Hochschule Bochum
Bochum University
of Applied Sciences

Hochschule Bochum
Fachbereich Wirtschaft

Bachelorarbeit

Einbeziehung des Luftverkehrs in den europäischen Emissionshandel

Erstprüfer: Prof. Dr. …

Zweitprüfer: Prof. Dr. …

Jana Mustermann

Matrikelnummer: 123456789

Musterstraße. 1

44801 Bochum

Tel.: 0234/123456

E-Mail: jana.mustermann@hs-bochum.de

Studiengang:

Internationales Management

4. Fachsemester, Sommersemester 2013

Abgabedatum 21. Mai 2013

Inhaltsverzeichnis

Abkürzungsverzeichnis

ABl. ..... Amtsblatt der Europäischen Union

APU ..... Auxiliary Power Units

BIP ..... Bruttoinlandsprodukt

BMU ..... Bundesministerium für Umwelt

DEHSt ..... Deutsche Emissionshandelsstelle

EEX ..... European Energy Exchange

EG ..... Europäische Gemeinschaft

ETS ..... Emission Trading System

EU ..... Europäische Union

EUA ..... European Union Allowances

FCOM ..... Flight Crew Operating Manual

IATA ..... International Air Transport Association

ICAO ..... International Civil Aviation Organisation

IPCC ..... Intergovernmental Panel on Climate Change

LCC ..... Low Cost Carrier

OECD ..... Organisation für wirtschaftliche Zusammenarbeit und Entwicklung

TEHG ..... Treibhausgas-Emissionshandelsgesetz

UBA ..... Umweltbundesamt

Abbildungsverzeichnis

1 Einleitung

// Wird später erstellt

2 Maschinelles lernen

Hinter dem Begriff Maschinelles Lernen (engl. machine learning) verstecken sich Algorithmen die fähig sind von Daten zu lernen um mit neuen gleichartigen Daten umzugehen.
Mitchell liefert eine Definition zu dem Lernprozess:
„A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."[1]
Dabei ist die Erfahrung E eine Sammlung von Datensätzen, die von der Aufgabe abhängig sind, wie beispielsweise Sensordaten, Bilder, Texte oder ähnliches. Die Aufgabe T ist was der Algorithmus tun soll wie beispielsweise Werte vorhersagen, Datensätze klassifizieren, Bilder generieren usw. Wie gut der Algorithmus diese Aufgaben ausführt wird durch die Messung von P bestimmt. P wird häufig auch als Fehlerfunktion, Verlustfunktion oder auch als Zielsetzungsfunktion bezeichnet.[2]
Generell wird zwischen drei verschiedenen übergeordneten Lerntypen unterschieden:

- Überwachtes Lernen: Dabei hat jeder Datensatz neben den eigentlichen Daten noch den gewünschten Output bzw. ein Label. Das Ziel ist es eine Funktion zu finden die den Datensatz auf den Output mappt. Der Term „überwacht" kommt daher, dass ein „Lehrer" oder auch Supervisor dem Algorithmus beim Lernen hilft und ihm den richtigen Wert vorgibt. Das überwachte lernen stellt das Herz des maschinellen Lernens da und viele Algorithmen des maschinellen Lernens versuchen, wie wir später sehen werden, die Aufgabe auf eine Problemstellung des überwachten Lernens zurückzuführen.
- Unüberwachtes Lernen: Bei dem unüberwachten lernen wird dem Algorithmus kein entsprechendes Zielattribut vorgegeben. Es geht überwiegend darum verborgenes Wissen bzw. Information zu extrahieren und dem Anwender tiefere Einblicke zu gewähren um Schlüsse über Verhalten oder Kausalitäten aufzudecken. Es ist eng verwandt mit der Mustererkennung und Data Mining. Eine Besonderheit an dieser Problemstellung ist, dass die Anzahl der verfügbaren Daten enorm ist im Vergleich zu dem überwachten Lernen, in der der Output vorgegeben wird. Unüberwachtes lernen wird auch als „Information Retrieval"-Prozess bezeichnet.
- Bestärkendes Lernen: Eine weitere Lernform ist das bestärkende Lernen dabei interagiert der Algorithmus mit der Umgebung und erhält Inputreize aufgrund von vorheri-

[1] Mitchell (1997) Seite 2
[2] Goodfellow 82

gen Handlungen in einer Umgebung. Ziel ist es, basierend auf dem Muster von Inputreizen entsprechende Aktionen, bzw. Reaktionen auszuführen mit dem Ziel seine Belohnungen (engl. reward) zu maximieren. Der Algorithmus lernt mit fortschreitender Simulation wie gut oder wie schlecht eine Aktion war und passt seine Aktionen dementsprechend an.[3]

Das Lernen im eigentlichen Sinne geschieht meistens dadurch, dass der Algorithmus durch eine Reihe von Parametern θ sein Verhalten gegenüber den Inputmustern bestimmt wird. Durch gezielte Beeinflussung dieser Parameter lernt der Algorithmus das gewünschte Verhalten. Zusammengefasst lässt sich also sagen das wir Erfahrungen bzw. Daten brauchen, ein Model bzw. einen Algorithmus, der diese Daten verarbeitet und eine Performance Funktion w.r.t. der Aufgabe.

2.1 **Modelle des Überwachten Lernens**

Ziel des überwachten lernen ist es ein Model zu finden das in der Lage ist, auf neue noch nicht gesehene Daten richtig zu reagieren[4]. Damit ein Model diese Anforderung erfüllt führen wir die Begriffe Unteranpassung, Überanpassung und die Fähigkeit zu generalisieren ein, um Modelle nach diesen Punkten zu bewerten. Als Beispiel nehmen wir uns ein Regressionsproblem, bei dem wir versuchen eine Reihe von Datenpunkten mit einer Funktion zu approximieren sodass die Punkte auf oder möglichst nahe an dem Graphen der Funktion liegen. Das Model erhält eine Eingabe x und generiert dazu den geschätzten Wert für das Zielattribut y.

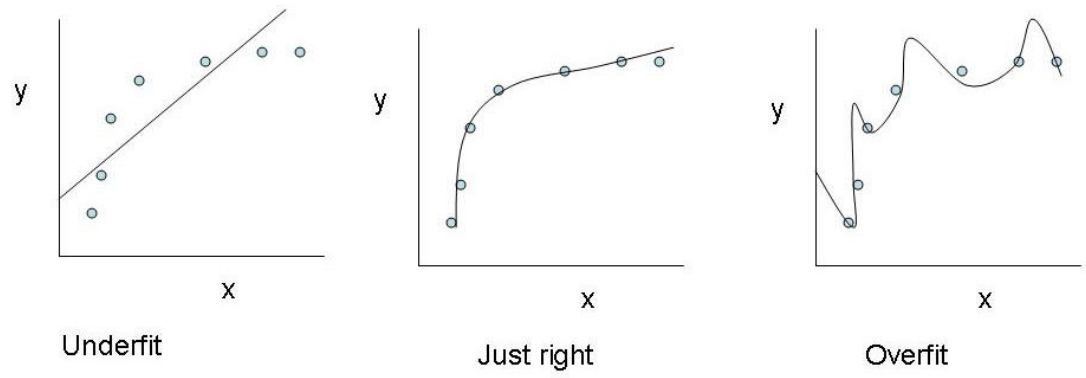

Abbildung 1: Unter- und Überanpassung anhand eines Regressionsproblems
Quelle: https://gigadom.files.wordpress.com/2014/01/41.jpg
// https://gigadom.in/2014/01/03/simplifying-machine-learning-bias-variance-regularization-and-odd-facts-part-4/

[3] David Kriesel 53
[4] Goodfellow 110

Es ist sehr deutlich erkennbar das der Graph auf der linken Seite die ursprünglichen Daten suboptimal abbildet. Das liegt daran, dass das Model versucht eine nicht lineare Funktion mit einer linearen Funktion zu approximieren. Allerdings liegt der Fehler nicht an dem Model, sondern an dem Anwender, der den Daten einen linearen Verlauf unterstellt und das Model falsch designt hat. Der Graph auf der rechten Seite zeigt eine Überanpassung des Graphen. Dabei wird versucht ein Polynom höheren Grades auf ein Polynom kleineren Grades abzubilden. Wenn wir die Funktion nutzen würden um einen neuen Wert abzubilden würden wir untypische Werte erhalten, die zwar den Trainingsdaten entsprechen allerdings nicht den neuen Daten. Das Bild in der Mitte zeigt ein Model, dass eine Funktion gelernt hat das die Daten am ehesten repräsentiert. Eine Aufgabe ist es einen guten Trade-off zwischen einer Unter- und Überanpassung zu finden um die Funktion möglichst exakt zu approximieren. Generell lässt sich sagen das die Komplexität der Aufgabe ungefähr der Komplexität des Models entsprechen soll.[5] Um die Fähigkeit des Generalisierens zu testen werden in der Praxis häufig gezielt Datensätze entfernt bzw. dem Model vorenthalten um das Model auf genau diese Eigenschaften hin zu überprüfen. Diese Überprüfung kann beispielsweise durch die Performance Messung stattfinden, sofern die Labels bekannt, um einen Fehlerwert zu berechnen und diese miteinander zu vergleichen.

2.2 **Algorithmen des Überwachten Lernens**

Es existieren eine Vielzahl an Modellen um die Aufgabestellung des überwachten Lernens zu lösen. Je nach Menge der Daten, der Problemstellung sowie Verfügbarkeit von Rechenressourcen hängt die Wahl des entsprechenden Algorithmus ab. Wir werden in dieser Arbeit zwei gängige Vertreter betrachten, die für eine Vielzahl an Problemstellungen einsetzbar sind. Nämlich künstliche neuronale Netze die sich insbesondere durch ihre Vielfältige Einsatzmöglichkeiten auszeichnen und konvolutionale Netze, die meist in der Bilderkennung eingesetzt werden.

2.2.1 Neuronale Netze

In dem vergangenen Jahrzehnt haben sich künstliche neuronale Netze als effiziente Möglichkeit herausgestellt um nicht lineare Funktionen zu approximieren[6]. Wie der Name bereits vermuten lässt war die ursprüngliche Idee dahinter das zentrale Nervensystem auf einer mathematischen Ebene abzubilden. Zu den Pionieren in diesem Themenfeld gehören Warren McCul-

[5] Goodfellow S.112
[6] Vgl. Goodfellow et al (2016), S. 224

loch und Walter Pitts[7] die in ihrer Arbeit im Jahr 1943 erste Ideen für ein vorwärts gerichtetes neuronales Modell vorstellen.

Die Idee hinter neuronalen Netzen ist, dass ein Neuron durch die eine Reihe von Inputs aktiviert wird bzw. Informationen erhält und diese dann durch eine mathematische Vorschrift miteinander kombiniert und anschließend den berechneten Wert weitergibt. Wir werden uns in diesem Kapitel mit den wesentlichen Elementen von neuronalen Netzen vertraut machen, die in einer Problemstellung des überwachten Lernens angewendet werden.

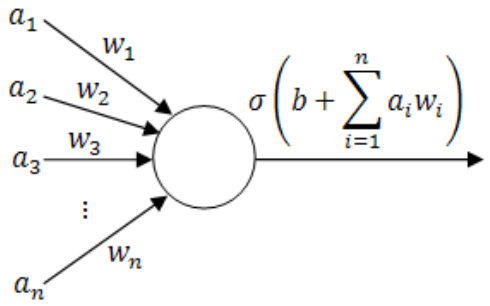

Abbildung 2: Neuronales Netz mit einem einzigen Neuron

Die Abbildung zeigt den schematischen Aufbau Neuronalen Netzes mit einer Inputschicht und einem Ausgabeneuron bzw. einer Ausgabeschicht mit einem einzigen Neuron. Auf der linken Seite befinden sich die Inputreize in Form eines Vektors $a \in \mathbb{R}^n$ (a entspricht in dem Falle des Eingabevektors x) die mit einem Gewichtsvektor $w \in \mathbb{R}^n$ multipliziert werden und so den vorläufigen Output des Neurons darstellen. Es wird noch ein Offset Bias Parameter b hinzuaddiert da der vorhergesagte Graph sonst nur durch den Ursprung laufen würde[8]. Beispielswiese könnte die Funktion $f(x) = 1$ nicht abgebildet werden ohne den Bias Parameter. Abschließend wird noch eine sogenannte Aktivierungsfunktion angewandt um nicht lineare Funktionen abzubilden zu können. Die Gewichte lassen sich wie folgt interpretieren. Ein negatives Gewicht ist ein für das Neuron hemmender Reiz das die Aktivierung des Neurons mindern soll. Ein positives Gewicht hingegen hat einen erregenden Einfluss und soll das Neuron in einen aktiven Status zu versetzen. Wenn das Gewicht nahe bei null ist übt es keine oder nur eine geringe Wirkung aus, es hat also weder eine animierende noch hemmende Wirkung.[9]

Dieses Model ist allerdings nur beschränkt möglich eine beliebig komplexe Funktion zu erlernen deswegen wird in dem Kontext von neuronalen Netzen häufig der Begriff „Deep learning“ verwendet. Es handelt sich dabei um die Aneinanderreihung von neuronalen Netzschichten um komplexe geschachtelte Funktionen zu erlernen.

[7] Warren S. McCulloch and Wallter H Pitts (1943)
[8] Goodfellow 109
[9] Vgl. Neuronale Netze S.15

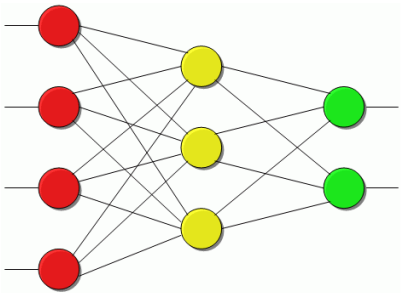

Abbildung 3: Der Aufbau eines tiefen neuronalen Netzes
Quelle: http://www.methoden-psychologie.de/neuronale_netze.html

http://www.methoden-psychologie.de/neuronale_netze.html

Die Abbildung zeigt ein tiefes neuronales Netz mit einer Inputschicht (rot), einer versteckten Schicht (gelb) und einer Ausgabeschicht (grün). Die mathematische Vorschrift um die Ausgabe zu berechnen sieht wie folgt aus:

$$f(x) = f^2\big(f^1(x)\big)\, mit\, f^i(x) = \ activation\ (W_i\, x + \ b_i)$$

Jede Schicht bekommt eine Gewichtsmatrix W und einen Bias Parameter b zugeordnet, mit denen sich die Eingaben einer höheren Schicht berechnen lassen. Diese Prozedur wird auch als vorwärts gerichtete Propagieren bezeichnet. [10]

Das designen von versteckten Schichten wie beispielsweise die Auswahl von Aktivitätsfunktionen sowie Festlegung der Anzahl der Neuronen pro Schicht wird im Zusammenhang mit der Struktur, mit der die einzelnen Schichten in Verbindung stehen wird auch als Model Architektur bezeichnet.[11] Die Auswahl wird vor dem Trainingsprozess festgelegt und kann in der Regel nicht nachträglich beeinflusst werden.

Ein weiterer wichtiger Faktor ist die Wahl der Aktivierungsfunktionen. Üblicherweise werden in dem gesamten Netzwerk die gleichen Aktivierungsfunktionen verwendet. Außer der Ausgabeschicht in der diese nach den Bedürfnissen angepasst werden (wie beispielsweise der Wertebereich). Dabei ist es in den meisten Fällen so, dass zwingend Aktivierungsfunktionen benötigt werden um die Mächtigkeit des Netzwerkes zu erhöhen. Präziser ausgedrückt werden Aktivierungsfunktionen benötigt um nicht lineare Funktionen approximieren zu können. Um die Relevanz dieser zu zeigen modifizieren wir die Funktion für die vorwärts gerichtete Propagation. In ihr wird auf die Aktivierungsfunktion verzichtet sowie auf dem Bias Parameter um die Gleichung übersichtlicher zu gestalten.

$$f(x) = f^2\big(f^1(x)\big)\, mit\, f^i(x) = \ W_i x$$

Anders formuliert entsteht:

$$f(x) = W_2 W_1 x = W^* x$$

[10] NN 52
[11] Goodfellow 197

Das tiefe neuronale Netz hat also nicht die Fähigkeit komplexere Funktion zu erlernen, da sie mit einer einzigen Gewichtsmatrix W* ausgetauscht werden kann. Es werden also zwingend Aktivierungsfunktionen benötigt um nicht lineare Funktionen zu erlernen.[12]

Prinzipiell erfüllt jede Funktion diese Eigenschaft, die nicht ausschließlich linear ist. Allerdings haben sich in der Literatur sowie in der Praxis einige Aktivitätsfunktionen etabliert wobei in diesem Bereich noch aktiv Forschung betrieben wird. Die Auswahl kann nach verschiedenen Kriterien erfolgen wie beispielsweise der Begrenzung des Wertebereichs, die biologische Plausibilität oder aber auch die Differenzierbarkeit der Funktion.[13] Eine weitere interessante Eigenschaft ist wie sich die Ableitung der Funktion verhält, also ob Gradient auf bestimmten Teilbereichen der Funktion sehr klein ist oder sehr groß ist.[14] Sollte der Gradient klein sein macht diese Eigenschaft das Lernen für den Algorithmus schwerer und es wird mehr Zeit benötigt um ähnliche Ergebnissen zu erzielen wie bei anderen Aktivierungsfunktionen. Nachfolgend sind einige gängige Aktivierungsfunktionen aufgelistet.

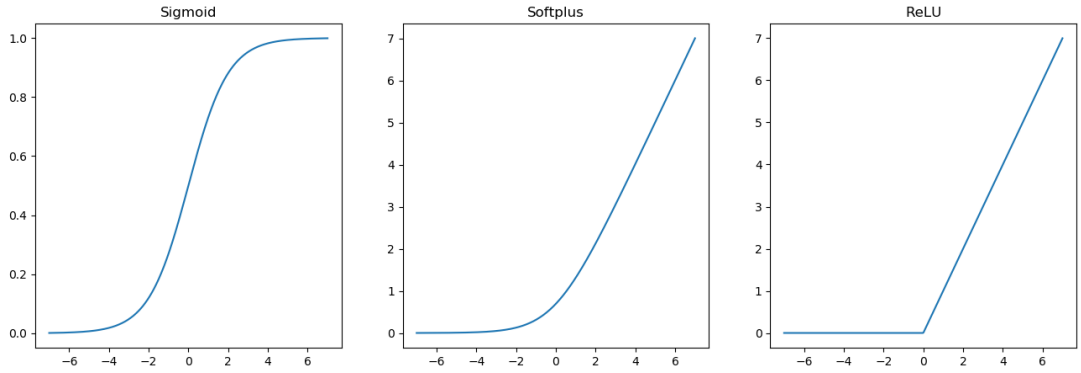

Abbildung 4: Gängige Aktivierungsfunktion. Links die Sigmoid Funktion in der Mitte die Softplus Funktion und rechts die ReLU Funktion

Quelle: Eigener Plot

- Sigmoid: Bei der Sigmoid Funktion liegt der Wertebereich zwischen 0 und 1. Daher wird diese Funktion häufig bei Klassifizierungsproblemen eingesetzt um zu bestimmen mit welcher Sicherheit der Algorithmus eine Klassenzugehörigkeit vorhersagt. Eine weitere nützliche Eigenschaft ist die leichte Differenzierbarkeit $\sigma'(z) = \sigma(z)(1 - \sigma(z))$, da die Werte beim Maschinellen lernen oftmals bereits berechnet wurden und sie somit wiederverwendet werden können. Ein Nachteil dieser Funktion ist, dass der Gradient bei z≪0 und z≫0 sehr klein ist und das Lernen somit erschwert wird.

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

[12] Goodfellow 172
[13] NN 22
[14] Goodfellow 179

- Softplus: Die Softplus Funktion hat einen Wertebereich 0 bis ∞. Sie kommt zur Vorhersage von positiven Werten zum Einsatz.

$$\varsigma(x) = \ln(1 + e^x)$$

- Rectified Linear Unit (ReLU): Die ReLU Funktion verhält sich ähnlich wie die Softplus Funktion allerdings ist der Übergang härter. Die ReLU Funktion ist streng genommen nicht differenzierbar bei x = 0 jedoch wird sie in der Praxis häufig verwendet, da sie einfach zu implementieren ist und Steigung konstant bleibt.[15]

$$ReLU(x) = \max(0, x)$$

2.2.2 Konvolutionale Netzwerke

Konvolutionale Netze werden überwiegend bei der Analyse von Bilddaten verwendet. Der Unterschied zu klassischen neuronalen Netzen ist, dass nicht länger ein Vektor an Daten analysiert wird, sondern eine Matrix an Daten. Das heißt die Bilddaten stehen in einer Korrelation zueinander. Bildpunkte die näher beieinander liegen stehen in einer engeren Verbindung als weit entfernte Bildpunkte[16]. Vergleichbar wie bei einem tieferen neuronalen Netz ist das Netzwerk auch in verschiedenen Schichten aufgeteilt. Jede dieser Schichten besteht typischerweise aus:

- einer Konvolutionsoperation
- einer Aktivierungsfunktion
- und einer Poolingsoperation

Durch das verketten dieser Schichten ist es möglich, genau wie bei neuronalen Netzen, komplexere Funktionen zu erlernen die leistungsfähiger als alleinstehende sind. Die letzten Schichten eines konvolutionalen Netzes sind üblicherweise neuronale Netze dessen Input die Ausgabe der letzten Konvolutionsschicht darstellt die als Vektor neu arrangiert worden sind.

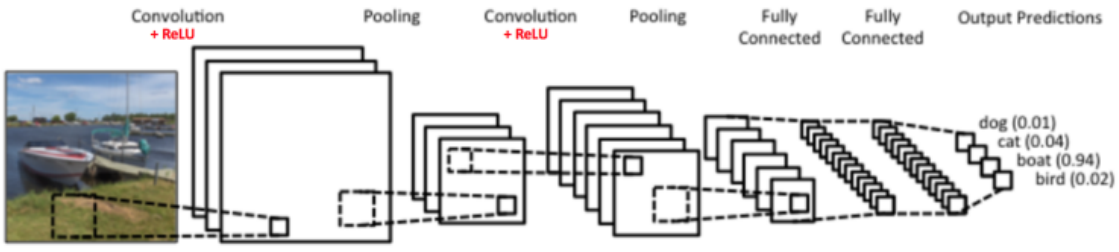

Abbildung 5: Schematischer Aufbau eines Konvolutionalen Netzwerkes
Quelle: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

[15] Goodfellow S 192
[16] Bishop S 267

Bei der Konvolutionsoperation wird ein Filter auf einen Teilbereich einer Matrix, bzw. einem Array an Matrizen, angewandt um einen neuen Wert zu berechnen. Durch das Schrittweise verschieben des Filters, dem sogenannten Stride, wird die ganze Matrix durchlaufen und eine neue Matrix generiert die komprimierte Information des Bildes enthält. Der Filter ist üblicherweise quadratischer Natur und umfasst n^2 Gewichte. In der Vergangenheit wurden viele dieser Filter per Hand entworfen um bestimmte Effekte bei der digitalen Bildverarbeitung zu erzielen, wie einen Glättungseffekt oder Kantenerfassung um beispielsweise abrupte Änderungen in dem Bild aufzuspüren. Allerdings anstelle Filter von Hand zu entwerfen benutzen wir das Maschinelle lernen um Filter zu entdecken, die genau für die Aufgabenstellung relevante Informationen extrahieren. Die nachfolgende Abbildung zeigt die Konvolutionsoperation mit einem 3x3 Filter. Die Anzahl der Parameter umfasst somit 9. Hinzukommt noch ein Bias Parameter und somit erhöht sich die Anzahl der Parameter auf 10. Die mathematische Vorschrift um den Wert der neuen Matrix an der Stell i,j zu berechnen lautet:

$$S(i,j) = (I * K)(i,j) = \sum_n \sum_m I(i+m, j+n) * K(m,n)$$

$S(i,j)$ entspricht den berechneten Wert der generierten Matrix an Stelle $S_{i,j}$.

I entspricht der Input Matrix bzw. der Eingabe der Schicht.

K entspricht dem Filter, der in der englischen Literatur auch als Kernel bezeichnet wird.

i,j bezeichnen die jeweilige Reihe und die jeweilige Spalte, an der die Operation durchgeführt werden soll. Die zuvor festgelegte Schrittweise gibt an, welcher Wert auf i und j hinzuaddiert wird und die nächste Operation berechnet werden soll.

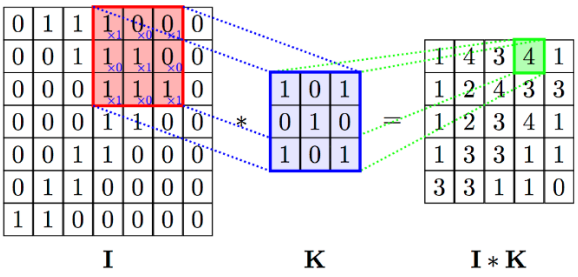

Abbildung 6 Konvolutionsoperation

Quelle: https://github.com/PetarV-/TikZ/tree/master/2D%20Convolution

Würde in der Praxis nur einen Filter pro Inputmatrix verwendet werden, würde es zu einer starken Unteranpassung kommen. Neun Parameter, bzw. 10 Parameter, sind nicht ausreichend genug um genügend Informationen aus der Matrix zu extrahieren um eine brauchbare Analyse durchführen zu können. Deshalb werden mehrere Filter gleichzeitig angewendet und die resultierende Matrix wird ein Array an Matrizen mit der Länge von der Anzahl der Filter. Die erste Matrix, das ursprüngliche Bild, ist häufig auch ein Array an Matrizen da die einzelnen

Kanäle für den rot, grün und blauen Farbton separat gespeichert werden. Im Falle eines Graubildes handelt es sich um ein allerdings um eine einfache Matrix.

Nachdem jede Konvolutionsoperation in der Schicht vollzogen wurde wird auf jeden berechneten Wert ein Bias Parameter hinzuaddiert und eine Aktivierungsfunktion auf die berechneten Werte angewendet.

Eine weitere Operation ist die Poolingoperation. In ihr wird auf den zuvor berechneten Schritt ein weiterer Filter angewendet, der die Werte gegenüber kleinen Änderungen weniger anfällig machen soll. [17] Die Poolingoperation erreicht dieses Verhalten dadurch, dass der Filter je nach Wahl den Mittelwert oder den Maximalwert zurückgibt.

[17] Goodfellow 357

Literaturverzeichnis

// Wird später ausgefüllt… Es sind noch die alten Daten von dem Muster drin.

Albers, S./Peters, H. (2007): Emission Trading – Strategy Implications for Airlines, in: Association for European Transport (Hrsg.): Proceedings of the 2007 European Transport Conference, Noordwijkerhout.

Brack, D./Grubb, M., Windram, C., et al. (2000): International trade and climate change policies, London, Earthscan.

Bauchmüller M./Gammelin, C. (2010): Deutscher Streit blockiert Klimaplan/Emissionshandel für Flugzeuge auf der Kippe, in: Süddeutsche.de, 17.05.2010, Online im Internet: http://www.sueddeutsche.de/wirtschaft/emissionshandel-fuer-flugzeuge-auf-der-kippe-deutscher-streit-blockiert-klima plan-1.219438, abgerufen am 17.02.2015.

Brack, D./Grubb, M., Windram, C., et al. (2000): International trade and climate change policies, London, Earthscan.

Coase, R. H. (1960): The Problem of Social Cost, in: Journal of Law and Economics Vol. 3. (Oct. 1960), S. 1-44.

o.V. (2011): Opel kämpft um Bochum, in: Frankfurter Allgemeine Zeitung, Nr. 15, 18.03.2011, S. 5.

Europäische Union (2009): Amtsblatt der Europäischen Union ABl. L 103 vom 23.4.2009, S. 10-29 (DE), Online im Internet: http://eur-lex.europa.eu/legal-content/DE/TXT/PDF/?uri= CLEX:32 00 9D0339&rid =1, abgerufen am 17.02.2015.

Fees, E. (2015): Art. Coase-Theorem, in: Gabler Wirtschaftslexikon/Das Wissen der Experten, Online im Internet: http://wirtschaftslexikon.gabler.de/Definition/coase-theorem.html, abgerufen am 17.02.2015.

International Civil Aviation Organisation (2007): ICAO Environmental Report 2007, Online im Internet: http://www . icao.int/environmental-protection/Documents/Env_Report_07.pdf, abgerufen am 18.02.2015.

Umweltbundesamt (2015): Der Europäische Emissionshandel, Online im Internet: http://www.umweltbundesamt.de/daten/klimawandel/der-europaeische-emissionshandel, abgerufen am 19.02.2015.

Eidesstattliche Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe angefertigt und mich anderer als der in den beigefügten Verzeichnissen angegebenen Hilfsmittel nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Der Durchführung einer elektronischen Plagiatsprüfung stimme ich hiermit zu. Die eingereichte elektronische Fassung der Arbeit entspricht der eingereichten schriftlichen Fassung exakt. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht. Ich bin mir bewusst, dass eine unwahre Erklärung rechtliche Folgen hat.

Ort, den Vorname, Name