Thoughts on Systems for Large Datasets:
Problems and Opportunities

Jeff Dean
Google Senior Fellow
jeff@google.com

Many of the systems mentioned in this talk represent joint work with many, many colleagues at Google
Areas I Wish New Grads Knew More About

• Ability to do back-of-the-envelope calculations and quickly evaluate many alternative designs

• Understanding the importance of locality at all levels (caches & memory systems, disk I/O, cross-machine, geographic regions, etc.)

• Low-level encoding and compression schemes and their tradeoffs

• More math and statistics knowledge
  – e.g. use of randomized, probabilistic algorithms in distributed systems
Overview

• A collection of problems I believe are difficult/interesting:
  – For some, significant work has been done/published
  – Others are less explored

• Not meant to be exhaustive catalog of problems/areas
  – I care (and Google cares) about many other problems, too!

• Roughly in two main areas:
  – issues that arise in building systems that store and manipulate large datasets
  – automatically extracting higher-level information from raw data

• Feedback and suggestions are welcome!
Programming Models

• Large datasets already require use of large numbers of cores and machines for analyses

• Moore’s law is now scaling # cores instead of MHz
  – parallelism likely to be even more important in the future

• Parallelism is key to getting good performance out of large-scale systems
Distributed Systems Abstractions

• High-level tools/languages/abstractions for building distributed systems
  – e.g. For batch processing, MapReduce handles parallelization, load balancing, fault tolerance, I/O scheduling automatically within a simple programming model

• Challenge: Are there unifying abstractions for other kinds of distributed systems problems?
  – e.g. systems for handling interactive requests & dealing with *intra*-operation parallelism
    • load balancing, fault-tolerance, service location & request distribution, ...
  – systems that seamlessly divide, expand, and contract processing subsystems?
Building Applications on top of Weakly Consistent Storage Systems

• Many applications need state replicated across a wide area
  – For reliability and availability

• Two main choices:
  – consistent operations (e.g. use Paxos)
    • often imposes additional latency for common case
  – inconsistent operations
    • better performance/availability, but apps harder to write and reason about in this model

• Many apps need to use a mix of both of these:
  – e.g. Gmail: marking a message as read is asynchronous, sending a message is a heavier-weight consistent operation
Building Applications on top of Weakly Consistent Storage Systems

• Challenge: General model of consistency choices, explained and codified
  – ideally would have one or more “knobs” controlling performance vs. consistency
  – “knob” would provide easy-to-understand tradeoffs

• Challenge: Easy-to-use abstractions for resolving conflicting updates to multiple versions of a piece of state
  – Useful for reconciling client state with servers after disconnected operation
  – Also useful for reconciling replicated state in different data centers after repairing a network partition
Design of Very Large-Scale Computer Systems

- Future scale: \( \sim 10^6 \) to \( 10^7 \) machines, spread at 100s to 1000s of locations around the world, \( \sim 10^9 \) client machines

- zones of semi-autonomous control
- consistency after disconnected operation
- power adaptivity
Adaptivity and Self-Tuning in World-Wide Systems

- Challenge: automatic, dynamic world-wide placement of data & computation to minimize latency and/or cost, given constraints on:
  - bandwidth
  - packet loss
  - power
  - resource usage
  - failure modes
  - ...

- Users specify high-level desires:
  "99%ile latency for accessing this data should be <50ms"
  "Store this data on at least 2 disks in EU, 2 in U.S. & 1 in Asia"
ACLs in Information Retrieval Systems

• Retrieval systems with mix of private, semi-private, widely shared and public documents
  – e.g. e-mail vs. shared doc among 10 people vs. messages in group with 100,000 members vs. public web pages

• Challenge: building retrieval systems that efficiently deal with ACLs that vary widely in size
  – best solution for doc shared with 10 people is different than for doc shared with the world
  – sharing patterns of a document might change over time
Automatic Construction of Efficient IR Systems

• Currently use several retrieval systems
  – e.g. one system for sub-second update latencies, one for very large # of documents but daily updates, ...
  – common interfaces, but very different implementations primarily for efficiency
  – works well, but lots of effort to build, maintain and extend different systems

• Challenge: can we have a single parameterizable system that automatically constructs efficient retrieval system based on these parameters?
Information Extraction from Semi-structured Data

• Data with clearly labelled semantic meaning is a tiny fraction of all the data in the world
• But there’s lots semi-structured data
  – books & web pages with tables, data behind forms, ...

• Challenge: algorithms/techniques for improved extraction of structured information from unstructured/semi-structured sources
  – noisy data, but lots of redundancy
  – want to be able to correlate/combine/aggregate info from different sources
Learning from Raw Data

• Large datasets of very raw data
  – images, videos, user activity logs, genetics, other sciences, ...

• Want to answer high-level questions:
  – “what is a user in this situation likely to do?”
  – “which users are likely to buy items for more than $1000”
  – “give me a textual summary of this video”
  – “what are the most likely genetic markers of this disease, given genetic data and medical records of millions of people?”
  – “find a picture of three scarlet macaws in a tree”

• Need systems that automatically build high level representations and abstractions from the raw data

• Want to generalize from one task to others
Broadly Applicable

- We have been building systems that apply these techniques in the following domains:
  - image recognition, object detection, video processing
  - speech recognition
  - language modeling
  - user activity prediction
  - neuroscience
  - ad system optimization
  - language understanding
  - ...

Thursday, January 23, 14
Plenty of Data

- **Text**: trillions of words of English + other languages
- **Visual**: billions of images and videos
- **Audio**: spoken queries, audio portion of video data, ...
- **User activity**: queries, result page clicks, map requests, etc.
- **Knowledge graph**: billions of labelled relation triples
- **Biology and Health**: genetic data, health care records, ...
- **Physical sciences**: physics, astronomy, ...
- ...
Image Models

- stone wall [0.95, web]
- dishwasher [0.91, web]
- car show [0.99, web]
- judo [0.96, web]
- judo [0.92, web]
- judo [0.91, web]
- tractor [0.91, web]
- tractor [0.91, web]
- tractor [0.94, web]
What are these numbers?
What are all these words?
My dear Father,

Arrived at Tiverton yesterday I found your letter awaiting my arrival. I then made straight for my customer so that I could spend an hour or so at St. Peter Church which I did up to within 10 minutes of service being held when I had to clear out, but at any rate I have traced the register back as far as 1933. If you leaf over you will find a copy of as many Baptismal acts as I happened to come across I have only noticed
How about these words?
Goal: Unified System

Visual task 1 + Visual task 2 + Visual task N + Unsupervised training

Common visual representation

Image data
What is being said?
Goal: Unified System

- visual tasks
  - visual representation
  - Image data

- auditory tasks
  - auditory representation
  - Audio data

- textual tasks
  - textual representation
  - Textual data
Goal: Unified System

Common representation

visual representation
- visual tasks
  - Image data

auditory representation
- auditory tasks
  - Audio data

... (omitted)

textual representation
- textual tasks
  - Textual data
Goal: Unified System

various cross-modal tasks

Common representation

visual tasks

visual representation

Image data

auditory tasks

auditory representation

Audio data

textual tasks

textual representation

Textual data
One Key Approach: Deep Learning

- Algorithmic approach
  - automatically learn high-level representations from raw data
  - can learn from both labeled and unlabeled data

- Recent academic deep learning results improve on state-of-the-art in many areas (Hinton, Ng, Bengio, LeCun, et al.):
  - images, video, speech, NLP, ...
  - ... using modest model sizes ($\leq \sim 50M$ parameters)

- We want to scale this to much bigger models & datasets
  - general approach: parallelize at many levels
Input data

Layer 1

Layer N

Representation

...
Partition model across machines

Layer N
...
Layer 1
Layer 0
Minimal network traffic: The most densely connected areas are on the same partition.
Partition model across machines

Minimal network traffic: The most densely connected areas are on the same partition

One replica of our biggest model: 144 machines, ~2300 cores
Initial Focus on Unsupervised Learning

- Always: unlabeled data >> labeled data
- Experiment: unsupervised training on 10M random YouTube frames
- Trained 9 layer model with local connections

Visualization of optimal stimuli for two different neurons in top layer:
Initial Focus on Unsupervised Learning

- Always: unlabeled data >> labeled data
- Experiment: unsupervised training on 10M random YouTube frames
- Trained 9 layer model with local connections
  Visualization of optimal stimuli for two different neurons in top layer:
Initial Focus on Unsupervised Learning

- Always: unlabeled data >> labeled data
- Experiment: unsupervised training on 10M random YouTube frames
- Trained 9 layer model with local connections

Visualization of optimal stimuli for two different neurons in top layer:
Initial Focus on Unsupervised Learning

- Always: unlabeled data >> labeled data
- Experiment: unsupervised training on 10M random YouTube frames
- Trained 9 layer model with local connections
  Visualization of optimal stimuli for two different neurons in top layer:

Le, Ranzato, Monga, Devin, Chen, Corrado, Dean, & Ng. *Building High-Level Features Using Large Scale Unsupervised Learning*, ICML 2012.
We made a cat detector!

It uses a few CPUs!
Acoustic Modeling for Speech Recognition

Trained in <5 days on cluster of 800 machines

Close collaboration with Google Speech team
Acoustic Modeling for Speech Recognition

Trained in <5 days on cluster of 800 machines

30% reduction in Word Error Rate for English
(“biggest single improvement in 20 years of speech research”)

Close collaboration with Google Speech team
Acoustic Modeling for Speech Recognition

Trained in <5 days on cluster of 800 machines

30% reduction in Word Error Rate for English
(“biggest single improvement in 20 years of speech research”)

Launched at time of Jellybean release of Android

Close collaboration with Google Speech team
Convolutional Models for Object Recognition

Softmax to predict object class

Fully-connected layers

Convolutional layers
(same weights used at all spatial locations in layer)

Basic architecture developed by Krizhevsky, Sutskever & Hinton
(all now at Google)

Convolutional nets developed by Yann LeCun (of NYU)
Wow.

The new Google plus photo search is a bit insane.

I didn't tag those... :)

[Images of statues and sea views from Google+ search results]
Google Plus photo search is awesome. Searched with keyword 'Drawing' to find all my scribbles at once :D
CIANO MOTOR ENGINEERS
MECHANICAL REPAIRS TO ALL MAKES AND MODELS
SPECIALIZING IN BMW, MINI & TOYOTA
8 REGATA ROAD FIVE DOCK 9745 3173

- Latest diagnostic equipment
- REGO inspections
- New & second-hand servicing
- Brakes & clutches
- Steering suspension
- Tyres
- Wheel alignments
- Transmissions
- Air conditioning
- EMI towers
- Fuel injection servicing
- Batteries
- Auto electrical

Factory trained technicians
Recent results from ICDAR 2013 Competition for Task 2.3: “Reading Text in Scene Images”

**TABLE VIII. RANKING OF SUBMITTED METHODS TO TASK 2.3**

<table>
<thead>
<tr>
<th>Method</th>
<th>Total Edit Distance</th>
<th>Correctly Recognised Words (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PhotoOCR</td>
<td>122.7</td>
<td>82.83</td>
</tr>
<tr>
<td>PicRead [27]</td>
<td>332.4</td>
<td>57.99</td>
</tr>
<tr>
<td>NESP [19]</td>
<td>360.1</td>
<td>64.20</td>
</tr>
<tr>
<td>PLT [18]</td>
<td>392.1</td>
<td>62.37</td>
</tr>
<tr>
<td>MAPS [17]</td>
<td>421.8</td>
<td>62.74</td>
</tr>
<tr>
<td>Feild’s Method</td>
<td>422.1</td>
<td>47.95</td>
</tr>
<tr>
<td>PIONEER [28], [29]</td>
<td>479.8</td>
<td>53.70</td>
</tr>
<tr>
<td>Baseline</td>
<td>539.0</td>
<td>45.30</td>
</tr>
<tr>
<td>TextSpotter [20], [21], [22]</td>
<td>606.3</td>
<td>26.85</td>
</tr>
</tbody>
</table>

Recent results from ICDAR 2013 Competition for Task 2.3: “Reading Text in Scene Images”

### Table VIII. Ranking of Submitted Methods to Task 2.3

<table>
<thead>
<tr>
<th>Method</th>
<th>Total Edit Distance</th>
<th>Correctly Recognised Words (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PhotoOCR</td>
<td>122.7</td>
<td>82.83</td>
</tr>
<tr>
<td>PicRead [27]</td>
<td>332.4</td>
<td>57.99</td>
</tr>
<tr>
<td>NESP [19]</td>
<td>360.1</td>
<td>64.20</td>
</tr>
<tr>
<td>PLT [18]</td>
<td>392.1</td>
<td>62.37</td>
</tr>
<tr>
<td>MAPS [17]</td>
<td>421.8</td>
<td>62.74</td>
</tr>
<tr>
<td>Feild’s Method</td>
<td>422.1</td>
<td>47.95</td>
</tr>
<tr>
<td>PIONEER [28], [29]</td>
<td>479.8</td>
<td>53.70</td>
</tr>
<tr>
<td>Baseline</td>
<td>539.0</td>
<td>45.30</td>
</tr>
<tr>
<td>TextSpotter [20], [21], [22]</td>
<td>606.3</td>
<td>26.85</td>
</tr>
</tbody>
</table>

How about text-related tasks?
Embeddings

\ (~1000-D joint embedding space

\[\text{porpoise} \quad \text{dolphin}\]
Embeddings

≈1000-D joint embedding space

porpoise  dolphin
Embeddings

~1000-D joint embedding space

porpoise

dolphin

SeaWorld
Embeddings

~1000-D joint embedding space

Obama

porpoise

dolphin

SeaWorld
Embeddings

~1000-D joint embedding space
Skip-Gram Model

Predictions

the cat the cheese

ate

E
Embedding sparse tokens in an N-dimensional space

Example: 50-D embedding trained for semantic similarity

apple

<table>
<thead>
<tr>
<th>Id</th>
<th>Distance</th>
<th>Adjust</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>11114</td>
<td>0.000000</td>
<td>Remove</td>
<td>apple</td>
</tr>
<tr>
<td>5026</td>
<td>0.652580</td>
<td>Add</td>
<td>fruit</td>
</tr>
<tr>
<td>14080</td>
<td>0.699192</td>
<td>Add</td>
<td>apples</td>
</tr>
<tr>
<td>48657</td>
<td>0.717818</td>
<td>Add</td>
<td>melon</td>
</tr>
<tr>
<td>28498</td>
<td>0.722390</td>
<td>Add</td>
<td>peach</td>
</tr>
<tr>
<td>39795</td>
<td>0.729893</td>
<td>Add</td>
<td>blueberry</td>
</tr>
<tr>
<td>35570</td>
<td>0.730500</td>
<td>Add</td>
<td>berry</td>
</tr>
<tr>
<td>25974</td>
<td>0.739561</td>
<td>Add</td>
<td>strawberry</td>
</tr>
<tr>
<td>46156</td>
<td>0.745343</td>
<td>Add</td>
<td>pecan</td>
</tr>
<tr>
<td>11907</td>
<td>0.756422</td>
<td>Add</td>
<td>potato</td>
</tr>
<tr>
<td>33847</td>
<td>0.759111</td>
<td>Add</td>
<td>pear</td>
</tr>
<tr>
<td>30895</td>
<td>0.763317</td>
<td>Add</td>
<td>mango</td>
</tr>
<tr>
<td>17848</td>
<td>0.768230</td>
<td>Add</td>
<td>pumpkin</td>
</tr>
<tr>
<td>39133</td>
<td>0.770143</td>
<td>Add</td>
<td>almond</td>
</tr>
<tr>
<td>14395</td>
<td>0.773105</td>
<td>Add</td>
<td>tomato</td>
</tr>
<tr>
<td>18163</td>
<td>0.782610</td>
<td>Add</td>
<td>onion</td>
</tr>
<tr>
<td>10470</td>
<td>0.782994</td>
<td>Add</td>
<td>pie</td>
</tr>
<tr>
<td>3023</td>
<td>0.787229</td>
<td>Add</td>
<td>tree</td>
</tr>
<tr>
<td>20340</td>
<td>0.793602</td>
<td>Add</td>
<td>bean</td>
</tr>
<tr>
<td>34968</td>
<td>0.794979</td>
<td>Add</td>
<td>watermelon</td>
</tr>
</tbody>
</table>
Embedding sparse tokens in an N-dimensional space

Example: 50-D embedding trained for semantic similarity

apple

stab
Embedding sparse tokens in an N-dimensional space

Example: 50-D embedding trained for semantic similarity

<table>
<thead>
<tr>
<th>Cluster 1:</th>
<th>apple</th>
<th>Cluster 1:</th>
<th>stab</th>
<th>Cluster 1:</th>
<th>iPhone</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>Distance↑</th>
<th>Adjust</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>11114</td>
<td>0.000000</td>
<td>Remove</td>
<td>apple</td>
</tr>
<tr>
<td>5026</td>
<td>0.652580</td>
<td>Add</td>
<td>fruit</td>
</tr>
<tr>
<td>14080</td>
<td>0.699192</td>
<td>Add</td>
<td>apples</td>
</tr>
<tr>
<td>48657</td>
<td>0.717818</td>
<td>Add</td>
<td>melon</td>
</tr>
<tr>
<td>28498</td>
<td>0.722390</td>
<td>Add</td>
<td>peach</td>
</tr>
<tr>
<td>39795</td>
<td>0.729893</td>
<td>Add</td>
<td>blueberry</td>
</tr>
<tr>
<td>35570</td>
<td>0.730500</td>
<td>Add</td>
<td>berry</td>
</tr>
<tr>
<td>25974</td>
<td>0.739561</td>
<td>Add</td>
<td>strawberry</td>
</tr>
<tr>
<td>46156</td>
<td>0.745343</td>
<td>Add</td>
<td>pecan</td>
</tr>
<tr>
<td>11907</td>
<td>0.756422</td>
<td>Add</td>
<td>potato</td>
</tr>
<tr>
<td>33847</td>
<td>0.759111</td>
<td>Add</td>
<td>pear</td>
</tr>
<tr>
<td>30895</td>
<td>0.763317</td>
<td>Add</td>
<td>mango</td>
</tr>
<tr>
<td>17848</td>
<td>0.768230</td>
<td>Add</td>
<td>pumpkin</td>
</tr>
<tr>
<td>39133</td>
<td>0.770143</td>
<td>Add</td>
<td>almond</td>
</tr>
<tr>
<td>14395</td>
<td>0.773105</td>
<td>Add</td>
<td>tomato</td>
</tr>
<tr>
<td>18163</td>
<td>0.782610</td>
<td>Add</td>
<td>onion</td>
</tr>
<tr>
<td>10470</td>
<td>0.782994</td>
<td>Add</td>
<td>pie</td>
</tr>
<tr>
<td>3023</td>
<td>0.787229</td>
<td>Add</td>
<td>tree</td>
</tr>
<tr>
<td>20340</td>
<td>0.793602</td>
<td>Add</td>
<td>bean</td>
</tr>
<tr>
<td>34968</td>
<td>0.794979</td>
<td>Add</td>
<td>watermelon</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>Distance↑</th>
<th>Adjust</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>14979</td>
<td>0.000000</td>
<td>Remove</td>
<td>stab</td>
</tr>
<tr>
<td>7728</td>
<td>0.868853</td>
<td>Add</td>
<td>punch</td>
</tr>
<tr>
<td>469</td>
<td>0.909304</td>
<td>Add</td>
<td>shot</td>
</tr>
<tr>
<td>12820</td>
<td>0.909750</td>
<td>Add</td>
<td>thrust</td>
</tr>
<tr>
<td>8934</td>
<td>0.939908</td>
<td>Add</td>
<td>shell</td>
</tr>
<tr>
<td>10880</td>
<td>0.951466</td>
<td>Add</td>
<td>hammer</td>
</tr>
<tr>
<td>6975</td>
<td>0.951679</td>
<td>Add</td>
<td>bullet</td>
</tr>
<tr>
<td>1848</td>
<td>0.962053</td>
<td>Add</td>
<td>push</td>
</tr>
<tr>
<td>10888</td>
<td>0.962319</td>
<td>Add</td>
<td>eyed</td>
</tr>
<tr>
<td>718</td>
<td>0.965448</td>
<td>Add</td>
<td>hand</td>
</tr>
<tr>
<td>5865</td>
<td>0.966663</td>
<td>Add</td>
<td>grab</td>
</tr>
<tr>
<td>4611</td>
<td>0.967574</td>
<td>Add</td>
<td>swing</td>
</tr>
<tr>
<td>302</td>
<td>0.975696</td>
<td>Add</td>
<td>hit</td>
</tr>
<tr>
<td>869</td>
<td>0.976967</td>
<td>Add</td>
<td>force</td>
</tr>
<tr>
<td>1597</td>
<td>0.977625</td>
<td>Add</td>
<td>attempt</td>
</tr>
<tr>
<td>5977</td>
<td>0.978384</td>
<td>Add</td>
<td>finger</td>
</tr>
<tr>
<td>6162</td>
<td>0.978776</td>
<td>Add</td>
<td>knife</td>
</tr>
<tr>
<td>3434</td>
<td>0.980028</td>
<td>Add</td>
<td>sharp</td>
</tr>
<tr>
<td>1504</td>
<td>0.980160</td>
<td>Add</td>
<td>struck</td>
</tr>
<tr>
<td>39157</td>
<td>0.980219</td>
<td>Add</td>
<td>slug</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>Distance↑</th>
<th>Adjust</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>2964</td>
<td>0.000000</td>
<td>Remove</td>
<td>iPhone</td>
</tr>
<tr>
<td>6377</td>
<td>0.359153</td>
<td>Add</td>
<td>iPad</td>
</tr>
<tr>
<td>22542</td>
<td>0.554838</td>
<td>Add</td>
<td>iOS</td>
</tr>
<tr>
<td>10081</td>
<td>0.585379</td>
<td>Add</td>
<td>smartphone</td>
</tr>
<tr>
<td>5824</td>
<td>0.587948</td>
<td>Add</td>
<td>iPod</td>
</tr>
<tr>
<td>43921</td>
<td>0.608292</td>
<td>Add</td>
<td>PlayBook</td>
</tr>
<tr>
<td>18025</td>
<td>0.653021</td>
<td>Add</td>
<td>iPhones</td>
</tr>
<tr>
<td>6439</td>
<td>0.656983</td>
<td>Add</td>
<td>Android</td>
</tr>
<tr>
<td>38104</td>
<td>0.681779</td>
<td>Add</td>
<td>3GS</td>
</tr>
<tr>
<td>8088</td>
<td>0.690880</td>
<td>Add</td>
<td>BlackBerry</td>
</tr>
<tr>
<td>24581</td>
<td>0.696648</td>
<td>Add</td>
<td>Zune</td>
</tr>
<tr>
<td>33435</td>
<td>0.713150</td>
<td>Add</td>
<td>Smartphone</td>
</tr>
<tr>
<td>19186</td>
<td>0.714883</td>
<td>Add</td>
<td>BlackBerry</td>
</tr>
<tr>
<td>9326</td>
<td>0.715027</td>
<td>Add</td>
<td>handset</td>
</tr>
<tr>
<td>26020</td>
<td>0.739856</td>
<td>Add</td>
<td>Droid</td>
</tr>
<tr>
<td>30557</td>
<td>0.756973</td>
<td>Add</td>
<td>Treo</td>
</tr>
<tr>
<td>12057</td>
<td>0.762164</td>
<td>Add</td>
<td>smartphones</td>
</tr>
<tr>
<td>6878</td>
<td>0.769016</td>
<td>Add</td>
<td>app</td>
</tr>
<tr>
<td>8211</td>
<td>0.779153</td>
<td>Add</td>
<td>iTunes</td>
</tr>
<tr>
<td>28120</td>
<td>0.787939</td>
<td>Add</td>
<td>iPads</td>
</tr>
</tbody>
</table>
Solving Analogies

- Embedding vectors trained for the language modeling task have very interesting properties (especially the skip-gram model).
Solving Analogies

- Embedding vectors trained for the language modeling task have very interesting properties (especially the skip-gram model).

\[ \mathbf{E}(\text{hotter}) - \mathbf{E}(\text{hot}) + \mathbf{E}(\text{big}) \approx \mathbf{E}(\text{bigger}) \]

\[ \mathbf{E}(\text{Rome}) - \mathbf{E}(\text{Italy}) + \mathbf{E}(\text{Germany}) \approx \mathbf{E}(\text{Berlin}) \]
Solving Analogies

- Embedding vectors trained for the language modeling task have very interesting properties (especially the skip-gram model).

\[ E(\text{hotter}) - E(\text{hot}) + E(\text{big}) \approx E(\text{bigger}) \]

\[ E(\text{Rome}) - E(\text{Italy}) + E(\text{Germany}) \approx E(\text{Berlin}) \]

Skip-gram model w/ 640 dimensions trained on 6B words of news text achieves 57% accuracy for analogy-solving test set.
Solving Analogies

- Embedding vectors trained for the language modeling task have very interesting properties (especially the skip-gram model).

\[ E(\text{hotter}) - E(\text{hot}) + E(\text{big}) \approx E(\text{bigger}) \]

\[ E(\text{Rome}) - E(\text{Italy}) + E(\text{Germany}) \approx E(\text{Berlin}) \]

Skip-gram model w/ 640 dimensions trained on 6B words of news text achieves 57% accuracy for analogy-solving test set.

Visualizing the Embedding Space

Country and Capital Vectors Projected by PCA
Important Problems w.r.t. Representations

- Representing data in both raw form and in terms of high level representations derived from raw data will be important.

- If we want to store and manipulate derived features in addition to raw data:
  - how do we design systems to perform fast high-level queries against large corpora?
  - how do we automatically and quickly incorporate new data into our model of the world?
  - how do we generalize from one particular task to many other tasks?
  - how do we minimize human effort for accomplishing all of this?
Automatic Representations

• In the future, I believe:
  – Systems will become more self-managing and self-tuning
  
  – Automatically building high-level representations from raw data will be key to answering difficult queries about raw data
  
  – Being able to combine many different types of data together will be important
Thanks!

• Questions? Thoughts?

Further reading:


• Corbett, Dean, ... Ghemawat, et al. Spanner: Google’s Globally-Distributed Database, OSDI 2012

• Dean & Barroso, The Tail at Scale, CACM Feb. 2013.

• Le, Ranzato, Monga, Devin, Chen, Corrado, Dean, & Ng. Building High-Level Features Using Large Scale Unsupervised Learning, ICML 2012.

• Dean et al., Large Scale Distributed Deep Networks, NIPS 2012.

• Mikolov, Chen, Corrado and Dean. Efficient Estimation of Word Representations in Vector Space, ICLR 2013.

• http://research.google.com/people/jeff