage von Zeitbasierten Daten wie Audiosignalen, Messdaten oder eben Aktienkursen.

Ein Paper der "Arab Academy for Science and Technology" vergleicht verschiedene ML-Algorithmen zur Vorhersage von Aktienkursen. Die besten Ergebnisse wurden mit LSTM Zellen erzielt (Areej Abdullah Baasher, 2011).

Aus diesen Gründen hat man sich entschieden ebenfalls ein Neuronales Netz mit LSTM Zellen in unserer Arbeit zu verwenden.

8.2 ÜBERLEGUNGEN ZUR STRATEGIE

Um zu bestimmen, welche Modelle eine sinnvolle Vorhersage der Kursdaten treffen, muss zuerst ein einziges Modell gefunden werden, welches in der Lage ist eine Trefferquote (Kurs steigend oder fallend) von über 50% zu erzielen. Erst dann ist die Vorhersage präziser als Zufall. Bis dahin können die jeweiligen Modelle kaum voneinander unterschieden werden.

Dies ist schwieriger als man denkt. Die meisten Modelle erzielen ab und zu eine Quote von 52%, dies bringt aber wenig. Da sich diese noch immer kaum von anderen unterscheiden lassen.

Die Strategie war nun folgende:

1. Nach dem Clustering ein Portfolio zusammenstellen, von welchem jeweils ein Kurs vorhergesagt wird.
2. Input definieren, Indikatoren berechnen, Zeitraum festlegen.
3. Ein Modell parametrisieren und mittels Grid Search evaluieren. Ist man noch immer unter ~55% wird nochmals bei Schritt 1 gestartet.

Die Modellkonfiguration, sowie der Dateninput erlauben unendlich viele Kombinationsmöglichkeiten.

Da es ausserdem keinen Sinn macht, alle Fehlritte zu dokumentieren, ist das gesamte Vorgehen im nächsten Kapitel zusammengefasst.

8.3 MODELLE / STRATEGIEN

Es gibt verschiedene Ansätze den zukünftigen Verlauf eines Aktienkurses vorherzusagen. In dieser Arbeit wurden folgende Ansätze angewendet.

8.3.1 REGRESSION (BASISMODELL)

Ein erster Ansatz ist es, den genauen Preis des Kurses für eine bestimmte Anzahl Zeitschritte in die Zukunft vorherzusagen. Das bedeutet, es wird versucht, den exakten Verlauf des Kurses abzubilden. Der ML-Algorithmus muss also ein sogenanntes Regression-Problem lösen. Der Loss wird aus dem Abstand des vorhergesagten und des reellen Kurses berechnet. Je näher der vorhergesagte und der reelle Kurs beieinander liegen, desto kleiner der Loss.

8.3.1.1 AUFBAU – ARCHITEKTUR

- **Input:** Als Input können beliebig viele Kurse und Indikatoren verwendet werden. Die Kurse wurden wie im Abschnitt 0 beschrieben, zugeschnitten und im Bereich zwischen Null und Eins skaliert.
- **Output:** Für jeden zu vorhersagenden Zeitpunkt gibt das Netz einen Wert zwischen Null und Eins aus. Dieser Wert kann wiederum auf einen aktuellen Kurswert zurückgerechnet werden. Das Zeitfenster, welches vorhergesagt werden soll, wird mit dem Parameter predictionLength bestimmt. Wird bei einer Skalierung von

15min eine predictionLength von vier angegeben, wird also die nächste Stunde vorhergesagt.

Input → LSTM → [0..1]

Abbildung 9: Regression Workflow

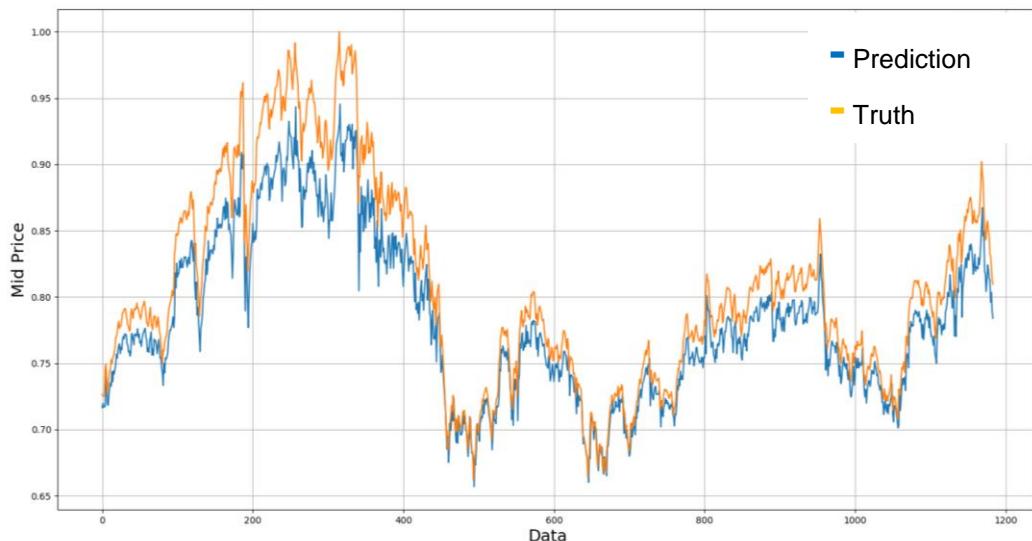


Abbildung 10: Regression Vorhersage Plot, vergleicht reellen Kurs und Vorhersage

Damit das Modell weiss, in welcher Form die Input Daten vorliegen, muss beim ersten Layer ein Input Shape angegeben werden. Dieser wird im Programm automatisch aus den Input Daten bestimmt.

```
1. input_shape=(conf['blockSize'], len(df.columns))
```

Die erste Dimension wird von der Grösse der Samples bestimmt, mit der die Daten zugeschnitten wurden. Die Anzahl Spalten der Input Daten, also die Anzahl Kurse und Indikatoren, bestimmt die zweite Dimension des Input Shapes.

Die Anzahl Layer und Nodes des Netzwerks werden ebenfalls anhand der Config Datei generiert (siehe Kapitel 7.2.2.1). Mittels Parameter dropout wird jedem Hidden Layer im Netzwerk ein Dropout mit dem entsprechenden Wert angehängt.

```
1. if conf['dropout'] > 0:  
2.     model.add(Dropout(conf['dropout']))
```

Im Anschluss wird das Modell kompiliert, also für das Training vorbereitet. Hier wird sowohl die Loss-Funktion als auch der Optimizer definiert. Die Loss-Funktion bestimmt die Formel, mit der der Loss berechnet wird. In dieser Arbeit wurden verschiedene Loss Funktionen getestet. Zu Beginn wurde der Mean Squared Error (MSE) eingesetzt. Der Optimizer bestimmt, wie die Gewichte im Netzwerk angepasst werden, um den Loss zu minimieren. Auch hier wurden verschiedene Optionen verwendet. In einem ersten Schritt wurde der Adam-Optimizer verwendet. Beide Funktionen sind für das erfolgreiche Lernen des Netzwerks essenziell.

```

1. optimizer = optimizers.Adam(lr=conf['adamLR'], beta_1=conf['adamBeta_1'],
   beta_2=conf['adamBeta_2'], epsilon=conf['adamEpsilon'], decay=conf['adamDecay'],
   amsgrad=conf['adamAmsgrad'])
2. model.compile(loss=mse, optimizer=optimizer)

```

8.3.1.2 TRAINING

Das Modell kann nun mit den vorbereiteten Testdaten trainiert werden. Das Training wird mit `model.fit()` gestartet.

```

1. batch_size = conf['batchSize']
2. if batch_size == -1:    batch_size = trainX.shape[0]
3. model.fit(trainX, trainY, batch_size=batch_size,
4.            epochs=conf['numberOfEpochs'], shuffle=conf['shuffle'],
5.            callbacks=[tensorboard, mcpSave, moneyMaker],
6.            validation_data=(testX, testY)) # train the model

```

Wie im Code ersichtlich können nebst den Trainingsdaten (`trainX`, `trainY`) noch weitere Parameter übergeben werden. Was die verschiedenen Parameter bedeuten wird hier aufgelistet:

Parameter	Beschreibung
<code>batch_size</code>	Dieser Wert bestimmt, wie viele Blocks/Samples der Input Daten das Netzwerk sieht, bis die Gewichte angepasst werden. $1 < \text{batch_size} < \text{Anzahl Samples im Trainingsset}$ Je höher die <code>batch_size</code> ist, desto mehr Grafikkartenspeicher wird benötigt. Hier ist also die Hardware der limitierende Faktor.
<code>epochs</code>	Die Anzahl der zu trainierenden Epochen. Dies bestimmt, wie oft das Training mit dem kompletten Trainingsdatensatz durchgeführt werden soll. Dieser Wert kann ebenfalls in der Config Datei angegeben werden.
<code>callbacks</code>	Hier werden verschiedenen Funktionen angegeben, welche bei einer vollendeten Epoche aufgerufen werden. <ul style="list-style-type: none"> Das Tensorboard ist eine davon und dient dazu, den Lernprozess zu visualisieren. Die Keras Funktion ModelCheckpoint speichert das beste Modell mit dem kleinsten Loss. Zusätzlich wird eine eigene Funktion zur Ermittlung der korrekten Vorhersagen mitgegeben (siehe 9.1.3.2).
<code>validation_data</code>	Dies sind die Daten, welche nach jeder Epoche zur Validierung verwendet werden. Die Validierungsdaten haben keinen direkten Einfluss auf den Lernprozess.

Erläuterung der Parameter von Keras für das Training eines Netzwerkes.

Sobald alle Epochen abgeschlossen sind, ist das Training beendet. Nun wird das finale Modell lokal gespeichert und kann somit für zukünftige Evaluationen oder Tests einfach erneut geladen werden.

```
1. model.save(os.path.join(logDirTensorboard, 'lastModel.h5'))
```

Der Loss bildet sich aus dem MeanSquareError der absoluten Abweichung.

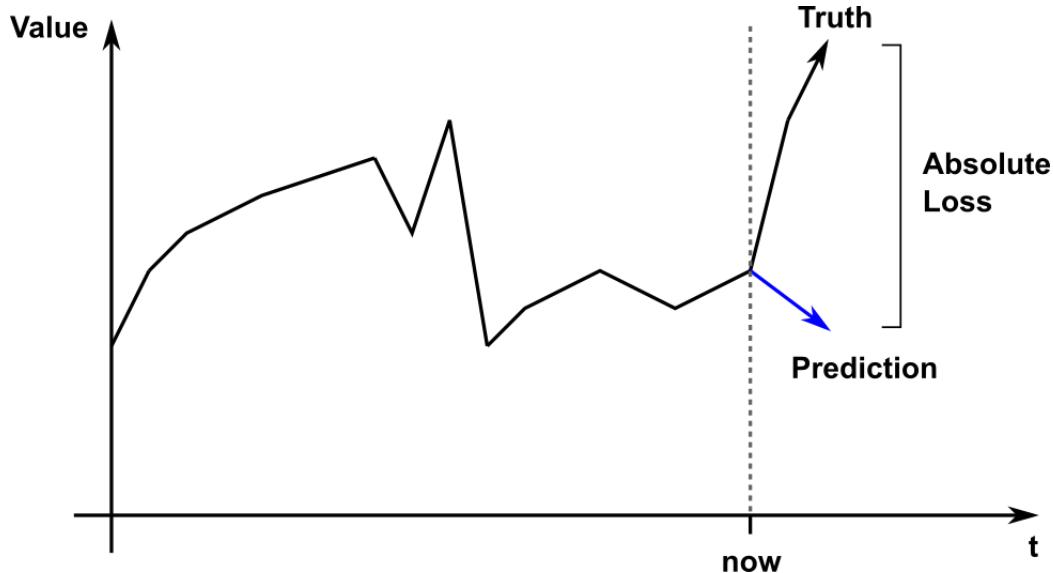


Abbildung 11: Visualisierung der Absolute Loss Funktion. Diese Setzt sich aus der Abweichung des Kursverlaufs und der Vorhersage zusammen (kleinere Abweichung ist besser).

8.3.1.3 EVALUATION

Nachdem das Training abgeschlossen ist, soll das Modell auf die korrekte Vorhersage geprüft werden.

Zu Beginn des ganzen Trainingsvorgangs werden 20% der Gesamtdaten für die Evaluation zur Seite gelegt, welche das Netzwerk während des Trainings nie erhält, jedoch vorhersagen muss.

Um die Qualität der Testdaten zu beurteilen, wurden folgende Ansätze gewählt.

8.3.1.3.1 VERGLEICH MITTELS VALIDATION LOSS

Mittels Tensorboard kann der Verlauf des Loss und Validation Loss visualisiert und somit verglichen werden.

Der Validation Loss wird im Gegensatz zum Loss nicht anhand der Trainingsdaten, sondern Anhand der Testdaten berechnet. Dies bietet den Vorteil, dass während des Trainings direkt ersichtlich ist, ob sich das Modell auch bei der Vorhersage der Testdaten verbessert. Steigt beispielsweise der Validation Loss während der Loss weiter sinkt, könnte dies ein Zeichen für Overfitting sein. Das bedeutet, dass das Modell beginnt, die Testdaten auswendig zu lernen. Die Stabilität des Validation Loss gibt Auskunft über Zufallstreffer.

Zur Evaluation kann der Validation Loss verglichen werden.

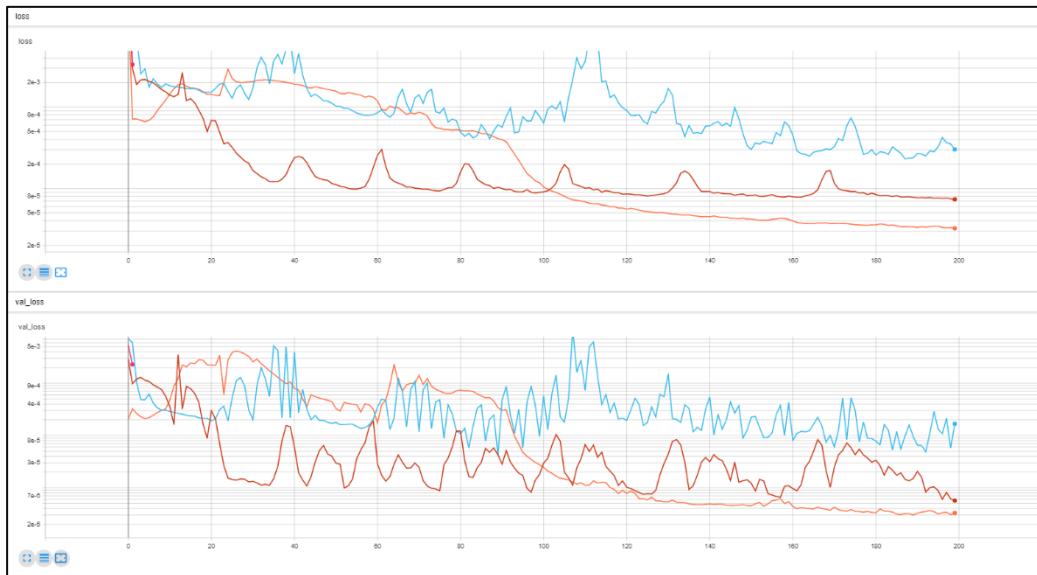


Abbildung 12: Screenshot Tensorboard. Visualisierung der Loss-Funktion von drei verschiedenen Modellen (tiefer ist besser).

8.3.1.3.2 KORREKTHEIT DER VORHERSAGE DER KURSRICHTUNG

Damit die Resultate dieser Evaluierungsdaten gerecht verglichen werden können, bedarf es an präzisen Evaluierungsfunktionen. Der Fokus ist so gewählt, dass diejenigen Aspekte der Vorhersage richtig ausgewertet werden, welche auch im Aktienhandel relevant wären. Also starke Steigungen sollen dabei richtig erkannt werden. Ein flacher Kursverlauf hingegen generiert weder grössere Verluste noch Gewinne. Diese geraden Abschnitte werden demnach gar nicht berücksichtigt.

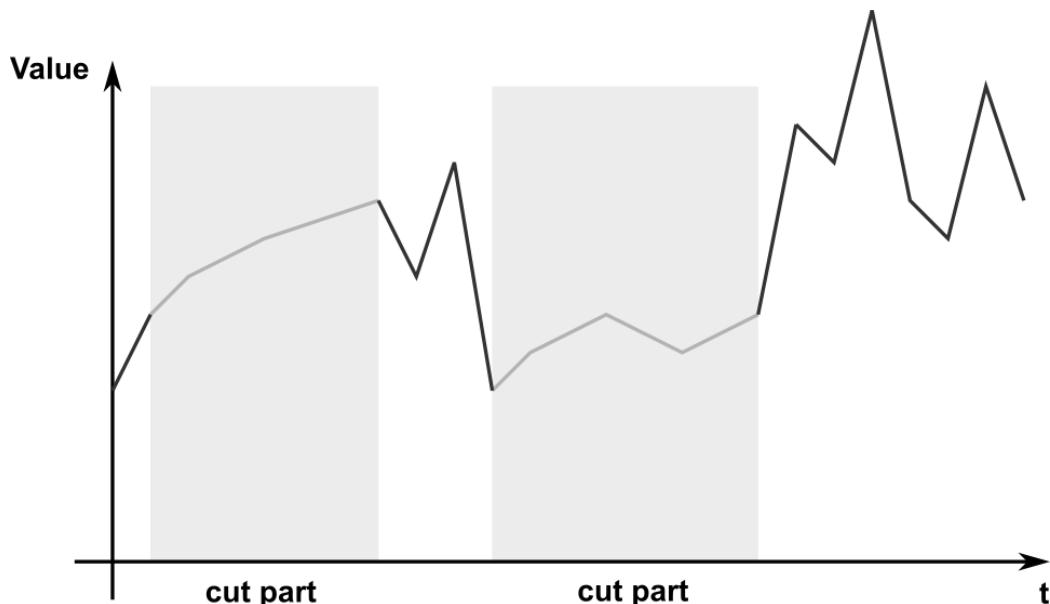


Abbildung 13: Evaluation der Vorhersage (zeigt die relevanten Veränderungen)

20% der Abschnitte, welche die kleinste Kursänderung enthalten, werden bei der Evaluierung nicht berücksichtigt, da hier weder ein Kauf- noch ein Verkaufssignal

vorliegt. Über die restlichen 80% der Daten wird mittels Spearman zwischen Vorhersage und Wahrheit ermittelt, wie gut die Vorhersage ausgefallen ist.

Mit dieser Methode erhält man von jedem Test eine Prozentzahl der Vorhersage und kann die Tests auf einen Blick vergleichen. Das Resultat wird als DirectionMatch festgehalten.

8.3.1.4 WEITERE GETESTETE VARIATIONEN DER REGRESSION

Im Verlauf der Arbeit wurden weitere Variationen des Regression-Modells getestet. Explizit wurden Input und Output Daten angepasst, während das Netz unverändert blieb.

8.3.1.4.1 LOG-RETURN

Einerseits wurde versucht, den sogenannten Log-Return eines Aktienkurses vorherzusagen.

```
1. @staticmethod
2. def logReturn(df):
3.     df = df.copy()
4.     if isinstance(df, pd.DataFrame):
5.         for column in df:
6.             df[column] = Calculator.logReturn(df[column])
7.     df = df.fillna(0.)
8.     return df
9. # fnc for Series
10. sr = df
11. sr = np.log(sr / sr.shift(1)) #log = natural logarithm (to base e)
12. sr = sr.replace([np.inf, -np.inf], 0.)
13. sr = sr.fillna(0.)
14. return sr
```

Die Berechnung des Log-Returns aus Zeile 11 basiert auf folgender Formel:

$$R = \ln\left(\frac{P_{t-1}}{P_t}\right)$$

Wobei R der Log-Return, P_t der aktuelle Preis und P_{t-1} der Preis der letzten Zeitperiode ist. Werden diese berechneten Werte in einem Plot dargestellt, sehen die Input Daten wie folgt aus:

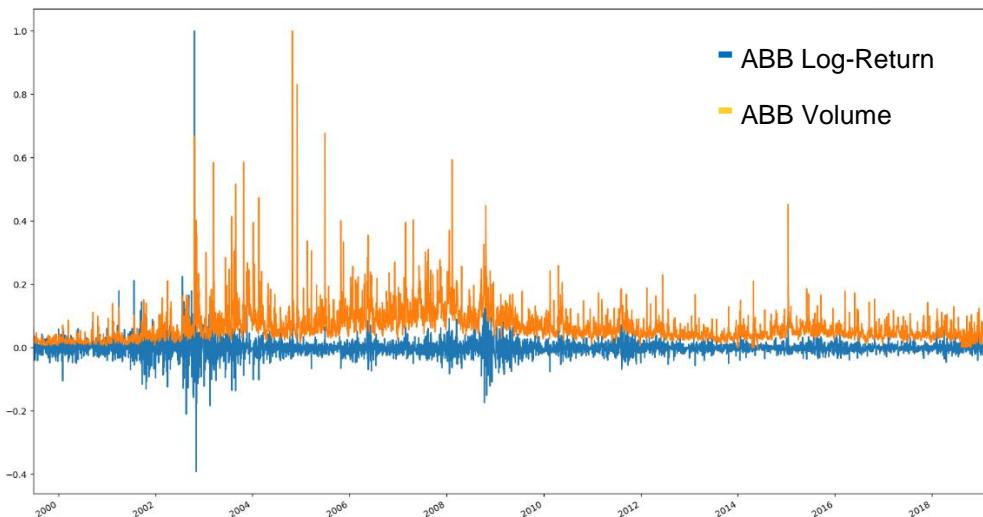


Abbildung 14: Plot Log-Return (blau) von ABB Aktienkurs und Handelsvolumen (orange)

Die Ergebnisse fallen leider schlecht aus. Das Netzwerk kann keinen Kurs signifikant besser als den Zufall vorhersagen. Wie im Plot ersichtlich scheint es, als seien die Daten zu stark verrauscht, um dem neuronalen Netz verwertbare Informationen zu liefern. Aus diesem Grund hat man sich dazu entschieden, diese Methodik nicht weiter zu verfolgen.

8.3.1.4.2 SHIFTING

Nebst dem Log-Return wurde eine weitere Funktion implementiert. Die Output Daten wurden hier insofern angepasst, dass jede Sequenz auf der Nulllinie startet. Damit soll erreicht werden, dass sich das Netz mehr auf die Richtung des Kurses konzentrieren kann. Die Input Daten blieben unverändert. Das Netz sieht also weiter eine normale Sequenz eines Kurses. Die vorherzusagende Sequenz beginnt jedoch immer bei null.

```
1. @staticmethod
2. def shiftY(x, y):
3.     for i in range(len(y)):
4.         y[i] = y[i] - x[i,-1,0]
```

Wie im Code ersichtlich, wird dies erreicht, indem von jedem Punkt in der Output Sequenz, der Wert des letzten Punkts in der Input Sequenz abgezogen wird.

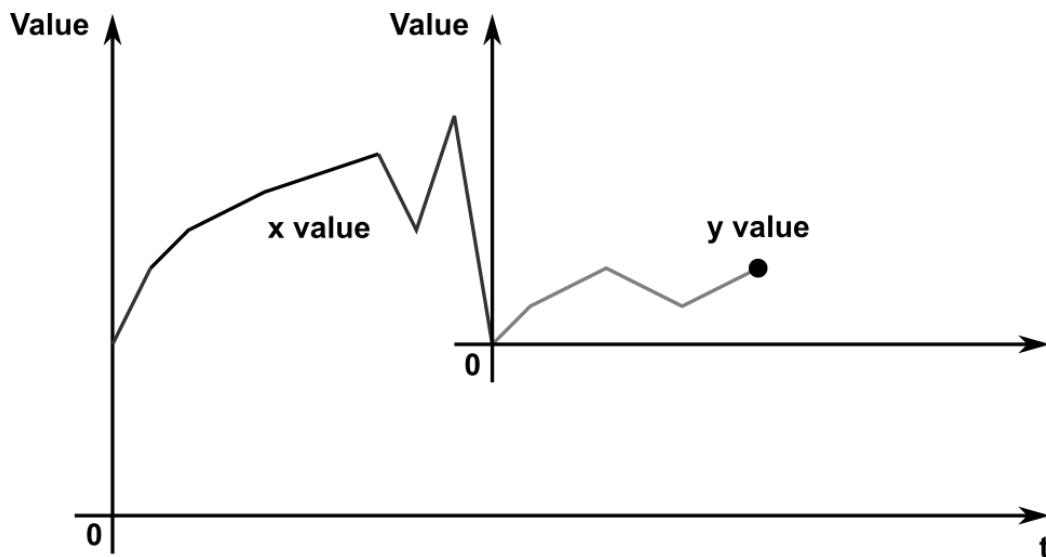


Abbildung 15: Visualisierung der ShiftY Funktion. Jede Output Sequenz wird so verschoben, dass sie auf der Nulllinie beginnt.

Nach einigen Tests wird auch dieser Ansatz aufgrund schlechter Vorhersagen verworfen. Auch hier kann das LSTM-Netzwerk keine signifikant bessere Vorhersage als der Zufall machen.

8.3.2 CLASSIFICATION

Ein weiterer Ansatz welcher in dieser Arbeit behandelt wird, ist die Klassifikation. Das neuronale Netz soll hier nicht den genauen Kurswert bestimmen, sondern entscheiden, ob der Kurs in der Zukunft steigen, fallen oder unverändert sein wird. Das Netzwerk versucht also, die Input-Sequenz einer bestimmten Klasse zuzuordnen. Dieser Ansatz wird oft in Bilderkennungs-Modellen verwendet.

8.3.2.1 AUFBAU – ARCHITEKTUR

Die Anzahl Klassen kann in der Config angegeben werden und muss ungerade sein. Für jede Klasse gibt das Netzwerk einen Wert zwischen Null und Eins aus. Pro Sequenz ergeben alle Klassen aufsummiert genau eins. Je sicherer das Netzwerk ist, die korrekte Klasse anzugeben, desto höher ist der entsprechende Wert.

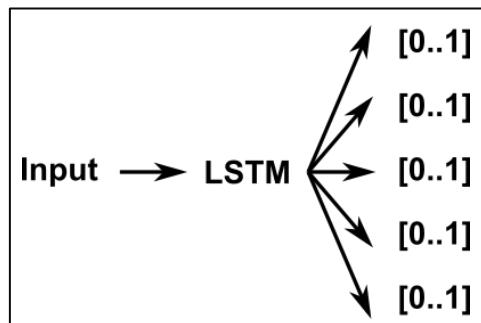


Abbildung 16: Output Klassifizierung LSTM-Netzwerk mit fünf Klassen

Um solch einen Output zu erreichen, wird die bestehende Konfiguration entsprechend angepasst. Der Output Layer erhält nun genau so viele Nodes, wie Klassen in der Config angegeben werden. Zusätzlich erhält er eine Aktivierungsfunktion namens Softmax.

```
1. model.add(Dense(conf['numCategories'], activation='softmax'))
```

Diese ist dafür verantwortlich, dass die Summe aller vorhergesagten Kategorien Eins ergibt.

8.3.2.2 LABELING

Um das Netzwerk trainieren zu können, müssen die verschiedenen Input Sequenzen einer Klasse zugewiesen und mit dem entsprechendem Label versehen werden.

```
1. # gauss borders
2. yAbs = [abs(val) for val in y]
3. yAbs.sort()
4. # biggestDiff = yAbs[int(0.75*length)]
5. biggestDiff = yAbs[int(0.75*length)]
6.
7. #---Classification-----
8. classSize = biggestDiff*2/numCategories
9. for i in range(length):
10.     classFound = False
11.     classArray = np.zeros(numCategories)
12.     # check (from top to bottom) if value fits to the class
13.     # Class '0' is biggest diff upwards, '4' is biggest downwards, '2'
14.     # is flat (with 5 classes)
15.     for clas in range(numCategories):
16.         if (y[i] >= (biggestDiff-((clas+1)*classSize))):
17.             classArray[clas] = 1
18.             classFound = True
19.             break
20.         #if no class fits, the value is the most negative of all time so it
21.         #fits to the last class
22.         if not classFound:
23.             classArray[numCategories-1] = 1
24.             #set classArray as label
25.             y[i]=classArray
26.
27. #-----
28. x = np.array(x)
29. y = np.array(y)
30. return x, y
```

Um dies zu erreichen, wurden für jede Sequenz die Steigungen zwischen dem letzten Datenpunkt der Sequenz und dem vorherzusagenden Datenpunkt berechnet. Anhand dieser Steigungen wurden die Klassen so bestimmt, dass möglichst jede Klasse ähnlich viele Samples beinhaltet.

Jede Sequenz wurde mit dem entsprechenden Label versehen. Beispielsweise eine Sequenz mit sehr grosser Steigung, bei fünf Klassen, erhält folgendes Label:

[1, 0, 0, 0, 0]

8.3.2.3 TRAINING

Das Training des LSTM-Netzwerks verläuft nun, wie bereits in Kapitel 8.3.1.2 beschrieben, jedoch wurde für die Klassifizierung eine andere Loss Funktion verwendet.

Als Loss Funktion wird die Categorical Crossentropy verwendet, welche in Keras bereits implementiert ist und einfach importiert werden kann.

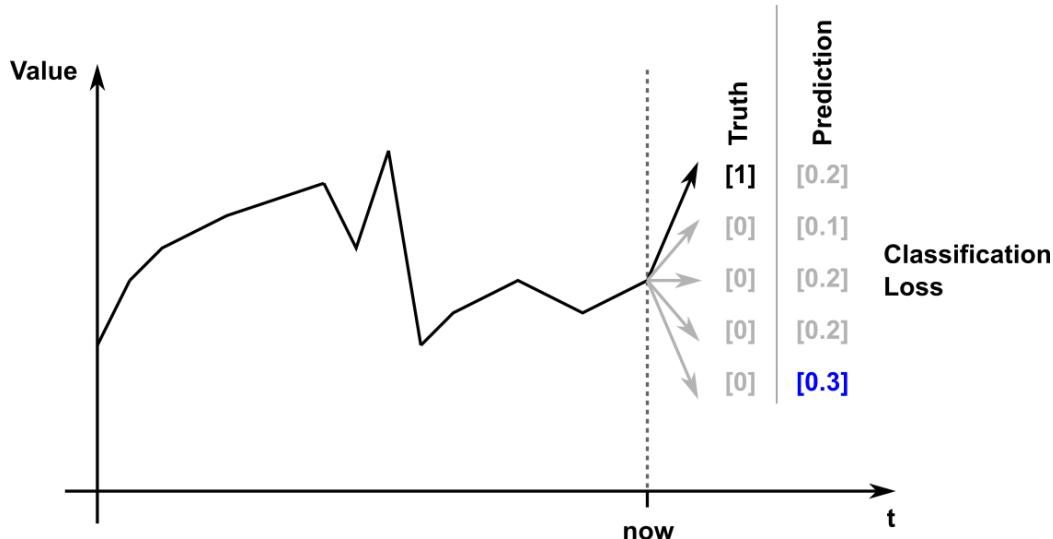


Abbildung 17: Visualisierung der Classification Loss Funktion. Im Vergleich stehen der Vorhersage Vektor (Prediction) und der Vektor zur korrekten Kursrichtung (Truth).

8.3.2.4 EVALUATION

Wie im Regressionsmodell wird eine Funktion implementiert, welche die Qualität der Vorhersage in einer Prozentzahl berechnet. Es wird überprüft, welche Klasse in der Vorhersage den grössten Wert enthält und verglichen, ob dies mit dem Label übereinstimmt. Die Funktion checkMoneymakerClassification gibt folgende Prozentwerte zurück:

Korrekte Klasse vorhergesagt [%]	Jede Vorhersage wird gewertet und die Anzahl korrekt erkannten Klassen in Prozent berechnet. Überlegung: Es soll auf einen Blick ersichtlich sein, wie viel Prozent der Klassen korrekt erkannt wurden.
Korrekte Richtung vorhergesagt [%]	Jede Vorhersage, die die mittlere Klasse wählt (keine Signifikante Kursänderung) wird nicht in der Berechnung berücksichtigt. Überlegung: Da bei solch einer Vorhersage kein Kauf oder Verkauf stattfinden würde, kann der Investor auch kein Geld gewinnen oder verlieren. Diese Vorhersagen sollten also ignoriert werden.

Korrekte Richtung vorhergesagt mit einem Wert über X [%]	Gleich wie zuletzt genannte Funktion, jedoch werden hier nur die Vorhersagen gewertet, die über einem in der Config bestimmten Wert liegen. Überlegung: Je höher der Wert der Vorhersage ist, desto sicherer ist das Netzwerk mit der Klassifizierung. Es könnte also eine Handelsstrategie sein, nur dann einen Trade einzugehen, wenn die Klassifizierung eindeutig ist.
---	--

Erstellte Auswertungsfunktionen für den direkten Vergleich der trainierten Netzwerke.

Die korrekt vorhergesagten Klassen lassen sich mittels implementierter Confusion Matrix visualisieren.

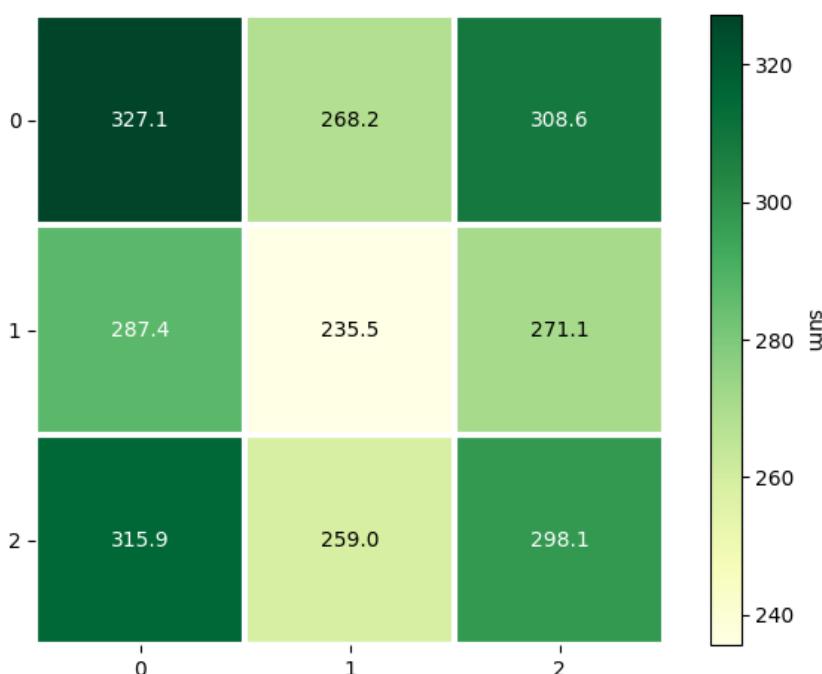


Abbildung 18: Confusion Matrix mit drei Klassen (Punkte auf der diagonalen von links oben sind besser). Beispiel von einem Grid Search mit dem SMI-Kurs.

Die Y-Achse beschreibt die reellen Klassen und die X-Achse die entsprechende Vorhersage des Netzwerks. Die Zahl in einem Feld entspricht der Summierung aller Vorhersagen dieser Klasse. Je höher diese Zahl, desto dunkler wird das Feld dargestellt. Im Idealfall sind also alle Felder, diagonal von oben links nach unten rechts dunkel und die übrigen hell.

Nach abgeschlossenem Training, wird automatisch der gesamte Testbericht inklusive Confusion Matrix auf Slack hochgeladen.

8.3.3 KOMBINATION - CUSTOM LOSS

Mit diesem Modell lassen sich die zwei Loss Funktionen Absolute-Loss und Gradient-Loss (Winkelunterschied) miteinander verbinden.

8.3.3.1 AUFBAU – ARCHITEKTUR

Die Input Daten und der Netzwerkaufbau sind dem Regressionsmodell gleichgestellt.
Nur die Evaluierung des Outputs wird aufgeteilt.



Abbildung 19: Custom-Loss Netzwerkarchitektur

8.3.3.2 TRAINING

Das Training gestaltet sich gleich wie unter Kapitel 8.3.1.2 beschrieben. Der Loss wird zusammengesetzt aus der absoluten Abweichung und der Winkelabweichung zwischen des vorhergesagten und dem absoluten Wert.

1. `Loss = a * mse(pred,truth) + b * mse(tanh(pred-previousValue), tanh(truth-previousValue))`
2. `a = absolute Loss weights`
3. `b = gradient Loss weights tanh`
4. `a + b = 1`

8.3.3.3 EVALUATION

Vorhersagen mit diesem Modell sind die erfolgreichsten im Vergleich mit allen anderen getesteten Modellen. Ein kompletter Grid Search ist unter Kapitel 9.1 beschrieben.

9 ERGEBNISSE DER ARBEIT

In diesem Kapitel sind die Ergebnisse der gesamten Arbeit festgehalten. Gegliedert sind diese nach Modell und Eigenschaften. Die jeweiligen Evaluierungsmethoden sind in Kapitel 8.3 unter dem jeweiligen Modell beschrieben.

9.1 GRID SEARCH MIT DEM CUSTOM LOSS MODELL

9.1.1 KORRELATION UND DATENAUSWAHL

Die Zeitspanne der Korrelation ist durch die gegebene Rechenleistung beschränkt und wurde so detailreich wie möglich gewählt (15min Takt).

Parameter	Wert
Korrelationszeitraum	180 Tage
Maximale Verschiebung	2 Tage
Zeiteinheit	15min Takt
Korrelationsmethode	Spearman

Parametrisierung der Kreuzkorrelation aller gegeben Daten.

Diese Korrelation ergab folgendes Ergebnis:

Kurs	Spearman-Index	Shift
ADENCHCHF	0.9419	0
ESPIDXEUR	0.9364	5
NLDIDXEUR	0.9356	0
GBRIDXGBP	0.9344	0
EUSIDXEUR	0.9340	1
DEUIDXEUR	0.9230	1
HEIDEEUR	0.9158	1
...

Auszug aus den Ergebnissen der Kreuzkorrelation aller gegeben Daten.

Die markierten Zeilen markieren vorauslaufende Kurse, welche mit einer positiven Verschiebung in der Zeitachse eine höhere Korrelation ergeben haben, als ohne Verschiebung. Diese Kurse werden als Input hinzugefügt.

9.1.2 GRID SEARCH PARAMETRISIERUNG

Aus vorhergehenden Analysen können einige Parameter bereits voreingestellt werden und müssen nicht durch einen Grid Search bestimmt werden. Daraus resultiert folgende Konfiguration:

Parameter	Konfigurationen
selectedSymbols	[["SIEDEEUR_SMA8", "SIEDEEUR-Volume", "ESPIDXEUR_SMA8", "EUSIDXEUR_SMA8", "DEUIDXEUR_SMA8", "HEIDEEUR_SMA8"]]
absoluteMass	[0,1]
startdate	['2018-05-01']
scale	['15min']
adamLR	[0.001]
adamBeta_1	[0.9]
adamBeta_2	[0.999]
adamEpsilon	[None]
adamDecay	[0.0]
adamAmsgrad	[True]
activation	['relu']
trainsetSize	[0.8]
predictionLength	[12,24]
blockSize	[2*4*24]
numNodes	[[1200]]
dropout	[0.2,0.4]
numberOfEpochs	[1200]
batchSize	[4*24*5]
shuffle	[True, False]
specialTimeWindows	[[2,3,4,5]]

Parametrisierung des Grid Searches mit dem Kurs der Firma Siemens.

Die markierten Zeilen zeigen die ändernden Parameter. Es wird rekursiv jede mögliche Kombination getestet.

9.1.3 AUSWERTUNG

9.1.3.1 VALIDATION LOSS (TENSORBOARD)

Loss der Trainingsdaten:

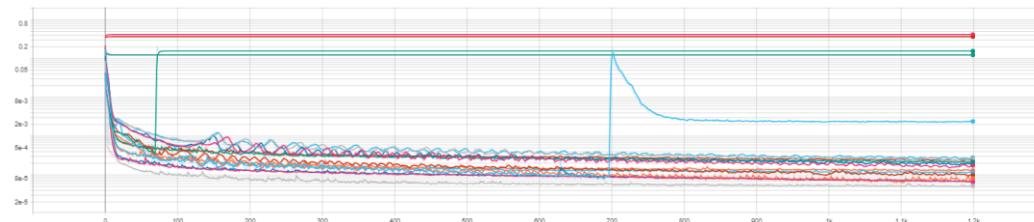


Abbildung 20: Grid Search Loss der Trainingsdaten (Logarithmisch). Screenshot aus dem Tensorboard.

Validation Loss aus den Testdaten:



Abbildung 21: Grid Search Validation Loss aus den Testdaten (Logarithmisch). Screenshot aus dem Tensorboard.

Dies zeigt folgendes:

- Ein Overfitting findet noch nicht statt, der Loss auf den Testdaten sinkt bis zum Ende.
- Einige Modelle konnten gar nicht optimiert werden oder der Dropout zerstörte wichtige Komponenten. So kommt der hohe Loss zustande. Diese Modelle zeigen folgende Vorhersage:

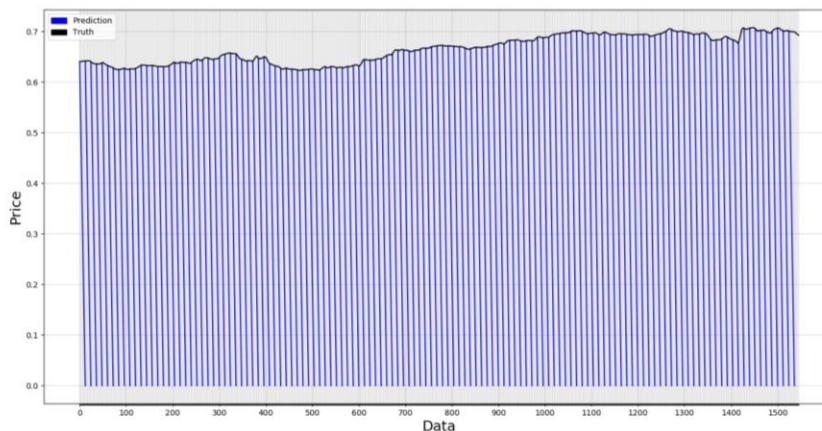


Abbildung 22: Grid Search eines fehlgeschlagenen Modells. Blaue Linien zeigen die falsche Vorhersage des Kurses, hier stark fallend

- Einige Modelle sind noch nicht fertig trainiert, der Loss ist bis zum Ende immer fallend.

9.1.3.2 RICHTIGKEIT DER VORHERSAGE DER KURSRICHTUNG

Der “beste” Test nach unserer eigenen Evaluation war folgender:

Parameter	Konfigurationen
absoluteMass	0 # Nur die Steigung wird vorhergesagt
predictionLength	24 # 24*15min = 6h
dropout	0.2 # 20% dropout
shuffle	False # kein Mischen der Batches

Beste Parametrisierung aus dem vorgehenden Grid Search.

Die richtige Vorhersage der Datenpunkte veränderte sich im Verlaufe des Tests wie folgt:

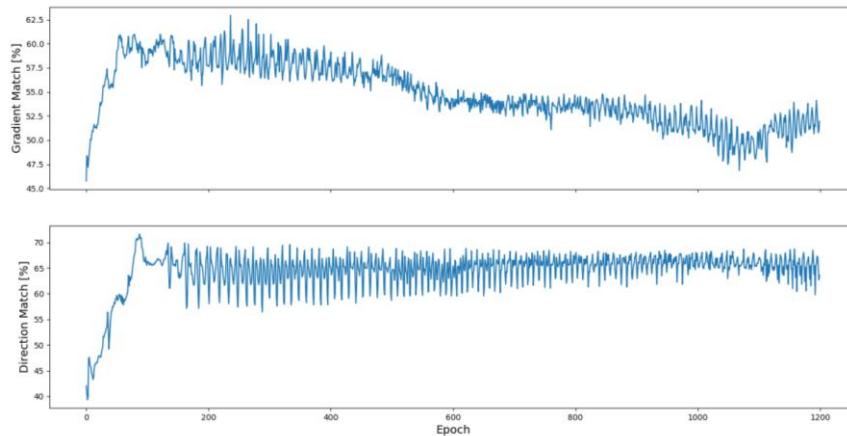


Abbildung 23: Grid Search, Vorhersagegenauigkeit nach eigener Evaluierungsfunktion

Dabei kann man gut erkennen wie das Modell in den ersten Epochen Vorschritte machte.

Im Tensorboard sieht die Veränderung des Losses wie folgt aus:

Loss der Trainingsdaten:

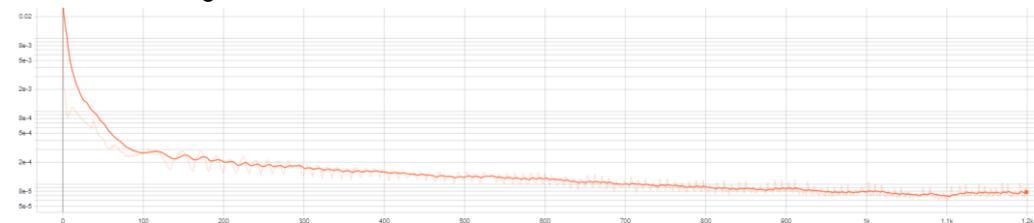


Abbildung 24: Grid Search Result, Loss der Trainingsdaten

Validation Loss aus den Testdaten:

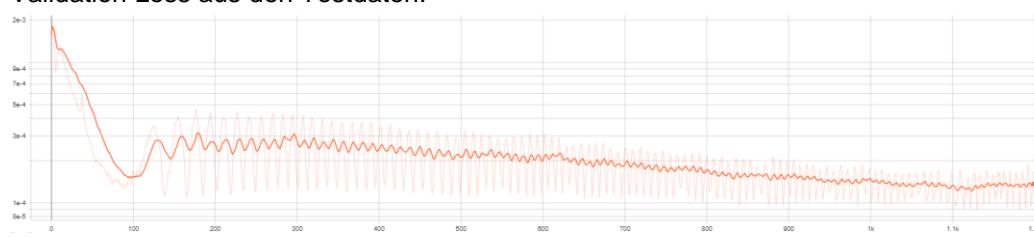


Abbildung 25: Grid Search Resultate des Validation Loss aus den Testdaten

Der Loss in den Testdaten scheint einen Tiefpunkt erreicht zu haben.

Am Ende dieses Tests stehen drei Modelle zur Verfügung, welche im Verlaufe des Trainings gespeichert werden:

- Modell mit dem tiefsten erreichten Loss:
Direction Match: 64.02% (Gradient Match: 50.76%)
- Modell mit dem grössten Direction Match:
Direction Match: 71.63% (Gradient Match: 62.93%)

- Letztes Modell des Testlaufs
Direction Match: 63.59% (Gradient Match: 51.96%)
Dieses Modell zeigt, wie sich das aktuelle/letzte Modell entwickelte. Es ist jedoch nur hilfreich, wenn sich das Modell mit dem tiefsten Loss per Zufall am Anfang des Training-Vorgangs befindet.

Evaluiert wird aufgrund des Trainingsverlaufs also das Modell mit dem tiefsten Loss.
Dieses Modell erstellte folgende Vorhersage: (Blaue Linien zeigen die Vorhersage des Kurses)

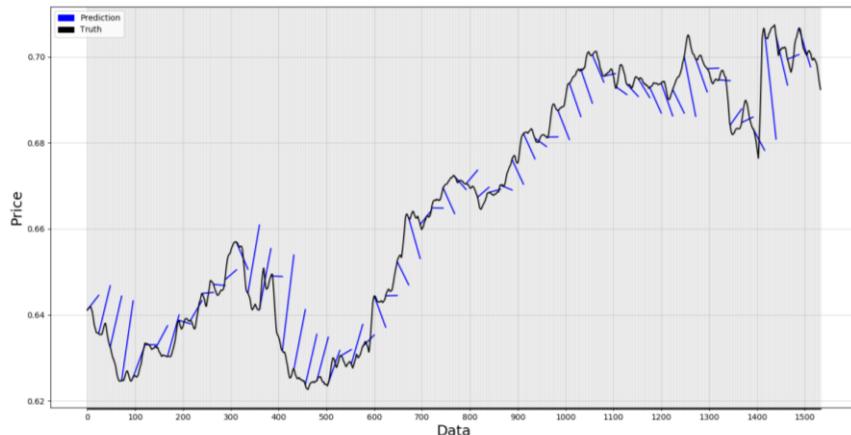


Abbildung 26: Grid Search Results, Vorhersage des besten Modells

Man kann durchaus erkennen, dass einige Vorhersagen des Kurses erfolgreich sind.

9.1.4 CROSS VALIDATION

Nun lassen sich die Daten mit neuen Kursdaten evaluieren. Dazu kann man einfach den aktuellen Monat des Kurses laden und mit dem Modell Vorhersagen.

Die Kursdaten des Trainigs sind insgesamt (Test- und Trainings-Set) bis zum 2019-05-13 datiert. Zur Evaluierung stehen Daten im Zeitraum vom 2019-05-15 bis zum 2019-07-10 zur Verfügung. Der Test wurde dem entsprechend am 10. Juli 2019 erstellt, so stehen aktuell (2. August 2019) nicht mehr als zwei Monate zur Verfügung.

Das Modell erreicht folgende Vorhersagegenauigkeit:

Direction Match in [%]: 52.85% (Gradient Match in [%]: 58.26%)

Dazu gibt es zwei naheliegende Interpretationen:

1. Die Vorhersage von über 60% ist reiner Zufall, aufgrund der grossen Anzahl an Tests, die durchgeführt wurden.
2. Die neuen Datenpunkte liegen zu weit entfernt von den Trainingsdaten, dass die Kursdaten keinen Bezug mehr haben zum Verhalten der ursprünglichen Trainingsdaten.
3. Die Vorhersage ist per Zufall auf den beiden getesteten Monaten schlecht. Weiter Daten würden eventuell zu einer besser Vorhersage führen.

9.2 GRID SEARCH MIT DEM CUSTOM LOSS MODELL UND UNTERSCHIEDLICHER KURSSKALIERUNGEN

9.2.1 PARAMETRISIERUNG

Durch den vorhergehenden Grid Search für Siemens können bewusst die Anzahl der Parameter ab sofort minimiert werden. Der Fokus dieser Tests basiert auf der Skalierung des Kurses. Es wird bewusst nur ein einziger Forex Kurs als Input gegeben. Dies hat folgende Vorteile:

- Die Datenmenge wird grösser, indem ein Kurs gewählt wird, von welchem eine lange Historie vorhanden ist. Somit sind genügend Daten vorhanden um Vorhersagen auf Tagesbasis zu machen.
- Der Trainingsvorgang dauert nicht so lange, als jener mit mehreren Kursen. Bei Vorhersagen in der 5min-Skalierung dauert ein einziger Kurs, mit der verwendeten Hardware, bereits mehr als einen Tag.

Andere Parameter sind ebenfalls leicht angepasst, um den Trainingsvorgang zu beschleunigen. Ziel ist es für jede Skalierung die gleiche Umgebung zu bieten.

Parameter	Konfigurationen
selectedSymbols	[["EURUSD", "EURUSD_EMA20", "EURUSD_EMA80", "EURUSD-Volume"]]
absoluteMass	[0,1]
startdate	['2018-05-01']
scale	['1h', '4h', '1d']
adamLR	[0.001]
adamBeta_1	[0.9]
adamBeta_2	[0.999]
adamEpsilon	[None]
adamDecay	[0.0]
adamAmsgrad	[True]
activation	['relu']
trainsetSize	[0.8]
predictionLength	[1]
blockSize	[100]
numNodes	[[1200]]
dropout	[0.2,0.4]
numberOfEpochs	[100]
batchSize	[500]
shuffle	[True, False]
specialTimeWindows	[[2,3,4,5]]

Parametrisierung des Grid Searches mit unterschiedlichen Kursskalierungen.

9.2.2 AUSWERTUNG

Das Tensorboard zeigt folgendes Ergebnis für den Loss:

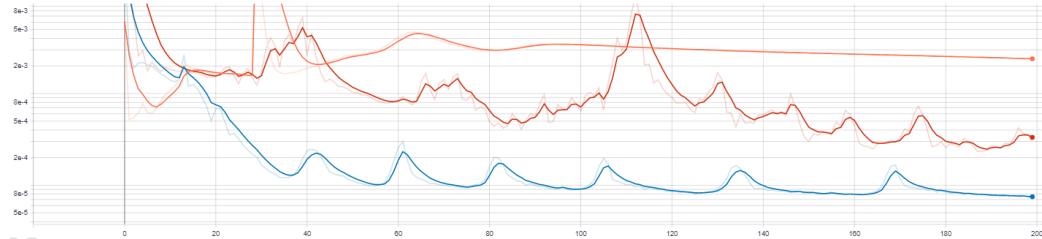


Abbildung 27: Loss aus dem Tensorboard für einen Grid Search mit unterschiedlichen Zeitskalierungen des Kurses EUR/USD, logarithmische Darstellung (tiefer ist besser).

Der Validation Loss zeigt folgenden Verlauf:

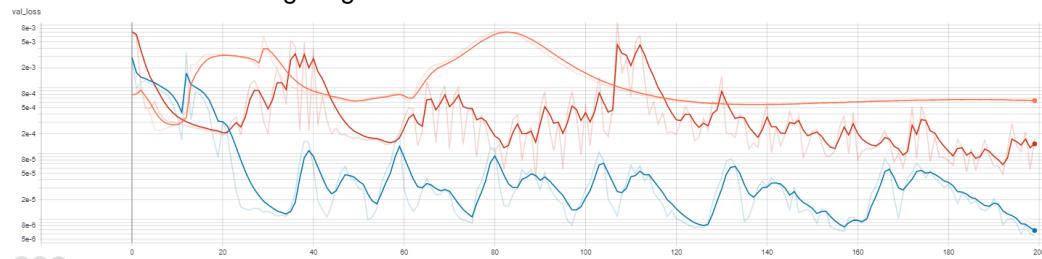


Abbildung 28: Validation Loss aus dem Tensorboard für einen Grid Search mit unterschiedlichen Zeitskalierungen des Kurses EUR/USD, logarithmische Darstellung (tiefer ist besser).

Daraus kann man folgendes schliessen:

- Ein Overfitting findet nicht statt, da der Validation Loss insgesamt nicht steigt.
- Die Netzwerke sind fast vollständig auskonvergiert. Der Validation Loss erreicht nur noch selten einen neuen Bestwert.

Die Auswertungsfunktion zeigt folgendes Resultat für das Modell mit dem tiefsten Loss. Es handelt sich dabei um das Modell mit der Skalierung auf vier Stunden.

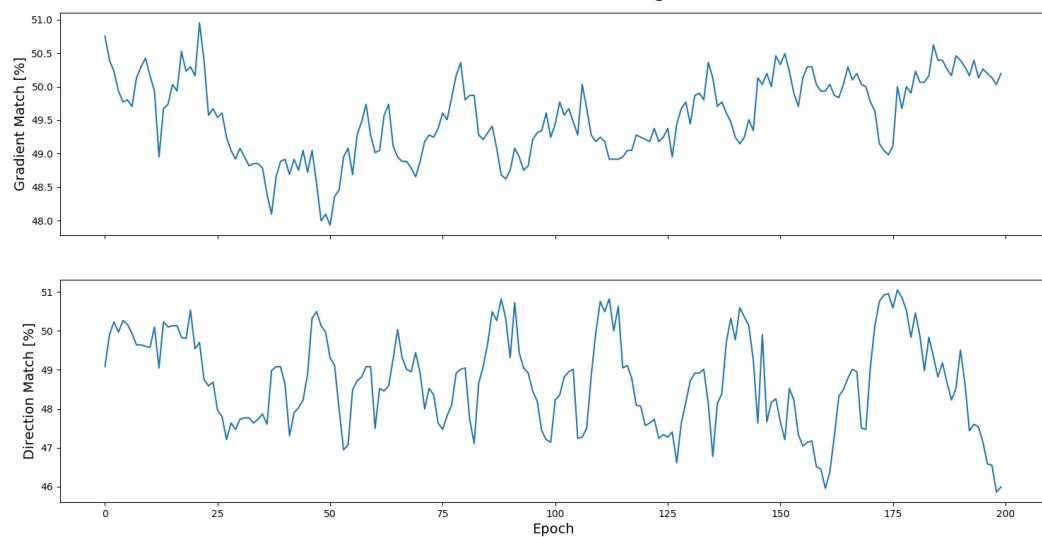


Abbildung 29: Korrekte Richtungsbestimmung für das Modell mit dem tiefstem Loss auf dem Kurs EUR/USD (höher ist besser).

Die anderen Modelle konnten ebenfalls keine signifikant bessere Vorhersage erreichen:

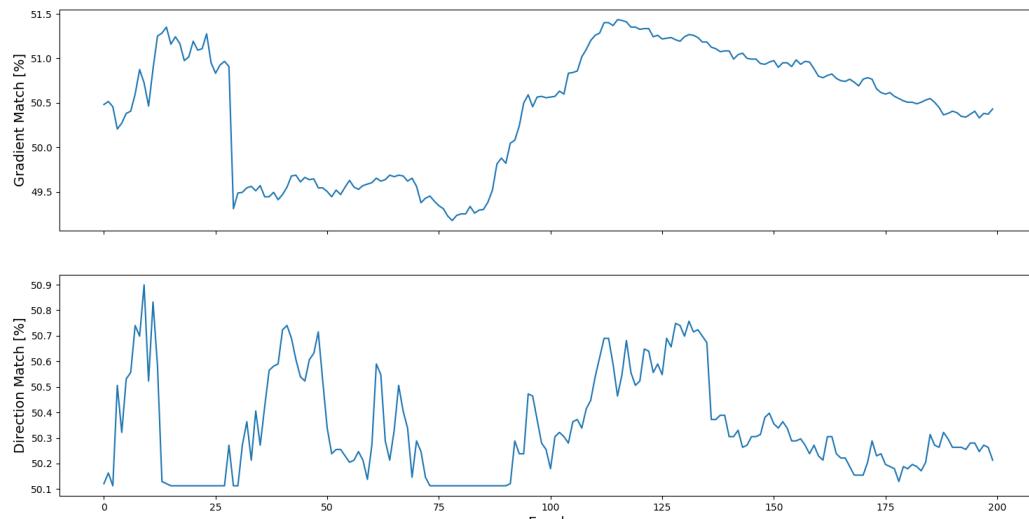


Abbildung 30: Korrekte Richtungsbestimmung für ein Modell auf dem Kurs EUR/USD (höher ist besser).

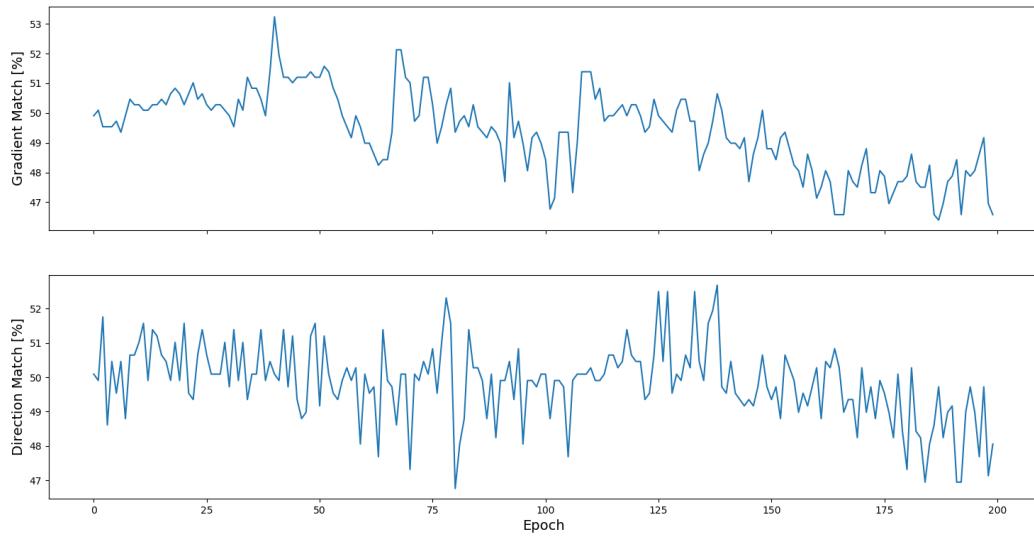


Abbildung 31: Korrekte Richtungsbestimmung für ein Modell auf dem Kurs EUR/USD (höher ist besser).

Aus diesem Grid Search lassen sich also keine relevanten Informationen gewinnen, da alle Tests eine Vorhersage erzielen, die nicht besser als der Zufall ist. Dass das Modell mit der kleinsten Skalierung den tiefsten Loss aufweist, kann auch damit zusammenhängen, dass mehr Datenpunkte zur Verfügung stehen. So konnte das Modell über einen grösseren Zeitraum trainieren.

9.3 GRID SEARCH MIT DEM CUSTOM LOSS – NEUER ANSATZ

9.3.1 KORRELATION - DATENAUSWAHL

Es werden nochmals neue Daten ausgewählt, dieses Mal nur mit Schweizer Kursen. Es wird ein allgemein hoher Zusammenhang der Kurse auf demselben Finanzplatz vermutet. Als Beispiel dient der Kurs der Bank UBS mit folgenden Kreuzkorrelationen.

Kurs	Spearman-Index	Shift
CSGNCHCHF	0.9443453594349751	1
ABBNCHCHF	0.9032154056937381	-1
ADENCHCHF	0.8846942104789727	1
SIKCHCHF	0.8581192311505742	-1
BAERCHCHF	0.8466257280451086	-1
KNINCHCHF	0.8460319626616858	0
LHNCHCHF	0.8353792704842449	-2
UHRCHCHF	0.833686513385184	0
CLNCHCHF	0.7962531174015793	-6
SGSNCHCHF	0.7008199618041611	-3
SLHNCHCHF	0.6278929551696457	48
NOVNCHCHF	0.6202966344815787	48
ROGCHCHF	0.6127948169060348	48
NESNCHCHF	0.5674934413652624	41
SOONCHCHF	0.5566277726706232	48
GIVNCHCHF	0.4363397477181433	41
SCMNCHCHF	0.41477938586657337	-48
SRENCHCHF	0.13526303515720578	-48
ZURNCHCHF	0.11759767853069114	1
LONNCHCHF	0.07542487104860106	48

Auszug aus den Ergebnissen der Kreuzkorrelation Schweizer Kurse.

Gewählt werden Kurse aus der gleichen oder ähnlichen Branchen mit hohen Korrelationskoeffizienten (gelb markiert).

9.3.2 PARAMETRISIERUNG

Die Parametrisierung wird ohne Berücksichtigung vorhergehender Grid Search Resultate erstellt, da keines der vorhergehenden Modelle den gewünschten Erfolg erzielt.

Parameter	Konfigurationen
selectedSymbols	["UBSGCHCHF_EMA4", "UBSGCHCHF_BB4", "CSGNCHCHF_EMA4", "BAERCHCHF_EMA4", "UHRCHCHF_EMA4"]
absoluteMass	[1,0,0.2]
startdate	['2018-05-01']
scale	['1h', '4h', '1d']
adamLR	[0.0001, 0.00001, 0.000001]
adamBeta_1	[0.9]
adamBeta_2	[0.999]
adamEpsilon	[None]
adamDecay	[0.0]
adamAmsgrad	[False]
Activation	['relu']
trainsetSize	[0.8]
predictionLength	[4]
blockSize	[4]
numNodes	[[600, 600, 600]]
dropout	[0.15, 0.25, 0.35]
numberOfEpochs	[1000]
batchSize	[96]
shuffle	[False]
specialTimeWindows	[[12, 24, 36, 48, 60, 72], []]

Parametrisierung des Grid Search mit Schweizer Kursen.

Die geänderten Parameter (gelb markiert) basieren auf folgender Überlegung:

Parameter	Überlegung
absoluteMass	Die beiden Loss Funktionen (Absolutwertabweichung und Winkelabweichung) werden unterschiedlich gewichtet. So kann evaluiert werden, welche der beiden einen besseren Lerneffekt erzielen.
scale	Die unterschiedliche Skalierung der Kursdaten kann Ausschluss darüber geben, in welcher Zeitspanne das Netzwerk am besten Vorhersagen erstellt.
adamLR	Der Startwert der LR wird unterschiedlich gewählt, um eine flache Lernkurve zu erzwingen. Dies soll starke Schwankungen im Loss verhindern.
dropout	Durch die neue Anzahl Layer und Nodes ist unklar, welcher Dropout am besten passt.
specialTimeWindows	Dies hilft auf vergangene Kursdaten zuzugreifen. Mit der unterschiedlichen Parametrisierung kann so evaluiert werden, ob dies hilft.

Überlegungen zur Parametrisierung des Grid Search mit Schweizer Kursen.

9.3.3 RESULTATE

Im Tensorboard fallen zwei Arten von Modellen auf.

- Jene mit einer flachen Lernkurve, diese erreichen einen konstant tiefen Validation Loss.
- Jene mit schwankender Lernkurve, diese erreichen im Vergleich den tiefsten Loss, weisen jedoch starke Schwankungen auf.

Die besten Modelle der beiden Kategorien sind folgende:

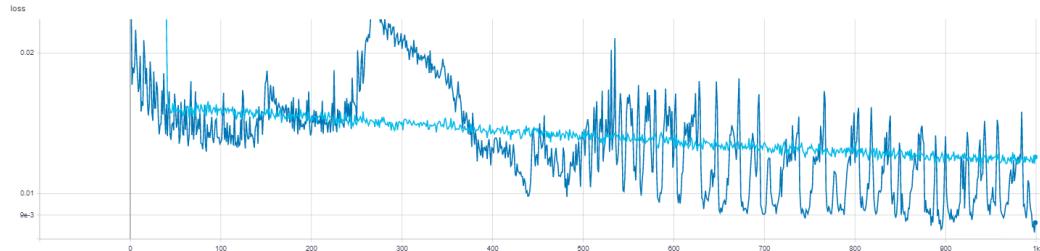


Abbildung 32: Loss aus dem Tensorboard zweier Modelle unterschiedlicher Charakteristik mit dem besten Validation Loss (tiefer ist besser)

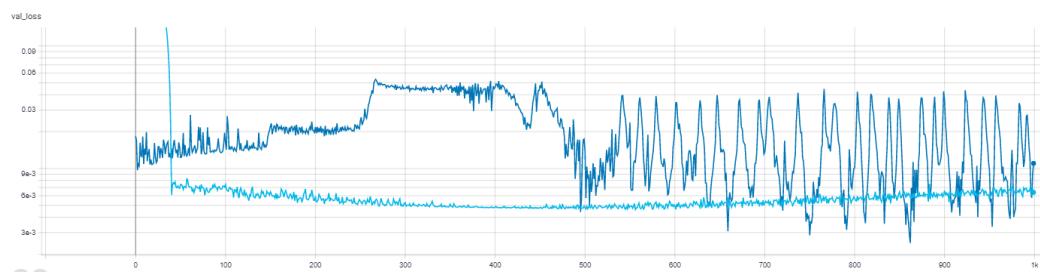


Abbildung 33: Validation Loss aus dem Tensorboard zweier Modelle unterschiedlicher Charakteristik mit dem besten Validation Loss (tiefer ist besser)

Dabei weist das Modell mit flacher Lernkurve folgende Auswertungsfunktion vor:

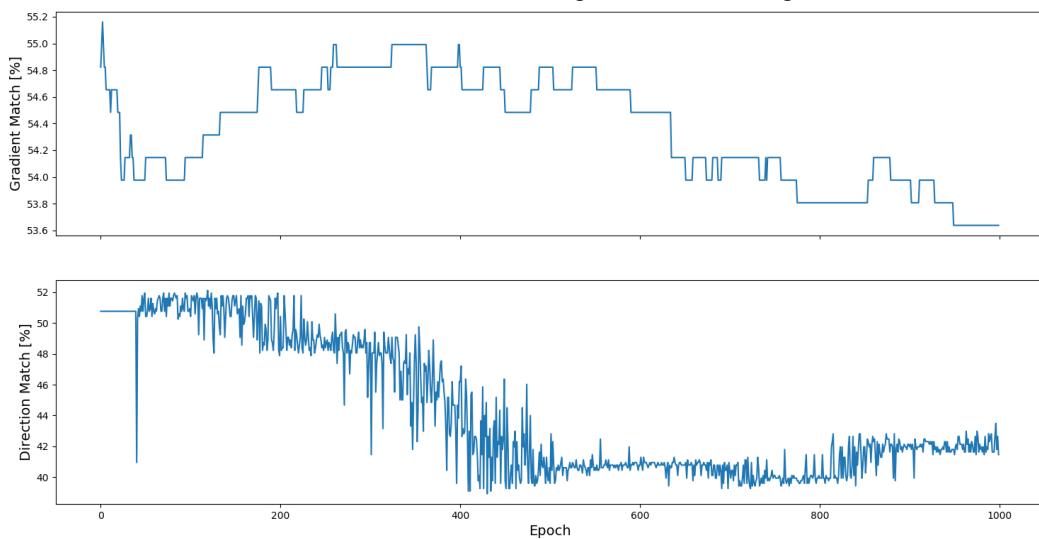


Abbildung 34: Auswertungsfunktion des Modells mit der besten schwankenden Lernkurve auf dem Schweizer Aktienmarkt

Das Modell mit flacher Lernkurve folgende:

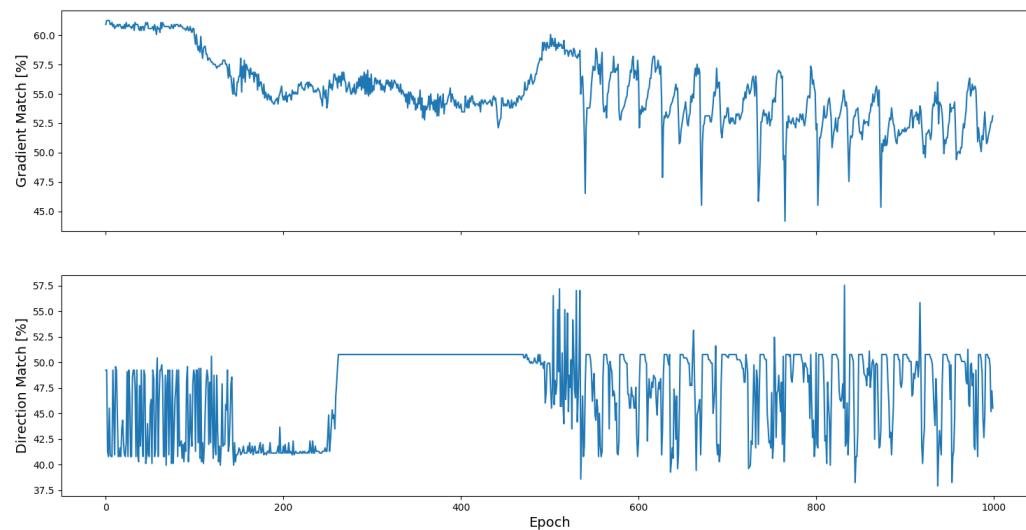


Abbildung 35: Auswertungsfunktion des Modells mit der besten flachen Lernkurve auf dem Schweizer Aktienmarkt

Auch bei diesem Grid Search gibt es somit kein Modell, welches eine herausragende Vorhersage der Kursrichtung erreichte. Präferenzen in der Parametrisierung können aufgrund der Zufälligen Vorhersage keine gefällt werden.

9.4 GRID SEARCH MIT DER CLASSIFICATION METHODE

9.4.1 GRID SEARCH PARAMETRISIERUNG

Als Beispiel eines Grid Search mit der Klassifizierung-Methode wurde derselbe Test wie im Kapitel 9.1 mit der Siemens Aktie durchgeführt.

Dies ist die Konfiguration für den Grid Search mit drei Klassen.

Parameter	Konfigurationen
selectedSymbols	[["SIEDEEUR_SMA8", "SIEDEEUR-Volume", "ESPIDXEUR_SMA8", "EUSIDXEUR_SMA8", "DEUIDXEUR_SMA8", "HEIDEEUR_SMA8"]]
startdate	['2018-05-01']
scale	['15min']
adamLR	[0.001]
adamBeta_1	[0.9]
adamBeta_2	[0.999]
adamEpsilon	[None]
adamDecay	[0.0]
adamAmsgrad	[True]
activation	['relu']
trainsetSize	[0.8]

<code>predictionLength</code>	[12, 24]
<code>blockSize</code>	[2*4*24]
<code>numNodes</code>	[[1200]]
<code>dropout</code>	[0.2, 0.4]
<code>numberOfEpochs</code>	[50]
<code>batchSize</code>	[4*24*5]
<code>shuffle</code>	[False]
<code>numCategories</code>	[3]

Parametrisierung des Grid Searches mit der Classification Methode.

9.4.2 AUSWERTUNG

Das Modell, welches am besten abgeschlossen hat, beinhaltet folgende Parameter:

Parameter	Konfigurationen
<code>predictionLength</code>	24 # 24*15min = 6h
<code>dropout</code>	0.2 # 20% dropout

Die beste Parameterwahl für den vorgehenden Grid Search.

Die Kursrichtung kann mit diesem Modell in **55.29 %** aller Samples korrekt vorhergesagt werden.

Die dazugehörige Confusion Matrix sieht wie folgt aus:

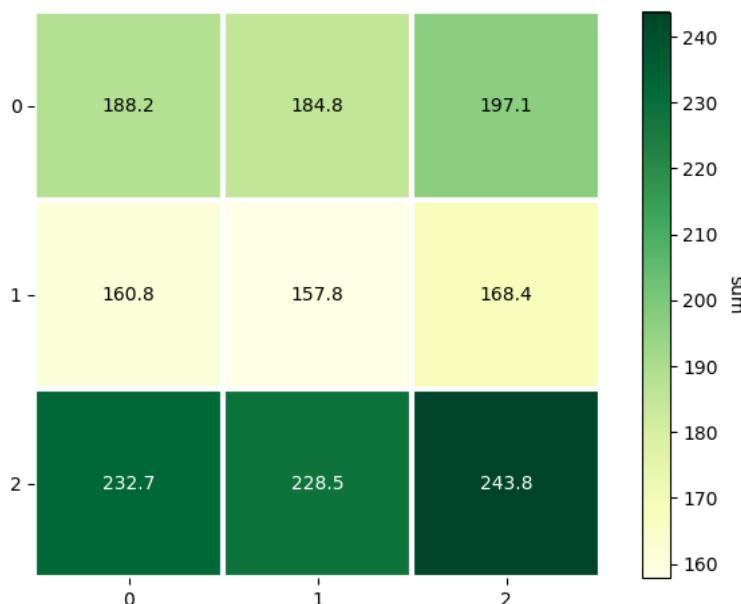


Abbildung 36: Confusion Matrix - Siemens Aktie Gridsearch (Punkte auf der diagonalen Achse von oben links sind besser)

Die Confusion Matrix zeigt, dass die Klassifizierung des Aktienkurses nicht wie gewünscht ausgefallen ist. Das Netzwerk setzt sehr oft auf Klasse 2 (fallender Kurs), auch wenn der Kurs steigt. Beim Training ist auffällig, dass nach wenigen Epochen keine grossen Änderungen im mehr stattfinden im Loss. Dieser Effekt konnte auch bei allen anderen Test mit dieser Methode nachgewiesen werden.

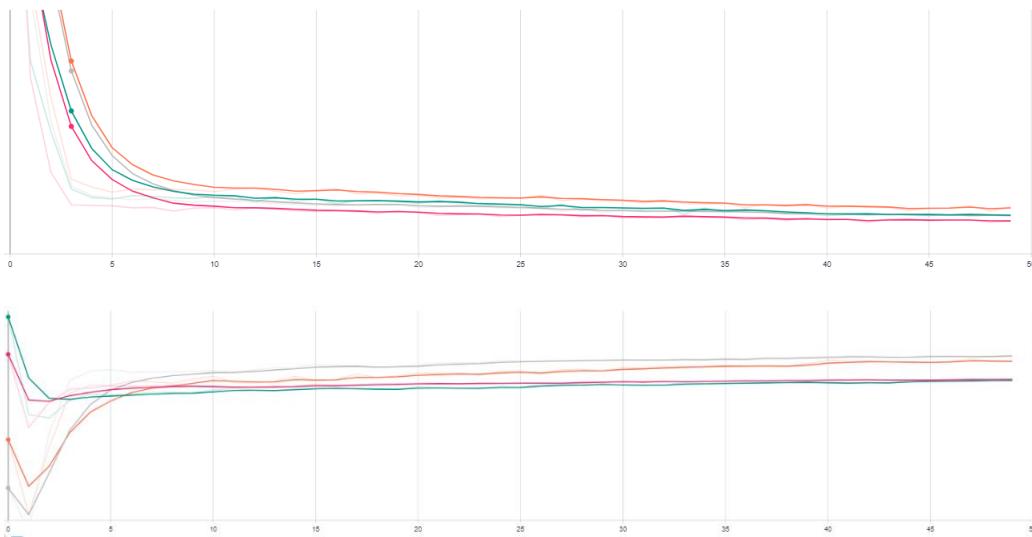


Abbildung 37: Loss (oben), Validation Loss (unten) - Siemens Classification Gridsearch

Bereits nach 10 Epochen hat der Loss sein Minimum erreicht und ändert sich nicht mehr signifikant, was zur Vorhersage einer anderen Klasse führen würde. Der Validation Loss verbessert sich über die gesamte Trainingsphase überhaupt nicht.

9.4.3 DEBUGGING MIT SINUS

Aufgrund der schlechten Ergebnisse wird die Classification Methode mittels Dummykurs evaluiert.

Um die gesamte Pipeline der Klassifizierung zu testen, wird eine Sinuskurve generiert und als CSV-Datei gespeichert. Diese Datei kann nun als Input in der Konfiguration angegeben werden. Als Test werden im Input auch noch weitere Kurse mitgegeben, um zu belegen, dass dies auf die Vorhersage keinen negativen Einfluss hat.

Die Parametrisierung wird dabei wie folgt festgelegt:

Parameter	Konfigurationen
activation	sigmoid
adamBeta_1	0.9
adamBeta_2	0.999
adamDecay	0.0
adamEpsilon	1e-07
adamLR	0.01
amsgrad	False
batchSize	1008
blockSize	24

checkOverValue	0.3
debug	False
dropout	0.01
name	Cluster_Sinus_Oil_US_Test
numCategories	3
numNodes	[200, 200, 100, 50]
numberOfEpochs	10
predictionLength	3
scale	10min
selectedSymbols	["SINUS", "BRENTCMDUSD", "APAUSUSD", "APCUSUSD"]
shuffleInput	False
trainsetSize	0.8
useGPU	True

Parametrisierung für den Debugging-Versuch mit einer Sinuskurve.

Wie in folgender Confusion Matrix ersichtlich, kann das Model die Sinuskurve nach 10 Epochen perfekt klassifizieren.

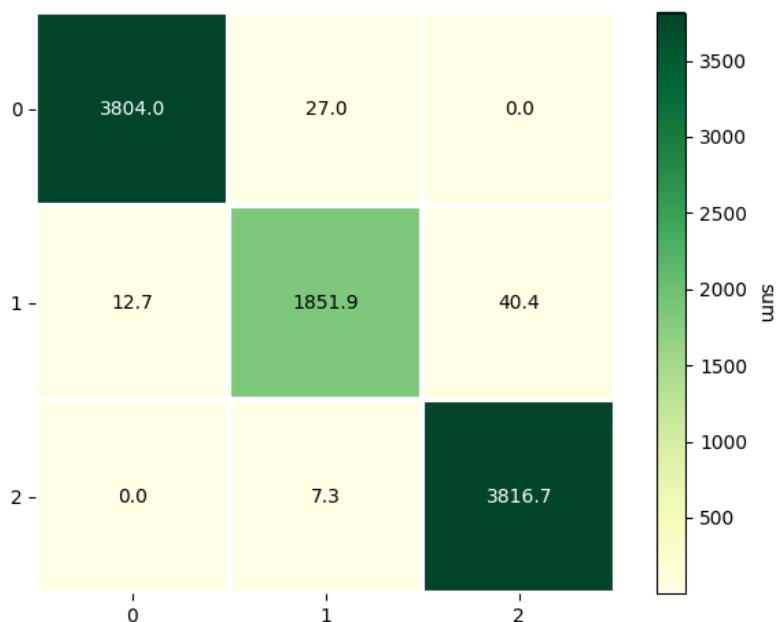


Abbildung 38: Confusion Matrix für die Vorhersage eines Sinus mit drei Klassen/Steigungsrichtungen
(Punkte auf der diagonalen Achse von oben links sind besser)

Die gleiche Vorhersage mit fünf Klassen funktioniert ebenfalls wie erwartet:



Abbildung 39: Confusion Matrix für die Klassifizierung eines Sinus mit fünf Klassen/Steigungsrichtungen (Punkte auf der diagonalen Achse von oben links sind besser)

Dies ist der Beweis, dass die gesamte Pipeline funktioniert. Die einzelnen Funktionen sind zusätzlich zu diesem Versuch auch separat getestet und als korrekt empfunden worden.

10 FAZIT

So zufällig die Börse zu sein scheint, so zufällig fallen meist auch die Vorhersagen aus. Und doch gibt es einige Vorhersagen, welche erstaunlich gute Resultate liefern. Nur eben sind diese nicht allgemein gültig. Die Frage, ob sich nun die Börse mittels LSTM vorhersagen lässt, kann wie folgt beantwortet: Die Vorhersage ist besser als der Zufall und doch nicht gut genug, um sich blind darauf zu verlassen. Oft kommt es vor, dass die Vorhersage des Netzwerkes eine Art Wertung des Kurses ist, und ob mit grösseren Schwankungen zu rechnen ist (siehe Abbildung 26, Timestamp 450-500). Trotzdem sollte man sich bei Kaufentscheidungen nicht nur auf die Vorhersage des Netzwerkes verlassen, sondern auch andere Wirtschaftliche Aspekte und Indikatoren betrachten. Im Grossen und Ganzen kann man jedoch durchaus sagen, dass Potential besteht, LSTM Netzwerke als Indikatoren für Kaufentscheidungen im Finanzmarkt einzusetzen.

10.1 ANWENDUNGSBEREICH DER RESULTATE DIESER ARBEIT

Man kann durchaus in Betracht ziehen, eine Vorhersage des Kurses dieser Arbeit als weiteren Indikator für andere Tätigkeiten an der Börse einzusetzen. Ohne weitere Indikatoren oder Strategien sollten damit aber keine Kaufentscheide gefällt werden.

10.2 KOMMENTAR DER AUTOREN

Gerne hätten wir noch mehr Zeit in diese Arbeit investiert. Man kann gut erkennen in welche Richtung sich diese entwickelt und welches Potential sich dahinter verbirgt. Die Zeit, in welcher die Arbeit durchgeführt wurde, hat gezeigt, wie endlos viele Optionen und Variationen für die Vorhersage zur Verfügung stehen. Das trainieren Neuronaler Netze ist auch mit aktueller Hardware noch extrem Zeitintensiv und es können nicht unbegrenzt grosse Grid Searches durchgeführt werden. Wir glauben, noch mehr aus den Daten der Kurse lesen zu können, als dies aktuelle Grid Searches getan haben.

11 MÖGLICHES WEITERES VORGEHEN

Während der Projektarbeit entstanden neue Ideen und Konzepte, welche im Rahmen dieser Bachelorarbeit nicht mehr durchgeführt und überprüft werden konnten.

11.1 WEITERE INDIKATOREN

Für Zeitreihen gibt es unzählige, technische Indikatoren, welche bei der Vorhersage helfen können. Es kann durchaus helfen, möglichst viele zu sammeln und ein Grid Search nur mit verschiedenen Indikatoren und deren Parameter aufzubauen. Bereits einfache Indikatoren wie der Moving-Average konnten zu einer besseren Vorhersage führen. Die Finanzwelt bietet hier diverse Indikatoren an. Da das Entwickelte Programm sehr modular aufgebaut ist, könnte es relativ einfach mit weiteren Indikatoren Ergänzt werden und automatische Grid Searches durchführen.

11.2 KURZFRISTIGE VORHERSAGE

Die Kreuzevaluation in Kapitel 9.1.4 lässt die Vermutung offen, dass eine Vorhersage nur kurzfristig gültig ist und sich der Kurs nach kürzester Zeit nicht mehr an die gelernten Regeln hält. Ein möglicher Ansatz könnte folgender sein: Nach jedem vorhergesagten Datenpunkt der Testdaten wird das Netzwerk mit dem neuen Datenpunkt erneut trainiert. Dieses neu trainierte Netzwerk kann wiederum den nächsten Punkt vorhersagen usw. Dieser Ansatz wird auch Online Learning genannt.

11.3 NEUER ANSATZ - GAUSSIAN PROCESS REGRESSION (GPR)

Neben LSTM Netzwerken gibt es auch andere Ansätze zur Vorhersage. Eine dieser Möglichkeiten ist die Vorhersage mittels Gaussian Process Regression (GPR). Diese Methode lässt sich direkt auf einen Kursverlauf anwenden, oder auch in Kombination mit einem neuralen Netzwerk als zusätzlichen Indikator einsetzen.

Die Vorhersage mittels GPR implementiert Gaussian Processes (GP) und basiert auf stochastischer Vorhersage. Dies erlaubt die Vorhersage von Datenpunkten auf einer Zeitachse und zeigt den wahrscheinlichsten Verlauf des Kurses. Diese gibt anschliessend Auskunft über den gemittelten, geschätzten Kurswert, sowie der Streuung oder dem Rauschen (scikit-learn.org, 2019).

Ausreisser im Handelsverlauf des Kurses können so problemlos verarbeitet werden. Ein Rauschen auf dem Kurswert bringt hiermit keine Nachteile mit sich.

11.3.1 REALISIERUNG

Dieses Vorgehen lässt sich mittels Python und der Bibliothek Sklearn realisieren.

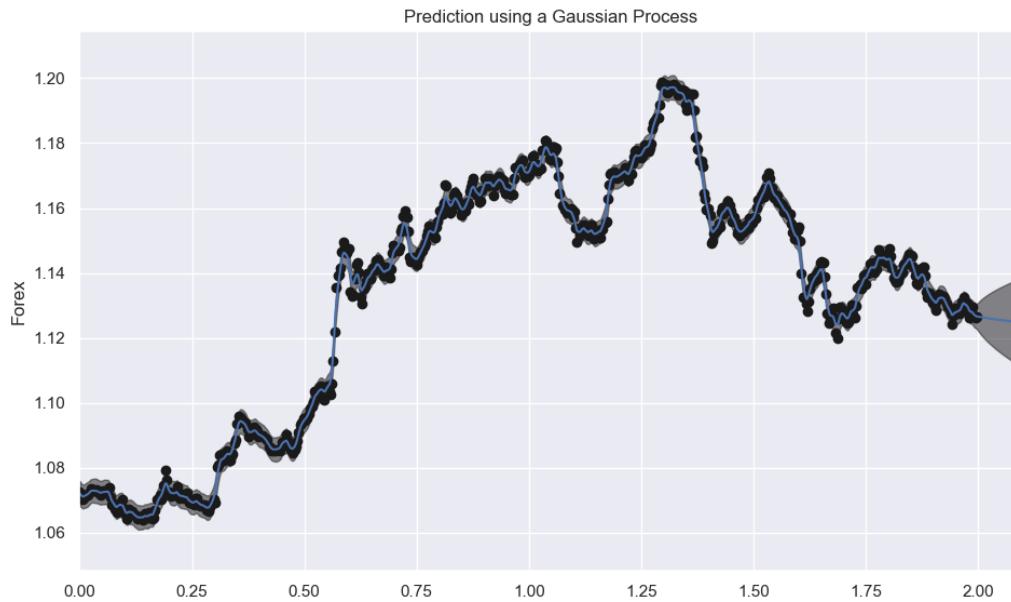


Abbildung 40: Vorhersage des Kurses EUR/CHF auf Stundenbasis mittels GPR Methode. Zu sehen ist die Vorhersage ab 2.0 auf der X-Achse. Der graue Bereich zeigt die Streuung an.

In den nächsten Schritten gilt es, eine optimale Parametrisierung zu finden, die Vorhersage automatisch zu überprüfen und zusätzliche Kurse zum Vergleich zu implementieren. Dies wird in dieser Arbeit jedoch nicht weiter behandelt.

12 TIPPS FÜR ZUKÜNTIGE ARBEITEN

12.1 GRÖSSTE PROBLEME

12.1.1 DATENQUELLE

Ein grosses Problem im Zusammenhang mit dieser Arbeit war die Beschaffung der Kursdaten. Eine Quelle zu finden ist eine Sache, jedoch bieten diese meist eine schlechte Auflösung oder gar Lücken in den Daten.

12.1.2 KERAS

Keras bietet ein einfacheres Interface für Tensorflow. Das mag hilfreich erscheinen, beschränkt jedoch oftmals die Modularität und Anpassungsfähigkeit der Netzwerke. Im Verlauf der Arbeit wurde ausserdem Tensorflow 2.0 veröffentlicht. Hierbei ist Keras ohnehin ein direkter Bestandteil von Tensorflow. Für weitere Arbeiten sollte also direkt Tensorflow 2.0 verwendet werden.

12.1.3 MACHINE LEARNING ALLGEMEIN

Machine Learning mit Tensorflow steckt noch immer in den Kinderschuhen (stand 2019). Das macht sich dadurch bemerkbar, dass es einerseits an Lösungsansätzen für Problemstellungen mangelt und andererseits fortlaufend Updates erscheinen, welche eventuell eine Umstrukturierung der Arbeit erfordern.

12.2 EMPFOHLENES VORGEHEN BEI DER ENTWICKLUNG MIT ML

Im Zusammenhang mit Keras gibt es einige Tricks, welche angewendet werden können, sollte das Netzwerk die Zusammenhänge der Input-Daten nicht richtig interpretieren. Dazu gibt es verschiedene Ansätze und Grundprobleme welche hier beschrieben sind.

12.2.1 DAS NETZWERK LERNT AUSWENDIG, DER VALIDATION LOSS STEIGT

In jedem Training hat man irgendwann den Punkt erreicht, in welchem das Netzwerk nur noch schleppend dazulernen kann. Das Problem bei mehreren Epochen ist jedoch, dass sich die Nodes mit der Zeitachse verknüpfen, so dass das Netzwerk beginnt den Inputkurs auswendig zu lernen.

Um dies zu verhindern muss der Dropout richtig gewählt werden. Dieser löscht den Output einiger Nodes im Netzwerk. So können über mehrere Epochen gelernte Kursausschnitte wieder zurückgesetzt werden. Ein Dropout Layer kann entweder zwischen jeder Schicht oder am Ende des Netzwerkes eingesetzt werden. Bei der Vorhersage von Aktienkursen lieferte das Einsetzen von Dropout Schichten zwischen jedem Layer präzisere Vorhersagen.

Der richtige Dropout muss jedoch für jeden Fall selbst ermittelt werden (no free lunch).

12.2.2 DER VALIDATION LOSS STAGNIERT NACH WENIGEN EPOCHEN

Hierbei wurde der Optimizer etwas zu optimistisch eingestellt. Grundsätzlich hilft es dabei die LearningRate höher zu setzen. Im Zusammenhang mit dem Adam Optimizer, welcher die LearningRate ohnehin eigenständig korrigiert, müssen andere Parameter angepasst werden. (siehe Anhang Adam-Optimizer)

12.2.3 DAS NETZWERK BRAUCHT UNERWARTET LANGE FÜR DAS TRAINING

Das liegt in der Regel daran, dass nicht die volle Hardwareleistung ausgereizt wird.

- Das Training auf der Grafikkarte beschleunigt den Vorgang massiv. Besonders der Einsatz der CuDNNLSTM Zellen anstatt der normalen LSTM Zellen haben einen signifikanten Unterschied ausgemacht.
- Die Batchsize kann je nach Art der Daten vergrößert werden, so werden die Gewichte nicht nach jedem Sample angepasst. Solange dies nicht den Speicher der Grafikkarte und Co. zum Überlaufen bringt, beschleunigt dies den Trainingsvorgang massiv.

13 ANHANG

1. Slack
2. Docker
3. Multiprocessing in Python
4. Adam-Optimizer
5. Korrelationskarte und Kreuzkorrelationen zu Siemens

14 ABBILDUNGSVERZEICHNISS

Abbildung 1: Auszug aus der Korrelationskarte aller Kurse (unvollständig)	11
Abbildung 2: Korrelationskarte der Schweizer Kurse von Dukascopy (vollständig, rein quantitativ)	11
Abbildung 3: Handelszeiten (quantitative Grafik)	12
Abbildung 4: Öffnungszeiten der Schweizer Börse (www.six-group.com , 2019).....	12
Abbildung 5: Gültiger Handelszeitraum europäischer Kurse (weisse Flächen sind gültig, rein quantitative Grafik)	13
Abbildung 6: Slicing Beispiel, visualisiert die Verpackung der Daten (von oben nach unten gelesen). Die Datenmenge visualisiert die horizontale Achse (von links nach rechts gelesen).	14
Abbildung 7: Gliederung Softwarearchitektur. Gestartet wird mit einer Konfiguration (Grid Search / Config). Die Datei LSTM ist der Knotenpunkt, welcher das Netzwerk aufbaut und Hilfsbibliotheken verwendet (mitte). Die Datenquellen (rechts) werden via Collector geladen.....	16
Abbildung 8: Aufbau einer LSTM Zelle, $x_{(t)}$ =Eingabevektor, $y_{(t)}$ =Ausgabevektor, $h_{(t)}$ =Kurzzeitgedächtnis, $c_{(t)}$ =Langzeitgedächtnis, t ist jeweils der aktuelle Zeitschritt, (Géron, Machine Learning mit Scikit-Learn & Tensorflow, 2018)	21
Abbildung 9: Regression Workflow	23
Abbildung 10: Regression Vorhersage Plot, vergleicht reellen Kurs und Vorhersage ...	23
Abbildung 11: Visualisierung der Absolute Loss Funktion. Diese Setzt sich aus der Abweichung des Kursverlaufs und der Vorhersage zusammen (kleinere Abweichung ist besser).....	25
Abbildung 12: Screenshot Tensorboard. Visualisierung der Loss-Funktion von drei verschiedenen Modellen (tiefer ist besser).....	26
Abbildung 13: Evaluation der Vorhersage (zeigt die relevanten Veränderungen)	26
Abbildung 14: Plot Log-Return (blau) von ABB Aktienkurs und Handelsvolumen (orange)	28
Abbildung 15: Visualisierung der ShiftY Funktion. Jede Output Sequenz wird so verschoben, dass sie auf der Nulllinie beginnt.	29
Abbildung 16: Output Klassifizierung LSTM-Netzwerk mit fünf Klassen.....	29
Abbildung 17: Visualisierung der Classification Loss Funktion. Im Vergleich stehen der Vorhersage Vektor (Prediction) und der Vektor zur korrekten Kursrichtung (Truth).	31
Abbildung 18: Confusion Matrix mit drei Klassen (Punkte auf der diagonalen von links oben sind besser). Beispiel von einem Grid Search mit dem SMI-Kurs.	32
Abbildung 19: Custom-Loss Netzwerkarchitektur	33
Abbildung 20: Grid Search Loss der Trainingsdaten (Logarithmisch). Screenshot aus dem Tensorboard.	35
Abbildung 21: Grid Search Validation Loss aus den Testdaten (Logarithmisch). Screenshot aus dem Tensorboard.	36
Abbildung 22: Grid Search eines fehlgeschlagenen Modells. Blaue Linien zeigen die falsche Vorhersage des Kurses, hier stark fallend.....	36

Abbildung 23: Grid Search, Vorhersagegenauigkeit nach eigener Evaluierungsfunktion	37
Abbildung 24: Grid Search Result, Loss der Trainingsdaten	37
Abbildung 25: Grid Search Resultate des Validation Loss aus den Testdaten.....	37
Abbildung 26: Grid Search Results, Vorhersage des besten Modells	38
Abbildung 27: Loss aus dem Tensorboard für einen Grid Search mit unterschiedlichen Zeitskalierungen des Kurses EUR/USD, logarithmische Darstellung (tiefer ist besser).	40
Abbildung 28: Validation Loss aus dem Tensorboard für einen Grid Search mit unterschiedlichen Zeitskalierungen des Kurses EUR/USD, logarithmische Darstellung (tiefer ist besser).....	40
Abbildung 29: Korrekte Richtungsbestimmung für das Modell mit dem tiefstem Loss auf dem Kurs EUR/USD (höher ist besser).....	40
Abbildung 30: Korrekte Richtungsbestimmung für ein Modell auf dem Kurs EUR/USD (höher ist besser).....	41
Abbildung 31: Korrekte Richtungsbestimmung für ein Modell auf dem Kurs EUR/USD (höher ist besser).....	41
Abbildung 32: Loss aus dem Tensorboard zweier Modelle unterschiedlicher Charakteristik mit dem besten Validation Loss (tiefer ist besser)	44
Abbildung 33: Validation Loss aus dem Tensorboard zweier Modelle unterschiedlicher Charakteristik mit dem besten Validation Loss (tiefer ist besser)	44
Abbildung 34: Auswertungsfunktion des Modells mit der besten schwankenden Lernkurve auf dem Schweizer Aktienmarkt.....	44
Abbildung 35: Auswertungsfunktion des Modells mit der besten flachen Lernkurve auf dem Schweizer Aktienmarkt.....	45
Abbildung 36: Confusion Matrix - Siemens Aktie Gridsearch (Punkte auf der diagonalen Achse von oben links sind besser)	46
Abbildung 37: Loss (oben), Validation Loss (unten) - Siemens Classification Gridsearch	47
Abbildung 38: Confusion Matrix für die Vorhersage eines Sinus mit drei Klassen/Steigungsrichtungen (Punkte auf der diagonalen Achse von oben links sind besser).....	48
Abbildung 39: Confusion Matrix für die Klassifizierung eines Sinus mit fünf Klassen/Steigungsrichtungen (Punkte auf der diagonalen Achse von oben links sind besser).....	49
Abbildung 40: Vorhersage des Kurses EUR/CHF auf Stundenbasis mittels GPR Methode. Zu sehen ist die Vorhersage ab 2.0 auf der X-Achse. Der graue Bereich zeigt die Streuung an.	52

15 LITERATURVERZEICHNIS

alphavantage. (01. 09 2019). Von <https://www.alphavantage.co/> abgerufen

Areej Abdullah Baasher, M. W. (2011). *FOREX Trend Classification using Machine Learning Techniques*. Cairo, Egypt: Arab Academy for Science and Technology.

Cavalcante, R. C., Brasileiro, R. C., Souza, V. L., Nobrega, J. P., & Oliveira, A. L. (2016). *Computational Intelligence and Financial Markets: A Survey and Future Directions*. Elsevier.

Dukascopy. (09. 01 2019). Von <https://www.dukascopy.com/swiss/deutsch/marketwatch/historical/> abgerufen

Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. Sebastopol,: O'Reilly Media, Inc.

Géron, A. (2018). *Machine Learning mit Scikit-Learn & Tensorflow*. 69123 Heidelberg: dpunkt.verlag GmbH.

pandas.pydata.org. (01. 08 2019). Von <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html> abgerufen

quandl. (09. 01 2019). Von <https://www.quandl.com/> abgerufen

scikit-learn.org. (7. 8 2019). Von https://scikit-learn.org/stable/modules/gaussian_process.html abgerufen

six-group. (3. Januar 2019). Von https://www.six-group.com/exchanges/index_de.html abgerufen

worldtradingdata. (09. 01 2019). Von <https://www.worldtradingdata.com/> abgerufen

www.six-group.com. (12. 07 2019). Von https://www.six-group.com/exchanges/exchange_traded_products/trading/trading_hours_en.html abgerufen

www.tradingtechnologies.com. (11. 07 2019). Von <https://www.tradingtechnologies.com/xtrader-help/x-study/technical-indicator-definitions/list-of-technical-indicators/> abgerufen

SLACK

1 EINLEITUNG

Slack ist eine Webapplikation, welche die Kommunikation im Team vereinfacht. Nebst direkter Kommunikation zwischen Benutzern, können auch sogenannte Channels erstellt werden. Die Benutzer können diesen Channels beitreten und sie bei Bedarf auch wieder verlassen. Jeder Channel widerspiegelt ein Thema wie beispielsweise "Todo's" oder "Meetings". Nebst einfachem Text, können auch Bilder und Videos geteilt werden. Slack ist also ein zentraler Ort für die Zusammenarbeit in einem Projekt und dank Apps auf allen gängigen Betriebssystemen und Geräten verfügbar.

2 SLACK-API

Slack bietet zudem eine offene API. Somit können Programme auch direct mit Slack kommunizieren. Zum dokumentieren unserer Trainingsvortschrifte haben wir eine Funktion geschrieben, welche die gesamte Konfiguration des Tests mit den Resultaten auf Slack postet. Es gibt dazu ein passendes PIP-Packet für Python namens "slackclient", das die Implementierung der Slack-API erheblich erleichtert.

```
1. from slackclient import SlackClient
2.
3. class SendSlack():
4.     sc = SlackClient("your-api-key")
5.
6.     @staticmethod
7.     def sendFile(filePath, fileTitle):
8.         with open(filePath, 'rb') as fileContent:
9.             SendSlack.sc.api_call(
10.                 "files.upload",
11.                 channels='the-neural-net',
12.                 file=fileContent,
13.                 title=fileTitle,
14.             )
15.
16.     @staticmethod
17.     def sendText(text_content):
18.         SendSlack.sc.api_call(
19.             "chat.postMessage",
20.             channel="the-neural-net",
21.             text=text_content
22.         )
```

Für unseren Slack-Bot wurde ein separater Channel Namens "the-neural-net" erstellt. Die Tests werden somit von unserem Programm automatisch in der Slack Timeline dokumentiert und wir konnten die Resultate sofort und überall online abrufen.

3 DEMO UND FAZIT

Slack hat uns die Arbeit im Team sehr erleichtert und wir würden es für ein zukünftiges Projekt sofort wieder einsetzen. Auch der Mehraufwand mit dem Slack-Bot hat sich gelohnt. Das Dokumentieren der Tests hat unser Programm somit beinahe selber übernommen.

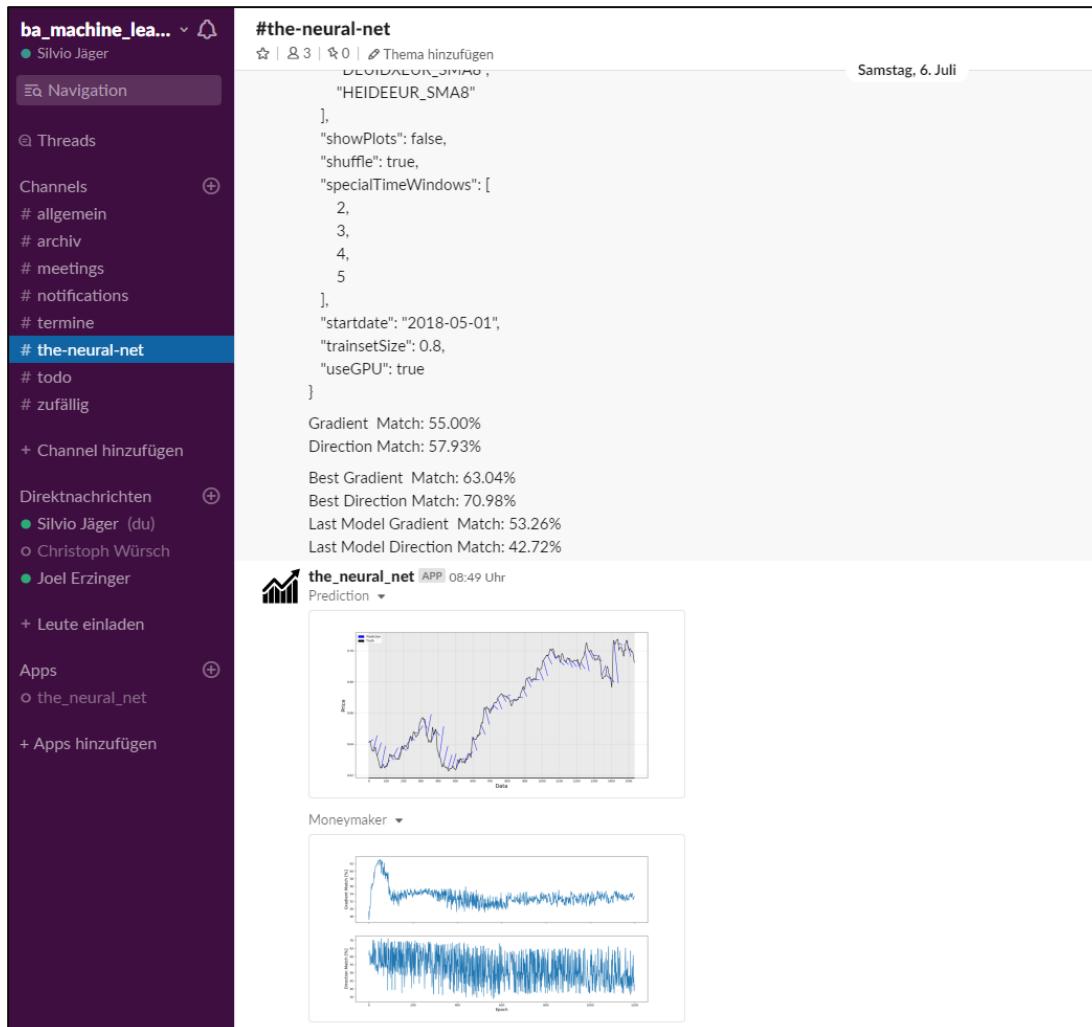


Abbildung 1: Ausschnitt eines Posts per Slack API

4 ABBILDUNGSVERZEICHNIS

Abbildung 1: Ausschnitt eines Posts per Slack API..... 2

5 LITERATURVERZEICHNIS

slack.com. (7. August 2019). Von <https://slack.com/intl/de-ch/> abgerufen

DOCKER

1 EINLEITUNG

Für grössere Berechnungen auf externen Cluster-Computern ist das Thema Docker oftmals unausweichlich und kann erhebliche Mehrarbeit ersparen. Dieses Dokument soll einen Überblick verschaffen und den Einstieg in Docker erleichtern.

Im Cluster steht für dieses Beispiel eine Nvidia Grafikkarte zur Verfügung. Aus diesem Grund behandelt dieses Dokument nur die Verwendung von Nvidia Grafikkarten.

2 FUNKTIONSPRINZIP

Docker wird dazu eingesetzt, Applikationen in sogenannten Containern laufen zu lassen. Ein Container ist von allen anderen laufenden Prozessen auf dem Betriebssystem isoliert. Im Gegensatz zu einer virtuellen Maschine läuft jedoch nicht zwingend ein eigenes Betriebssystem in einem Container. Es wird kein Hypervisor wie z.B. VMWare oder VirtualBox benötigt. Somit ist Docker flexibel und lässt sich auf diversen Geräten und Betriebssystemen ausführen.

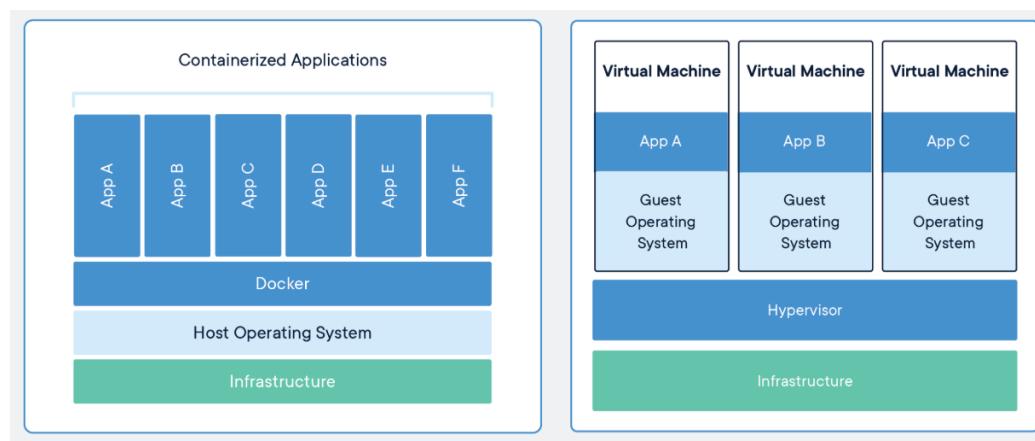


Abbildung 1: Unterschied der Architektur von Docker im Vergleich zu virtuellen Maschinen (Docker Website, 2019)

Ähnlich wie bei virtuellen Maschinen benötigt man auch für einen Docker Container ein Image. Jeder Container basiert auf einem eigenen Image und kann beliebig erweitert werden. Um selbst ein solches zu generieren, wird ein sogenanntes Dockerfile benötigt. Dieses ist ähnlich aufgebaut wie ein Linux Skript. Das Beispielskript basiert auf dem offiziellen "nvidia/cuda" Image von Nvidia. Wie im Dockerfile ersichtlich, wird zusätzlich Python 3.6 und alle benötigten PIP-Pakete installiert.

3 DOCKERFILE

```

1. # from https://hub.docker.com/r/nvidia/cuda/
2.
3. ARG cuda_version=9.0
4. ARG cudnn_version=7
5. ARG flavor=devel
6. FROM nvidia/cuda:${cuda_version}-cudnn${cudnn_version}-${flavor}-
  ubuntu16.04
7.
8. #update ubuntu stuff
9. RUN apt-get update
10.
11. # Install Python 3.6
12. RUN apt-get install -y software-properties-common
13. RUN add-apt-repository -y ppa:deadsnakes/ppa
14. RUN apt-get purge -y software-properties-common
15. RUN apt-get purge -y python3
16. RUN apt-get update
17. RUN apt-get install -y python3.6
18. RUN apt-get install -y python3-pip
19. RUN python3.6 -m pip install pip
20.
21. #Install graphviz, nano and git
22. RUN apt-get install -y graphviz git nano
23.
24. #Open Port for Tensorboard
25. EXPOSE 6006
26.
27. #Install all required pip packages
28. RUN python3.6 -m pip install pandas==0.24.1
29. RUN python3.6 -m pip install tensorflow-gpu==1.12.2
30. RUN python3.6 -m pip install matplotlib==3.0.2
31. RUN python3.6 -m pip install Keras==2.2.4
32. RUN python3.6 -m pip install sklearn==0.0
33. RUN python3.6 -m pip install ta==0.4.5
34. RUN python3.6 -m pip install slackclient==1.3.1
35. RUN python3.6 -m pip install tables==3.5.1

```

Das Dockerfile muss je nach verwendetem System angepasst werden. Vor allem die Version des Grafikkartentreibers ist entscheidend. Alle verfügbaren Versionen sind hier aufgelistet: <https://hub.docker.com/r/nvidia/cuda>

Falls die Möglichkeit besteht, einen neuen Grafikkartentreiber zu installieren empfiehlt es sich, den aktuellsten Treiber zu verwenden. Dadurch kann auch das aktuellste Image von Nvidia verwendet werden, was wiederum eine aktuelle Ubuntu Version enthält.

Dieses Dockerfile eignet sich für folgende Versionen:

Treiber nvidia-smi	384.81
Cuda Version	9.0
CUdnn Version	7.6.0

4 ABBILDUNGSVERZEICHNIS

Abbildung 1: Unterschied der Architektur von Docker im Vergleich zu virtuellen Maschinen (Docker Website, 2019).....	1
---	---

5 LITERATURVERZEICHNIS

Docker Website. (Juli 2019). Von <https://www.docker.com/> abgerufen

PYTHON MULTIPROCESSING

1 EINLEITUNG

Berechnungen in Python unter Windows können viel Zeit in Anspruch nehmen. Das muss aber nicht unbedingt die Regel sein. Oft liegt es einfach nur daran, dass nicht die gesamte Prozessorleistung verwendet wird. Dies lässt sich jedoch mit wenigen Anpassungen am Programm realisieren.

Python unterscheidet grundsätzlich zwischen Multithreading (mehrere Thread in einem Programm) und Multiprozessing (Aufteilung der Threads auf mehrere Prozessorkerne). Weiteres funktioniert leider nicht unter jedem OS gleich. Diese Anleitung beschäftigt sich mit Windows.

1.1 ANWENDUNGSBEREICH

Mit der Auslagerung der Threads auf mehrere oder alle Prozessorkerne kann die volle Leistung der Hardware für Berechnungen genutzt werden.

2 THEORIE

Beim erstellen eines neuen Prozesses gibt es folgende Konzepte:

Spawn	Der übergeordnete Prozess startet einen neuen Python-Interpreter-Prozess. Die Ressourcen werden vererbt. <i>Available on Unix and Windows</i>
Fork	Der übergeordnete Prozess teilt sich auf. Der untergeordnete Prozess, wenn er beginnt, ist praktisch identisch mit dem übergeordneten Prozess. <i>Available on Unix only</i>
Forkserver	Wenn ein neuer Prozess benötigt wird, verbindet sich der übergeordnete Prozess mit dem Forkserver und fordert ihn auf, einen neuen Prozess zu starten. <i>Available on Unix only</i>

Unter Windows können also nur neue Prozesse "spawnen". Das bedeutet, dass beim Erstellen eines neuen Prozesses jeweils alle Imports nochmals auf dem neuen Prozess durchgeführt werden.

3 PRAXISBEISPIELE

In der Praxis lässt sich Multiprozessing unter Berücksichtigung einiger Regeln ohne grossen Aufwand realisieren.

1. Kein Code ausserhalb von Klassen und Funktionen, alles muss verstaut werden. Andernfalls wird dieser Code beim "spawn" unnötigerweise nochmals ausgeführt.
2. Der Einstiegspunkt darf nur für den Hauptprozess greifbar sein. Dies lässt sich mit einer Abfrage realisieren: if `__name__ == '__main__'`:

3.1 PRAKTISCHE BEISPIELE

3.1.1 SHARED MEMORY

Unterprozess mit geteilten Objekten.

- Es können natürlich auch mehr als nur einen Prozess erstellt werden.
- In den Objekten können auch wieder eigene Objekte gespeichert werden usw.

```

1. from multiprocessing import Process, Value, Array
2.
3. def f(n, a):
4.     n.value = 3.1415927
5.     for i in range(len(a)):
6.         a[i] = -a[i]
7.
8. if __name__ == '__main__':
9.     num = Value('d', 0.0)
10.    arr = Array('i', range(10))
11.
12.    p = Process(target=f, args=(num, arr))
13.    p.start()
14.    p.join()
15.
16.    print(num.value) # 3.1415927
17.    print(arr[:]) # [0, -1, -2, -3, -4, -5, -6, -7, -8, -9]
```

3.1.2 ABARBEITEN EINER AUFGABENLISTEN

Kann die Aufgabe in einer Liste verpackt werden, ist ein Aufgabenpool oftmals die einfachste Lösung. Dabei muss man sich um fast nichts selber kümmern.

```

1. from multiprocessing import Pool
2.
3. def add(arg):
4.     return arg[0] + arg[1]
5.
6. if __name__ == "__main__":
7.     pool = Pool(4)
8.     results = pool.map(add, [(1,2), (2,3), (3,4)])
9.     print(results) # [3, 5, 7]
```

3.1.2.1 LADEBALKEN SIND IMMER GUT

Eine Anzeige über den aktuellen Fortschritt lässt sich wie folgt realisieren.

```

1. import sys
2. import time
3. import multiprocessing as mp
4.
```

```
5. def add(arg):
6.     time.sleep(arg[0])
7.     return arg[0] + arg[1]
8.
9. if __name__ == "__main__":
10.    array = [(1,2), (2,3), (3,4)]
11.    pool = mp.Pool(processes=mp.cpu_count())
12.    results = pool imap_unordered(add, array)
13.    pool.close() # No more work -> start!
14.
15.    while (True): # show progress ...
16.        progress = results._index
17.        sys.stdout.write(f'\r{progress} of {len(array)} finished')
18.        if (progress >= len(array)): break
19.        sys.stdout.write('\n')
20.
21.    results = [i for i in results] # convert to list
22.    print(results) # [3, 5, 7]
```

4 LITERATURVERZEICHNIS

docs.python.org. (01. 07 2019). Von
<https://docs.python.org/3/library/multiprocessing.html> abgerufen

ANWENDUNG DES ADAM-OPTIMIZER MIT LSTM

1 ABSTRACT

Dieser Kurzbericht fasst die Erkenntnisse von Jason Brownlee zusammen und bildet eine Anleitung für die praktische Anwendung. Vorausgesetzt werden Grundkenntnisse zu LSTM Netzwerken. Diese Zusammenfassung soll zukünftigen Studenten helfen den Optimizer Adam richtig zu konfigurieren. Eine Erläuterung der genauen Funktionsweise wird bewusst weggelassen.

2 EINLEITUNG

Der Adam Optimizer ist ähnlich wie Stochastic Gradients, mit dem Unterschied, dass die Learning Rate (LR) dynamisch angepasst wird.

3 ANWENDUNGSBEREICH

Der Algorithmus kombiniert zwei andere Algorithmen und eignet sich besonders gut für folgende Problemstellungen:

- **Adaptive Gradient Algorithmen (AdaGrad):** Bei Problemen mit spärlichen Gradienten
- **Root Mean Square Propagation:** Bei Online- und nicht-stationären Problemen

4 PARAMETER

Die Standartwerte der jeweiligen Parameter können je nach Version und Framework variieren. Es empfiehlt sich die aktuellen Werte der jeweiligen Dokumentation zu entnehmen.

Zu jedem Parameter gibt es eine Konfigurationsempfehlung der Autoren, welche durch eigene Tests und Erfahrungen zusammengestellt sind. Als Test-Set dient die Vorhersage eines beliebigen Aktienkurses, dessen Charakteristik ist eine schwer Vorherzusagenden Datenkurve mit Rauschen.

4.1 LEARNING RATE

Default Value: 0.001

Die gegebene Learning Rate entscheidet über die LR zum Start des Trainings. Diese hat nur einen geringen Einfluss, solange diese nicht zu tief festgelegt wird (<0.00001). Tiefe Werte haben zur Folge, dass die Lernkurve flacher ausfällt.

4.1.1 EMPFEHLUNG

Die LR muss nicht verändert werden. Um das Training zu beschleunigen kann jedoch die LR zum Start etwa gleich gross gewählt werden wie die maximale Änderungsrate der Input-Daten. Mit dieser Faustregel kann nach eigenen Tests am schnellsten gelernt werden.

4.2 DECAY

Default Value: 0.0

Dieser Parameter beschreibt die Anpassung der LR. Nach jedem Batch wird die LR mit diesem Faktor multipliziert.

4.2.1 EMPFEHLUNG

Es empfiehlt sich bei komplexeren Netzwerken den Wert des Parameters auf null zu setzen. So kann verhindert werden, dass das Netzwerk am Ende die LR stagniert.

4.3 BETA 1 UND 2

Default Value Beta1: 0.9

Default Value Beta2: 0.999

Beta1 und 2 entscheiden über die Anpassung des Gradienten und geben somit Kontrolle über die Anpassung des Decay Wertes.

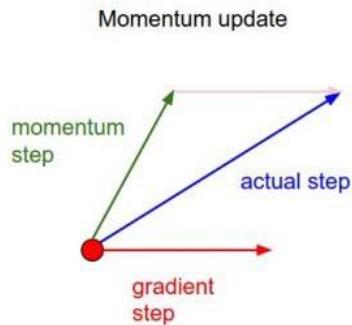


Abbildung 1: Momentum Update (Bushaeiv, 2019)

4.3.1 EMPFEHLUNG

Die Werte müssen nicht angepasst werden. Nach eigenem Test wurde die Lernkurve bei einer Vergrösserung oder Verkleinerung der Werte nur schlechter.

4.4 EPSILON

Default Value: 10E-8

Dieser Wert wird jeweils Addiert, um eine Division durch Null in den Berechnungen zu vermeiden.

4.4.1 EMPFEHLUNG

Kann auf dem Standardwert belassen werden. Zu Testzwecken kann dieser auch mal auf Null gesetzt werden, um festzustellen, ob sich eine Division durch Null bildet.

4.5 AMSGRAD

Default Value: False

Mit diesem Parameter kann auf den AMSGrad Algorithmus umgeschaltet werden.

4.5.1 EMPFEHLUNG

Je nach Situation kann es helfen auf den AMSGrad umzuschalten. Es empfiehlt sich von Zeit zu Zeit auf diesem Algorithmus zu wechseln.

5 ABBILDUNGSVERZEICHNIS

Abbildung 1: Momentum Update (Bushaeve, 2019) 2

6 LITERATURVERZEICHNIS

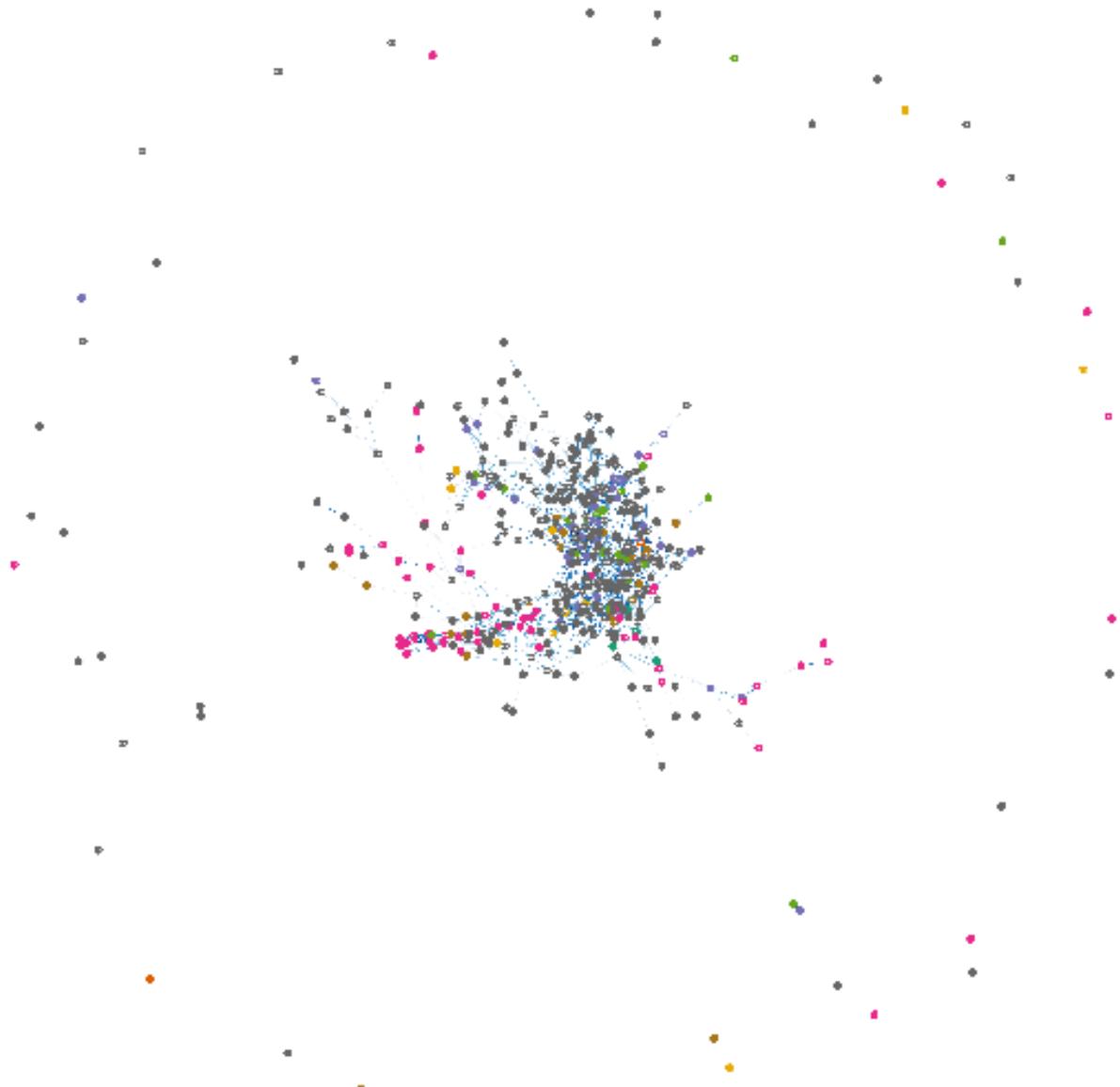
Brownlee, J. (July 3, 2017). *machinelearningmastery.com*. Von <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/> abgerufen

Bushaeve, V. (01. 07 2019). *towardsdatascience.com*. Von <https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c> abgerufen

KORRELATIONSKARTE

Folgende Korrelationskarte beschreibt den Zusammenhang aller Kurse. Es empfiehlt sich jedoch die digitale Version als SVG Datei zu betrachten. Diese ist unter folgendem Repository zu finden:

https://github.com/silviojaeger/ba_machine_learning_2019/tree/master/08_correlation_network



BEISPIEL KREUZKORRELATION

Dieses Beispiel zeigt eine Kreuzkorrelation des Aktienkurses der Firma Siemens mit allen verfügbaren Kursen.

Folgend die Parametrisierung:

Parameter	Wert
Korrelationszeitraum	180 Tage
Maximale Verschiebung	2 Tage
Zeiteinheit	15min Takt
Korrelationsmethode	Spearman

Parametrisierung der Kreuzkorrelation aller gegeben Daten.

Dabei erhält man beispielsweise folgende Korrelationstabelle für Siemens:

Kurs	Index	Shift
ADENCHCHF	0,9419533	0
ESPIDXEUR	0,93648397	5
NLDIDXEUR	0,93567244	0
GBRIDXGBP	0,93444599	0
EUSIDXEUR	0,93403688	1
DEUIDXEUR	0,92303425	1
HEIDEEUR	0,91585312	1
DPWDEEUR	0,91429094	0
VGKUSUSD	0,91157137	-29
EZUUSUSD	0,90991078	-27
FRAIDXEUR	0,90921643	0
UBSGCHCHF	0,90903576	0
MSUSUSD	0,90887168	-14
AUSIDXAUD	0,90652913	0
KNINCHCHF	0,90457751	0
VEAUSUSD	0,90385581	-38
UHRCHCHF	0,90383287	0
IJRUSUSD	0,9036158	-33
EFAUSUSD	0,9035458	-37
IWMUSUSD	0,90180623	-33
JPMUSUSD	0,89986166	-23
TROWUSUSD	0,89907028	-33
BAERCHCHF	0,89887053	0
BACUSUSD	0,89882805	-33

IJHUSUSD	0,89863055	-39
XLFUSUSD	0,89822559	-34
template	0,89463896	0
ABBNCHCHF	0,89463896	0
CONDEEUR	0,89310753	1
FCXUSUSD	0,89305176	-36
BOSSDEEUR	0,89246229	0
MUUSUSD	0,89126532	-144
BASDEEUR	0,88941482	0
SGSNCHCHF	0,88897136	0
FDXUSUSD	0,88684803	89
KEYUSUSD	0,88569625	-33
STIUSUSD	0,88518528	-13
IFXDEEUR	0,88352686	0
CSGNCHCHF	0,88274968	0
EEMUSUSD	0,88209224	-38
BIDUUSUSD	0,88107992	0
SIKCHCHF	0,88024954	0
IBBUSUSD	0,87965278	-144
XLYUSUSD	0,87812453	-70
LENUSUSD	0,87774269	117
SNAPUSUSD	0,87728824	-14
LXSDEEUR	0,8755228	0
RFUSUSD	0,87454959	-14
IWDUSUSD	0,87448246	-144
TIFUSUSD	0,87438663	0
DAIDEEUR	0,87380445	0
USA500IDXUSD	0,87325267	-144
JCIUSUSD	0,86993668	-144
PNCUSUSD	0,86979187	-23
SPYUSUSD	0,86965932	-144
USATECHIDXUSD	0,86927613	-144
TXNUSUSD	0,8687634	-144
AMATUSUSD	0,8686397	-1
VFCUSUSD	0,86859504	-134
BMWDEEUR	0,8675986	0
NKEUSUSD	0,86759075	-38
QQQUSUSD	0,8664637	-144
JNKUSUSD	0,86628048	-34
MARUSUSD	0,86549123	-18
EBAYUSUSD	0,86508711	-47

DVYUSUSD	0,8646769	-144
CELGUSUSD	0,86447598	-144
CSUSUSD	0,86413582	-15
IWFUSUSD	0,86293112	-144
GOOGLUSUSD	0,86249939	-25
VXXUSUSD	0,86219279	-39
IVWUSUSD	0,86200713	-144
EXPEUSUSD	0,86092871	95
SGDIDXSGD	0,86026132	0
GOOGUSUSD	0,86009317	-23
BBYUSUSD	0,85945455	74
BDXUSUSD	0,85925287	-144
SAPDEEUR	0,85899378	0
WYNNUSSUSD	0,8582988	-39
XLKUSUSD	0,8581593	-144
LVSUSUSD	0,85811606	-38
IVEUSUSD	0,85761252	-144
WDCUSUSD	0,8552636	-35
CTSHUSUSD	0,85377117	-144
LUSUSD	0,85305831	-144
DFSUSUSD	0,85299927	-14
DBKDEEUR	0,85222198	0
LRCXUSUSD	0,85172058	-1
XUSUSD	0,85167891	-39
STTUSUSD	0,85110045	-26
MATUSUSD	0,85045209	-23
PSMDEEUR	0,85004184	1
GSUSUSD	0,84990539	-7
METUSUSD	0,84944574	-39
AAUSUSD	0,84868174	-35
BIIBUSUSD	0,84746155	-144
EAUSUSD	0,847407	0
VMCUSUSD	0,84736131	-14
FMEDEEUR	0,8471127	-144
DVNUSUSD	0,84689807	-144
VLOUSUSD	0,84626607	-144
CSXUSUSD	0,84540577	-39
CLNCHCHF	0,84476934	0
ADSDEEUR	0,84384426	0
SDFDEEUR	0,84363935	0
CUSUSD	0,84333601	-39

GWWUSUSD	0,84321831	-144
TSMUSUSD	0,83966403	-39
OXYUSUSD	0,83778592	-144
PSXUSUSD	0,83649459	-144
DHIUSUSD	0,83644401	89
SCHWUSUSD	0,83636991	-38
FBUSUSD	0,83594622	-2
AIGUSUSD	0,83486905	-144
EWJUSUSD	0,83416243	-39
WFCUSUSD	0,83333087	77
HKGIDXHKD	0,83324296	28
HDUSUSD	0,83241366	-144
NVDAUSUSD	0,83040907	-144
EURDKK	0,83023396	-144
AMZNUSUSD	0,82735777	-144
SHWUSUSD	0,82617523	-39
ISRGUSUSD	0,82588901	-39
NSCUSUSD	0,82534944	-39
AABAUSUSD	0,82388846	-2
SWKUSUSD	0,82218774	-39
DB1DEEUR	0,82196813	-144
AALUSUSD	0,8211781	71
EFXUSUSD	0,82077425	75
UPSUSUSD	0,81993877	-144
BKUSUSD	0,8197966	-1
XLIUSUSD	0,81578848	-144
USA30IDXUSD	0,81576727	-144
NFLXUSUSD	0,81540821	-144
XLEUSUSD	0,81448829	-144
PHUSUSD	0,81433528	-144
ALLUSUSD	0,81397077	-144
DIAUSUSD	0,81377084	-144
EMRUSUSD	0,81316069	-144
PPGUSUSD	0,81002748	71
SYYUSUSD	0,8075064	-134
MGMUSUSD	0,80724277	-3
VUSUSD	0,80711114	-38
ADSKUSUSD	0,80528734	-144
PAYXUSUSD	0,80509741	-144
BABAUSUSD	0,80359676	-1
ETHUSD	0,80154532	-136

LHADEEUR	0,80050834	100
PCARUSUSD	0,79978412	-39
JPNIDXJPY	0,79752408	-144
LUVUSUSD	0,79718842	-144
MPCUSUSD	0,79706883	-144
APDUSUSD	0,7949615	-144
GLWUSUSD	0,79297518	-37
ADIUSUSD	0,79281376	-6
TGTUSUSD	0,7911482	-144
SOONCHCHF	0,79073913	-144
TRVUSUSD	0,79040954	-142
OJUICECMDUSX	0,79037997	-29
UTXUSUSD	0,78956648	0
USDIIS	0,78900549	-144
EWWUSUSD	0,78839919	35
HONUSUSD	0,78756565	-144
GPSUSUSD	0,78730957	73
MAUSUSD	0,78506559	-144
LOWUSUSD	0,78478828	-144
CTLUSUSD	0,78437726	90
XOPUSUSD	0,78267698	-144
XAUUSD	0,782385	-144
TUI1DEEUR	0,78187975	0
RTNUSUSD	0,78183746	-144
GLDUSUSD	0,7804876	-144
PRUUSUSD	0,77992106	-144
IBMUSUSD	0,776726	-144
CRMUSUSD	0,77578301	-144
ADBEUSUSD	0,77553873	-39
HALUSUSD	0,77209798	-144
ILMNUSUSD	0,77183926	-76
ITWUSUSD	0,76692058	-1
HPQUSUSD	0,76544915	-144
GILDUSUSD	0,76414339	-144
AUDNZD	0,76002847	-77
HEN3DEEUR	0,7596443	0
MROUSUSD	0,75936554	-144
BAYNDEEUR	0,75820605	-144
EURTRY	0,75790567	-144
CMEUSUSD	0,75768006	144
KUSUSD	0,75753272	144

PCLNUSUSD	0,75484997	-144
ABEVUSUSD	0,75411183	-144
EMBUSUSD	0,75286302	-144
PXDUSUSD	0,75272264	-144
HESUSUSD	0,75149178	-144
EQTUSUSD	0,75047899	72
KHCUSUSD	0,74976763	67
MMMUSUSD	0,74954806	-144
MCHPUSUSD	0,74944401	0
LINDEEUR	0,74892852	-144
HUMUSUSD	0,74740915	-144
QCOMUSUSD	0,74477755	-144
USDTRY	0,74452261	-144
FUSUSD	0,74120797	144
ZBHUSUSD	0,74026685	-39
TUSUSD	0,74003533	-144
TRYJPY	0,73979862	-144
KMIUSUSD	0,73871563	-144
CATUSUSD	0,73710549	-144
BTCUSD	0,73633253	-144
GASCMDUSD	0,73593122	144
CHEIDXCHF	0,7354408	0
BEIDEEUR	0,73469486	0
NRGUSUSD	0,73430932	144
TKADEEUR	0,73414517	-144
DUSUSD	0,73310371	116
MUSUSD	0,73178003	-37
USDDKK	0,73175647	-144
SBUXUSUSD	0,73048637	144
DVAUSUSD	0,73004177	-144
GDXUSUSD	0,72931001	-144
EURHKD	0,72801097	-144
BBDUSUSD	0,72716131	144
AEPUSUSD	0,72699792	115
FXIUSUSD	0,72467764	-39
MCDUSUSD	0,72267571	144
OKEUSUSD	0,72235443	-143
EURUSD	0,72208731	-144
EWHUSUSD	0,71937191	-39
LHNCHCHF	0,71903622	0
USDPLN	0,71897107	-144

USDRON	0,71874864	-144
BMYUSUSD	0,7153311	-144
FREDEEUR	0,71299484	-144
COPUSUSD	0,71183181	-144
APAUSUSD	0,70862252	-144
LLYUSUSD	0,70846291	144
USDCZK	0,70615492	-144
INDIDXUSD	0,7041312	41
DUKUSUSD	0,70032347	11
CLUSUSD	0,70012418	-144
RWEDEEUR	0,69712635	0
VRTXUSUSD	0,69680614	-144
INTUUSUSD	0,69603881	-144
ATVIUSUSD	0,69235344	-144
PYPLUSUSD	0,68816551	-34
MRKUSUSD	0,68640141	144
AMDUSUSD	0,6855486	-144
WBAUSUSD	0,68515664	144
GEUSUSD	0,68238715	-144
DOLLARIDXUSD	0,68099171	-144
TEVAUSUSD	0,67918785	144
UNPUSUSD	0,67873045	-144
DTEDEEUR	0,67771111	-144
CAGUSUSD	0,6746844	144
EURCZK	0,67371809	144
CADHKD	0,67316427	32
CFUSUSD	0,67284263	-144
ADPUSUSD	0,67178458	-144
BBTUSUSD	0,67034397	72
AAPLUSUSD	0,6682172	-144
EURRUB	0,66406951	87
USDCAD	0,66352528	9
EOGUSUSD	0,66327452	-144
OMCUSUSD	0,66150213	144
NBLUSUSD	0,65911184	-144
EURSGD	0,6579244	-144
WHRUSUSD	0,65730346	144
USDCNH	0,65575547	26
BPUSUSD	0,65210879	-144
KSSUSUSD	0,65161337	-64
REGNUSUSD	0,65074491	-144

PBRUSUSD	0,64943274	144
XOMUSUSD	0,64799105	-144
LMTUSUSD	0,64671644	-144
USDMXN	0,64575276	144
CVXUSUSD	0,6444722	-144
EWZUSUSD	0,64433362	144
MSFTUSUSD	0,64375325	-144
SUGARCMDUSD	0,64336111	144
LIGHTCMDUSD	0,63911986	-144
XLVUSUSD	0,639097	-144
FEUSUSD	0,63822671	144
EURSEK	0,63127193	144
APCUSUSD	0,62801945	-144
TSNUSUSD	0,62650248	-23
AMGNUSUSD	0,62152912	-144
USBUSUSD	0,62000846	-23
USDHUF	0,61659338	67
USOUSUSD	0,61412621	-144
USDCHF	0,61393219	-144
DALUSUSD	0,61316341	-92
ESRXUSUSD	0,61010815	144
SLHNCHCHF	0,60835567	144
COSTUSUSD	0,60814085	-144
NEMUSUSD	0,60633324	-144
NOVNCHCHF	0,59960067	144
CSCOUSUSD	0,59953018	-144
FOXAUSUSD	0,594359	144
CHFSGD	0,5938752	-144
NOCUSUSD	0,59117418	-144
MCKUSUSD	0,59045098	-144
VZUSUSD	0,58821566	-39
AMTUSUSD	0,58787973	-144
TJXUSUSD	0,58645155	-144
ORCLUSUSD	0,58574251	-39
VNQUSUSD	0,583792	-144
SCMNCHCHF	0,58364936	-144
NEEUSUSD	0,57821349	-85
EURJPY	0,57682702	-144
XLUUSUSD	0,57496833	144
NZDCHF	0,57181373	-144
SOUSUSD	0,57161914	-94

CBKDEEUR	0,56943568	-144
GISUSUSD	0,56822625	144
IYRUSUSD	0,56119847	-144
PGUSUSD	0,56093214	-144
BRENTCMDUSD	0,5564785	-144
CMGUSUSD	0,55638924	126
KOUSUSD	0,555005	-93
COFFEECMDUSX	0,55429829	144
CHIIDXUSD	0,55378814	-144
USDNOK	0,54889868	-144
EURNZD	0,54067397	-144
KMBUSUSD	0,54066038	144
ALVDEEUR	0,53698727	0
LONNCHCHF	0,52501485	-144
EXCUSUSD	0,51492763	-52
DHRUSUSD	0,51452	-144
CHFJPY	0,51442773	-144
EURHUF	0,50949425	-144
PMUSUSD	0,5033654	144
XLNXUSUSD	0,50090688	144
MRKDEEUR	0,4953539	-144
GBPUSD	0,49370497	-144
ROGCHCHF	0,48755128	144
CIUSUSD	0,48753036	144
STZUSUSD	0,4849742	-144
VNADEEUR	0,47932858	144
USDHKD	0,47878347	144
SYKUSUSD	0,47237657	-144
BRKBUSUSD	0,46904222	-144
JWNUSUSD	0,46758043	-34
HCPUSUSD	0,46477206	-144
ANTMUSUSD	0,4628254	144
ALXNUSUSD	0,4617848	-144
TAPUSUSD	0,46108519	86
AETUSUSD	0,45937218	144
COTTONCMDUSX	0,45897166	68
NESNCHCHF	0,45809313	144
BAUSUSD	0,45406765	-144
KRUSUSD	0,45339308	144
TMOUSUSD	0,44800539	-144
USDTB	0,44601175	-144

HCNUSUSD	0,44142232	-85
DIESELCMDUSD	0,42751838	-144
TLTUSUSD	0,42504999	144
AZOUSUSD	0,4236076	144
PGRUSUSD	0,42170109	-144
YUMUSUSD	0,41341634	144
NZDCAD	0,40993157	-144
PFEUSUSD	0,40957898	144
USDSGD	0,40427404	56
TSLAUSUSD	0,4016353	-144
GBPNZD	0,40004857	-144
ABCUSUSD	0,39362436	-144
NZDJPY	0,39258091	144
ZARJPY	0,38909077	144
CMIUSUSD	0,38450798	-144
HKDJPY	0,37966626	144
AXPUSUSD	0,37270914	-144
UKGILTRGBP	0,3726725	-144
ROSTUSUSD	0,37122718	-144
RHTUSUSD	0,37003838	-144
USDJPY	0,36829548	144
CVSUSUSD	0,36662253	-144
USDRUB	0,36614308	144
JNJUSUSD	0,35477002	144
AUDUSD	0,35084446	44
ABTUSUSD	0,34457633	144
EOANDEEUR	0,33832695	144
RRCUSUSD	0,33649985	-144
AUDCHF	0,32318345	-144
CADJPY	0,31401638	-144
NZDUSD	0,31165711	-144
PCGUSUSD	0,3058742	-144
ELUSUSD	0,30247968	-144
AZNUSUSD	0,30198936	144
EURAUD	0,30057648	-144
COCOACMDUSD	0,29003601	125
USDZAR	0,28765644	144
AUDCAD	0,28560875	-144
EURGBP	0,28398145	144
GDXJUSUSD	0,27823651	-144
AUDSGD	0,27018196	45

GMUSUSD	0,2675278	0
BUNDTREUR	0,266988	-144
MOUSUSD	0,26564031	-144
GBPCHF	0,25418799	144
DISUSUSD	0,25339487	144
SRENCHCHF	0,25262745	144
SYMCUSUSD	0,25222531	144
PSAUSUSD	0,24795708	144
GBPJPY	0,24675694	-144
ICEUSUSD	0,23816674	144
SJMUSUSD	0,23681878	144
BSXUSUSD	0,23276563	-144
CBSUSUSD	0,23012831	-144
COPPERCMDUSD	0,22604408	144
EIXUSUSD	0,22537437	-20
DEUSUSD	0,21426273	-144
PEPUSUSD	0,21300162	121
SGDJPY	0,21071334	144
UNHUSUSD	0,20973526	-144
GBPCAD	0,20835521	144
ORLYUSUSD	0,20723087	144
MUV2DEEUR	0,20611577	144
XLPUSUSD	0,20544909	144
AUSUSD	0,20049637	-144
ULTAUSUSD	0,19271536	144
AVGOUSUSD	0,18875521	144
PLNIDXPLN	0,18691577	-144
NWLUSUSD	0,18619716	-38
XAGUSD	0,17696966	-144
INTCUSUSD	0,17687384	-144
VIABUSUSD	0,17075467	-144
CMCSAUSUSD	0,16482848	144
VALEUSUSD	0,16088288	144
IEFUSD	0,15626308	144
SPGUSUSD	0,15594107	144
GBPAUD	0,15061382	-144
DGUSUSD	0,14247951	-37
EURPLN	0,13200434	4
SLVUSUSD	0,13125404	-144
ITUBUSUSD	0,12952375	144
CAHUSUSD	0,12601151	-144

EURCAD	0,12095974	144
DLTRUSUSD	0,11727547	137
IPGUSUSD	0,1118535	144
ZURNCHCHF	0,11001627	144
PAH3DEEUR	0,10259132	0
MDLZUSUSD	0,09901539	144
EURNOK	0,089311	-144
WMTUSUSD	0,08705094	-144
CADCHF	0,08700196	144
GIVNCHCHF	0,08005581	144
VOW3DEEUR	0,07511694	-144
AVBUSUSD	0,0721295	144
EURCHF	0,0699902	144
USDSEK	0,0654003	144
AUDJPY	0,06123428	-144
TWTRUSUSD	0,01278586	-24
SINUS	0,0050331	136