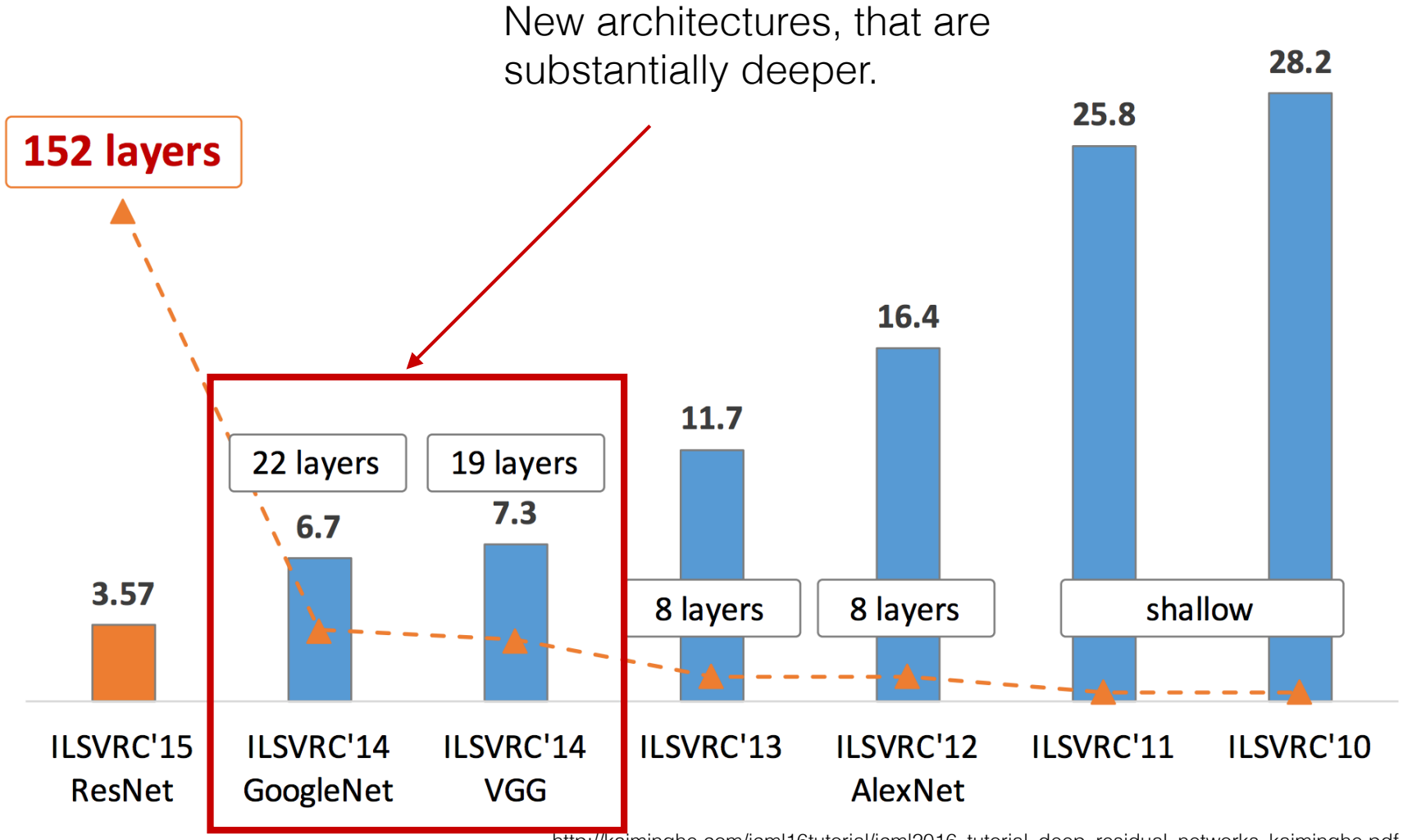
Lecture 14: CNNs + RNNs

Announcements:

- HW #5 is due today **Monday, March 4**, uploaded to Gradescope. Appreciate that you went from 35-40% accuracy with softmax on Homework 2 to 65+% accuracy with CNNs!
- Remaining schedule: Today: CNNs + RNNs, 3/6: RNNs + object detection, 3/11: object detection + adversarial examples, 3/13: adversarial + overview.
- The project and its accompanying data have been uploaded to Bruin Learn. It is due **March 15, 2024** (Friday of Week 10).
    - You will be allowed to use PyTorch, Keras, or other deep learning libraries for the project.
    - We will cover RNNs today, though we may not finish LSTMs. Our lectures on LSTM go over why it's a good idea and why it works; you can feel free to implement them using LSTM cells in PyTorch (https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html) or Keras (https://keras.io/api/layers/recurrent_layers/lstm/) and play with hyperparameters.
- We will release midterm grades tonight; regrades due within a week.

What about depth?



http://kaiminghe.com/icml16tutorial/icml2016_tutorial_deep_residual_networks_kaiminghe.pdf

VGGNet

IGNORE BIASES

				# params
INPUT	[224x224x3]	224*224*3	~ 150K	0
CONV (64)	[224x224x64]	224*224*64	~ 3.2M	(3*3*3)*64 = 1,728
CONV (64)	[224x224x64]	224*224*64	~ 3.2M	(3*3*64)*64 = 36,864
POOL	[112x112x64]	112*112*64	~ 800K	0
CONV (128)	[112x112x128]	112*112*128	~ 1.6M	(3*3*64)*128 = 73,728
CONV (128)	[112x112x128]	112*112*128	~ 1.6M	(3*3*128)*128 = 147,456
POOL	[56x56x128]	56*56*128	~ 400K	0
CONV (256)	[56x56x256]	56*56*256	~ 800K	(3*3*128)*256 = 294,912
CONV (256)	[56x56x256]	56*56*256	~ 800K	(3*3*256)*256 = 589,824
CONV (256)	[56x56x256]	56*56*256	~ 800K	(3*3*256)*256 = 589,824
POOL	[28x28x256]	28*28*256	~ 200K	0
CONV (512)	[28x28x512]	28*28*512	~ 400K	(3*3*256)*512 = 1,179,648
CONV (512)	[28x28x512]	28*28*512	~ 400K	(3*3*512)*512 = 2,359,296
CONV (512)	[28x28x512]	28*28*512	~ 400K	(3*3*512)*512 = 2,359,296
POOL	[14x14x512]	14*14*512	~ 100K	0
CONV (512)	[14x14x512]	14*14*512	~ 100K	(3*3*512)*512 = 2,359,296
CONV (512)	[14x14x512]	14*14*512	~ 100K	(3*3*512)*512 = 2,359,296
CONV (512)	[14x14x512]	14*14*512	~ 100K	(3*3*512)*512 = 2,359,296
POOL	[7x7x512]	7*7*512	~ 25K	0
FC	[1x1x4096]		4096	7*7*512*4096 = 102,760,448
FC	[1x1x4096]		4096	4096*4096 = 16,777,216
FC	[1x1x1000]		1000	4096*1000 = 4,096,000

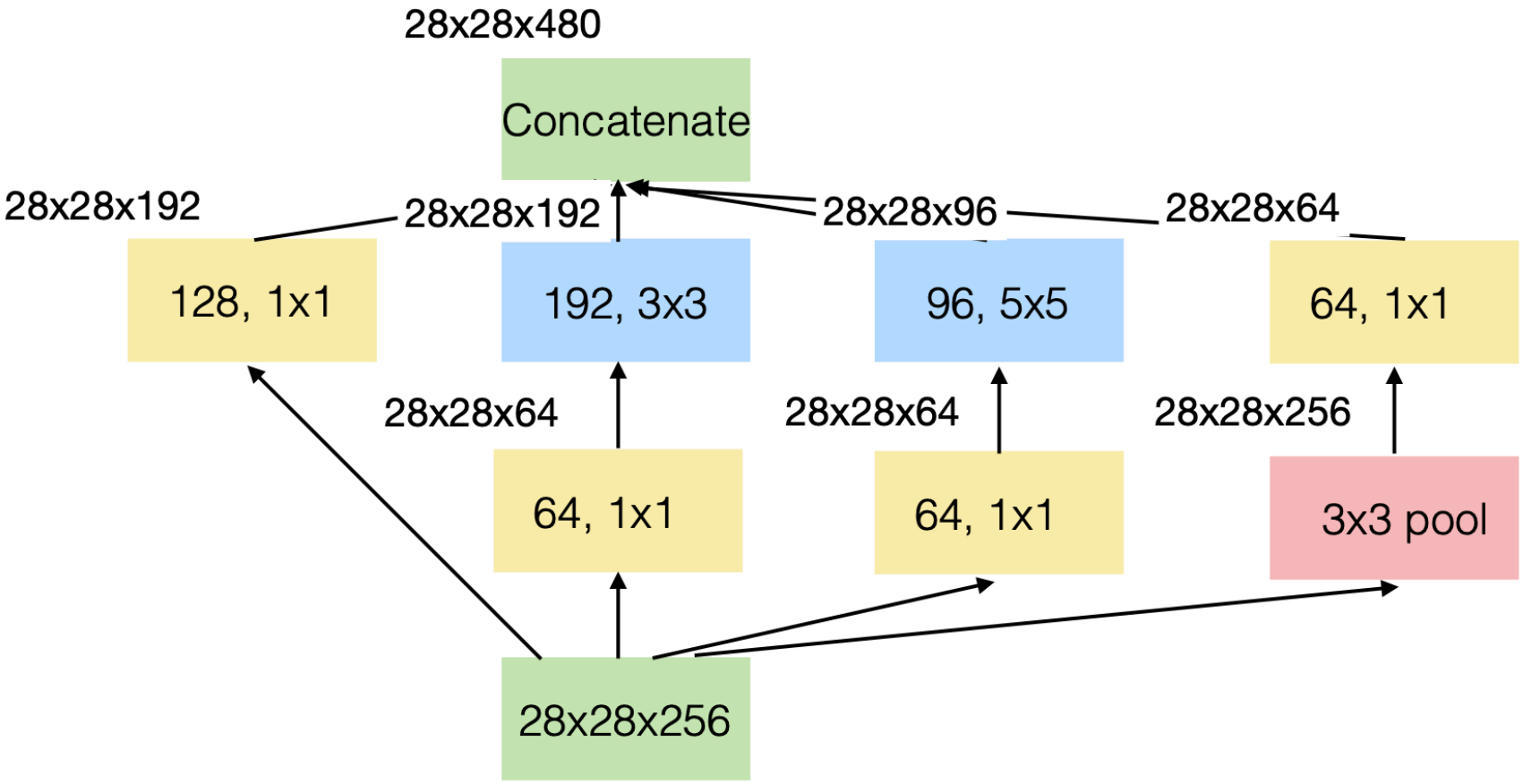
VGGNet: 138 M params

FC layers: 122 M

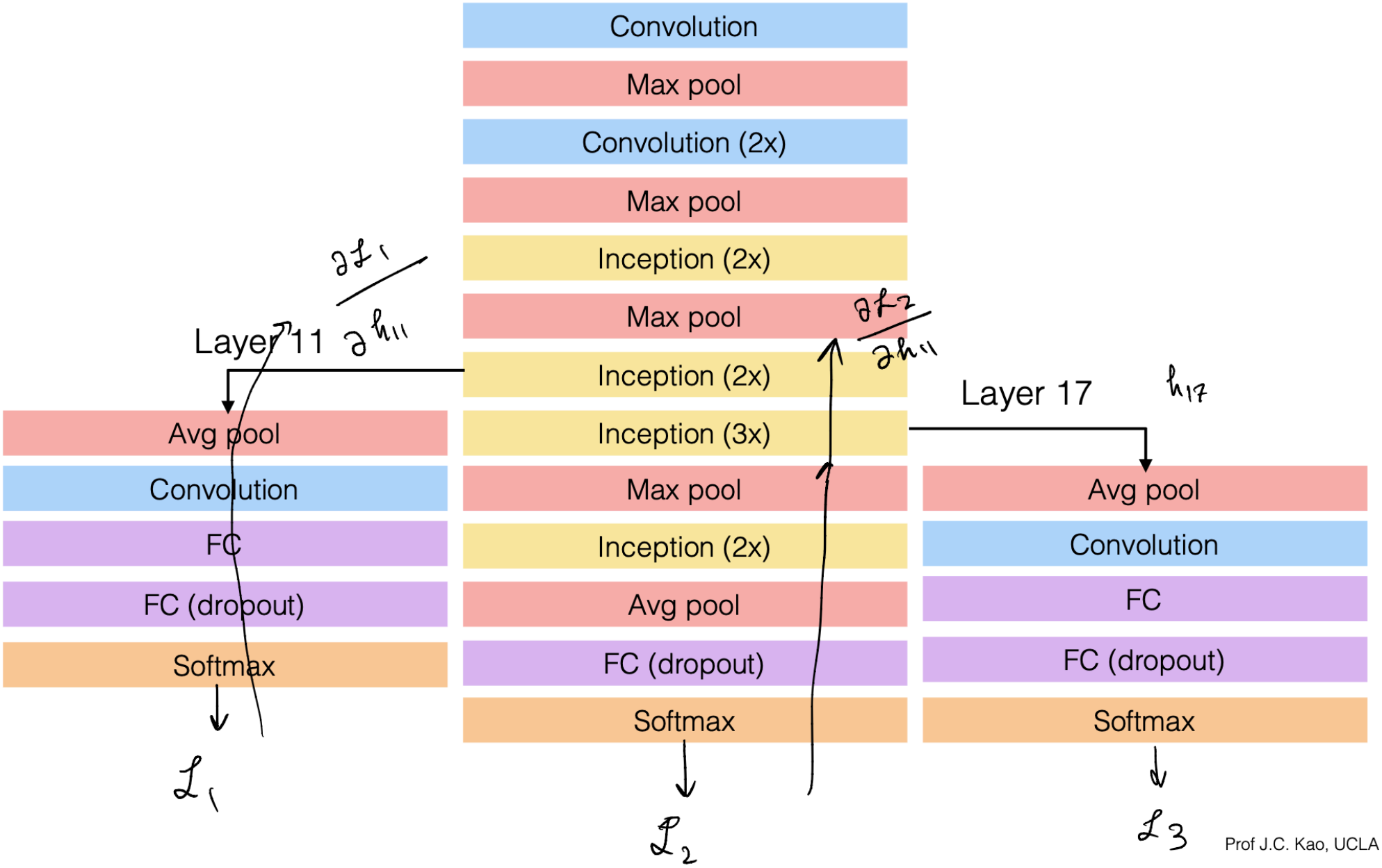
GoogLeNet

To address this, in GoogLeNet, 1x1xF convolutional layers are added, that reduce the number of feature maps to substantially reduce the number of operations.

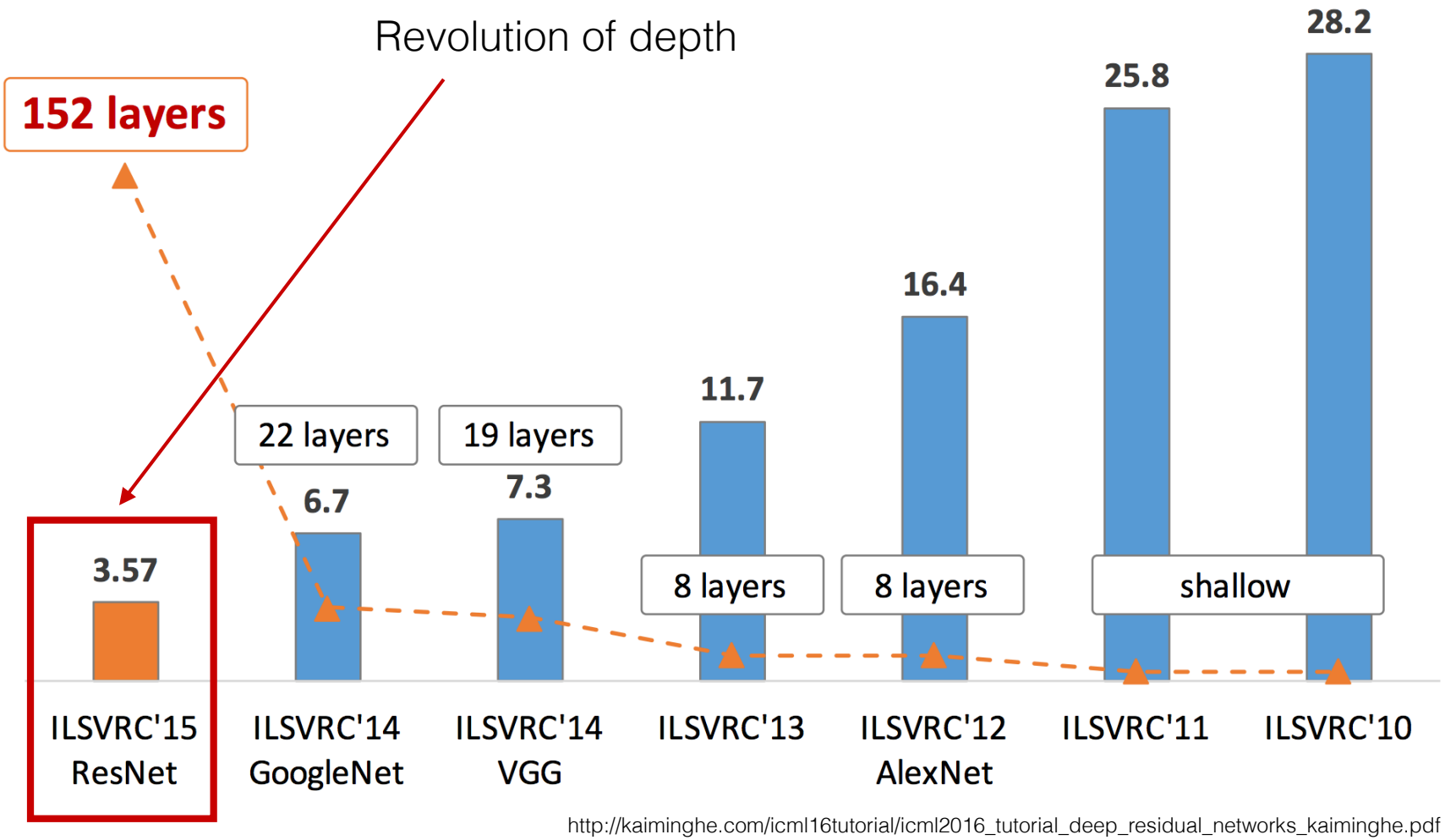
Question: Say F = 64. Where should we put these convolutional layers?



GoogLeNet


Convolution
Max pool
Convolution (2x)
Max pool
Inception (2x)
Max pool
Inception (2x)
Inception (3x)
Max pool
Inception (2x)
Avg pool
FC (dropout)
Softmax
Layer 11
Layer 17
Avg pool
Convolution
FC
FC (dropout)
Softmax
Avg pool
Convolution
FC
FC (dropout)
Softmax

ResNet



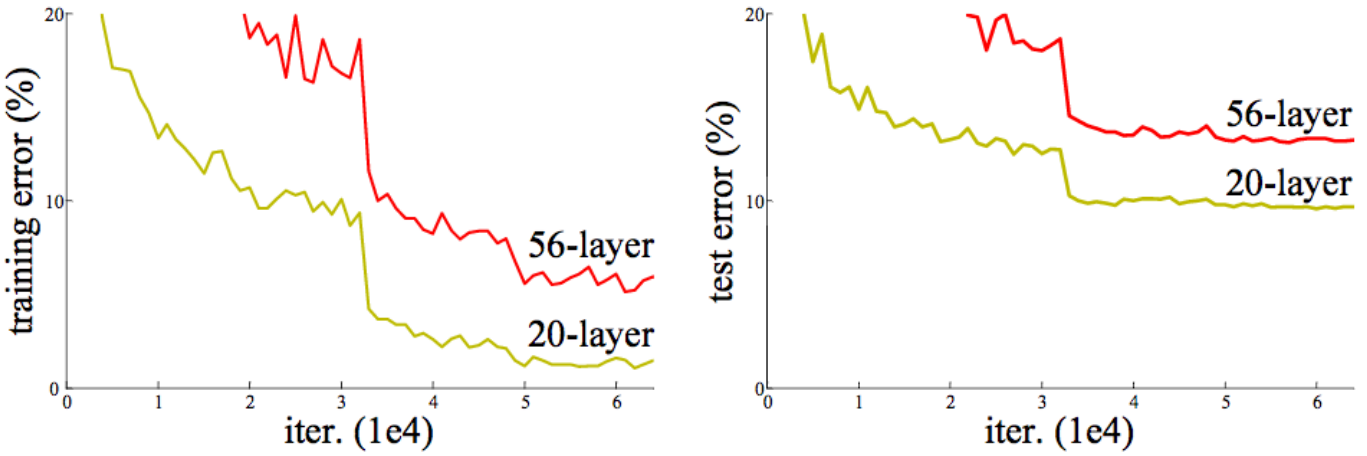
http://kaiminghe.com/icml16tutorial/icml2016_tutorial_deep_residual_networks_kaiminghe.pdf

ResNet

Idea so far:

- AlexNet and ZFNet were 8 layers.
- VGG Net was 16-19 layers.
- GoogLeNet was 22 layers?
- Why not just keep adding layers?

Vanilla CNNs



He et al., 2016

ResNet

This result is non-intuitive. (Why?) Why should a 56-layer NN always do at least as well as a 20-layer NN?

For a 56 layer NN,

I could copy the params of a 20 layer NN and then set the remaining 36 layers to implement the identity fn

ResNet

The degradation problem suggests that the solvers might have difficulties in approximating identity mappings by multiple nonlinear layers. With the residual learning re-formulation, if identity mappings are optimal, the solvers may simply drive the weights of the multiple nonlinear layers toward zero to approach identity mappings.

- He et al., 2016

The goal is to change the NN architecture so that it is easy to learn the identity mapping.

Standard: $h_{i+1} = \text{relu}(Wh_i + b)$

$W \leftarrow W - \epsilon \frac{\partial \mathcal{L}}{\partial W}$

ResNet: $h_{i+1} = h_i + \text{relu}(\tilde{W} h_i + \tilde{b})$

if $\tilde{W}, \tilde{b}$ are small,

then $h_{i+1} \approx h_i$.

ResNet

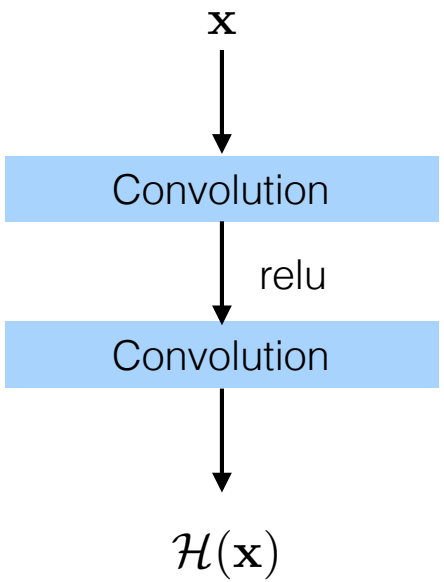
In real cases, it is unlikely that identity mappings are optimal, but our reformulation may help to precondition the problem. If the optimal function is closer to an identity mapping than to a zero mapping, it should be easier for the solver to find the perturbations with reference to an identity mapping, than to learn the function as a new one.

- He et al., 2016

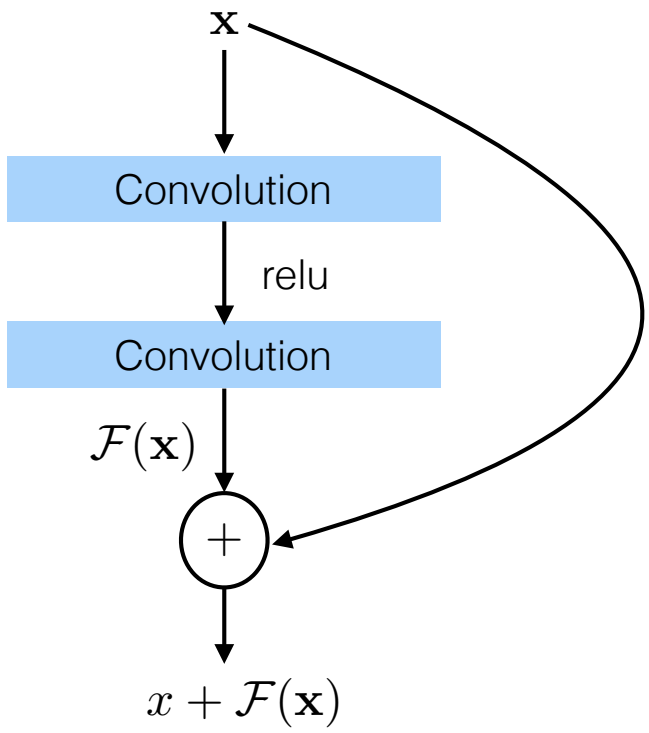
ResNet

The main idea of the ResNet architecture is to facilitate the network training by causing each layer to learn a residual to add to the input.

Normal architecture



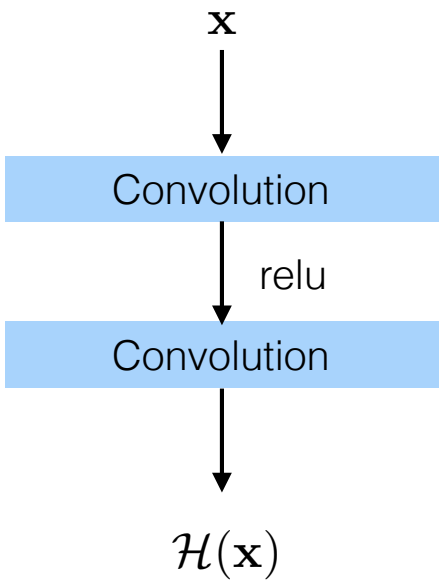
ResNet architecture



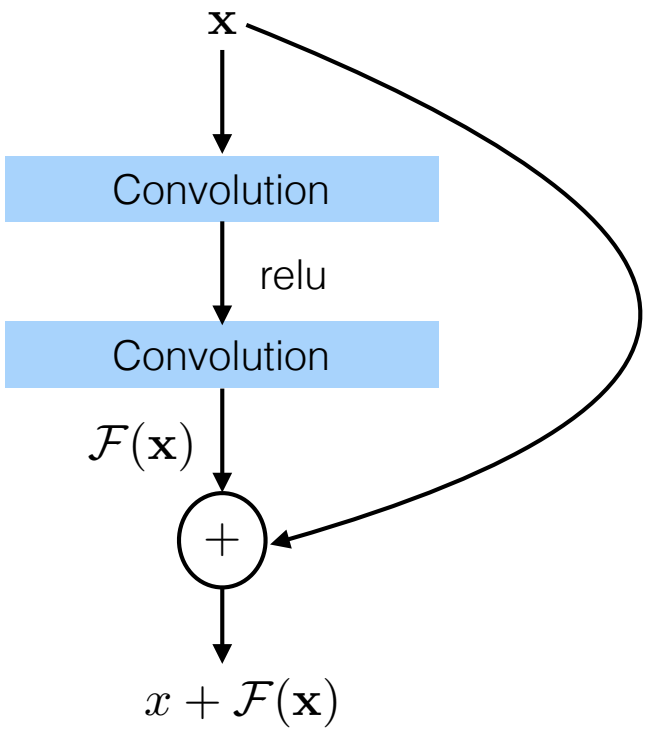
ResNet

The main idea of the ResNet architecture is to facilitate the network training by causing each layer to learn a residual to add to the input.

Normal architecture



ResNet architecture



Residual layers are used to fit $\mathcal{F}(\mathbf{x}) \approx \mathcal{H}(\mathbf{x}) - \mathbf{x}$

To make dimensions work out, sometimes a linear mapping is used: $\mathbf{W}_s\mathbf{x}$

Feature maps are added by doing 1x1 convolutions or padding.

ResNet

"We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping."

"To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers."

(He et al., 2016)

ResNet

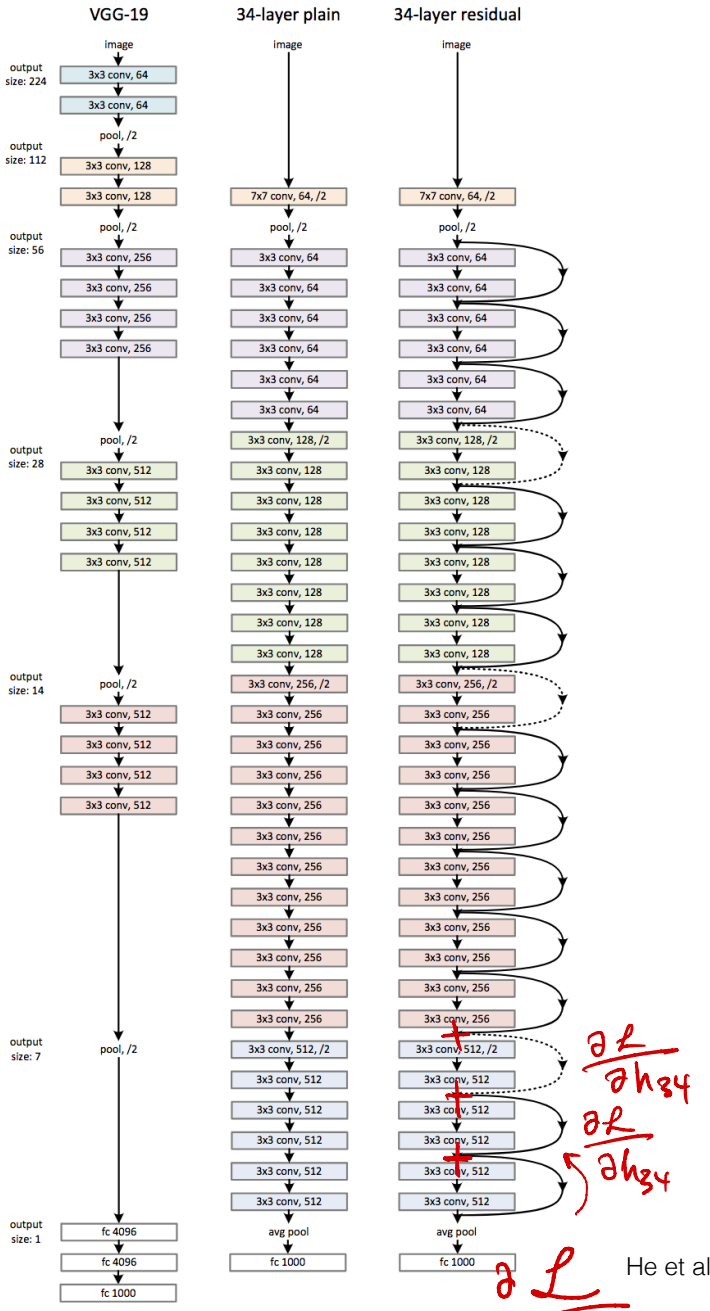
Network architecture:

- They follow the design rules of VGG Net.
  - All conv layers are 3x3 filters with the same number of filters.
  - If the feature map size is halved, the number of filters is doubled so that the computational complexity in each layer is the same.
  - The output ends with average pooling and then a FC 1000 layer to a Softmax.
- Conv layers are residual network layers.

Other notes:

- They performed data augmentation (image scaling, different crops)
- SGD with momentum 0.9, with mini batch size 256.
- L2 regularization with weight 0.0001
- Learning rate starts off at 0.1 and is decreased by an order of magnitude when the error plateaus.
- No dropout.

ResNet

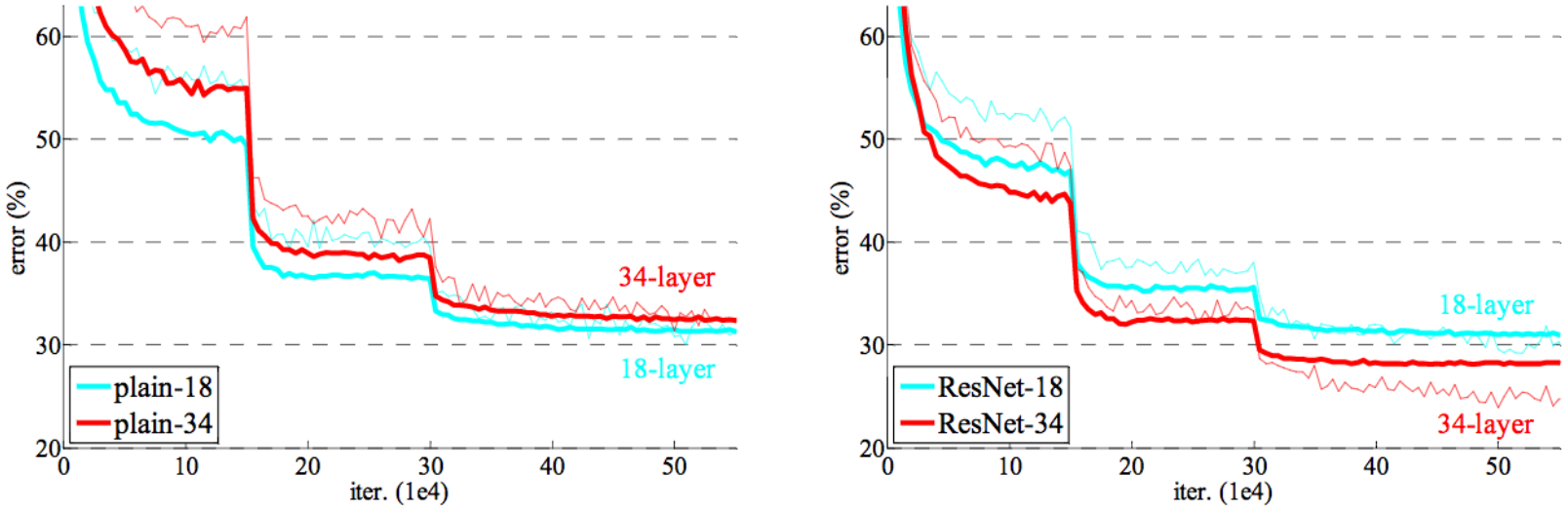


He et al., 2016

ResNet

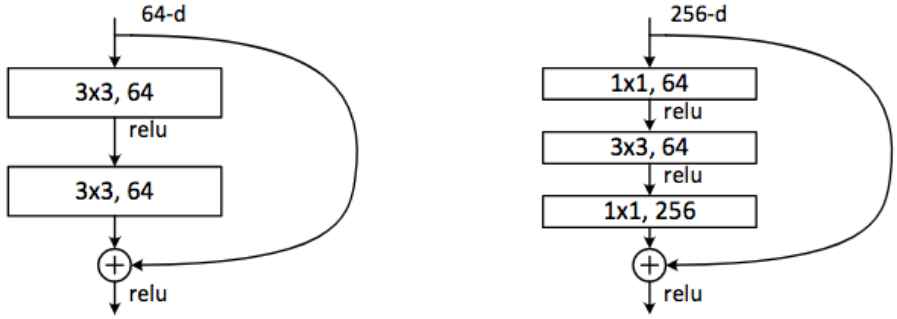
Vanilla CNN

ResNet CNN



He et al., 2016

For deeper networks, use an idea similar to inception:



He et al., 2016

ResNet

CIFAR-10 performance:

Vanilla CNNs

ResNet

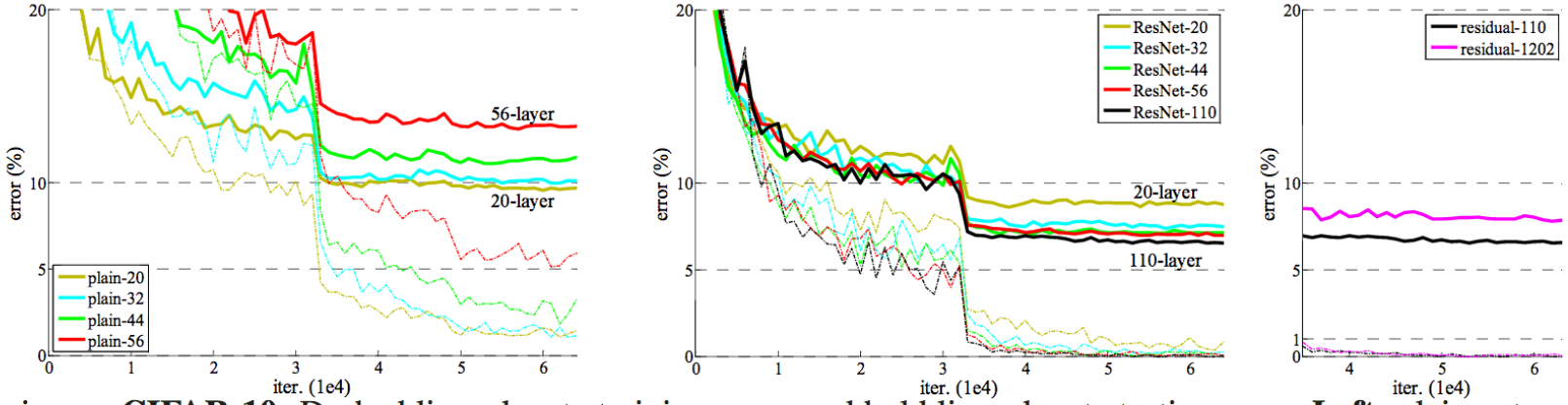


Figure 6. Training on **CIFAR-10**. Dashed lines denote training error, and bold lines denote testing error. **Left**: plain networks. The error of plain-110 is higher than 60% and not displayed. **Middle**: ResNets. **Right**: ResNets with 110 and 1202 layers.

He et al., 2016

Fractal Nets

"…thereby demonstrating that ***residual representations may not be fundamental to the success of extremely deep convolutional neural networks. Rather, the key may be the ability to transition, during training, from effectively shallow to deep.****"*

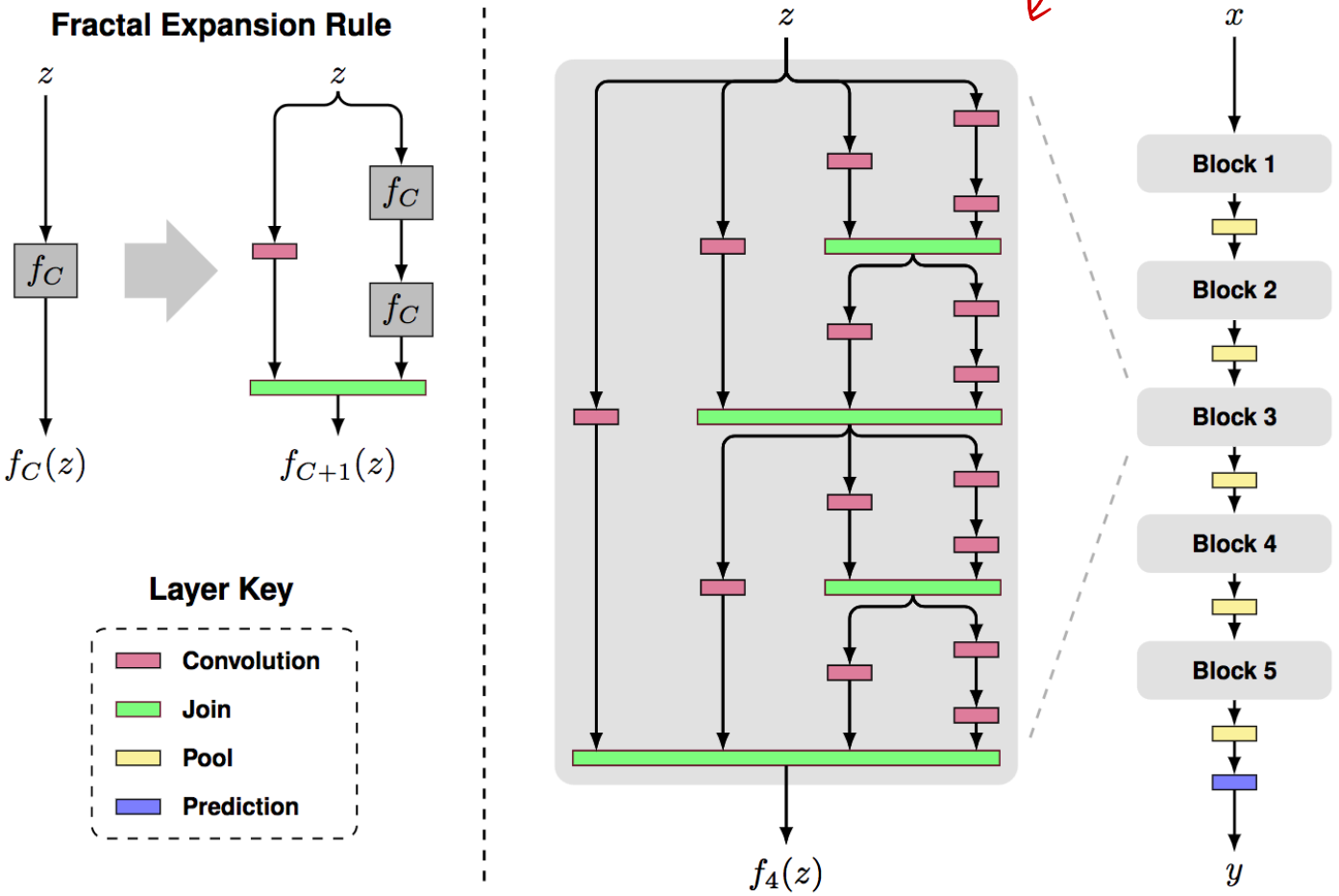
Larsson et al., 2017

Regarding ResNet:

First, the objective changes to learning residual outputs, rather than unreferenced absolute mappings. Second, these networks exhibit a type of deep supervision (Lee et al., 2014), as near-identity layers effectively reduce distance to the loss. ***He et al. (2016a) speculate that the former, the residual formulation itself, is crucial.***

Larsson et al., 2017

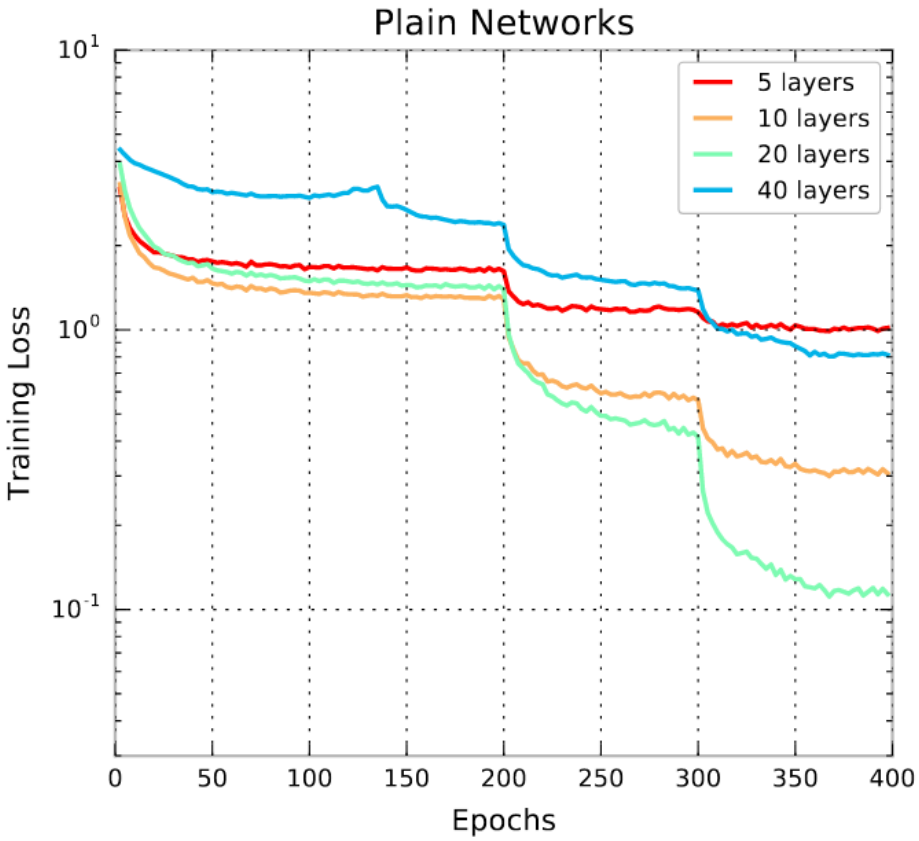
Fractal Nets

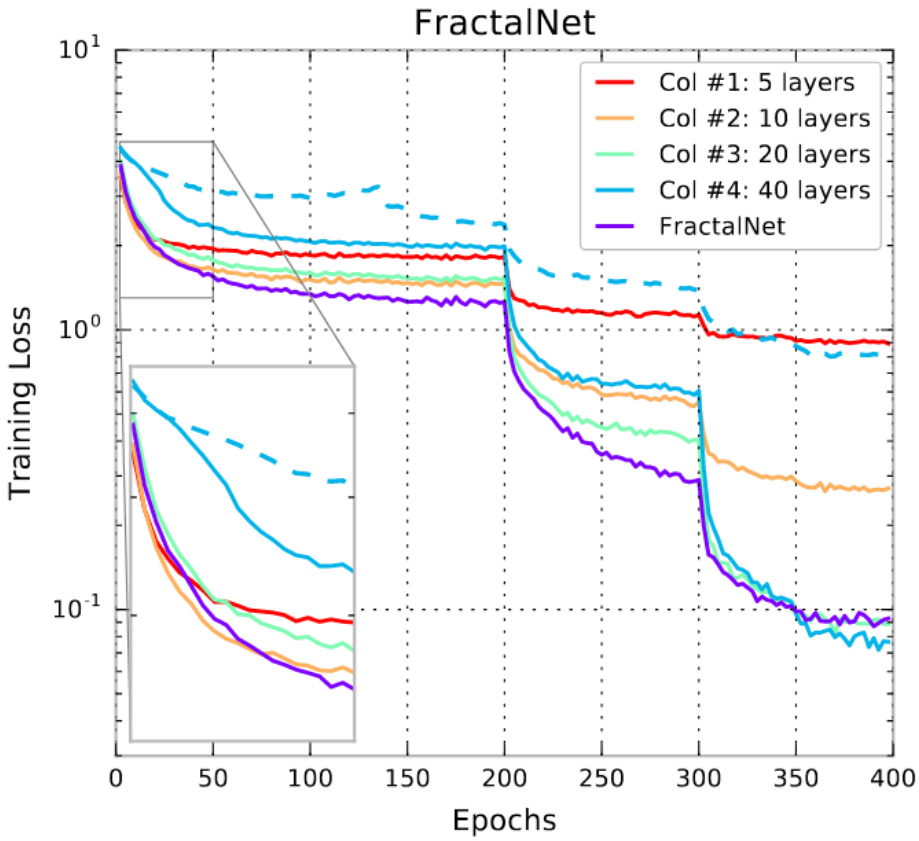


Larsson et al., 2017

Fractal Nets

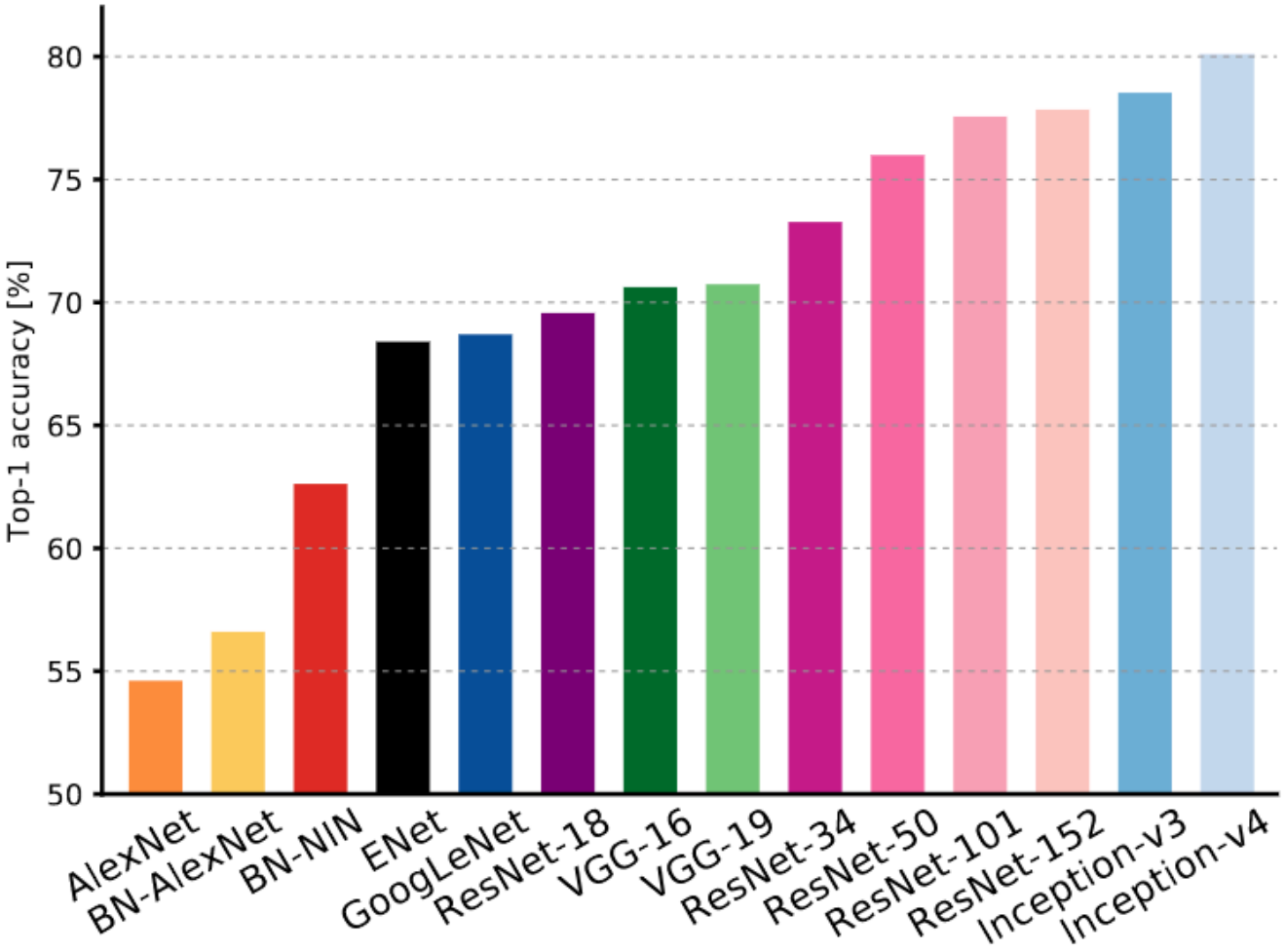
Vanilla CNNs





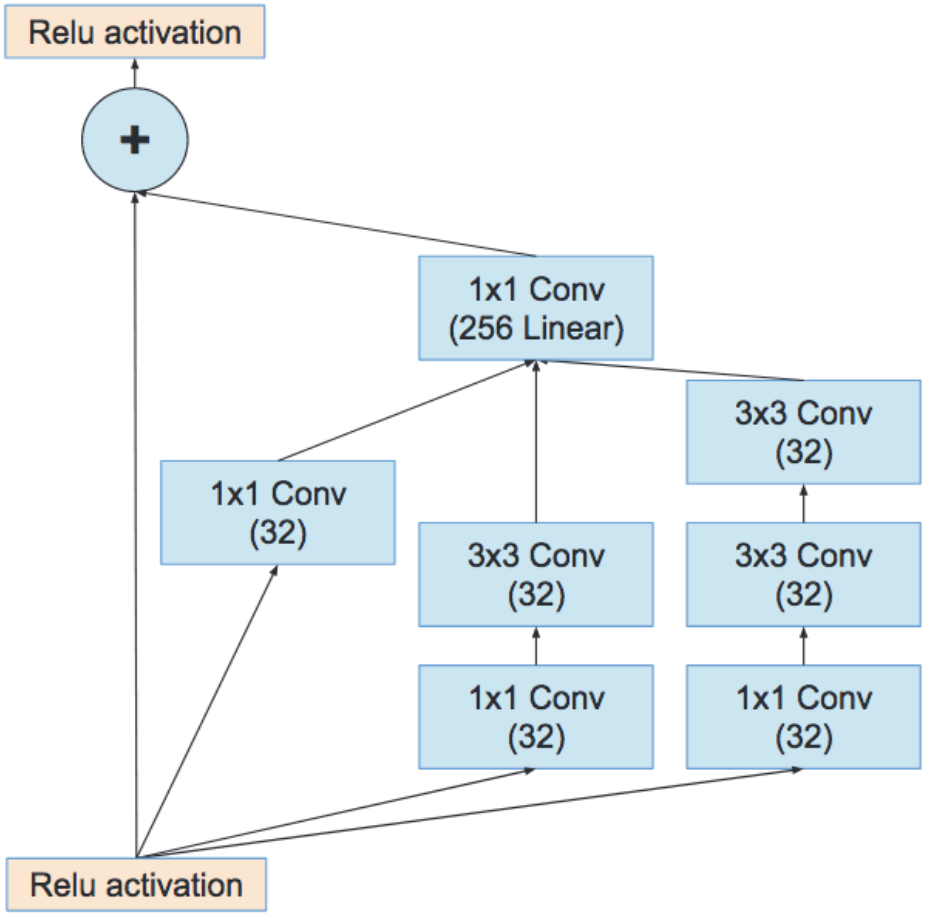
Larsson et al., 2017

An overall view of architectures



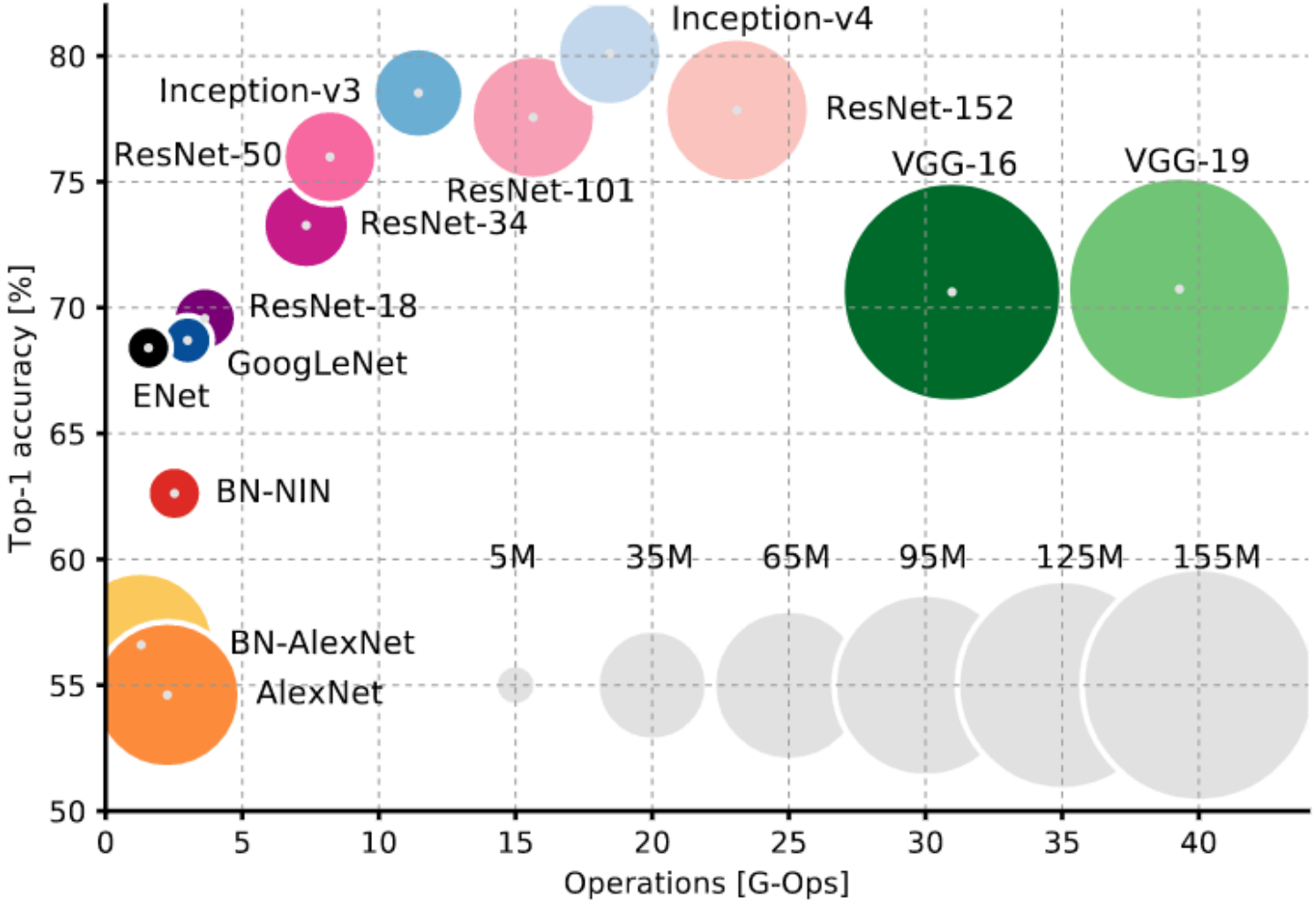
Canziani et al., 2017

Inception-v4



Szegedy et al., 2017

An overall view of architectures



Canziani et al., 2017

Take home points

For convolutional neural networks:

- Architecture can play a key role in the performance of the network.
- There is evidence that deeper and wider (i.e., more feature maps) result in better performance.
- Going deeper helps to a point; beyond that, new architectures have to be considered (like the ResNet).
- It appears that what is key about these different architectures is that they reduce the *effective depth* of the network, i.e., they shorten the longest path from the output loss to the network input. This helps to avoid the fundamental problem of deep learning.

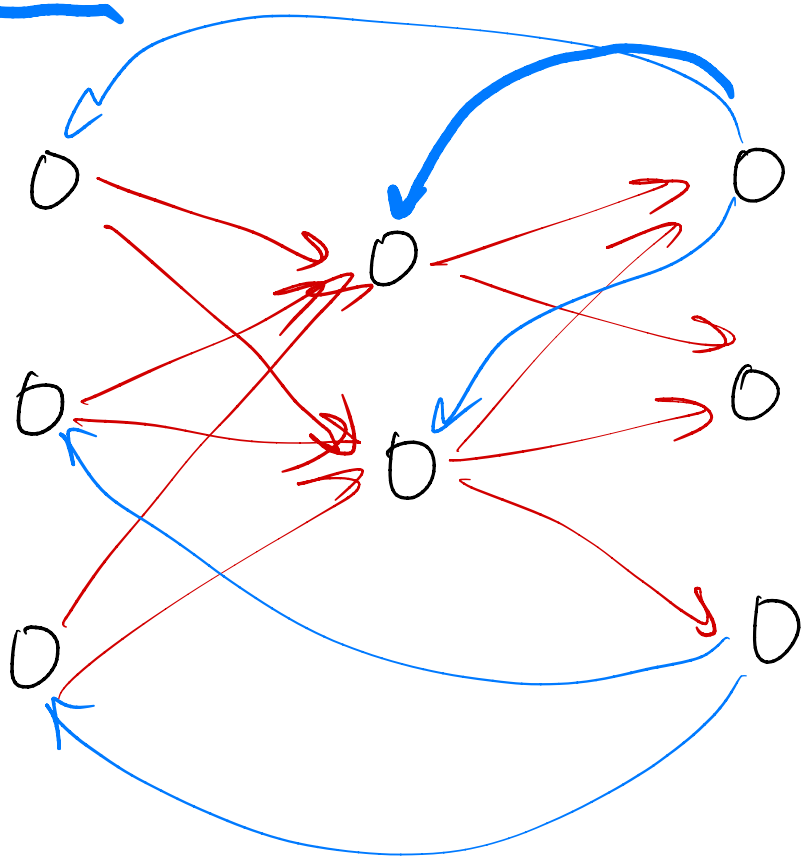
Recurrent neural networks

In this lecture, we'll talk about recurrent neural networks. In particular:

- Why RNNs?
- The basic RNN structure.
- Backpropagation through time.
- LSTMs
- GRUs

Why recurrent neural networks?

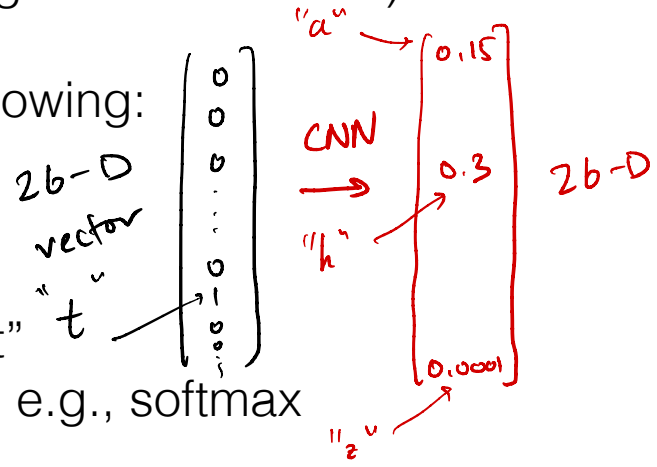
A key missing feature of FC and convolutional neural networks is that they lack recurrent connectivity, and hence have no dynamics.



A motivating thought experiment

Say we're training a neural network to do character prediction (or word prediction, e.g., when you type on your phone and it suggests a next word).

In the case of character prediction, we have done the following:



1) Downloaded every NY Times article ever written.
2) Trained a CNN to do the following:
    1) Take as input one character, e.g., the letter "t"
    2) Output a distribution over the next character, e.g., softmax probabilities on the 26 letters in the alphabet.
    3) From here, take the character with the highest probability, e.g., we may find it's the letter "h" because in the English language, h commonly follows t.
3) Question for you. Consider the next character to be output following these two strings:
    1) "th"
    2) "though"

same input "h" → CNN → the same softmax distr.

4) If you trained a CNN with all the bells and whistles to make it as good as possible in prediction, what ought the output look like?

A motivating thought experiment

This gets to the problem of state.

The CNN will always produce the same output given the same input, irrespective of what happened in the past.

In some cases, this is a totally fine thing, e.g., classifying images that don't have temporal structure.

But what if we wanted to classify *videos*? What if we wanted to generate *text*? What if what happened in the past matters?

At this point, we need a new construction.

A motivating thought experiment

What is state?

State captures — usually in a succinct manner — what has happened in the past.

s_t

For example, say we want to infer some output $\mathbf{z}_t$ from some inputs $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$

A motivating thought experiment

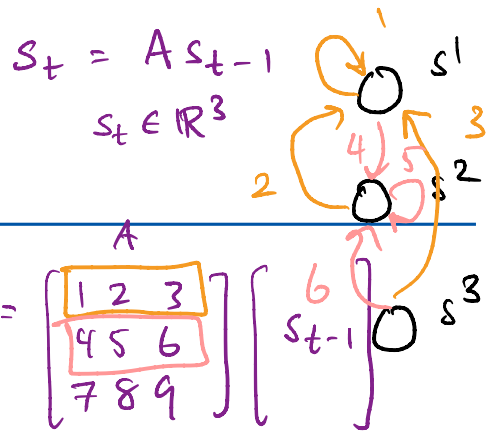
There are a few ways you could think of doing this. One way is to define a function that takes all of the history and produces an output.

$$\mathbf{z}_t = f(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t)$$

Then this f() could be, e.g., a CNN or a FC net!

So haven't we solved the problem?

A motivating thought experiment



The other way is to introduce a variable called state.

The state is influenced by its past value(s), as well as by the current input.

Usually, we also make what is called the Markov assumption, which states that all information about my past inputs is stored in my current state.

contains ALL relevant info about $x_1, \ldots, x_t$

state var. that contains ALL the relevant info about $x_1, x_2, \ldots, x_{t-1}$ to perform my task

$$\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{x}_t)$$

Here, "s" denotes the state, and critically, $\mathbf{s}_{t-1}$ contains all the information we need to know about $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{t-1}$.

With this more compact representation of the history, we can infer:

$$\mathbf{z}_t = g(\mathbf{s}_{t-1}, \mathbf{x}_t)$$

In short, we encapsulate all the history into this state variable.

Why recurrent neural networks?

Motivation for recurrent neural networks

Recurrent neural networks are neural networks that, in addition to feedforward connections, have feedback connections.

- A network that is purely feedforward has no *internal dynamics*. Such a network always produces the same output $\mathbf{y}_t = f(\mathbf{x}_t)$ no matter the time.
- In a recurrent neural network (RNN), the feedback connections cause input activations to persist for some amount of time. Because artificial units provide inputs to each other with feedback, an input provided to the network will, after one time step, still propagate within the network through its recurrent connectivity.
- In this manner, RNNs have internal dynamics, so that the output at any given time is also a function of the activation of the hidden units at that time, i.e., $\mathbf{z}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$.

Why recurrent neural networks?

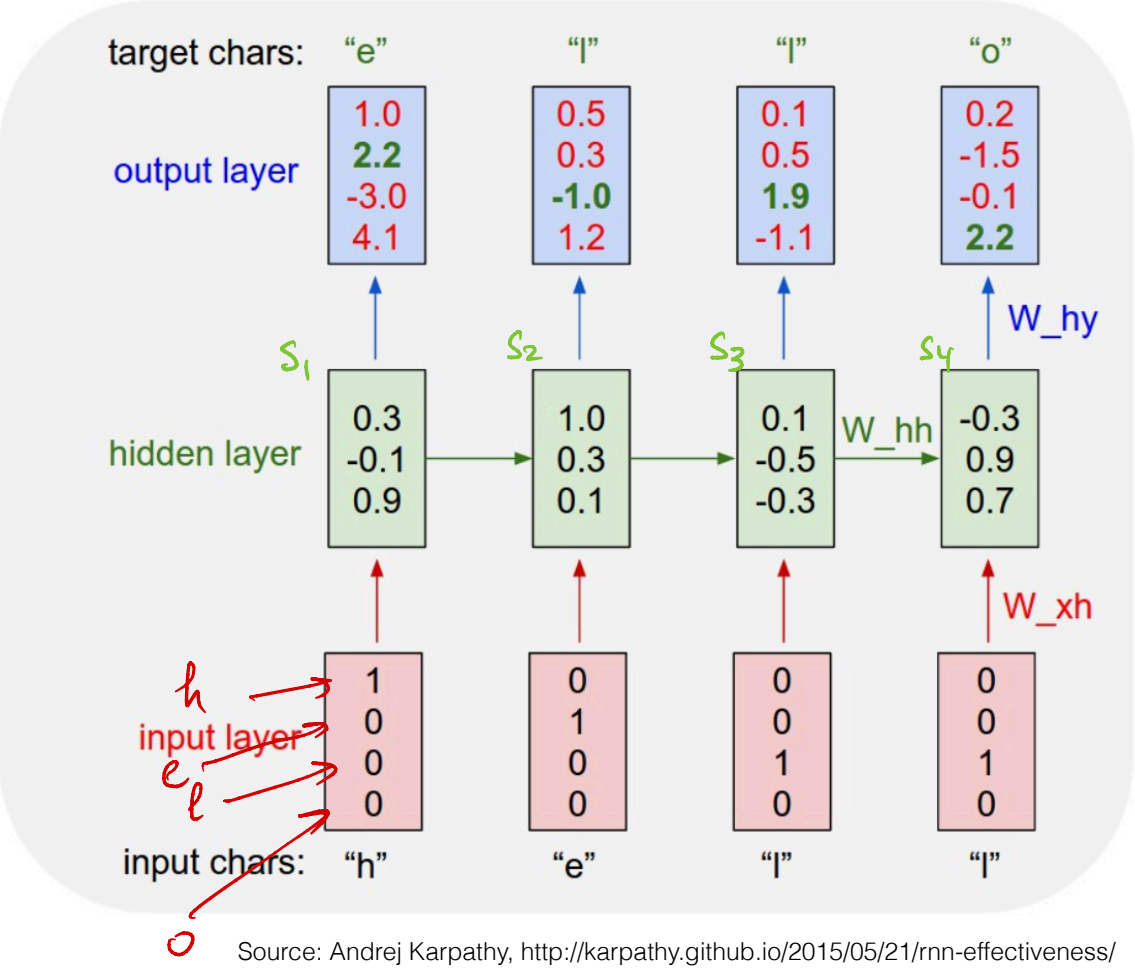
Connection to biology

In biology, neurons are known to be recurrently connected. Evidence suggests that while earlier areas of sensorimotor processing have more to do with encoding external stimuli, later areas such as the motor cortex are inherently *dynamical*. In such dynamical circuits, neurons, through their recurrent connectivity, drive themselves in lawful ways through time.

In a similar way, recurrent neural networks have recurrent connectivity that define a dynamical system that governs how it evolves through time. Note that a CNN does not capture these features of neural activity.

Some cool results

hello



Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

"The unreasonable effectiveness of RNNs"

Learning Shakespeare

100 iters

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

train more

300 iters

```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

train more

700 iters

```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

train more

2000 iters

```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Learning Shakespeare

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Learning Shakespeare

```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Generating Wikipedia

```
Naturalism and decision for the majority of Arab countries' capitalide was grounded
by the Irish language by [[John Clair]], [[An Imperial Japanese Revolt]], associated
with Guangzham's sovereignty. His generals were the powerful ruler of the Portugal
in the [[Protestant Immineners]], which could be said to be directly in Cantonese
Communication, which followed a ceremony and set inspired prison, training. The
emperor travelled back to [[Antioch, Perth, October 25|21]] to note, the Kingdom
of Costa Rica, unsuccessful fashioned the [[Thrales]], [[Cynth's Dajoard]], known
in western [[Scotland]], near Italy to the conquest of India with the conflict.
Copyright was the succession of independence in the slop of Syrian influence that
was a famous German movement based on a more popular servicious, non-doctrinal
and sexual power post. Many governments recognize the military housing of the
[[Civil Liberalization and Infantry Resolution 265 National Party in Hungary]],
that is sympathetic to be to the [[Punjab Resolution]]
(PJS)[http://www.humah.yahoo.com/guardian.
cfm/7754800786d17551963s89.htm Official economics Adjoint for the Nazism, Montgomery
was swear to advance to the resources for those Socialism's rule,
was starting to signing a major tripad of aid exile.]]
```

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Faked algebraic geometry

```
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on $\mathcal{C}$.
\item Given a morphism $\Delta : \mathcal{F} \to \mathcal{I}$
is an injective and let $\mathfrak q$ be an abelian sheaf on $X$.
Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a category.
\begin{enumerate}
\item \hyperref[setain-construction-phantom]{Lemma}
\label{lemma-characterize-quasi-finite}
Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on $\mathcal{C}$.
Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
$\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
\item The following are equivalent
\begin{enumerate}
\item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
\end{lemma}
```

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Faked algebraic geometry

For $\bigoplus_{n=1,\ldots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the compariciol y in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

is an open subset of X. Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S.

Proof. See discussion of sheaves of sets. □

The result for prove any open covering follows from the less of Example ??. It may replace S by $X_{spaces,étale}$ which gives an open subspace of X and T equal to S_{Zar}, see Descent, Lemma ??. Namely, by Lemma ?? we see that R is geometrically regular over S.

Lemma 0.1. *Assume (3) and (3) by the construction in the description.*

Suppose $X = \lim |X|$ *(by the formal open covering* X *and a single map* $\underline{Proj}_X(\mathcal{A}) = \mathrm{Spec}(B)$ *over* U *compatible with the complex*

$$Set(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X,\mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_{Z/X}$ *is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If* T *is surjective we may assume that* T *is connected with residue fields of* S*. Moreover there exists a closed subspace* $Z \subset X$ *of* X *where* U *in* X' *is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem*

(1) f *is locally of finite type. Since* $S = \mathrm{Spec}(R)$ *and* $Y = \mathrm{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X. But given a scheme U and a surjective étale morphism $U \to X$. Let $U \cap U = \coprod_{i=1,\ldots,n} U_i$ be the scheme X over S at the schemes $X_i \to X$ and $U = \lim_i X_i$. □

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\ldots,0}$.

Lemma 0.2. *Let* X *be a locally Noetherian scheme over* S, $E = \mathcal{F}_{X/S}$. *Set* $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$. *Since* $\mathcal{I}^n \subset \mathcal{I}^n$ *are nonzero over* $i_0 \leq \mathfrak{p}$ *is a subset of* $\mathcal{J}_{n,0} \circ \overline{A}_2$ *works.*

Lemma 0.3. *In Situation ??. Hence we may assume* $\mathfrak{q}' = 0$.

Proof. We will use the property we see that $\mathfrak{p}$ is the mext functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where K is an F-algebra where δ_{n+1} is a scheme over S. □

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Faked algebraic geometry

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

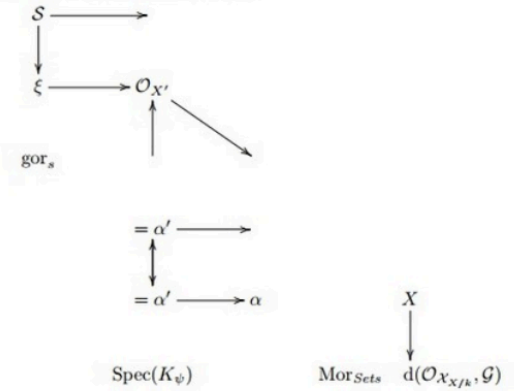
be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_t}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{n}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Learning long term dependencies

Cell that turns on inside quotes:

```
"You mean to imply that I have nothing to eat out of.... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."
```

Cell that is sensitive to the depth of an expression:

```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

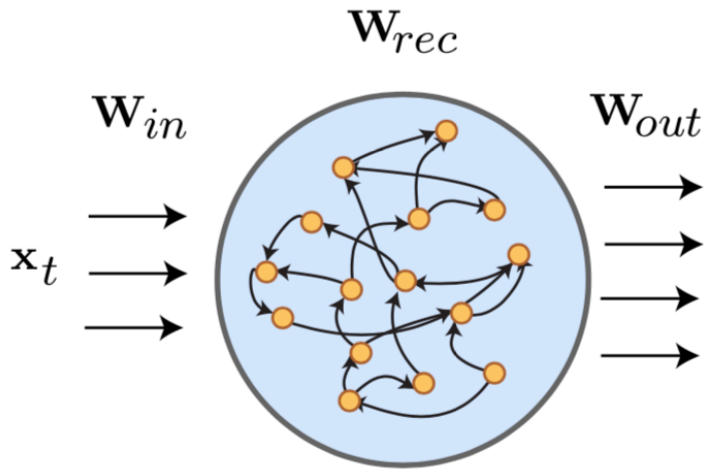
Source: Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Vanilla RNN architecture

state at time t

$$h_t = \text{relu}\left(W_{rec} h_{t-1} + W_{in} x_t + b\right)$$

At a high-level, the RNN can be diagrammed as follows:



$h_t \in \mathbb{R}^D \quad \forall t$

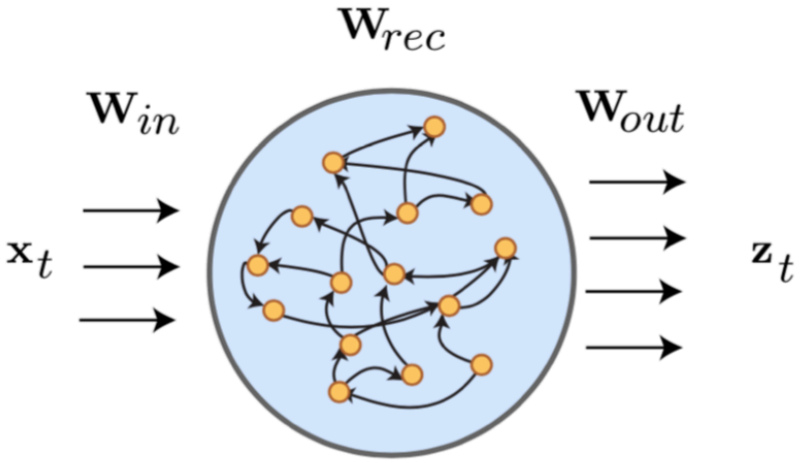
$$z_t = W_{out} h_t + b_{out}$$

$z_t \in \mathbb{R}^{26}$, W_{out}: $26 \times D$, $h_t \in \mathbb{R}^D$

The RNN has three major components:

- $\mathbf{W}_{\text{in}}$: An input at time t, denoted $\mathbf{x}_t$, is transformed via $\mathbf{W}_{\text{in}}$ onto artificial neurons, whose activations are $\mathbf{h}_t$.
- $\mathbf{W}_{\text{rec}}$: Each artificial neuron is the network is denoted by an orange circle, and these artificial neurons have recurrent connections. recurrent connections are defined by the matrix $\mathbf{W}_{\text{rec}}$.
- $\mathbf{W}_{\text{out}}$: Finally, the artificial neuron activations are mapped linearly to the output $\mathbf{z}_t$ through the matrix $\mathbf{W}_{\text{out}}$.

Vanilla RNN



$$\mathbf{h}_t = f(\mathbf{W}_{\text{rec}}\mathbf{h}_{t-1} + \mathbf{b}_{\text{rec}} + \mathbf{W}_{\text{in}}\mathbf{x}_t)$$

Note, this is the vanilla RNN formulation used in Goodfellow (equations 10.8 and 10.9).

Output of the RNN is based on its state

RNN output

The output of the RNN is typically a linear mapping of the RNN's activations.

$$\mathbf{z}_t = \mathbf{W}_{\text{out}}\mathbf{h}_t + \mathbf{b}_{\text{out}}$$

This can be used e.g., for regression, or the linear outputs could e.g., be passed to a softmax classifier if the goal is to classify each time point. For example, the probability of class i at time t can be denoted via:

$$\hat{y}_{t,i} = \text{softmax}_i(\mathbf{z}_t)$$

Loss functions are usually accumulated through time

RNN cost function

Loss functions are straightforward:

- In the case of regression, loss functions include the mean-square error, where $\hat{y}_t = \mathbf{z}_t$ and the cost function is to minimize the sum of residuals

$$\sum_t \|\hat{\mathbf{y}}_t - \mathbf{y}_t\|^2$$

  across all time (modulo some scaling constant).

- In the case of classification, loss functions include those derived from maximum-likelihood. If $\mathcal{L}_t$ is the softmax loss at time t, then the cost function can be to minimize the sum of losses

$$\sum_t \mathcal{L}_t$$

50 chars → RNN → 51st char.

$\mathcal{L}_1$ $\mathcal{L}_2$ $\mathcal{L}_3$ ... $\mathcal{L}_{51}$

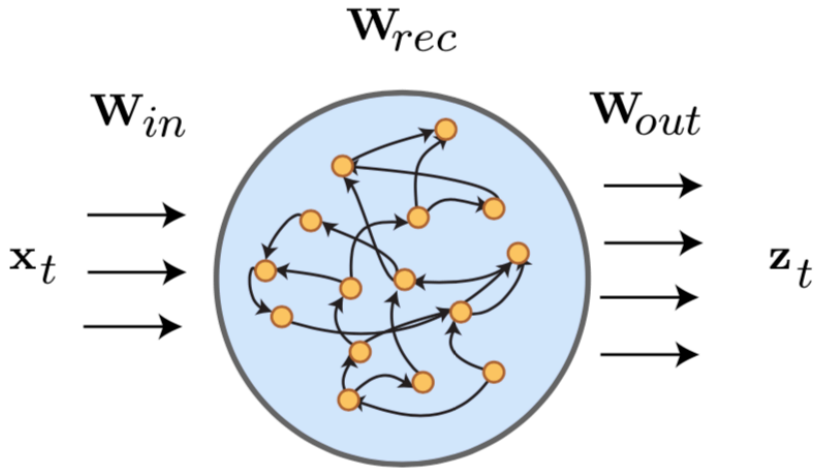
  across all time (modulo some scaling constant).

- Sometimes, we only care about the output after some time τ. Maybe we'll even just care about the last output at the horizon of the data, T. In these scenarios, the loss would be

$$\sum_{t \geq 30}^{51} \mathcal{L}_t$$

30 - 51

What do we need to train an RNN?



Let's take stock of what we know:

- We know the RNN equations, and we can define a loss function.
  - **So we know how to do a forward pass and calculate a loss.**

- In general, we know how to do optimization (i.e., with SGD and your favorite optimizer on top of that, e.g., Adam or RMSprop).

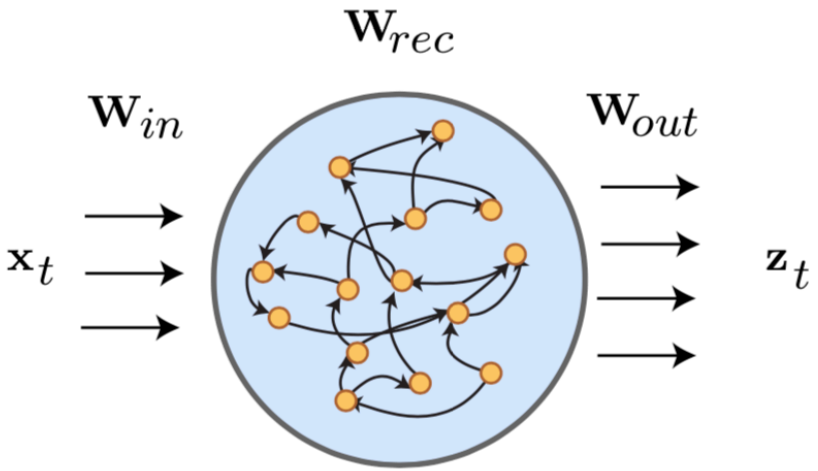
- Do we know how to take gradients of the weight matrices?

- Is there any problem in applying backpropagation as in feedforward networks (e.g., CNNs, FC nets) to RNNs?

What do we need to train an RNN?

RNN training

Training an RNN is not immediately as straightforward as a feedforward neural network. This is because the RNN has recurrent connections with loops, and backpropagation is not straightforward.



The upstream gradients at any given time come from units who are themselves potentially receiving inputs (directly or indirectly) from the node we're trying to calculate the gradient of. Further, the activations at any given node for an input depends on time; for even an input $\mathbf{x}_t$ that is static across time, the activations $\mathbf{h}_t$ will not be static across time.

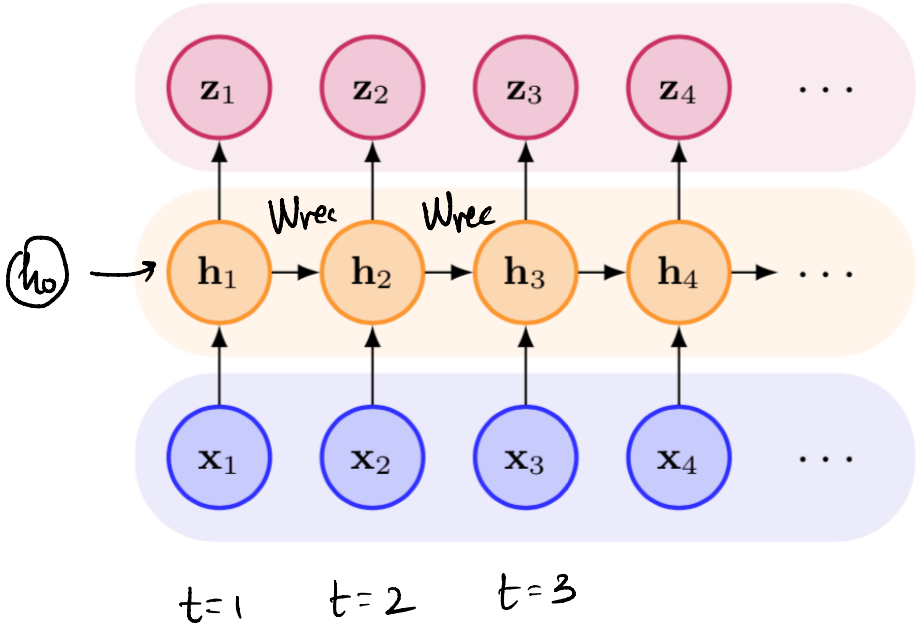
Multiplicative RNN, $W_{rec}(t)$

Key insight: unroll the computational graph

$$h_t = \text{relu}\left(W_{rec}\, h_{t-1} + W_{in}\, x_t\right)$$

RNN training (cont.)

To get around this confound, we consider the RNN as a computational graph through time.



h_0, x_1

h_1, x_2

h_2

"Backpropagation through time" BPTT