Seminar in Deep Reinforcement Learning

Part I
Deep Learning and Neural Architecture

Zhao MA
25.02.2020
Deep Learning in One Slide

• What is it:
  Extract useful patterns from data.
• How:
  Neural network + optimization
• How (Practical):
  Python + TensorFlow & friends
• Hard Part:
  Good Questions + Good Data
• Why now:
  Data, hardware, community, tools, investment
• Where do we stand?
  Most big questions of intelligence have not been answered nor properly formulated

Exciting progress:
• Face recognition
• Image classification
• Speech recognition
• Text-to-speech generation
• Handwriting transcription
• Machine translation
• Medical diagnosis
• Cars: drivable area, lane keeping
• Digital assistants
• Ads, search, social recommendations
• Game playing with deep RL
Deep Learning is **Representation Learning**

*Representation Learning:*

*the automated formation of useful representations from data.*
Task: Draw a line to separate the blue curve and red curve
Representation Matters

How we represent the world can make the complex appear simple both to us humans and to the machine learning models we build.
“AI began with an ancient wish to forge the gods.”
- Pamela McCorduck, *Machines Who Think*, 1979

3% of the neurons and 0.0001% of the synapses in the brain.

*Thalamocortical system visualization via DigiCortexEngine.*

Visualization of MNIST dataset classification.

*www.cybercontrols.org*
Neuron: Biological Inspiration for Computation

Neuron: computational building block for the brain.

(Artificial) Neuron: computational building block for the “neural network”

Why does it work?
Content

Common DL Architectures

Universal Approximation Theorem

Selected NNs in details
- CNN (+ResNet), RNN (+LSTM), Transformer (+Attention)

Deep Double Decent
Universal approximation theorem

“A feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$, under mild assumptions on the activation function.”

Two caveats of “any function”:
1. “approximation” instead of “exactly”;
2. the continuous functions;

-- Universal Approximation Theorem, Wikipedia

Lazy version:
“A Neural Network can approximate almost any functions.”
Universal approximation theorem

how to construct a neural network which approximates a function with just one input and one output
Universal approximation theorem
Universal approximation theorem

\[ y = \frac{1}{1+e^{-(w^T x + b)}} \]
Supervised Learning

1. Feed Forward Neural Networks
   - Input: A few numbers
   - Network: Dense Encoder
   - Representation
   - Output: Prediction
   - Ground Truth: Prediction

2. Convolutional Neural Networks
   - Input: An image
   - Network: Convolutional Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

3. Recurrent Neural Networks
   - Input: Sequence
   - Network: Recurrent Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

4. Encoder-Decoder Architectures
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Output: Image, Text, etc.
   - Ground Truth: Image, Text, etc.

Unsupervised Learning

5. Autoencoder
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Network: Exact copy of input
   - Ground Truth: Exact copy of input

6. Generative Adversarial Networks
   - Input: Noise
   - Network: Generator
   - Output: Fake Image
   - Network: Discriminator
   - Prediction: Real or Fake
   - Ground Truth: Real Image

Reinforcement Learning

7. Networks for Actions, Values, Policies, and Models
   - Input: World State Sample
   - Network: Any Encoder
   - Representation
   - Action
   - Reward
   - Ground Truth: Reward
FFNNs

1. Feed Forward Neural Networks

- dating back to 1940s;
- data passes from input to output in a **single pass** without any “state memory” of what came before.
2. Convolutional Neural Networks

- Densely-connected layers + convolutional layers (convolutional encoder).
- Feed forward neural networks that use a spatial-invariance trick to efficiently learn local patterns; (most commonly, in images)
RNNs

- Have cycles and therefore have “state memory”;
- Can be unrolled in time to become feed forward networks where the weights are shared;
- CNN – weights shared across “space” v.s. RNN – weights shared across “time”; → sequential data
Encoder-Decoder Architectures

- FFNNs – dense encoder, CNNs – convolutional encoder, RNNs – recurrent encoder;
- Encoder: find patterns in raw data to form compact, useful representations;
- Decoder: generate high-resolution data from those representations.

*e.g. image caption: encoder-CNN, decoder-RNN*;
Common architecture of neural networks

1. **Feed Forward Neural Networks**
   - Input: A few numbers
   - Network: Dense Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

2. **Convolutional Neural Networks**
   - Input: An image
   - Network: Convolutional Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

3. **Recurrent Neural Networks**
   - Input: Sequence
   - Network: Recurrent Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

4. **Encoder-Decoder Architectures**
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Image, Text, etc.
   - Ground Truth: Image, Text, etc.

5. **Autoencoder**
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Exact copy of input
   - Ground Truth: Exact copy of input

6. **Generative Adversarial Networks**
   - Input: Noise
   - Network: Generator
   - Fake Image
   - Discriminator
   - Prediction: Real or Fake
   - Ground Truth: Real Image

7. **Reinforcement Learning**
   - Input: World State Sample
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Action
   - Reward
   - Ground Truth: Reward
Autoencoders

5. Autoencoder

- **self-supervised**: the ground truth data comes from the input data, no human effort is required;
- Application: unsupervised embeddings, image denoising, etc.
Generative Adversarial Networks (GANs)

a framework for training networks optimized for generating new realistic samples from a particular representation.

images generated by BigGAN.
The GAN Zoo

- 500+ different named GAN variations.
Common architecture of neural networks

**Supervised Learning**

1. **Feed Forward Neural Networks**
   - Input: A few numbers
   - Network: Dense Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

2. **Convolutional Neural Networks**
   - Input: An image
   - Network: Convolutional Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

3. **Recurrent Neural Networks**
   - Input: Sequence
   - Network: Recurrent Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

4. **Encoder-Decoder Architectures**
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Image, Text, etc.
   - Ground Truth: Image, Text, etc.

**Unsupervised Learning**

5. **Autoencoder**
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Exact copy of input
   - Ground Truth: Exact copy of input

6. **Generative Adversarial Networks**
   - Input: Noise
   - Network: Generator
   - Fake Image
   - Discriminator
   - Prediction: Real or Fake
   - Ground Truth: Real Image

**Reinforcement Learning**

7. **Networks for Actions, Values, Policies, and Models**
   - Input: World State Sample
   - Network: Any Encoder
   - Representation
   - Action
   - Reward
   - Ground Truth: Reward
Deep Reinforcement Learning (Deep RL)

- Based on what the NN is tasked with learning: **policy-based, value-based, and model-based**;
Selected NNs in Detail

Supervised Learning

1. Feed Forward Neural Networks
   - Input: A few numbers
   - Network: Dense Encoder
   - Output: Representation, Prediction
   - Ground Truth: Prediction

2. Convolutional Neural Networks
   - Input: An image
   - Network: Convolutional Encoder
   - Output: Representation, Prediction
   - Ground Truth: Prediction

3. Recurrent Neural Networks
   - Input: Sequence
   - Network: Recurrent Encoder
   - Output: Representation, Prediction
   - Ground Truth: Prediction

4. Encoder-Decoder Architectures
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Output: Image, Text, etc.
   - Ground Truth: Image, Text, etc.

Unsupervised Learning

5. Autoencoder
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Output: Any Decoder
   - Ground Truth: Exact copy of input

6. Generative Adversarial Networks
   - Input: Noise
   - Network: Generator
   - Output: Discriminator
   - Ground Truth: Prediction: Real or Fake

Reinforcement Learning

7. Networks for Actions, Values, Policies, and Models
   - Input: World State Sample
   - Network: Any Encoder
   - Output: Action
   - Ground Truth: Reward
Pure Perception is HARD

CNN is a revolutionary tool in the Computer Vision field.
Image Understanding is HARD
CNN

Images are Numbers

Convolutional filters: take advantage of spatial invariance;
CNN
why CNN works

“Exploring Neural Networks with Activation Atlases”, distill, 2019

https://distill.pub/2019/activation-atlas/
Why CNN over FFNN?

- Spatial variant v.s. Spatial invariant;
- Scale ill v.s. Scale well;
Classification: CNNs beat Human

- **AlexNet (2012):** First CNN (15.4%)
  - 8 layers
  - 61 million parameters

- **ZFNet (2013):** 15.4% to 11.2%
  - 8 layers

- **VGGNet (2014):** 11.2% to 7.3%
  - Beautifully uniform:
    - 3x3 conv, stride 1, pad 1, 2x2 max pool
  - 16 layers
  - 138 million parameters

- **GoogleNet (2014):** 11.2% to 6.7%
  - Inception modules
  - 22 layers
  - 5 million parameters
  (throw away fully connected layers)

- **ResNet (2015):** 6.7% to 3.57%
  - More layers = better performance
  - 152 layers

- **CUImage (2016):** 3.57% to 2.99%
  - Ensemble of 6 models

- **SENet (2017):** 2.99% to 2.25%
  - Squeeze and excitation block: network is allowed to adaptively adjust the weighting of each feature map in the convolutional block.

Human error (5.1%) surpassed in 2015
ResNet (residual network)

- Is deeper the better? **Vanishing Gradient!**

AlexNet: 8 layers  
VGGNet: 16 layers  
GoogLeNet: 22 layers  
ResNet: 152 layers!

The Residual Block

\[ F(x) + x \]

Activation Function

Derivative

\[ F(x) \]

weight layer

relu

x

identity

weight layer

relu

x
ResNet

Figure 11: A ResNet can be reformulated into a recurrent form that is almost identical to a conventional RNN.

"Bridging the gaps between residual learning, recurrent neural networks and visual cortex." (2016)

- RNNs without the explicit time based construction;
Selected NNs in Detail

1. Feed Forward Neural Networks
   - Input: A few numbers
   - Network: Dense Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

2. Convolutional Neural Networks
   - Input: An image
   - Network: Convolutional Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

3. Recurrent Neural Networks
   - Input: Sequence
   - Network: Recurrent Encoder
   - Representation
   - Prediction
   - Ground Truth: Prediction

4. Encoder-Decoder Architectures
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Image, Text, etc.
   - Ground Truth: Image, Text, etc.

5. Autoencoder
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Ground Truth: Exact copy of input

6. Generative Adversarial Networks
   - Input: Noise
   - Network: Generator
   - Fake Image
   - Discriminator
   - Prediction: Real or Fake
   - Ground Truth: Real Image

7. Networks for Actions, Values, Policies, and Models
   - Input: World State Sample
   - Network: Any Encoder
   - Representation
   - Any Decoder
   - Action
   - Ground Truth: Reward

Reinforcement Learning
Vanilla NN

RNNs are amazing. But tricky to train.
Recurrent Neural Networks have loops.
RNN (LSTM)

Input: (example: word of a sentence)
Hidden state: function of previous hidden state and new input
Output: (example: predict next word in the sentence)
“Bob is eating an apple.”

“Bob likes apples. He is hungry and decided to have a snack. So now he is eating an apple.”

In theory, RNNs could learn this long-term dependencies. In practice, it is difficult.
The repeating module in a standard RNN contains a single layer.

The repeating module in a standard LSTM contains four interacting layers.

RNN v.s. Long short-term memory (LSTM)
Bob and Alice are having lunch. Bob likes apples. Alice likes oranges. She is eating an orange.
LSTM Conveyor Belt

- State run through the cell
- 3 sigmoid layers output deciding which information is let through (~1) and which is not (~0)
LSTM Conveyer Belt

Step 1: Decide what to forget / ignore

\[ f_t = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f) \]
Step 2: Decide which state values to update (w/sigmoid) and what values to update with (w/ tanh)
Step 3: Perform the forgetting and the state update

\[ C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \]
**LSTM Converyer Belt**

**Step 4:** Produce output with tanh [-1, 1] deciding the values and sigmoid [0, 1] deciding the filtering.

\[
o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)
\]

\[
h_t = o_t \times \tanh(C_t)
\]
Applications

Machine Translation

Handwriting Generation from Text

Input:

Text --- up to 100 characters, lower case letters work best
Deep Learning for Self Driving Cars

Output:

Deep Learning
for Self-Driving Cars

Applications

Image Caption Generation

Video Description Generation

Correct descriptions.

S2VT: A man is doing stunts on his bike.

S2VT: A herd of zebras are walking in a field.

Relevant but incorrect descriptions.

S2VT: A small bus is running into a building.

S2VT: A man is cutting a piece of a pair of a paper.

Venugopalan et al. “Sequence to sequence-video to text.” 2015.

Code: https://vsabhachh.github.io/c2vt.html
Selected NNs in Detail

1. Feed Forward Neural Networks
   - Input: A few numbers
   - Network: Dense Encoder
   - Output: Representation, Prediction
   - Ground Truth: Prediction

2. Convolutional Neural Networks
   - Input: An image
   - Network: Convolutional Encoder
   - Output: Representation, Prediction
   - Ground Truth: Prediction

3. Recurrent Neural Networks
   - Input: Sequence
   - Network: Recurrent Encoder
   - Output: Representation, Prediction
   - Ground Truth: Prediction

4. Encoder-Decoder Architectures
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Output: Representation
   - Network: Any Decoder
   - Output: Image, Text, etc.
   - Ground Truth: ----- (No specific ground truth mentioned)

5. Autoencoder
   - Input: Image, Text, etc.
   - Network: Any Encoder
   - Output: Representation
   - Network: Any Decoder
   - Ground Truth: Exact copy of input

6. Generative Adversarial Networks
   - Input: Noise
   - Network: Generator
   - Output: Fake Image
   - Network: Discriminator
   - Ground Truth: Prediction: Real or Fake
   - Throw away after training

7. Networks for Actions, Values, Policies, and Models
   - Input: World State Sample
   - Network: Any Encoder
   - Output: Representation
   - Network: Any Decoder
   - Output: Action
   - Ground Truth: Reward

Reinforcement Learning
Seq2Seq: Encoder & Decoder

Neural Machine Translation
SEQUENCE TO SEQUENCE MODEL

Je suis étudiant → ENCODER → DECODER

CONTEXT

0.11 0.03 0.81 -0.62

0.11 0.03 0.81 -0.62
LSTM: Unrolled Process

Neural Machine Translation
SEQUENCE TO SEQUENCE MODEL

Encoding Stage
Encoder RNN

Decoding Stage
Decoder RNN

Je suis étudiant
Attention is all you need

“Bob likes apples. He is hungry and decided to have a snack. So now he is eating an apple.”

LSTMs does not solve the problem of RNNs completely: when sentences are long, the model often forgets the content of distant positions in the sequence.
LSTM v.s. Attention
Attention

Je suis étudiant
The Transformer – a model that uses attention to boost the speed with which these models can be trained.
The Transformer – a model that uses **attention** to boost the speed with which these models can be trained.

Speed, accuracy, parallelization.
Transformer: more encoder & decoder
Transformer: more network
Transformer: Self-Attention

When encoding "it" in encoder #5 (the top encoder in the stack):

part of the attention mechanism was focusing on "The Animal", and baked a part of its representation into the encoding of "it".

More details: http://jalammar.github.io/illustrated-transformer/
Deep Double Descent

“An effect occurs in CNNs, ResNets, and transformers: performance first improves, then gets worse, and then improves again with increasing model size, data size, or training time.”
Challenges “conventional wisdoms.”

- Bias-variance trade-off: “larger models are worse.”
- Modern NN: “larger models are better.”
- “early stopping” is sometimes good.
**Effective model complexity (EMC)**
the maximum number of samples on which it can achieve close to zero training error.

**Under-parameterized regime**: $EMC(T) \ll n$:
any perturbation of $T$ that increases its effective complexity will decrease the test error.

**Over-parameterized regime** $EMC(T) \gg n$:
any perturbation of $T$ that increases its effective complexity will decrease the test error.

**Critically parameterized regime** $EMC(T) \approx n$:
a perturbation of $T$ that increases its effective complexity might decrease or increase the test error.
Deep Double Descent: A Stable Phenomenon

- Model-wise Double Descent
- Epoch-wise Double Descent
- Sample-wise Non-monotonicity

(a) ResNet18 on CIFAR10. (b) ResNet18 on CIFAR100. (c) 5-layer CNN on CIFAR 10.
- Model behaves unexpectedly in transition regime
- Training longer reverses overfitting
  - Double the training epoch is a technique in some task
- Bigger models are worse
- More data hurts

“While this behaviour appears to be fairly universal, we don’t yet fully understand why it happens, and view further study of this phenomenon as an important research direction.”
Reference

- [https://deeplearning.mit.edu](https://deeplearning.mit.edu)
- [http://jalammar.github.io](http://jalammar.github.io)

RNN $\rightarrow$ LSTM $\rightarrow$ **Attention** $\rightarrow$ Transformer

Attention at time step 4
RNN $\rightarrow$ LSTM $\rightarrow$ Attention $\rightarrow$ Transformer
RNN → LSTM → **Attention** → Transformer

**Neural Machine Translation**
SEQUENCE TO SEQUENCE MODEL WITH ATTENTION

**Encoding Stage**

**Attention Decoding Stage**

*Hidden State 1*
*Hidden State 2*
*Hidden State 3*

*Je suis étudiant*

*Attention Decoder RNN*
RNN $\rightarrow$ LSTM $\rightarrow$ Attention $\rightarrow$ **Transformer**

**Input**
- Embedding
- Queries
- Keys
- Values

**Score**
- $q_1 \cdot k_1 = 112$
- $q_1 \cdot k_2 = 96$
- Divide by $8 (\sqrt{d_k})$
- Softmax
- Softmax
- X
- Value
- Sum

**Thinking**
- x
- q
- k
- v

**Machines**
- x
- q
- k
- v

**Examples**
- $Q \times W^Q = Q$
- $X \times W^K = K$
- $X \times W^V = V$

$Q^T \times K \sqrt{d_k}$

Softmax($\frac{Q^T \times K \sqrt{d_k}}{d_k}$)

= Z

74