V13 ML-Klassifikation – Teil 2

- Balancierte Trainingsdaten: oversampling vs undersampling
- Random Forest-Methode
- Grundlegende Methoden im Deep Learning
- Ein Beispiel zum Einsatz von aktuellen Deep Learning-Methoden

Problem: unbalancierter Datensatz

Für beide Klassen sollte eine ähnliche Anzahl an Trainings- und Testdaten vorliegen.

Falls dies nicht der Fall ist (es z.B. neun mal so viele Datenpunkte in einer Klasse als in der anderen gibt), wird der ML-Predictor vermutlich so trainiert, dass er stets die größere Klasse ausgibt.

Damit würde zwar eine recht hohe Genauigkeit von ca. 90% erzielt.

Der Predictor wäre aber gar nicht dafür trainiert, die Tendenzen für beide Klassen abzuwägen.

Hier werden eine Reihe von Algorithmen vorgestellt, mit denen man beim undersampling gezielter vorgehen kann:
https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/

Undersampling

Falls die beiden Klassen nicht balanciert sind, aber mehr als ausreichend Daten vorhanden sind, kann man von der größeren Klasse einfach eine zufällige Auswahl treffen, die der Anzahl an Datenpunkten in der kleineren Klasse entspricht. Das bezeichnet man als **random undersampling**.

Evtl. verliert man hierbei aber nützliche Information. Man kann auch gezielter vorgehen (siehe link im Kommentarfeld). Darauf gehen wir aber hier nicht ein.

In einer 10-fold cross validation kann man auch jedes Mal eine andere zufällige Auswahl treffen.

Meist kombiniert man undersampling der größeren Klasse mit oversampling der kleineren Klasse.

Hier werden eine Reihe von Algorithmen vorgestellt, mit denen man beim undersampling gezielter vorgehen kann:
https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/

Oversampling

Für beide Klassen sollte eine ähnliche Anzahl an Trainings- und Testdaten vorliegen.

Falls dies die Anzahl an Datenpunkten sehr begrenzt ist, so dass ein zuverlässiger Klassifikator kaum noch erzeugt werden kann, kann man für die kleinere Klasse synthethische zusätzliche Datenpunkte erzeugen.

Diese sollen sich von den bisherigen Datenpunkten unterscheiden, aber nicht einfach zufällig ausgewählt sein.

Eine häufig verwendete Methode heißt SMOTE:

SMOTE

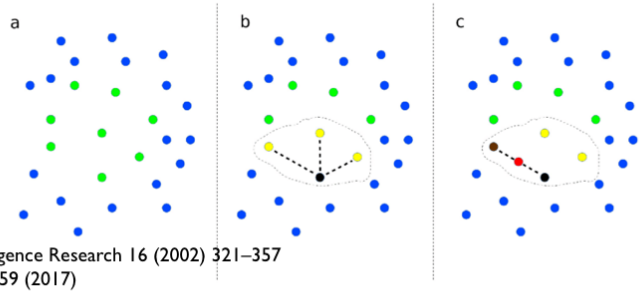
Wähle einen zufälligen Datenpunkt **a** aus der kleineren Klasse.

Identifiziere die k nächsten Nachbarn unter den Datenbanken in dieser Klasse.

Wähle einen Nachbarn **b** zufällig aus und berechne den Verbindungsvektor zu **a**: **b - a**

Multipliziere den Vektor mit einer Zufallszahl aus dem Intervall [0,1] und addiere das Ergebnis zu **a**.

Das ist der neue synthetisch erzeugte Datenpunkt. Er liegt auf der Verbindungsgerade zwischen 2 bekannten Datenpunkten.



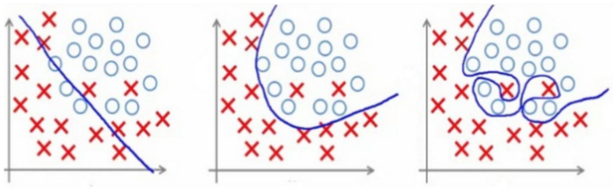
Chawla et al. Journal of Artificial Intelligence Research 16 (2002) 321–357
Schubach et al. Scientific Reports 7, 2959 (2017)

Overfitting in Entscheidungsbäumen

Für die Konstruktion von Entscheidungsbäumen benutzen wir greedy Algorithmen. Das spart zwar viel Zeit, liefert aber evtl. nicht die optimale Lösung für den gesamten Baum.

Ein mögliches Problem ist **overfitting (Übertraining)**. D.h. es werden zwar die Datenpunkte im Trainingsdatensatz sehr gut vorhergesagt, aber nicht die Datenpunkte im unabhängigen Testdatensatz.

Eine Lösung für dieses Problem besteht darin, die Ergebnisse aus vielen, leicht unterschiedlichen Entscheidungsbäumen zu kombinieren.



Under-fitting

(too simple to explain the variance)

Appropriate-fitting

Over-fitting

(forcefitting -- too good to be true)

https://towardsdatascience.com/

Link für Abbildung: https://towardsdatascience.com/a-visual-look-at-under-and-overfitting-using-u-s-states-7fd0d8ade053

bootstrap aggregation (bagging) von Entscheidungsbäumen

Bagging: man erzeugt eine Anzahl an Entscheidungsbäumen.

Jeder einzelne Baum wird mit einer unterschiedlichen Bootstrap-Untermenge an Trainingsdaten erzeugt.

Da jeder Entscheidungsbaum mit leicht unterschiedlichen Trainingsdaten erzeugt wurde, hat jeder Baum eine leicht unterschiedliche Performanz. Bagging ist daher ein effektiver Ensemble-Algorithmus.

Die Klassifikationsergebnisse der einzelnen Entscheidungsbäume werden gemittelt (Mehrheitsentscheidung). In der Summe erhält man dadurch stets eine bessere Performanz als mit jedem einzelnen Baum.

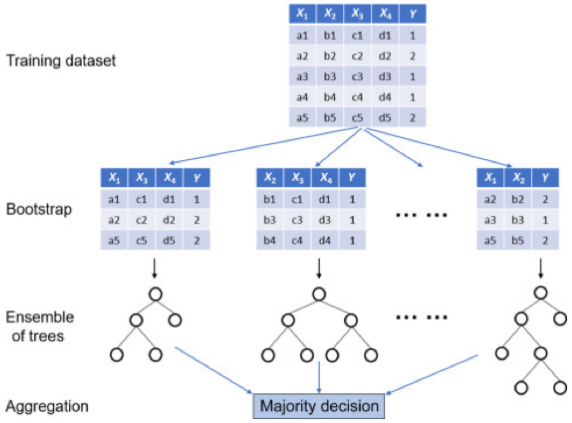
https://machinelearningmastery.com/random-forest-ensemble-in-python/

Random Forest

Ein Random Forest kombiniert (wie beim Bagging) die Vorhersagen von vielen einzelnen Entscheidungs-bäumen.

Wie beim Bagging werden die einzelnen Bäume auf unterschiedlichen Teilmengen der Trainingsdaten trainiert.

Im Unterschied zu Bagging können die Bäume bei der Random Forest-Methode unterschiedliche Splitknoten enthalten.



https://machinelearningmastery.com/random-forest-ensemble-in-python/
https://pages.cms.hu-berlin.de/EOL/geo_rs/S09_Image_classification2.html

Random Forest: Parameter beim Trainieren

Ein wichtiger Trainingsparameter ist die Anzahl an zufällig ausgewählten Features, die bei jedem Split-Knoten betrachtet werden.

Eine Empfehlung lautet (Breiman 2001), diese Anzahl gleich der Quadratwurzel aus der Anzahl an Eingabefeatures zu setzen.

Ein weiterer Hyperparameter ist die Tiefe (depth) der Bäume bzw. die maximale Anzahl an rekursiven Splitknoten.

Tiefere Bäume verstärken das Übertraining auf den Trainingsdaten, sind aber weniger korreliert zueinander.

Eine typische Zahl liegt zwischen 1 und 10.

Auch die maximale Anzahl an Bäumen wird üblicherweise beschränkt.

https://machinelearningmastery.com/random-forest-ensemble-in-python/

Vorteile der Random Forest-Methode

- Der Klassifikator trainiert sehr schnell: Dieser Vorteil ergibt sich durch die kurze Trainings- bzw. Aufbauzeit eines einzelnen Entscheidungsbaumes und dadurch, dass die Trainingszeit bei einem Random Forest linear mit der Anzahl der Bäume steigt.
- Die Evaluierung eines Testbeispieles geschieht auf jedem Baum einzeln und ist daher parallelisierbar. Er evaluiert also schnell.
- Er ist sehr effizient für große Datenmengen (viele Klassen, viele Trainingsbeispiele, viele Merkmale).
- Wichtige Klassen können erkannt werden.
- Der Zusammenhang zwischen Klassen kann erkannt werden.

www.Wikipedia.org

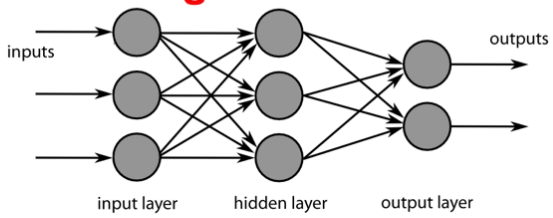
Deep Learning

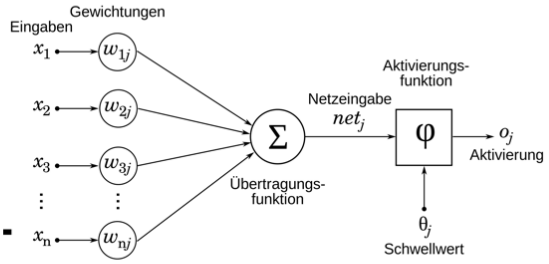
Deep Learning-Methoden basieren auf Neuronalen Netzwerken, die seit den 1960er Jahren entwickelt wurden.

„Deep“ sind diese Netzwerke, wenn sie eine große Anzahl an hintereinander geschachtelten *hidden layers* beinhalten.

Zwischen vielen Knoten existieren Kanten, deren Gewichtungen in einem Trainingsschritt trainiert werden müssen.

Für das Training sind daher sehr große Datenmengen erforderlich.



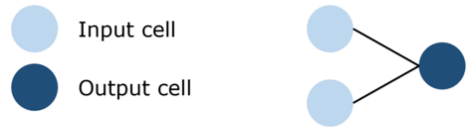


www.wikipedia.org

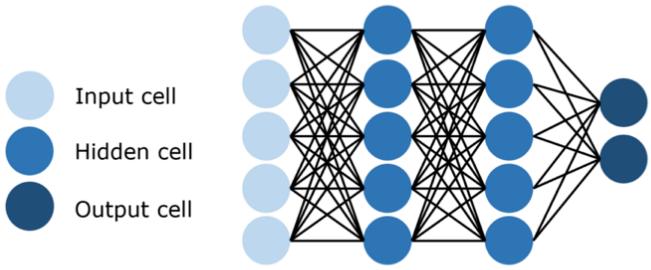
Die Abb. oben links zeigt ein sehr einfaches neuronales Netzwerk, das nur einen hidden layer enthält.
Die Abb. unten zeigt schematisch, wie durch eine gewichtete Summenbildung sowie ggf. eine anschließende mathematisch Operation (Aktivierungsfunktion mit Schwellwert) diese Summe weiterverarbeitet warden kann. Diese Topologie ist der Verschaltung von Neuronen im Menschen nachempfunden.

Typen von Neuronalen Netzwerken

Perceptron

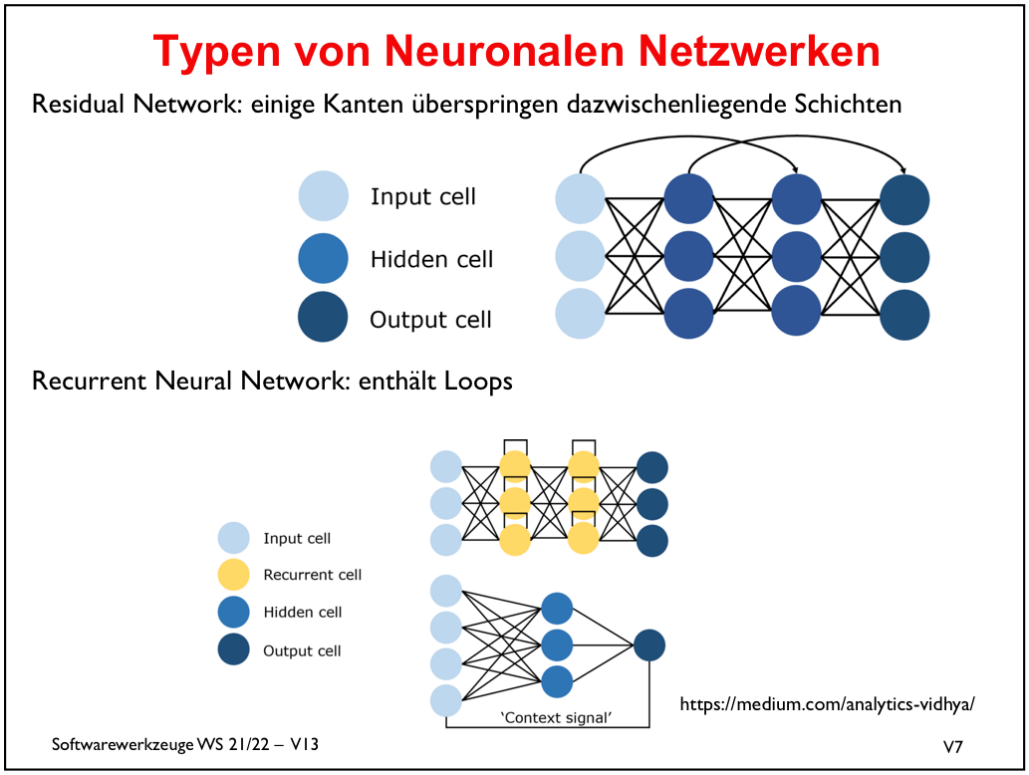


Feed-Forward Netzwerk: etliche Perceptrons werden kombiniert



https://medium.com/analytics-vidhya/

Bilder stammen von dieser Seite: https://medium.com/analytics-vidhya/11-essential-neural-network-architectures-visualized-explained-7fc7da3486d8
Das Perzeptron wurde zwischen 1955-57 von Frank Rosenblatt (1928-1971) entwickelt.



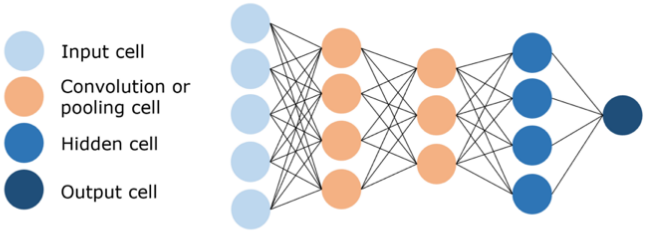
Bilder stammen von dieser Seite: https://medium.com/analytics-vidhya/11-essential-neural-network-architectures-visualized-explained-7fc7da3486d8

Ein Problem bei deep networks ist das aufwändige Training der Gewichte. Man versucht, Variationen der Parameter zu finden, die zu besseren Ergebnissen führen. Dazu muss man den Gradienten der Parameter bestimmen. Bei sehr tiefen Netzwerken wird dieser Gradient sehr, sehr klein. Eine deutliche Erleichterung des Trainings kam mit der Erfindung der Residual Networks. Die überspringenden Kanten führen einfache eine identity-Operation durch.

Typen von Neuronalen Netzwerken

Long Short Term Memory Network (populär für Spracherkennung)

Convolutional Neural Network (populär für Bilderkennung/Computer vision)



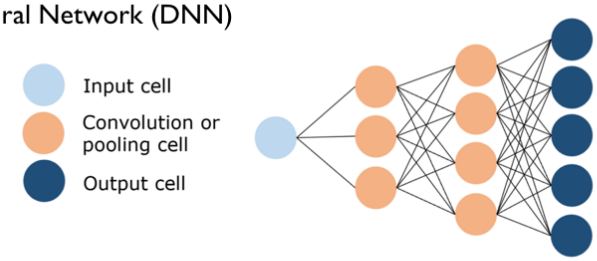
https://medium.com/analytics-vidhya/

Bilder stammen von dieser Seite: https://medium.com/analytics-vidhya/11-essential-neural-network-architectures-visualized-explained-7fc7da3486d8

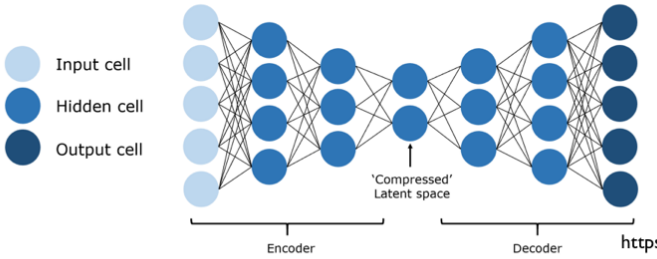
Aus Wikipedia:
1997 wurden LSTM-Netze von Sepp Hochreiter und Jürgen Schmidhuber in einer Veröffentlichung vorgestellt. „Beim Trainieren von künstlichen neuronalen Netzen werden Verfahren des Fehlersignalabstiegs genutzt, die man sich wie die Suche eines Bergsteigers nach dem tiefsten Tal vorstellen kann. Bei mehreren vertiefenden Schichten kann dies zu kurz greifen, so wie ein vergesslicher Bergsteiger beim Abstieg im ersten besten Tal landet und sein Dorf in einem tieferen Tal nicht finden kann. Das LSTM-Verfahren löst dieses Problem, indem es für eine LSTM-Zelle zur besseren Erinnerung drei Torsorten verwendet: Ein Eingangstor (Input Gate), ein Merk- und Vergesstor (Forget Gate) und ein Ausgangstor (Output Gate). LSTM ermöglicht auf diese Weise im Gegensatz zu herkömmlichen rekurrenten neuronalen Netzen eine Art Erinnerung an frühere Erfahrungen: Ein Kurzzeitgedächtnis, das lange anhält (weil das prinzipielle Verhalten des Netzes in den Gewichten kodiert ist).“

Typen von Neuronalen Netzwerken

Deconvolutional Neural Network (DNN)



Auto Encoder (AE) – Anwendungen: Datenkompression, Dimensions-reduktion, Rauschunterdrückung, Feature-Erkennung …



https://medium.com/analytics-vidhya/

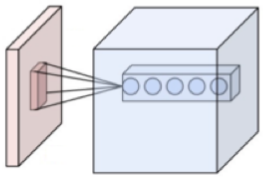
Bilder stammen von dieser Seite: https://medium.com/analytics-vidhya/11-essential-neural-network-architectures-visualized-explained-7fc7da3486d8

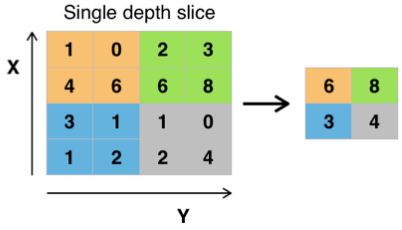
Convolutional Deep Learning Netzwerke

Convolution layers: integrieren über ein Feld in einer Schicht (z.B. 5 x 5 Knoten) und propagieren das Ergebnis dann an einen Knoten in der nächste Schicht.

Pooling layers: fassen die Ergebnisse von typischerweise 2 x 2 Knoten zusammen. „

Max pooling“ gibt den größten Wert weiter, „average pooling“ den Mittelwert.





www.wikipedia.org

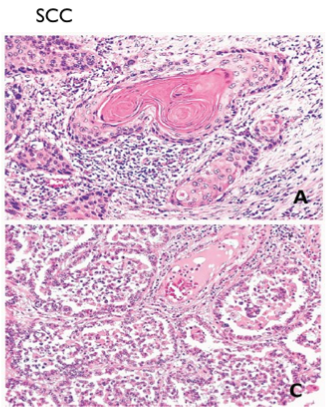
Fallstudie: Klassifizierung von Lungenkrebs anhand von histologischen Aufnahmen

nature medicine ARTICLES

Classification and mutation prediction from non-small cell lung cancer histopathology images using deep learning

Nicolas Coudray[1,2], Paolo Santiago Ocampo[2,3], Theodore Sakellaropoulos[4], Navneet Narula[2], Matija Snuderl[2], David Fenyö[5,6], Andre L. Moreira[2,7], Narges Razavian[1,8,9] and Aristotelis Tsirigos[1,2,9]

- 2 häufigste Formen von Lungentumoren:
- LUSC – lung squamous cell carcinoma (SCC):
- SCCs treten in Plattenepithelzellen auf.
- LUAD – lung adenocarcinoma - Adenocarcinome entstehen in Drüsen an verschiedenen Stellen im Körper.
- Beides sind **nicht-kleinzellige Lungentumore**



Coudray et al. Nature Medicine 24, 1559–1567 (2018)
http://www2.keelpno.gr/blog/?p=1391

Link to paper: https://www.nature.com/articles/s41591-018-0177-5
Deep Learning methods are nowadays known to be highly successful in automated classification of biomedical images.
This is only one example from an amazing mass of similar publications.
Here, the authors tried to distinguish two types of non-small cell lung cancers, LUSC (top figure) and LUAD (bottom figure).
Since Sept. 2018, this paper has been cited 330 times.

Behandlung von LUAC / LUSC

- Stadium I – Operation oder Bestrahlung
- Stadium II – Operation und Chemotherapie oder Bestrahlung
- Stadium III – sequenzielle oder gleichzeitige Chemotherapie und Bestrahlung
- **Stadium IV – Patienten-Genom wird wichtig**

- Zytotoxische Kombinationschemotherapie
- Kombinationschemotherapie mit monoklonalen Antikörpern
- Erhaltungstherapie nach primärer Chemotherapie
- EGFR Tyrosinkinase-Inhibitoren
- ALK Inhibitoren (für Patienten mit ALK-Translokationen)
- ROS1 Inhibitoren (für Patienten mit ROS1-Umlagerung)
- BRAFV600E und MEK Inhibitoren (für Patienten mit BRAFV600E Mutationen)
- Immun Checkpoint-Inhibitoren mit bzw. ohne Chemotherapie

https://www.cancer.gov/types/lung/hp/non-small-cell-lung-treatment-pdq#section/_48406

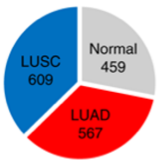
These are the treatment options at various stages of LUAC / LUSC, see table 7 at the given website of the National Cancer Institute of the US.
Further down, the stages become more serious, and the treatment more aggressive.
It is very important to identify in detail what type of tumor a patient has.
To select the best treatment in stage IV, it is also important to find out if the patient (or even the tumor tissue) carries particularly genomic mutations.

Klassifizierung von Tumorgewebe

Q: Können Deep Learning-Methoden LUAD / LUSC / normale (gesunde) Proben mit ähnlicher Genauigkeit wie ein medizinischer Experte (Pathologe) klassifizieren?

Anzahl an Proben aus

TCGA (The Cancer Genome Atlas)

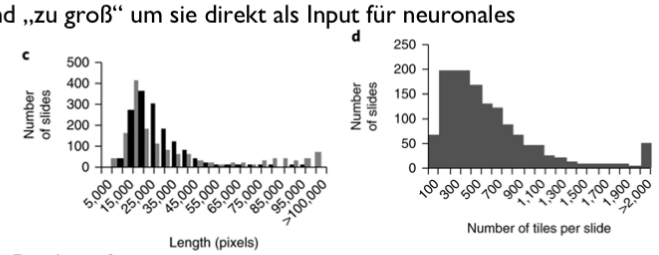


Coudray et al. Nature Medicine 24, 1559–1567 (2018)

The authors used tissue images (bottom figure) provided on the TCGA website.
As shown in the top figure, TCGA provides roughly equal portions of LUSC, LUAD, and normal (healthy).
But these numbers seem not enough for application of Deep Learning methods which require massive amounts of training data (100000s) in order to train a successful classifier.

Klassifizierung von Tumorgewebe

- Die einzelnen Aufnahmen sind „zu groß“ um sie direkt als Input für neuronales Netzwerk zu verwenden



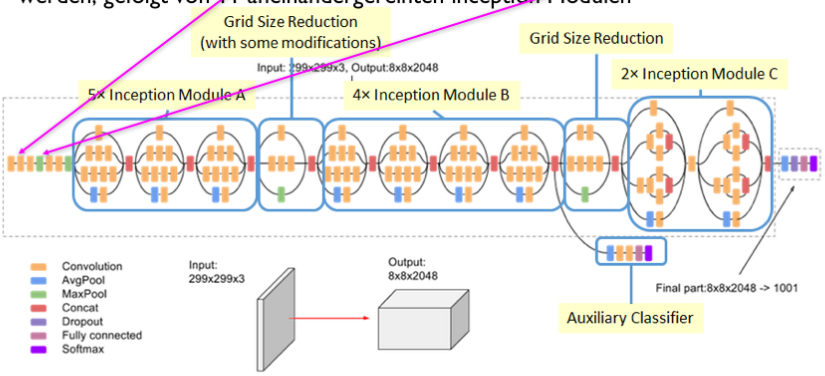
- Idee: spalte jedes Bild in kleine Quadraten mit 512 × 512 Pixeln auf.
- Dadurch vergrößert sich die Menge an Trainingsdaten stark.
- Spalte Daten in 70% fürs Training, 15% für Validierung, und 15% für Testen auf.
- -> **etwa 1 Million Bildblöcke**

Coudray et al. Nature Medicine 24, 1559–1567 (2018)

Here, there only 400 – 600 images each. The solution found by the authors was very simple.
They argued that the available images are „too large“ = have too high resolution to be used directly.
So one option would be to use only small parts of each image. But then, the amount of data is too small.
Also one would throw away a lot of potentially useful data.
Therefore, the idea was to split each slide into many small images and assume that the contained information is not largely redundant.
This increased the amount of data to more than 1 million smaller images (tiles).

Deep learning Modell

- Die Autoren verwendeten eine convolutional neural network-Architektur namens inception v3 architecture36, die von Google entwickelt wurde.
- Zu Beginn gibt es 5 convolution-Knoten, die mit 2 max pooling Operationen verknüpft werden, gefolgt von 11 aneinandergereihten inception Modulen



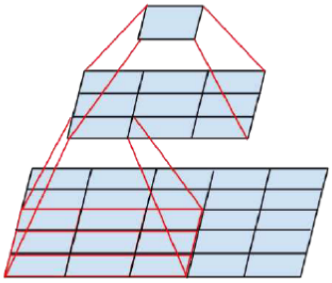
- „convolution“ = engl. für die Berechnung eines Faltungsintegrals = räumliche Summierung

medium.com

This is a brief introduction of the type of Deep Learning neural network architecture used in this study.
„Convolution“ nodes integrate density information from a region into a central pixel. See
https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c
for more information about this particular architecture (inception v3 architecture36).

inception v3 architecture36

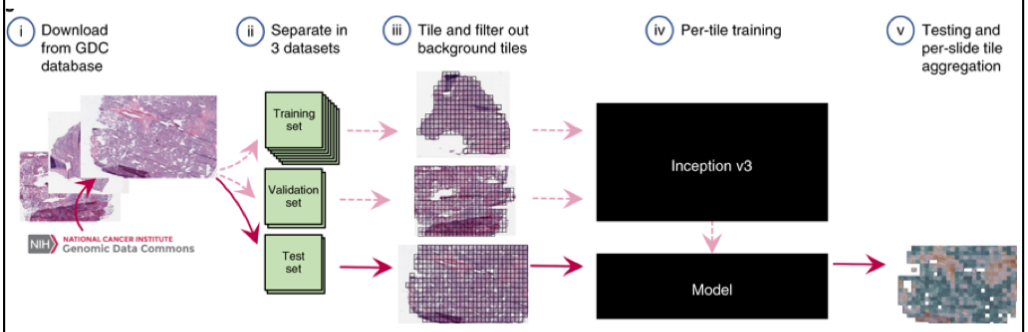
- Die hintereinander angeordneten convolution-Knoten dienen dazu, die Anzahl an Parametern zu reduzieren, die trainiert werden müssen.
- Im Beispiel unten ersetzen zwei 3×3 convolutions eine 5×5 convolution
- Statt einer Schicht mit 5×5 = 25 Parametern verwendet man daher zwei Schichten mit je 3×3 Parametern, also insgesamt nur 18.



medium.com

This is a brief introduction of the type of Deep Learning neural network architecture used in this study.
„Convolution“ nodes integrate density information from a region into a central pixel.
See
https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c
for more information about this particular architecture (inception v3 architecture36).

Workflow



Die Klassifizierung von gesundem versus Tumorgewebe klappte sehr gut (~0.99 AUC "area under the curve").
Unterscheidung von LUSC und LUAD ebenfalls mit 0.97 AUC möglich.

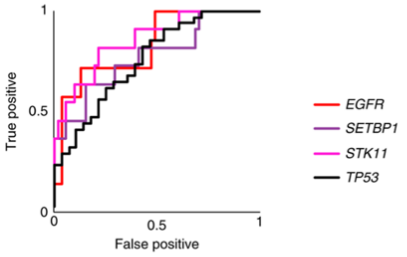
Das ist dieselbe Genauigkeit, die man erhält, wenn man 3 ausgebildete Pathologen bittet, dieselben Bilder zu klassifizieren.

Coudray et al. Nature Medicine 24, 1559–1567 (2018)

It has been reported in recent years that deep learning methods can be extremely successful in image recognition.
Many companies (also pharma companies) have now opened deep learning groups who work on such tasks.

Klassifizierung von genetischen Varianten

- Können CNNs das Auftreten bestimmter Genmutationen aus den Bildern erkennen?



- Einigermaßen. Die Genauigkeit (AUC = das Flächenintegral unter der Kurve) reicht von 0.64 (LRP1B) bis 0.84 (STK11).
- Falls es mehr Trainingsdaten gäbe, sollten sich die Ergebnisse noch verbessern.
- Tool kann Pathologen bereits heute bei Routine-Tätigkeiten unterstützen.

Coudray et al. Nature Medicine 24, 1559–1567 (2018)

Before this study, it was unclear whether gene mutations would affect the pattern of tumor cells on a lung cancer whole-slide image.
Interestingly, training the network using the presence or absence of mutated genes as a label revealed that there are certain genes whose mutational status can be predicted from image data alone: *EGFR*, *STK11*, *FAT1*, *SETBP1*, *KRAS*, and *TP53*. The ability to quickly and inexpensively predict both the type of cancer and the gene mutations from histopathology images could be beneficial to the treatment of patients with cancer given the importance and impact of these mutations

Klausur-relevanter Vorlesungsstoff

Vorlesung	Folien
1	16-25, 29
2	3-45
3	6-21, 32-45
4	21, 23
5	11-14, 19-34, 38-39, 41
6	1-35, 41
7	5-6, 8-12, 16-23
8	9-13, 21-29, 33-42
9	7-8, 14-18, 24-33
10	11-19
11	8-9, 13-15, 25
12	1-24
13	1-10

Relevant sind immer die Foliennummern unten rechts.