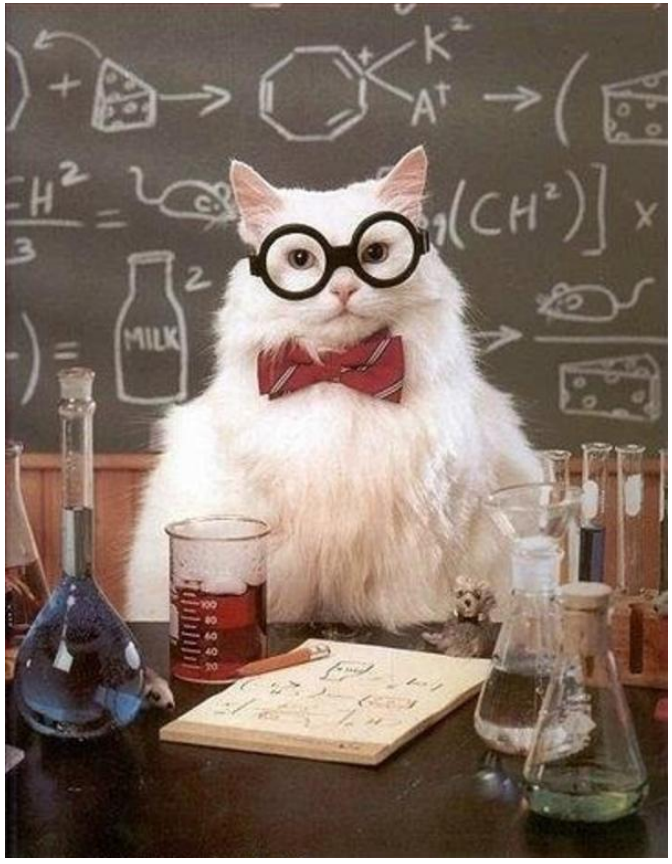
Introduction to Machine Learning feat. TensorFlow

Peter Goldsborough

July 11, 2016

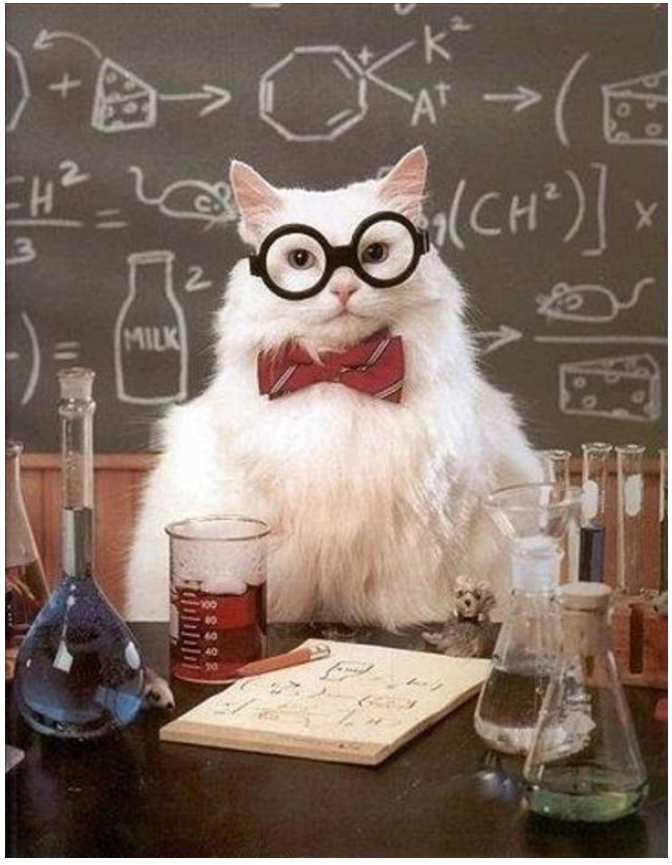
Table of Catents

Table of Catents

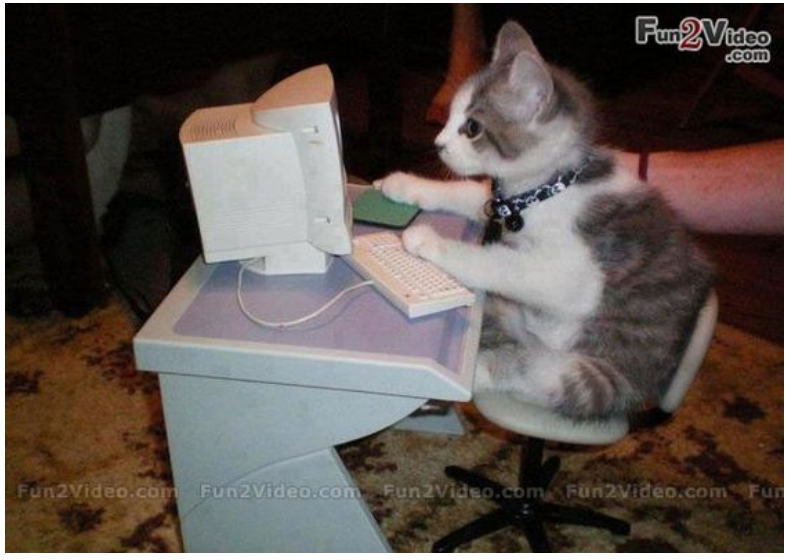


Theory

Table of Catents



Theory



Practice

Background

Background

- CS Student @ TUM

Background

- CS Student @ TUM
- Google & Bloomberg Intern

Background

- CS Student @ TUM
- Google & Bloomberg Intern
- I like cats

Background

- CS Student @ TUM
- Google & Bloomberg Intern
- I like cats

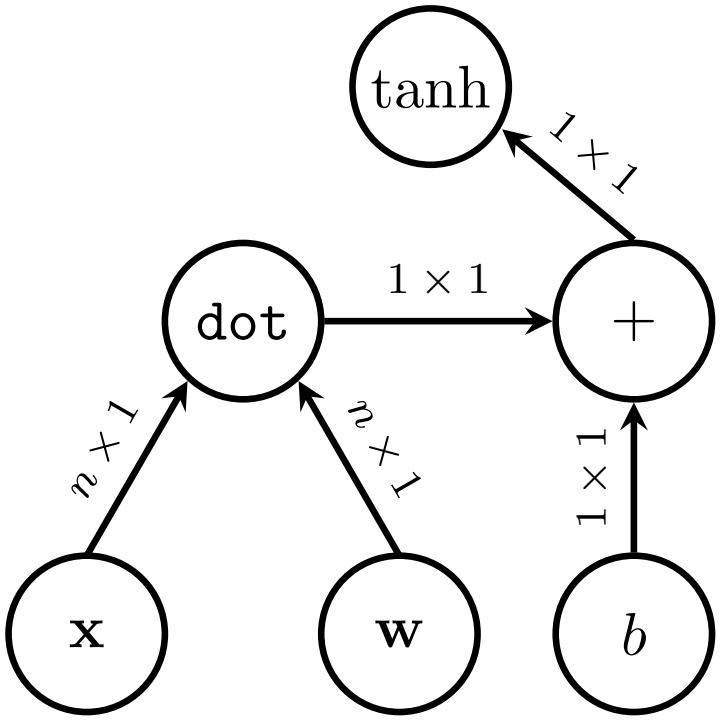
Seminar Topic: *Deep Learning With TensorFlow*

`github.com/peter-can-write/tensorflow-paper`

`github.com/peter-can-talk/python-meetup-munich-2016`

Computational Paradigms

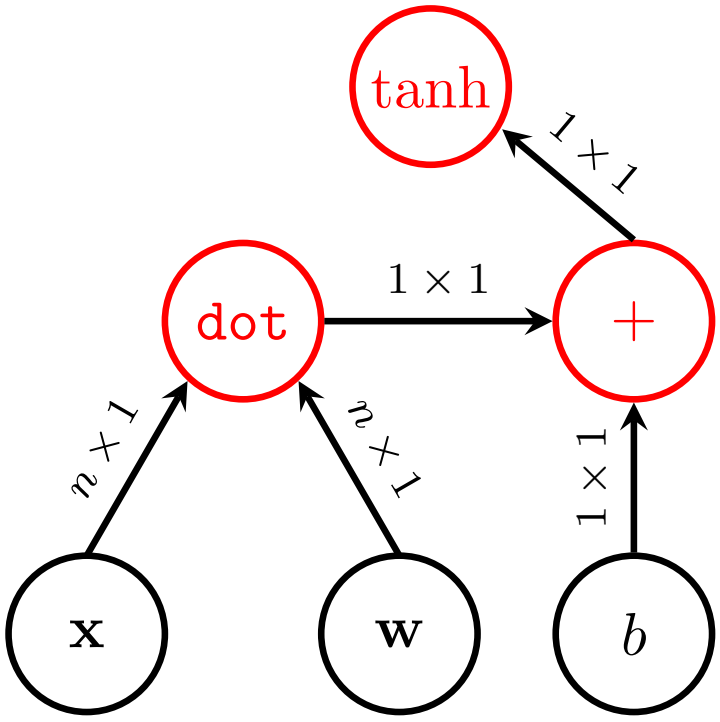
Computational Paradigms



$$\hat{y} = \tanh(\mathbf{x}^\top \mathbf{w} + b)$$

Computational Graphs

Computational Paradigms

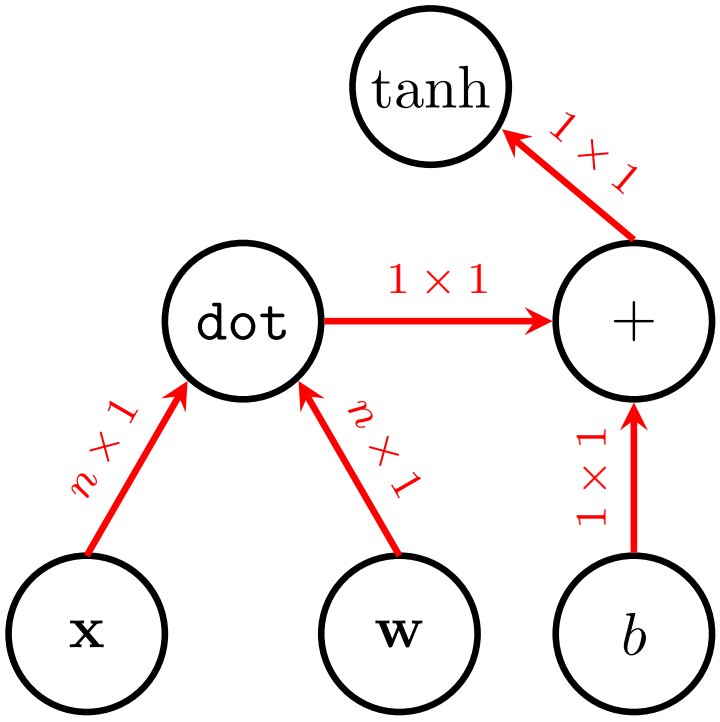


$$\hat{y} = \tanh(\mathbf{x}^\top \mathbf{w} + b)$$

Computational Graphs

1. Operations

Computational Paradigms

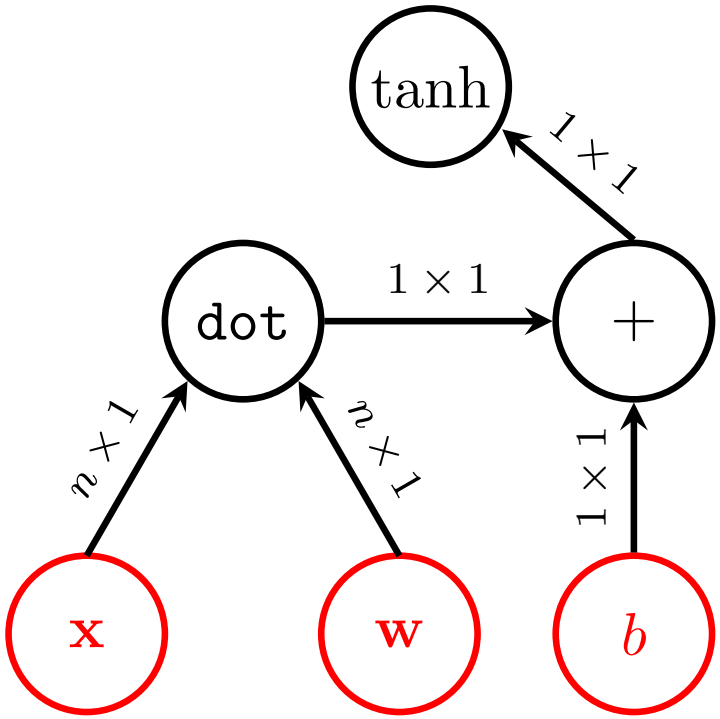


$$\hat{y} = \tanh(\mathbf{x}^\top \mathbf{w} + b)$$

Computational Graphs

1. Operations
2. Tensors

Computational Paradigms

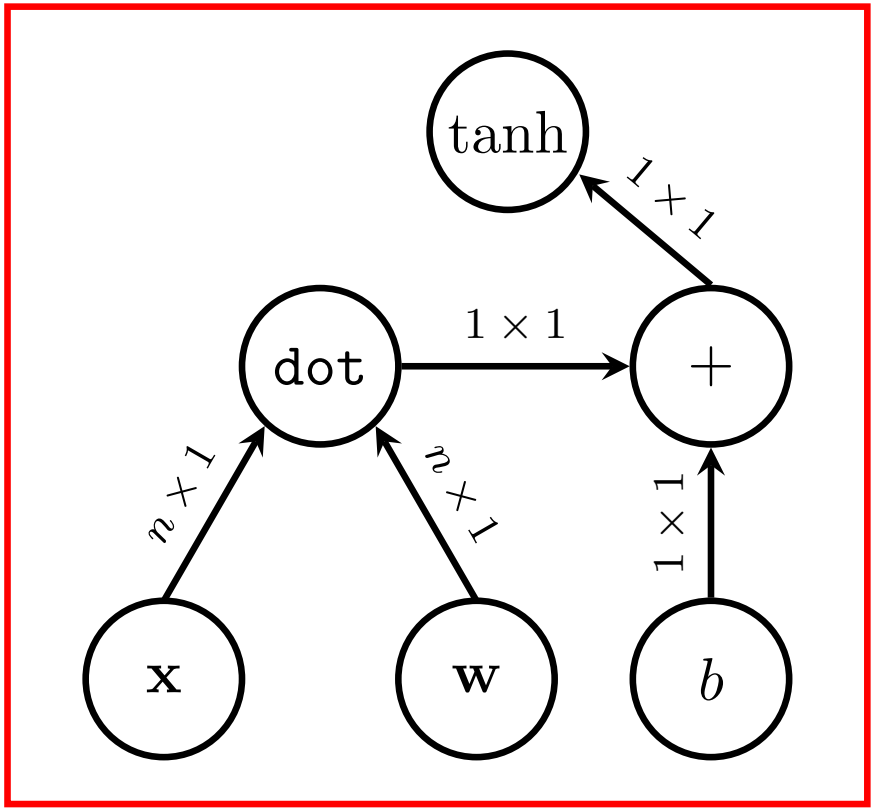


$$\hat{y} = \tanh(\mathbf{x}^\top \mathbf{w} + b)$$

Computational Graphs

1. Operations
2. Tensors
3. Variables

Computational Paradigms



$$\hat{y} = \texttt{session.run}(\tanh(\mathbf{x}^\top \mathbf{w} + b))$$

Computational Graphs

1. Operations
2. Tensors
3. Variables
4. Sessions

Execution Model

Execution Model

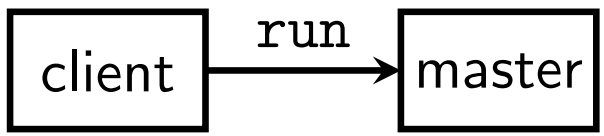
Actors

Execution Model

Actors

1. Client

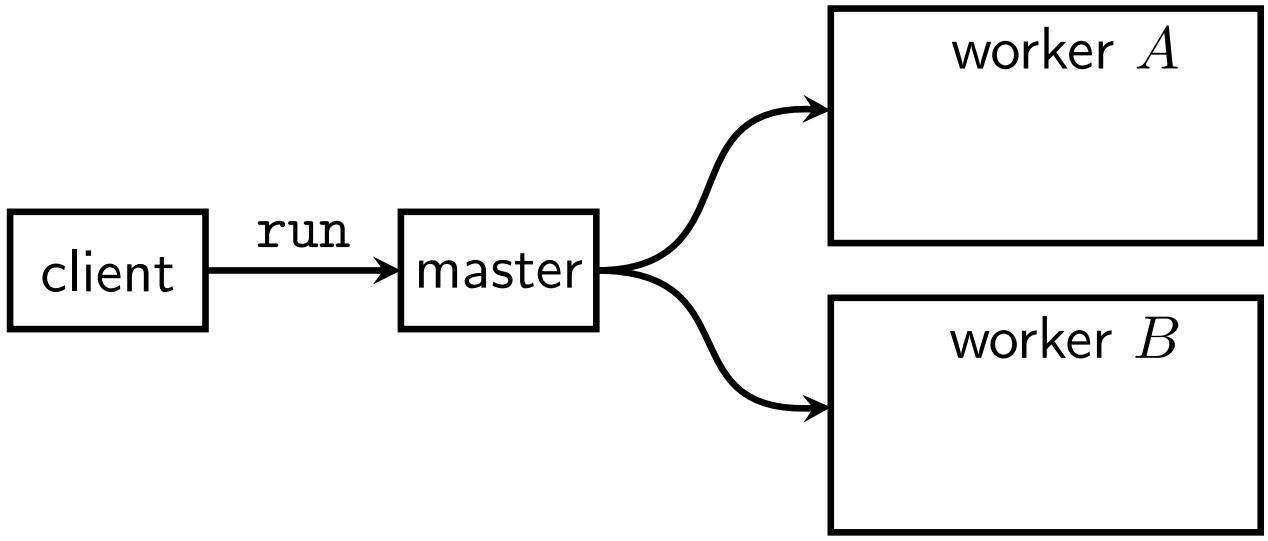
Execution Model



Actors

1. Client
2. Master

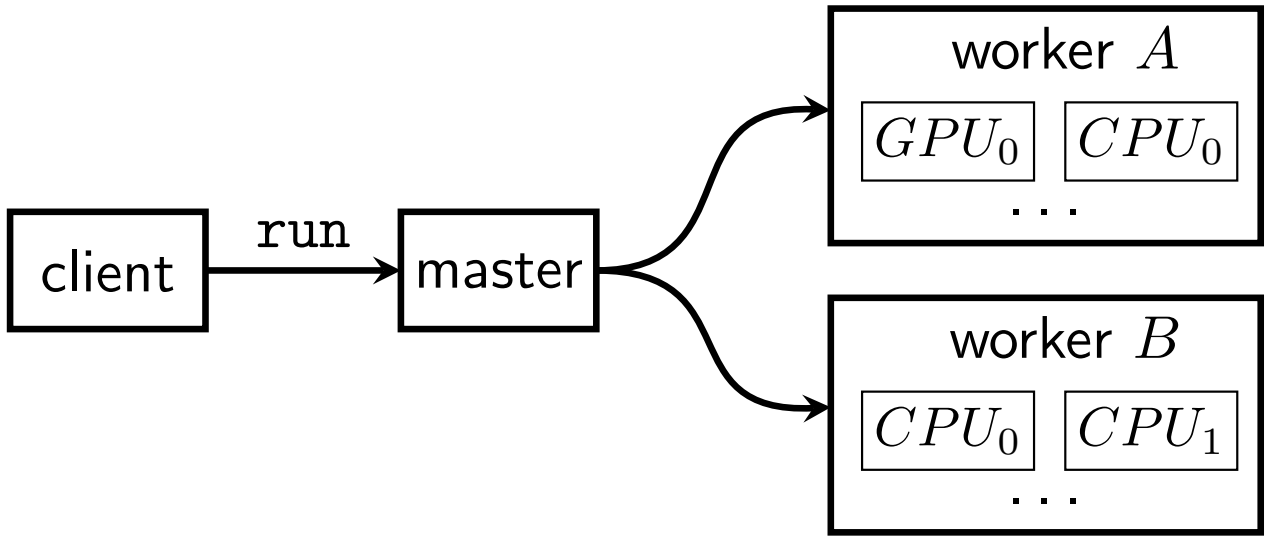
Execution Model



Actors

1. Client
2. Master
3. Workers

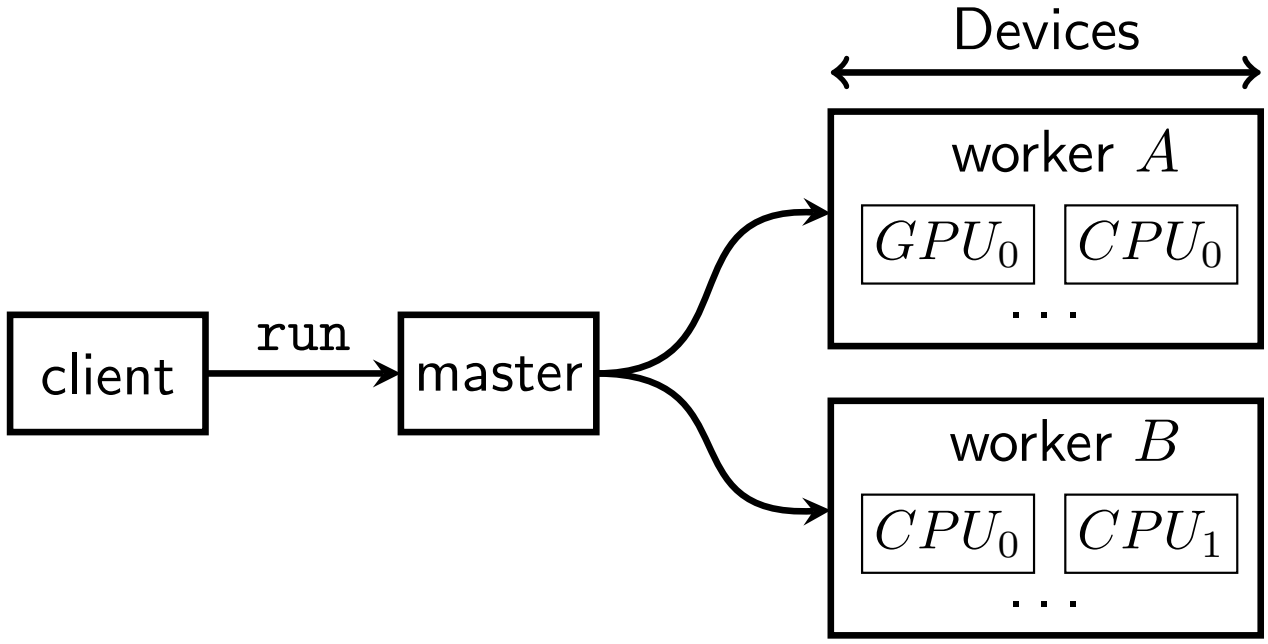
Execution Model



Actors

1. Client
2. Master
3. Workers
4. Devices

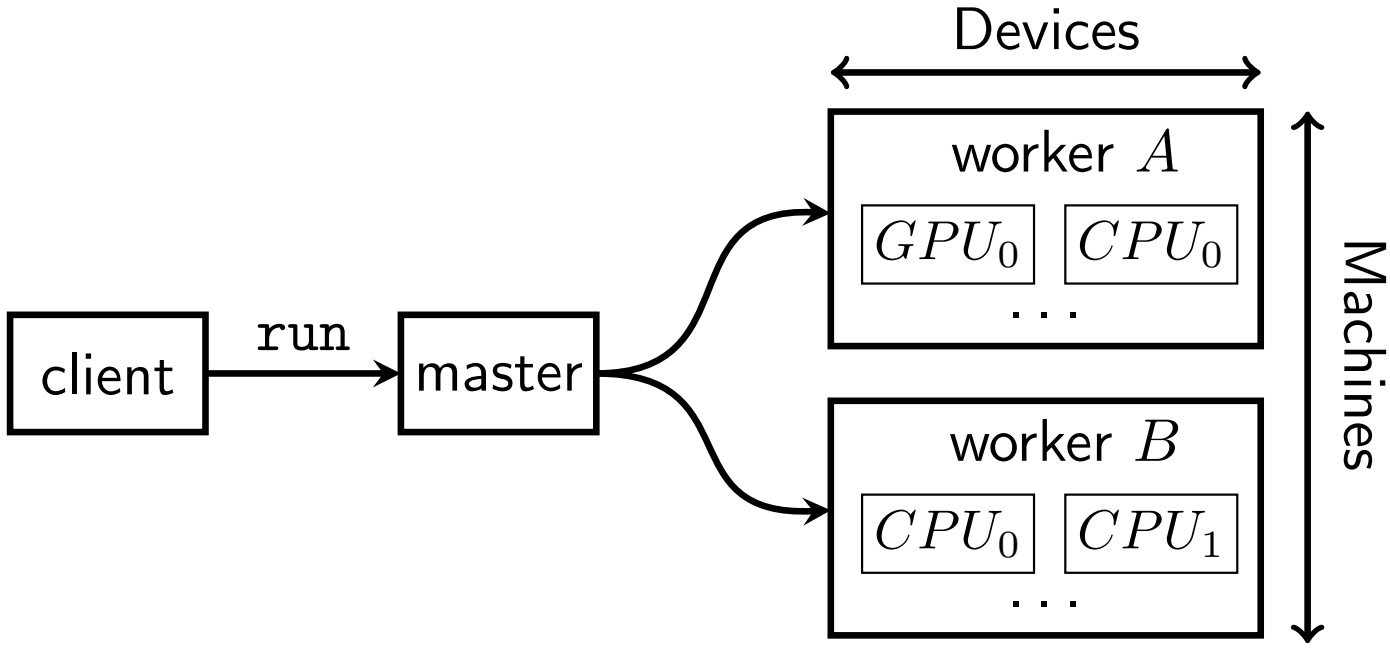
Execution Model



Actors

1. Client
2. Master
3. Workers
4. Devices

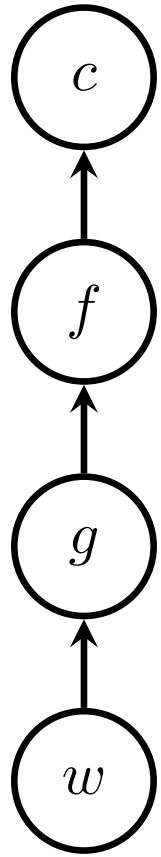
Execution Model



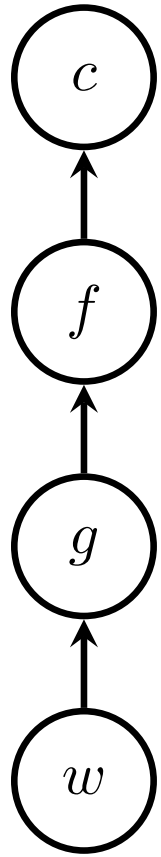
1. Client
2. Master
3. Workers
4. Devices

Back Propagation

Back Propagation



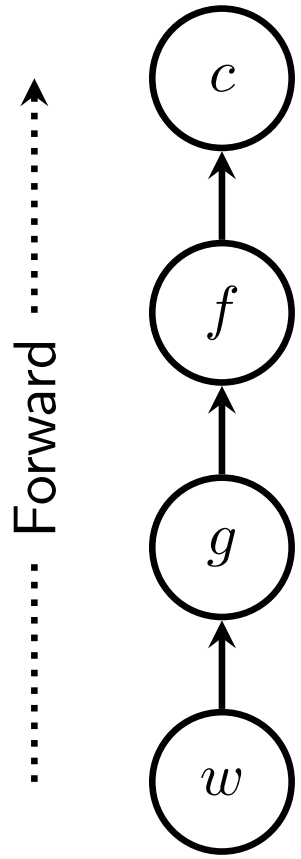
Back Propagation



Symbol to Number Differentiation

Back Propagation

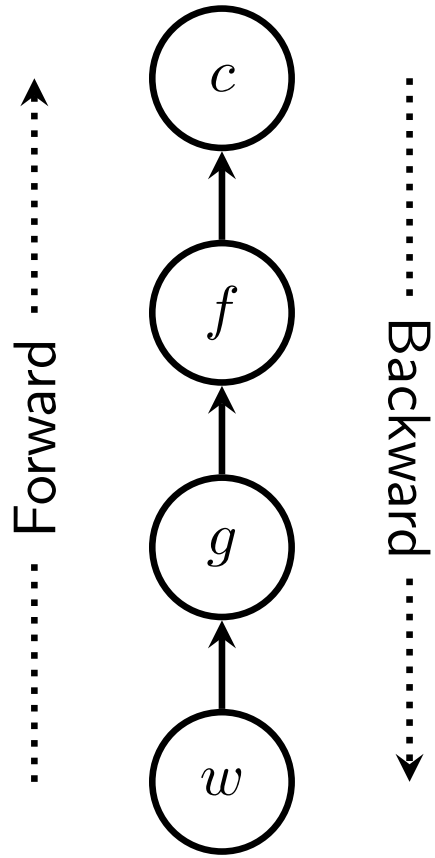
$$c = (f \circ g)(w)$$



Symbol to Number Differentiation

Back Propagation

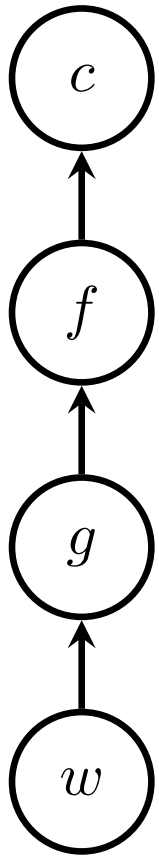
$$c = (f \circ g)(w)$$



$$\frac{\mathrm{d}c}{\mathrm{d}w} = f'(g(w)) \cdot g'(w)$$

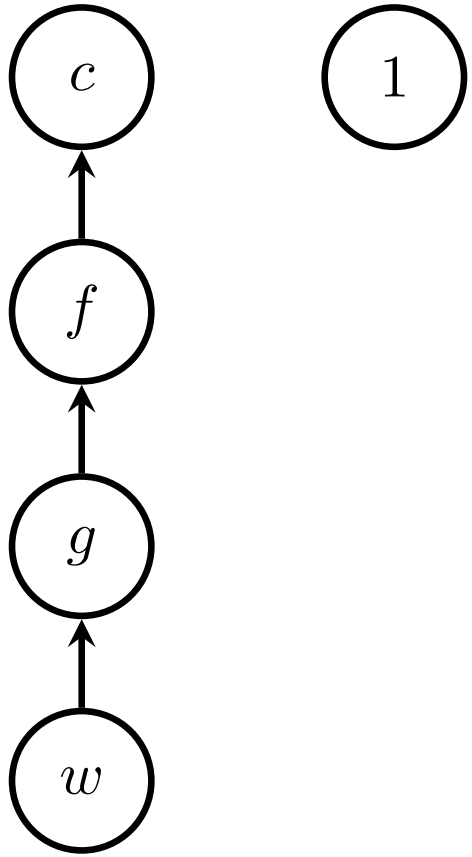
Symbol to Number Differentiation

Back Propagation



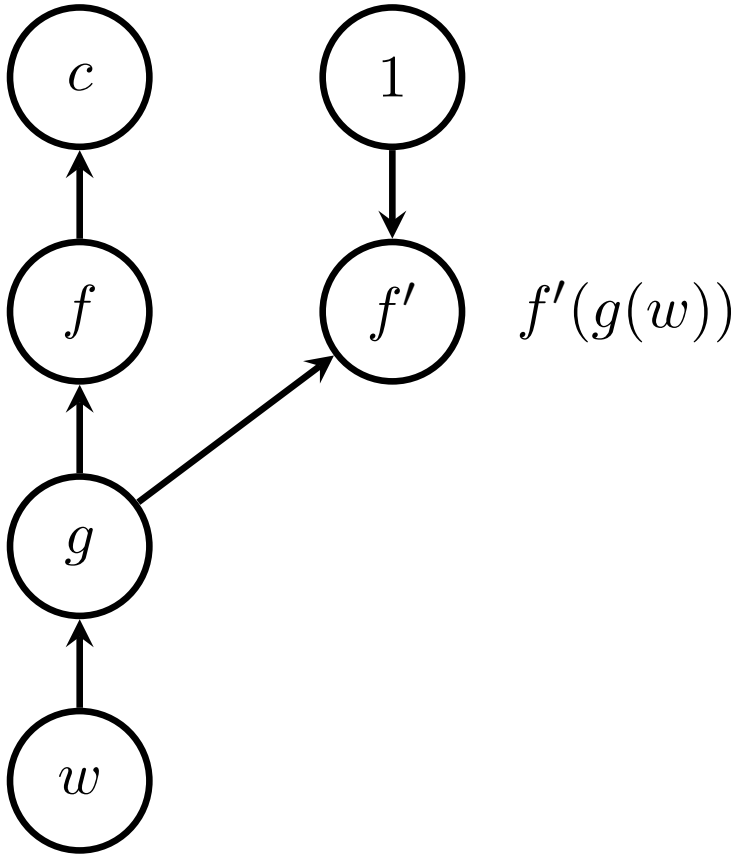
Symbol to Symbol Differentiation

Back Propagation



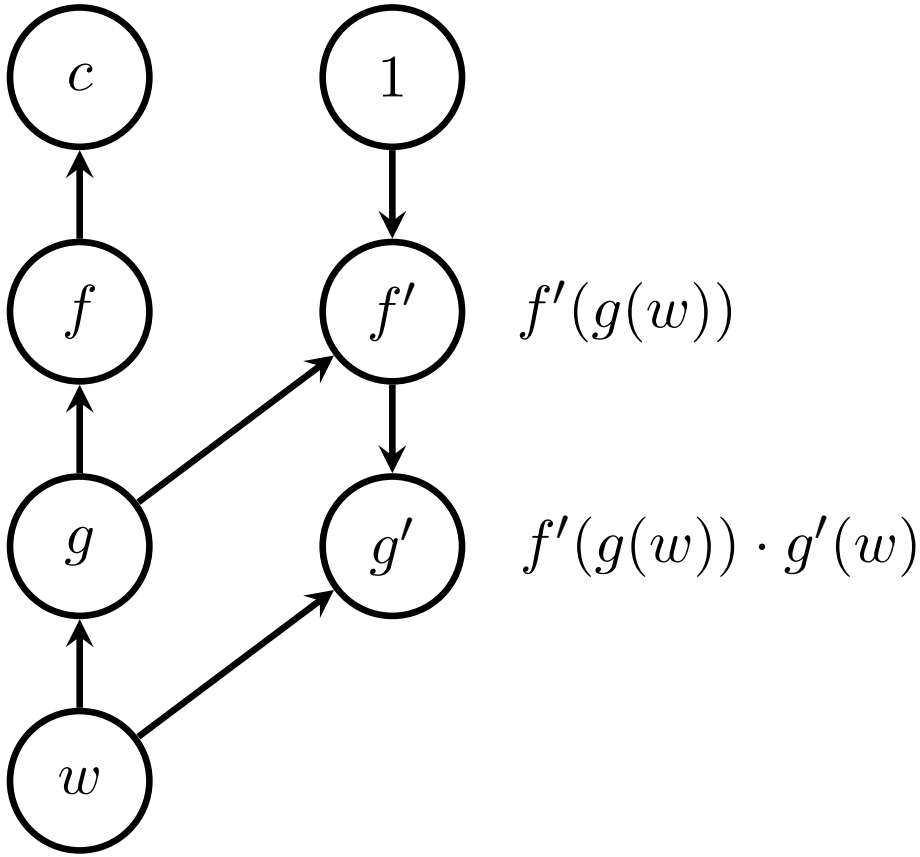
Symbol to Symbol Differentiation

Back Propagation



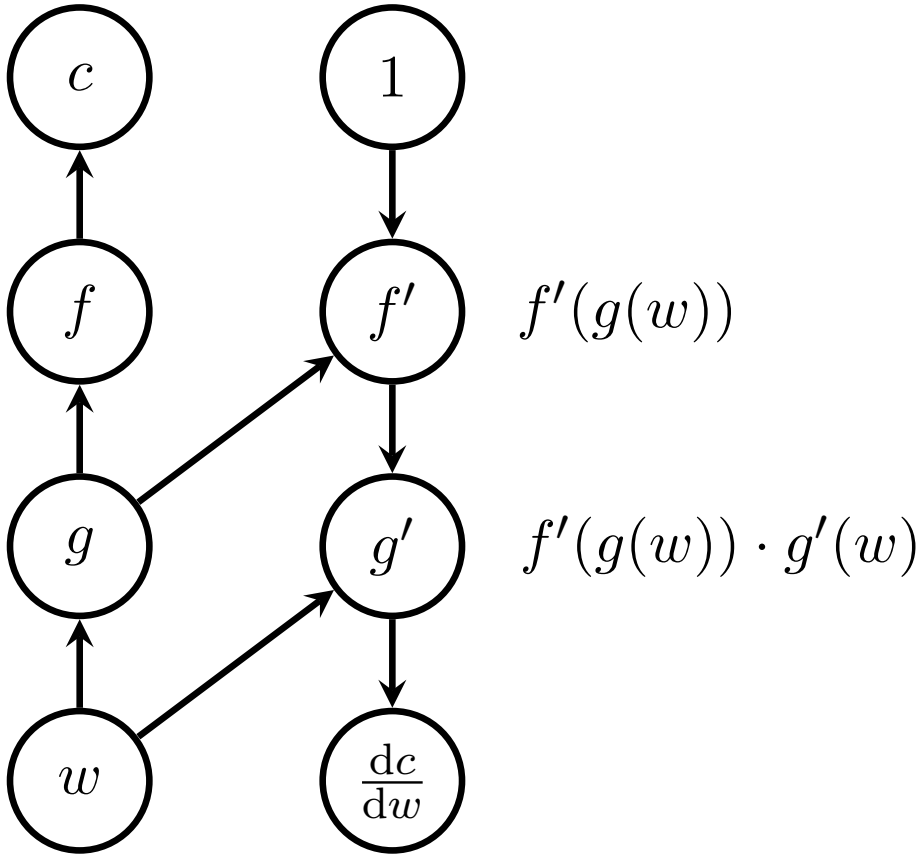
Symbol to Symbol Differentiation

Back Propagation



Symbol to Symbol Differentiation

Back Propagation



Symbol to Symbol Differentiation

Visualization Tools

Visualization Tools

- Deep Neural Networks have the tendency of being . . . deep

Visualization Tools

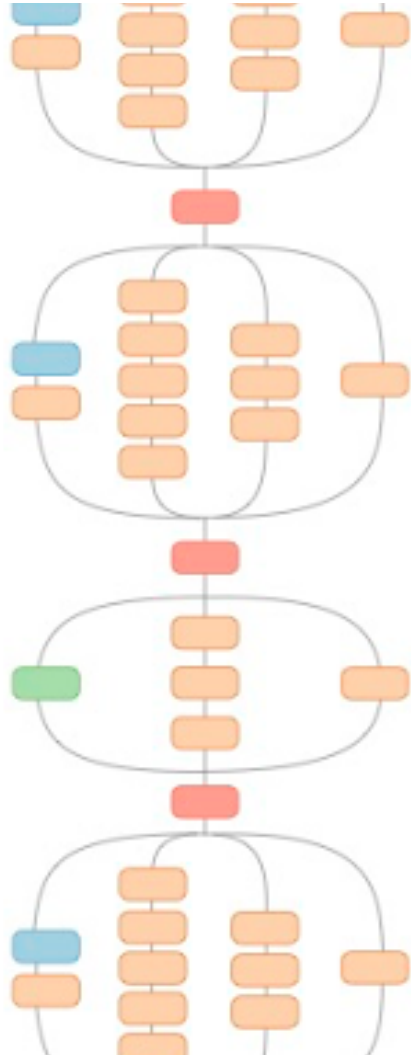
- Deep Neural Networks have the tendency of being . . . deep
- Easy to drown in the complexity of an architecture

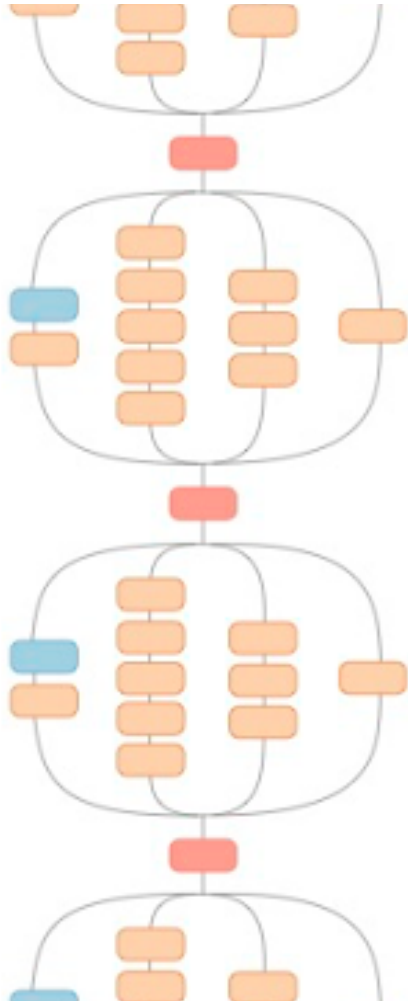
Visualization Tools

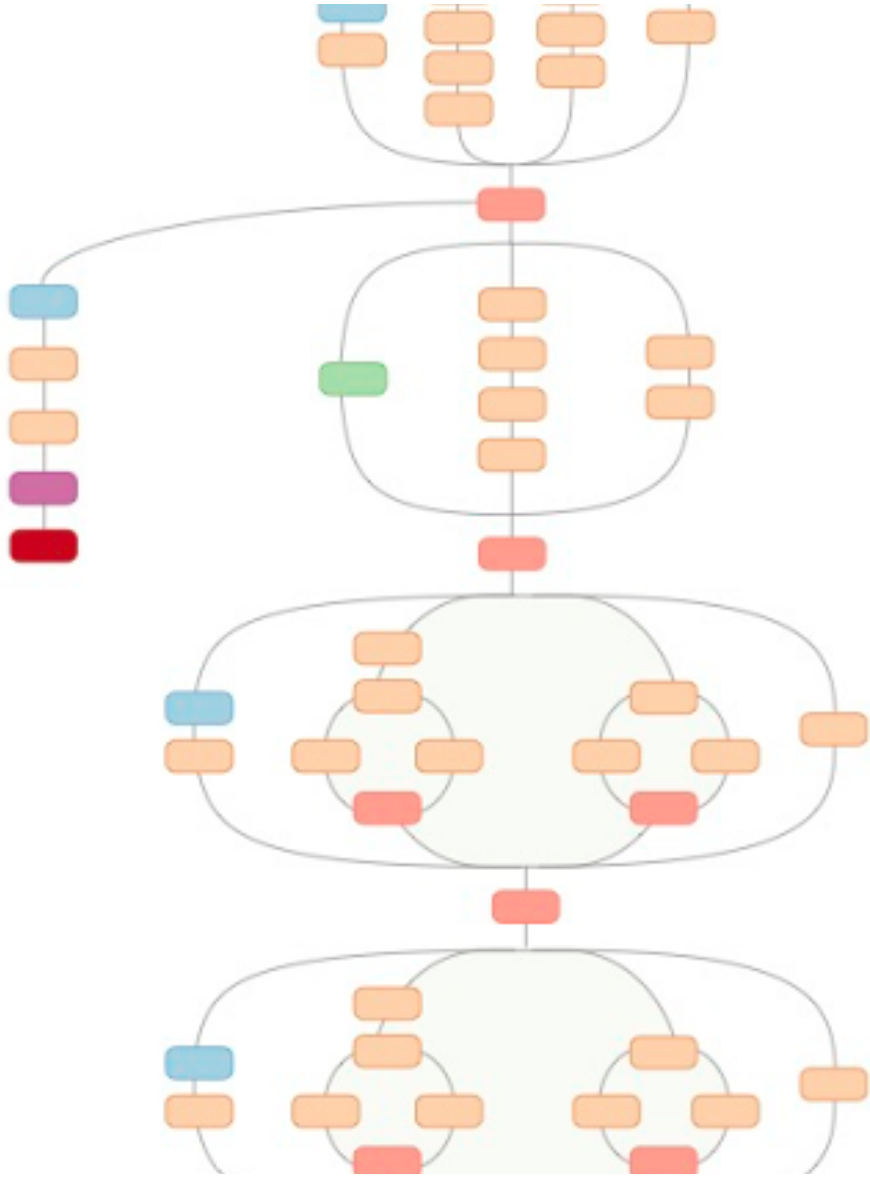
- Deep Neural Networks have the tendency of being . . . deep
- Easy to drown in the complexity of an architecture
- $>$ 36,000 nodes for Google's *Inception* model

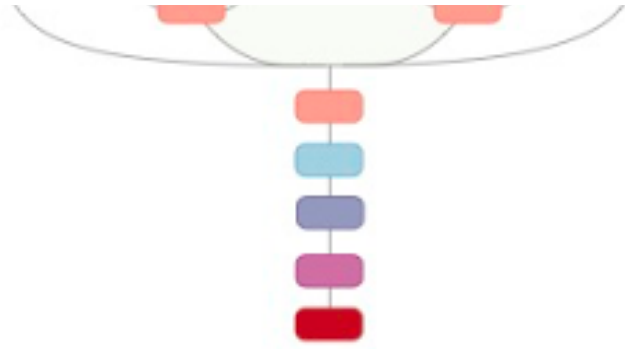
Visualization Tools

- Deep Neural Networks have the tendency of being . . . deep
- Easy to drown in the complexity of an architecture
- $>$ 36,000 nodes for Google's *Inception* model

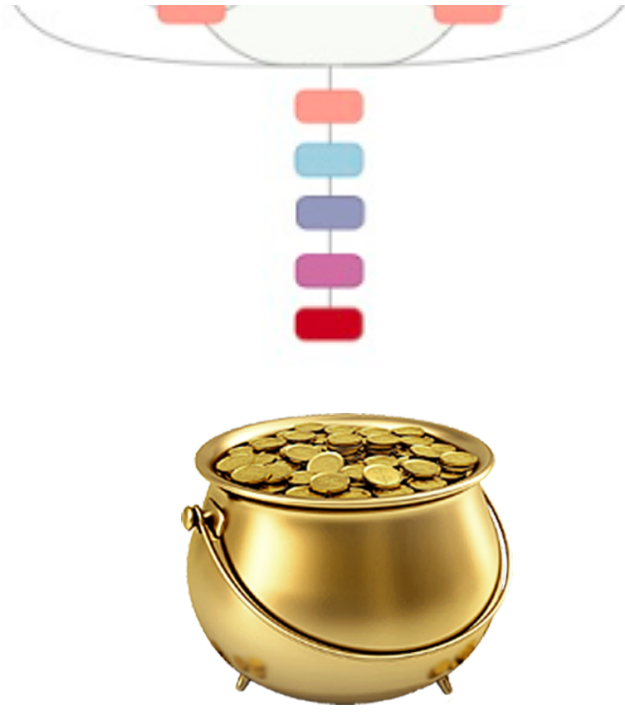








Source: http://googleresearch.blogspot.de/2016/03/train-your-own-image-classifier-with.html



Source: http://googleresearch.blogspot.de/2016/03/train-your-own-image-classifier-with.html

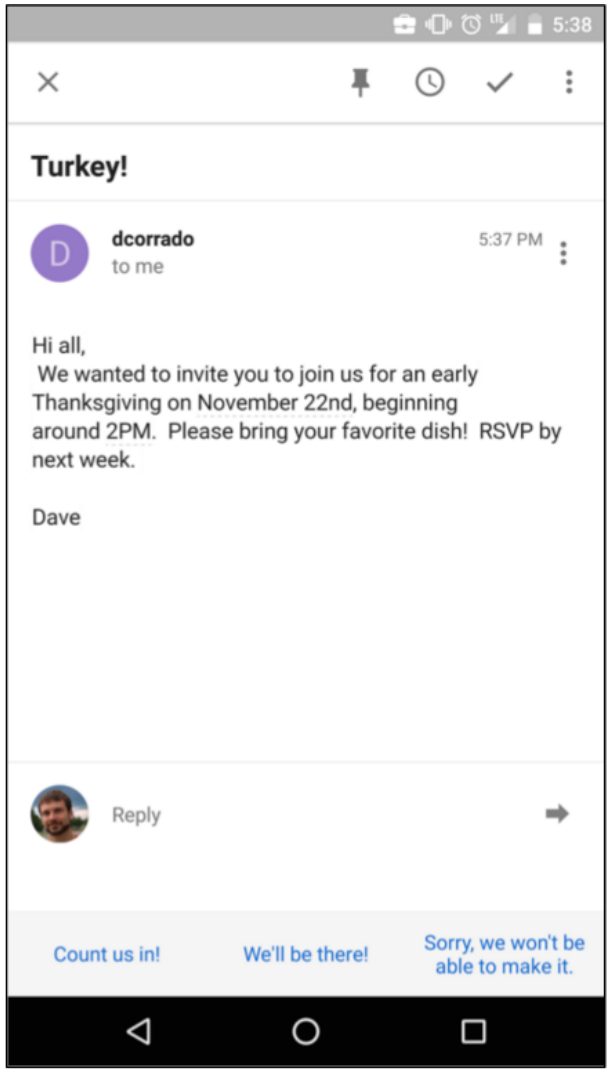
TensorBoard to the Rescue

Use Cases

Source: http://googleresearch.blogspot.de/2015/11/computer-respond-to-this-email.html

Use Cases

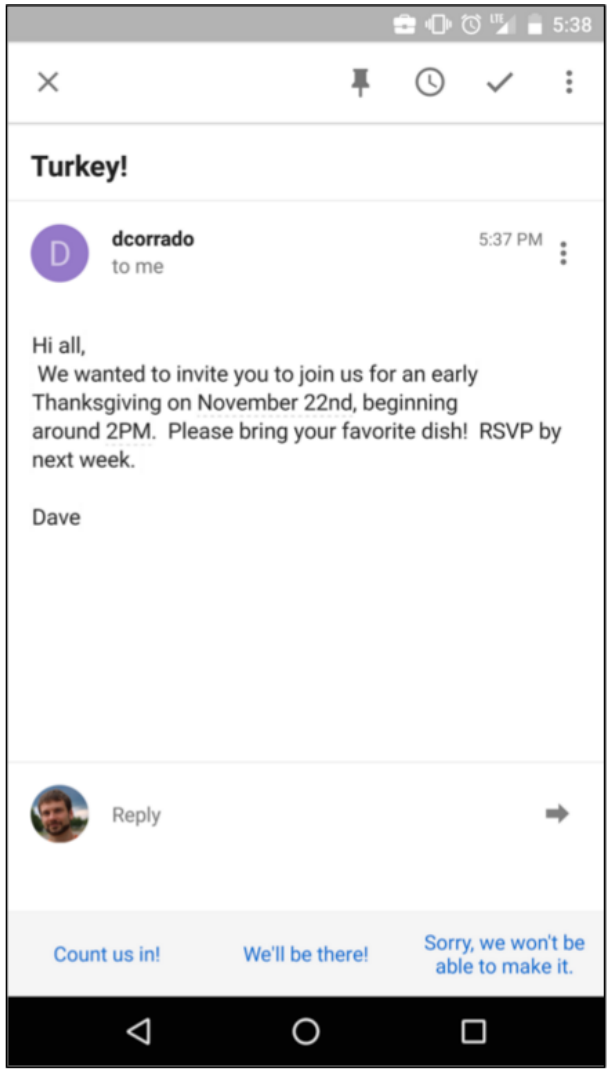
- Smart email replies in Google *Inbox*



Source: http://googleresearch.blogspot.de/2015/11/computer-respond-to-this-email.html

Use Cases

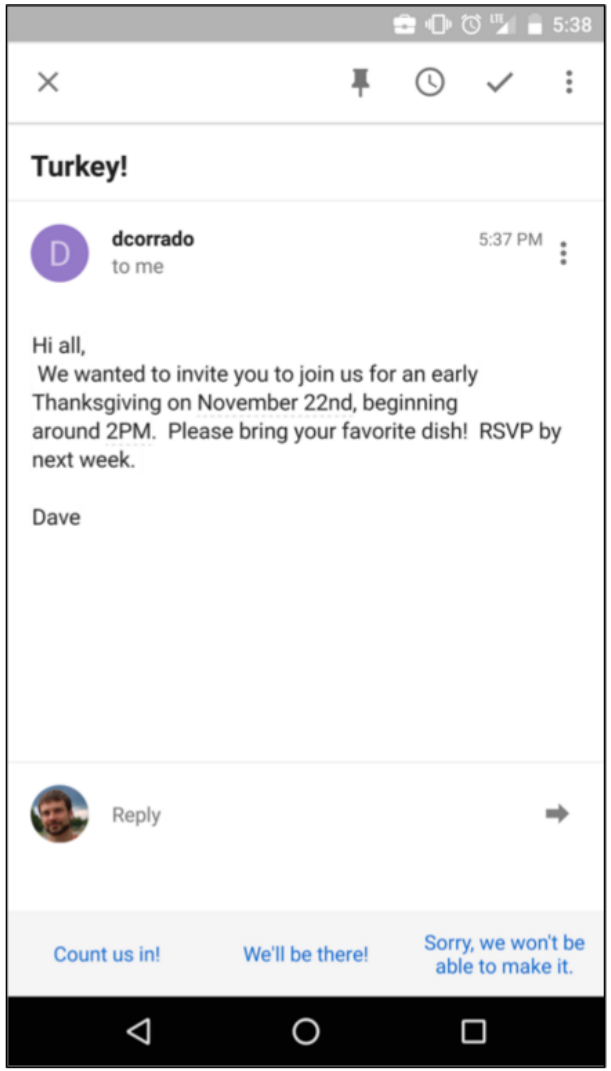
- Smart email replies in Google *Inbox*
- Emails mapped to "thought vectors"



Source: http://googleresearch.blogspot.de/2015/11/computer-respond-to-this-email.html

Use Cases

- Smart email replies in Google *Inbox*
- Emails mapped to "thought vectors"
- LSTMs synthesize valid replies



Source: http://googleresearch.blogspot.de/2015/11/computer-respond-to-this-email.html

Use Cases

Use Cases

- Google DeepMind now using TensorFlow

Use Cases

- Google DeepMind now using TensorFlow
- Already for *AlphaGo*

Source: https://deepmind.com/css/images/opengraph/alphago-logo.png

Use Cases

- Google DeepMind now using TensorFlow
- Already for *AlphaGo*
- According to a DeepMind SWE reasons are:

Source: https://deepmind.com/css/images/opengraph/alphago-logo.png

Use Cases

- Google DeepMind now using TensorFlow
- Already for *AlphaGo*
- According to a DeepMind SWE reasons are:
  - Python,

Source: https://deepmind.com/css/images/opengraph/alphago-logo.png

Use Cases

- Google DeepMind now using TensorFlow
- Already for *AlphaGo*
- According to a DeepMind SWE reasons are:
  - Python,
  - Integration with Google Cloud Platform,

Source: https://deepmind.com/css/images/opengraph/alphago-logo.png

Use Cases

- Google DeepMind now using TensorFlow
- Already for *AlphaGo*
- According to a DeepMind SWE reasons are:
  - Python,
  - Integration with Google Cloud Platform,
  - Support for TPUs,

Source: https://deepmind.com/css/images/opengraph/alphago-logo.png

Use Cases

- Google DeepMind now using TensorFlow
- Already for *AlphaGo*
- According to a DeepMind SWE reasons are:
  - Python,
  - Integration with Google Cloud Platform,
  - Support for TPUs,
  - Ability to run on many GPUs.

Source: https://deepmind.com/css/images/opengraph/alphago-logo.png

Walkthrough

References


Andrej Karpathy, *The unreasonable effectiveness of recurrent neural networks*, May 21 2015 (accessed Jul 10, 2016), http://karpathy.github.io/2015/05/21/rnn-effectiveness/.