

# Going Deep

Kate Farrahi

ECS Southampton

February 21, 2019

# The Universal Approximation Theorem

Let  $\psi : \mathbb{R} \rightarrow \mathbb{R}$  be a nonconstant, bounded, and continuous function.

# The Universal Approximation Theorem

Let  $\psi : \mathbb{R} \rightarrow \mathbb{R}$  be a nonconstant, bounded, and continuous function. Let  $I_m$  denote the  $m$ -dimensional unit hypercube  $[0, 1]^m$ .

# The Universal Approximation Theorem

Let  $\psi : \mathbb{R} \rightarrow \mathbb{R}$  be a nonconstant, bounded, and continuous function. Let  $I_m$  denote the  $m$ -dimensional unit hypercube  $[0, 1]^m$ . The space of real-valued continuous functions on  $I_m$  is denoted by  $C(I_m)$ .

# The Universal Approximation Theorem

Let  $\psi : \mathbb{R} \rightarrow \mathbb{R}$  be a nonconstant, bounded, and continuous function. Let  $I_m$  denote the  $m$ -dimensional unit hypercube  $[0, 1]^m$ . The space of real-valued continuous functions on  $I_m$  is denoted by  $C(I_m)$ . Then, given any  $\epsilon > 0$  and any function  $f \in C(I_m)$ , there exist an integer  $N$ , real constants  $v_i, b_i \in \mathbb{R}$  and real vectors  $w_i \in \mathbb{R}^m$  for  $i = 1, \dots, N$ , such that we may define:

$F(x) = \sum_{i=1}^N v_i \phi(w_i^T x + b_i)$  as an approximate realization of the function  $f$  ; that is,

# The Universal Approximation Theorem

Let  $\psi : \mathbb{R} \rightarrow \mathbb{R}$  be a nonconstant, bounded, and continuous function. Let  $I_m$  denote the  $m$ -dimensional unit hypercube  $[0, 1]^m$ . The space of real-valued continuous functions on  $I_m$  is denoted by  $C(I_m)$ . Then, given any  $\epsilon > 0$  and any function  $f \in C(I_m)$ , there exist an integer  $N$ , real constants  $v_i, b_i \in \mathbb{R}$  and real vectors  $w_i \in \mathbb{R}^m$  for  $i = 1, \dots, N$ , such that we may define:

$F(x) = \sum_{i=1}^N v_i \phi(w_i^T x + b_i)$  as an approximate realization of the function  $f$  ; that is,

$$|F(x) - f(x)| < \epsilon$$

for all  $x \in I_m$ .

# Then Why Go Deep?

- ▶ There are functions you can compute with a deep neural network that shallow networks require exponentially more hidden units to compute.
- ▶ The following function is more efficient to implement using a deep neural network:  $y = x_1 \oplus x_2 \oplus x_3 \oplus \cdots \oplus x_n$

# Vanishing and Exploding Gradients

- ▶ The vanishing and exploding gradient problem is a difficulty found in training NN with gradient-based learning methods and backpropagation.



# Vanishing and Exploding Gradients

- ▶ The vanishing and exploding gradient problem is a difficulty found in training NN with gradient-based learning methods and backpropagation.
- ▶ In training, the gradient may become vanishingly small (or large), effectively preventing the weight from changing its value (or exploding in value).

# Vanishing and Exploding Gradients

- ▶ The vanishing and exploding gradient problem is a difficulty found in training NN with gradient-based learning methods and backpropagation.
- ▶ In training, the gradient may become vanishingly small (or large), effectively preventing the weight from changing its value (or exploding in value).
- ▶ This leads to the neural network not being able to train.

# Vanishing and Exploding Gradients

- ▶ The vanishing and exploding gradient problem is a difficulty found in training NN with gradient-based learning methods and backpropagation.
- ▶ In training, the gradient may become vanishingly small (or large), effectively preventing the weight from changing its value (or exploding in value).
- ▶ This leads to the neural network not being able to train.
- ▶ This issue affects many-layered networks (feed-forward), as well as recurrent networks.

# Residual Connections

- ▶ One of the most effective ways to resolve the vanishing gradient problem is with residual neural networks (ResNets)<sup>1</sup>.

---

<sup>1</sup>K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," CVPR, Las Vegas, NV, 2016, pp. 770-778.

# Residual Connections

- ▶ One of the most effective ways to resolve the vanishing gradient problem is with residual neural networks (ResNets)<sup>1</sup>.
- ▶ ResNets are artificial neural networks that use *skip connections* to jump over layers.

---

<sup>1</sup>K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," CVPR, Las Vegas, NV, 2016, pp. 770-778.

# Residual Connections

- ▶ One of the most effective ways to resolve the vanishing gradient problem is with residual neural networks (ResNets)<sup>1</sup>.
- ▶ ResNets are artificial neural networks that use *skip connections* to jump over layers.
- ▶ The vanishing gradient problem is mitigated in ResNets by reusing activations from a previous layer.

---

<sup>1</sup>K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," CVPR, Las Vegas, NV, 2016, pp. 770-778.

# Residual Connections

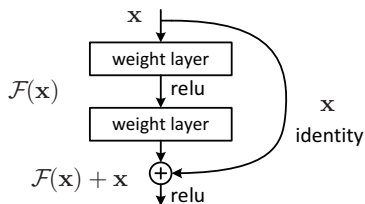


Figure 2. Residual learning: a building block.<sup>2</sup>

---

<sup>2</sup>K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," CVPR, Las Vegas, NV, 2016, pp. 770-778.

# Residual Connections

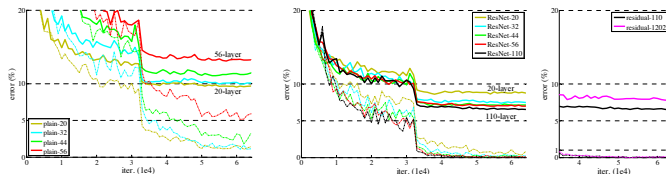


Figure 6. Training on **CIFAR-10**. Dashed lines denote training error, and bold lines denote testing error. **Left:** plain networks. The error of plain-110 is higher than 60% and not displayed. **Middle:** ResNets. **Right:** ResNets with 110 and 1202 layers.

<sup>3</sup>K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," CVPR, Las Vegas, NV, 2016, pp. 770-778.



# Regularization in Deep Networks

- ▶  $L1$  and  $L2$  norm
- ▶ Dropout

# Dropout

- ▶ Neural networks with a large number of parameters are powerful, however, overfitting is a serious problem in such systems.

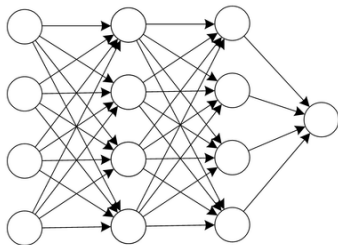
# Dropout

- ▶ Neural networks with a large number of parameters are powerful, however, overfitting is a serious problem in such systems.
- ▶ Dropout is a form of regularization

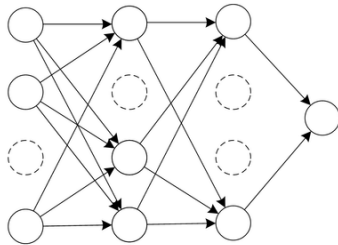
# Dropout

- ▶ Neural networks with a large number of parameters are powerful, however, overfitting is a serious problem in such systems.
- ▶ Dropout is a form of regularization
- ▶ The key idea in dropout is to randomly drop neurons, including all of the connections, from the neural network during training.

# Dropout



(a) Standard Neural Network



(b) Network after Dropout

4

---

<sup>4</sup>Image from: [https://www.researchgate.net/figure/Dropout-neural-network-model-a-is-a-standard-neural-network-b-is-the-same-model-with-dropout-fig3\\_309206911](https://www.researchgate.net/figure/Dropout-neural-network-model-a-is-a-standard-neural-network-b-is-the-same-model-with-dropout-fig3_309206911)

# Why Does Dropout Work?

- ▶ Neurons cannot co-adapt to other units (they cannot assume that all of the other units will be present)

# Why Does Dropout Work?

- ▶ Neurons cannot co-adapt to other units (they cannot assume that all of the other units will be present)
- ▶ By breaking co-adaptation, each unit will ultimately find more general features