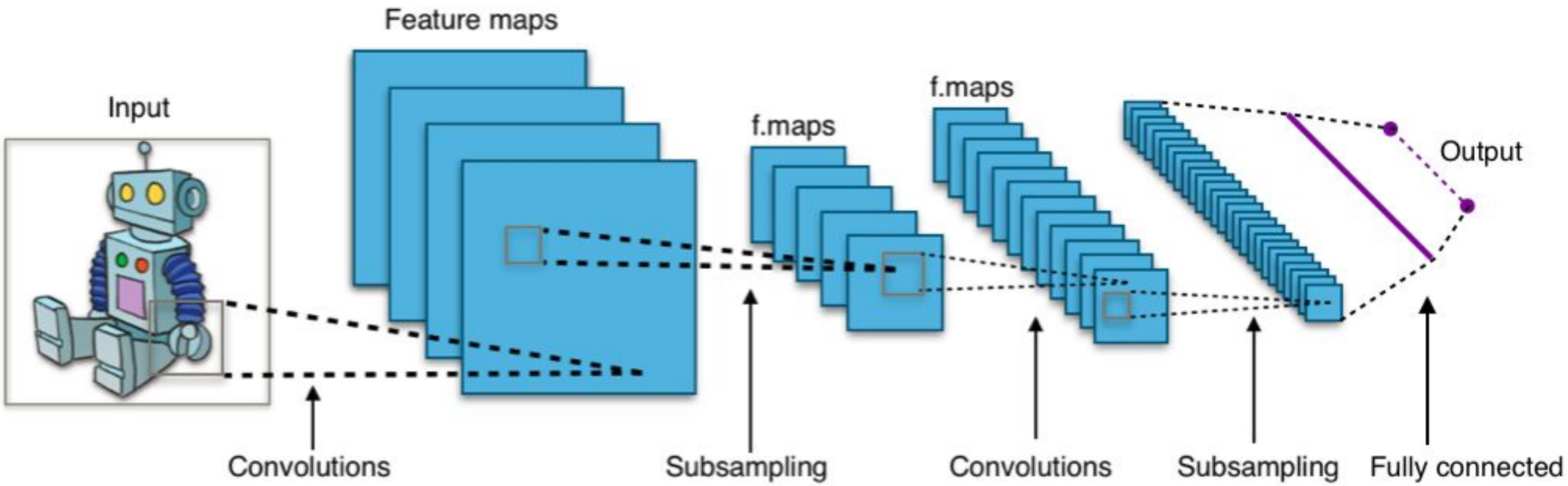
Interpretable counting for VQA

Alexander Trott, Caiming Xiong , & Richard Socher
Salesforce Research
Palo Alto, CA

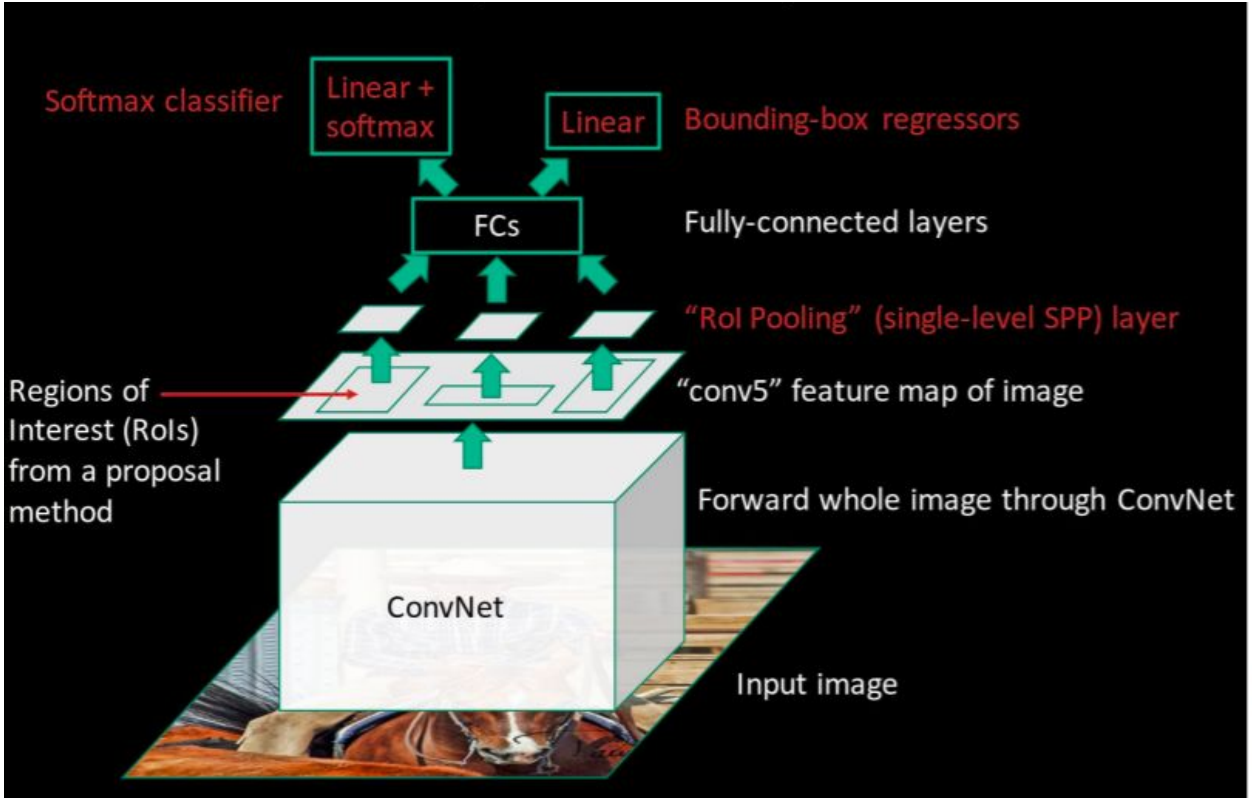
Presented by:
Anjali Shenoy,
MaLL Lab,
IISc Bangalore

Some basics..

A typical CNN



Object detection - Faster RCNN



Girschick, "Fast R-CNN", ICCV 2015

Slide credit: Ross Girschick

Countable VQA

To sequentially select from detected objects in images and learn interactions between objects that influence subsequent selections.

Dataset

VQA 2.0 dataset

- 1.1M questions pertaining to the 205K images from COCO

VQA 2.0 + Visual Genome dataset (108k images, ~50% belong to COCO)

↓ Filter based on counting question

HowMany-QA dataset

Examples of question-answer pairs that are excluded from HowMany-QA.

What time is on the clock?
ground truth = 9:35

What is the age of the man?
ground truth = 60

How many people does the jet seat?
ground truth = 200

What number is the batter?
ground truth = 59

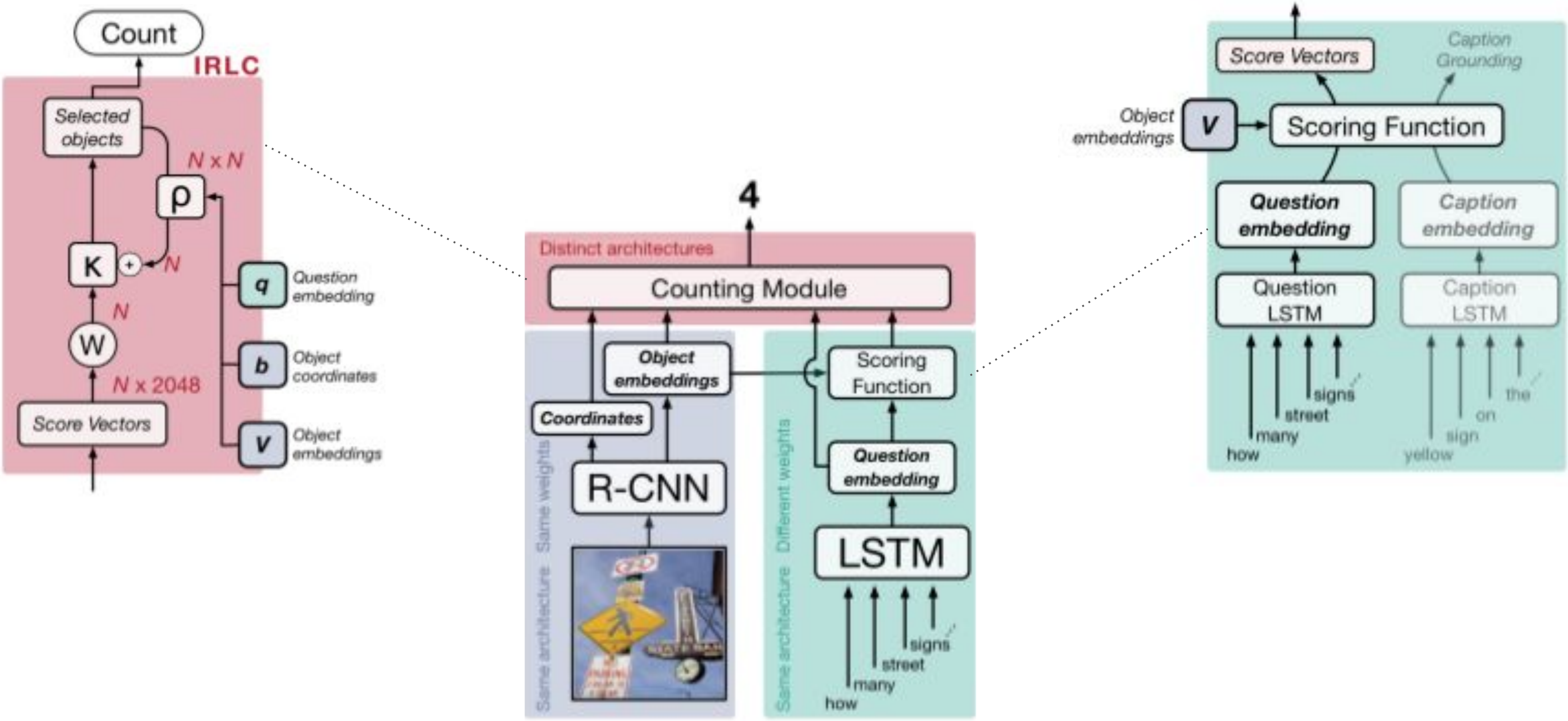
Dataset

Split	QA Pairs	Images
Train	83,642	31,932
from VQA 2.0	47,542	31,932
from VG	36,100	0
Dev.	17,714	13,119
Test	5,000	2,483

Table 1: Size breakdown of HowMany-QA. Neither development or test included VG data.

** Require that the ground-truth answer is a number between 0 to 20 (inclusive)

Model

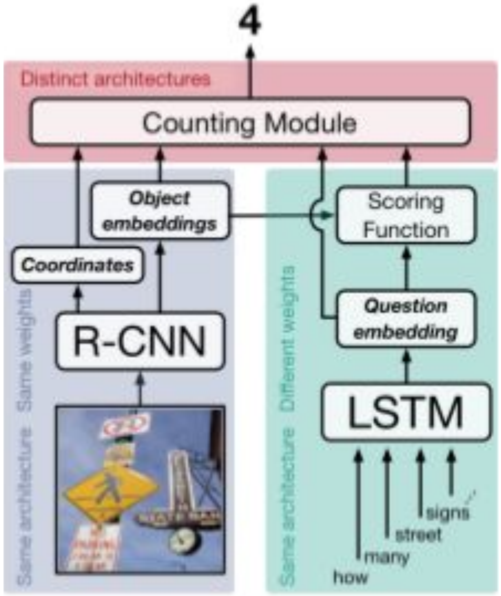


Count
IRLC
Selected objects
N x N
ρ
K
N
N
W
N x 2048
Score Vectors
q
Question embedding
b
Object coordinates
V
Object embeddings
4
Distinct architectures
Counting Module
Object embeddings
Coordinates
R-CNN
Same architecture Same weights
Scoring Function
Question embedding
LSTM
Same architecture Different weights
how many street signs
Score Vectors
Caption Grounding
Object embeddings
V
Scoring Function
Question embedding
Caption embedding
Question LSTM
Caption LSTM
how many street signs
yellow sign on the

Model

- FRCNN output
  - N detections
  - Bounding boxes $\{b_1, ..., b_N\}, b_i \in \mathbb{R}^4$
  - Object embeddings $\{v_1, ..., v_N\}, v_i \in \mathbb{R}^{2048}$
- LSTM
  - Final hidden state of LSTM $q = h^T$
  - For each detected object *i* we have a scoring function

$$s_i = f^S\left([q, v_i]\right)$$



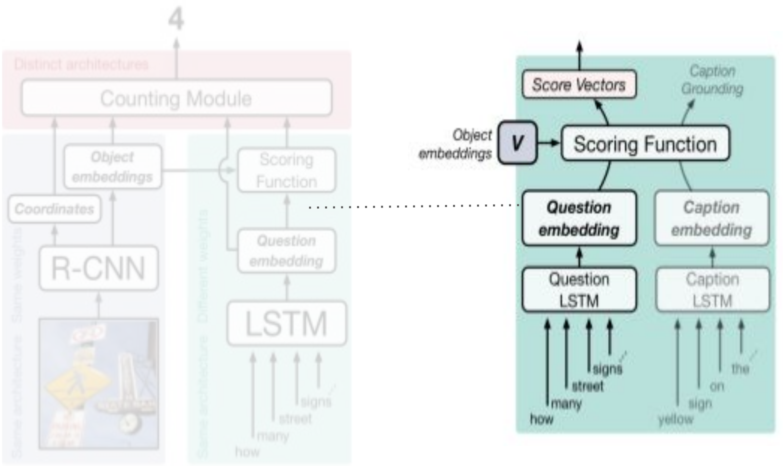
Model

<u>Training the scoring function</u>

Jointly train the scoring function along with LSTM to perform **caption grounding** for each region of image in Visual Genome.

Caption grounding

- to identify which object a given caption describes
- Trained similarly to Qs embedding
  - LSTM - encode caption
  - Scoring function f^S to encode relevance of object



Model

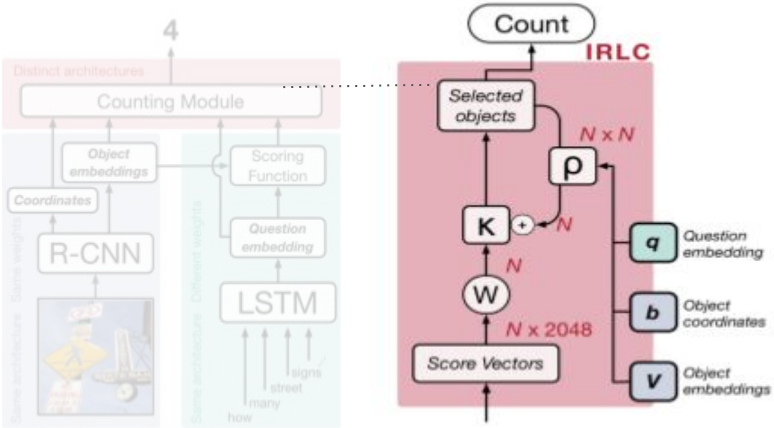
Counting

- Object scores for *N* objects $s \in \mathbb{R}^{N\mathrm{x}n}$ projected to vector of *logits* $\kappa \in \mathbb{R}^{N}$, representing how likely each object is to be counted

$$\kappa = W s + b$$

- Matrix of interaction terms $\rho \in \mathbb{R}^{N\mathrm{x}N}$ to update logits K
  - ρ_{ij} represents how selecting object *i* will change κ_j.

$$\rho_{ij} = f^{\rho}\left(\left[W q, \hat{v}_i^{\mathrm{T}} \hat{v}_j, b_i, b_j, \mathrm{IoU}_{ij}, \mathrm{O}_{ij}, \mathrm{O}_{ji}\right]\right)$$



Model

$$\rho_{ij} = \boxed{f^{\rho}}\left(\left[\boxed{Wq,}\ \boxed{\hat{v}_i^{\mathrm{T}}\hat{v}_j,}\ \boxed{b_i, b_j,}\ \boxed{\mathrm{IoU}_{ij}, \mathrm{O}_{ij}, \mathrm{O}_{ji}}\right]\right)$$

Qs Embedding

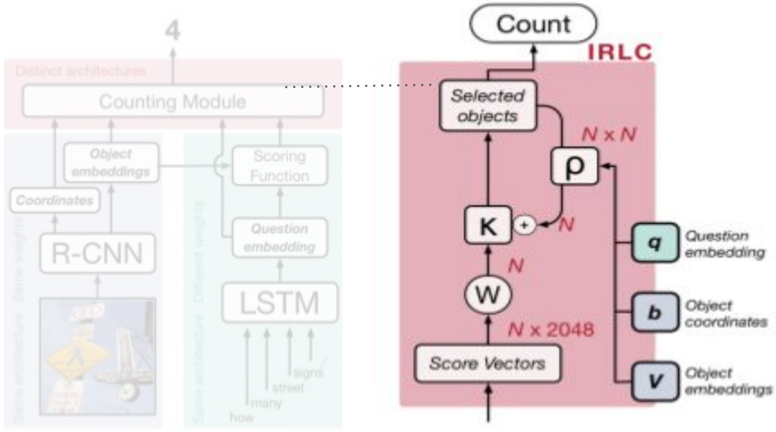
Object coordinates

Overlap statistics

$f^{\rho} : x \in \mathbb{R}^m$

2-layer MLP with ReLU

Dot product of normalised object vectors

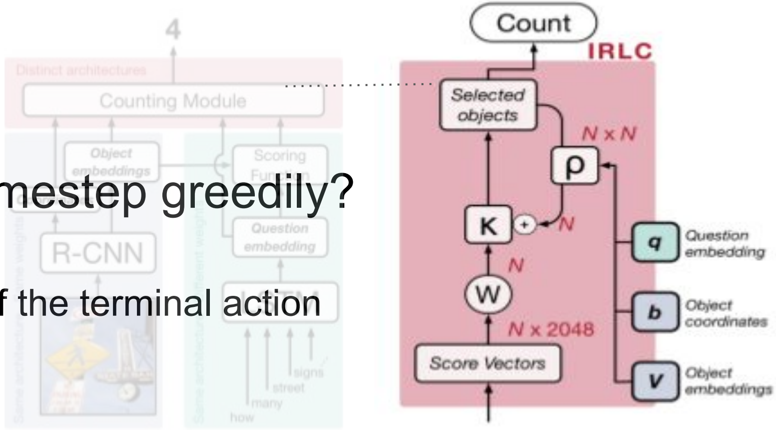


Model

Counting (at time step *t*)

- If a^t is index of selected object at time *t* selected using greedy algorithms $\rho(a^t, \cdot)$ represents the row of Interaction matrix for selected object then the update in the logits odds are

$$\kappa^{t+1} = \kappa^t + \rho(a^t, \cdot)$$



- How do we select the best object at current timestep greedily?
  - $a^t = \mathrm{argmax}_i \left[\kappa^t, \zeta\right]$
  - ζ Is a learnable scalar representing the logit value of the terminal action
  - κ^0 Is the result of $\kappa = Ws + b$

Model

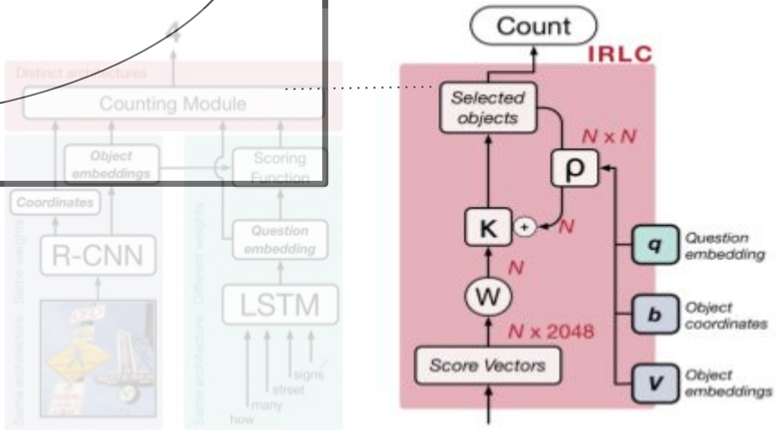
Counting

$$\kappa^{t+1} = \kappa^{t} + \rho(a^{t}, \cdot)$$

$$\rho_{ij} = f^{\rho}\left(\left[Wq, \hat{v}_i^{\mathrm{T}}\hat{v}_j, b_i, b_j, \mathrm{IoU}_{ij}, \mathrm{O}_{ij}, \mathrm{O}_{ji}\right]\right)$$

$$a^{t} = \mathrm{argmax}_i \left[\kappa^{t}, \zeta\right]$$

$$\kappa^{0} = Ws + b$$



Terminate when terminal is selected

Model

Try softmax instead of argmax to smoothen training curve

$$p^t = \mathrm{softmax}\left(\left[\kappa^t, \zeta\right]\right) \qquad a^t \sim p^t$$
$$\kappa^{t+1} = \kappa^t + \rho(a^t, \cdot)$$

Model

Loss (Reinforcement Learning Theory)

- Count error $E = |C - C^{\mathrm{GT}}|$
- Reward $R = E^{\text{greedy}} - E$ (baseline count error obtained by greedy action selection)
- 3 Losses
  - $\tilde{L}_C = -R \sum_t \log p^t\left(a^t\right)$ which is variation of a policy gradient
  -
  - Total negative policy entropy *H* $\tilde{P}_{\mathrm{H}} = -\sum H\left(p^t\right)$ (is a common strategy when using policy gradient) and is used to improve exploration
  -
  - Average interaction strength $\tilde{P}_{\mathrm{I}} = \sum_{i \in \{a^0 \ldots a^t\}} \frac{1}{N} \sum_j L_1\left(\rho_{ij}\right)$ where L_1 is Huber Los. The interaction penalty is motivated by the a priori expectation that interactions should be sparse.

Baselines

Soft Count

$$C = \sum_i \sigma\left(W s_i\right)$$

Project its score vector to a scalar value and apply a sigmoid nonlinearity, denoted as σ, to assign the object a count value between 0 and 1 and trained using Huber Loss

$$L_1 = \begin{cases} 0.5e^2 & \text{if } e \leq 1 \\ e - 0.5 & \text{otherwise} \end{cases} \qquad e = |C - C^{\text{GT}}|$$

Attention Baseline

$$\alpha = \text{softmax}\left(W s\right); \quad \hat{v} = \sum \alpha_i v_i$$

Learn attention weights based on score vector

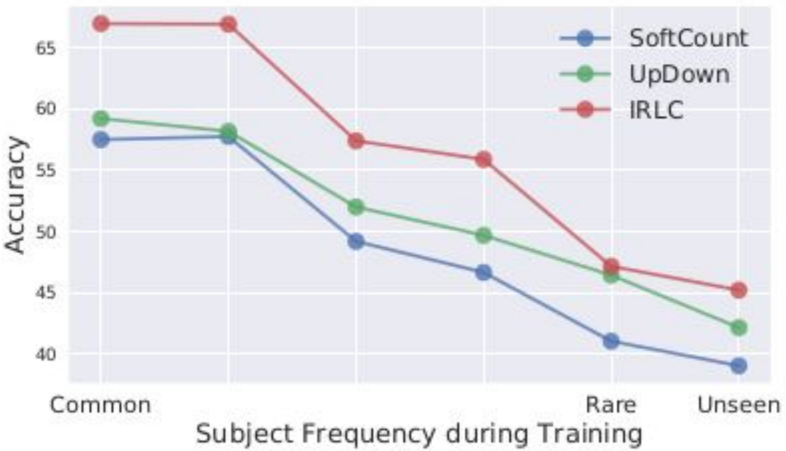
Then get the count as:

$$v' = f^V\left(\hat{v}\right); \quad q' = f^Q\left(q\right)$$
$$p = \text{softmax}\left(f^C\left(v' \otimes q'\right)\right)$$

At test time, use the most probable count given by p (from 1-20).

Results

Accuracy and RMSE used as metric



Results

Model	Accuracy	RMSE
SoftCount	50.2 (49.2)	**2.37** (2.45)
UpDown	52.7 (51.5)	2.64 (2.69)
IRLC	**57.7** (56.1)	**2.37** (2.45)

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \left(P_i - O_i \right)^2}{n}}$$

Sample failure cases

Mainly if object detection fails

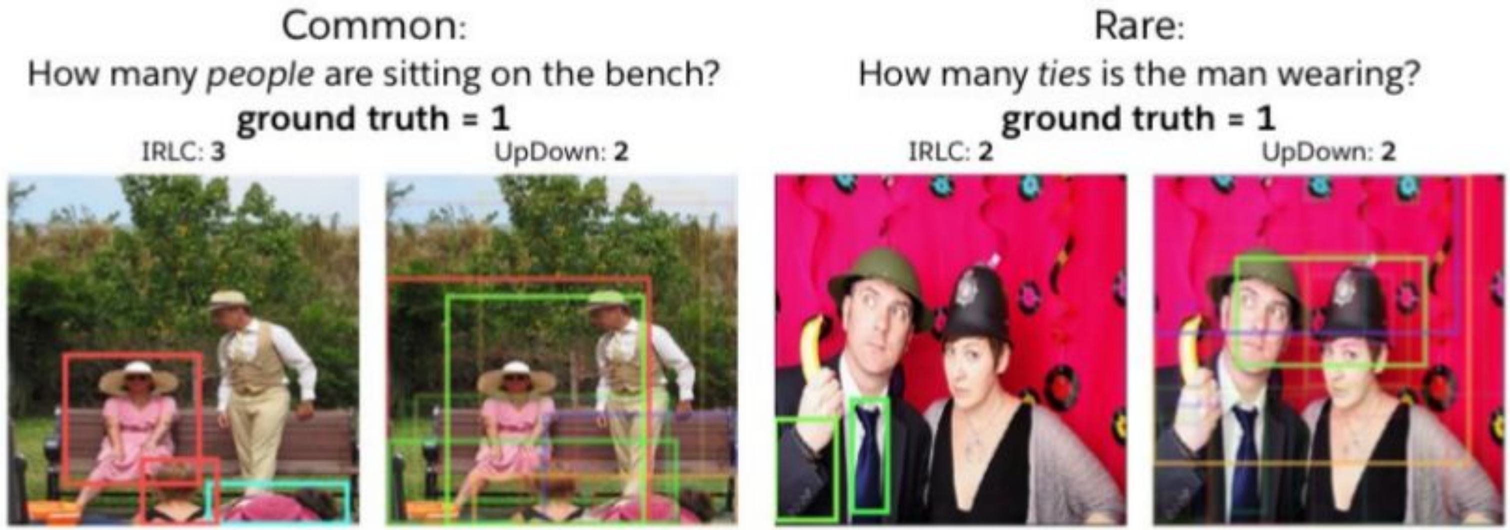


Figure 7: Examples of failure cases with common and rare subjects ("people" and "ties," respectively). Each example shows the output of IRLC, where boxes correspond to counted objects, and the output of UpDown, where boxes are shaded according to their attention weights.


That's all Folks!