

# ICML 2017 Recap

Matt Dickenson  
Uber - Map Creation  
August 17, 2017



# Themes



Modeling



Scalability



Human-Computer Interaction



# Modeling

 Recurrent

 Pixelwise

 Vision





# Modeling



**Recurrent**



Pixelwise



Vision





# The Statistical Recurrent Unit

Junier B. Olivia, Barnabás Póczos, Jeff Schneider

- Problem: gated recurrent units require learning where we are in a sequence, when to open the gate, and the feature itself
- Innovation: Use exponentially weighted moving averages to preserve information about both temporality and scale
- Keep multiple moving averages to distinguish the type of data seen in the past
- Examples: moving MNIST, music (x2), climate, NBA

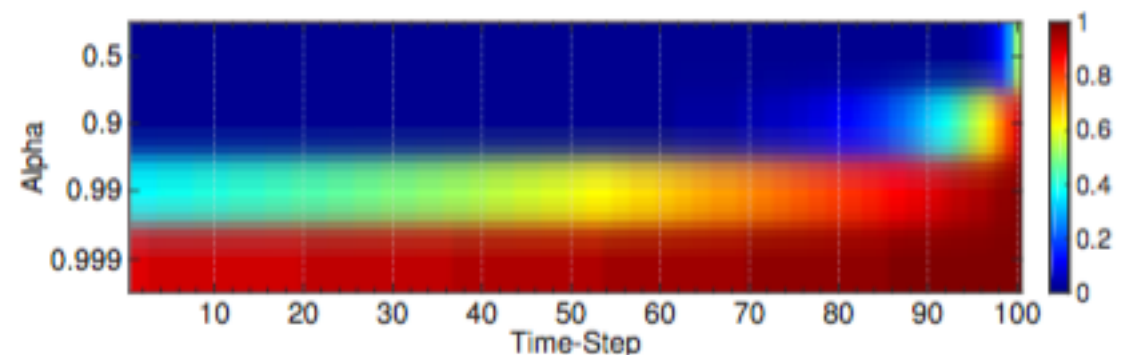
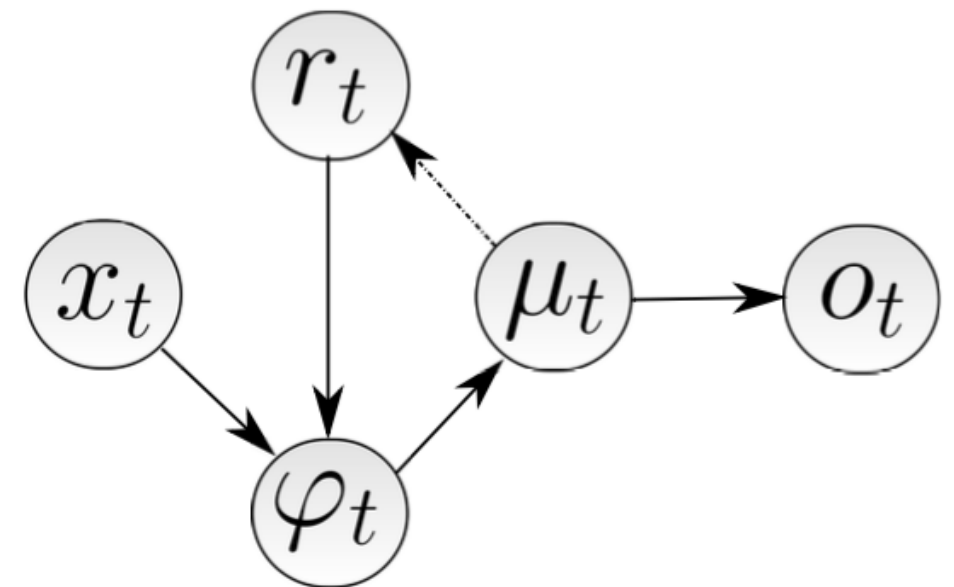




# The Statistical Recurrent Unit

Junier B. Olivia, Barnabás Póczos, Jeff Schneider

- $x_t$ : current data point
- $r_t$ : summary of previous data
- $\phi_t$ : statistics
- $\mu_t$ : moving averages
- $o_t$ : output





# Modeling



Recurrent



**Pixelwise**



Vision



# Latent Variable PixelCNNs for Natural Image Modeling

Alexander Kolesnikov, Christoph H. Lampert

- Original PixelCNN focuses on local rather than global structure
- Approach 1: Introduce latent variables in the form of grayscale images or down sampled images (equivalent to superresolution)

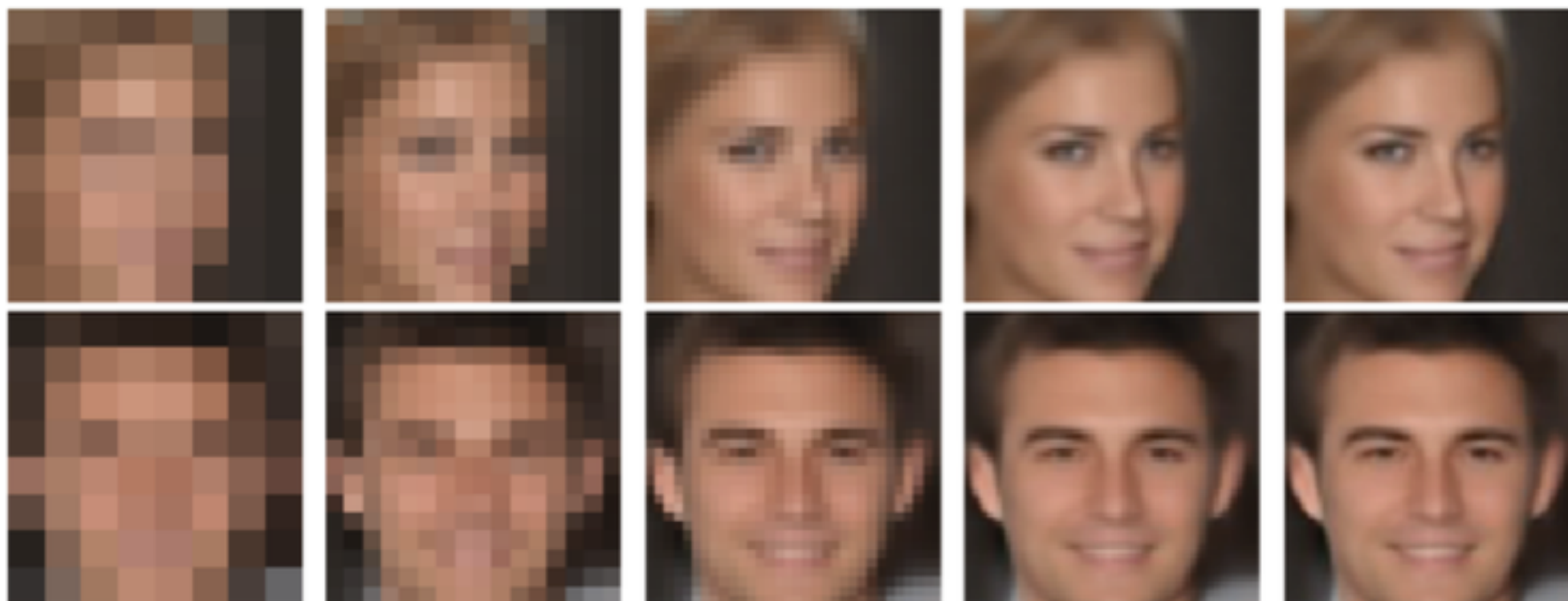




# Latent Variable PixelCNNs for Natural Image Modeling

Alexander Kolesnikov, Christoph H. Lampert

- Approach 2: Introduce latent variables in the form of downsampled images (equivalent to super-resolution)





# Autoregressive Modeling

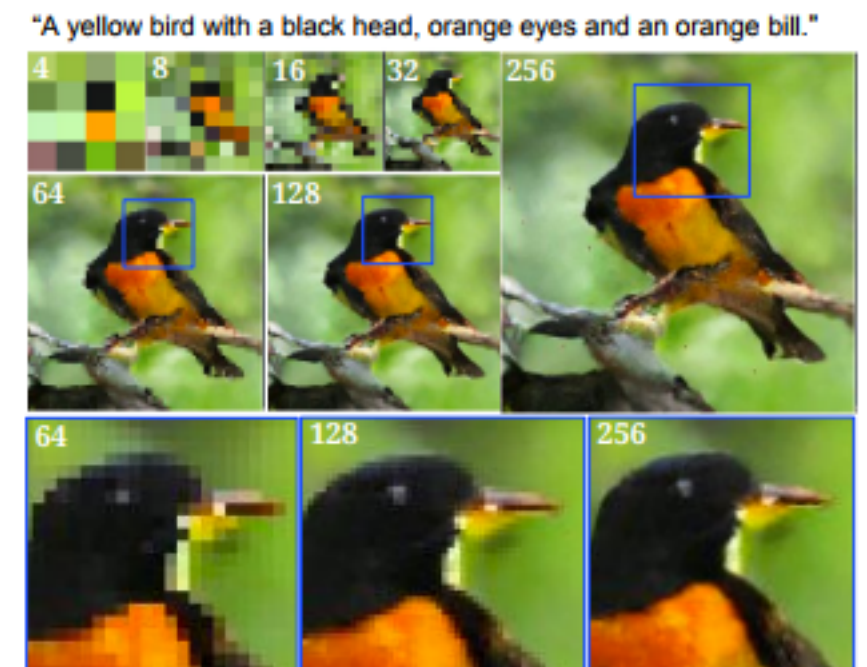
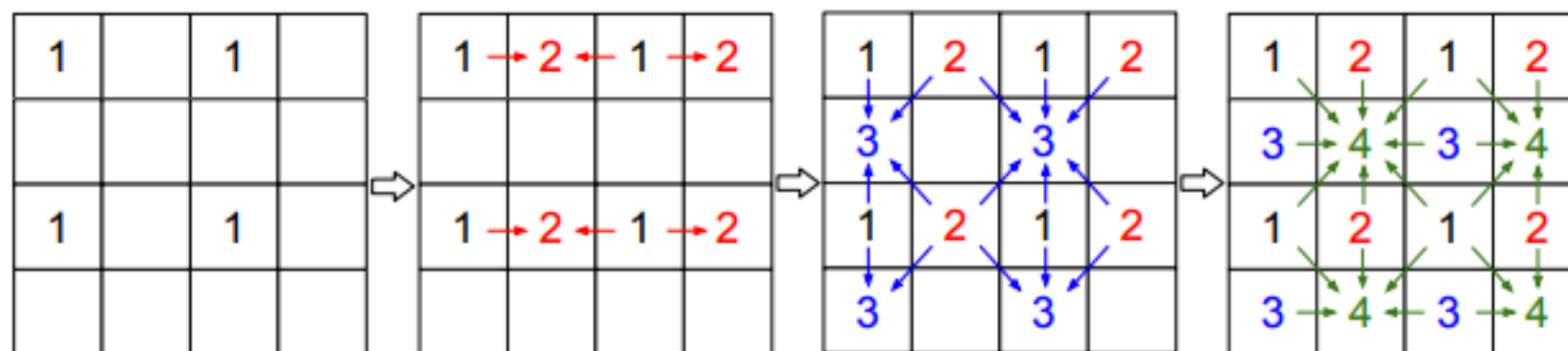
- Guess the next letter:
  - `reve {n|r}`
- Guess again:
  - `annual reve {n|r}`

# Parallel Multiscale Autoregressive Density Estimation



Scott Reed et al.

- Instead of sampling the whole image in sequence, downsample and run PixelCNN on every  $n$ th pixel in parallel
- Can initialize from latent variables (captions, keypoints)





# Modeling



Recurrent



Pixelwise



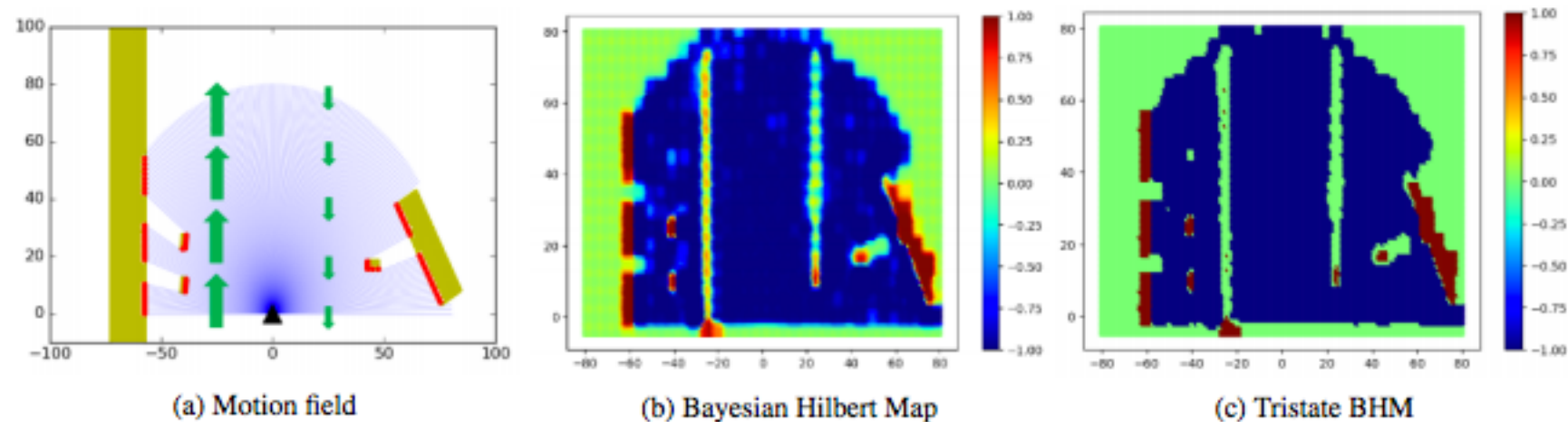
**Vision**



# Bayesian Hilbert Maps for Continuous Occupancy Mapping in Dynamic Environments

Ransalu Senanayake, Fabio Ramos

- Old way: occupancy grid
- Newer way: Gaussian process in continuous space, (slow)
- Newest approach: Bayesian Hilbert maps





# Dynamic Active Vision Sensor (DAViS)

Jonathan Binas, Daniel Neil, Shih-Chii Liu, Tobi Delbruck

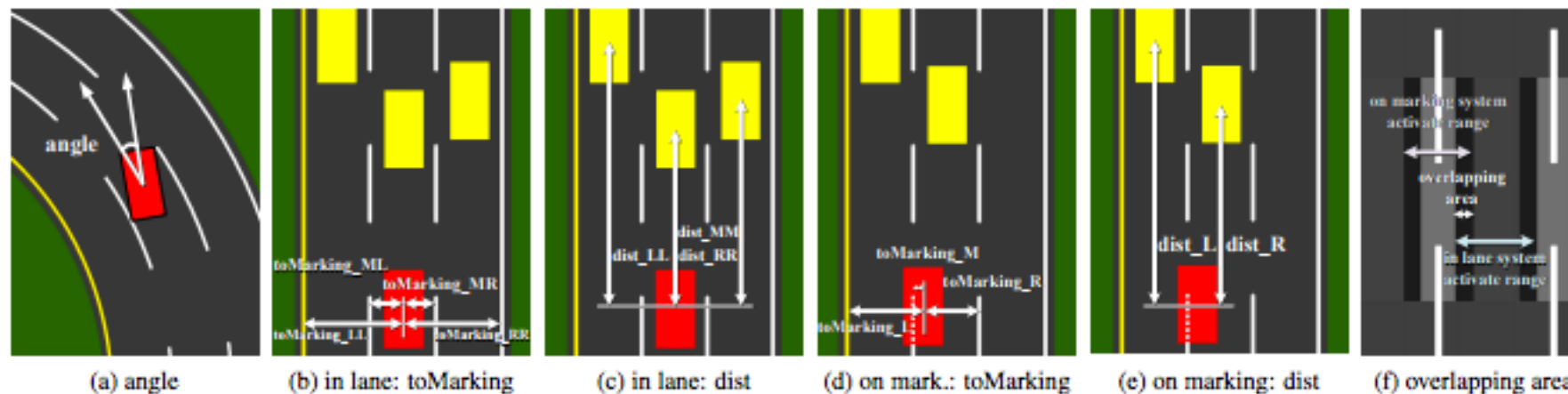
- Cameras have well-known issues
  - Motion blur
  - Fixed temporal resolution (frame rate)
  - High data rates
  - Limited dynamic range
- Add a brightness change sensor to overcome this
- <https://inilabs.com/products/dynamic-and-active-pixel-vision-sensor/>



# Learning Affordance for Direct Perception in Autonomous Driving

Chenyi Chen et al.

- How fast should you drive if you haven't seen a speed limit sign in a while?
- What are the things we actually care about in a vehicle?
  - Distance to nearest solid objects
  - Drivable surfaces

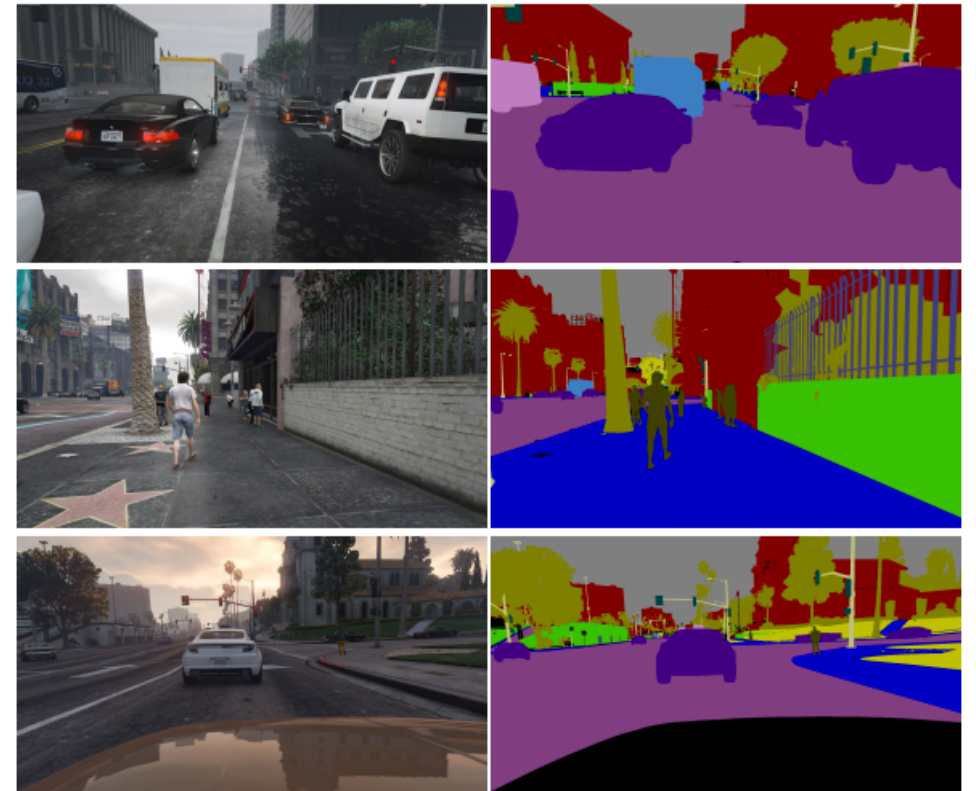




# Crash To Not Crash: Playing Video Games To Predict Vehicle Collisions

Kangwook Lee, Hoon Kim, Changho Suh

- GTA V simulator for crashes
- 10k scenes, 10 frames per scene
- Half crashes (artificial imbalance)
- Style transfer to KITTI
- CNN is better than a detection model because it accounts for clues like wheel angle





# Scalability



Learning to Scale



Hardware



Compression



Dimensionality Reduction





# Scalability



## **Learning to Scale**



Hardware



Compression



Dimensionality Reduction





# Memory-Efficient Convolution

Minsik Cho, Daniel Brand

- Convolution tends to be one of the most expensive layers in terms of memory
- Convolution is an unusual memory access pattern and relatively simple computation

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 2 & 1 & 1 & 2 & 0 \\ 0 & 2 & 0 & 1 & 1 & 0 & 0 \\ 0 & 2 & 0 & 1 & 2 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 4 & 6 & 3 & 5 & 4 \\ 2 & 6 & 2 & 4 & 4 \\ 1 & 5 & 3 & 4 & 4 \\ 2 & 4 & 3 & 3 & 4 \\ 0 & 2 & 2 & 4 & 3 \end{bmatrix}$$

(a) direct convolution

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 0 & 2 & 2 & 1 & 2 \\ 0 & 0 & 0 & 2 & 1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 0 & 1 & 2 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}_{25 \times 1} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}_{1 \times 1} = \begin{bmatrix} 4 & 6 & 3 & 5 & 4 \\ 2 & 6 & 2 & 4 & 4 \\ 1 & 5 & 3 & 4 & 4 \\ 2 & 4 & 3 & 3 & 4 \\ 0 & 2 & 2 & 4 & 3 \end{bmatrix}$$

(b) im2col-based convolution with lowered matrix



# Memory-Efficient Convolution

Minsik Cho, Daniel Brand

- Using BLAS friendly memory layout is more efficient, which is applicable to a large range of devices and any convolutional configuration
- BUT the intermediate representation has higher memory redundancy. We get computational speed up by abusing the memory  $\backslash(\text{ツ})/_$

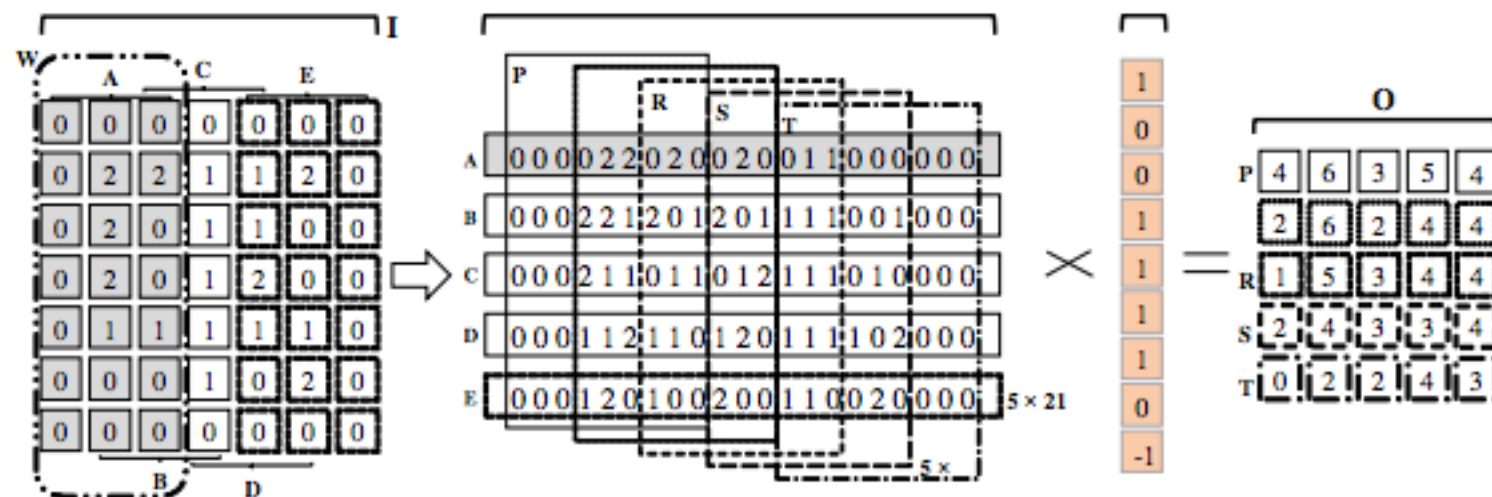


Figure 2. MEC example for the same problem in Fig. 1



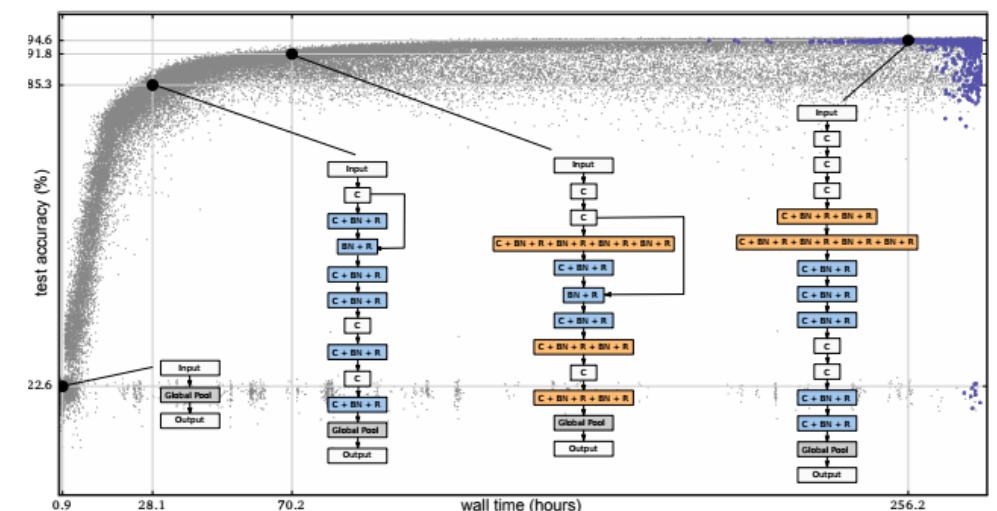


# Large-Scale Evolution of Image Classifiers

Esteban Real et al.

- There exist orders of magnitude more problems that could benefit from deep learning than there exist developers who can create them
- Main goal: minimize human participation; start from scratch, use known components, produce fully trained models
- Tournament selection over 1000 candidates; copy + mutate winner at each stage

- ALTER-LEARNING-RATE (sampling details below).
- IDENTITY (effectively means “keep training”).
- RESET-WEIGHTS (sampled as in He et al. (2015), for example).
- INSERT-CONVOLUTION (inserts a convolution at a random location in the “convolutional backbone”, as in Figure 1. The inserted convolution has  $3 \times 3$  filters, strides of 1 or 2 at random, number of channels same as input. May apply batch-normalization and ReLU activation or none at random).
- REMOVE-CONVOLUTION.
- ALTER-STRIDE (only powers of 2 are allowed).
- ALTER-NUMBER-OF-CHANNELS (of random conv.).
- FILTER-SIZE (horizontal or vertical at random, on random convolution, odd values only).
- INSERT-ONE-TO-ONE (inserts a one-to-one/identity connection, analogous to insert-convolution mutation).
- ADD-SKIP (identity between random layers).
- REMOVE-SKIP (removes random skip).





# Large-Scale Evolution of Image Classifiers

Esteban Real et al.

- Initialize with trivial models (fully connected)
- Many 10s of thousands of networks trained in 10 days across 250 workers
- This shows that it's possible to version and manage many models
- Two metaparameters mattered: population size and training steps. Both are monotonically increasing, so do the best you can
- The models arrived at were pretty simple, why? It's possible that when you're in an ok spot it's hard to get out of it (just like in nature)



# Scalability



Learning to Scale



**Hardware**



Compression



Dimensionality Reduction



# Device Placement with Reinforcement Learning

Azalia Mirhoseini et al.

- Suppose we have a model that can be deployed in parallel on many devices. How can we decide where to deploy optimally?
- Currently this is done heuristically (where we have the most bandwidth, memory, etc). Can we replace it with RL?
- Input: neural architecture graph
- Goal: assign operations in a neural model to devices to optimize a reward function (e.g. runtime). The size of the policy is very large ( $\text{num\_operations}^{\text{num\_devices}}$ )
- The model learns to optimize resource usage (e.g. if given the option of 4 GPUs and one CPU, it learns to use only two GPUs because of I/O bottleneck)



# Deep Tensor Convolution on Multicores

David Budden et al.

- $|\text{CPUs}| \gg |\text{GPUs}|$ .
- View a one-dimensional convolution as a polynomial multiplication:
  - $\mathbf{s} = \mathbf{g} * \mathbf{d}$
  - $s(x) = g(x)d(x)$
  - $s_i = \sum_k g_{i-k} d_k$
  - $\mathbf{s} = \mathbf{A}[(\mathbf{Cg}) \odot (\mathbf{Bd})]$



# Deep Tensor Convolution on Multicores

David Budden et al.

- $|\text{CPUs}| \gg |\text{GPUs}|$ .
- View a one-dimensional convolution as a polynomial multiplication:

- $\mathbf{s} = \mathbf{g} * \mathbf{d}$

- $s(x) = g(x)d(x)$  ← polynomials

- $s_i = \sum_k g_{i-k} d_k$  ← coefficients

- $\mathbf{s} = \mathbf{A}[(\mathbf{Cg}) \odot (\mathbf{Bd})]$  ← kernel  
← data tile  
← inverse transform



# Deep Tensor Convolution on Multicores

David Budden et al.

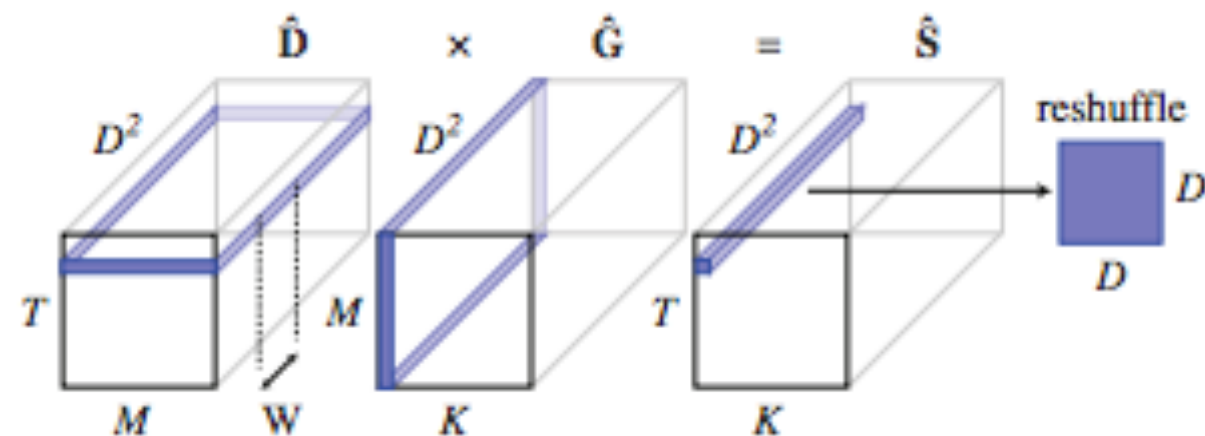
- Algorithm:
  - Input:  $g(x)$ ,  $d(x)$ ,  $m(x)$
  - for  $k=1$  to  $r$  do
    1. Compute residual polynomials  $g^{(k)}(x)$  and  $d^{(k)}(x)$
    2. Compute residual polynomial mults,  $s^{(k)}(x)$
    3. Reduce partial convolutions to solve  $s(x)$



# Deep Tensor Convolution on Multicores

David Budden et al.

- Sparse matrices
- We can even be dumb and use for loops, which is nice especially in 3D





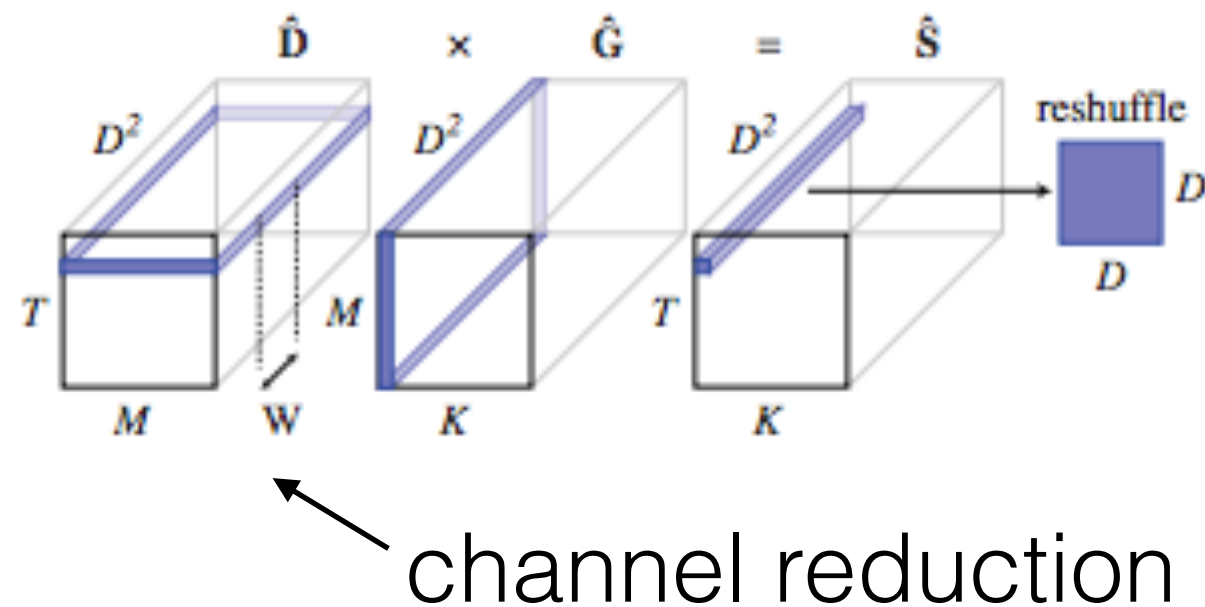


# Deep Tensor Convolution on Multicores

David Budden et al.

- 🥰 Sparse matrices 🥰
- We can even be dumb and use for loops, which is nice especially in 3D

$$\mathbf{s} = \mathbf{A}[(\mathbf{C}\mathbf{g})] \odot (\mathbf{B}\mathbf{d})$$





# Deep Tensor Convolution on Multicores

David Budden et al.

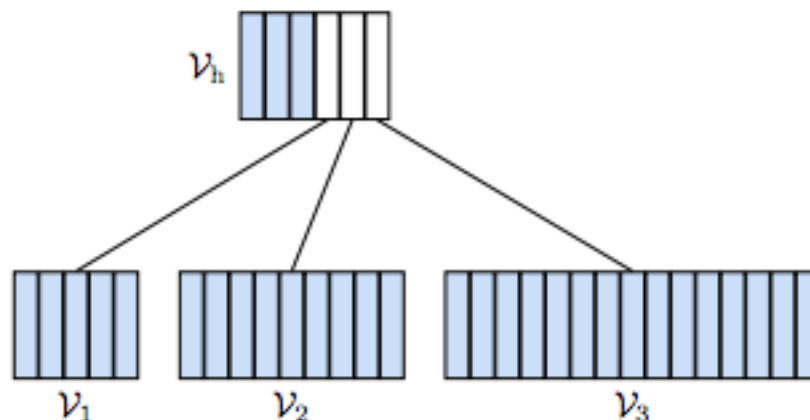
- Key idea: perform a reduction on the channels (size of inputs) and features (size of outputs) jointly, and only do it once so that the cost is amortized across all forward props (i.e. It makes inference cheap)
- Utilization tip: never do elementwise products. That's what makes you memory bound and slow
- Reduce the number of times you load each element into memory
- Gets to 70% real utilization (as a percent of the theoretical flops)
- Scalability tip: don't parallelize into the batch. This pollutes your cache with a bunch of things that are discontinuous in memory.
- This is 5-25x faster than Caffe or TF on CPU



# Efficient Softmax for GPUs

Édouard Grave et al.

- Statistical long range modeling is commonly used in NLP to learn a distribution over sequences of words. Many different models for this, RNNs are the best today
- Long range RNNs are computationally intensive to train (days of training on relatively small datasets)
- Computing the softmax is expensive because vocabularies tend to be large
- One approach is to use hierarchical softmax (assign each word to a class, and use the class and the history hierarchically)

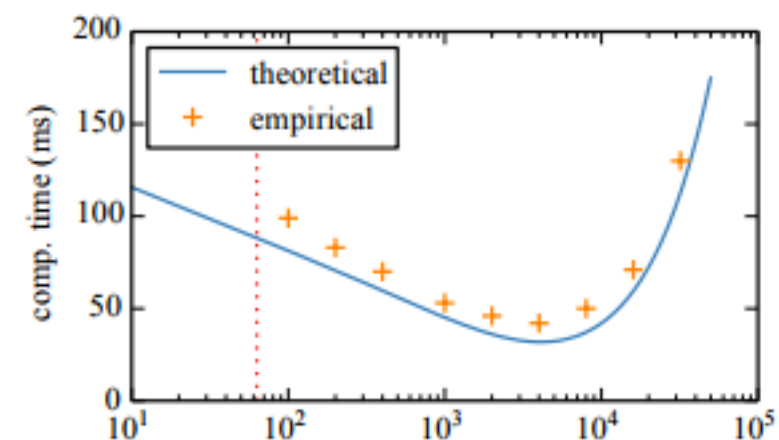
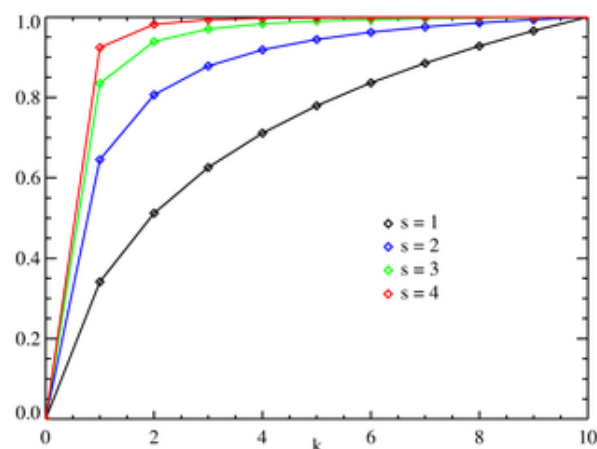




# Efficient Softmax for GPUs

Édouard Grave et al.

- Small clusters in your hierarchical softmax tend to not work well on GPU, because it splits up the batches
- Second problem is that word distribution follows Zipf's law. A small part of the vocabulary accounts for most frequent words
- Ideally we could have computation that's fast for frequent words





# Efficient Softmax for GPUs

Édouard Grave et al.

- Approach: partition the vocabulary into frequent and infrequent words (2 clusters), instead of classes
- This can generalize to multiple clusters, but there's not much gain after 5 clusters
- Bigger lesson: exploit prior knowledge about the distribution of your data!



# Scalability



Learning to Scale



Hardware



**Compression**



Dimensionality Reduction



# Beyond Filters: Compact Feature Map for Portable Deep Model

Yunhe Wang et al.

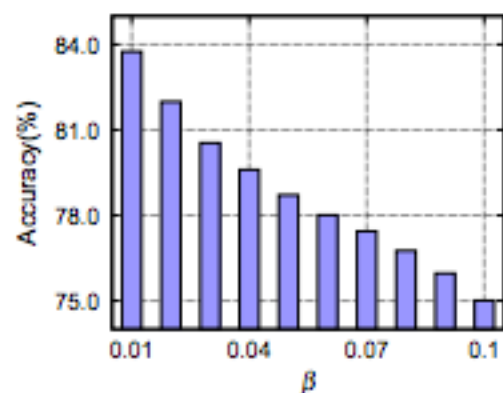
- Why compress CNNs?
  - Storage
  - Computational requirements
  - Network bandwidth (can't download over 100mb app on iPhone unless on wifi; resnet exceeds this).
- Does it hurt accuracy?
  - Not really: can compress 75% without performance loss
  - This suggests a high degree of redundancy



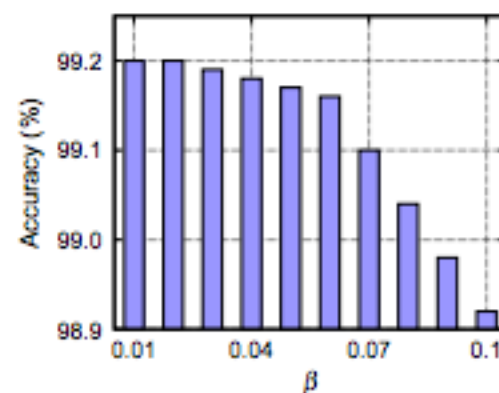
# Beyond Filters: Compact Feature Map for Portable Deep Model

Yunhe Wang et al.

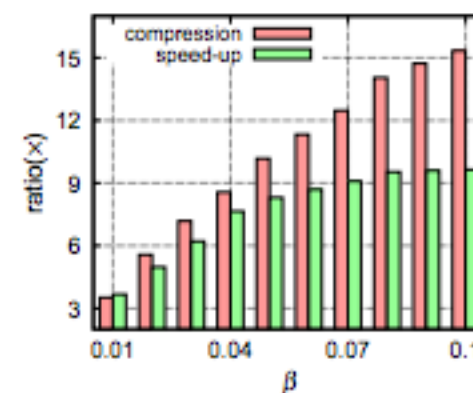
- Existing compression methods tend to compress the filters, don't reduce redundancy in feature maps or speed up computation
- Goal: maintain accuracy of the original network by preserving the distance between feature maps (i.e. learn a projection and remove redundant information in the feature maps)



(a) Accuracy of the reconstructed model.



(b) Model accuracy after fine-tuning.



(c) Compression and speed-up ratios.





# Scalability



Learning to Scale



Hardware



Compression



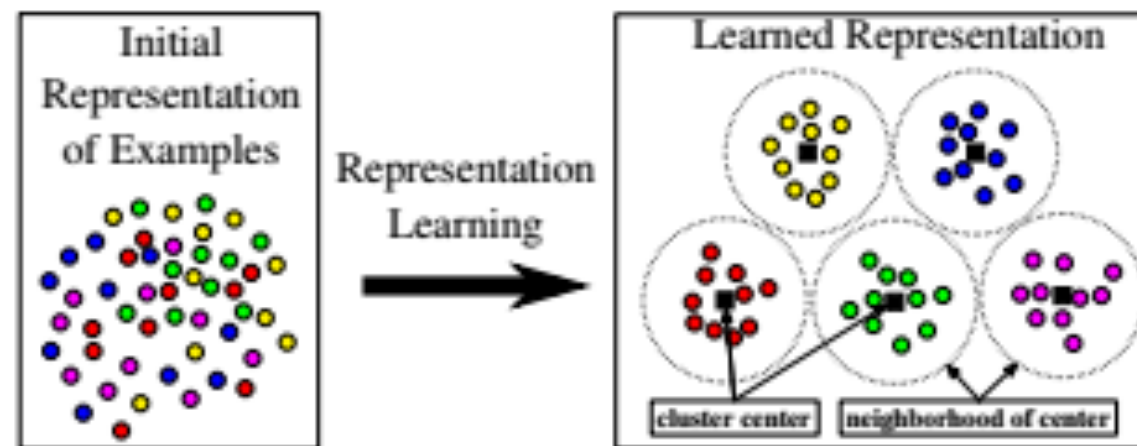
**Dimensionality Reduction**



# Deep Spectral Clustering Learning

Marc Law, Raquel Urtasun, Richard Zemel

- Goal is to group similar examples

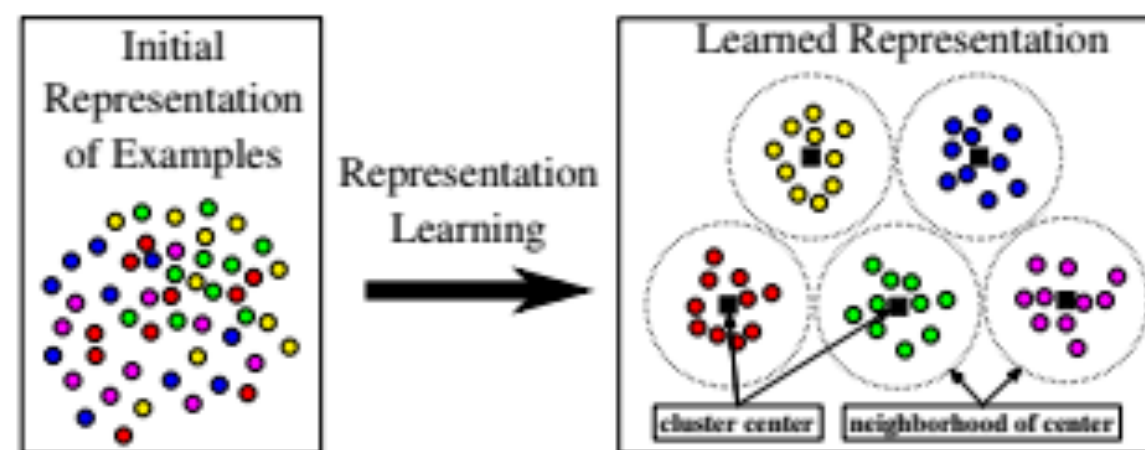
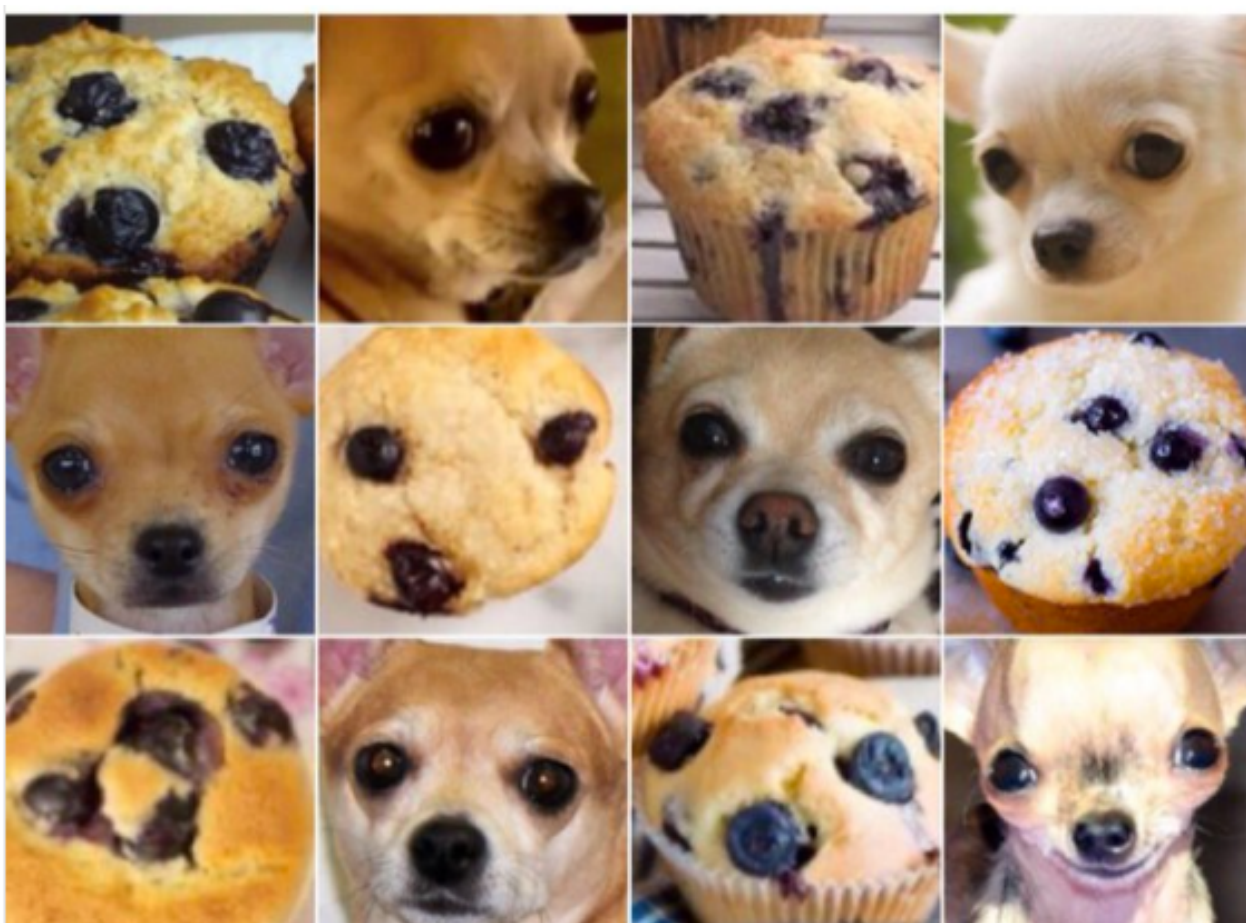




# Deep Spectral Clustering Learning

Marc Law, Raquel Urtasun, Richard Zemel

- Goal is to group similar examples

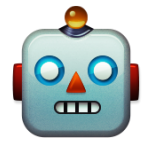




# Deep Spectral Clustering Learning

Marc Law, Raquel Urtasun, Richard Zemel

- Proposal is an efficient method to partition a set
- Reframe k-means as the search for
  - a vector of centroids,  $z$
  - a matrix of cluster assignments,  $Y$
- Problem reduces to gradient-based nonlinear regression



# Human-Computer Interaction



Active Learning for Vision



Interpretability

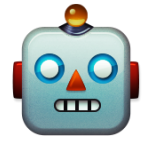


Machine Teaching



Ethics





# Human-Computer Interaction



## **Active Learning for Vision**



Interpretability



Machine Teaching



Ethics



# Explore/Exploit Trade-Off

- Optimal stopping problem
- Multi-armed bandits



VS







# Deep Active Learning with Image Data

Yarin Gal, Riashat Islam, Zoubin Ghahramani

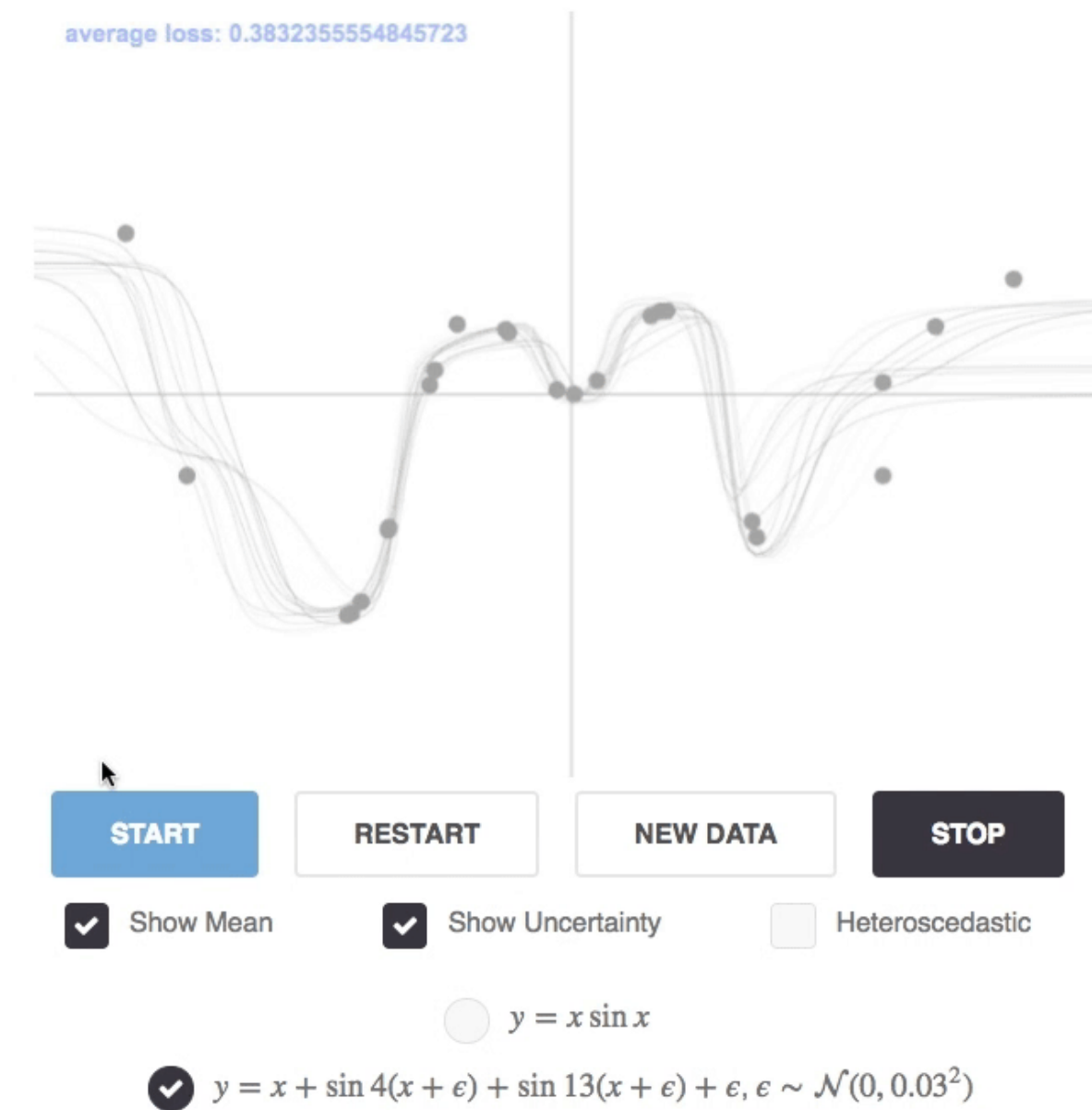
- Problem 1: most deep learning models do not give true measures of uncertainty. For active learning, we would ideally query the most uncertain examples.
- Problem 2: measures of uncertainty are computationally expensive
- Approach: use dropout as a proxy for uncertainty (cf. Gal 2016)





# Deep Active Learning with Image Data

Yarin Gal, Riashat Islam, Zoubin Ghahramani



[http://mlg.eng.cam.ac.uk/yarin/blog\\_2248.html](http://mlg.eng.cam.ac.uk/yarin/blog_2248.html)



# Deep Active Learning with Image Data



Yarin Gal, Riashat Islam, Zoubin Ghahramani

- With this uncertainty, we can use deep learning in "small data" domains
- This approach works with unbalanced datasets
- Application: better measures of uncertainty, experiment with different acquisition functions



# Automated Curriculum Learning for Neural Networks


Alex Graves et al.

- What do we “teach” our DNN and when do we teach it?
- How to define learning progress?
  1. Loss-based signals 
  2. Complexity based signals 



# Automated Curriculum Learning for Neural Networks


Alex Graves et al.

- Loss-based signals 
- Prediction Gain
- Gradient Prediction Gain
- Self Prediction Gain
- Target Prediction Gain
- Mean Prediction Gain



# Automated Curriculum Learning for Neural Networks

Alex Graves et al.

- Complexity based signals 
- Variational Complexity Gain
- Gradient Variational Complexity Gain
- L2 Gain
- Gradient L2 Gain



# Automated Curriculum Learning for Neural Networks

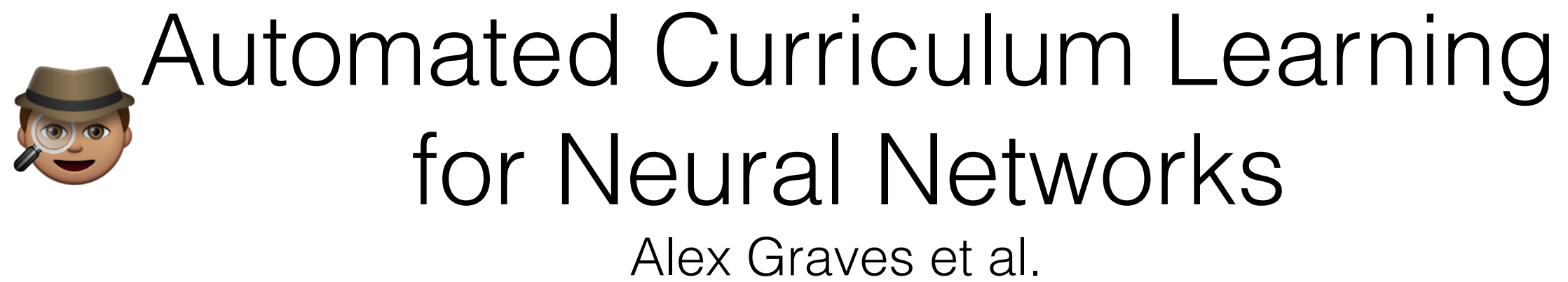
Alex Graves et al.

- Example: repeat copy curriculum
- Target



- Input





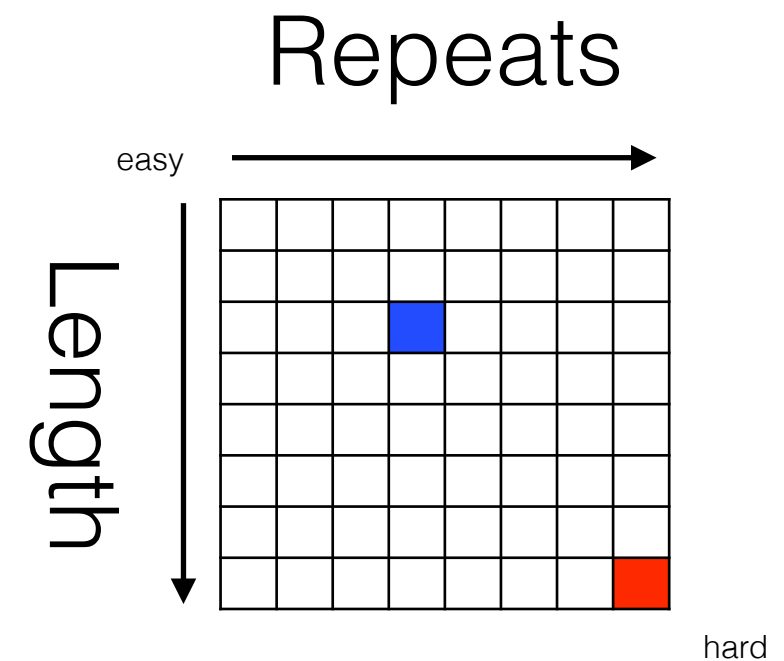
Alex Graves et al.

Alex Graves et al.

- Example: repeat copy curriculum
- Target



- Input





# Automated Curriculum Learning for Neural Networks

Alex Graves et al.

- Approach: sample with  $1-p(\text{correct})$
- Uniform sampling works well. Why?
  - Suppose out of  $N$  tasks, only 1 is learnable
  - If task  $i$  is too hard,  $\mathbb{E} \nabla L_i(x, \theta) = 0$
  - Then expected gradient is

$$\sum_N \frac{1}{N} \mathbb{E} \nabla L_i(x, \theta) = \frac{1}{N} \mathbb{E} \nabla L_1(x, \theta)$$

- Our only hope is for gradients to cancel out 😞

$$\mathbb{E} \nabla L_1(x, \theta) + \mathbb{E} \nabla L_2(x, \theta) = 0$$





# Leveraging Union of Subspace Structure to Improve Constrained Clustering

John Lipor, Laura Balzano

- Problem: low dimensional representations become less representative if we draw from  $n > 1$  distribution



$n = 1$



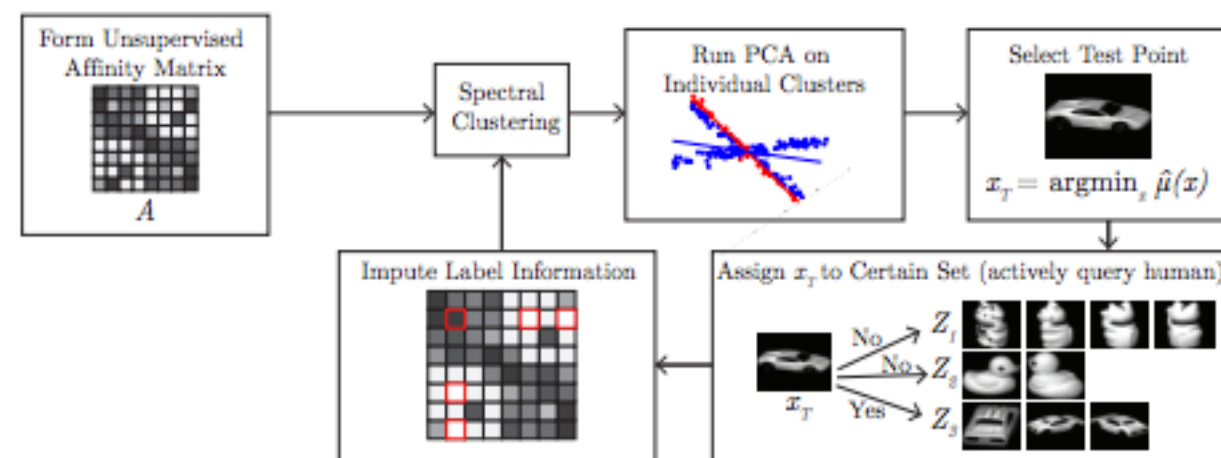
$n = 30$



# Leveraging Union of Subspace Structure to Improve Constrained Clustering


John Lipor, Laura Balzano

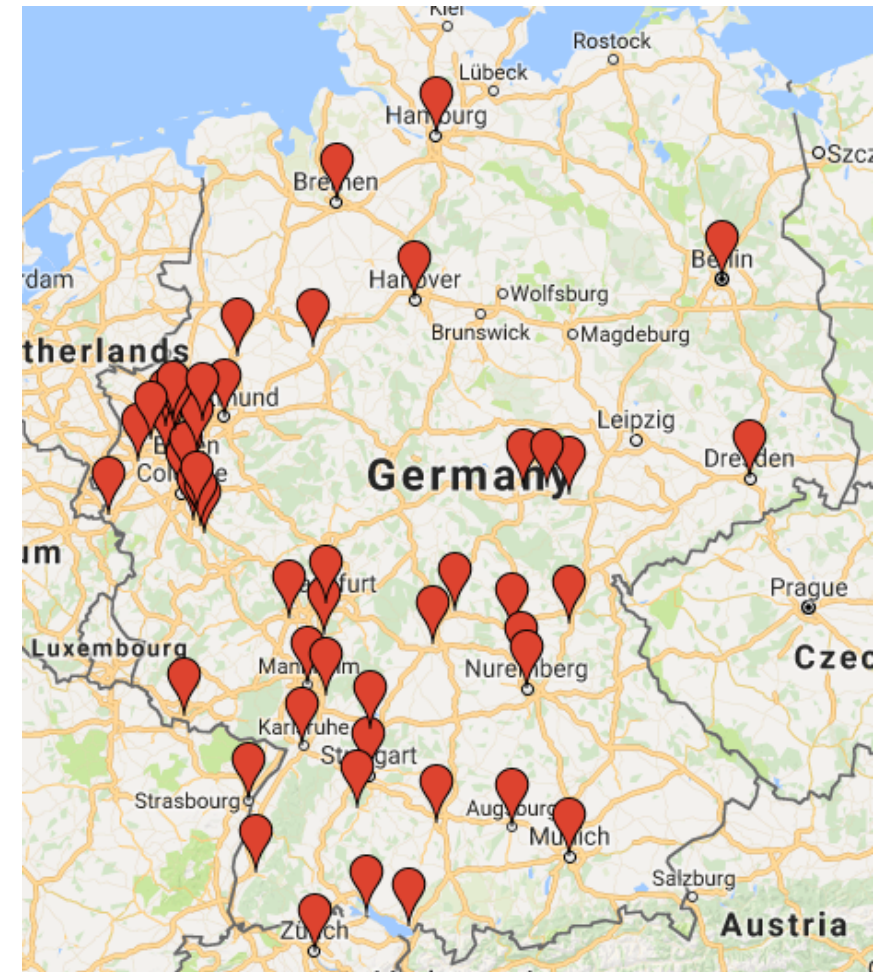
- Which pairs to query?
  - Points of minimum margin (those near the decision boundary)
- Need an analogous notion of margin for the problem of subspaces
- Let the margin be the norm after projecting into the nearest subspace minus the norm to the second nearest subspace



# Beyond Hand Labeling

# Matt Johnson

- In academia, data for AV consists of Kitty and Cityscapes
  - Lack of spatial/cultural variation
    - Kitty is from one city in Germany
    - Cityscapes is from a handful of cities
  - Can we use unlabeled data?
- 
- A map of Germany and surrounding regions (Netherlands, Denmark, Poland) showing numerous red pins. The pins are concentrated in the western part of Germany, particularly around the Rhine-Ruhr area, and are also scattered in the north (around Hamburg and Berlin) and south (around Munich and Frankfurt). This visualizes the limited spatial distribution of labeled AV datasets like KITTI and Cityscapes.





# Beyond Hand Labeling

Matt Johnson

- Approach 1: simulate data from GTA V







# Beyond Hand Labeling

Matt Johnson

- Approach 1: simulate data from GTA V
  - Easy to generate 500k images
  - DNN trained only on simulated data does better on real images than a network trained on a single city from Germany
  - Also beats KITTI-Cityscapes cross validation
- GTA is overkill, GANs and PixelCNN are probably good enough



# Beyond Hand Labeling

Matt Johnson

- Approach 2: mining for labels
- Stereo images: use detections in one to get detections in the other
  - Surprisingly often, CNNs succeed in one and fail in the other (interesting to find out why)
  - We should do this with sequential images
- You can find spatial areas where you're doing poorly and collect more data in those areas
  - We should do this too

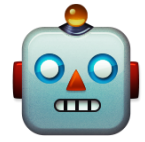


# Domain Adaptation

Min Sun

- Segmentation models tend to not transfer well between cities (Cairo, Taipei)
- Active learning: have human annotators label the most uncertain polygon in an image
- Dataset of dashcam crashes
  - Annotated and with geolocation
  - 968 videos from Taiwan, with labeled cars, times, and geo
  - 1.2 seconds before occurrence with 80% precision 45% accuracy





# Human-Computer Interaction



Active Learning for Vision



**Interpretability**



Machine Teaching



Ethics





# Network Dissection

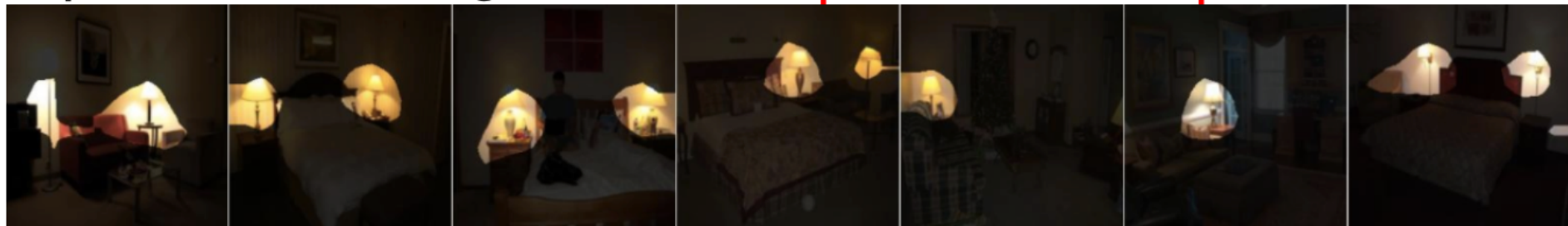
David Bau et al.

https://arxiv.org/pdf/1605.04871v1.pdf

- Main idea: see what your network is seeing
- Approach 1: look at the top activated images per class

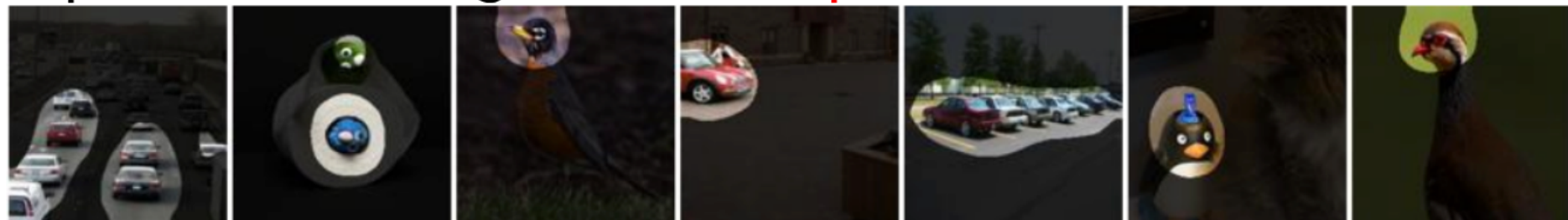
Top Activated Images

Interpretation: lamp Score: 0.15



Top Activated Images

Interpretation: car Score: 0.02

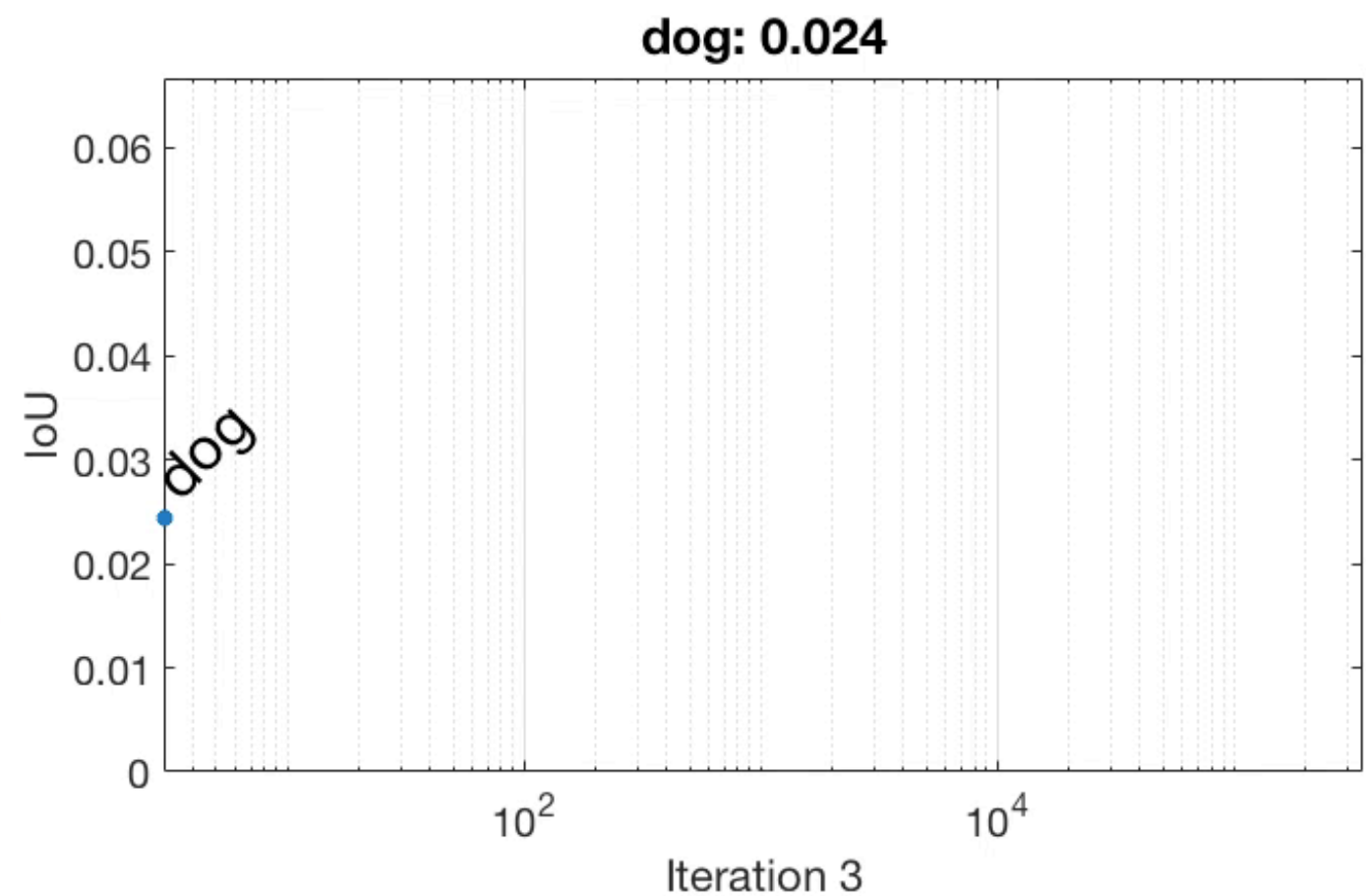




# Network Dissection

David Bau et al.

- Approach 2: look at individual across epochs





# Network Dissection

David Bau et al.

- Approach 2: look at individual across epochs



before

**fine-tuning**



after



# Network Dissection

David Bau et al.

<https://arxiv.org/abs/1610.02259>

- Caveat: requires pixel-wise labeling
- Open question: how does class imbalance impact this work?
- ImageNet has a ton of dog categories so you get very specific units such as ear detectors

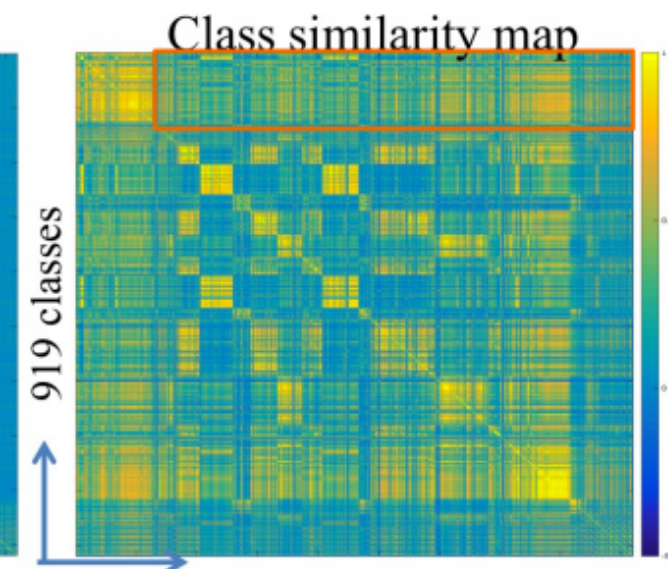
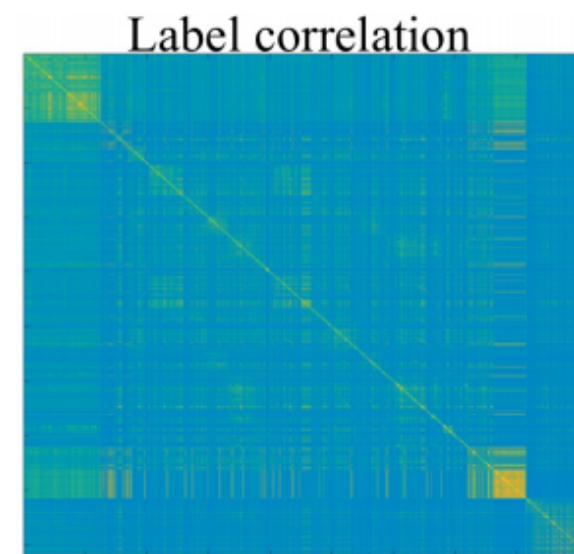
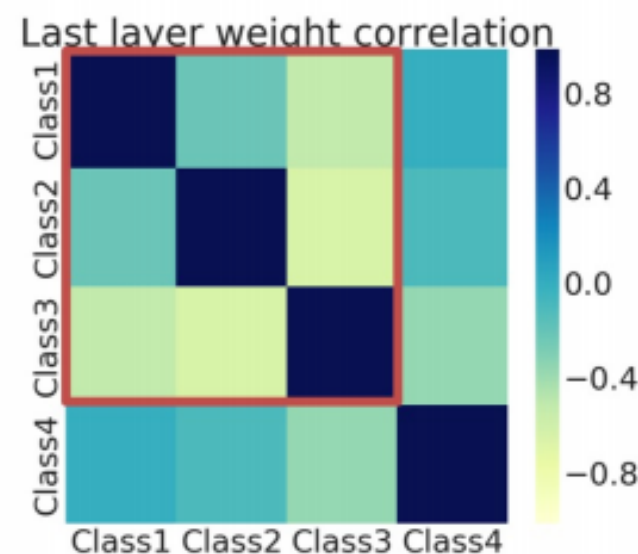
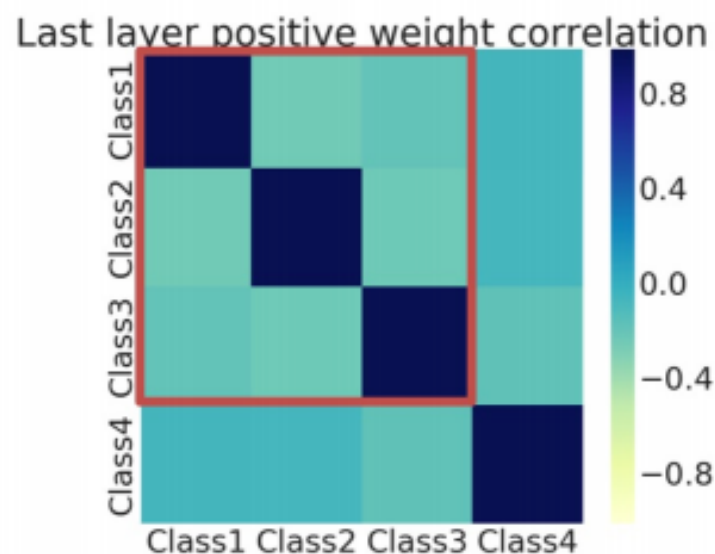




# Visualizing Feature Maps in Deep Neural Networks

Ge Liu, David Gifford

- Approach 1: invert neuron codes into image space or look at top activated inputs as a way to ask the question "which part of the input makes you say that?"
- Approach 2: break the network into separate parts and interpret them separately





# Bayesian Case Model

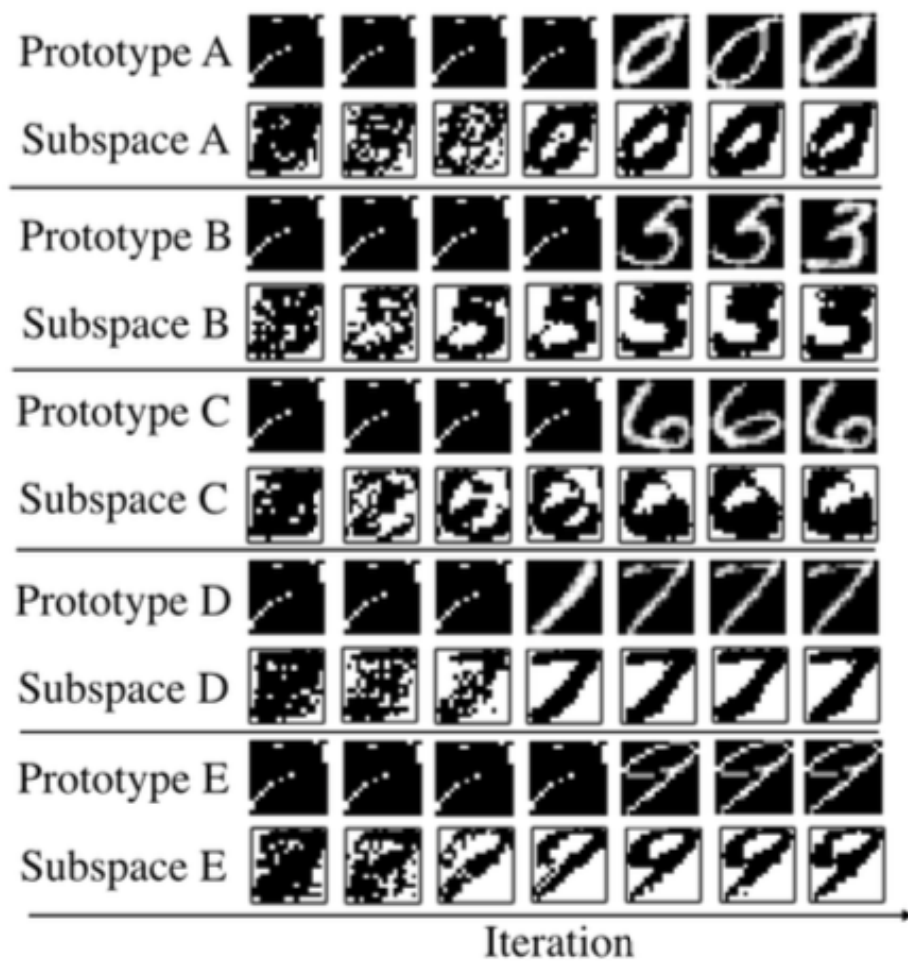
Been Kim and Cynthia Rudin

- Key notion: prototypes and subspaces
  - 🌮 or 🍝 is a prototype.
  - 🧀 or 🍅 is a subspace
- Clustering: a single dish has many ingredients
  - Each of its ingredients could belong to different sub spaces (bag of words, c.f. Ng and Jordan 2003)
- Explanation: prototypes are nice because you can point subject matter experts to real-world examples



# Bayesian Case Model

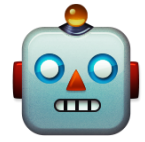
Been Kim and Cynthia Rudin



(a) *Handwritten Digit* dataset

| Prototype (Recipe names)         | Ingredients ( Subspaces )   |
|----------------------------------|---|
| <i>Herbs and Tomato in Pasta</i> | basil, garlic, Italian seasoning, oil<br>pasta pepper salt, tomato                        |
| <i>Generic chili recipe</i>      | beer chili powder cumin, gar-<br>lic, meat, oil, onion, pepper, salt,<br>tomato           |
| <i>Microwave brownies</i>        | baking powder sugar, butter,<br>chocolate chopped pecans, eggs,<br>flour, salt, vanilla   |
| <i>Spiced-punch</i>              | cinnamon stick, lemon juice<br>orange juice pineapple juice<br>sugar, water, whole cloves |

(b) *Recipe* dataset



# Human-Computer Interaction



Active Learning for Vision



Interpretability



**Machine Teaching**



Ethics

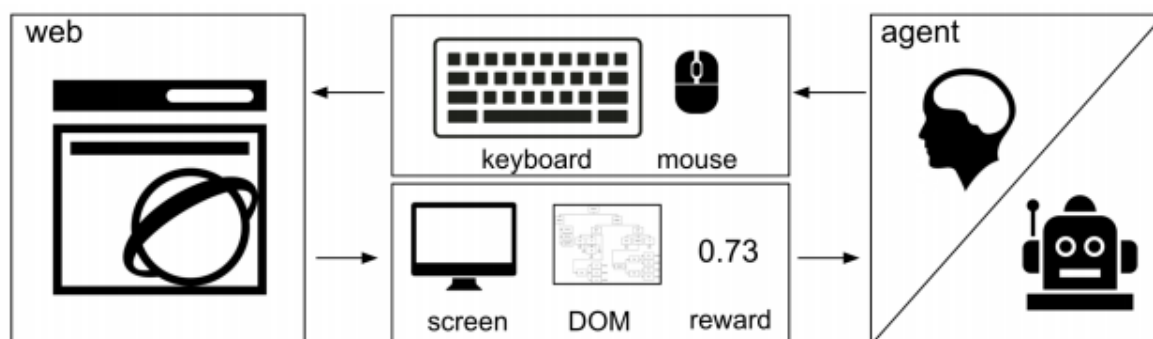




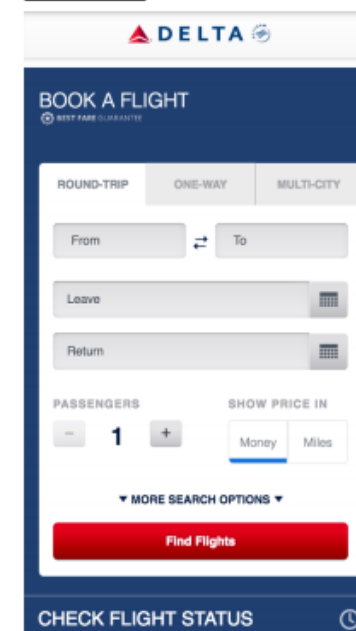
# World of Bits

Tianlin (Tim) Shi et al.

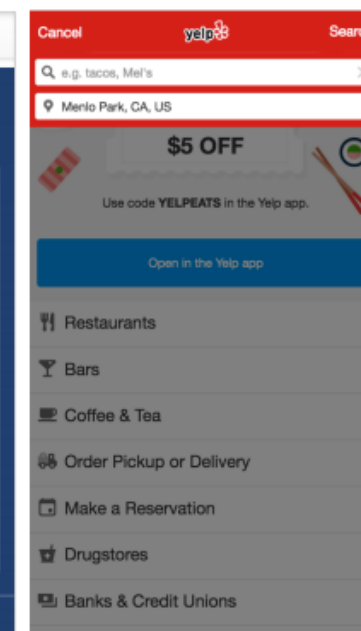
- Platform for RL tasks
- Wide variety of tasks (book a flight, online shopping, etc. basically a virtual assistant)
- Open domain (world wide web), easy to capture (JS)



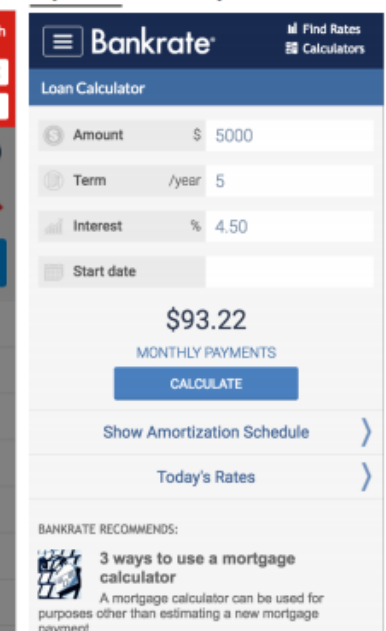
**Question:** Can you book a flight from San Francisco to New York?



**Question:** What is the top rated place to eat Korean food in SF?



**Question:** What is the monthly payment for \$2000 with a term of 2 years at today's rates?





# Dance Dance Convolution

Chris Donahue, Zachary C. Lipton, and Julian McAuley

- *Dance Dance Revolution*
  - “a popular, rhythm-based video game”
- Problem:
  - choreography only exists for a small number of songs
  - extant choreography is human generated (costly)

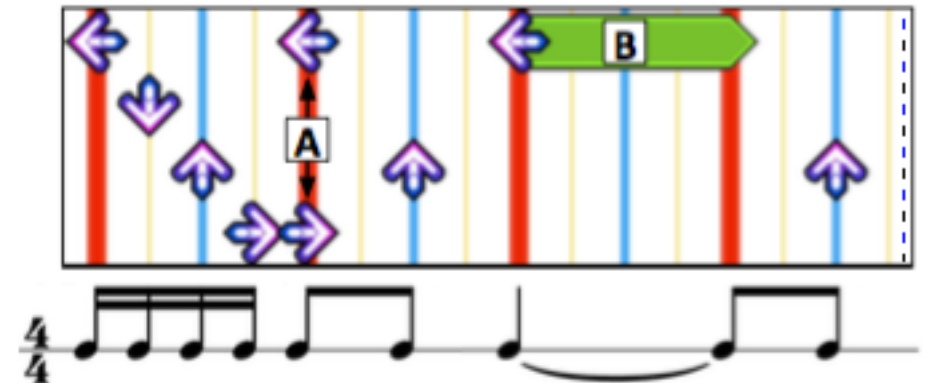




# Dance Dance Convolution

Chris Donahue, Zachary C. Lipton, and Julian McAuley

- Approach: machine-learned choreography
  - Input: audio sequence
  - Output: series of dance steps
- Four dance moves, four states ( $4^4$  possible steps)

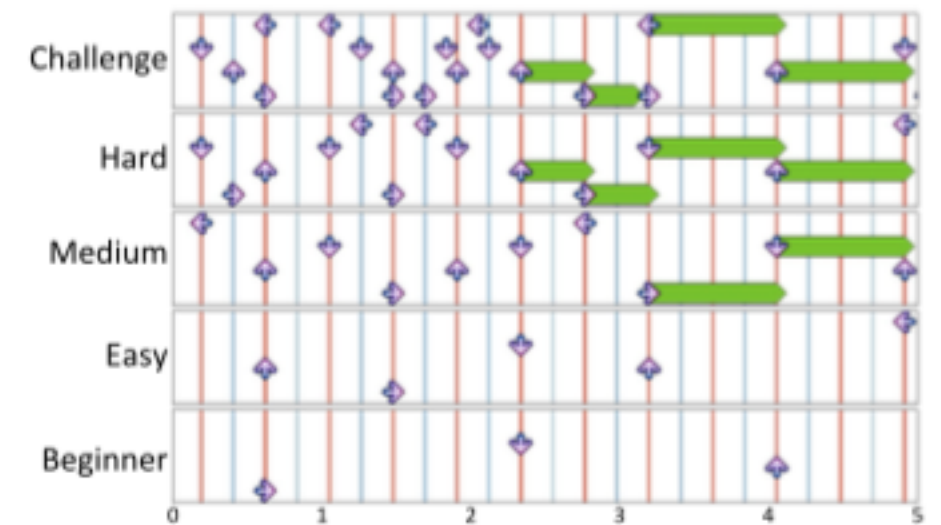




# Dance Dance Convolution

Chris Donahue, Zachary C. Lipton, and Julian McAuley

- 5 levels of difficulty
- Structure
  - interestingness
  - correlation with musical structure
  - can't have sequences where the player ends up looking away from the screen

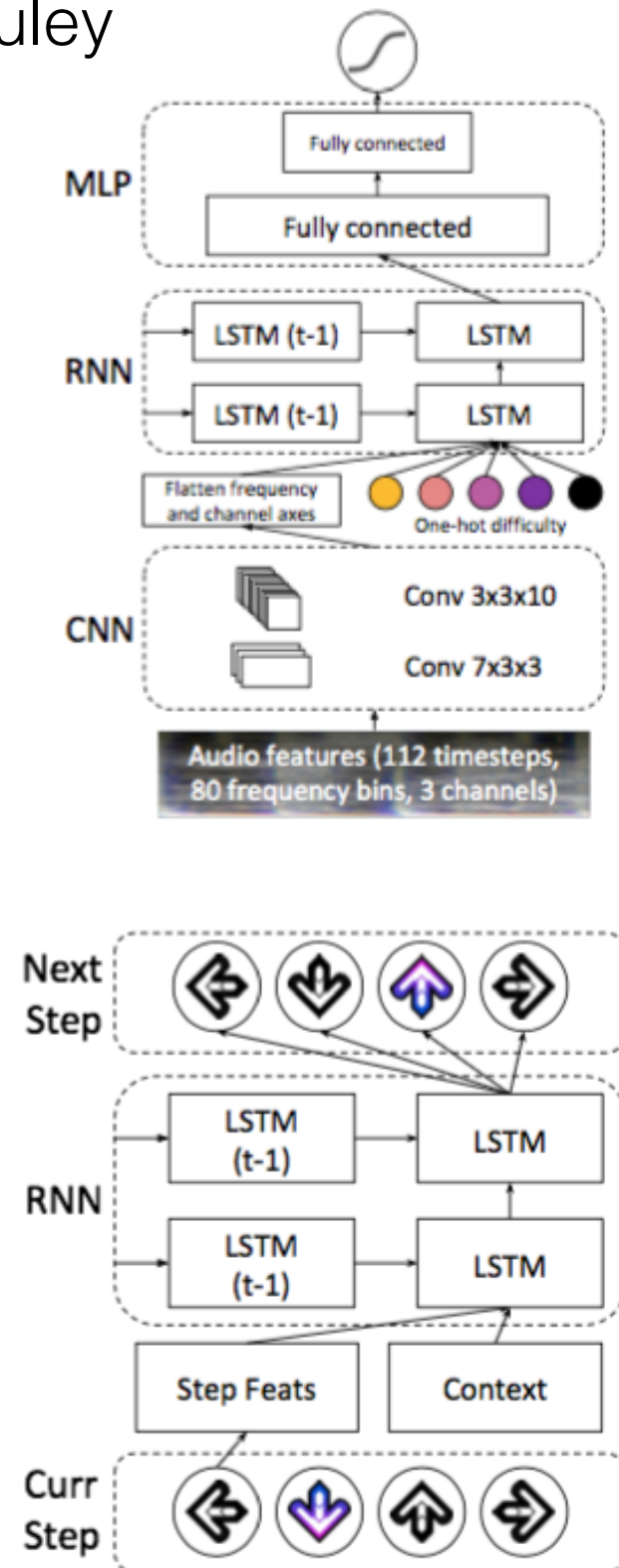




# Dance Dance Convolution

Chris Donahue, Zachary C. Lipton, and Julian McAuley

- Two-stage task
  1. When to place steps
    - likelihood model
    - mimic human annotator's placement
    - 2-layer CNN -> RNN -> MLP
  2. Which steps to place
    - RNN samples from a probability distribution
- Runs faster than real-time on commodity GPU

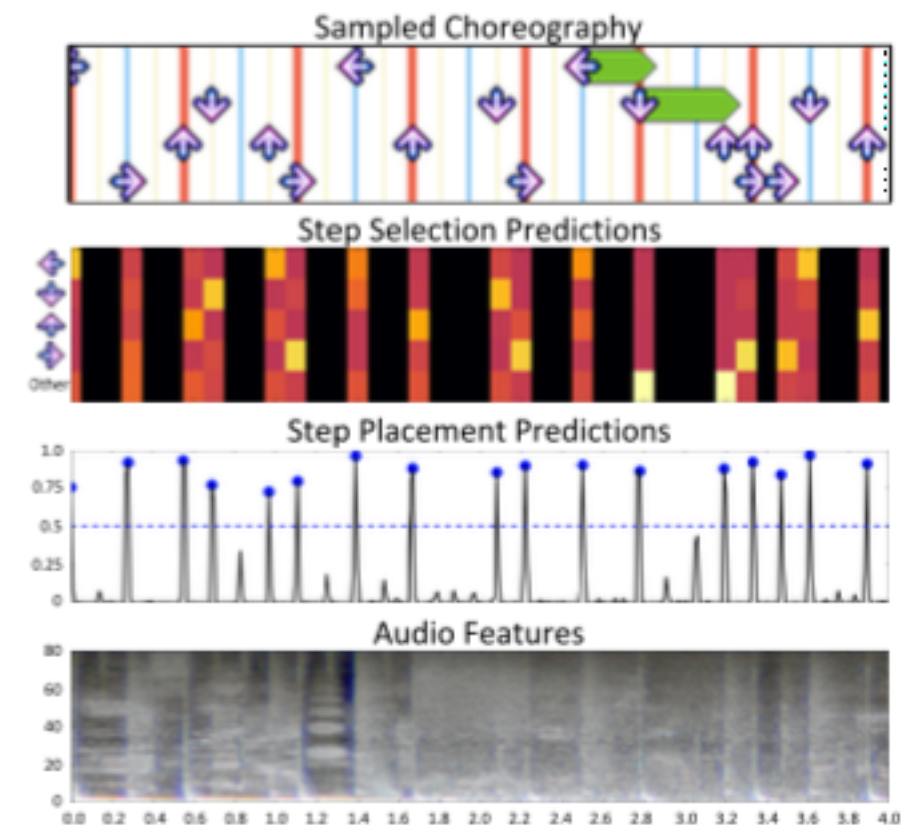




# Dance Dance Convolution

Chris Donahue, Zachary C. Lipton, and Julian McAuley

- Datasets
  - *Fraxtil*
    - 90 songs
    - 1 chart per difficulty level
    - single-author (model predicts w/61% accuracy)
  - *In the Groove*
    - 130 songs
    - 1 chart per difficulty level (for all but 13 songs)
    - multi-author
- [github.com/chrisdonahue/ddc](https://github.com/chrisdonahue/ddc)
- [deepx.ucsd.edu/ddc](https://deepx.ucsd.edu/ddc) (upload file, receive a chart)

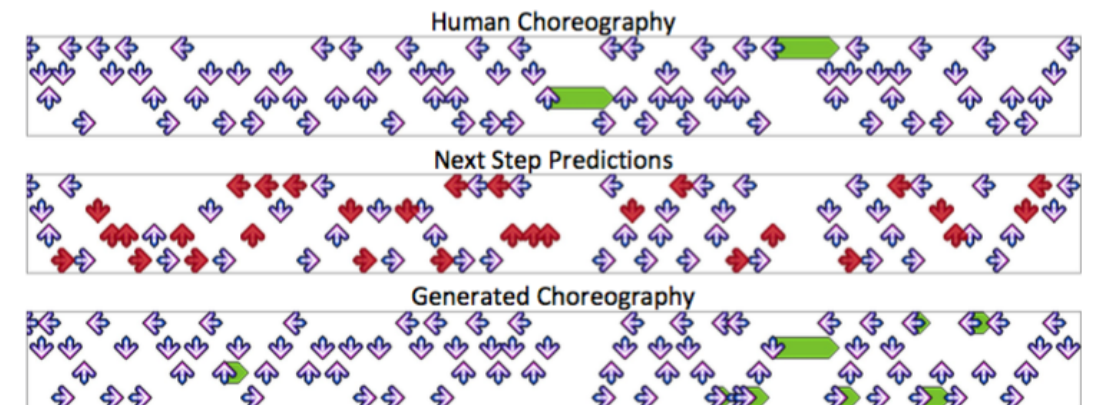




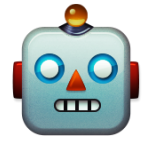
# Dance Dance Convolution

Chris Donahue, Zachary C. Lipton, and Julian McAuley

- Giving instructions to humans to do a task
- What other examples are there?
  - Driver routing
  - Chess moves
  - Choreography is safe, repeatable, and has quick feedback







# Human-Computer Interaction



Active Learning for Vision



Interpretability



Machine Teaching



**Ethics**

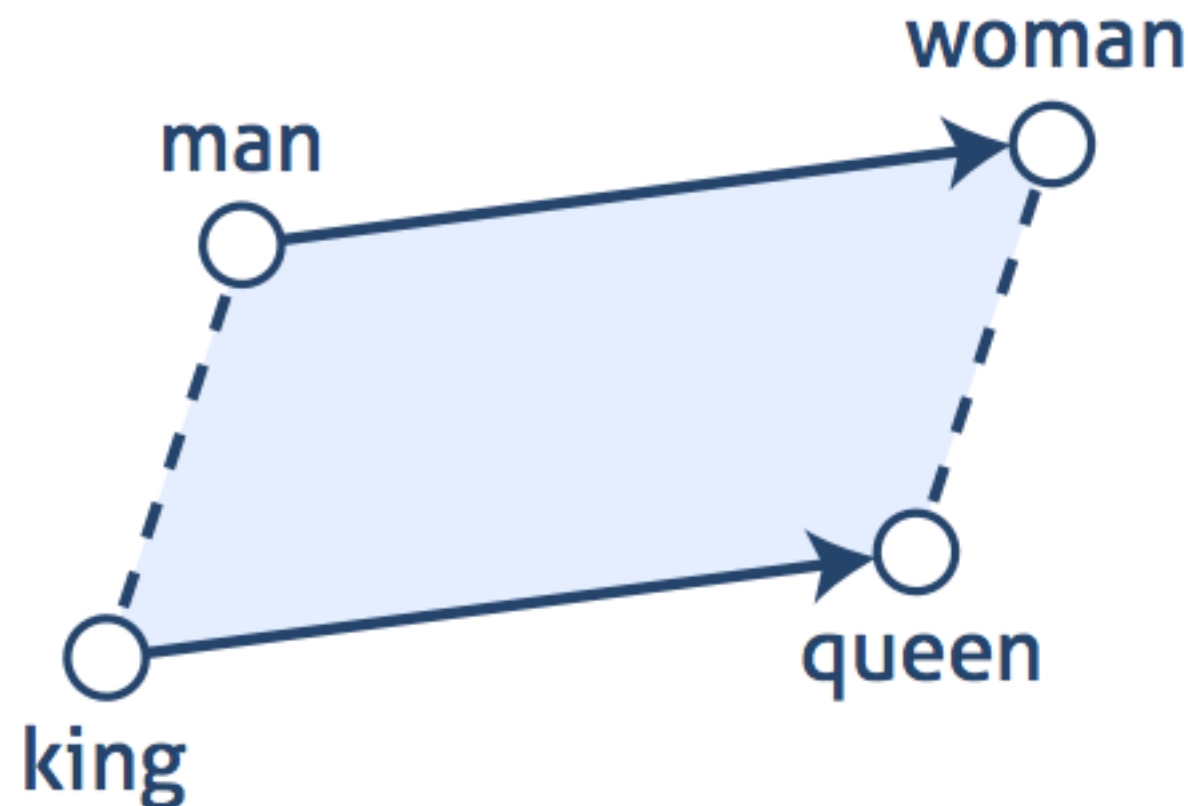




# De-Biasing Word Embeddings

Tolga Bolukbasi et al.

- Word2Vec trained on Google News corpus
- Man:King :: Woman:*Queen*
- Paris:France :: Tokyo:Japan





# De-Biasing Word Embeddings




Tolga Bolukbasi et al.

- He:Brother :: She:?
  - Sister
- He:Blue :: She:?
  - Pink
- He:Doctor :: She:?
  - Nurse
- He:Architect :: She:?
  - Interior designer
- She:Pregnancy :: He:?
  - Kidney stone
- He:Computer programmer :: She:?
  - Homemaker



# De-Biasing Word Embeddings

Tolga Bolukbasi et al.

- He:Brother :: She:?
  - Sister
- He:Blue :: She:?
  - Pink
- He:Doctor :: She:?
  - Nurse
- He:Architect :: She:?
  - Interior designer
- She:Pregnancy :: He:?
  - Kidney stone
- He:Computer programmer :: She:?
  - Homemaker   



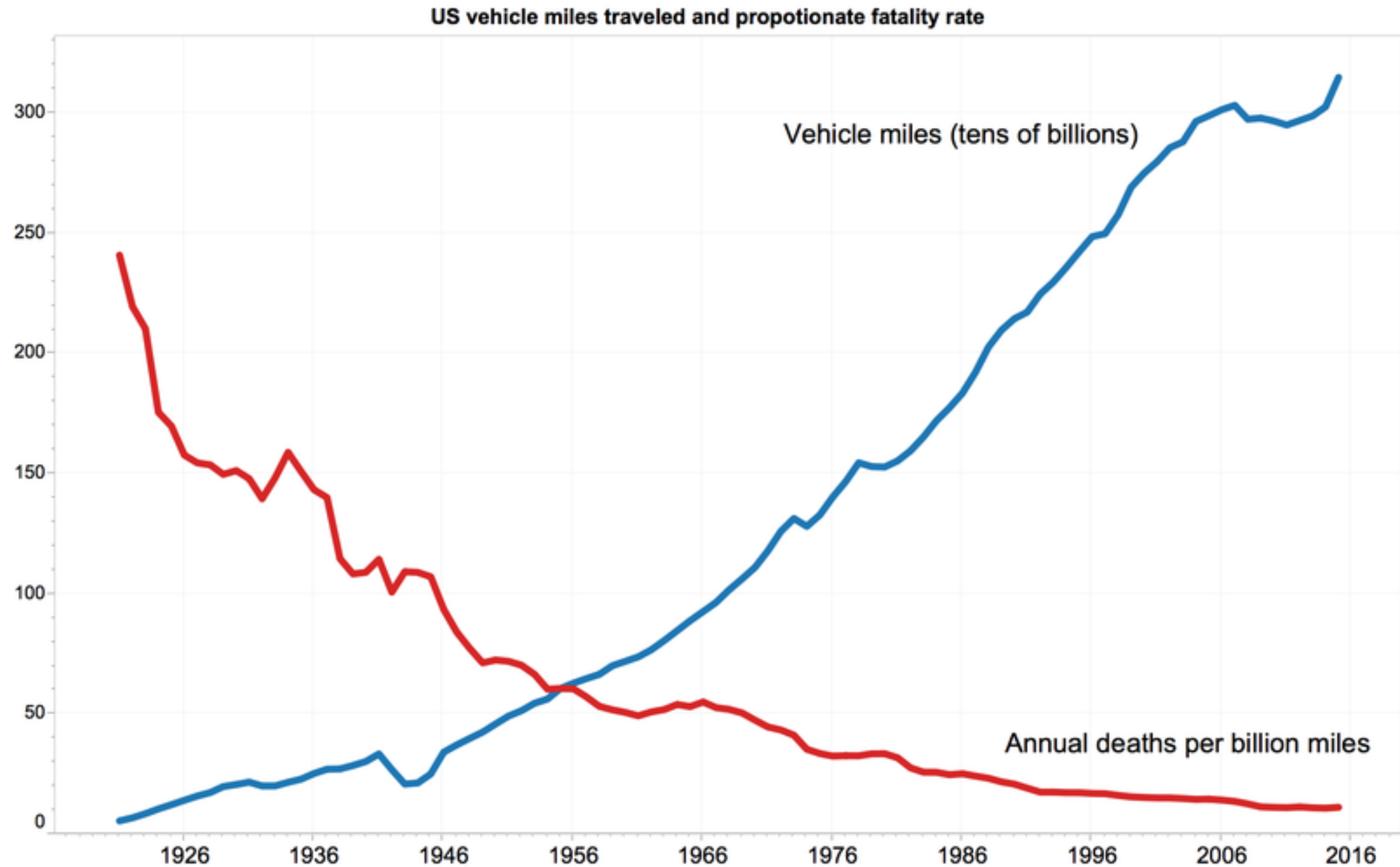
# De-Biasing Word Embeddings

Tolga Bolukbasi et al.

- ML is not just about optimizing an objective function
  - Our data and our results have social implications
- What are examples that could creep into mapping?
  1. If we take the mean route at a roundabout we will crash
  2. If Uber riders are systematically avoiding certain neighborhoods, we won't map them. This will then make it harder for those users to get rides (unlikely but possible). Also possible if drivers systematically cancel rides in certain areas such as favelas
- Note: this corpus includes historical data, so forthcoming paper looking at trends over time





# Ethical Implications of Self-Driving Cars





# Applications

- Use theory when you can 
  - Prior probabilities
  - Uncertainty
- Use empirics when you can't 
  - Evolving architectures
  - Active learning
  - Model visualization & interpretability





Thanks!

