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

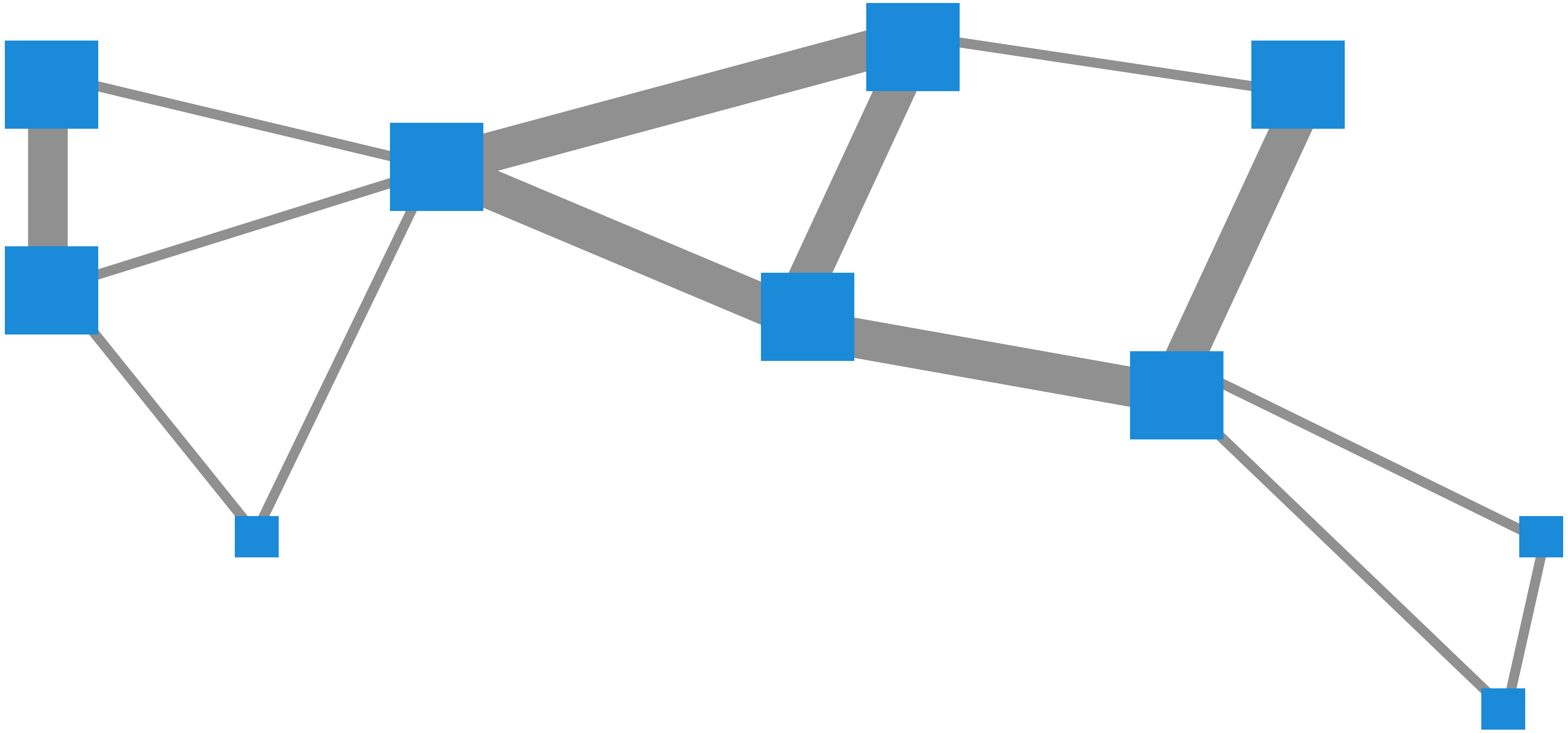
Building Applications on top of Weakly Consistent Storage Systems

- 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: ~10^6 to 10^7 machines, spread at 100s to 1000s of locations around the world, ~10^9 client machines



  - zones of semi-autonomous control
  - consistency after disconnected operation
  - power adaptivity

Google

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:

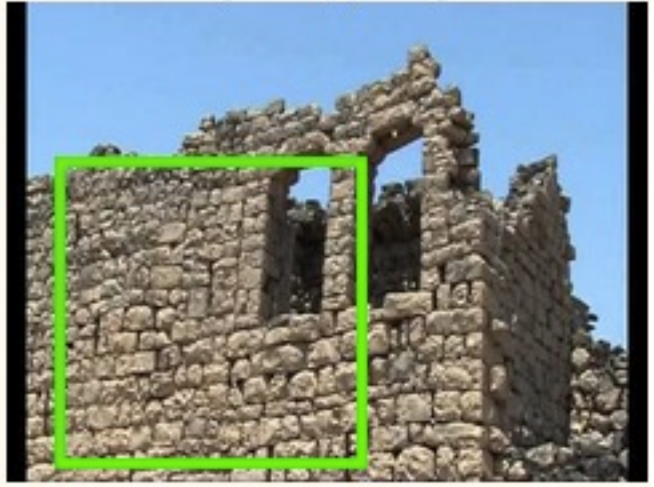
- image recognition, object detection, video processing
- speech recognition
- language modeling
- user activity prediction
- neuroscience
- ad system optimization
- language understanding
- ...

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]

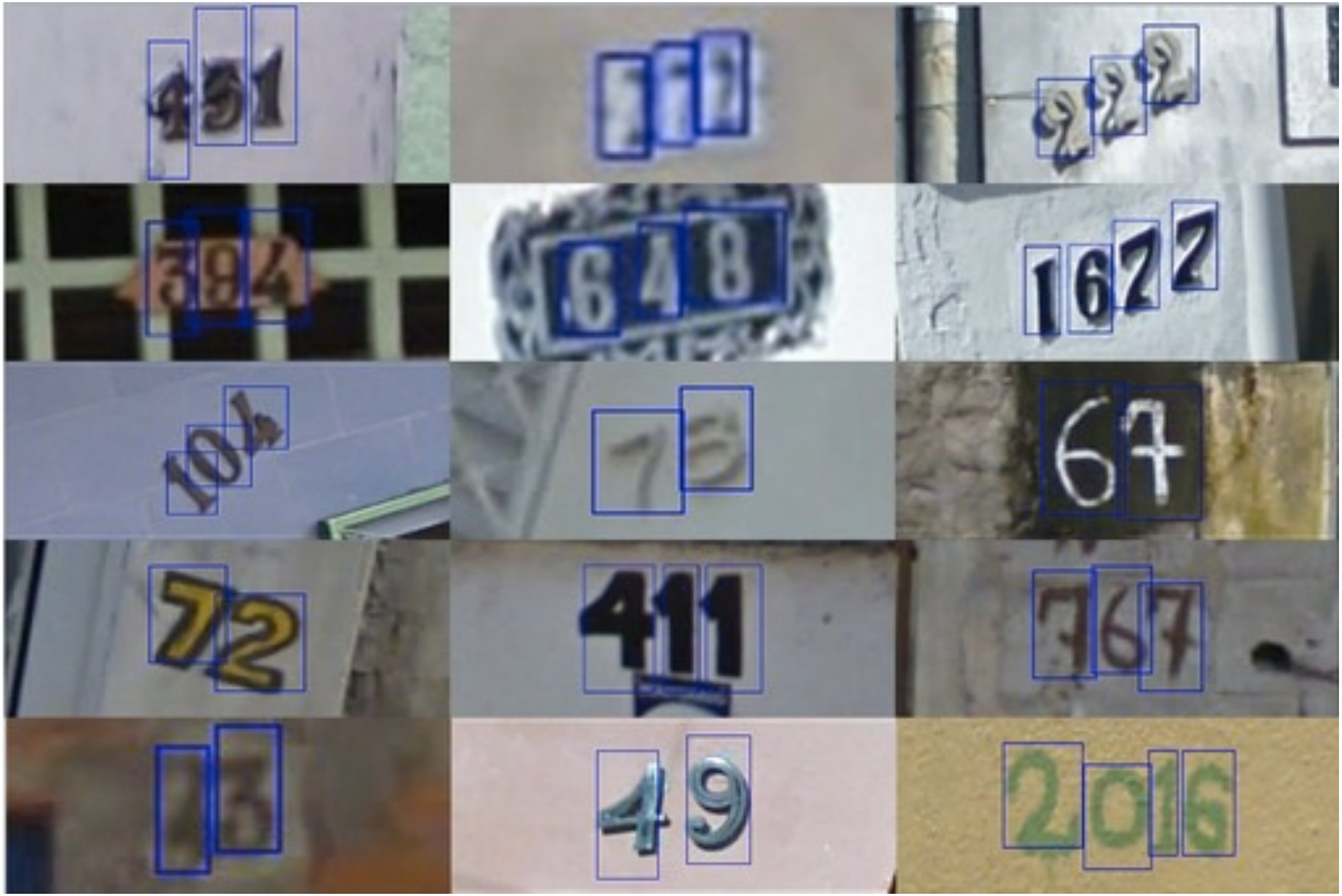
judo [0.91, web]

tractor [0.91, web]

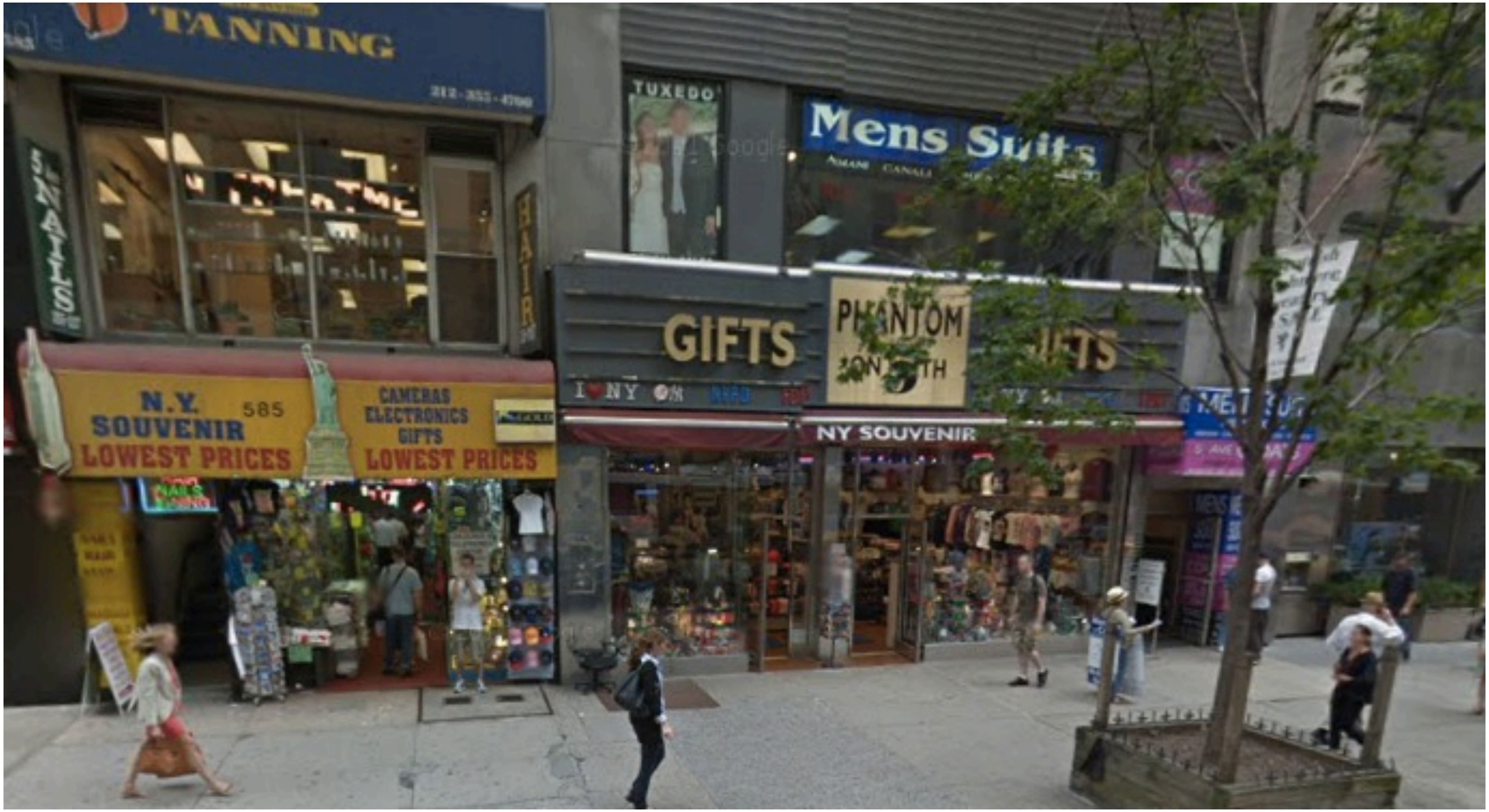
tractor [0.91, web]

tractor [0.94, web]

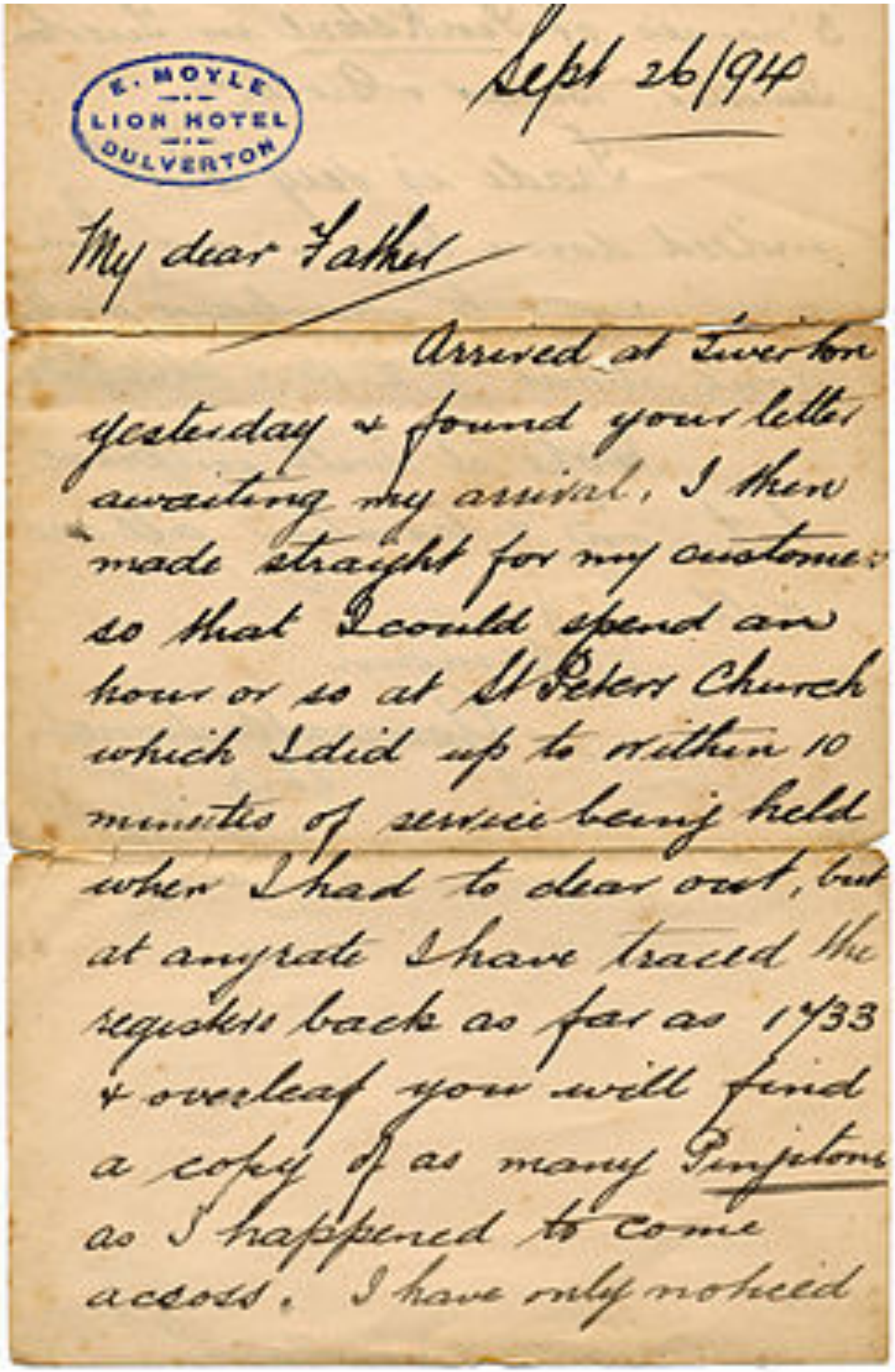
What are these numbers?



What are all these words?



How about these words?



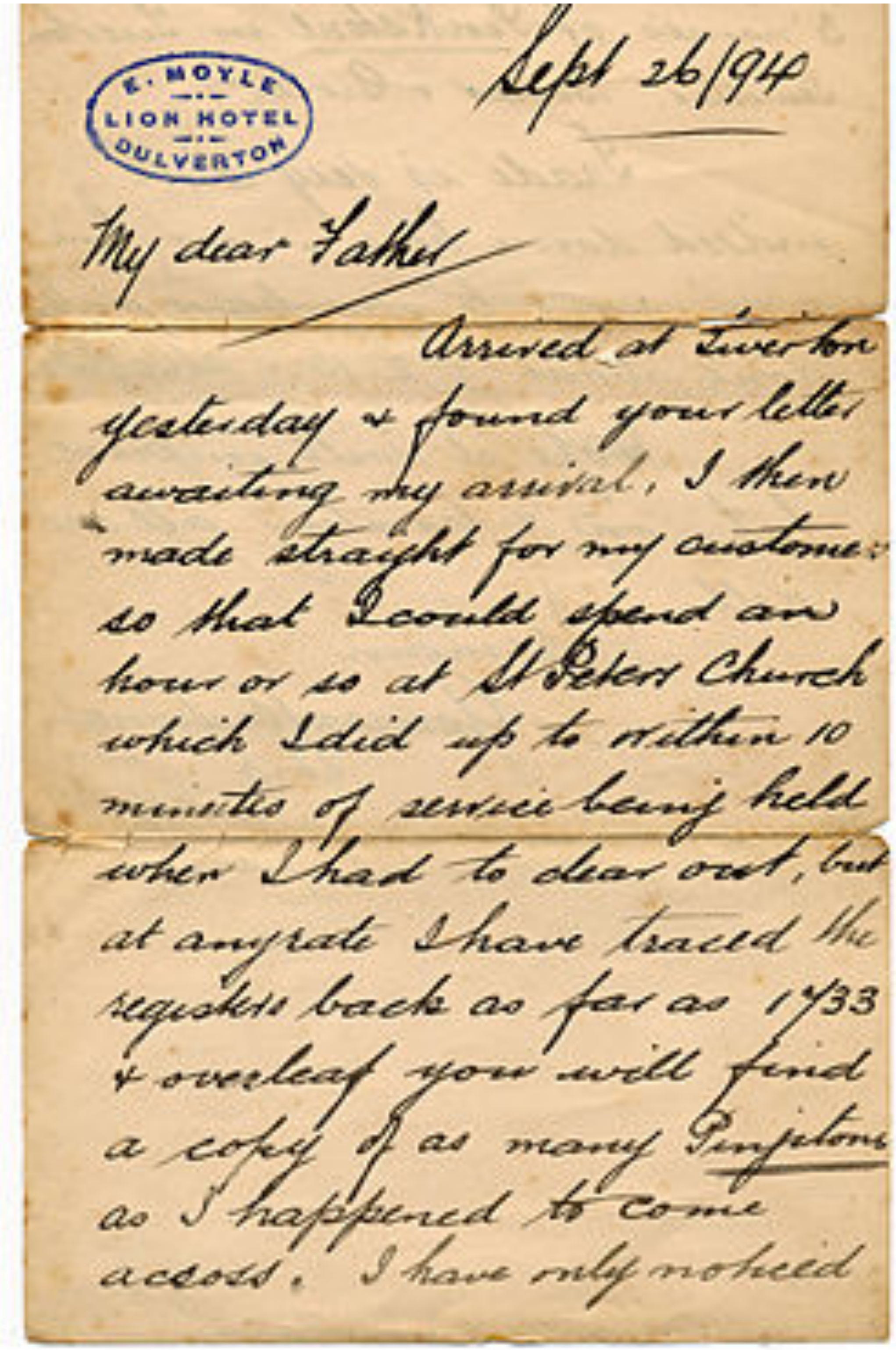
E. MOYLE
LION HOTEL
DULVERTON

Sept 26/94

My dear Father

Arrived at Tiverton yesterday & found your letter awaiting my arrival. I then made straight for my customer so that I could spend an hour or so at St Peters Church which I did up to within 10 minutes of service being held when I had to clear out, but at anyrate I have traced the registers back as far as 1733 & overleaf you will find a copy of as many Pimptone as I happened to come across. I have only noticed

How about these words?

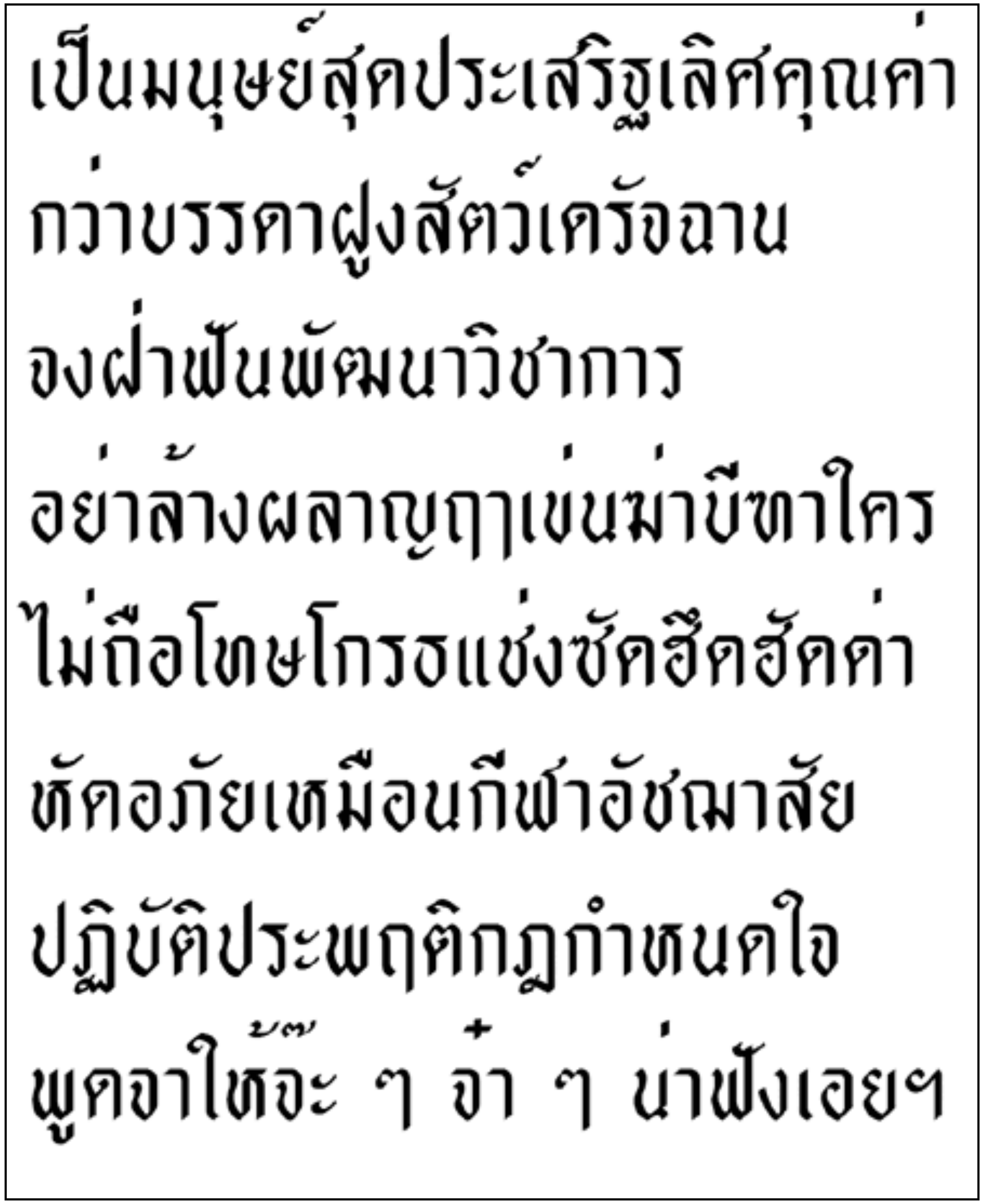


E. MOYLE
LION HOTEL
DULVERTON

Sept 26/94

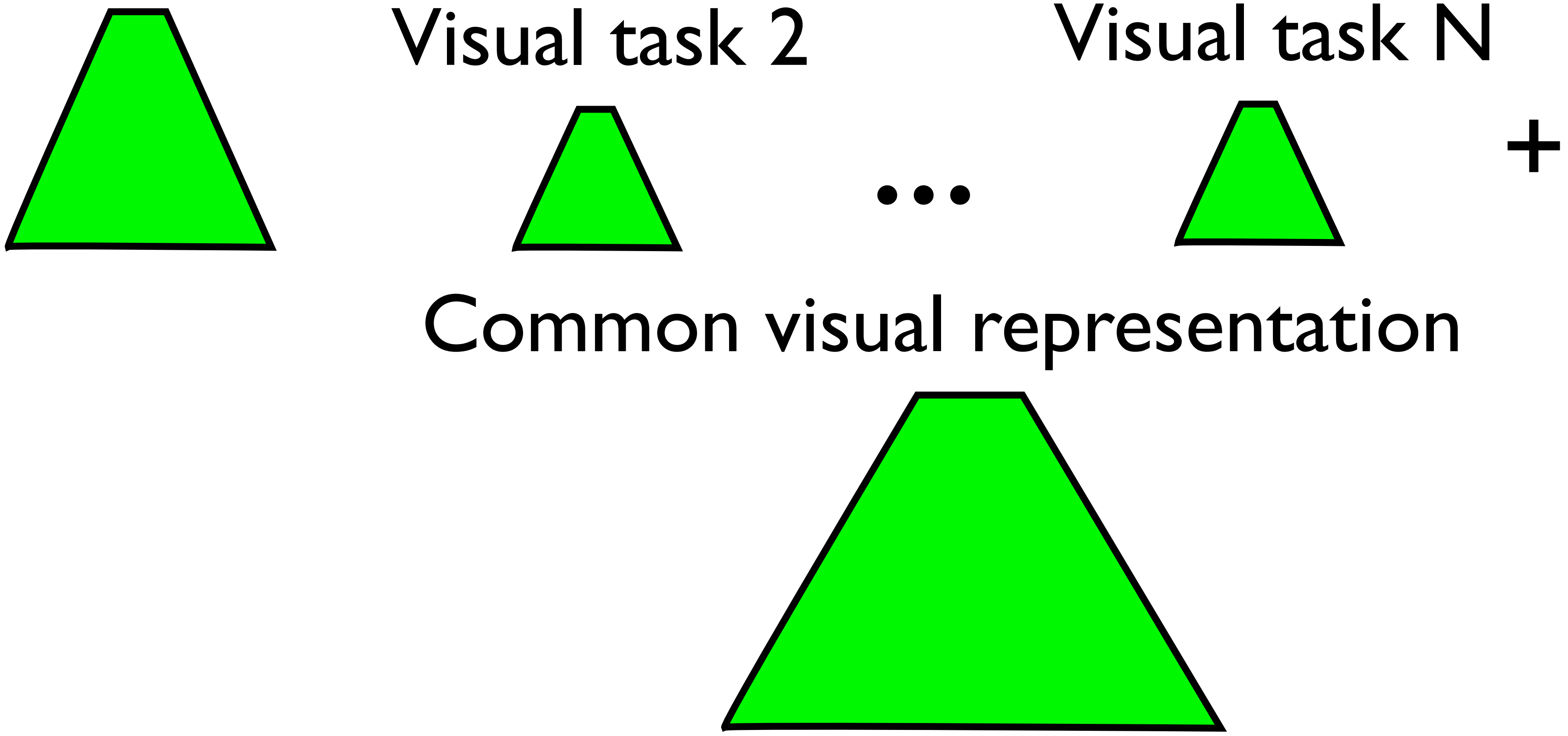
My dear Father

Arrived at Tiverton yesterday & found your letter awaiting my arrival. I then made straight for my customer so that I could spend an hour or so at St Peters Church which I did up to within 10 minutes of service being held when I had to clear out, but at anyrate I have traced the registers back as far as 1733 & overleaf you will find a copy of as many Pimptons as I happened to come across. I have only noticed



เป็นมนุษย์สุดประเสริฐเลิศคุณค่า
กว่าบรรดาฝูงสัตว์เดรัจฉาน
จงฝ่าฟันพัฒนาวิชาการ
อย่าล้างผลาญฤๅเข่นฆ่าบีฑาใคร
ไม่ถือโทษโกรธแช่งซัดฮึดฮัดด่า
หัดอภัยเหมือนกีฬาอัชฌาสัย
ปฏิบัติประพฤติกฎกำหนดใจ
พูดจาให้จ๊ะ ๆ จ๋า ๆ น่าฟังเอยฯ

Goal: Unified System



Visual task 1

Visual task 2

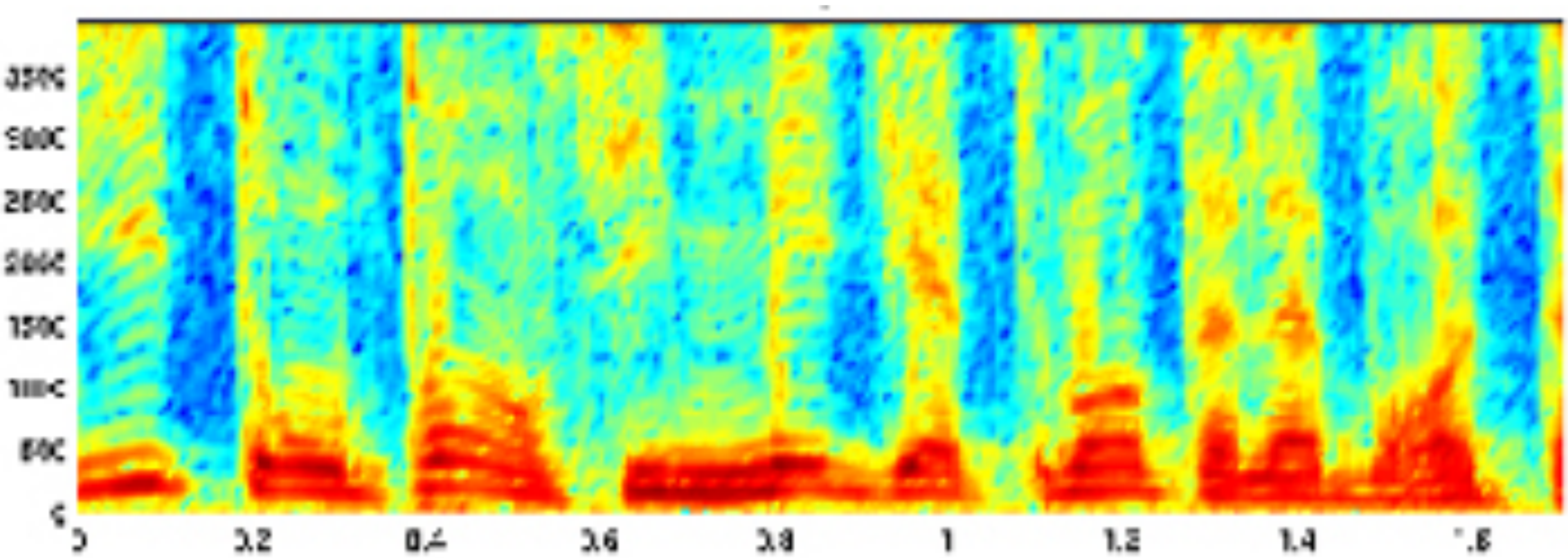
Visual task N

Unsupervised training

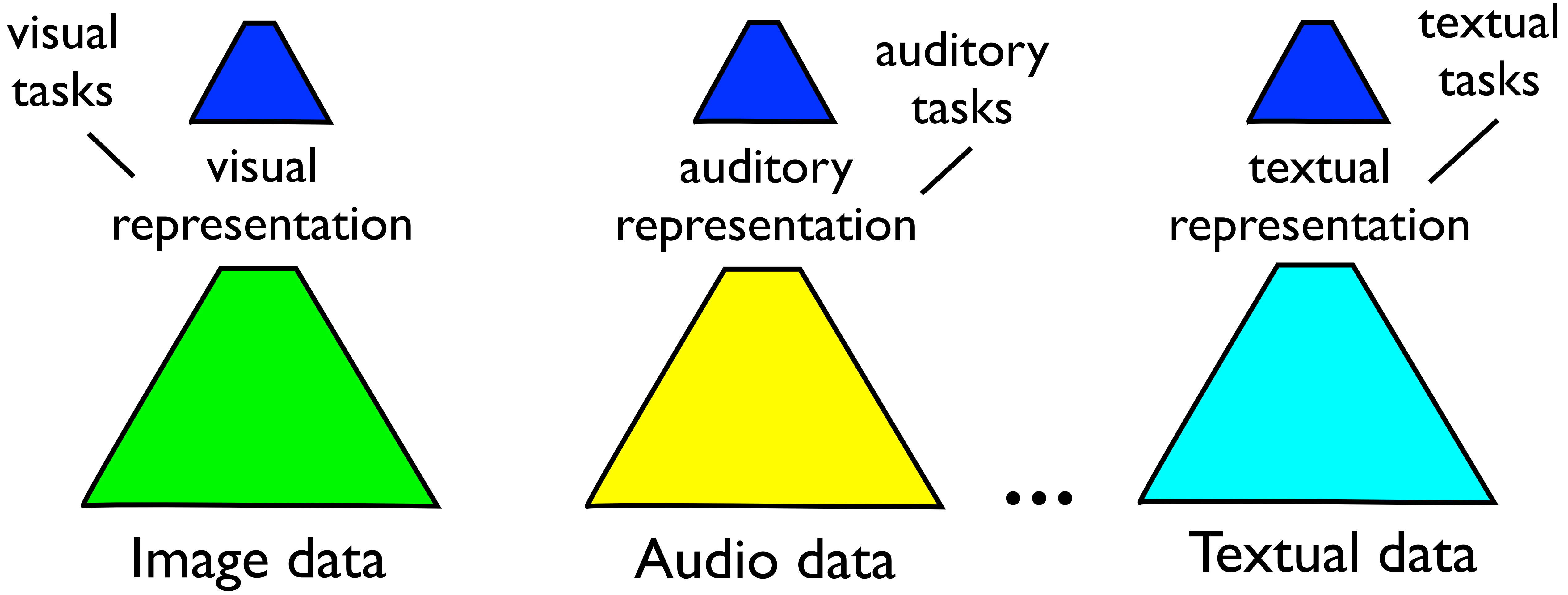
Common visual representation

Image data

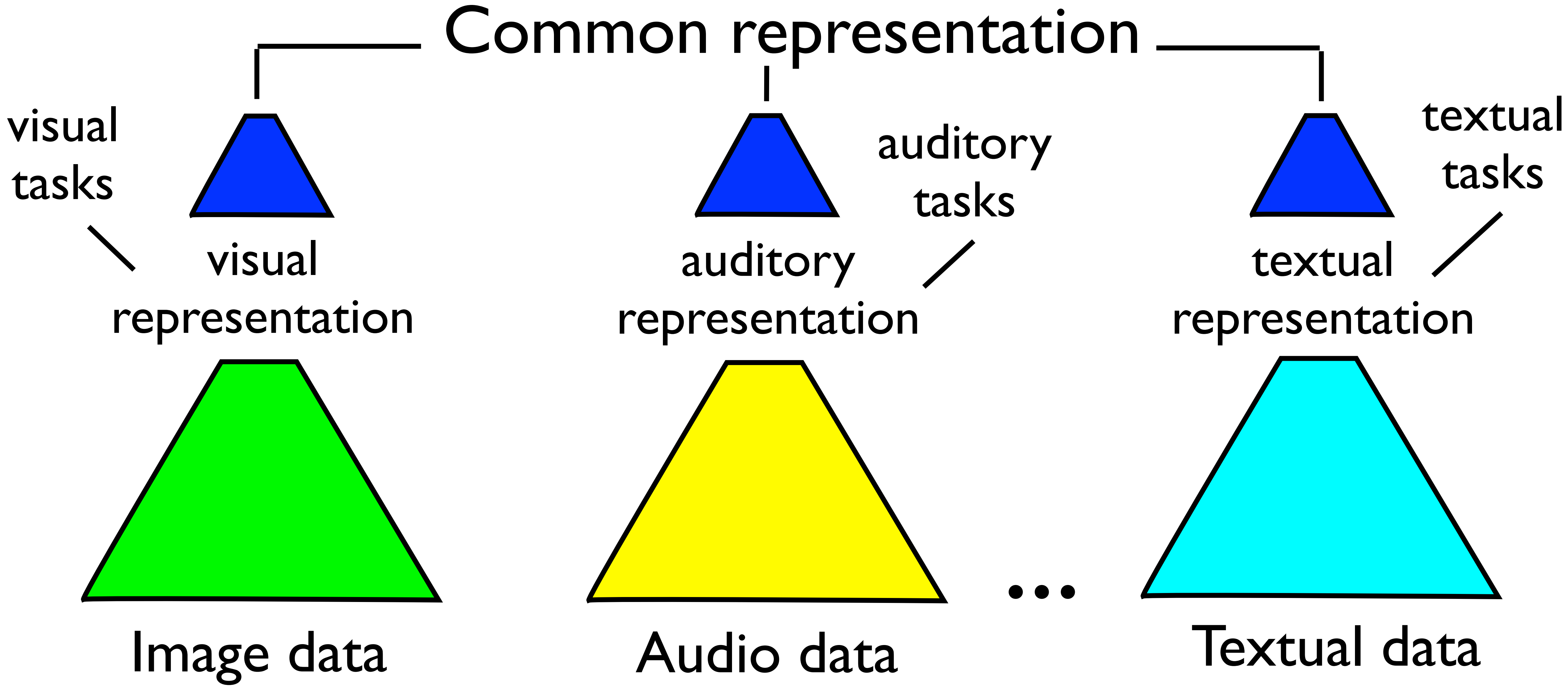
What is being said?



Goal: Unified System



Goal: Unified System


Common representation

visual tasks

visual representation

auditory tasks

auditory representation

textual tasks

textual representation

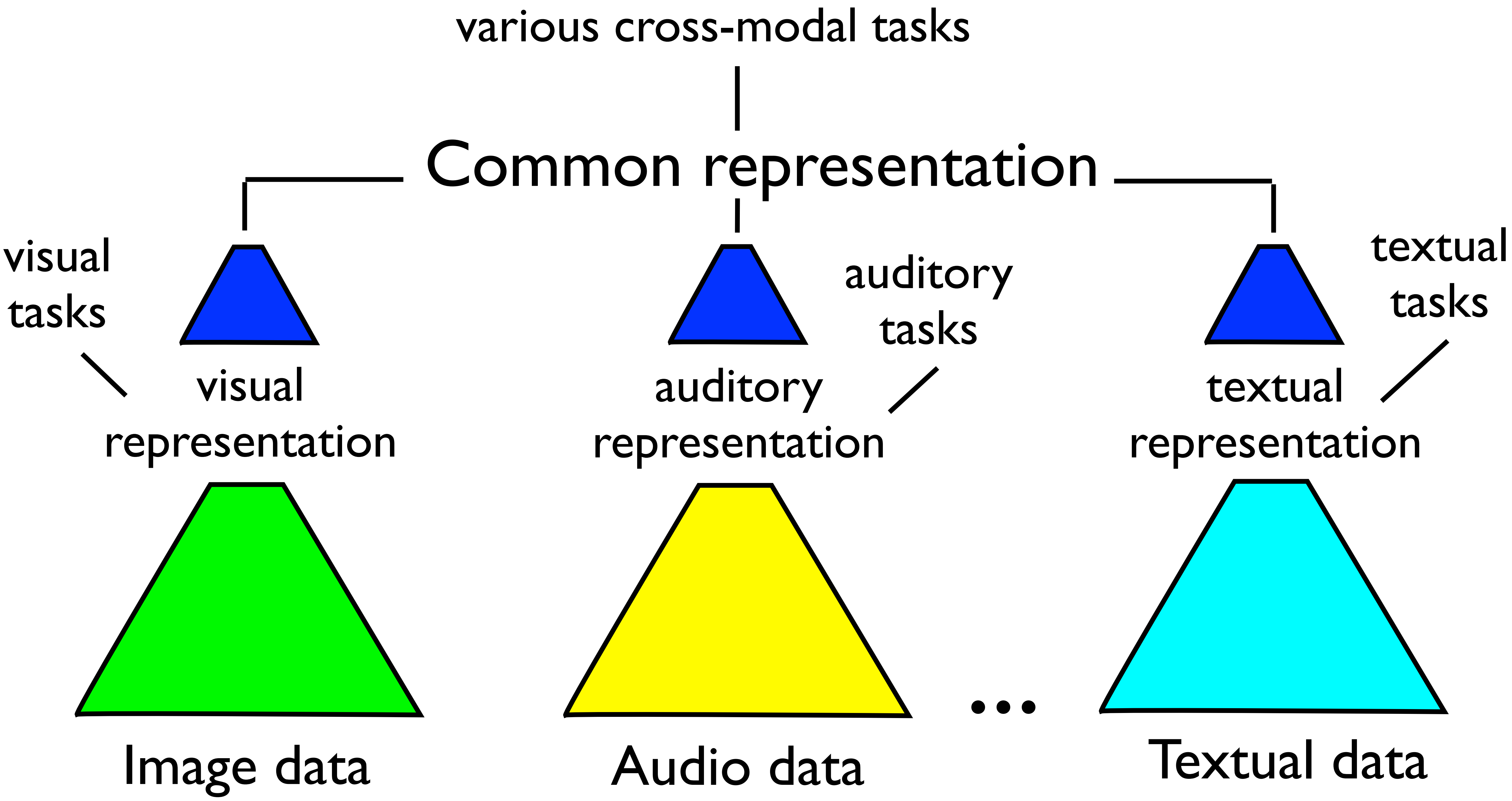
Image data

Audio data

...

Textual data

Goal: Unified System

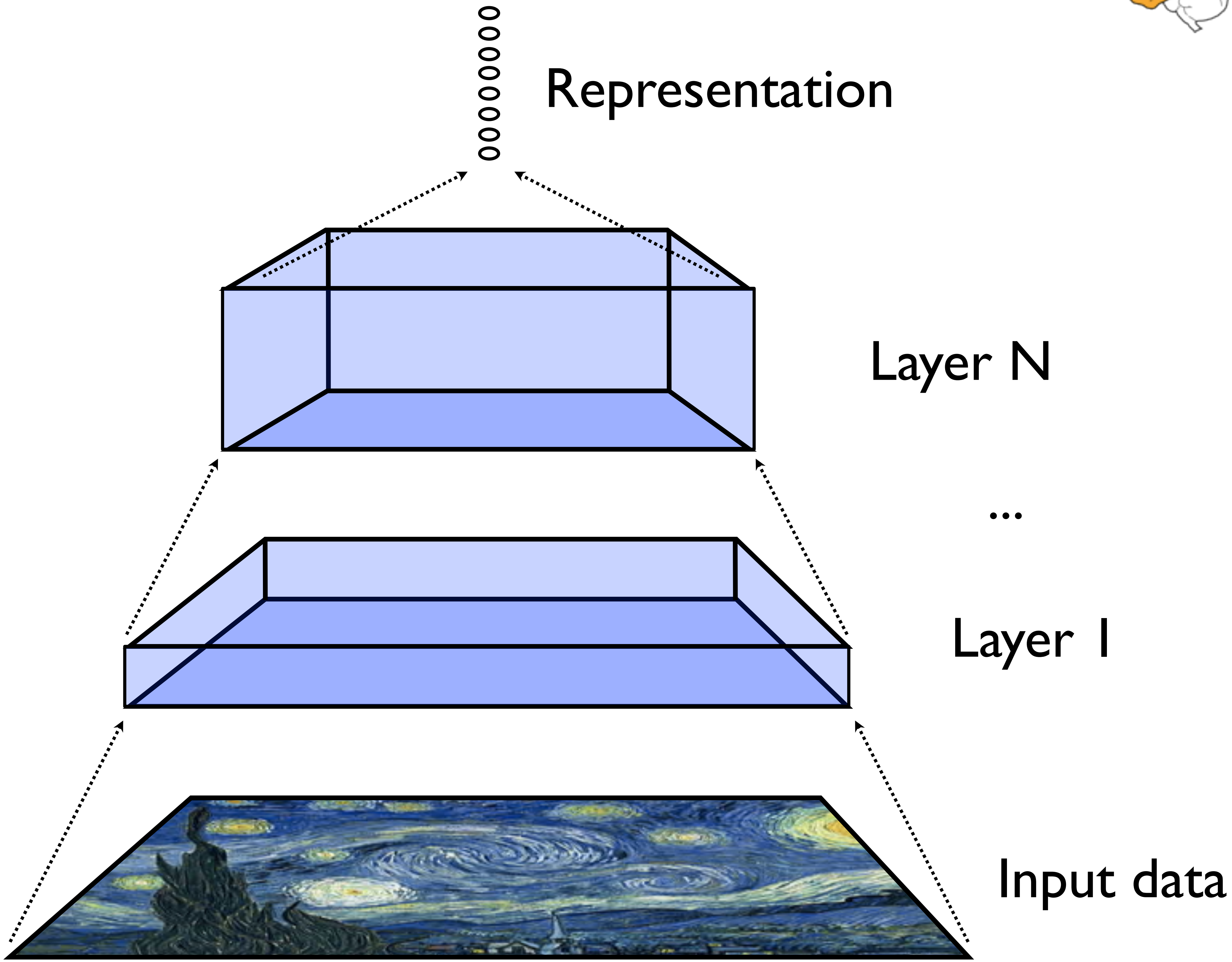


One Key Approach: Deep Learning

- 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 (<= ~50M parameters)

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



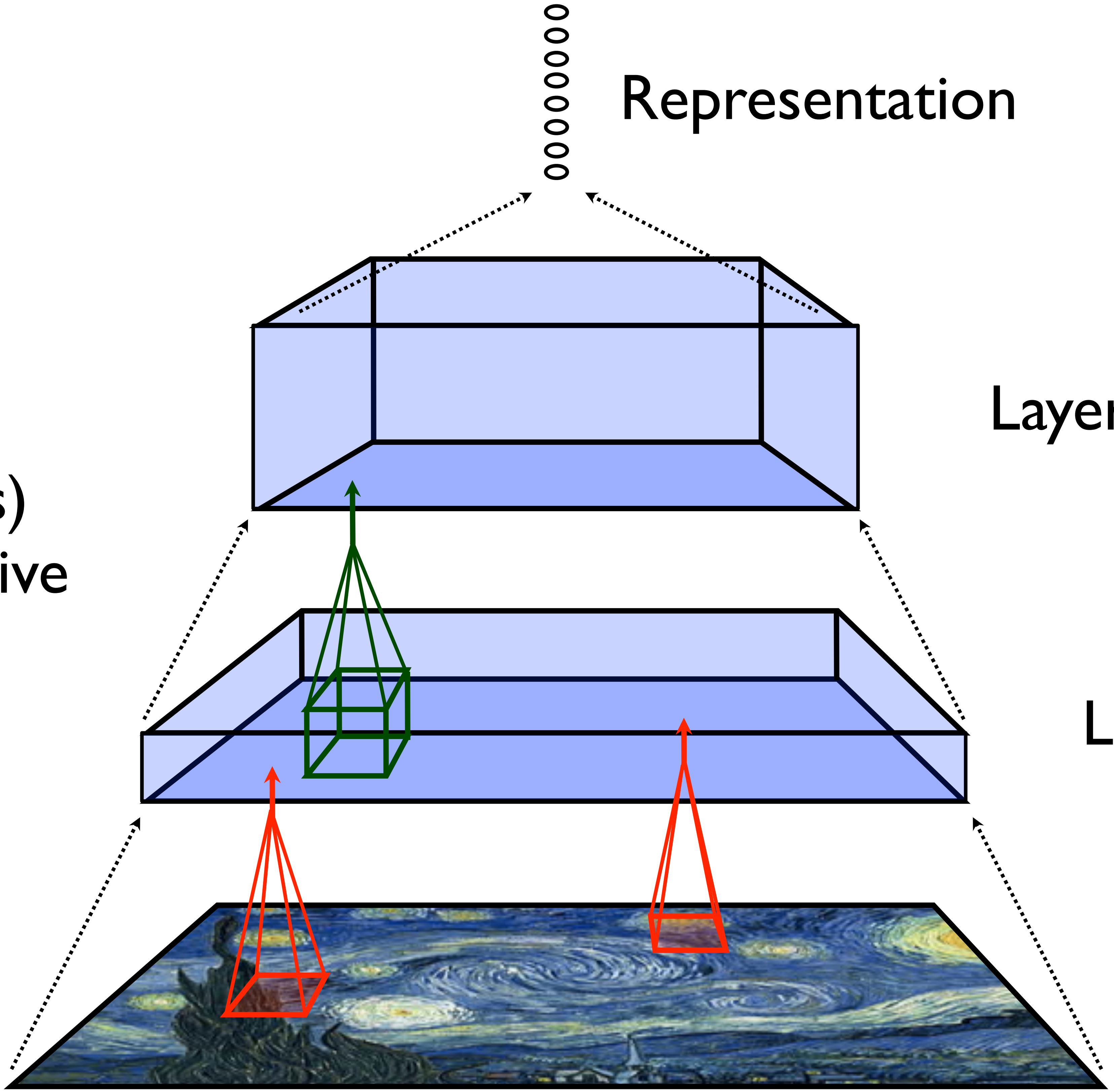
Representation

Layer N

...

Layer 1

Input data



Representation

Layer N

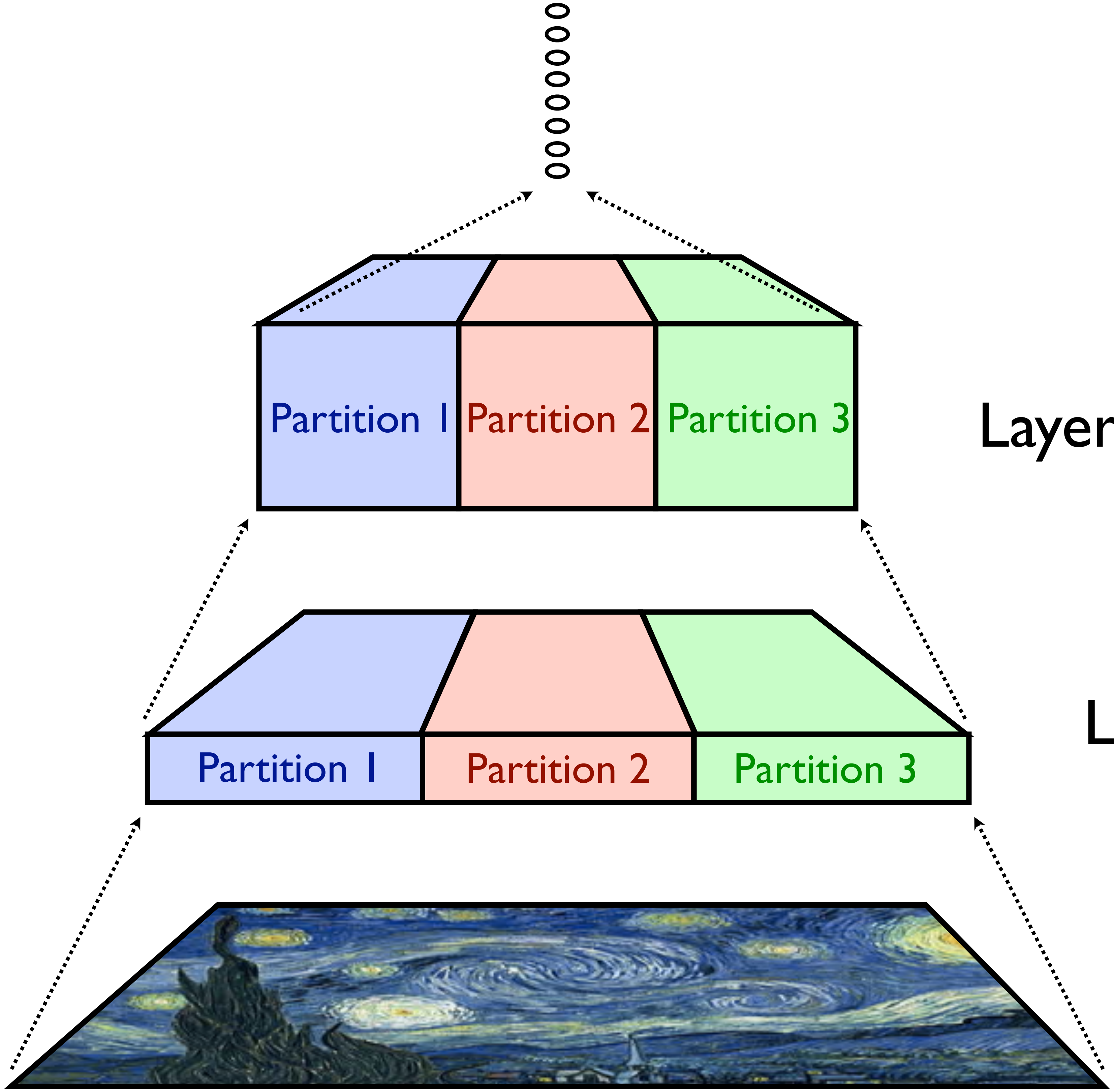
...

Layer 1

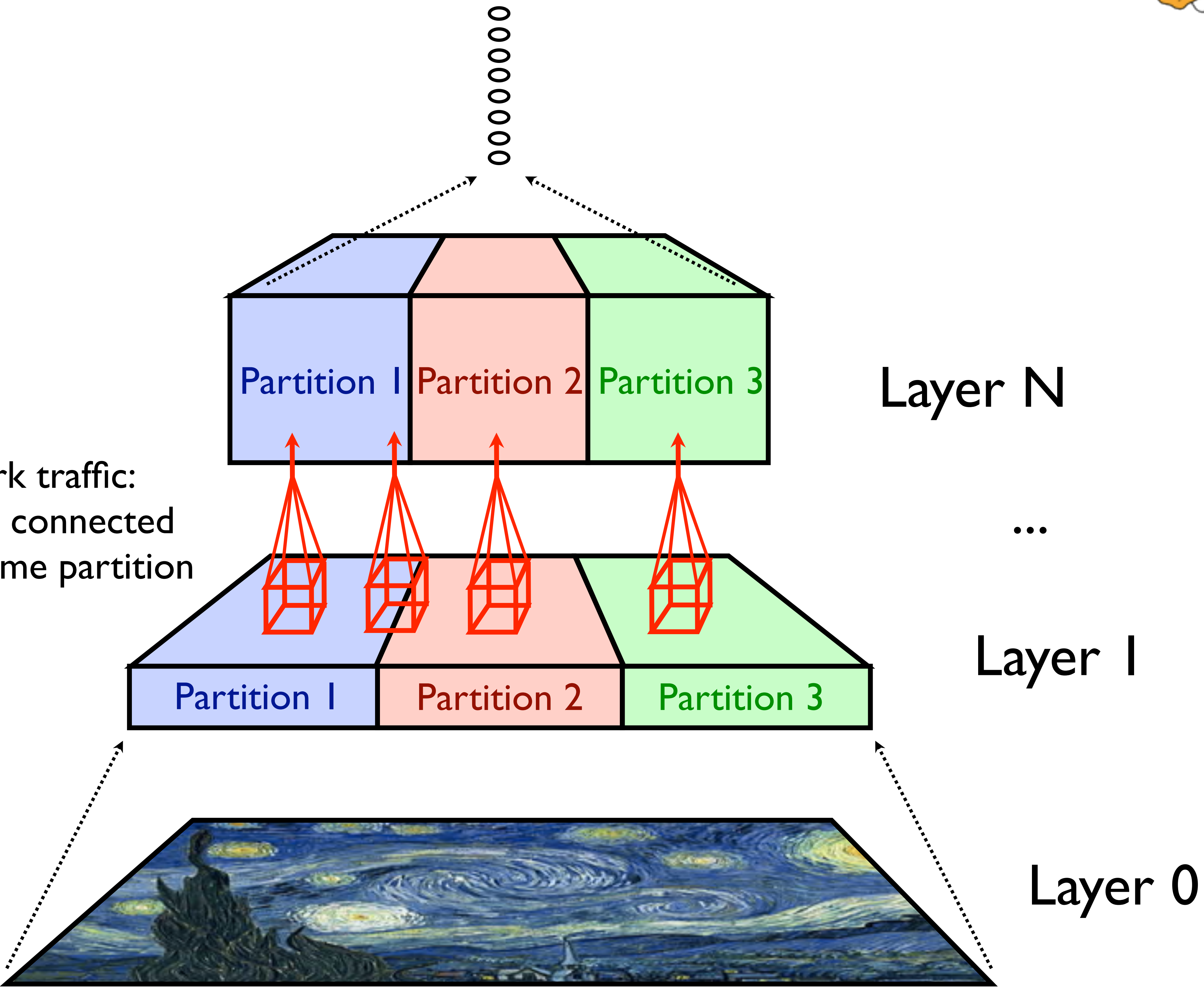
Input data

(Sometimes)
Local Receptive
Fields

Partition model across machines

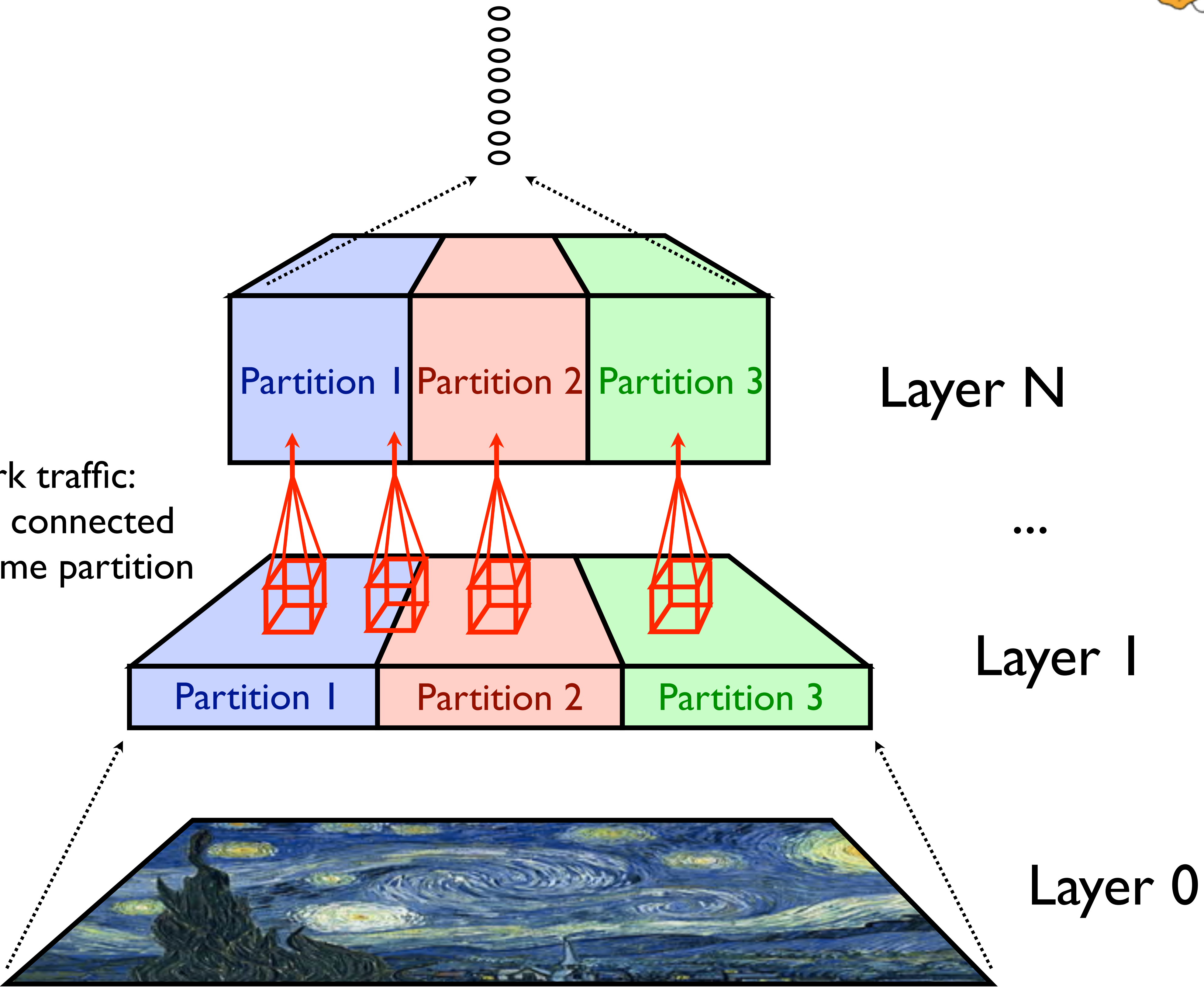


Partition model across machines



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 Upsupervised Learning

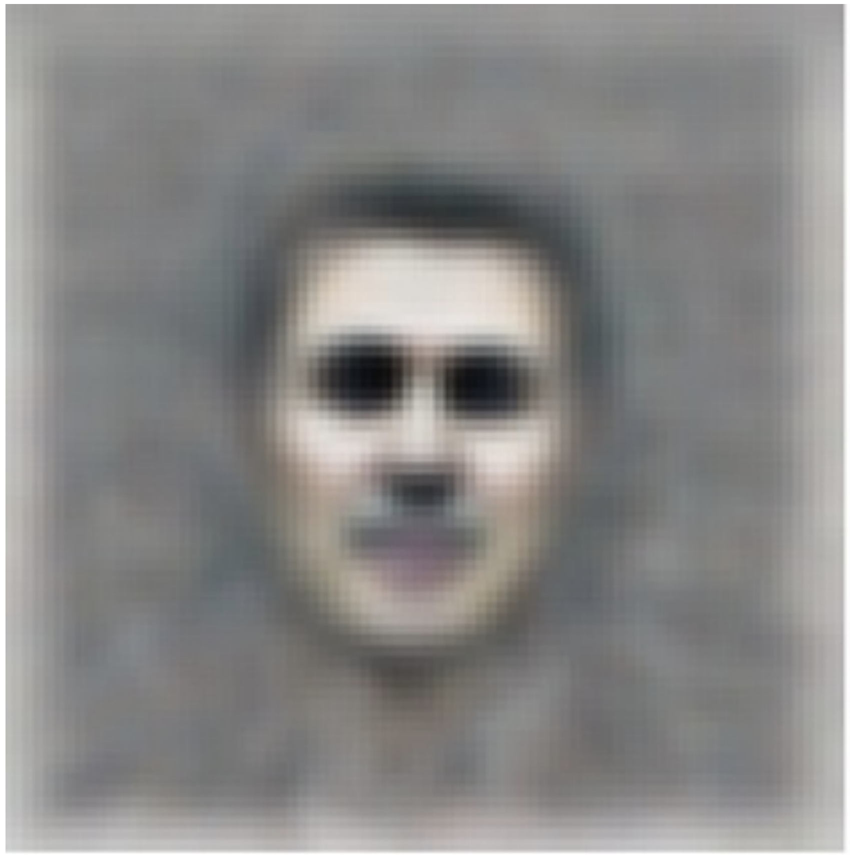
- Always: unlabeled data >> labeled data
- Experiment: unsupervised training on 10M random YouTube frames
- Trained 9 layer model with local connections

  Visualization of optimal stimuli for two different neurons in top layer:

Initial Focus on Upsupervised Learning

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

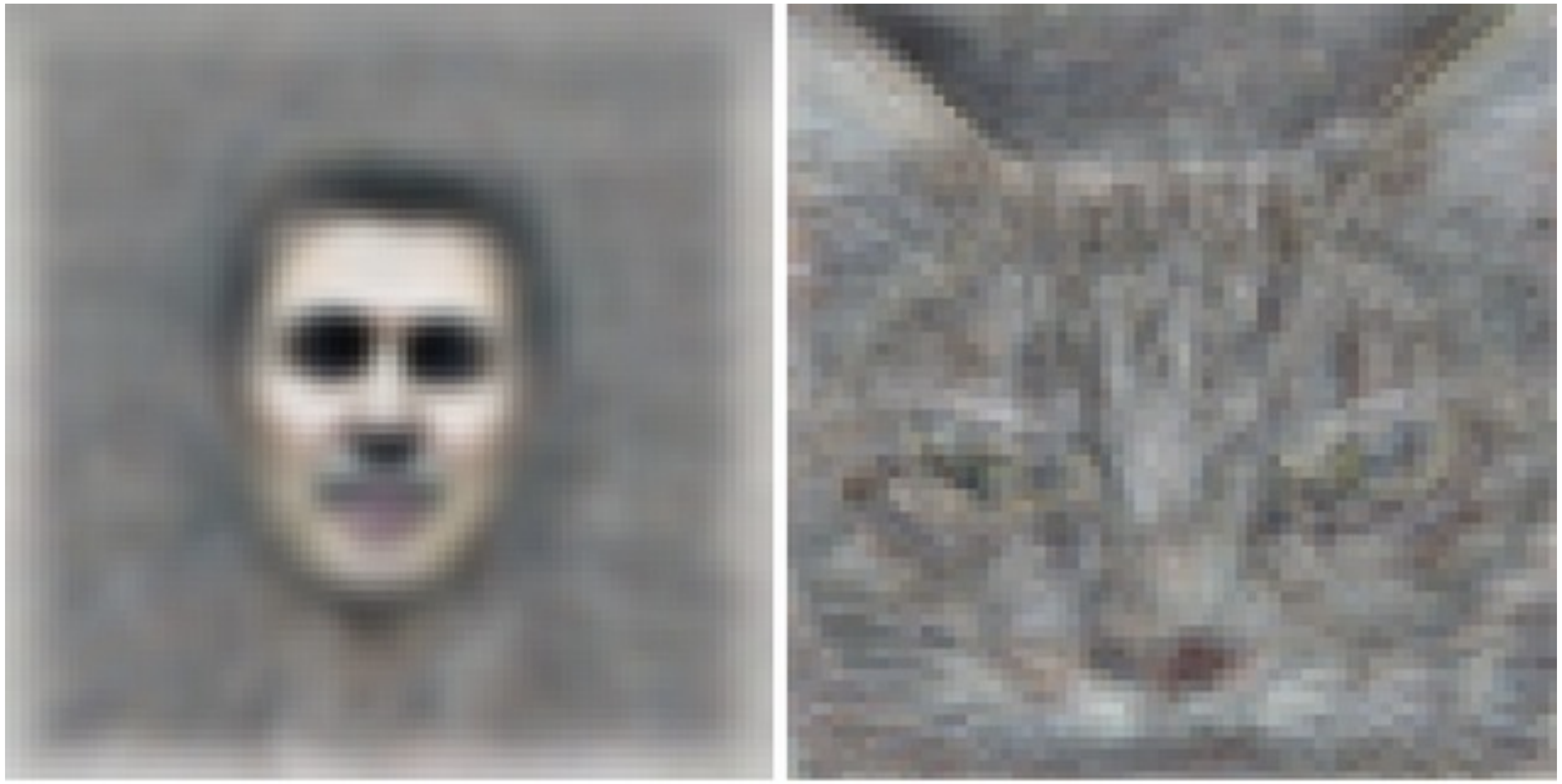
Visualization of optimal stimuli for two different neurons in top layer:



Initial Focus on Upsupervised Learning

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

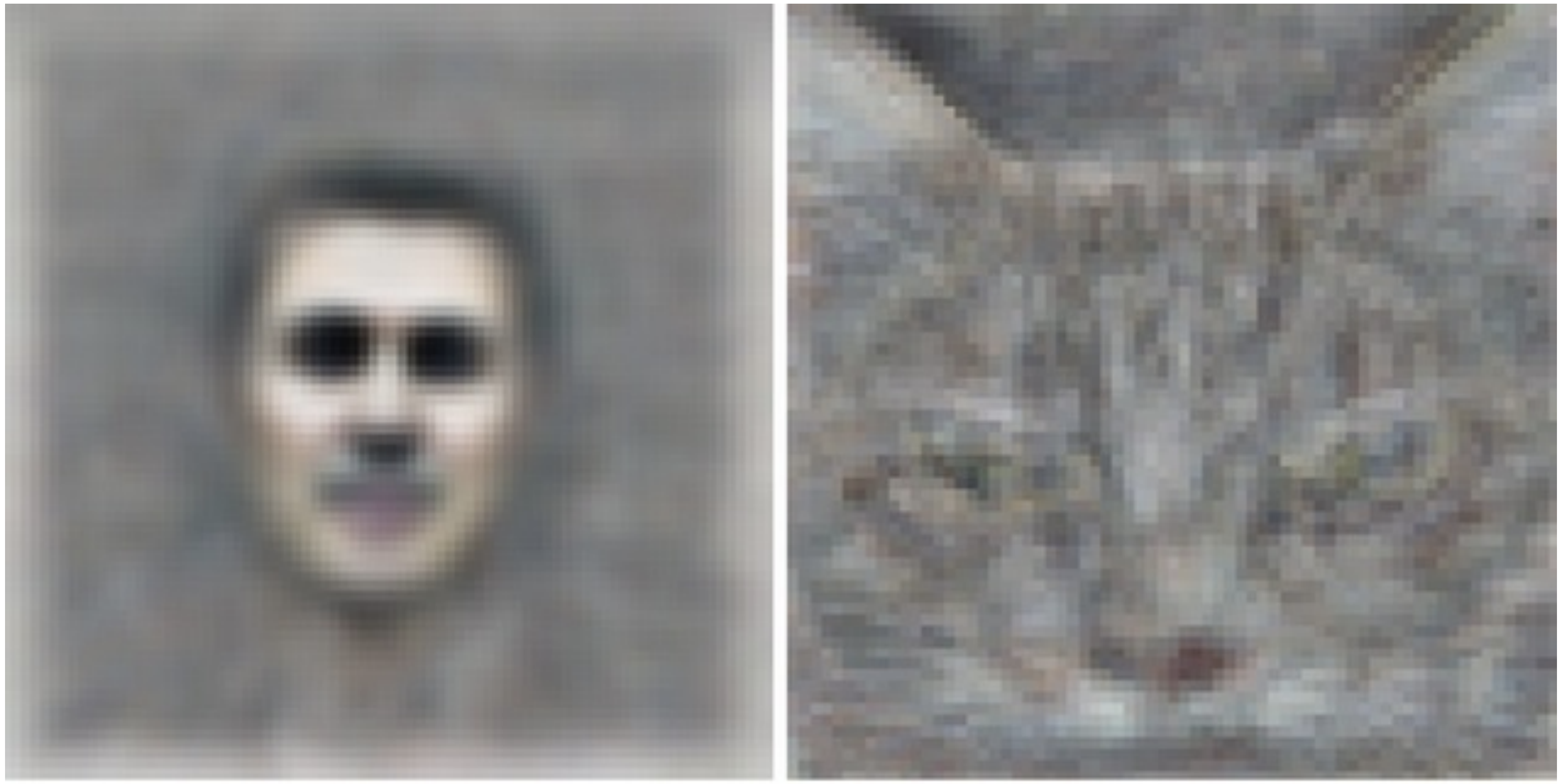
Visualization of optimal stimuli for two different neurons in top layer:



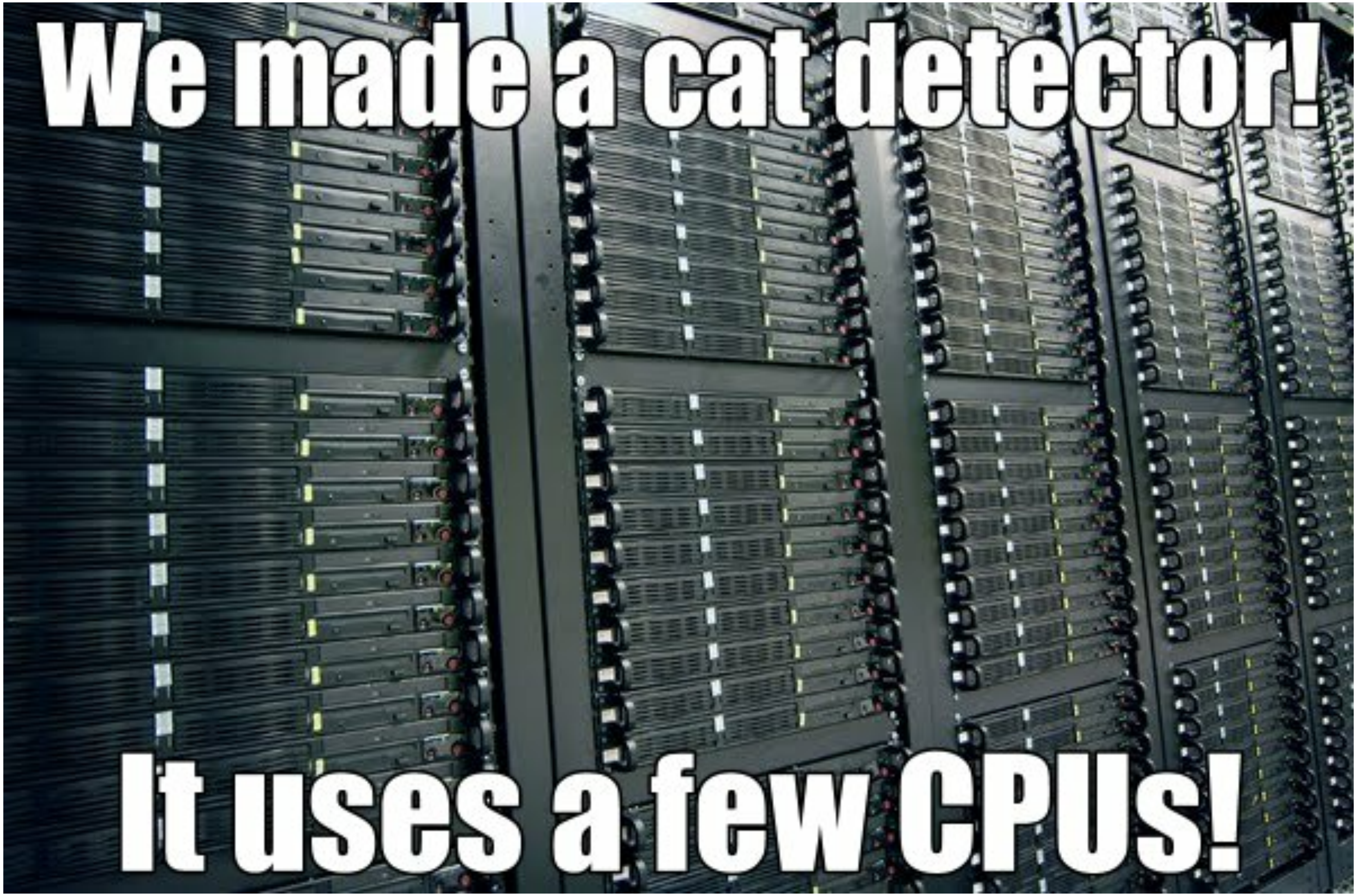
Initial Focus on Upsupervised Learning

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

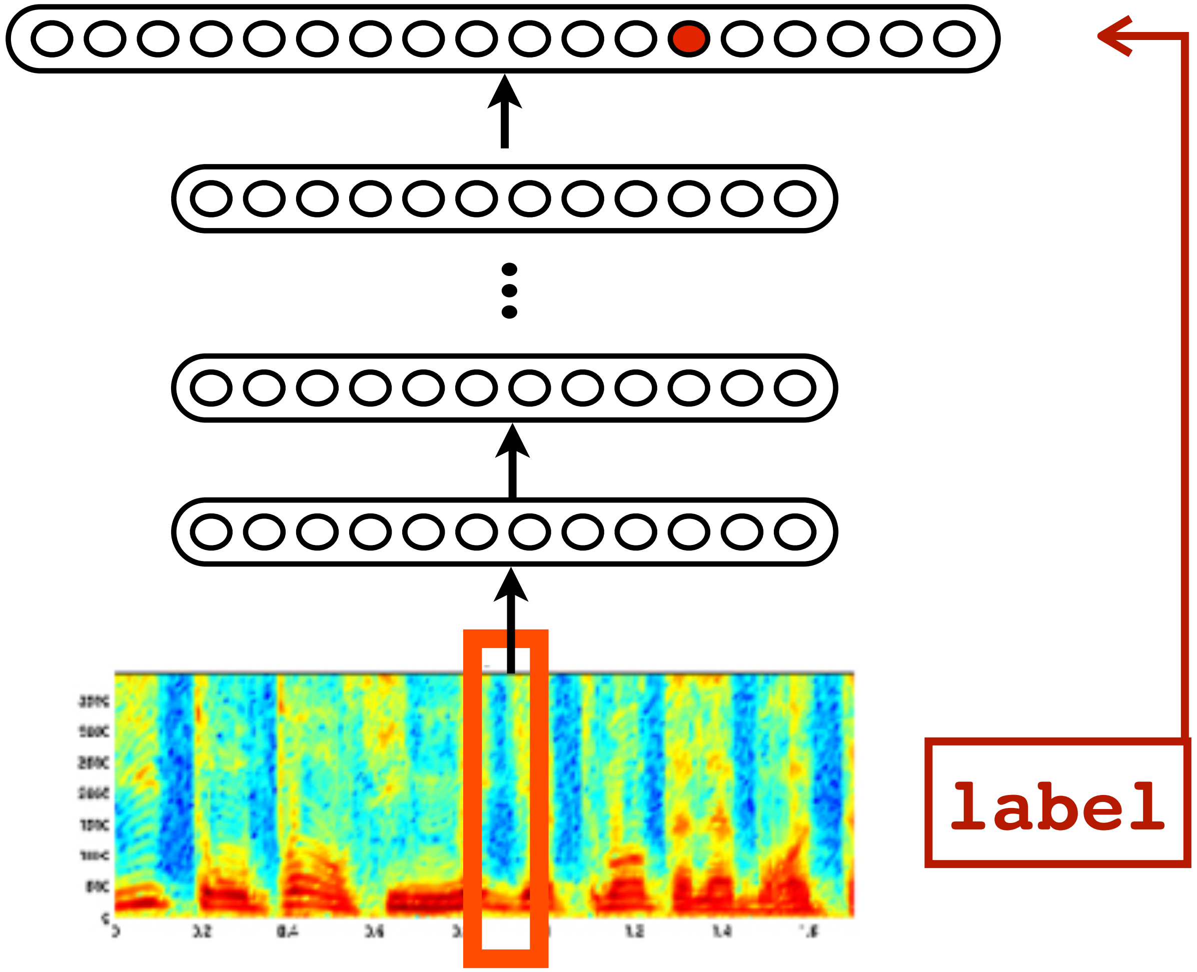
Visualization of optimal stimuli for two different neurons in top layer:



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!

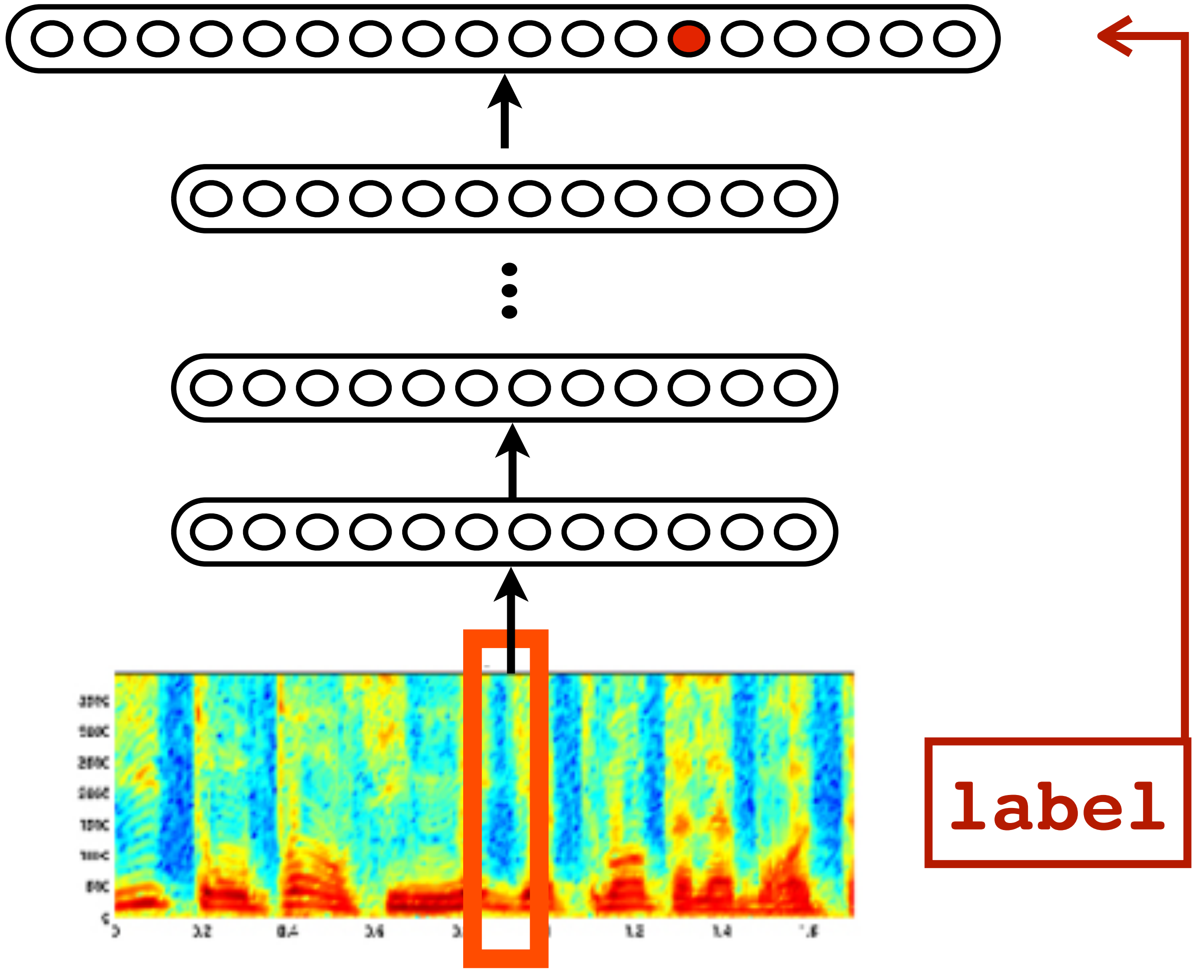
Acoustic Modeling for Speech Recognition



Close collaboration with Google Speech team

Trained in <5 days on cluster of 800 machines

Acoustic Modeling for Speech Recognition

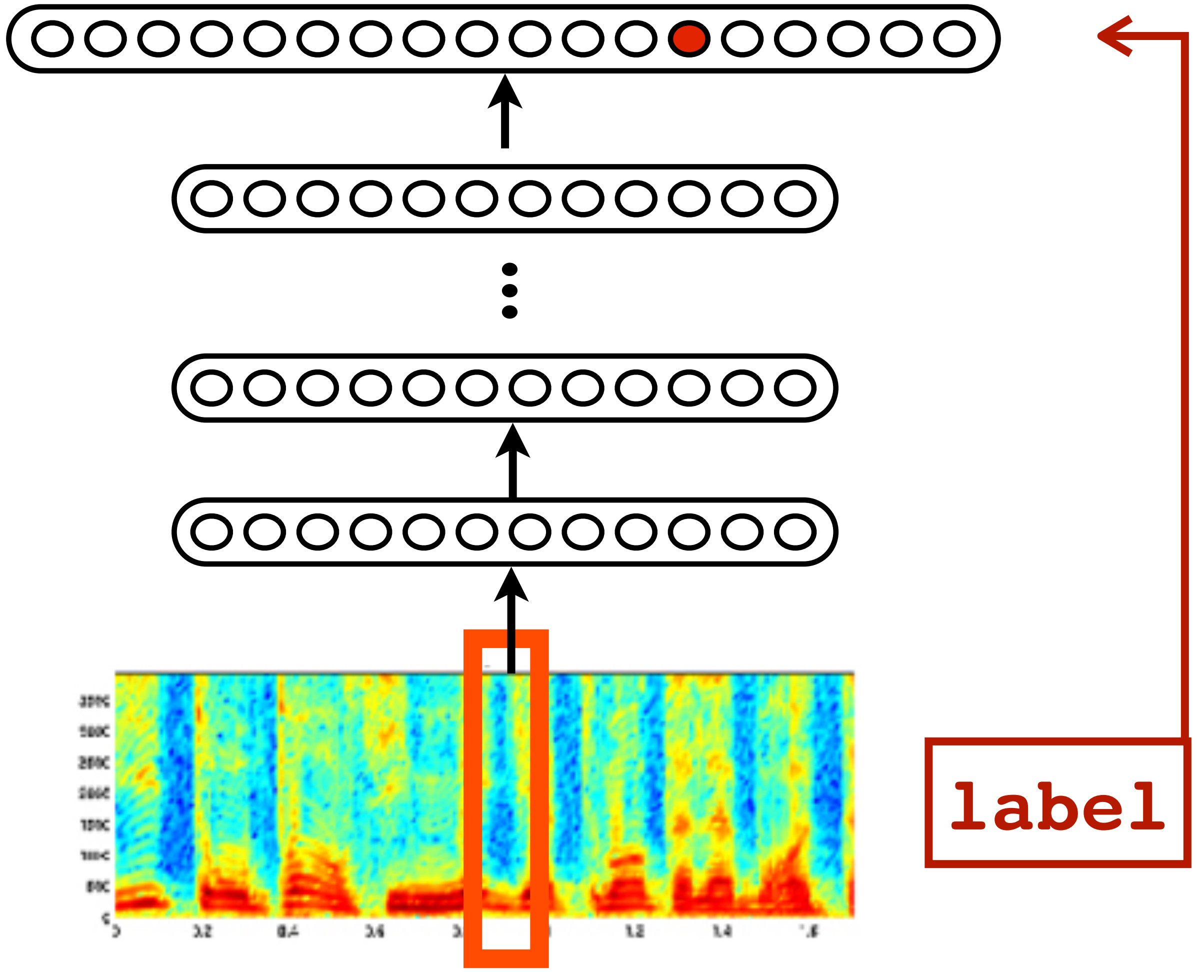


Close collaboration with Google Speech team

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")

Acoustic Modeling for Speech Recognition



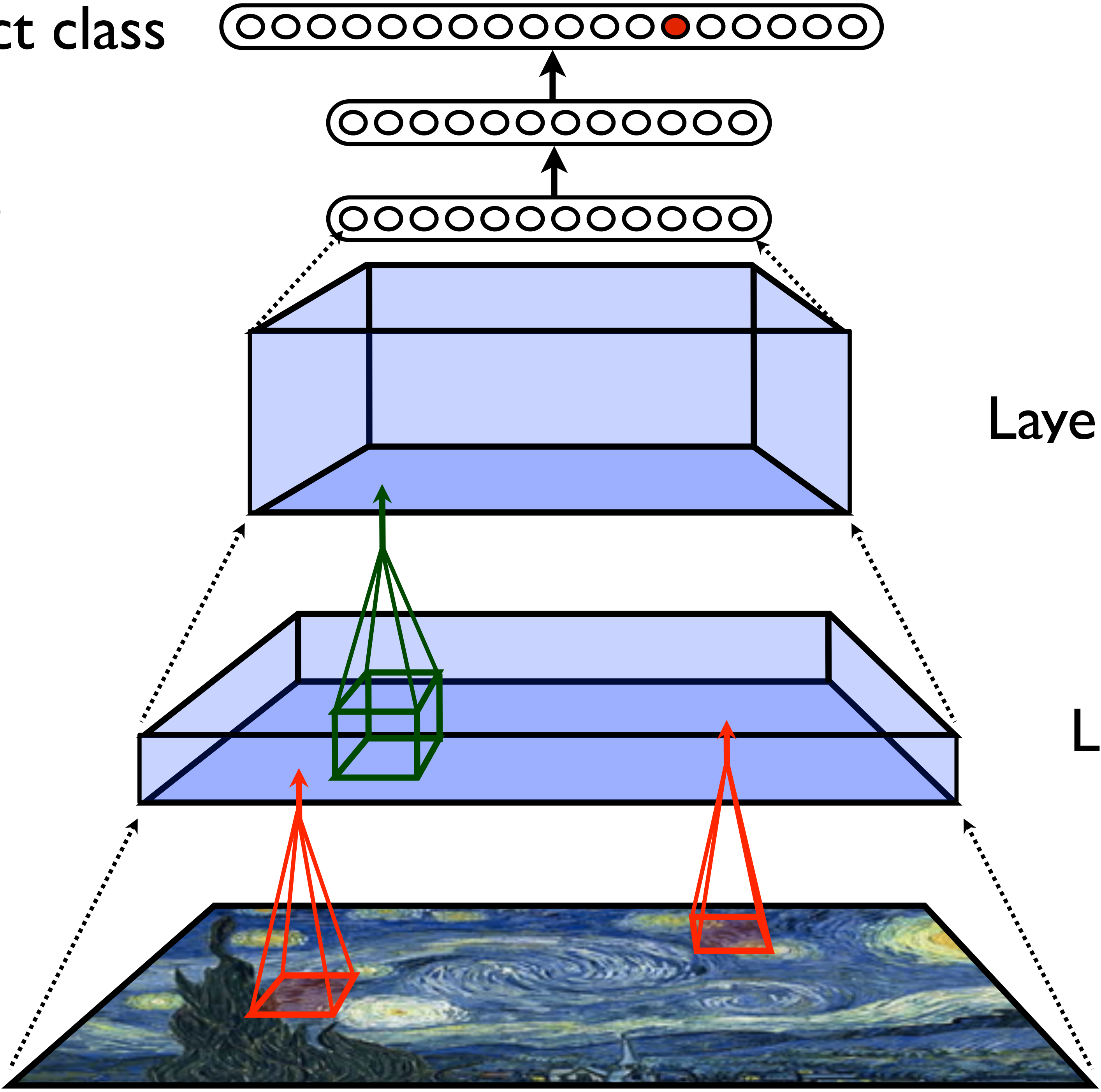
Close collaboration with Google Speech team

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

Convolutional Models for Object Recognition



Softmax to predict object class

Fully-connected layers

Layer 7

...

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

Layer 1

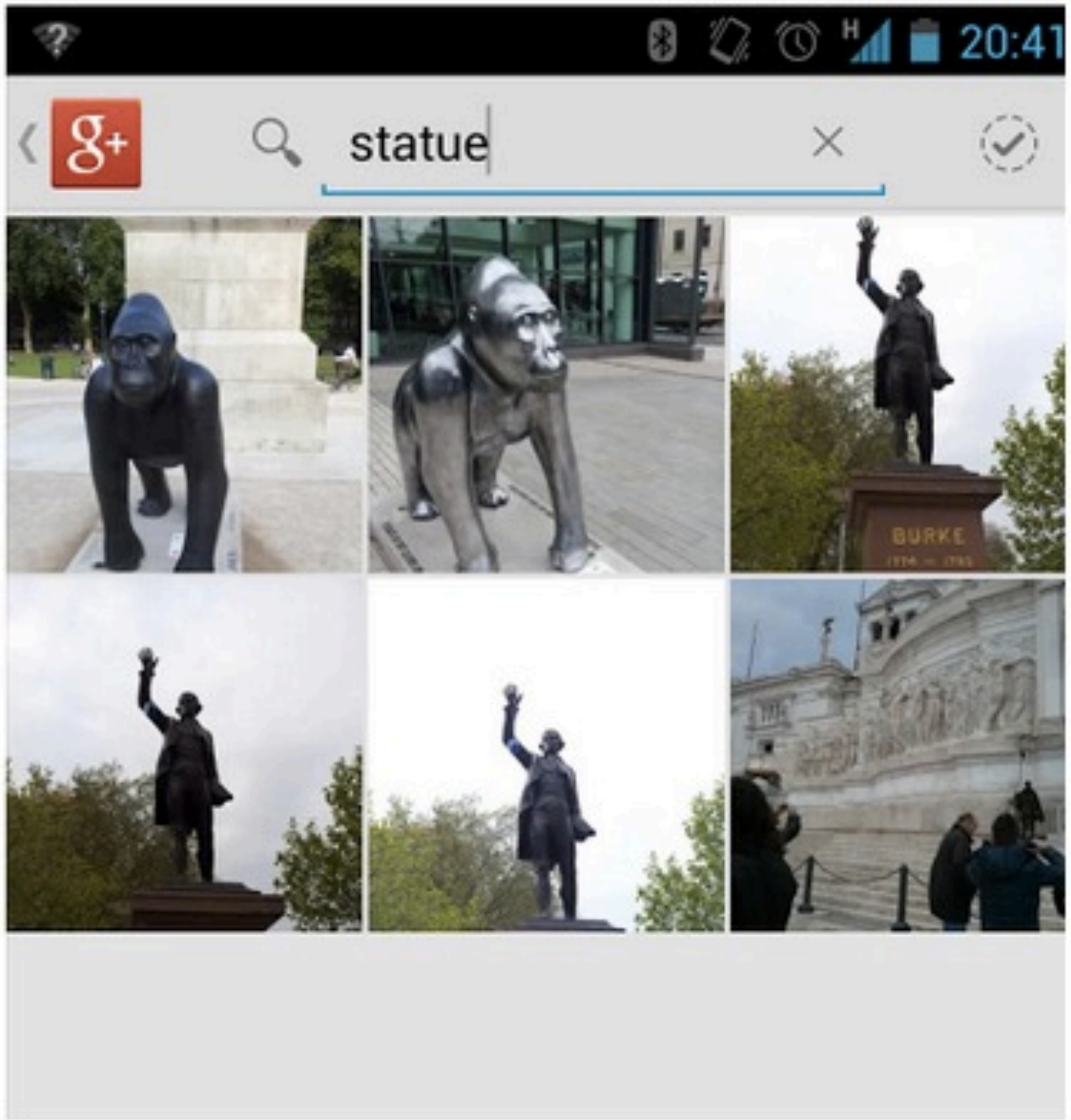
Input

Basic architecture developed by Krizhevsky, Sutskever & Hinton
(all now at Google)
Convolutional nets developed by Yann LeCun (of NYU)

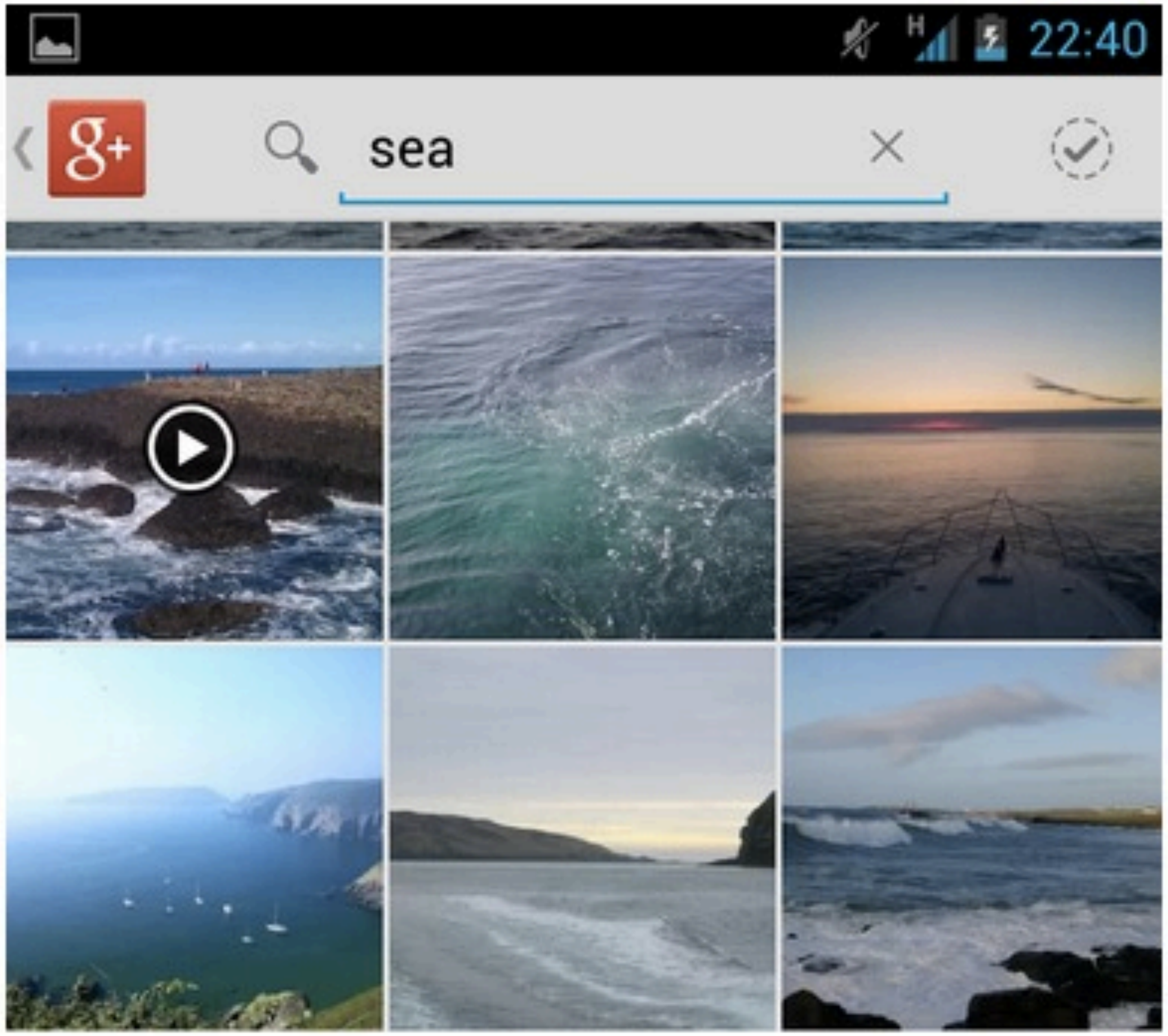
Wow.

The new Google plus photo search is a bit insane.

I didn't tag those... :)

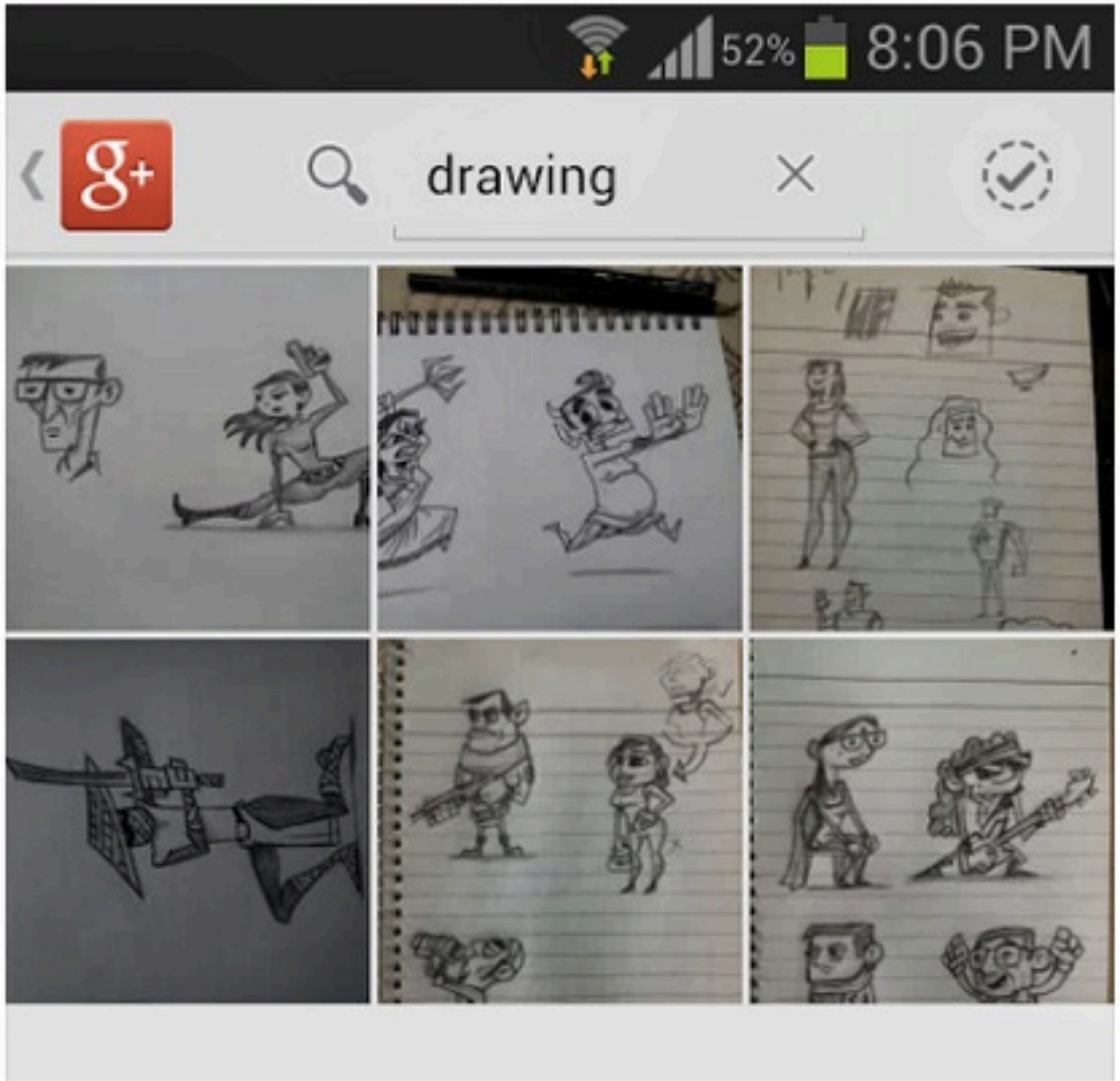


statue

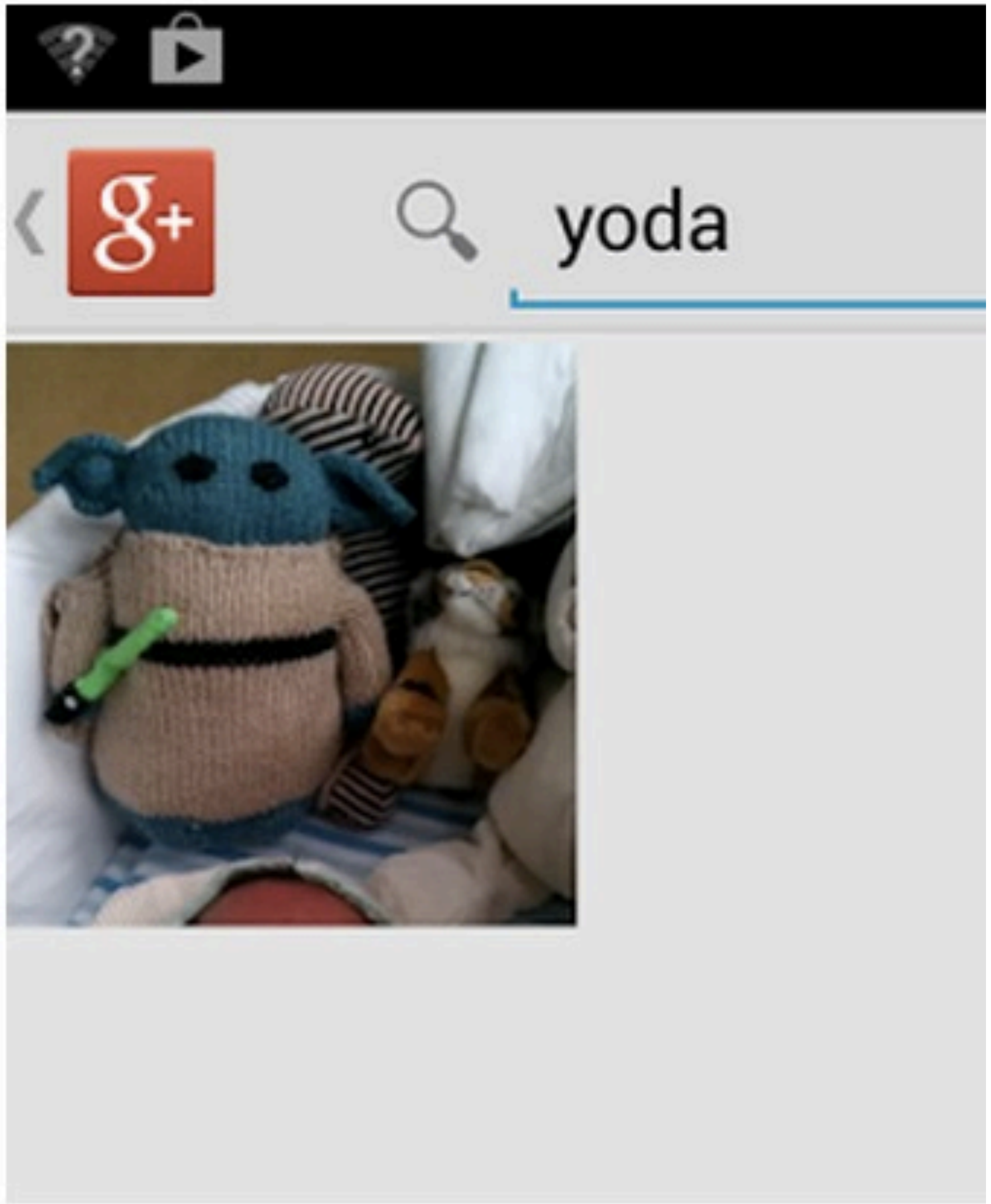


sea

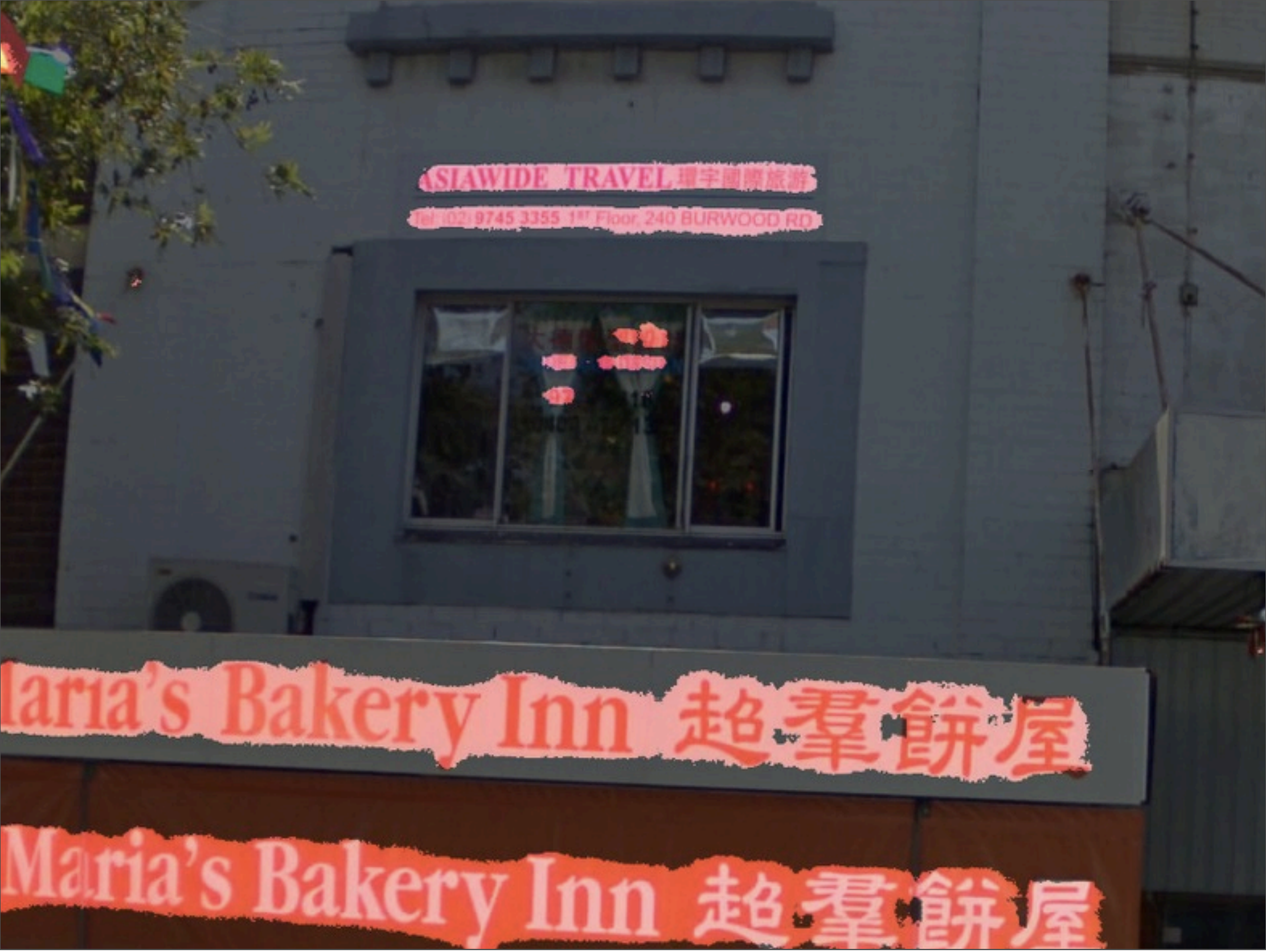
Google Plus photo search is awesome. Searched with keyword 'Drawing' to find all my scribbles at once :D

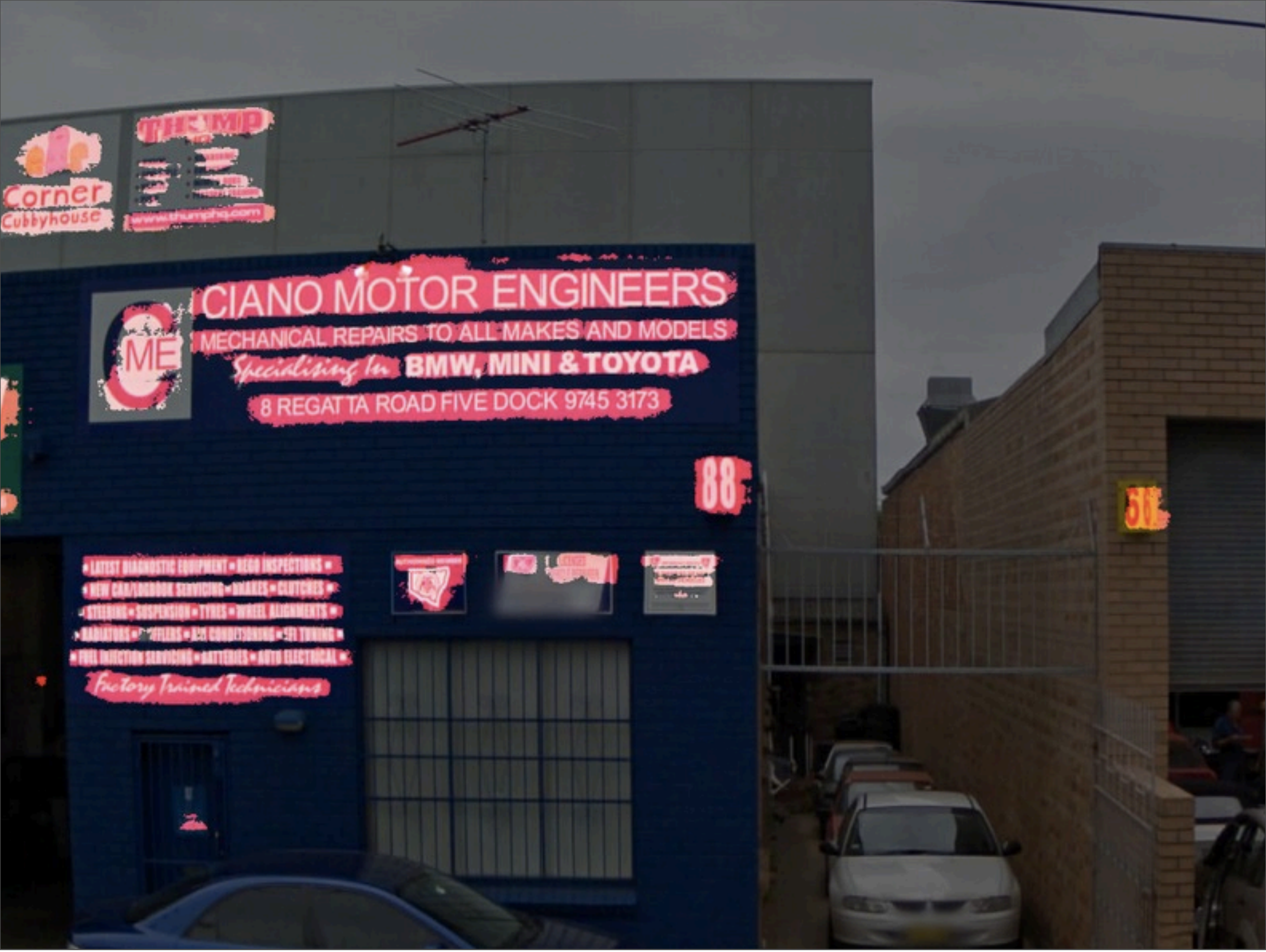


drawing



yoda


ASIAWIDE TRAVEL 環宇國際旅游
Tel (02) 9745 3355 1ST Floor, 240 BURWOOD RD
Maria's Bakery Inn 超羣餅屋
Maria's Bakery Inn 超羣餅屋


Corner
Cubbyhouse
THUMP
www.thumphq.com
CIANO MOTOR ENGINEERS
CME
MECHANICAL REPAIRS TO ALL MAKES AND MODELS
Specialising In BMW, MINI & TOYOTA
8 REGATTA ROAD FIVE DOCK 9745 3173
88
• LATEST DIAGNOSTIC EQUIPMENT • REGO INSPECTIONS •
• NEW CAR/LOGBOOK SERVICING • BRAKES • CLUTCHES •
• STEERING • SUSPENSION • TYRES • WHEEL ALIGNMENTS •
• RADIATORS • MUFFLERS • A/C CONDITIONING • EFI TUNING •
• FUEL INJECTION SERVICING • BATTERIES • AUTO ELECTRICAL •
Factory Trained Technicians
56

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

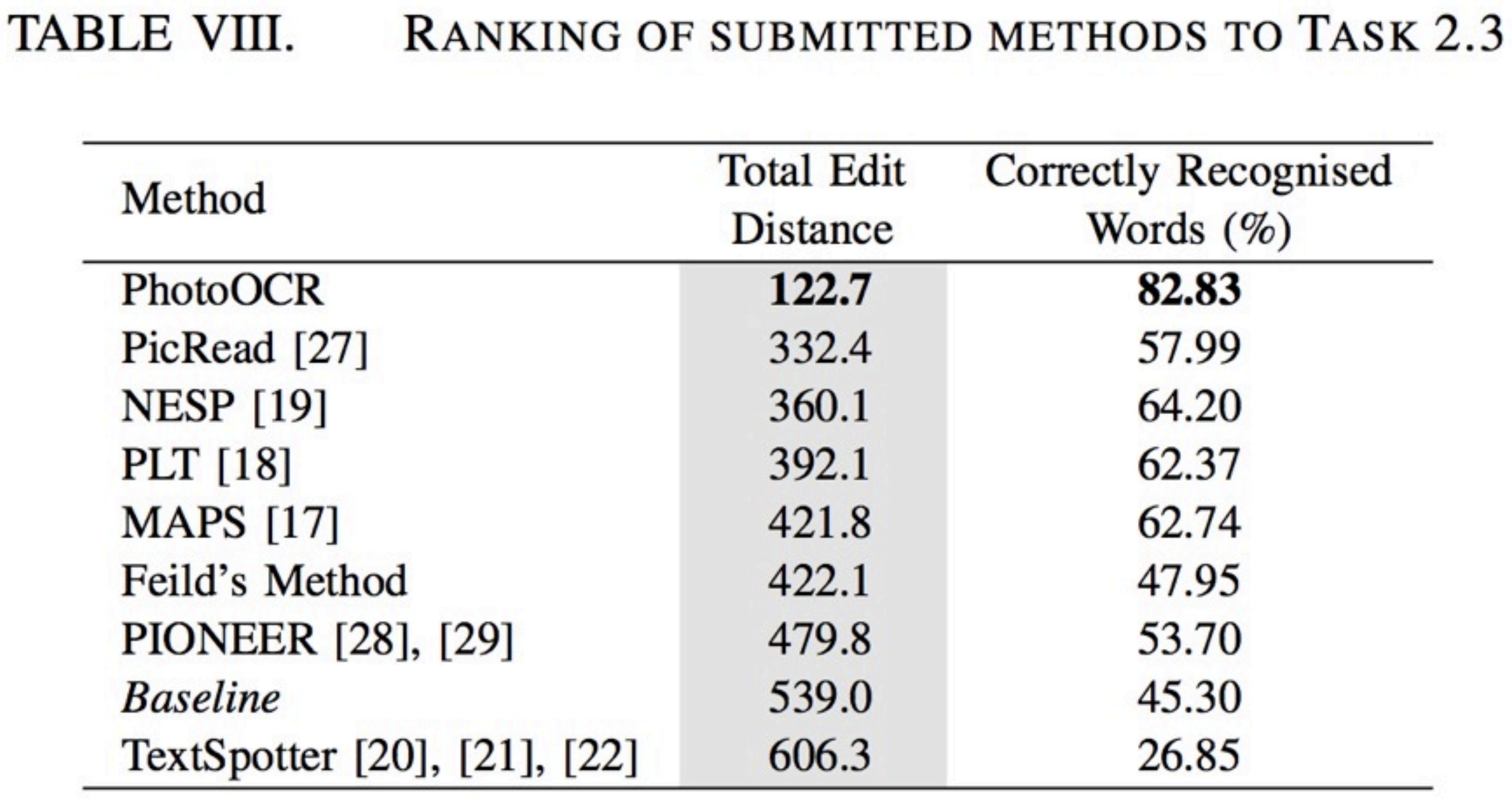


TABLE VIII. RANKING OF SUBMITTED METHODS TO TASK 2.3

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

http://dag.cvc.uab.es/icdar2013competition/

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

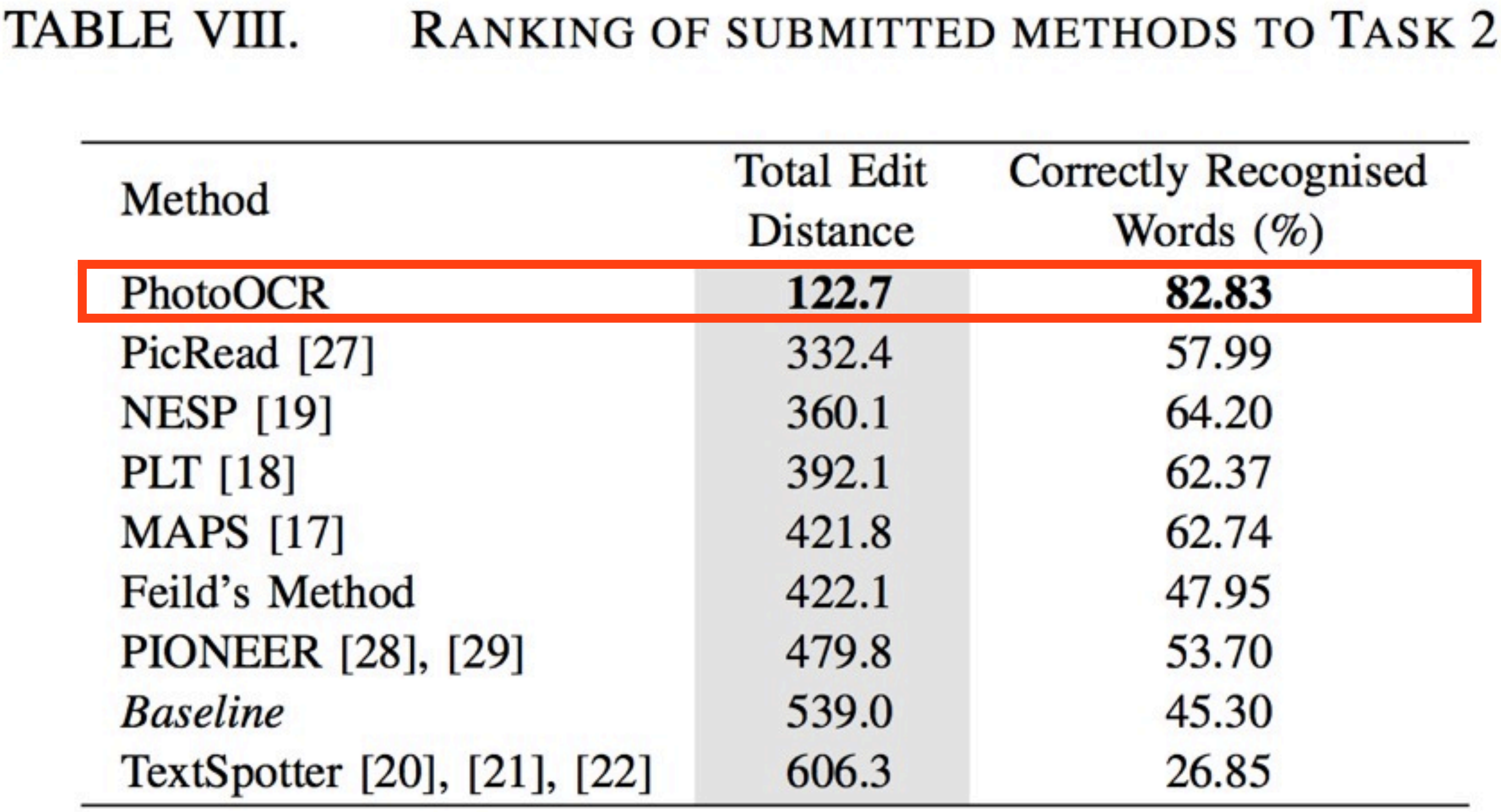
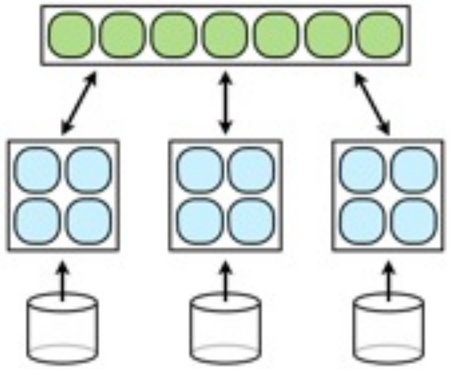


TABLE VIII. RANKING OF SUBMITTED METHODS TO TASK 2.3

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

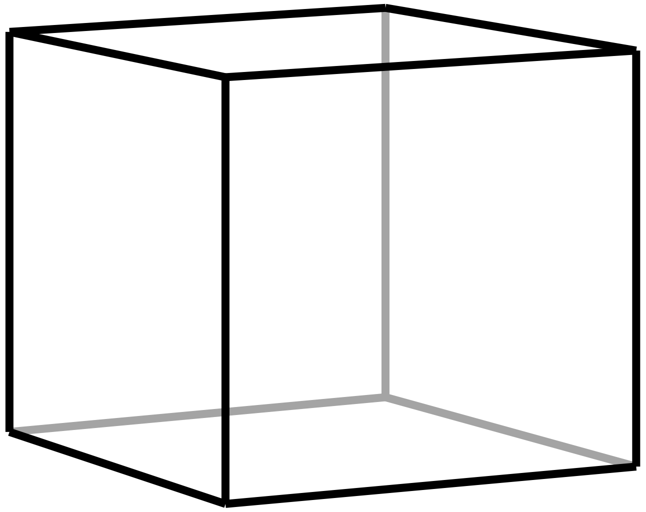
http://dag.cvc.uab.es/icdar2013competition/



How about text-related tasks?

Embeddings

~1000-D joint embedding space

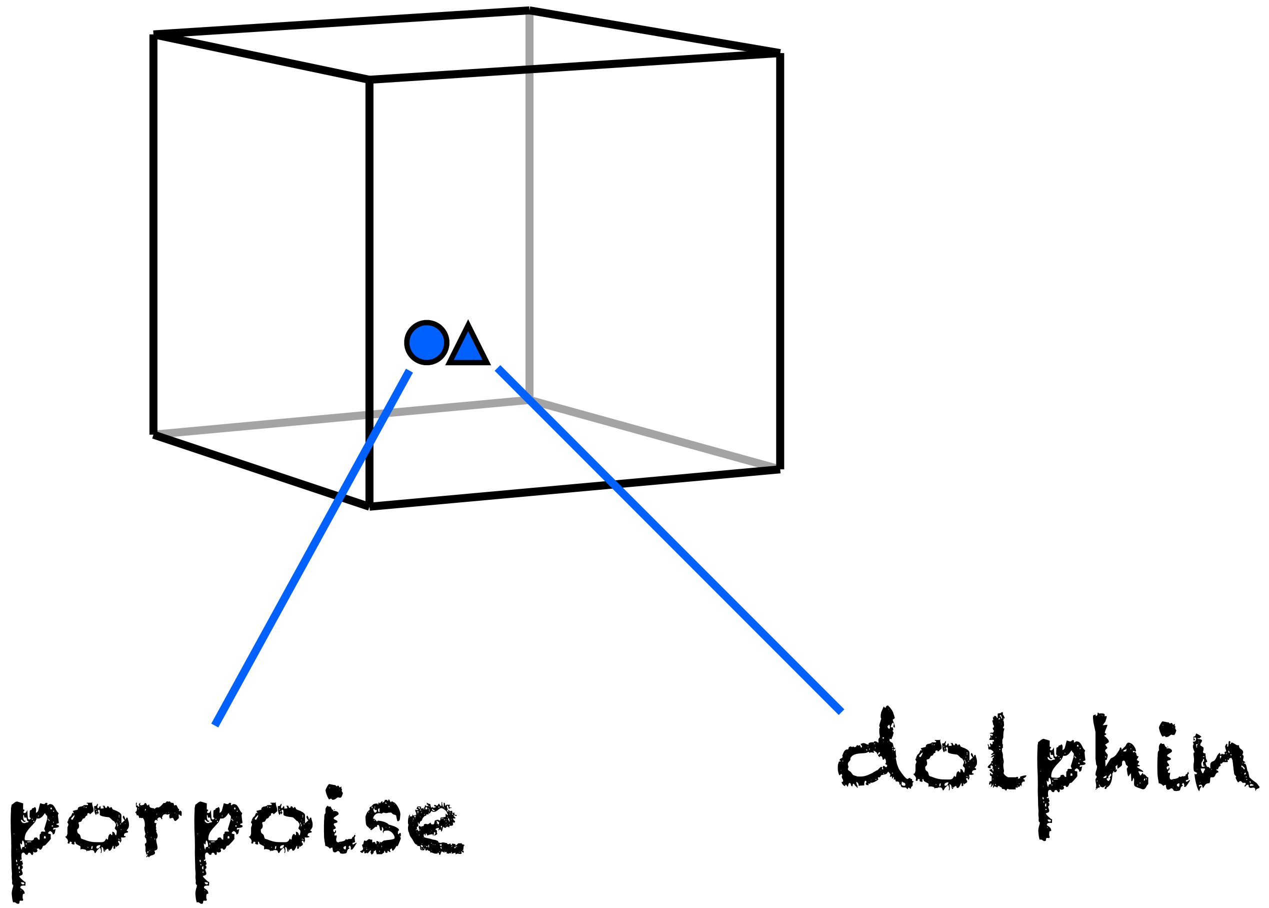


porpoise

dolphin

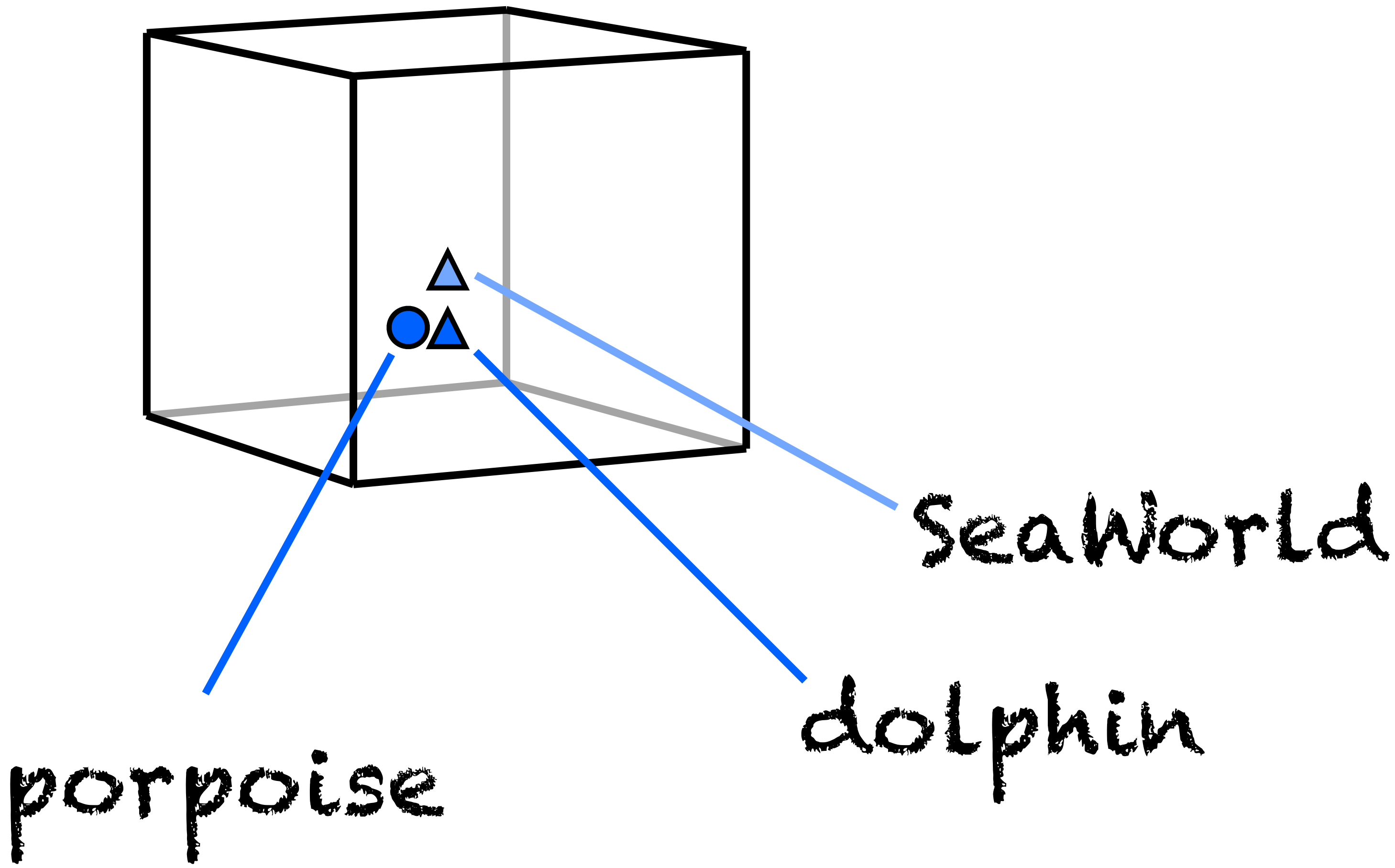
Embeddings

~1000-D joint embedding space



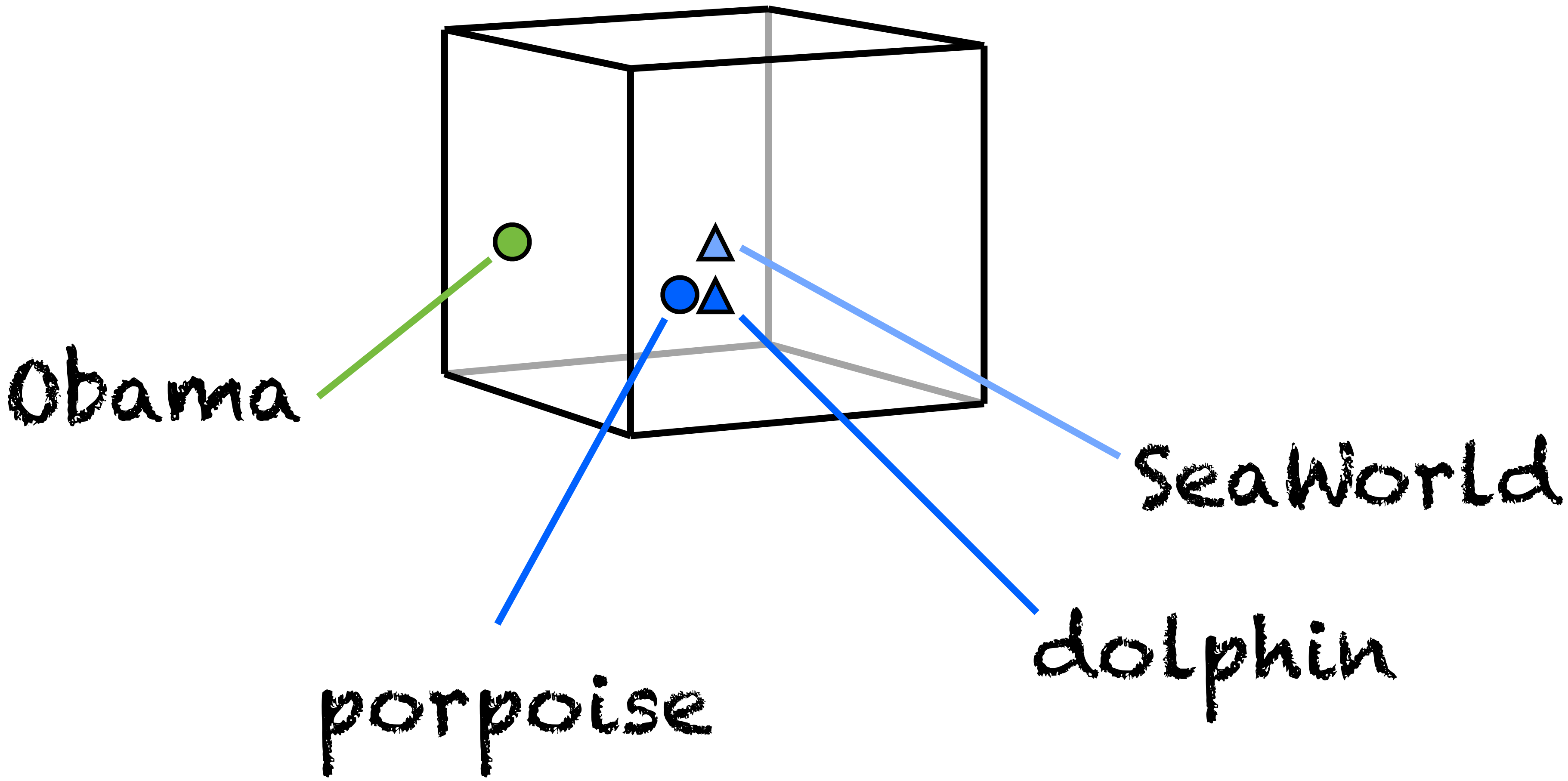
Embeddings

~1000-D joint embedding space



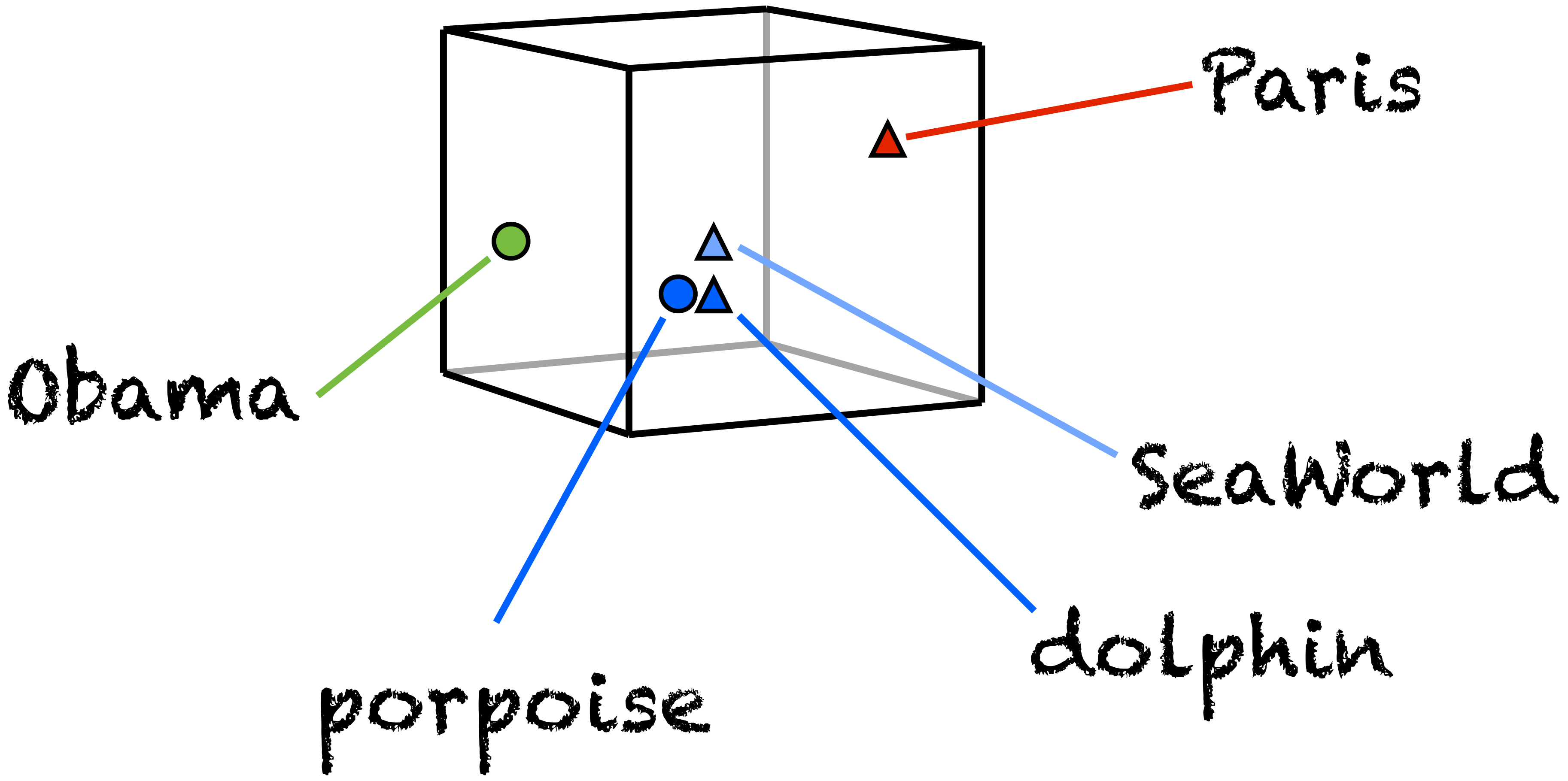
Embeddings

~1000-D joint embedding space

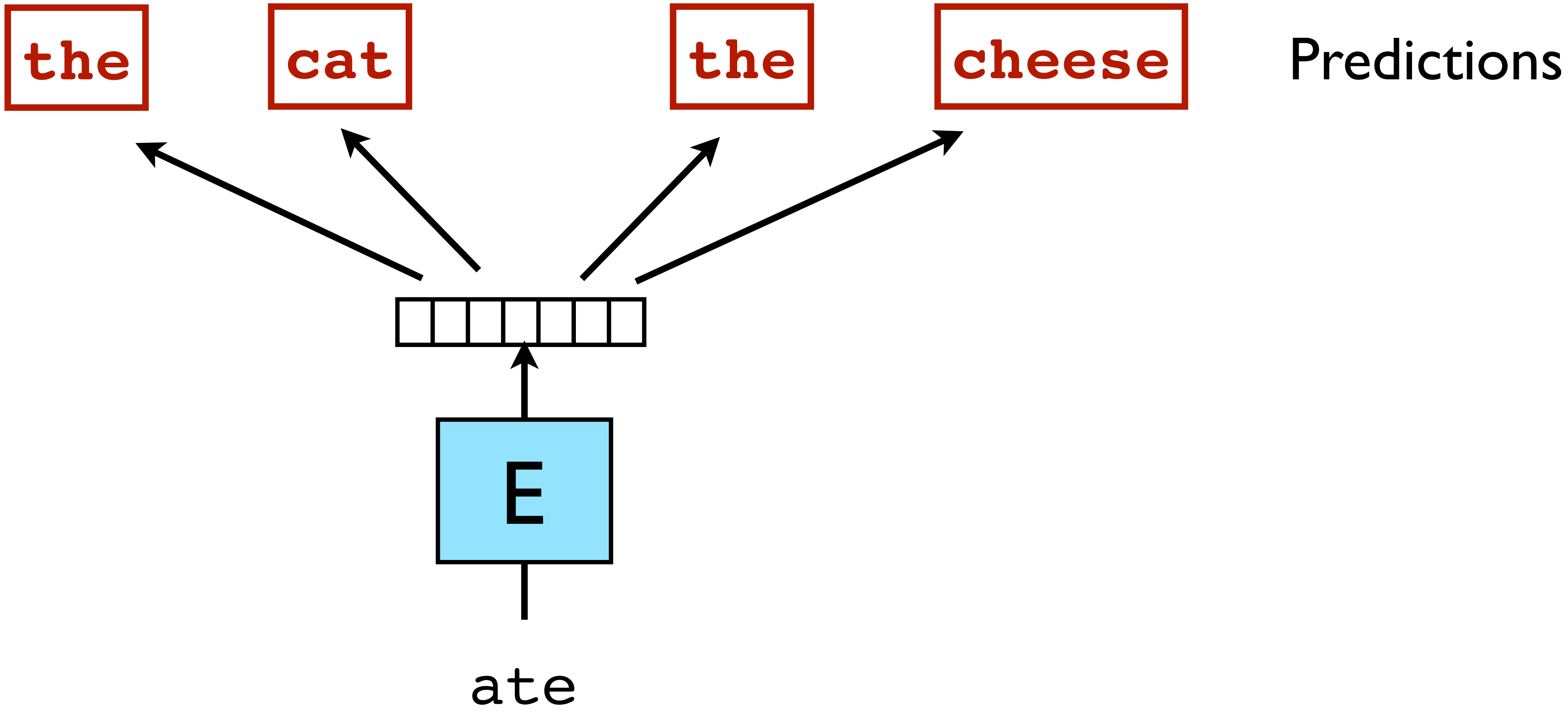


Embeddings

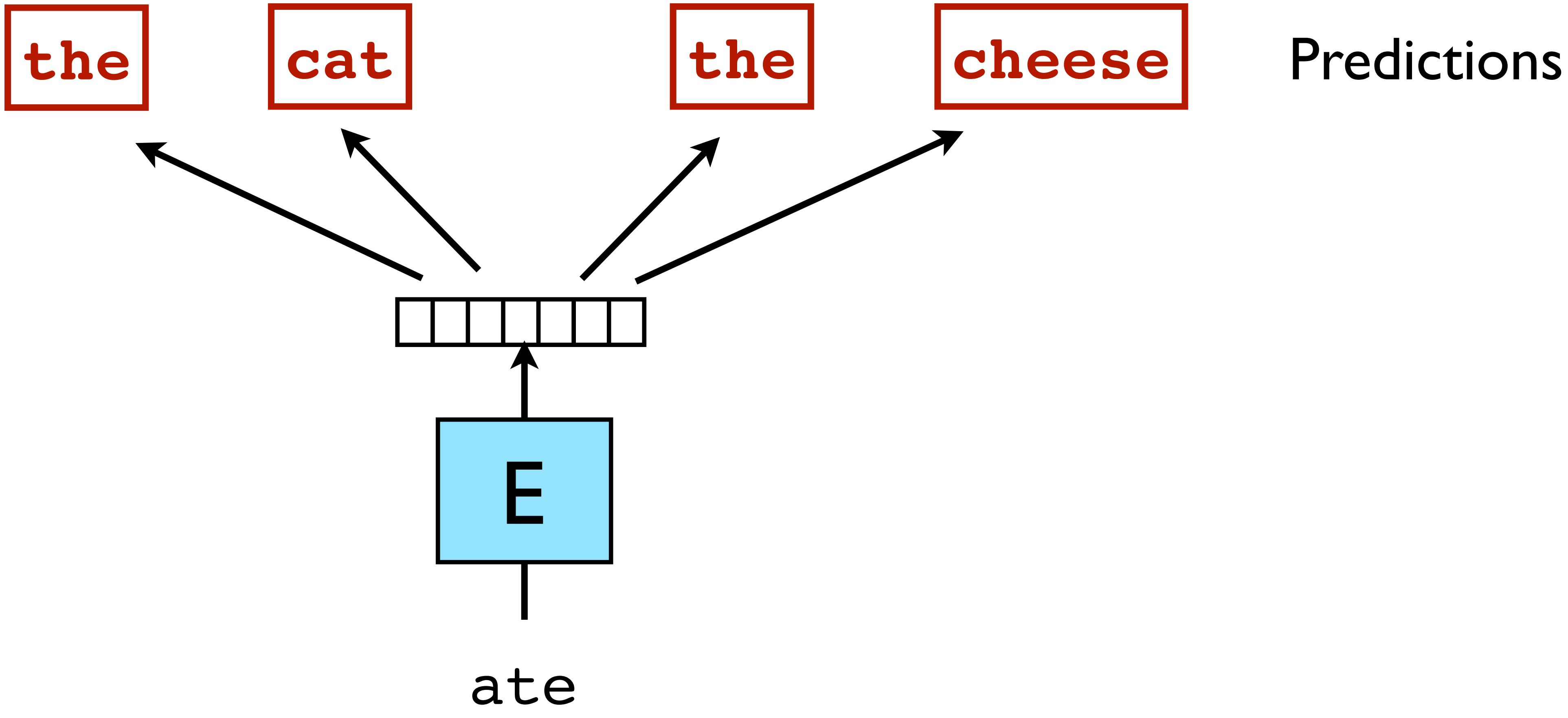
~1000-D joint embedding space



Skip-Gram Model



Skip-Gram Model

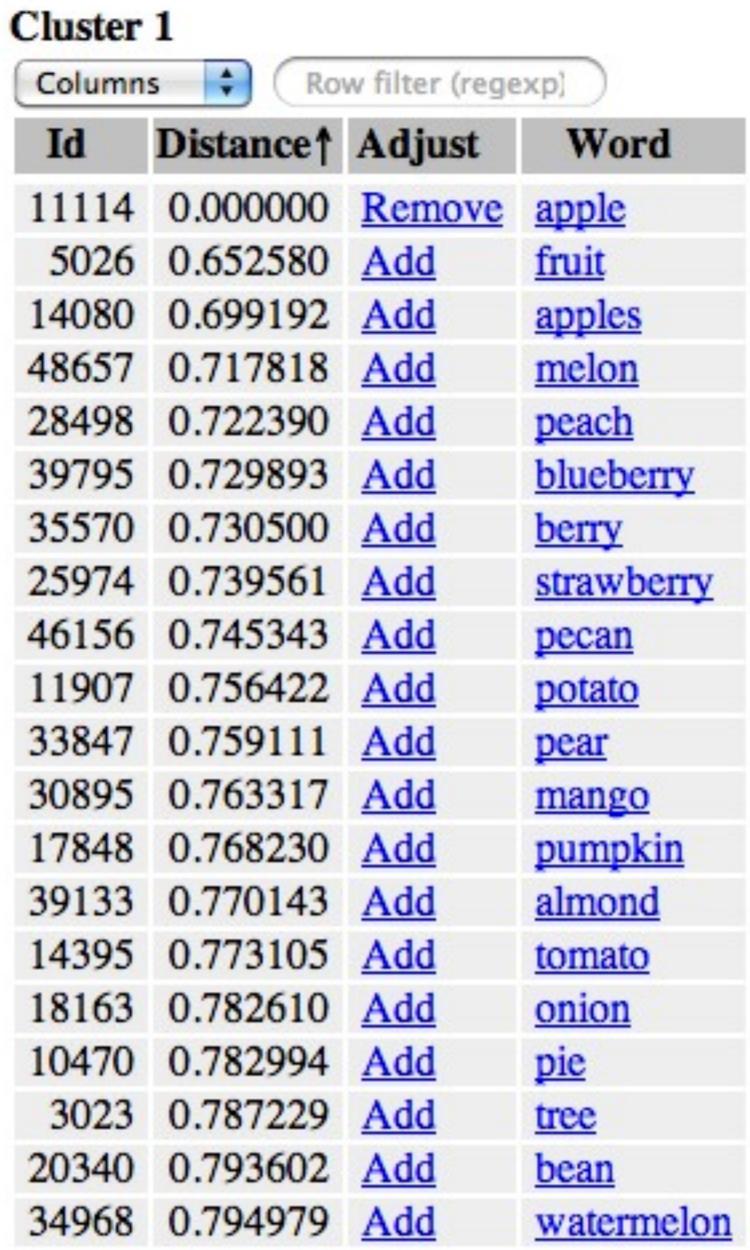


Mikolov, Chen, Corrado and Dean. *Efficient Estimation of Word Representations in Vector Space*, http://arxiv.org/abs/1301.3781

Embedding sparse tokens in an N-dimensional space

Example: 50-D embedding trained for semantic similarity

Cluster 1: apple



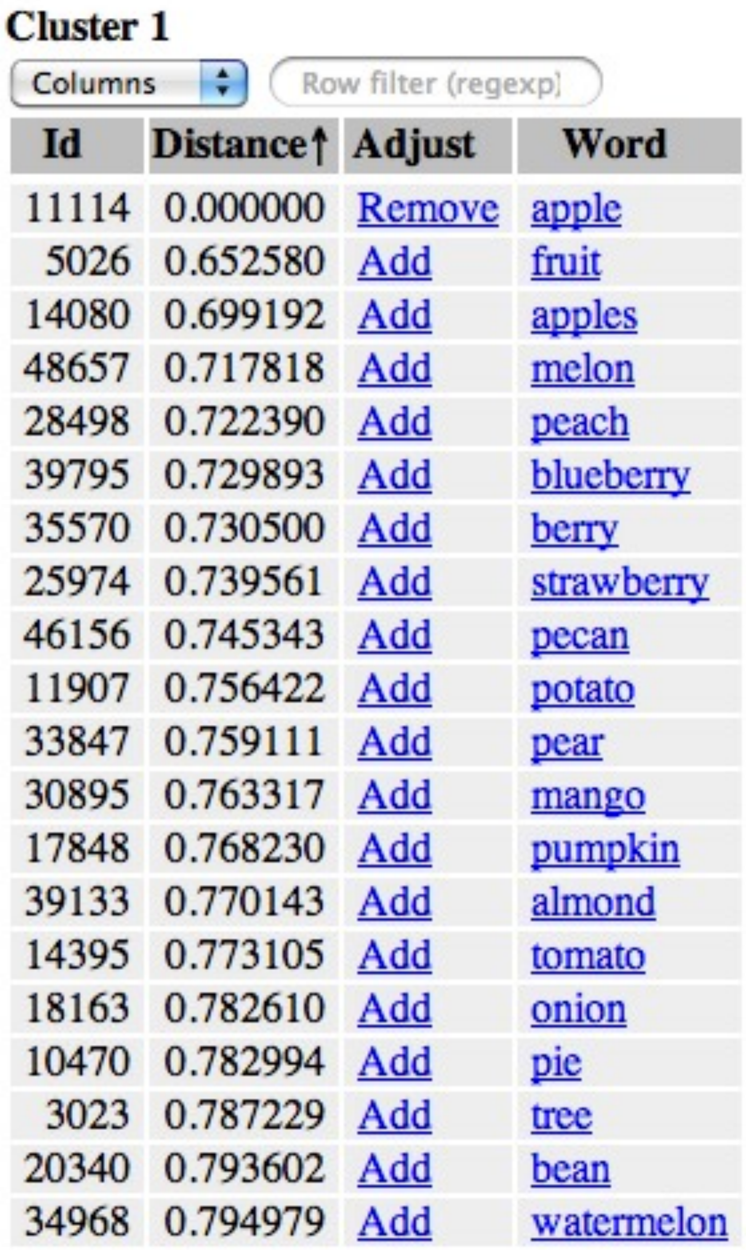
Cluster 1

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

Embedding sparse tokens in an N-dimensional space

Example: 50-D embedding trained for semantic similarity

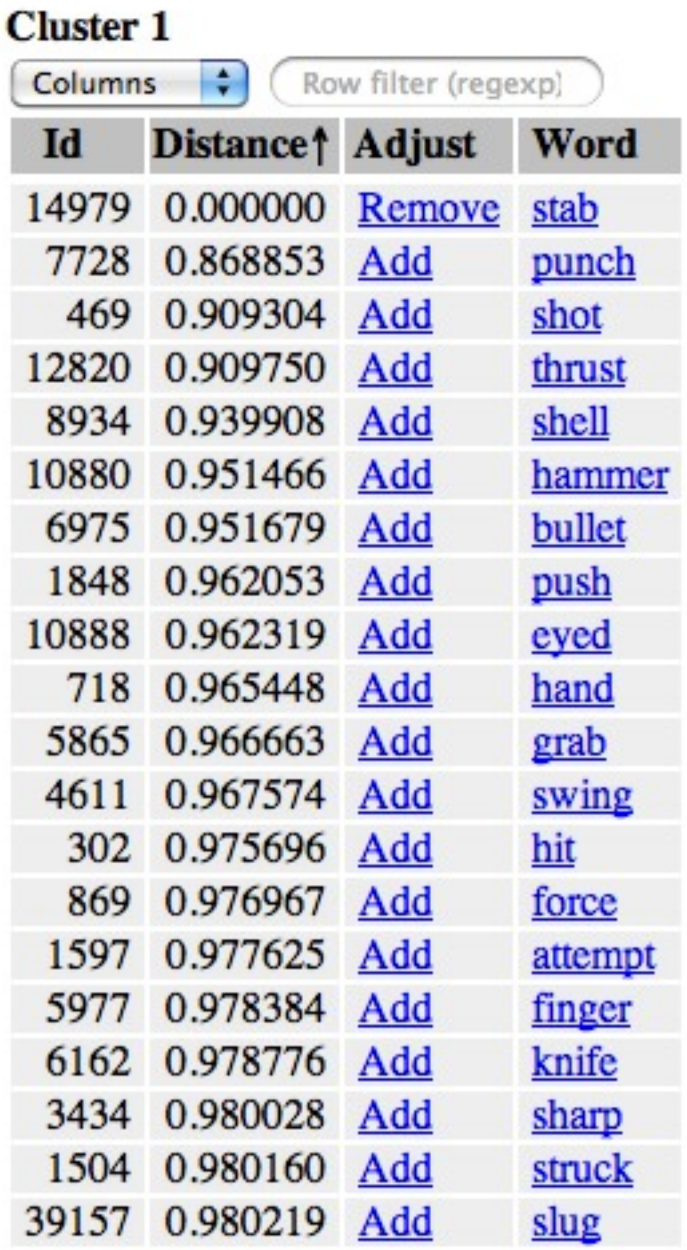
Cluster 1: apple



Cluster 1

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

Cluster 1: stab



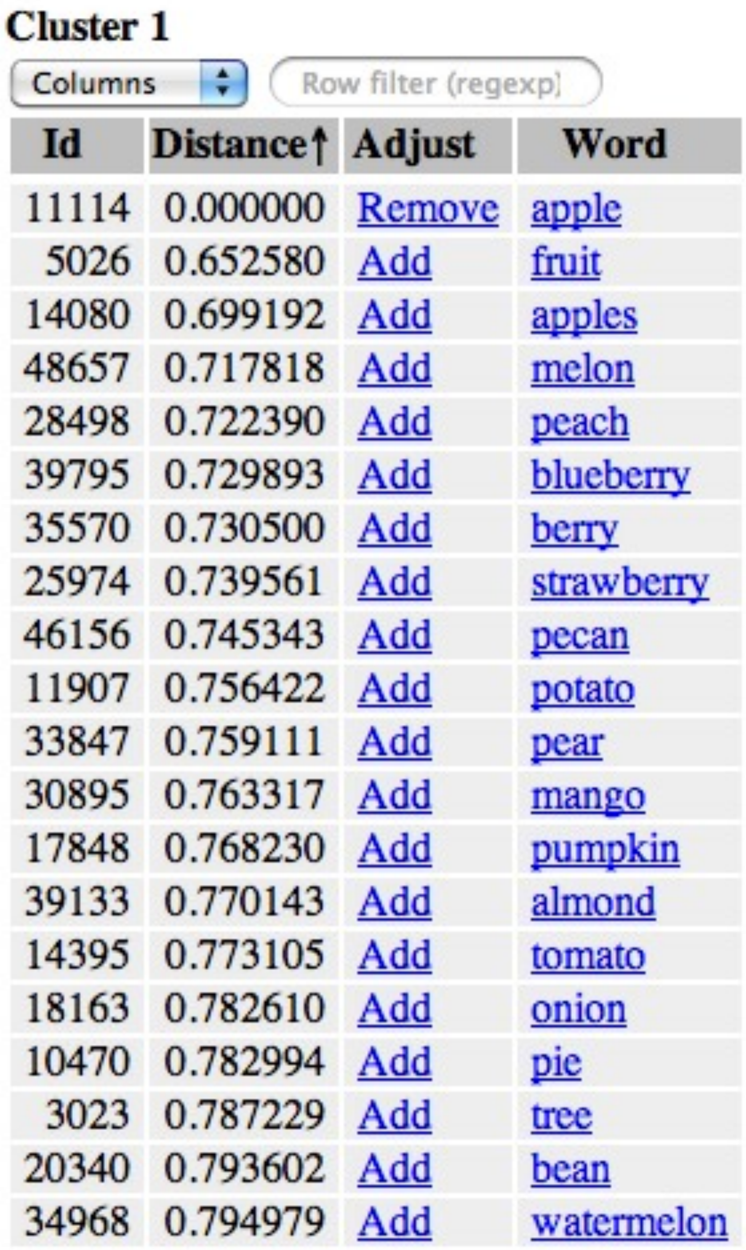
Cluster 1

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

Embedding sparse tokens in an N-dimensional space

Example: 50-D embedding trained for semantic similarity

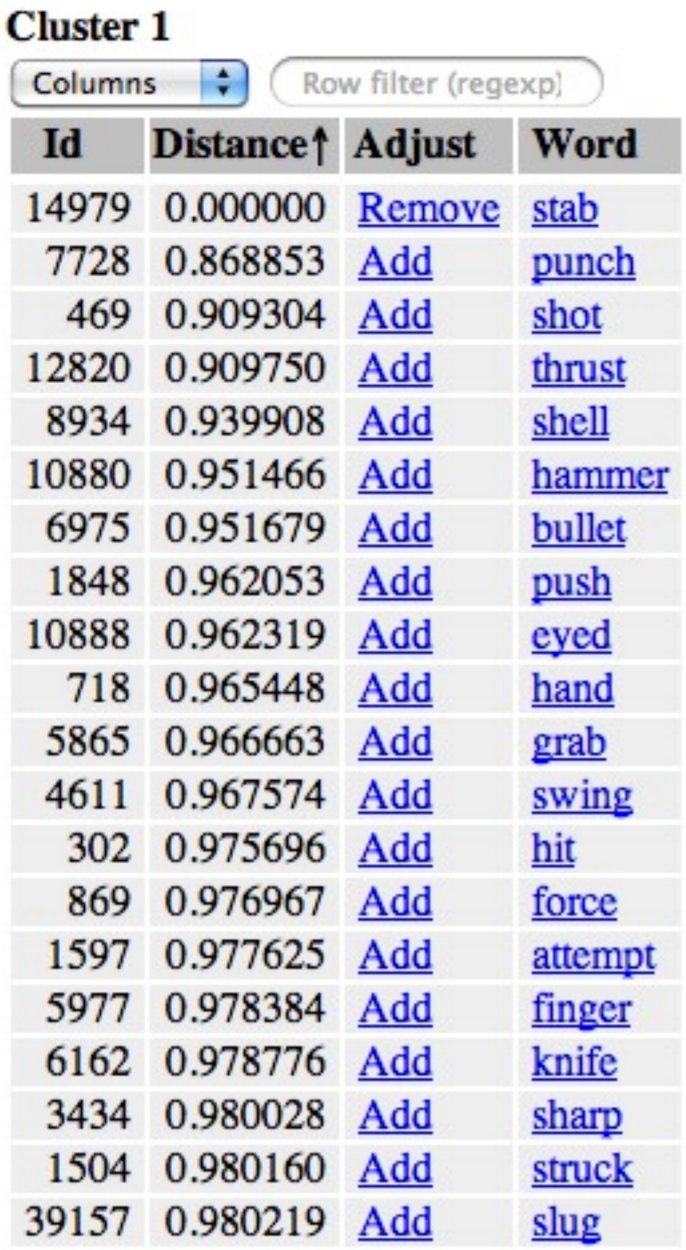
Cluster 1: apple



Cluster 1

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

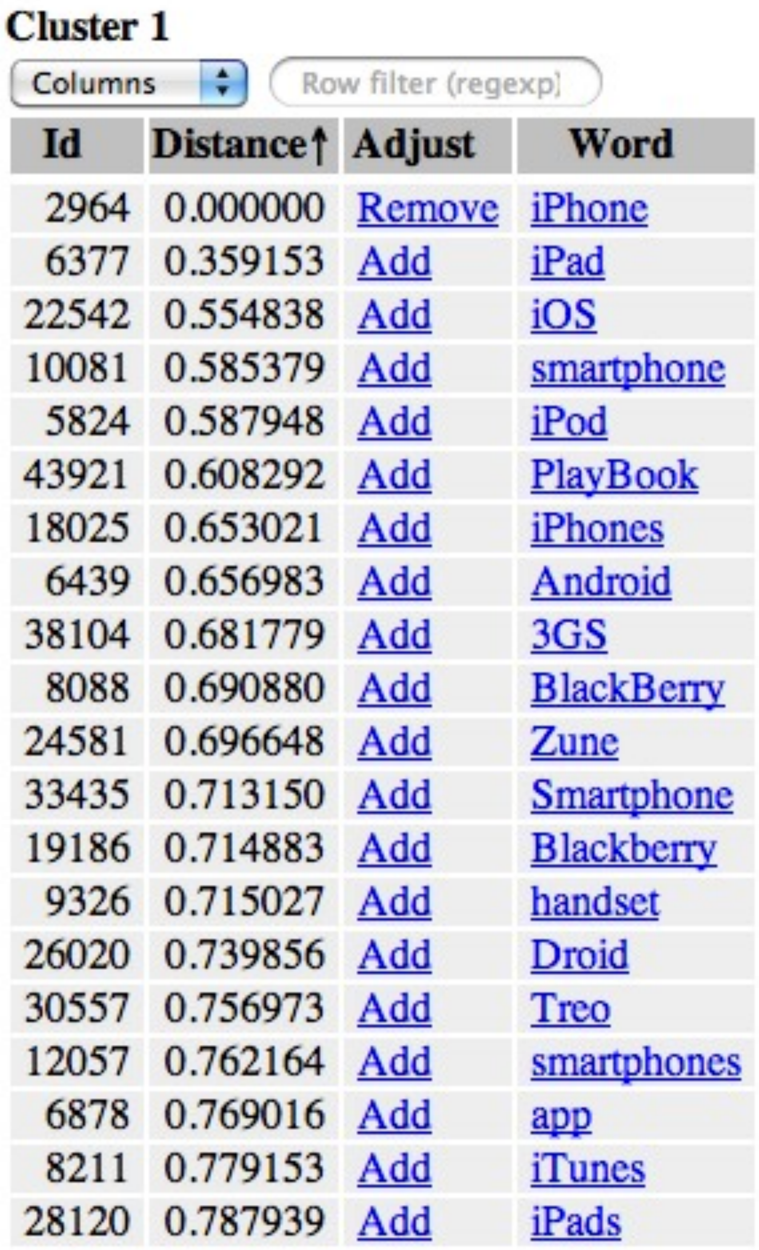
Cluster 1: stab



Cluster 1

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

Cluster 1: iPhone



Cluster 1

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

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).

E(*hotter*) - E(*hot*) + E(*big*) ≈ E(*bigger*)

E(*Rome*) - E(*Italy*) + E(*Germany*) ≈ E(*Berlin*)

Solving Analogies

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

E(*hotter*) - E(*hot*) + E(*big*) ≈ E(*bigger*)

E(*Rome*) - E(*Italy*) + E(*Germany*) ≈ E(*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).

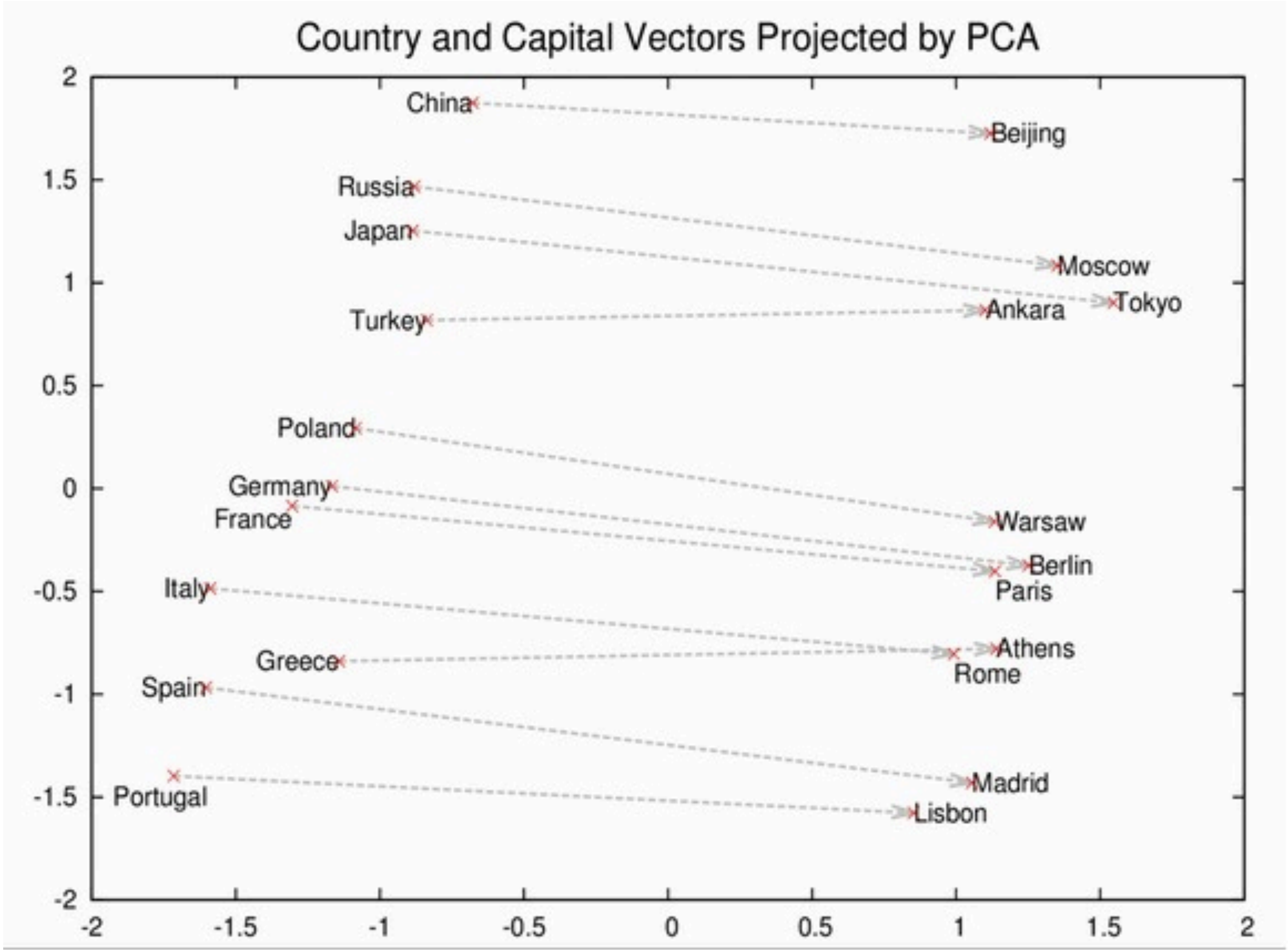
E(*hotter*) - E(*hot*) + E(*big*) ≈ E(*bigger*)

E(*Rome*) - E(*Italy*) + E(*Germany*) ≈ E(*Berlin*)

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

Details in: *Efficient Estimation of Word Representations in Vector Space*. Mikolov, Chen, Corrado and Dean. Posted on Arxiv.

Visualizing the Embedding Space



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:

- Dean & Ghemawat. *MapReduce: Simplified Data Processing on Large Clusters*, OSDI 2004.
- Chang, Dean, Ghemawat, Hsieh, Wallach, Burrows, Chandra, Fikes, & Gruber. *Bigtable: A Distributed Storage System for Structured Data*, OSDI 2006.
- 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