Survey on DL@UAV vision

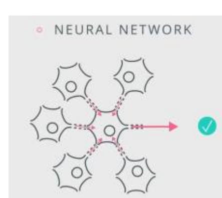
CONTENT

I — What is DL?
II — Application
III — Road Map
IV — Start a Project

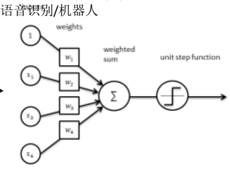
01 What is DL?

1_ What is DL

From **Neurons** to **Perceptron**

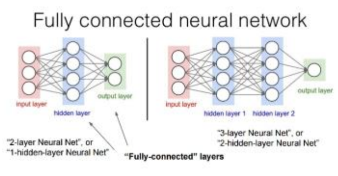


1957 年美国康乃尔大学的心理学家罗森勃拉特(Rosenblann)提出了感知器或感知机(Percepuon)的概念,并试图用它来模拟动物和人脑的感知与学习能力,而且用一个电子线路设计了著名的感知器神经网络模型,称为仿脑机,能够识别英文字母印刷体
深度学习应用的四个大的领域
CV/NLP/语音识别/机器人



1_ What is DL

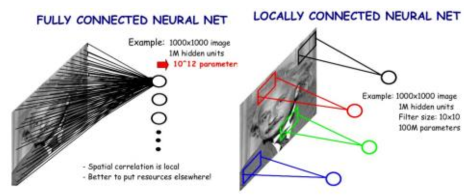
FC – Fully Connection



- Hidden Layer
- Nonlinear Activation

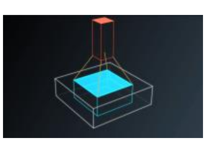
1_ What is DL

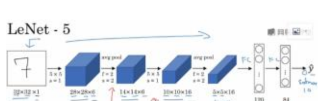
CNN – Convolutional Neural Network



1_ What is DL

CNN – Convolutional Neural Network



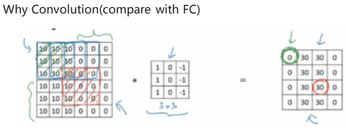


- Convolutional Kernel dimension

width * **height** * **input channel** * **filter number**

1_ What is DL

Why Convolution(compare with FC)



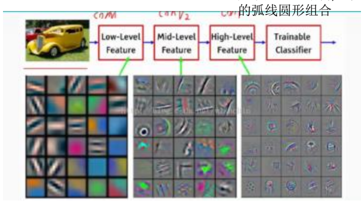
Vertical edge detector

- **Parameter sharing:**
  A feature detector(such as a vertical edge detector) that is useful in one prat of the image is probably useful in another part of the image
- **Sparse connection**
  In each layer, each output value depends only on a small number of inputs
- **Less Parameters, Low Overfitting Risk**

1_ What is DL

CNN as Feature Extractor

CNN就是一个特征提取的工具,通过特征可视化的手段,我们可以还原出对某一层激活函数激活值最大的一些图像,可以看到从低到高,CNN在逐渐抽取越来越抽象的特征,开始时边缘/颜色块等,然后就是纹理,然后就是一些结构,比如下面的蜂巢,上面的像眼睛一样的弧线圆形组合



edge/color texture structure

CNN Visualization

02 Application

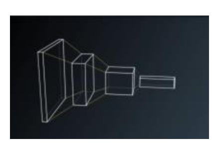
2_ Application

- Object Recognition
- Object Detection
- Visual Object Tracking(VOT)
- Visual Slam

2_ Application

Object Recognition

CNN用于OR的一般架构就是在空间上越来越小,深度上越来越深,就是把图像的信息从空间上抽象到深度上的过程,所以我们可以看到,CNN逐层空间大小在变小,深度在增加



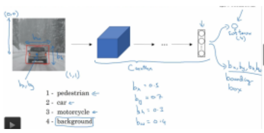
Classification

Dogs vs. Cats in Kaggle

2_ Application

当用于OD时,其实和OR是相同的思路,这个时候抽象出来的信息不只是对物体类型的识别,还有位置的描述,通常会用x,y,h,w来描述一个bb,现在主流的做法是通过回归的方法让网络学习bb的伸缩和偏移,从而确定物体的位置

Object Detection



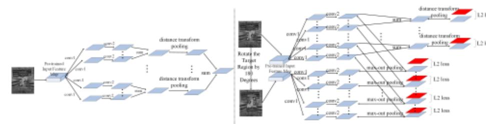
Classification + multi-object localization

VEDAI UAV Object Detection Dataset

2_ Application

- 本文旨在将CNN和KRR结合起来，其中CNN用于用于关注目标的局部信息，KRR用于关注目标的整体信息

VOT



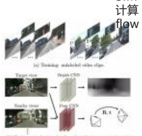
LSART – VOT 2017 Championship

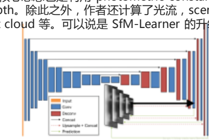
- Exploit both the KRR and CNN as two complementary regressions for visual tracking
- CNN focuses on the small localized region
- KRR focuses on the holistic target

2_ Application

- 它用两个网络分别/独自无监督地估计单帧的深度，和视频序列中的camera的pose变化
- 光度一致性, 就是对于同一个物体的点,在不同两帧图像上投影点图像灰度应该是一样的
- SfM-Net论文的核心思想也是利用 photometric constancy 来计算 pose，depth。除此之外，作者还计算了光流，scene flow，3D point cloud 等。可以说是 SfM-Learner 的升级版

Visual Slam





SfM-Learner – CVPR 2017 from Google

- Use two isolated network(Depth CNN & Pose CNN) to predict depth & camera pose
- Base on photometric consistency principle
- See more on SfM-Net(also compute optic flow/scene flow/3D point cloud)

03

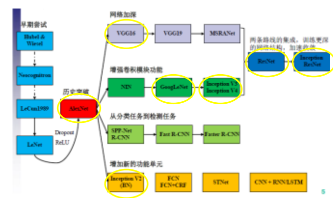
Road Map

3_ Road Map

- Introduction of CNN Development
- Small CNN for mobile application
- Network Compression

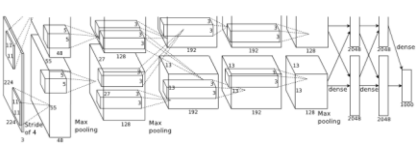
3_ Road Map

Development of CNN



3_ Road Map

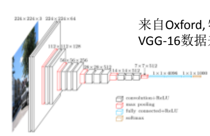
AlexNet – 2012 ImageNet Championship (Top-5 Error 16.4%)



- First time, CNN won the ImageNet Challenge
- First time, Group Convolution Concept proposed
- First time, Use Relu as activation function(used in the following networks till now)

3_ Road Map

VGGNet – 2014 ImageNet (Top-5 Error 9.9%)

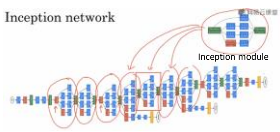


来自Oxford,特点就是深,但是参数特别多
VGG-16数据来自Xception论文

- Deep, 16/19 layers
- Conv+Pooling+FC
- Too much Parameters ~138,000,000

3_ Road Map

Inception(GoogLeNet) – 2014 ImageNet Championship (Top-5 Error 9.2%)



- Much less Parameters ~5,000,000(1/12 of AlexNet, 1/27 of VGGNet)
- 1x1 convolution(Network in Network)
- Group Convolution(Inception Module)
- Replace FC with Average Pooling

3_ Road Map

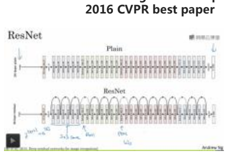
数据标准
Single-crop, Single-model experimental results(一个样本截取一张图的, 单一模型不用集成学习的)
On ILSVRC 2012 dataset validation set
目前在ImageNet数据集上人眼能达到的错误率大概在5.1%，这还是经过了大量训练的专家能达到的成绩，一般人要区分1000种类型的图片是比较困难的

Inception Family

- Inception-V2 (Top-5 Error 7.8%)
  - Batch Normalization/two 3x3 replace 5x5
- Inception-V3 (Top-5 Error 5.6%)
  - Conv Factorization 3x3 -> 1x3+3x1
- Xception (Top-5 Error 5.5%)
  - Depth-wise separable convolution(separate depth conv & spatial conv)

3_ Road Map

ResNet– 2015 ImageNet Championship (Top-5 Error 4.49%)
2016 CVPR best paper



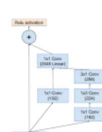
- Shortcut connection/solve Gradient vanishing
- Break the depth limitation

那么这里可以看到，本来从上一层传过来的梯度为1，经过这个block之后，得到的梯度已经变成了0.0001和0.01，**也就是说，梯度流过一个blcok之后，就已经下降了几个量级，传到前一层的梯度将会变得很小！**
这就是梯度弥散。假如模型的层数越深，这种梯度弥散的情况就更加严重，导致浅层部分的网络权重参数得不到很好的训练，这就是为什么在Resnet出现之前，CNN网络都不超过二十几层的原因
数据来自ResNet论文(Single-Model)
从此能够限制层数的只有内存大小了 – KM He
KM He 2017年还有两篇ICCV best paper, 与object detection有关, 下个专题再讨论

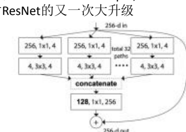
3_ Road Map

天下大势, 分久必合, 合久必分
Cardinality和Group Conv的区别在于Cardinality是把一个Conv操作分成了拓扑结构完全相同的几个path, 这样方便加速
2017年最后一届ImageNet分类的冠军, Momenta的SENet也要研究一下
以及2017 CVPR best Paper, DenseNet, 可以看做是对ResNet的又一次大升级

Inception-ResNet & ResNeXt(Top-5 Error 4.9% & 4.4%)



Inception Module + shortcut connection



Residual block + Group conv(**cardinality**)

3_ Road Map

Cost Comparison

CNN模型在不断逼近计算机视觉任务的精度极限的同时，其深度和尺寸也在成倍增长
巨大的模型无法在嵌入式平台上落地, 就算想通过网络传输, 较高的带宽也让很多用户望而生畏
另一方面，大尺寸的模型也对设备功耗和运行速度带来了巨大的挑战
所以, 模型的小型化和加速成了亟待解决的问题

Model	Model Size(MB)	Mult-Adds	Million Para
AlexNet	200	720	60
VGG16	500	15300	138
GooLeNet	~50	1550	6.8
Inception-V3	90-100	5000	23.2

3_ Road Map

Introduction of CNN Development

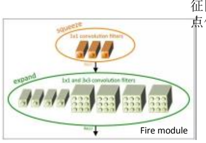
Small CNN for mobile application

Network Compression

3_ Road Map

SqueezeNet(ICLR-2017, Stanford)

1. 用1x1替换部分的3x3, 减少参数
2. 减少进入3x3 filter的通道数, 也就是先用1x1去压缩通道, 就是squeeze
3. 少用pooling或者是conv stride, 保留比较大的特征图, 信息保留的多, 最后的准确性更高, 但这点使模型解释的成本更高, 运算代价更大了



Fire module

- Replace part of 3x3 filters with 1x1 filters(small network)
- Decrease the number of input channels to 3x3 filters(small network)
- Downsample late in the network, conv layers has large activation map(high accuracy)

3_ Road Map

SqueezeNet(ICLR-2017, Stanford)

Table 2: Comparing SqueezeNet to model compression approaches. By *model size*, we mean the number of bytes required to store all of the parameters in the trained model.

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD (Denton et al., 2014)	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning (Han et al., 2015b)	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression (Han et al., 2015a)	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%

- Small Network, but high compute cost(large activation map)
- Nothing Fresh in architecture

3_ Road Map

1. 将空间conv和深度conv完全分离开, 提速
2. 相比SqueezeNet, 计算量方面有明显提升(SqueezeNet似乎并没有考虑这个)
3. Accuracy上, MobileNet其实和AlexNet差不了太多, 稍有提升

MobileNet(CVPR-2017, Google)

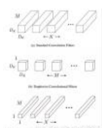


Table 9. Smaller MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

- 1 conv filter for 1 channel(totally separate spatial conv & depth conv)
- Low compute cost(compare with SqueezeNet)
- Almost equivalent accuracy level(SqueezeNet, AlexNet)

3_ Road Map

ShuffleNet (CVPR-2017, Face++)

ShuffleNet的几个创新点
1. 提出了一个类似于ResNet的BottleNeck单元
借鉴ResNet的旁路分支思想，ShuffleNet也引入了类似的网络单元。不同的是，在stride=2的单元中，用concat操作代替了add操作，用average pooling代替了1x1stride=2的卷积操作，有效地减少了计算量和参数。
2. 提出将1x1卷积采用group操作会得到更好的分类性能
在MobileNet中提过，1x1卷积的操作占据了约95%的计算量，所以作者将1x1也更改为group卷积，使得相比MobileNet的计算量大大减少。
3. 提出了核心的shuffle操作将不同group中的通道进行打散，从而保证不同输入通道之间的信息传递。解决多个group conv叠加出现的边界效应问题
Channels Shuffle 操作会导致内存不连续这个影响有待评估

- Channel shuffle for less cross-talk between channels(group conv issue)
- 1x1 group conv to reduce compute cost
- Use Residual block to raise accuracy

3_ Road Map

MFLOPs(Million Floating-point Operations per Second) 衡量计算复杂度的指标

ShuffleNet (CVPR-2017, Face++)

Model	Complexity (MFLOPs)	Cls err. (%)	Δ err. (%)
1.0 MobileNet-224	569	29.4	-
ShuffleNet 2× ($g = 3$)	524	**26.3**	3.1
ShuffleNet 2× (with *SE*[13], $g = 3$)	527	**24.7**	4.7
0.75 MobileNet-224	325	31.6	-
ShuffleNet 1.5× ($g = 3$)	292	**28.5**	3.1
0.5 MobileNet-224	149	36.3	-
ShuffleNet 1× ($g = 8$)	140	**32.4**	3.9
0.25 MobileNet-224	41	49.4	-
ShuffleNet 0.5× ($g = 4$)	38	**41.6**	7.8
ShuffleNet 0.5× (shallow, $g = 3$)	40	42.8	6.6

Table 5. ShuffleNet vs. MobileNet [12] on ImageNet Classification

- Lower Complexity than MobileNet
- Higher Accuracy than MobileNet

3_ Road Map

Small CNN Conclusion

- Separate convolution thoroughly (X-conv + Y-conv + Depth-conv)
- Raise utilization of channel information
- Residual block + Group Conv would be standard architecture

3_ Road Map

Introduction of CNN Development

Small CNN for mobile application

Network Compression

3_ Road Map

Deep Compression (ICLR-2016 best paper, Stanford)

1. 剪枝, 去掉一些不重要的连接(训练出来几乎为0权重的连接), Han Song在ICLR领奖的演讲这儿说的很有意思, 剪枝相当于一个人喝大了, 但又没有丧失意识, 可以做一些简单的任务, 所以对简单任务, 我们不需要那么多的冗余连接
2. 权值共享是用少的bit数来存之前都是浮点数的权重, 而且只存权重的shift和code book, 就相当于训练这边编密码, 解释的时候用code book来解码
3. Huffman按照符号出现的概率来进行变长编码, 可以进一步减轻存储参数的Memory压力, 过去图像压缩经常会用到

- Pruning/Sparse Connection
- Quantization/Weight sharing
- Huffman Encoding

3_ Road Map

Deep Compression (ICLR-2016 best paper, Stanford)

1. 几乎没有准确率的损失
2. 如果和小模型搭配使用, 将更有利于在移动端部署
3. 虽然模型被压缩了, 但模型的解释成本增加了, 这一点还要double confirmed一下

因为模型的稀疏连接, 需要特殊的库来处理, 所以这种压缩方法在模型解释的时候一定是增加了成本

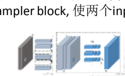
Network	Original Size	Compressed Size	Compression Ratio	Original Accuracy	Compressed Accuracy
Lenet-300-100	1070KB	27KB	40x	98.36%	98.42%
Lenet-5	1720KB	44KB	39x	99.20%	99.26%
AlexNet	240MB	6.9MB	35x	80.27%	80.30%
VGGNet	550MB	11.3MB	49x	88.68%	89.09%
GoogleNet	28MB	2.8MB	10x	88.90%	88.92%
SqueezeNet	4.8MB	0.47MB	10x	80.32%	80.35%

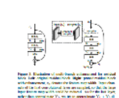
- Almost no accuracy lost
- Combining with small network would be more useful in embedded platform
- High model interpretation cost? Need to be verified

3_ Road Map

Channel Pruning (ICCV-2017, Face++)

1. 结构化剪枝, 不会造成稀疏连接, 所以解释的成本要更低
2. 把求解通道mask β 当做一个优化问题求解
3. 对于ResNet进行通道剪枝的时候, 增加一个sampler block, 使两个input的通道一样





- Structure Pruning, Lower model interpretation cost
- Compute mask β by solving a optimization problem
- Add *sampler* block for multi-branch networks(e.g. ResNet)

3_ Road Map

Channel Pruning (ICCV-2017, Face++)

Increase of top-5 error (1-view, baseline 89.9%)			
Solution	2×	4×	5×
Jaderberg *et al.* [22] ([52]'s impl.)	-	9.7	29.7
Asym. [52]	0.28	3.84	-
Filter pruning [31] (fine-tuned, our impl.)	0.8	8.6	14.6
Ours (without fine-tune)	2.7	7.9	22.0
Ours (fine-tuned)	0	1.0	1.7

Table 1. Accelerating the VGG-16 model [43] using a speedup ratio of 2×, 4×, or 5× (*smaller is better*).

1. 和Deep Compression不一样, channel pruning会造成精度损失, 但是结构化剪枝将来模型解释的成本低
2. 剪枝完fine-tune一下效果会好很多
3. 增加了很多hyper-para, 比如正则化系数λ, 还有压缩的比例

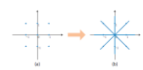
- Accuracy Lost
- Need fine-tune after pruning
- More hyper-para(regular coefficient λ, compress ratio) need be manually set, increase tuning difficult

3_ Road Map

Squeeze the Last Bit Out (AAAI-2018, Alibaba)

1. 基于低bit网络表示
2. ADMM算法来解决
3. 增加了很多hyper-para, 比如正则化系数λ, 还有压缩的比例

看了Squeeze the last bit之后, 把这里丰富一下
这里还应该有一个对于压缩方法的总结



$$
\begin{aligned}
W^{k+1} &:= \arg\min_{W} L_\rho(W, G^k, \lambda^k) \\
G^{k+1} &:= \arg\min_{G} L_\rho(W^{k+1}, G, \lambda^k) \\
\lambda^{k+1} &:= \lambda^k + W^{k+1} - G^{k+1}
\end{aligned} \qquad (10)
$$

- Low Bit Networks
- ADMM algorithm
- Even higher accuracy in VGG-16, Compression has regularization effect

3_ Road Map

Network Compression Conclusion

- Pruning
- Quantization
- Low-Bit Network(Binary Network, Ternary Network)
- Low rank decomposition
- Teacher-student Framework

04 Start a project

4_ Start a project

- Deep learning Toolkits
- Distribution Learning
- Deployment on Mobile
- Issues when Training

4_ Start a project

Deep Learning Toolkits Comparison

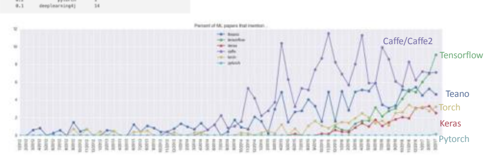
1. Caffe从UC Berkeley的实验室出来, 最开始的作者是贾扬清, 后来被Facebook收编, 因为Caffe灵活性不足, 贾扬清在Facebook内部又搞了Caffe2, 估计Caffe用户一两年内都要迁移到其他框架上去
2. PyTorch目前也由Facebook团队维护
3. Theano开创了深度学习框架符号编程的先河(相比于caffe的命令编程), 17年停止维护
4. Tensorflow由Google团队维护, 所以我们看到前面Google发的paper框架一定都是tensorflow的
5. MXNet之前是一帮大学的研究人员维护(Xgboost作者华盛顿大学陈天奇在内, 美国,加拿大,新加坡的多所高校), 17年被Amazon接盘

4_ Start a project

Deep Learning Toolkits Comparison

% of papers	framework	has been around for (months)
9.1	tensorflow	16
7.1	caffe	37
4.6	theano	54
3.3	torch	37
2.5	keras	19
1.7	matconvnet	26
1.2	lasagne	23
0.5	chainer	16
0.3	mxnet	17
0.3	cntk	13
0.2	pytorch	1
0.1	deeplearning4j	14

Tensorflow用户数量这两年稳定增长, 到2017.3已经处于不可撼动的地位
Caffe2/Pytorch都开源时间不长, 数量比较少
MXNet虽然性能很优, 但是一直没有火起来, Amazon加入后不知道是否会有改观



4_ Start a project

Deep Learning Toolkits Comparison

1. Tensorflow的优势在于社区庞大, 对移动端支持较好, 可以快速地开发一个部署在Android系统上的应用, 缺点是性能比较差, 据说google内部对于内存和GPU优化还是留了后门的, 而且调试困难
2. Facebook同时支持Caffe2和Pytorch两个框架, 意图很明显, Pytorch用于研究, 架构优雅简介, 方便快速的验证想法, Caffe2主攻工业应用, 性能已经可以和MXNet比肩了, 并且重视移动端的支持
3. MXNet最大的优势在于对内存的优化比较极致, 相同的内存可以训练更大的网络, 但这点估计很快要被Caffe2超越

Toolkit	Modeling Capability	Interfaces	Model Deployment	Architecture Complexity	Training Speed (VGG-style CNN on CIFAR-10)	Distribution Training
Tensorflow	CNN/RNN/LSTM/etc.	Python/C++/Go/Java…	iOS/Android/ARM	★★★★☆	173s	
Caffe2	CNN/RNN/LSTM/etc	C++/Python/Matlab	iOS/Android/ARM	★★★☆☆	149s	
MXNet	CNN/RNN/LSTM/etc.	Python/C++/R/Java/Julia…	iOS/Android/ARM	★★★★☆	149s	
Pytorch	CNN/RNN/LSTM/etc.	Python/Lua	iOS/Android	★★★★★	168s	

4_ Start a project

还有一些针对特定任务的框架,比如纯C写的darknet,
还有Facebook新出的针对目标检测的detectron

Deep Learning Toolkits Conclusion

- **TensorFlow** is a safe bet for most projects, Not perfect but has huge community, wide usage.
- **Pytorch** is best for research. However still new, there can be rough patches
- Use **TensorFlow** for one graph over many machines
- Consider **Caffe2** or **TensorFlow** for production deployment
- Consider **TensorFlow** or **Caffe2** for mobile

1. TensorFlow虽然不完美,但是利于部署,有稳定社群,同时上层API如Keras支持Tensorflow,便于开发
2. PyTorch很适合研究,但是它很新,因此用的时候有很多坑要自己填
3. 除了Tensorflow, Caffe2也可以考虑用于产品部署
4. 手机端可以考虑TensorFlow或Caffe2

总结:框架相当于刻刀,雕塑的好坏还是要取决于匠人,框架就是框架,关键是要出活儿

4_ Start a project

- Deep learning Toolkits
- Distribution Learning
- Deployment on Mobile
- Issues when Training

4_ Start a project

- Deep learning Toolkits
- Distribution Learning
- Deployment on Mobile
- Issues when Training

4_ Start a project

- Deep learning Toolkits
- Distribution Learning
- Deployment on Mobile
- Issues when Training

4_ Start a project

Issues when Training

- **Lack of Training Data ->** Data Augmentation/Transfer Learning
- **Low Accuracy ->** Ensemble learning/Feature stacking
- **Model convergence Issue ->** Gradient Descent Optimization/Activation function
- **Local Optima ->** Weight Initialization/Learning rate auto-tuning
- **Overfitting** -> Regularization

2018

Thanks