Revisiting Unreasonable Effectiveness of Data in Deep Learning Era

Chen Sun[1], Abhinav Shrivastava[1,2], Saurabh Singh[1], and Abhinav Gupta[1,2]

[1]Google Research
[2]Carnegie Mellon University

17 July 2017
Junho Song

1. introduction

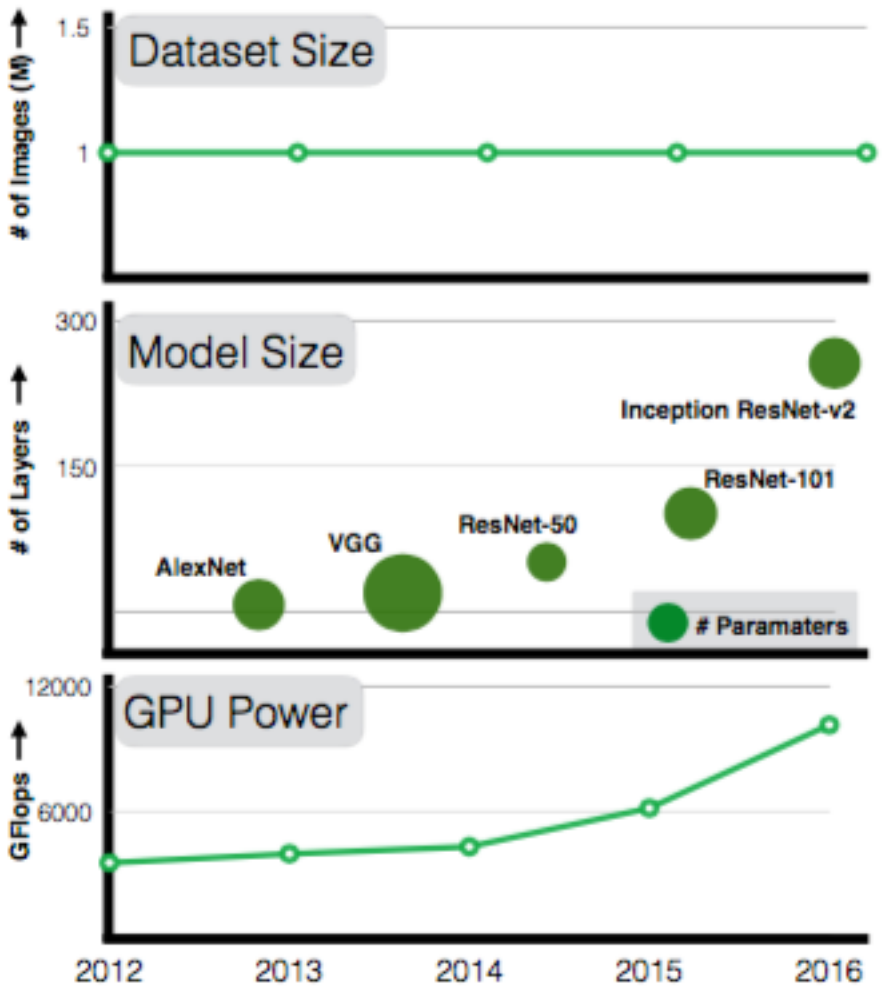
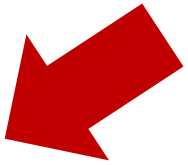


Figure 1. The Curious Case of Vision Datasets: While GPU computation power and model sizes have continued to increase over the last five years, size of the largest training dataset has surprisingly remained constant. Why is that? What would have happened if we have used our resources to increase dataset size as well? This paper provides a sneak-peek into what could be if the dataset sizes are increased dramatically.



if we scale up the amount of training data 10× or 100×, will the performance double?

1. introduction

already existing **JFT-image dataset**, first introduced by Hinton *et al*. [17] and expanded by [7].

more than 300M images

labeled with 18,291 categories

automatically obtained and, therefore, are noisy and not exhaustive.

These annotations have been cleaned using complex algorithms to increase the precision of labels;

however there is still approximately 20% error in precision.

← classification, object detection, semantic segmentation and human pose estimation.

1. introduction

If we have more data,

Better Representation Learning Helps! : more is better

Performance increases linearly with orders of magnitude of training data! : still linear

Capacity is Crucial : higher capacity

Training with Long-tail : long-tail OK

New state of the art results : best results

2. Related Work

trying deeper : GoogleNet, ResNet

fine-tune is better

correlation : models' capacity and classification performances on ImageNet

web-supervision or unsupervised paradigms

🡸 In this paper, we aim to shift the discussion from models to data.

2. Related Work

[Learning visual features from large weakly supervised data] showed plateauing of detection performance when trained on 100M images.

Why is that?

a) YFCC-100M images come only from Flickr.
JFT includes images all over the web, and has better visual diversity.

b) But more importantly, they did not see real effect of data due to use of smaller AlexNet of VGG models.

Therefore, not plateauing, but linear

3. The JFT-300M Dataset

300M images
375M labels,
on average each image has 1.26 labels.
18291 categories: *e.g.*, 1165 type of animals and 5720 types of vehicles
rich hierarchy with the maximum depth of hierarchy being 12

approximately 20% (in precision) of the labels in this dataset are noisy.
we have no way to estimate the recall of the labels

heavily long-tailed
more than 2M 'flowers',
3250 'subarau360'
but only 131 images of 'train conductors'.
<= more than 3K categories with less than 100 images each and
approximately 2K categories with less than 20 images per category

3. The JFT-300M Dataset

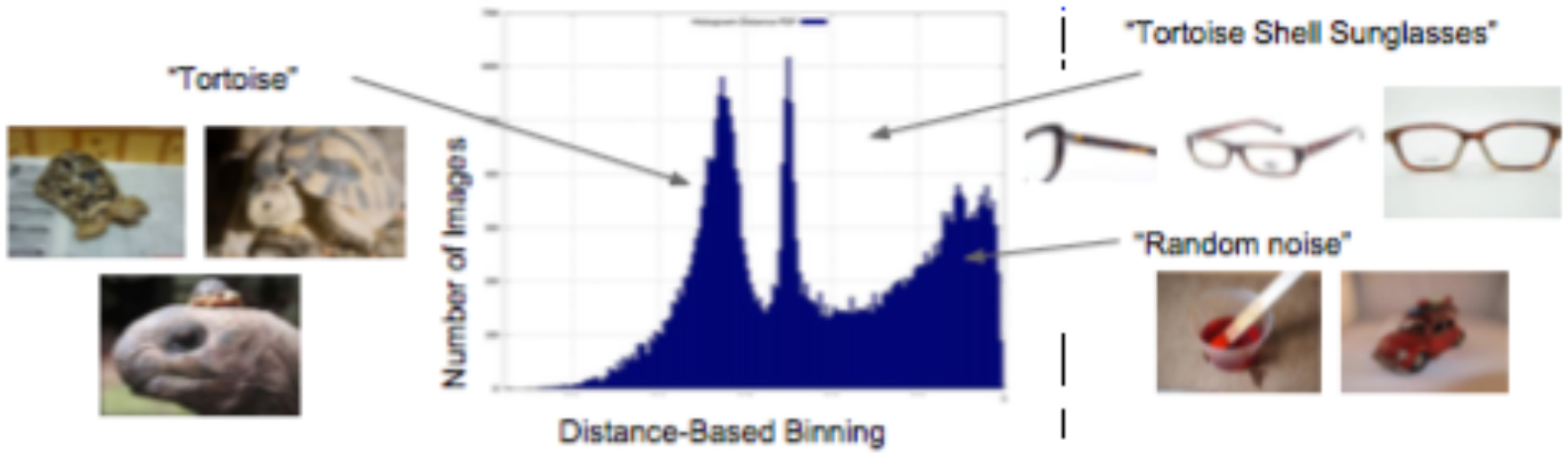


Figure 2. JFT-300M dataset can be noisy in terms of label confusion and incorrect labels. This is because labels are generated via a complex mixture of web signals, and not annotated or cleaned by humans. x-axis corresponds to the quantized distances to K-Means centroids, which are computed based on visual features.

4. Training and Evaluation Framework - 4.1. Training on JFT-300M Data

- ResNet101
- adding FC layer with 18291

(split it into 50 equal sized sub-fc layers, and distribute them different parameter servers.)

- auto-hierarchy labeling
- resized 340x340
- randomly crop 299x299
- normalized [-1, 1]
- random reflection for data augmentation
- weigh decay 0.0001
- BN
- RMSProp : 0.9
- batch size : 32
- learning rate : 0.001 (0.9 every 3M)
- asynchronous GD
- 50 NVIDIA K80
- tensorflow

4. Training and Evaluation Framework - 4.1. Training on JFT-300M Data

ImageNet baseline

- momentum of 0.9
- initial learning rate to 5×10^{-2}
- batch size to 32
- learning rate is reduced by a factor of 10 every 30 epochs (1.2M steps),
- 5M steps
- asynchronous gradient descent training
- 50 NVIDIA K80 GPUs and 17 parameter servers

🡸 baseline ResNet-101 performs 1% better than the open-sourced ResNet-101 checkpoint from the authors of [16]

4. Training and Evaluation Framework - 4.2. Monitoring Training Progress

Validation set : ‘FastEval14k’ [7]

- 14,000 images
- labels from 6000 classes (subset of 18291 classes from JFT-300M).
- densely annotated and there are around 37 labels per image

🡐 mAP@100 (top 100 prediction)

4. Training and Evaluation Framework - 4.2. Monitoring Training Progress

Validation set : ‘FastEval14k’ [7]

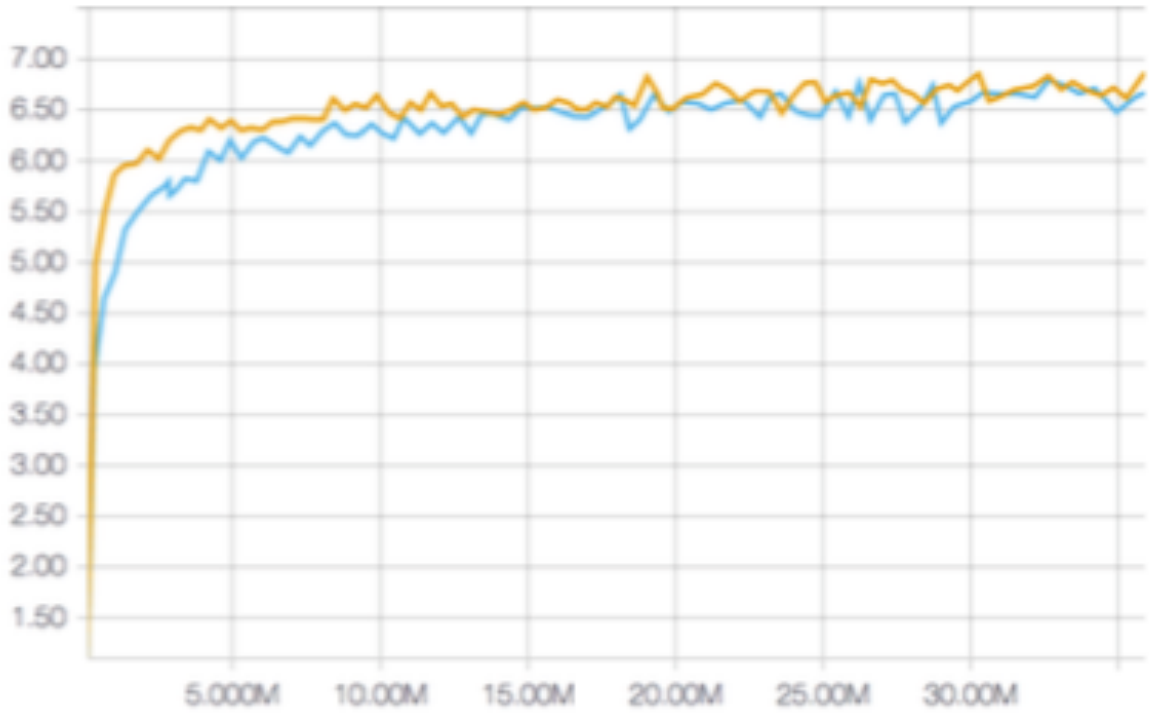


Figure 3. Comparison of training progress with random initialization (blue) and ImageNet initialization (yellow) on JFT-300M data. x-axis is the number of training steps, and y-axis shows the mAP@100 metric computed on FastEval14k.

Training

- ←36M iterations or 4 epochs
- ←approximately 2 months

4. Training and Evaluation Framework - 4.3. Evaluating the Visual Representations

- freeze the model weights and use these models as pure feature extractors.

- model weights as initialization and fine-tune
- object detection
- semantic segmentation
- human pose estimation.

5. Experiments - 5.1. Image Classification

Initialization	Top-1 Acc.	Top-5 Acc.
MSRA checkpoint [16]	76.4	92.9
Random initialization	77.5	93.9
Fine-tune from JFT-300M	**79.2**	**94.7**

Table 1. Top-1 and top-5 classification accuracy on the ImageNet 'val' set (single model and single crop inference are used).

initialize the model weights from the JFT-300M checkpoint trained for 36M
fine-tune on ImageNet for 4M

5. Experiments - 5.2. Object Detection

- performance vs. iterations and epochs?
- performance vs. saturate after certain amount of data?
  Do we see any plateauing effect with more and more data?
- How important is representational capacity?
- Is the number of classes a key factor in learning visual representation?
- How could clean data (*e.g.* imageNet) help improve the visual representations?

5. Experiments - 5.2. Object Detection - Experimental Setup

Val data
- COCO [25], we use a held-out 8000 images from the standard ‘val’ set.
- PASCAL VOC, we use the 16551 ‘trainval’

- TensorFlow Faster RCNN implementation [18]
- asynchronous training with 9 GPU workers and 11 parameter servers,
- momentum : 0.9
- each worker takes a single input image per step
- the batch size for RPN and box classifier training are 64 and 256 respectively.
- Input images are resized to have 600 minimum pixels and 1024 maximum pixels
- data augmentation used is random flipping

5. Experiments - 5.2. Object Detection - Experimental Setup

For COCO

- initial learning rate to be $4\times10-4$,
- decay the learning rate by a factor of 10 after 2.5M steps
- the total number of steps is 3M.

For PASCAL VOC

- initial learning rate to be $3 \times 10-4$,
- decay the learning rate by 0.1 after 500K steps,
- the model is trained for 700K steps
-

At inference

- 300 RPN proposals are used after NMS operation with a threshold of 0.7.
- The final detections go through another round of class-specific NMS (threshold of 0.6).

5. Experiments - 5.2. Object Detection - Comparison with ImageNet Models

Faster RCNN

Method	mAP@0.5	mAP@[0.5,0.95]
He *et al.* [16]	53.3	32.2
ImageNet	53.6	34.3
300M	56.9	36.7
ImageNet+300M	**58.0**	**37.4**
Inception ResNet [37]	56.3	35.5

ResNet 101

double # of layers

Table 2. Object detection performance comparisons with baseline methods on the COCO test-dev split. The first four Faster RCNN detectors are all based on ResNet-101 architecture, the last one is based on the InceptionResNet-v2 architecture. During inference, a single image scale and crop, and a single detection model are used for all experiments. Vanilla Faster RCNN implementations are used for all systems except for He *et al.* [16], which also includes box refinement and context.

5. Experiments - 5.2. Object Detection - Comparison with ImageNet Models

method	airplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	TV	mean
ImageNet	79.7	80.6	77.1	65.9	64.2	85.3	81.0	88.4	60.5	83.1	70.8	86.7	86.2	79.7	79.5	49.5	78.3	80.2	79.2	69.7	76.3
300M	87.2	88.8	79.6	75.2	67.9	88.2	89.3	88.6	64.3	86.1	73.6	88.7	89.1	86.5	86.4	57.7	84.2	82.1	86.7	78.6	81.4
ImageNet+300M	86.9	88.0	80.1	74.7	68.8	88.9	89.6	88.0	69.7	86.9	71.9	88.5	89.6	86.9	86.8	53.7	78.2	82.3	87.7	77.9	81.3

Table 3. Average Precision @ IOU threshold of 0.5 on PASCAL VOC 2007 'test' set. The 'trainval' set of PASCAL VOC 2007 and 2012 are used for training.

5. Experiments - 5.2. Object Detection - Impact of Epochs

#Iters on JFT-300M	#Epochs	mAP@[0.5,0.95]
12M	1.3	35.0
24M	2.6	36.1
36M	4	36.8

Table 4. mAP@[.5,.95] on COCO minival* with JFT-300M checkpoint trained from scratch for different number of epochs.

#Iters on ImageNet	#Epochs	mAP@[0.5,0.95]
100K	3	22.2
200K	6	25.9
400K	12	27.4
5M	150	34.5

Table 5. mAP@[.5,.95] on COCO minival* with ImageNet checkpoint trained for different number of epochs.

5. Experiments - 5.2. Object Detection - Impact of Data size

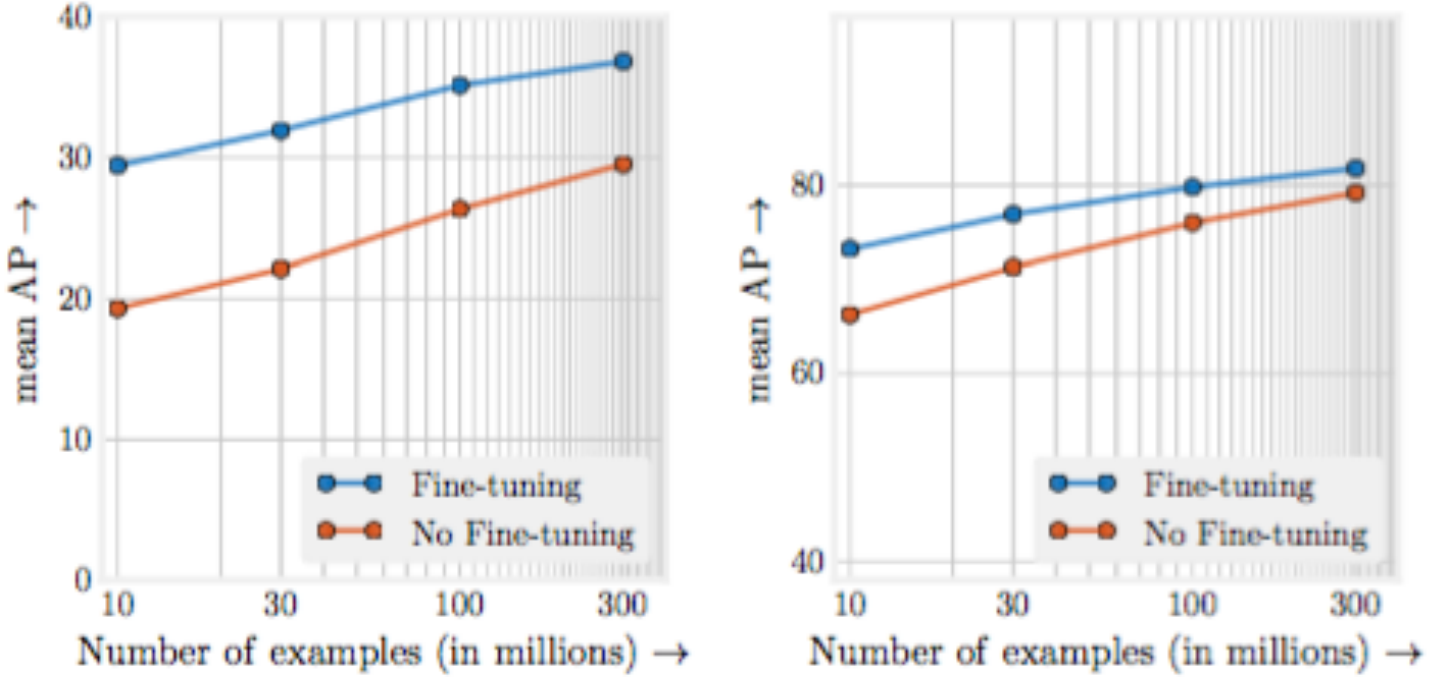


Figure 4. Object detection performance when initial checkpoints are pre-trained on different subsets of JFT-300M from scratch. x-axis is the data size in log-scale, y-axis is the detection performance in mAP@[.5,.95] on COCO minival* (left), and in mAP@.5 on PASCAL VOC 2007 test (right).

5. Experiments - 5.2. Object Detection - Impact of Classes

300M -> 30M with 18k label, 4epoch

Number of classes	mAP@[.5,.95]
1K ImageNet	31.2
18K JFT	31.9

Table 6. Object detection performance in mean AP@[.5,.95] on COCO minival* set. We compare checkpoints pre-trained on 30M JFT images where labels are limited to the 1K ImageNet classes, and 30M JFT images covering all 18K JFT classes.

← the performance benefit comes from more training images instead of more labels

5. Experiments - 5.2. Object Detection - Impact of Model Capacity

#Layers	ImageNet	300M
50	31.6	33.5
101	34.5	36.8
152	34.7	37.7

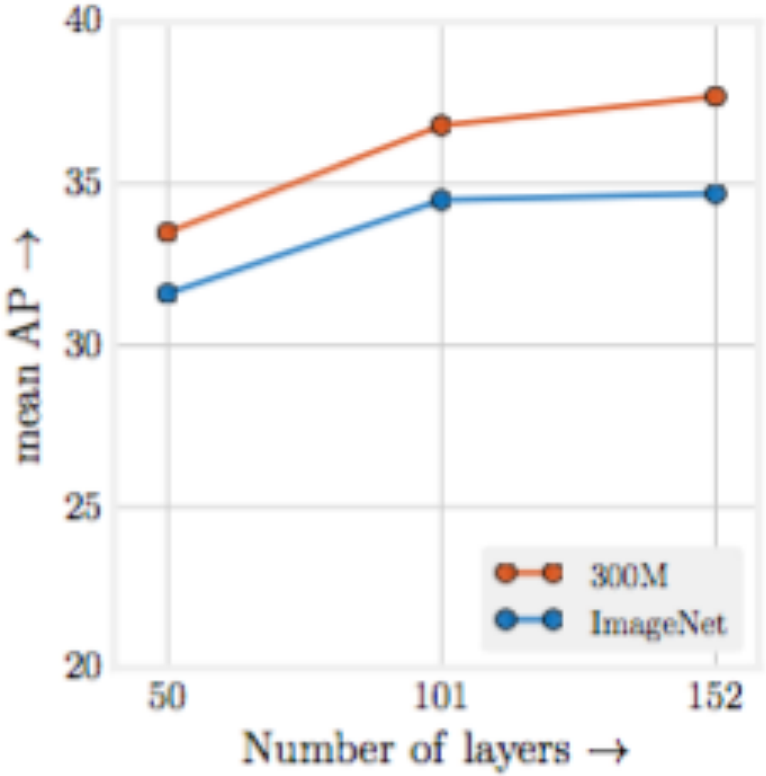


Figure 5. Object detection performance on COCO minival* on ResNet models with different number of layers.

5. Experiments - 5.3. Semantic Segmentation

Data

PASCAL VOC 2012 semantic segmentation benchmark [12]
20 foreground classes and one background class.
all models are trained on an augmented PASCAL VOC [12] 2012 'trainaug' set with 10582 images (extra annotations from [15]).
PASCAL VOC 2012 'val' set (1449 images) using the stan-dard mean intersection-over-union (mIOU) metric.

Model

DeepLab-ASPP-L model [4]

initialization	mIOU
ImageNet	73.6
300M	75.3
ImageNet+300M	**76.5**

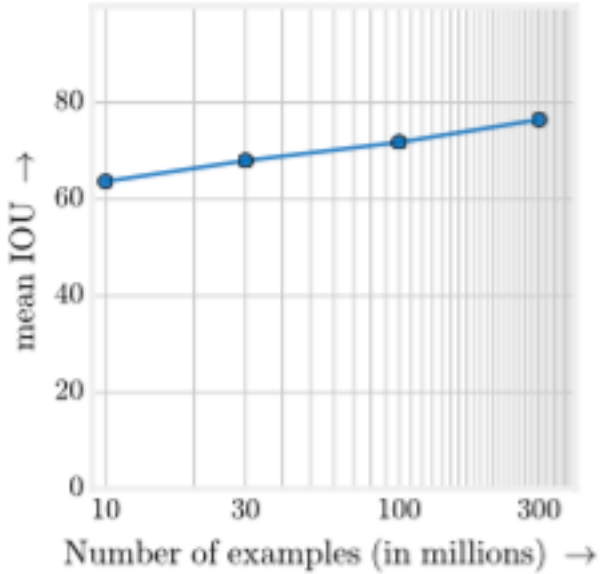


Figure 6. Semantic segmentation performance on Pascal VOC 2012 val set. (left) Quantitative performance of different initializations; (right) Impact of data size on performance.

5. Experiments - 5.4. Human Pose Estimation

Data

COCO 'test-dev' set

Model

fully-convolutional pose detector [28] by initializing the base ResNet model with our checkpoints

	AP	AP@.5	AR	AR@.5
CMU Pose [3]	61.8	84.9	66.5	87.2
ImageNet [28]	62.4	84.0	66.7	86.6
300M	64.8	85.8	69.4	88.4
ImageNet+300M	64.4	85.7	69.1	88.2

Table 7. Human pose estimation performance on COCO 'test-dev' split. We follow the implementation of G-RMI Pose [28], but change the ResNet-101 initial checkpoints from ImageNet pre-trained to JFT-300M pre-trained.

6. Discussions

Our paper is an attempt to put the focus back on the data.

the models seem to be plateauing,
but performance still increases linearly with respect to orders of magnitude of the data.

better models is not leading to substantial gains
because ImageNet is no more sufficient to use all the parameters