***Neural Networks***

**Week 1 recap:**

* Wat is Amazon Mechanical Turk, om je data te laten labelen?

*Amazon Mechanical Turk is een crowdsourcing marketplace waar bedrijven hun taken kunnen outsourcen. In de context van data te labelen betekent dit dat een bedrijf hier mensen betaald om hun data te labelen (bv dat de worker moet zeggen of hij een foto van een hond of een kat ziet).*

* Wat is k-fold cross-validation, en hoe moet je dit zien in de context van deep learning?

K-fold cross-validation is een resampling procedure die gebruikt wordt om een model te evalueren op een gelimiteerde data sample. De parameter K verwijst naar het aantal groepen waarin een data sample gesplit zou moeten worden.

Cross-validation wordt vooral gebruikt om te schatten hoe goed een model is op data die het model nog nooit gezien heeft.

* Bespreek de “goodness” van een model tijdens training, en tijdens testfase. Voor zowel classification als regression.

Tijdens training: wat voor type probleem? Regressie of classificatie?

Classificatie:

wiskundige berekening op logaritmen (standaard voorgekozen)

Regressie:

mean-squared error (hoever is men data punt verwijderd van men voorspelling)

Tijdens test: zware berekening (éénmalig), bij classificatie (confusion matrix), bij regressie (R-squared)

* Wat is een epoch, en een batch (vooral: wanneer doe je updating van je learnable parameters)?

Epoch:

Één iteratie over het deel meegegeven dataset.

Batch:

Een deel van de dataset dat door het neurale netwerk gestuurd wordt.

Het updaten van de learnable parameters doen we na elke batch update in het netwerk.

* Bespreek bias - variance, en wat dat te maken heeft met # of features (bias is meestal redelijk goed te begrijpen, maar focus op variance!).

Bias maakt assumptions van de data voor variance errors tegen te gaan (bias-variance tradeoff).

Variance neemt alle features (zelfs de onbelangrijke).

Bians generaliseerd features (te weinig features).

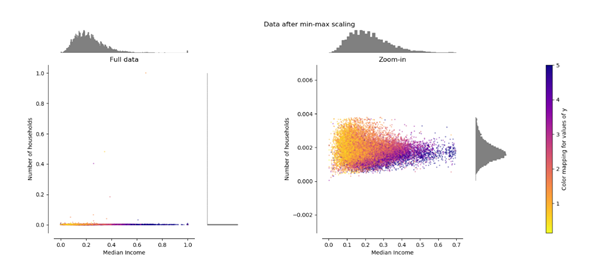
Balans tussen deze 2 proberen te zoeken (overfit ↔ underfit).

* Wat is data imputation?

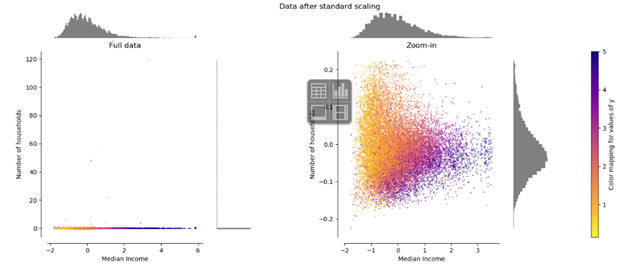
Missing data vervangen en opvullen met assumptions rekening houdend met andere features

* Bespreek normalization versus standardization, en wat is het effect hiervan op data met outliers?

Normalization:



Standardization:



**Week 2 neural network basics:**

* Een Neural Net is een hiërarchisch model. Leg uit.
* Wat is een affine transformation (in de context van een perceptron)?
* Om een multilayer perceptron, met minstens 1 hidden layer, te laten werken als universal approximator, heb je non-linear activation functions nodig. Verklaar.
* Leg de ReLu activation function uit, en waarom je soms een leaky ReLu, of ELU nodig hebt.
* Leg de loss function uit in de context van een regression problem, en in de context van een classification problem.
* Leg uit waarom de gradiënt van de loss function belangrijk is.
* Bij backpropagation gaan we de gradiënt van de loss function laten terugwerken naar het begin, via de chain rule. Daarom hebben we ook de gradients van de activation functions nodig(niet echt die van affine linear transforms). Leg in deze context het probleem van vanishing gradients uit (zie ook vraag over ReLu activation function).

**Week 3:**

* Leg uit wat stochastic gradient descent is (tov gewone ‘batch’ gradient descent), en wat het begrip ‘mini-batch’ in deze context betekent.
* Leg uit wat een goede learning rate is, en wat simulated annealing betekent.
* Leg de One Cycle learning rate uit.
* Leg de verschillende soorten regularisaties uit: L1 of L2 weight decay, early stopping, dropout, ensemble technieken, data augmentation en batch normalization.
* Leg verschillende soorten optimizations uit: batch normalization, momentum technieken (RMSProp, Adam), initializers (Xavier).

**Week 4:**

* Leg uit welke pre-processing technieken meestal nodig zijn om met images te kunnen werken in een CNN.

Uniform aspect ratio:

(Zo goed als alle libraries gebruiken vierkante images (gemakkelijk om te roteren, transpose,…))

- crop images to be square

- center of image most important

Uniform image size:

(Alle images in een batch (zelfs over alle batches heen, niet alleen in een ‘mini’-batch, maar in de ganse batch) moeten dezelfde afmetingen hebben

Truck: meestal gaat men de images zelfs eerst verkleinen, model snel trainen, en dan transfer learning techniek gebruiken om te starten met trained netwerk, images terug te vergroten, en dan fine tuning te doen)

- fit image size to CNN feature maps (pixels to convolutional layer)

- up-scaling and down-scaling

Mean and perturbed images:

(Dit is een pure ‘insight’ techniek, om feeling te krijgen voor de images. De meeste conv.nets doen geen specifieke preprocessing op mean, of normalisation niveau. Toch niet bij de inputs)

- average pixel across training dataset

Normalized image inputs:

(Scaling doet men wel regelmatig (0-1 range) (hoewel dit niet niet altijd nodig is), maar normalisation doet men meestal in een ‘batch normalize’ layer, na de convolution layer, en voor de ReLU activation. De convolution neurons hebben immers geen echte activation function, dus ook geen nood aan normalizing…)

- scaling en uitschieters wegdoen

Dimensionality reduction:

- reduce the 3 rgb channels to grayscale for just one channel (reduces dimensionality of all image tensors and the size of the problem so training completes faster)

Data augmentation:

- foto’s inzoomen, draaien, spiegelen voor meer data en overfitten te verminderen en het zo meer robuust te maken

* Waarom gebruiken we geen DNN om computer vision te doen op images?

DNN is te complex en fully connected met veel layers en gebruikt veel meer neuronen. CNN is meer shallow en traint ook sneller.

* Leg uit wat een convolution is (in de context van CNNs), en gebruik hiervoor de termen ‘kernel’, ‘locale receptive field’, ‘stride’ en ‘feature map’.

Kernel: een matrix met gewichten die toegepast wordt op de local receptive field

Local receptive field: deel van de image waar momenteel naar gekeken wordt

Stride: afstand dat uw local receptive field verschuift

Feature map: het resultaat van al de kernel berekeningen

* Leg uit wat een 2D convolution bewerking is met een 3D kernel op een RGB image.

Kleur images, met meerdere channels, vragen wel voor een 3D (of meerdere dimensies) kernel, maar het is een 2D convolution: we bewegen de kernel in een 2D vlak over de image (wel met meerdere channels tegelijkertijd). Output is ook een 2D matrix.

* Leg uit waarom we padding gebruiken bij convolution bewerkingen op images, en geef enkele voorbeelden van padding.

Bij CNN kan het zijn dat de randen van de image niet zo belangrijk worden dus kan er een lege border rond worden gezet zodat deze ‘genegeerd’ wordt ipv de image zelf.

Er kan ook padding gebruikt worden als met de strides het local receptive field niet past.

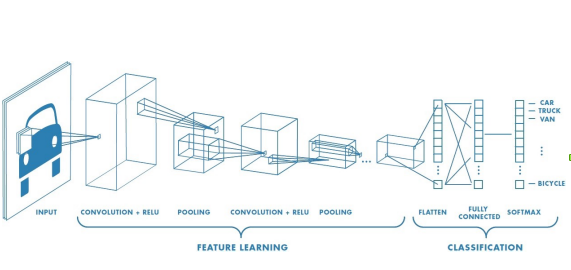
* Waarom is het belangrijk dat je de output\_size van een convolution bewerking moet weten?

Principe is niet zo belangrijk in praktijk, omdat de meeste frameworks zoals keras en fastai dit zelf opvangen maar bij tensorflow of fastai is dit wel belangrijk omdat je moet kunnen doorgeven aan je volgende layer hoe groot deze moet zijn, dit is vooral belangrijk als je van je CNN naar je fully connected gaat gaan omdat je moet weten hoe groot het gaat worden met je flattening layer.

* Leg uit waarvoor we pooling technieken gebruiken, en geef enkele voorbeelden.

Reduce the parameters of the image, when your image is too big. (max, average, sum pooling). Reduce memory usage, overfitten tegengaan, make NN recognize features independent of location.

* Leg een typische CNN architectuur uit voor image classification (convolutions, poolings, activation functions, normalisation layers, flattening layer, … ).

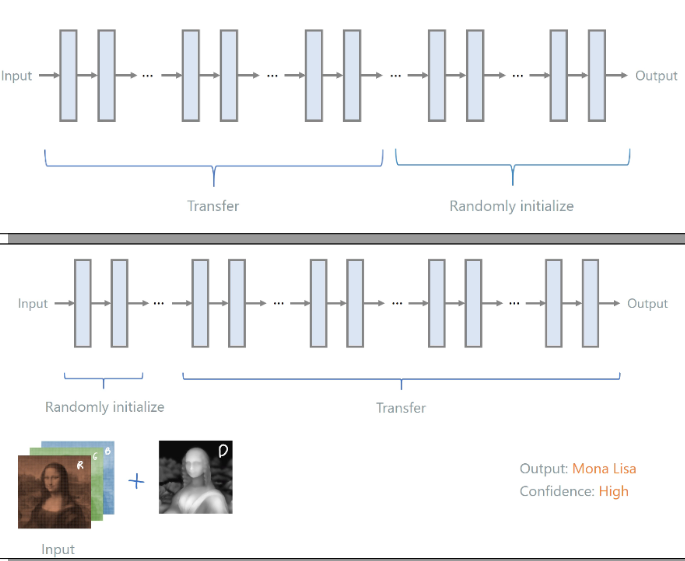


**Week 5:**

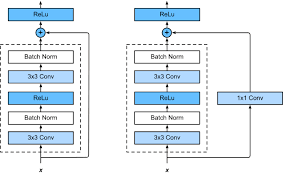
* Leg uit waarom deep learning data hungry is, maar dat transfer learning daar een mogelijke oplossing voor is. Geef hierbij ook aan welke limitaties transfer learning heeft.
  + Deep learning mist momenteel een mechanisme voor het leren van abstracties door expliciete, verbale definitie, en werkt het beste wanneer er duizenden, miljoenen of zelfs miljarden trainingsvoorbeelden zijn. Transfer learning gebruikt zijn kennis van een geleerde taak om een soortgelijke taak op dezelfde manier op dezelfde manier op te lossen (minder data en compute). Wanneer er een te groot verschil is tussen de architecturen van de 2 models, kunnen we moeilijk de knowledge, de weights, zomaar gaan transfereren (limitation).



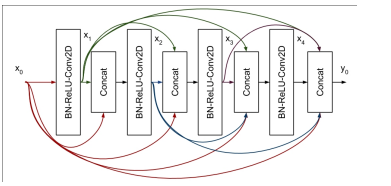
* Leg verschillende mogelijke strategieën (en hun context) uit om transfer learning toe te passen.



* De bovenste keuze hier is de meest gebruikte manier, maar de onderste manier wordt gebruik als je gelijkaardige input hebt die toch een klein beetje verschilt.
* De bovenste is de meest gebruikte om de main features te hergebruiken, terwijl de onderste kan werken als je gelijkaardige input hebt die een klein beetje anders is. De eerste lagen moeten dan eerst leren die input wat te projecteren in andere dimensies en kan daarna min of meer hetzelfde doen zoals bij het eerste netwerk.
* Wat is ImageNet, en waarom is dit zo belangrijk geweest voor deep learning?
  + ImageNet is een grote database die correct gelabeld is met 1000 categorieën en 1,2 miljoen training images en een test set van 100 duizend images. Dit is belangrijk geweest omdat dit enorm veel werk was en mensen hebben hier gebruik van gemaakt en is daardoor een heel grote boost door gekomen op het gebied van image recognition.
* Leg uit hoe de 1x1 convolution (identity) een optimalisatie is in de compute van CNNs.
  + Door een 1x1 te gebruiken zorg je ervoor dat er minder dimensies zijn en dus ook minder compute kracht nodig is om dit te berekenen. Dit lijkt niet logisch maar als je de wiskunde bekijkt zie je dat je bij een 1x1 veel minder berekeningen moet doen dan bij een 5x5 bijvoorbeeld.
* Leg de verschillende innovatieve verbeteringen uit van AlexNet, VGG, Inception (GoogLeNet), ResNet, DenseNet.
  + AlexNet: stacking of conv layers (no pooling in between), local response normalization (sterk geactiveerde neurons gaan tegen dichtbijzijnde neurons zeggen dat ze moeten zwijgen zodat neurons vechten om te specialiseren in verschillende features). Om overfitting te voorkomen is er een heel hoge drop out rate van 50% zodat niet 1 neuron instaat voor een bepaalde feature, maar een paar neurons hiervan. Randomly shifting training images by offsets helpt ook tegen de overfitting.
  + VGG: in plaats van een grotere filter te nemen neemt VGG meerdere kleinere filters na elkaar zonder een pooling layer om zo exponentieel minder parameters moet trainen en dus ook minder dimensies gaat hebben. Daarom heb je dus minder compute kracht nodig.
  + Inception (GoogLeNet): bij kleinere kernels is de detectie van info meer lokaal en bij grotere kernels meer globaal, daarom neemt het een mix van de twee. Diepere networks hebben meer de neiging om te gaan overfitten dus gaan we hier breder en dieper, het stacken van conv layers is wel heel compute intensive.
  + ResNet: extreem diep netwerk (152 layers), skipped connecties dit forceert het model om de rest te leren, wat de andere lagen niet doen. We kunnen dieper gaan met dit network omdat connecties skippen het makkelijker maakt for the gradient to backprop.



* DenseNet: bestaat uit dense blokken die fully connected zijn, hier ‘skippen’ connecties naar elke sublaag van het blok: BatchNorm layer, ReLu activation, Conv2D. Een voordeel hiervan is dat je minder parameters hebt om te trainen en je kan veel layers droppen. Elke layer heeft toegang tot de gradients van de loss function en de originele input image.



**Week 6:**

* Leg uit hoe, en waarom, we Active Learning gebruiken om data sneller te labelen (bv via een MachineLearning-human-in-the-loop-approach).
* Leg uit waarom het soms aangewezen is om, eerder dan voor een grote hoeveelheid labelled data, voor meer quality (tov quantity) van labelled datat te gaan.
* Leg de verschillende stappen van het kwaliteitsvol labellen van data, via een Active Learning approach uit. Leg ook uit waarom de querying stap hierin zo cruciaal is.
* Leg kort uit wat model distillation betekent (of knowledge distillation) in de context van Edge computing.

**Week 7:**

**Week 8:**

**Week 9:**

**Week 10:**

**Week 11:**

**Week 12:**