Maschinelle Bilderkennung mit Big Data und Deep Learning

Einblicke in die Königsdisziplin

Die Arbeit mit unstrukturierten Daten dient gerne als Paradebeispiel für Big Data, weil die technologischen Möglichkeiten das Speichern und Verarbeiten großer Datenmengen erlauben und die Mehrheit dieser Daten unstrukturiert ist [1]. Allerdings ist im Zusammenhang mit unstrukturierten Daten meist von der Analyse und der Extraktion von Informationen aus Texten die Rede. Viel weniger hingegen wird das Thema der Bildanalyse thematisiert. Diese gilt aber nach wie vor als eine Königdisziplin der modernen Computerwissenschaft.

AUTOR: DIMITRI GROSS

Wie unstrukturierte Texte, so tragen auch Bilder zahlreiche Informationen in sich. Ein Mensch kann diese Zusammenhänge sofort erkennen und Erkenntnisse daraus für sich ableiten. Maschinen haben es da schwerer und benötigen für die Bildanalyse komplexe Verarbeitungsmechanismen. Vor allem brauchen sie hierfür aber auch zwei technologische Grundbedingungen: Sie brauchen eine skalierbare Plattform zum Speichern der Bilder und eine ebenfalls skalierbare und objektunabhängige Methode zur Extraktion der Merkmale aus den Bildern.

Eine maschinelle Bildanalyse teilt sich in viele Unterbereiche auf, vom einfachen Sortieren nach Farbstimmungen bis hin zum komplexen Themen, die heutzutage unter Computer Vision zusammengefasst werden. Auf drei wichtige Bereiche gehen wir näher ein: ähnliche Bilder anzeigen, Objekterkennung und Objektklassifizierung.

Fangen wir mit der einfachsten Problemstellung in der Bildanalytik an, nämlich dem Zeigen von allen ähnlichen Bildern in einer Datenbank. Hierzu machen sich die Entwickler Hashing- und Histogrammverfahren zu Nutze. Sie erstellen pro Bild ein Histogramm und kodieren dieses entweder in einem Vektor oder einem Hash. Anschließend clustern sie die sortierten Vektoren entsprechend ihrem Abstand zueinander und errechnen aus diesen Clustern einen Index. LIRE, ein Plug-in für Lucene, funktioniert z. B. auf diese Weise und ermöglicht so eine Bildersuche nach ähnlich aussehenden Bildern (Lucene Image Retrieval). Dabei geht LIRE nach einem fest definierten Regelsatz vor. Das System unterscheidet nicht zwischen einem Pferd und einem Menschen, sondern abstrahiert die Farbstimmung des Bilds.

In der Bildanalytik geht es aber noch deutlich komplexer. So benötigt ein Programm für die Objekterkennung eine genaue Vorgabe und muss wissen, nach welchen Alleinstellungsmerkmalen (Features) in einem Bild gesucht werden soll. Damit nur eine bestimmte Art von Objekten gefunden wird, gibt ein Entwickler den Algorithmus vor, mit dem Features aus einem Bild extrahiert werden. So werden beispielsweise für eine Ge-

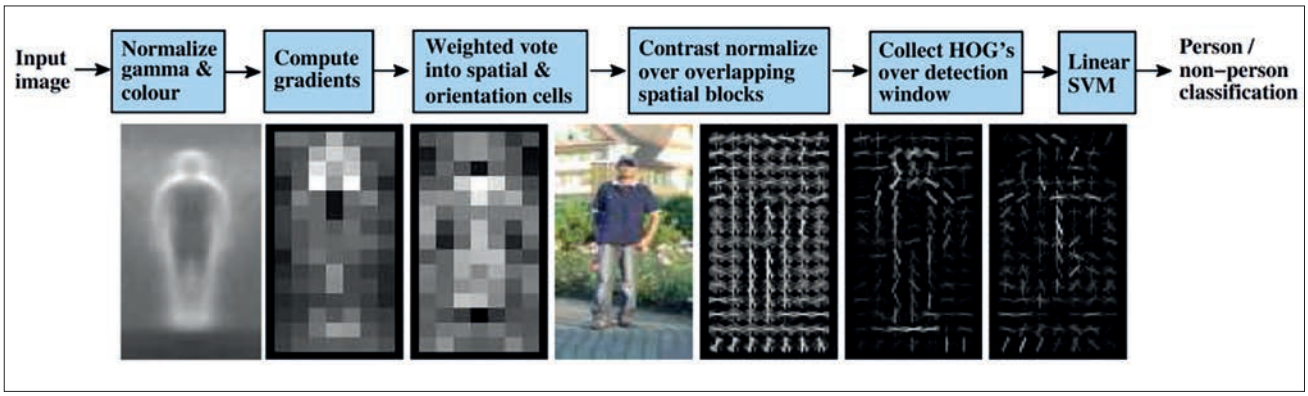

Abb. 1: Dedizierter Feature-Extraction-Algorithmus für Menschenerkennung [2]

sichtserkennung mehrere Merkmale definiert, die ein menschliches Gesicht ausmachen. Meist sind das Hell-Dunkel-Unterschiede (Haar Cascades), die der Maschine eine Schablone für die Suche vorgeben. Findet der Algorithmus in einem Bild eine gewisse Anzahl an Übereinstimmungen mit dieser Featureschablone, wird das Bild als *TRUE* klassifiziert. Zur Klassifizierung der gefundenen Features kommen meist Support Vector Machines (SVM) zum Einsatz. OpenCV ist eine bekannte Bibliothek, die viele solcher Objekterkennungsverfahren zusammenfasst. So lässt sich beispielsweise eine Gesichtserkennung ohne großen Aufwand implementieren.

Der anspruchsvollste Bereich in der Bildanalytik ist die Objektklassifizierung. Sie benötigt einen Algorithmus, der robust genug ist, alle Merkmale eines Objekts wiederzuerkennen, und zwar unabhängig vom Bildhintergrund.

MACHINE LEARNING UND DEEP LEARNING

Schon früher haben Experten versucht, die Merkmale eines Bilds von Hand vorzugeben und manuell zu definieren. Doch das funktionierte nur für rudimentäre Aufgaben, wie das Finden eines runden Objekts auf einem eintönigen Hintergrund. Während ein Mensch einem anderen Menschen problemlos in wenigen Sätzen ein neues Objekt verständlich beschreiben kann, war eine Maschine bislang nicht in der Lage, ein solches Objekt eigenständig zu identifizieren.

Viele Jahre wurde bereits in Featurealgorithmen investiert, die aus einem Bildobjekt Muster extrahieren. Das Lösen bestimmter Teilprobleme der Bilderkennung funktioniert mit diesen auch sehr gut (**Abb. 1**). Sie können zwar Gesichter erkennen oder menschliche Silhouetten unterscheiden, aber nicht beides zu einem Gesamteindruck zusammenfügen. Es fehlte also noch ein Verfahren, das ein Programm in die Lage versetzt, die Features aus einem Bild ohne vorgegebene Logik selbstständig zu extrahieren und mit einem Label zu verknüpfen. Und das immer wieder für jedes beliebige Objekt auf einer beliebigen Anzahl von Bildern.

Möchte man also beliebige Bilder klassifizieren, ohne das zugrunde liegende Verfahren zu verändern, braucht man eine dynamische und für jede Objektklasse gesonderte Extraktion von Features. Bis vor einigen Jahren war das noch nicht möglich. Erst seit 2010 wurde umfangreich mit gefalteten neuronalen Netzwerken experimentiert. Deren Schwerpunkt liegt darin, jedes einmal eintrainierte Objekt in vielen Variationen wiederzuerkennen, indem das Netzwerk repräsentative Features aus Bildern extrahiert und mit jeder weiteren Faltungsstufe die Low-Level-Features zu einer High-Level-Feature-Map verdichtet [3].

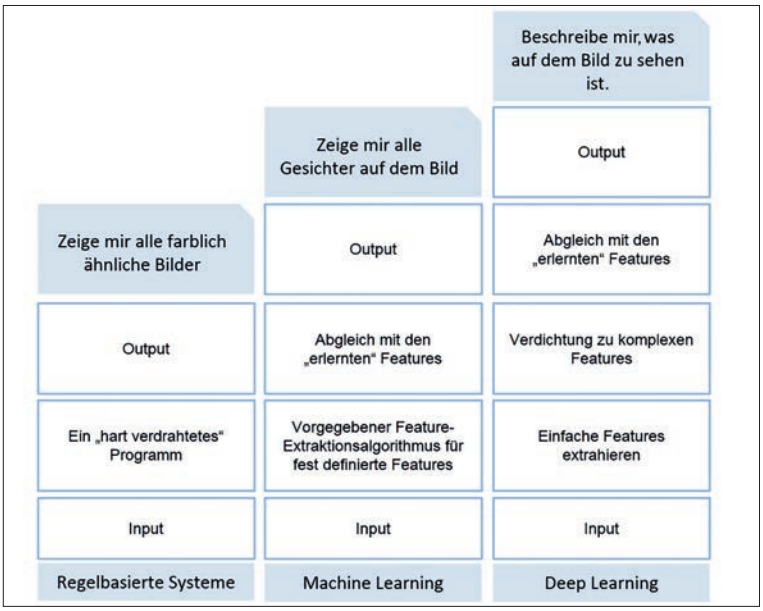

Abb. 2: Regelbasierte Systeme vs. Machine Learning vs. Deep Learning [4]

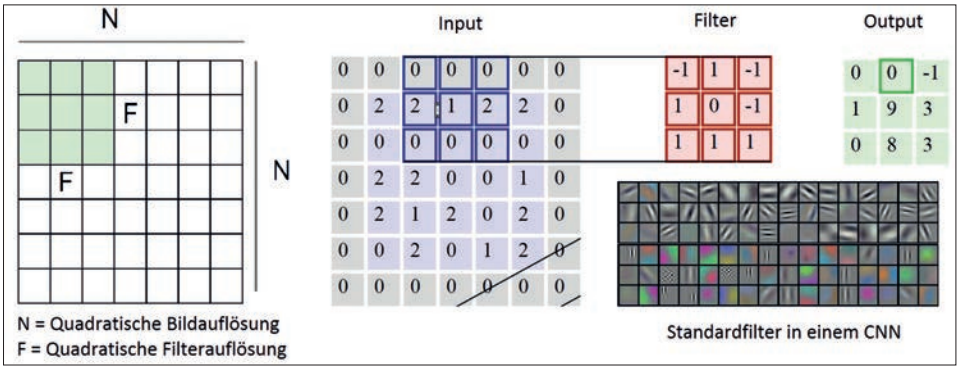

Abb. 3: Anwendung eines Filters im Convolutional Neural Network [3]

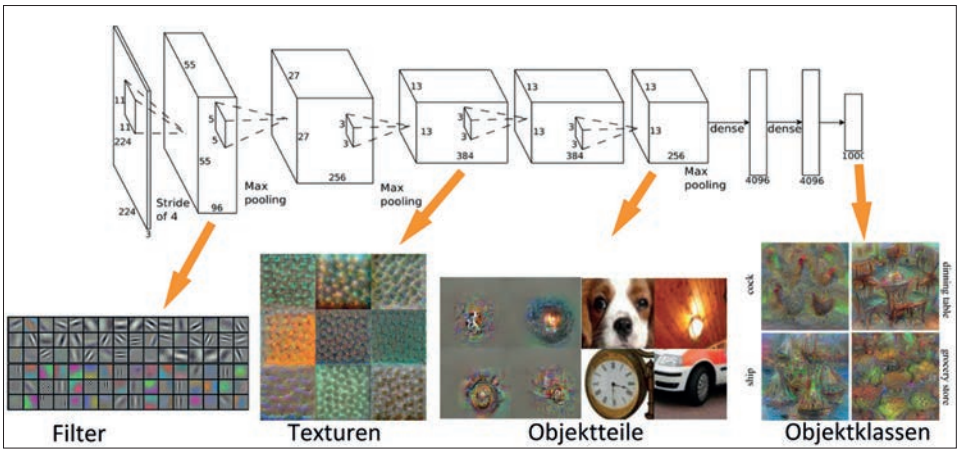

Abb. 4: Darstellung einer CNN-Architektur am Beispiel von AlexNet [7]

Abbildung 2 zeigt Unterschiede zwischen regelbasierten Systemen, Machine Learning und Deep Learning. So kann jedes Verfahren, das einen dedizierten Algorithmus zur Featureextraktion benutzt, um eine bestimmte Aufgabenstellung zu lösen, dem Bereich Machine Learning zugeordnet werden. Dazu gehört z. B. das Erkennen von bestimmten Objekten, wie Autos oder Fußgänger. Beim Deep Learning kennt das System nicht die repräsentativen Features eines Objekts, sondern lernt nach und nach, diese Features selbst zu extrahieren. Mit der Zeit entwickeln solche Systeme ein robustes Modell und können damit beliebige Objekte selbstständig erkennen.

Die heutige Technologie macht es möglich, seit Langem bekannte theoretische Ansätze in die Praxis umzusetzen. So können Big-Data-Experten heute ein Convolutional Neural Network (CNN) mithilfe von Millionen von Bildern trainieren und für die Klassifizierung von Tausenden von Objekten nutzen [5].

WIE FUNKTIONIERT EIN GEFALTETES NEURONALES NETZ?

Bei einem kurzen Blick auf ein Bild erkennt unser Gehirn zunächst Konturen und Farbmuster, die in unserem Gedächtnis mit bekannten Formen in Assoziation gebracht werden. So erkennen wir seit Kindheit bekannte Objekte wieder. Nach diesem Prinzip lässt sich unser Gehirn auch leicht täuschen. Das beweisen optische Darstellungen, die uns z. B. sowohl eine Vase als auch zwei Gesichtssilhouetten erkennen lassen.

Ein gefaltetes neuronales Netz wurde dem visuellen Cortex einer Katze nachempfunden. Seine Besonderheit liegt darin, dass die Bildsignale zunächst ein komplexes Netz an Neuronen durchlaufen, bevor sie im Gehirn weiter verarbeitet werden. Zusätzlich wurden zwei Zelltypen gefunden, die für das Extrahieren von Features zuständig sind. Ein Typ reagiert gut auf Kanten innerhalb eines mit dieser Zellgruppe verknüpften Sichtbereichs. Diese Zellen reagieren auf kleine Ausschnitte im Sichtfeld und fungieren somit als eine Art Filter. Der andere Zelltyp reagiert gut auf komplexere Muster, und zwar unabhängig von deren Position im Sichtfeld [6].

FILTER ANWENDEN

Bei einem CNN wird im ersten Schritt eine Reihe von unterschiedlichen Filtern auf ein Bild angewandt (**Abb. 3**). Diese können Konturen in einem Bild erkennen oder Farbmuster extrahieren. Dafür wird ein kleiner Filter (z. B. drei mal drei Pixel groß) iterativ jeweils um ein Pixel nach rechts und um eine Filterlänge (F) nach unten bewegt. Dabei errechnet das System bei jedem Iterationsschritt ein Skalarprodukt der Pixel, die unter dem Filter liegen. Diese Vorgehensweise wird wiederholt, bis alle für den ersten Schritt definierten Filter angewandt wurden (z. B. 96).

ACTIVATION MAPS AKTIVIEREN

Die mithilfe der Filter errechneten Werte werden in einem neuen Bild gespeichert. Dann wird eine so genannte Activation Map aktiviert, die besonders gut auf eine Art von Filter reagiert und somit die markanten Formen stark hervorhebt. Als Ergebnis der ersten Faltungsstufe wird an die nachfolgende Faltungsstufe eine entsprechende Anzahl an Activation Maps weitergereicht. Die Anzahl der Ac-

tivation Maps entspricht der Anzahl der Filter, also z. B. 96 Stück. Die Bildgröße (proportional X=Y=N) verringert sich bei jedem Faltungsschritt um *(N-F)/Schrittgröße des Filters+1*. Diese Operationen werden wiederholt, bis alle Ebenen des CNNs durchlaufen wurden.

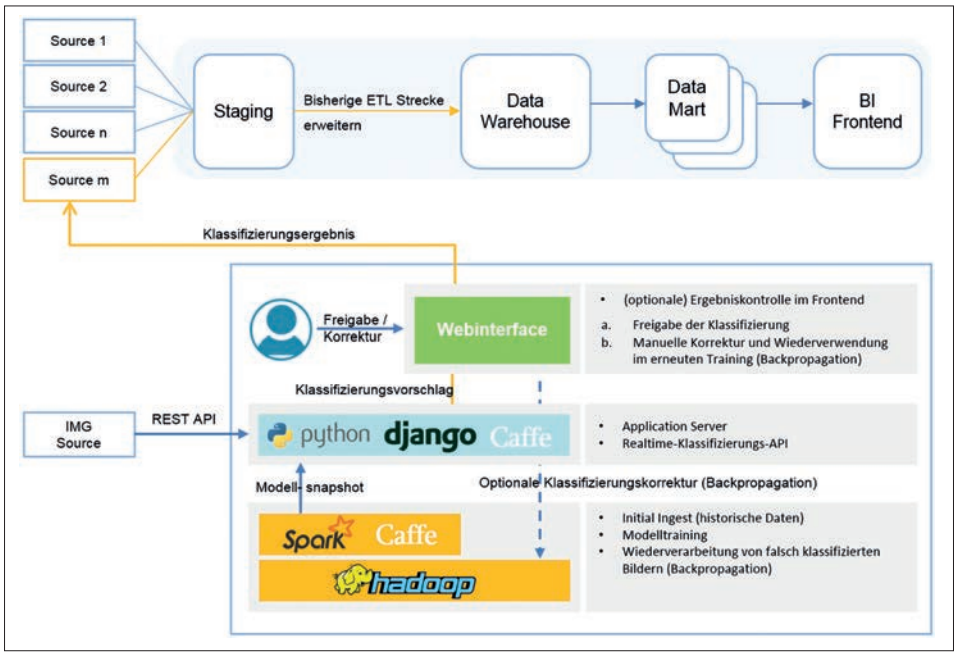

Abb. 5: Darstellung einer möglichen Architektur auf Basis von Caffe und Apache Spark mit Hadoop

OBJEKTKLASSEN ABBILDEN

Im letzten Schritt (Abschnitt *Objektklassen* in **Abb. 4**) werden die gewonnenen Features aus jeder Faltungsstufe in einer Objektklasse abgebildet. Die stärksten Aktivierungen werden hier höher gewichtet. Eine Objektklasse enthält somit sämtliche markanten Objektteile von allen Bildern, die zu dieser Objektklasse gehören. Ähnlich entstehen auch die surrealistischen Muster, die dem Werk von Salvador Dalí entstammen könnten, in denen alle möglichen Variationen von einem Objekt über Zeit in einer High-Level-Feature-Map zusammengefasst werden.

SKALIERBARKEIT UND SPEICHEREFFIZIENZ

Besonders im Bereich der Bildanalytik, wenn es nach dem Prinzip von gefalteten neuronalen Netzwerken um das Lernen in mehreren Iterationen geht, spielen Skalierbarkeit und Speichereffizienz eine wichtige Rolle. Beispielsweise erfordert ein kompletter Durchlauf eines CNNs mit einem Bild je nach Architektur des CNNs, je nach Anzahl der angewendeten Filter und je nach Bildgröße Speicherplatz im dreistelligen MB-Bereich [3]. Um dem CNN eine Objektklasse beizubringen, damit das Objekt auch bei Störfaktoren und wechselnden Hintergründen identifiziert werden kann, benötigt man sehr viele Bilder. Als Beispiel kann der ILSVRC2010 herangezogen werden. Beim ILSVRC2010 geht es darum, ein CNN mit 1,2 Millionen Bildern für das Unterscheiden von 1 000 Objektkategorien zu trainieren. Ein zusätzlicher Faktor für den Speicherbedarf sind die Iterationen, die für jedes einzelne Bild notwendig sind. Auf diese Weise summieren sich Rechenleistung und Speicherkapazität auf eine Größe, die nur mithilfe von modernen Skalierungsverfahren angegangen werden können [5].

Viele Deep-Learning-Frameworks berücksichtigen diesen Umstand bereits von Anfang an. Drei Frameworks sind hier besonders weit verbreitet: Torch von Facebook, TensorFlow von Google sowie Caffe von der Berkeley University. Diese Tools sind darauf spezialisiert, die Berechnungen auf eine GPU auszulagern. Die Obergrenze für die vertikale Skalierbarkeit ist durch den Einsatz von GPUs zwar sehr hoch, jedoch ist sie auch schnell erreicht. Aus diesem Grund veröffentlichte Yahoo! Anfang 2016 eine auf Apache Spark optimierte Implementierung von Caffe [8]. CaffeOnSpark skaliert nicht nur vertikal über die GPU, sondern auch horizontal über Spark.

ON PREMISE VERSUS CLOUD

Bildanalysen lassen sich On Premise oder über Cloud-APIs umsetzen. Ein Beispiel, das über die allgemeine Unterscheidung von Objekten, wie zwischen einer Katze und einem Hund, hinausgeht, verdeutlicht die Vorteile einer On-Premise-Lösung gegenüber einem Cloud-API. Bildklassifizierung soll den Prozess im Schadensmanagement einer KFZ-Versicherung optimieren. Dafür soll ein System die Schadenshöhe von Versicherungsfällen auf der Grundlage eines Schadensfotos automatisiert abschätzen. Zur Lösung dieses nicht trivialen Problems bedarf es eines speziell auf Schadenserkennung trainierten Modells. Dieses wird im Rahmen eines Modelltrainings mit anschließendem Feintuning aufgebaut. Das Modell ist am Ende in der Lage zu erkennen, welche Teile des Autos beschädigt wurden, und kann dem Vorgang eine grobe Schadenshöhe zuordnen.

Dieser Lösungsvorschlag besticht durch seine Flexibilität. Das System lässt sich wie ein REST-API nutzen

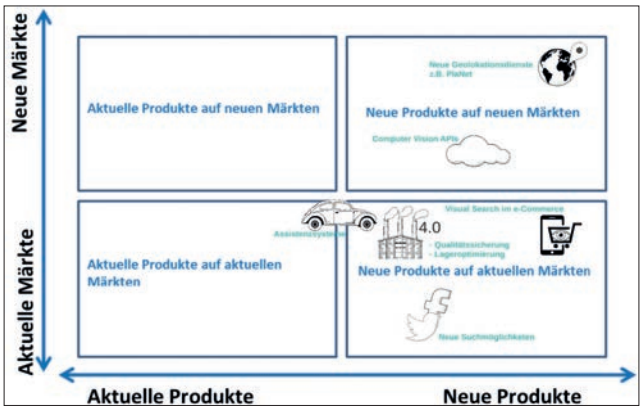

Abb. 6: Darstellung von geschäftlichen Potenzialen der Bildanalytik auf einer Produktmarktmatrix (nach Harry Igor Ansoff)

oder aber zur Optimierung der Ergebnisse auch mit einem Frontend ausstatten, damit ein Anwender die Gültigkeit der Klassifizierungsvorschläge überprüfen und gegebenenfalls korrigieren kann. Die korrigierten Klassifizierungen lassen sich dann wiederum zur Verbesserung der Lernergebnisse verwenden. CaffeOnSpark kommt in dem Beispiel zum Einsatz, als Alternative zu teuren dedizierten GPU-Clustern. So kann die gleiche Hardware, die für die Worker im Hadoop-Cluster verwendet wird, auch zum Training des CNN genutzt werden. Wie sich das auf die sonstige Clusterperformance auswirkt, sollte individuell geprüft und so konfiguriert werden, dass die Rechenlast, die durch Caffe erzeugt wird, nicht zu 100 Prozent die CPU allokiert.

Ein Cloud-API würde in diesem Fall nicht zum gewünschten Ergebnis führen, da der generalistische Ansatz hier nicht die genaue Detailtiefe in der Klassifizierung liefern kann. Dafür sind die aktuell verwendeten Modelle der Cloud-APIs noch nicht umfangreich genug. Google, IBM und Microsoft bieten Vision-APIs an, mit deren Hilfe sich Bilder klassifizieren und in natürlicher Sprache beschreiben lassen. Tabelle 1 listet die ausschlaggebenden Merkmale einer On-Premise-Lösung und eines Cloud-API auf.

Listing 1: Caffe per Shell-Befehle installieren

```
wget http://repo.continuum.io/archive/Anaconda2-4.1.1-Linux-x86_64.sh
bash Anaconda2-4.1.1-Linux-x86_64.sh
wget http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1404/
                                    x86_64/cuda-repo-ubuntu1404_7.5-18_amd64.deb
sudo dpkg -i cuda-repo-ubuntu1404_7.5-18_amd64.deb
sudo apt-get update
sudo apt-get install cuda
sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev
                                                    libhdf5-serial-dev protobuf-compiler
sudo apt-get install --no-install-recommends libboost-all-dev
sudo apt-get install libatlas-base-dev
sudo apt-get install libgflags-dev libgoogle-glog-dev liblmdb-dev
export PATH="/home/username/anaconda2/bin:$PATH"
conda update conda
wget https://github.com/BVLC/caffe/archive/master.zip
cp Makefile.config.example Makefile.config
#Nun soll Makefile.config angepasst werden
#Da Caffe in einer VM laufen wird, kann es nicht die GPU-zum Berechnen nutzen
#Es soll daher der CPU-Only-Mode eingeschaltet werden
CPU_ONLY := 1
#Um auf Anaconda umzustellen, sollen Python und Lib Path angepasst werden
ANACONDA_HOME := $(HOME)/anaconda2
PYTHON_INCLUDE := $(ANACONDA_HOME)/include \
                  $(ANACONDA_HOME)/include/python2.7 \
                  $(ANACONDA_HOME)/lib/python2.7/site-packages/numpy/
                                                                  core/include \
PYTHON_LIB := /usr/lib
make all –jXX (XX steht für die Anzahl der verfügbaren CPU-Kerne)
make pycaffe
export PYTHONPATH=/home/username/caffe/python
conda install protobuf
```

Kriterium	On Premise (Caffe)	Cloud-API (Google Vision)
Anschaffungskosten für Infrastruktur und Implementierung	Vergleichsweise hoch	Keine
Laufende Kosten	Relativ hoch, da Hardware und die Wartung der Infrastruktur den größten Kostenblock stellen	„Pay as you go“ mit Kontingentstaffeln
Spezifischer Know-how-Aufbau oder Einkauf	Erforderlich	Nicht erforderlich
Flexibilität	Sehr flexibel durch eine große Auswahl an vortrainierten Modellen, die auf viele Problemstellungen optimiert wurden, und die Möglichkeit der Anpassung eines Modells (Finetuning)	Bedingt flexibel, weil nur ein generalistisches Modell für jede Art von Bildern verwendet wird

Tabelle 1: Vergleich Flexibilität vs. Kostenfaktor eines Deep-Learning-Systems in der Cloud oder On Premise

Wer Caffe lokal in einer VM unter Ubuntu ausprobieren möchte, kann sich über folgende Shellbefehle alle nötigen Packages für Caffe installieren, um anschließend eine lokal anwendbare Caffe Library zu kompilieren.

WAS BRINGT DIE ZUKUNFT?

Selbstfahrende Autos sind heute keine Utopie mehr. Es gibt Systeme, die anhand eines Bilds die Geokoordinaten der Stelle berechnen können, an denen das Bild aufgenommen wurde. Dies funktioniert durch die enorme Rechenleistung der heutigen GPUs und durch die Möglichkeit, die Rechenlast horizontal zu skalieren. Speichermöglichkeiten und Skalierung gepaart mit der Möglichkeit, künstliche neuronale Netzwerke mit enorm hohen Datenmengen zu trainieren, machen diese sehr robust gegenüber Störfaktoren in einem Bild und liefern eine gute Erkennungsrate. **Abbildung 6** zeigt anhand einer Produktmarktmatrix, wie im Wirtschaftsbereich mithilfe von Deep Learning und Bildanalytik neue Märkte und Produkte entstehen.

FAZIT

Unstrukturierte Daten in Verbindung mit Deep Learning brachten in den letzten Jahren neue Geschäftsideen hervor. Es gibt eine starke Tendenz in Richtung Prozessautomatisierung. Bildanalytik und generell Computer Vision werden dabei eine wichtige Rolle spielen. So gibt es bereits komplett neue Geschäftsbereiche, die mit dieser Technologie geschaffen wurden. Wie sich der Trend bei den Infrastrukturen weiterentwickelt, wird die Zukunft zeigen. Dass die Cloud-First-Strategie auch in diesem Bereich Einzug halten wird, lassen viele erfolgreiche Deep-Learning-Start-ups bereits jetzt erahnen.

Links & Literatur

[1] Persistent Forecasting of Disruptive Technologies, National Academies Press, 2010 Committee on Forecasting Future Disruptive Technologies, Air Force Studies Board, Division on Engineering and Physical Sciences, National Research Council

[2] Histograms of Oriented Gradients for Human Detection: https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

[3] CS231n Convolutional Neural Networks for Visual Recognition: http://cs231n.github.io/convolutional-networks/

[4] Theoretical Motivations for Deep Learning: http://rinuboney.github.io/2015/10/18/theoretical-motivations-deep-learning.html

[5] Large Scale Visual Recognition Challenge 2010: http://image-net.org/challenges/LSVRC/2010/

[6] Convolutional Neural Networks (LeNet) DeepLearning 0.1 documentation: http://deeplearning.net/tutorial/lenet.html

[7] mNeuron: A Matlab Plugin to Visualize Neurons from Deep Models: http://vision03.csail.mit.edu/cnn_art/index.html

[8] Caffe on Spark: https://github.com/yahoo/CaffeOnSpark

[9] Dormehl, Luke: „This scrappy Russian startup is beating Google and Facebook at facial recognition": http://www.digitaltrends.com/cool-tech/ntechlab-facial-recognition-algorithm/

[10] „Google Unveils Neural Network with 'Superhuman' Ability to Determine the Location of Almost Any Image": https://www.technologyreview.com/s/600889/google-unveils-neural-network-with-superhuman-ability-to-determine-the-location-of-almost/

Dimitri Gross

arbeitet als Senior Consultant bei OPITZ CONSULTIG Deutschland GmbH. Als Kernteammitglied im Competence Center Big Data beschäftigt er sich mit Big-Data-Architektur, Werkzeugauswahl, Lösungsdesign und Aufbauorganisation in Big-Data-Projekten und unterstützt seine Kunden darüber hinaus in analytischen Fragestellungen.

Beispiele für neue Geschäftsmodelle

- Computer-Vision-API-Anbieter: Hier gibt es zahlreiche Produkte, von spezialisierten Systemen für Gesichtserkennung bis zu allgemeinen Systemen, die Objekte im Bild klassifizieren und die Positionsdaten des Objekts auf dem Bild zurückliefern. Google Vision API, IBM AlchemyAPI oder Face++ sind nur einige davon.
- E-Commerce: Hier geht es vielfach um neue User Experience bei der Produktsuche mit einem Foto, z. B. bei der Suche nach ähnlichen Schuhen in einem Onlineshop. Der Benutzer macht ein Foto von einem Produkt seines Interesses und die Suche zeigt das gesuchte Produkt an oder schlägt eine möglichst ähnliche Alternative vor. Slyce ist ein bekannter Anbieter in diesem Bereich.
- Industrie 4.0: In Produktionsstätten kann Bildanalytik der Qualitätssteigerung dienen, z. B. durch die Feststellung von Temperaturverteilungsmustern in einem Gussteil außerhalb gewisser Toleranzen.
- Social Media: Neue Suchmöglichkeiten, wie eine Rückwärtssuche anhand eines Fotos, eröffnen z. B. in einer Singlebörse neue Geschäftsmodelle. Auch im Securitybereich findet dieses System bereits Anknüpfungspunkte [9].
- Geolocation: Neue Erkenntnisse über die Zuordnung von Ortsinformationen zu einem Bild lassen sich durch Deep Learning erreichen. Experten bei Google stellten z. B. eine Möglichkeit der Geolocation basierend auf einem CNN vor, die Geokoordinaten anhand eines beliebigen Bilds mit einer hohen Genauigkeit errät. Hierfür wurde ein spezielles CNN namens PlaNet entwickelt [10].