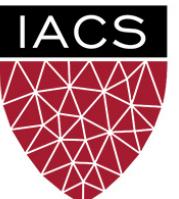
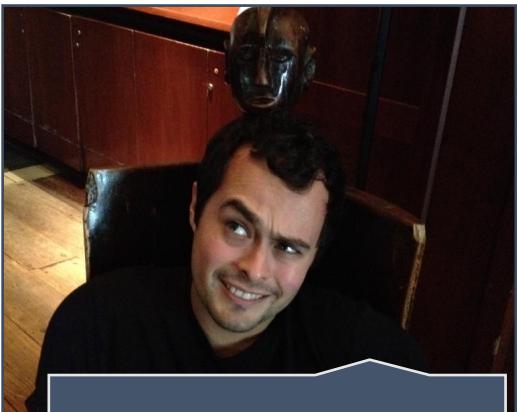


Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

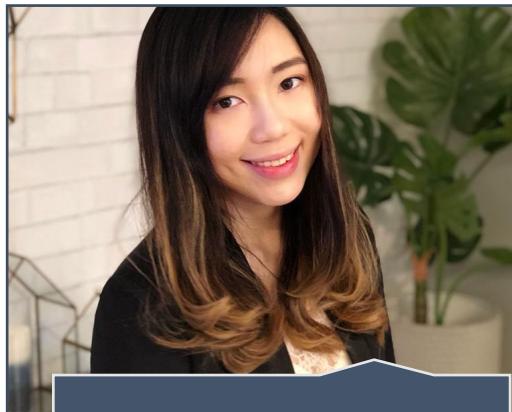
αβnormal Distribution
AC295/CS115



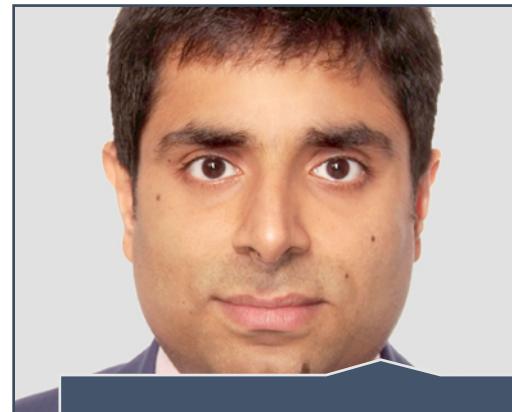
$\alpha\beta$ normal Distribution



Eduardo Peynetti



Jessica Wijaya

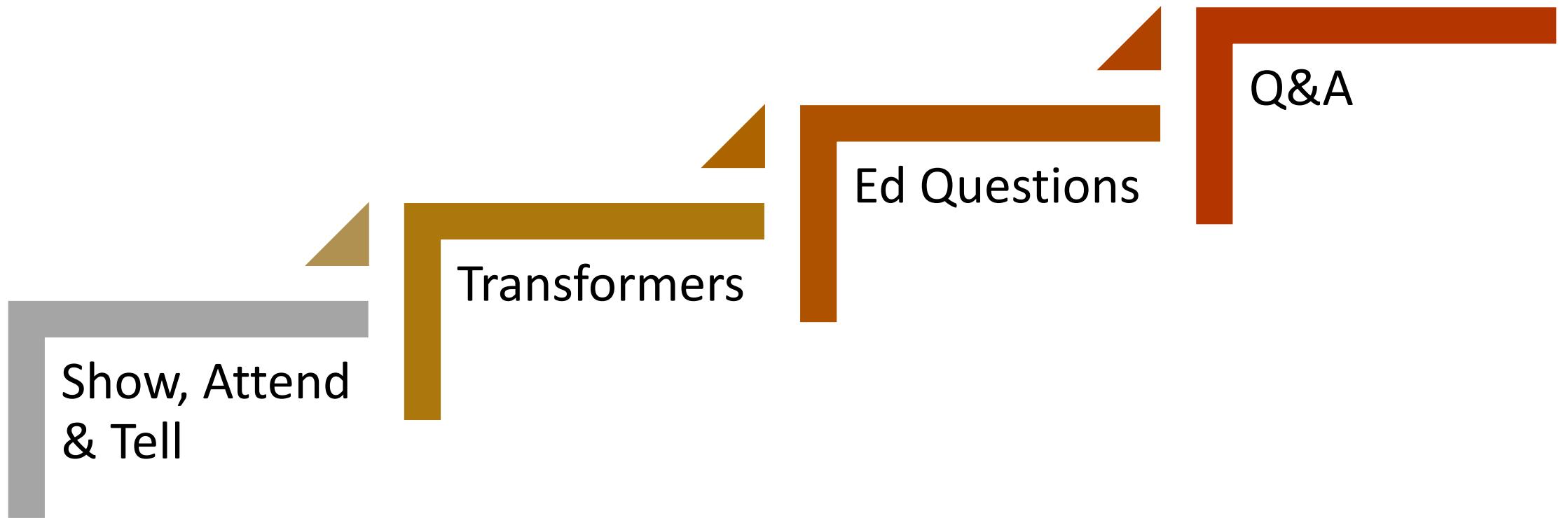


Rohit Beri



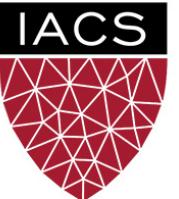
Stuart Neilson

Contents



Show, Attend and Tell

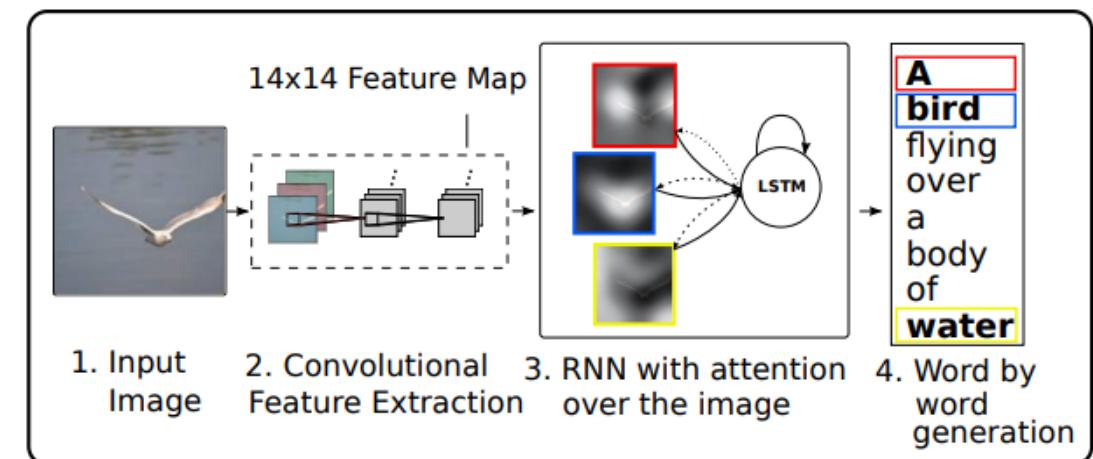
Neural Image Caption Generation with Visual Attention



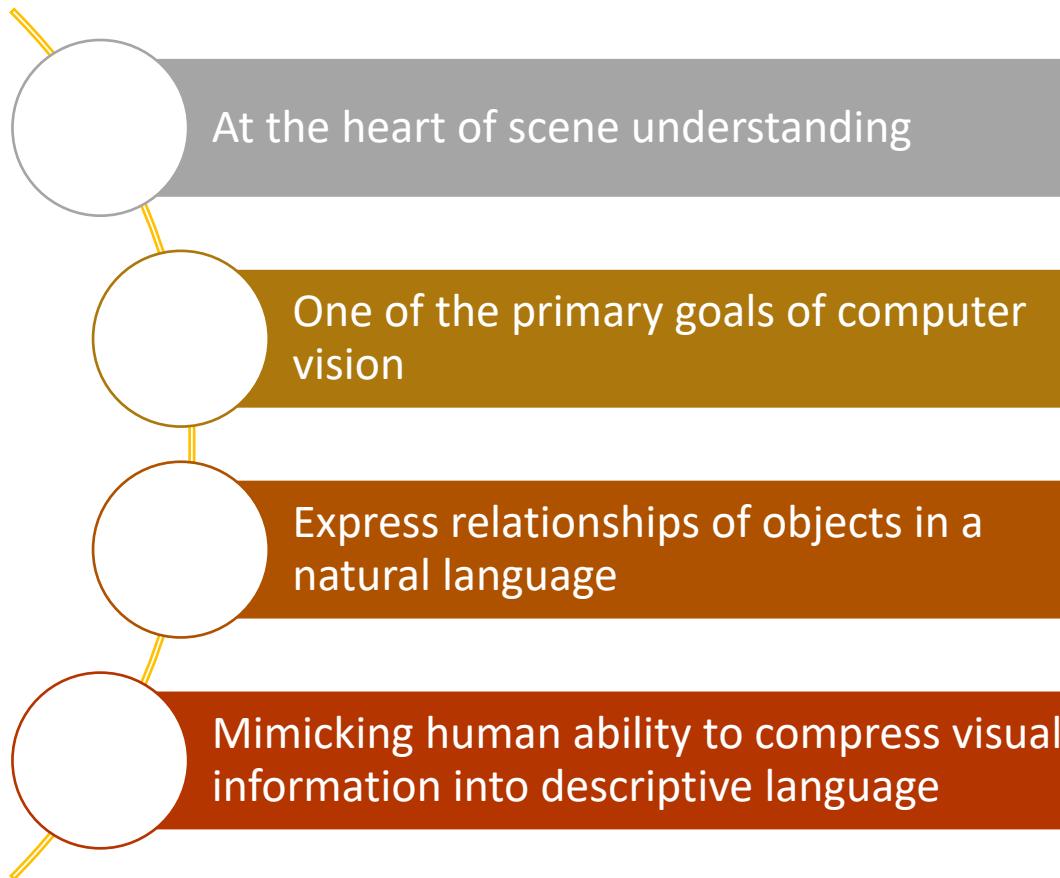
Show, Attend and Tell

Attention-based Model

- Automatically learns to describe the content of the images
- Two ways to train the model
- State-of-the-art performance on three different datasets



Automatic Caption Generation



A stop sign is on a road with a mountain in the background.



A giraffe standing in a forest with trees in the background.

Attention



One of the most curious facets of human visual system

Allows for salient features to dynamically come to the forefront as needed
Particularly important when there is lot of clutter



Top-layer representations distill information to most salient objects

Loss of information required for richer and more descriptive captions
Low-level representations can help preserve this information



Two attention-based image caption generators

Soft deterministic attention
Hard stochastic attention



Timeline – Image Captioning/Visual Attention

2011

- Generating caption templates which were filled in based on the results of object detections and attribute discovery



2012

- First retrieving similar captioned images for a large dataset and then modifying these retrieved captions to fit the query

2014

- Flurry of Neural Networks based approaches
- FFNN, RNN, LSTM & R-CNN based approaches

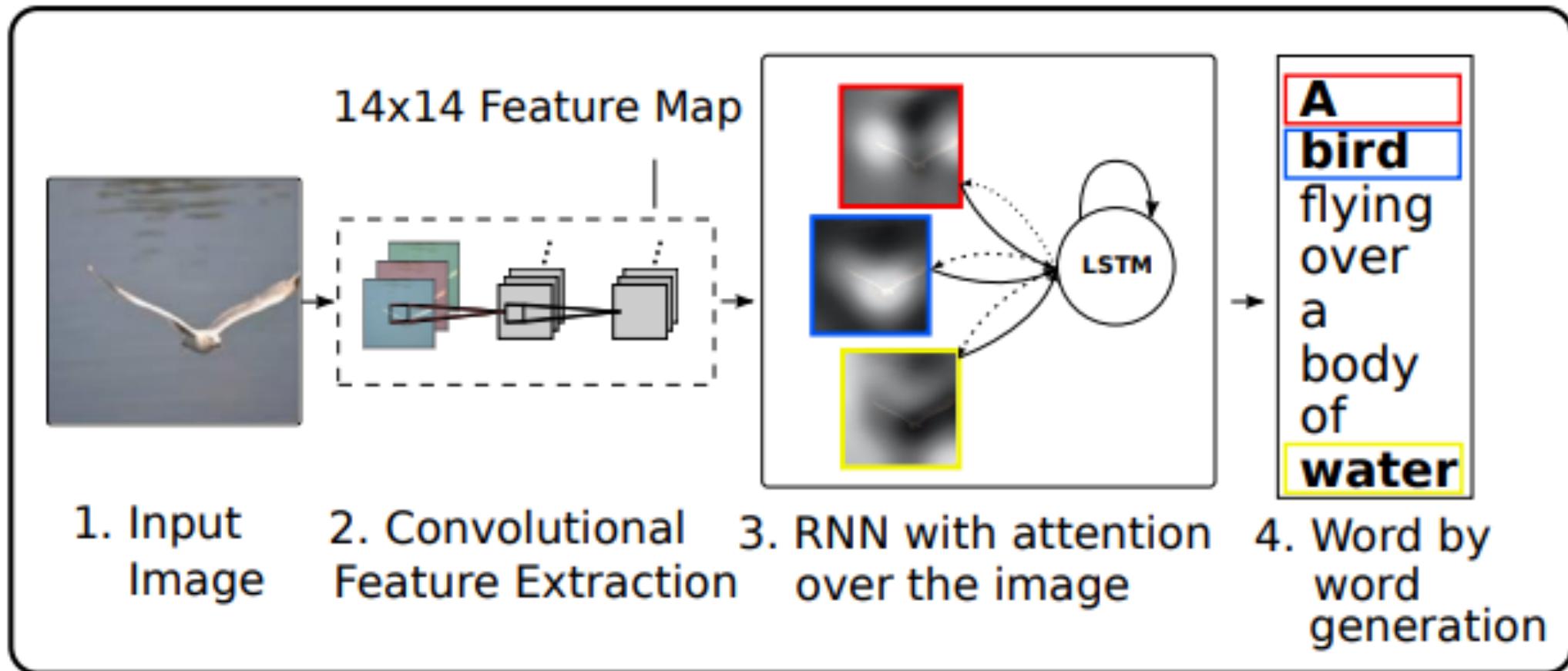
2016

- **Show, Attend and Tell**
- **Neural image caption generation with visual attention**

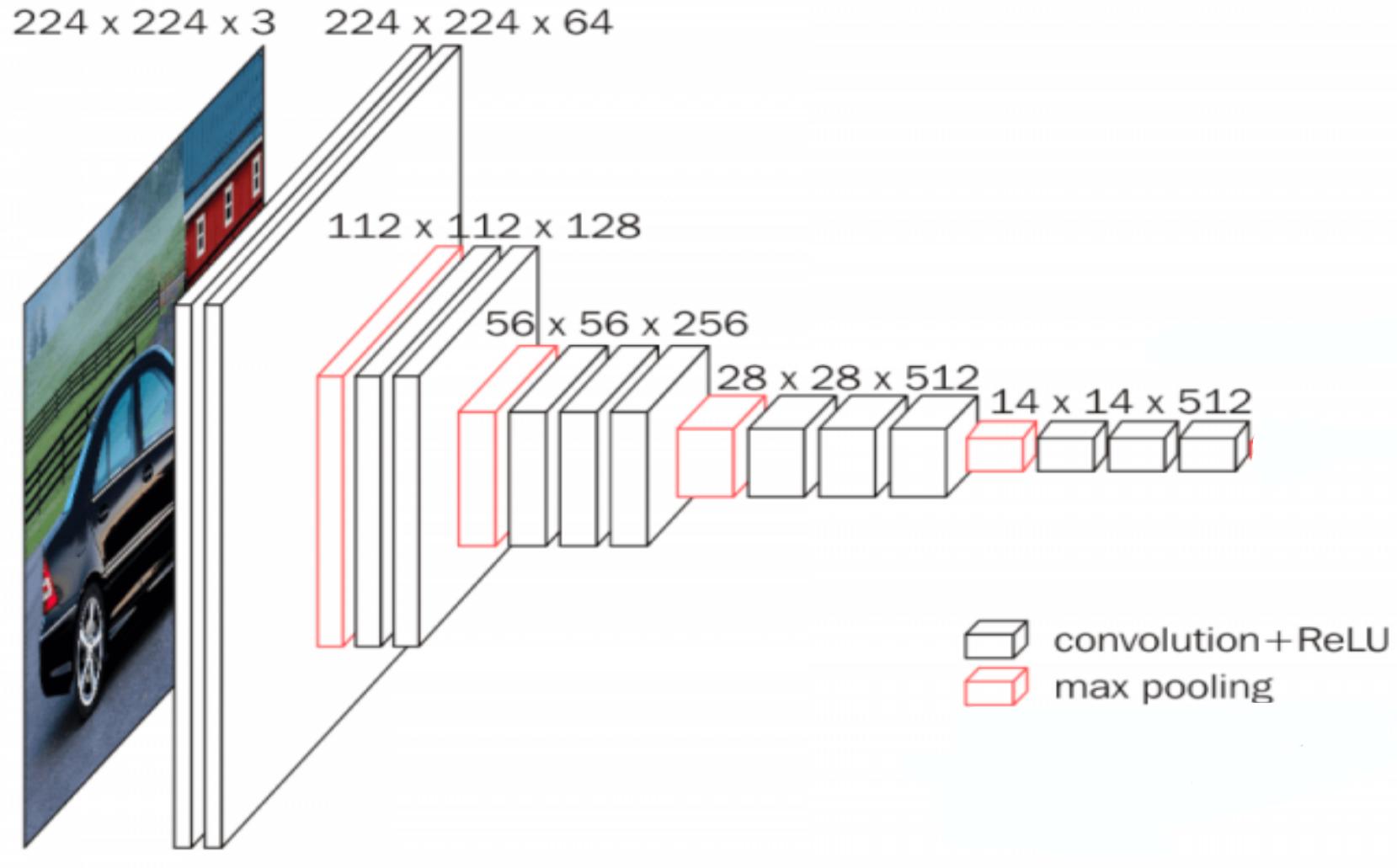
2020

- *An Image is worth 16 x 16 words*
- *Transformer based architecture for image recognition*

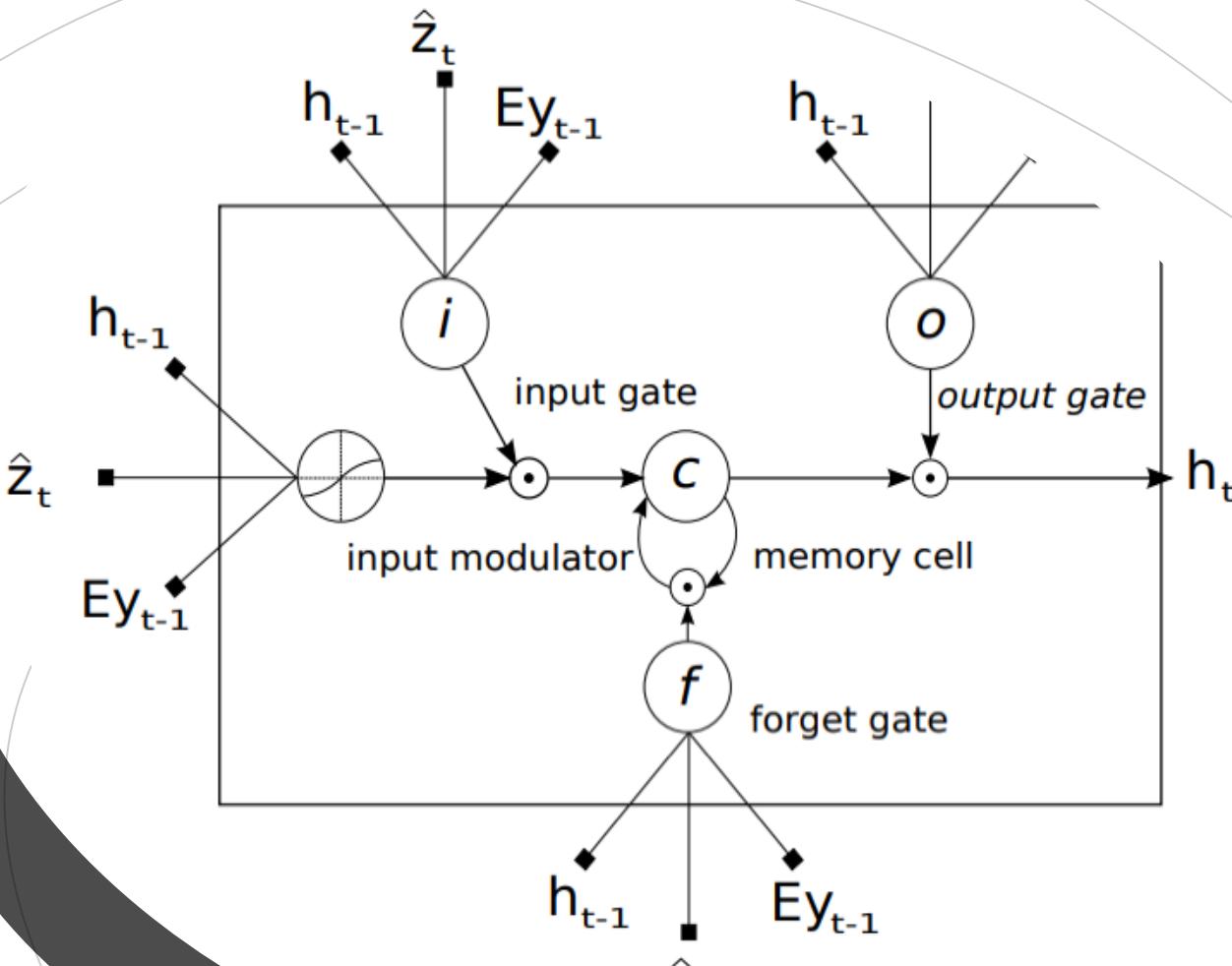
Image Caption Generation – Model Overview



Encoder – Oxford VGG



Decoder – LSTM with Attention



Decoder – LSTM with Attention

LSTM generates one word at every time step, ***conditioned on:***

Context vector z_t

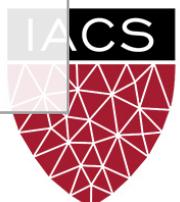
Previous hidden state h_{t-1}

Previously generated word y_{t-1}

LSTM

$$\bullet \begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n, n} \begin{pmatrix} E y_{t-1} \\ h_{t-1} \\ \hat{z}_t \end{pmatrix}$$

- $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
- $h_t = o_t \odot \tanh(c_{t-1})$



Decoder – LSTM with Attention

Initial
Memory
State and
Hidden State

- Average of the Annotation Vectors

Weight α_i of each Annotation Vector a_i

- Computed by the attention model f_{att}
 - MLP conditioned on the h_{t-1} .

$$c_0 = f_{init,c} \left(\frac{1}{L} \sum_i^L a_i \right)$$

$$h_0 = f_{init,h} \left(\frac{1}{L} \sum_i^L a_i \right)$$

$$e_{ti} = f_{att} (a_i, h_{t-1})$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^L \exp(e_{tk})}$$

Decoder – LSTM with Attention

Context Vector:
 \hat{z}_t

- Dynamic representation of the relevant part of the image at time t

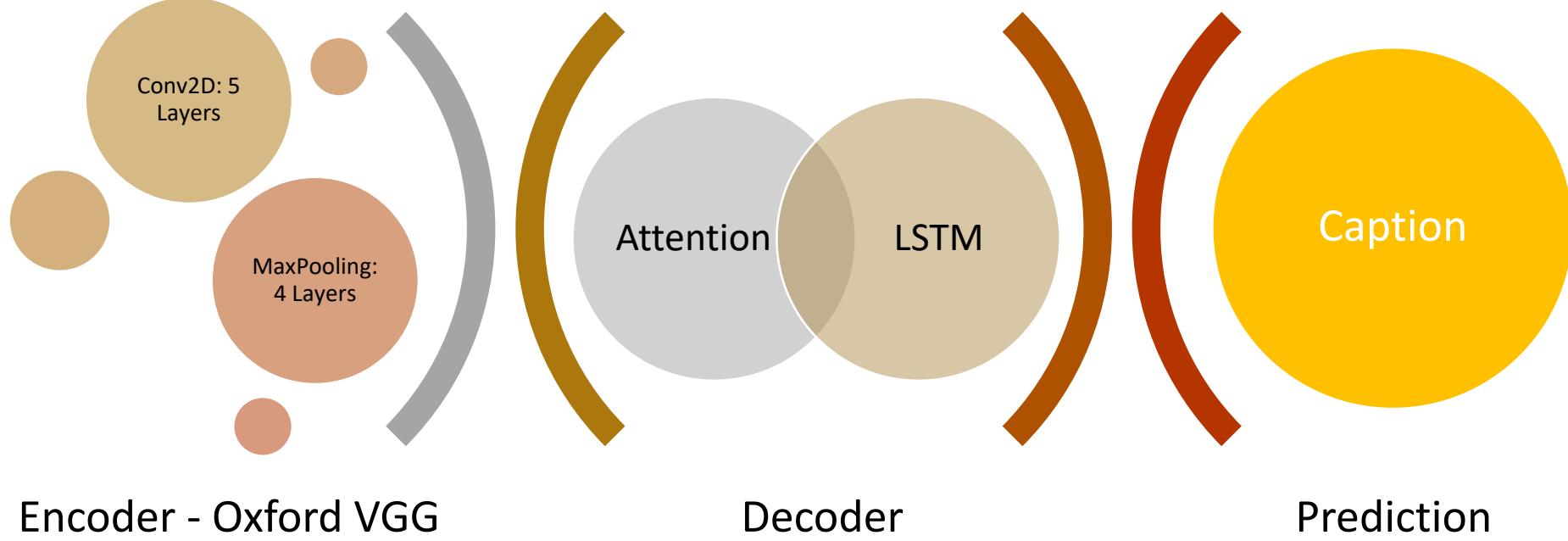
Mechanism: ϕ

- $\hat{z}_t = \phi(\{a_i\}, \{\alpha_i\})$
- Meaning of α_i
 - Probability: Hard attention
 - Relative Importance: Soft attention

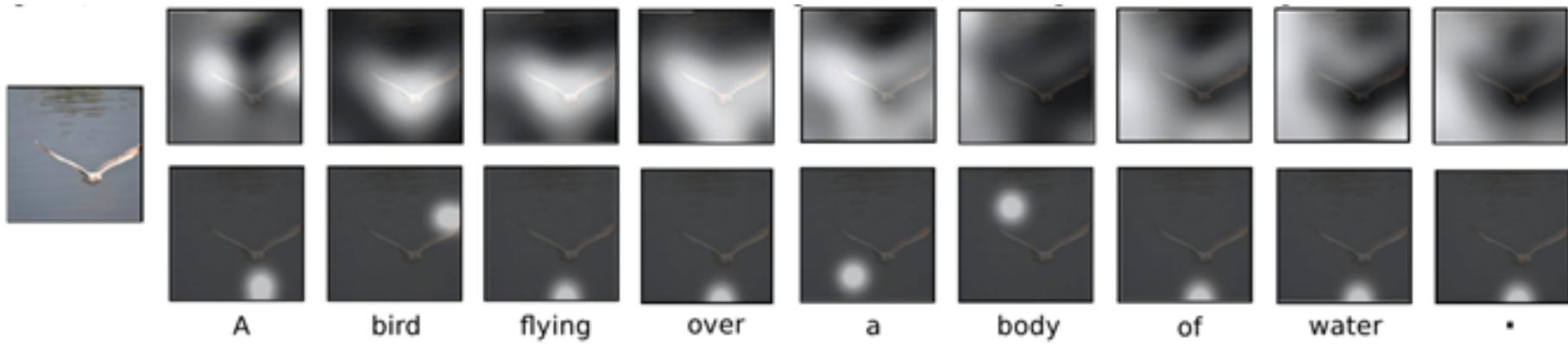
Deep output
layer

- $p(y_t | a, y_1^{t-1}) \propto e^{(L_0(Ey_{t-1} + L_h h_t + L_z \hat{z}_t))}$
- Probability of the output word
- Where: $L_0 \in \mathbb{R}^{K \times m}$, $L_h \in \mathbb{R}^{m \times n}$, $L_z \in \mathbb{R}^{m \times D}$, and E are learned parameters

Show, Attend & Tell – Architecture



Soft vs Hard Attention



Soft Attention

ϕ returns the expectation of the context vector

Differentiable and amenable to standard backpropagation

Hard Attention

ϕ returns a_i at every time step using multinouilli distribution sampling

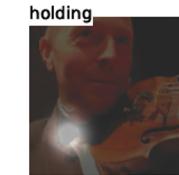
Monte-Carlo based sampling to compute the gradient

Hard (Stochastic) Attention

Correct Captioning



Incorrect Captioning



Hard (Stochastic) Attention

Objective function: L_s

- $\log p(y|a)$
 - Likelihood of observing words y given image feature a
- Hard to compute
 - Optimize the variational lower bound instead

Latent Variable: s_t

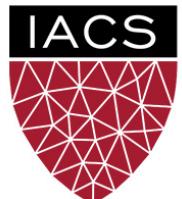
- Location where the model focuses on when generating the t^{th} word

$$\begin{aligned} L_s &= \sum_s p(s|a) \log p(y|s, a) \\ &\leq \log \sum_s p(s|a) p(y|s, a) \\ &= \log p(y|a) \end{aligned}$$

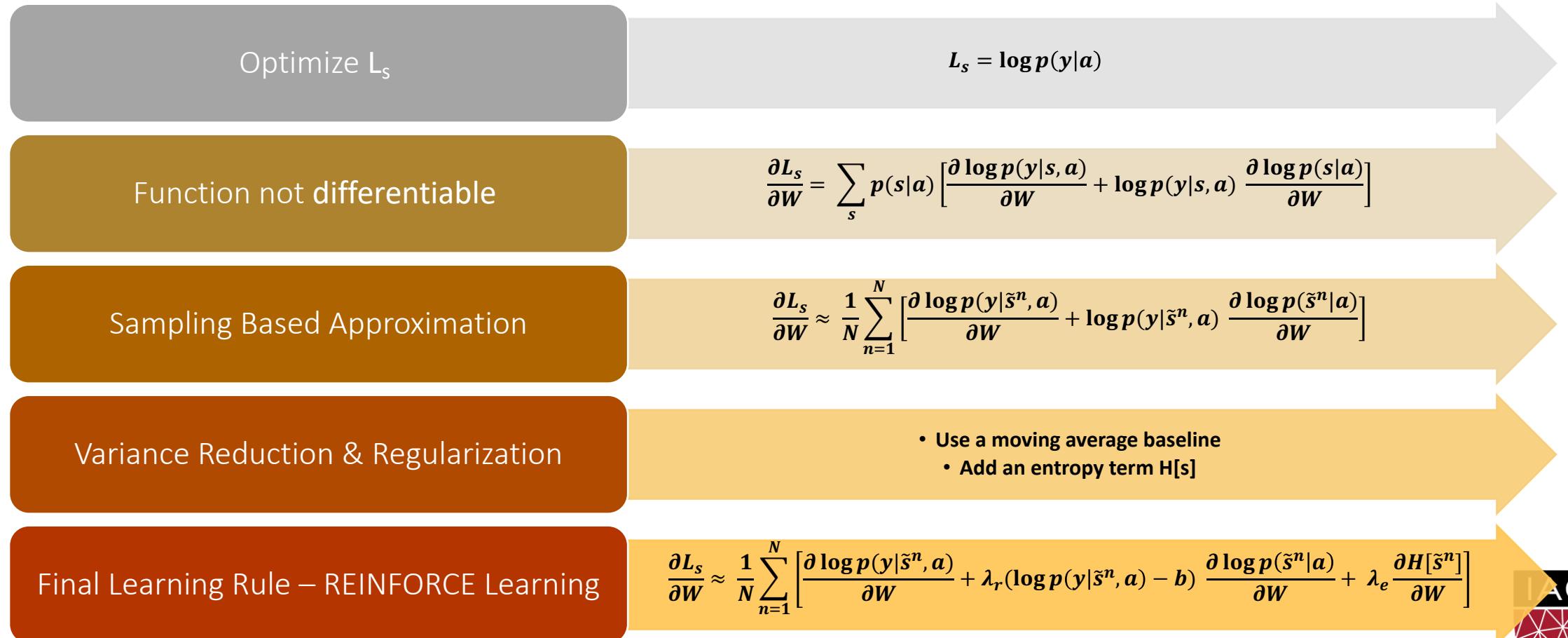
$$\tilde{s} \sim \text{Multinouilli}_L(\{\alpha_i\})$$

$$p(s_{t,i} = 1 | s_{j < t}, a) = \alpha_{t,i}$$

$$\hat{z}_t = \sum s_{t,i} a_i$$

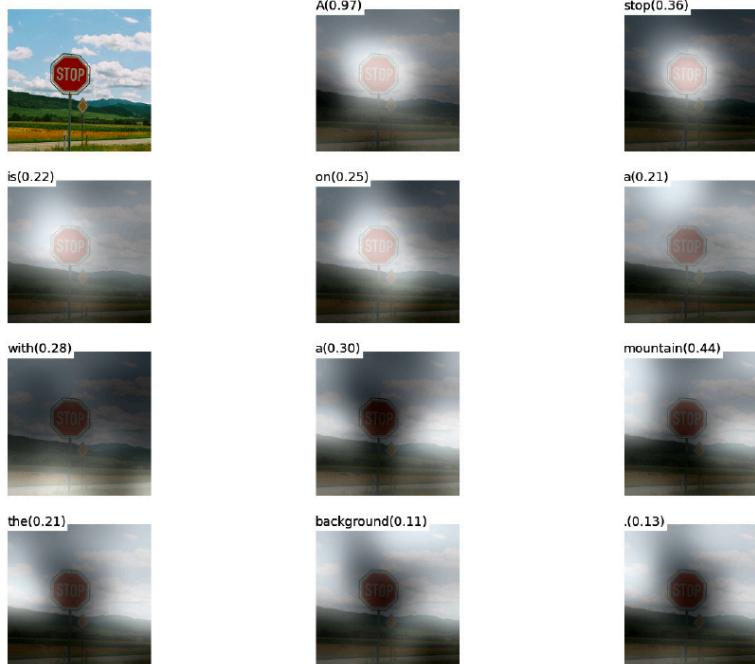


Hard (Stochastic) Attention

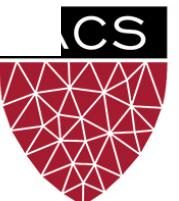
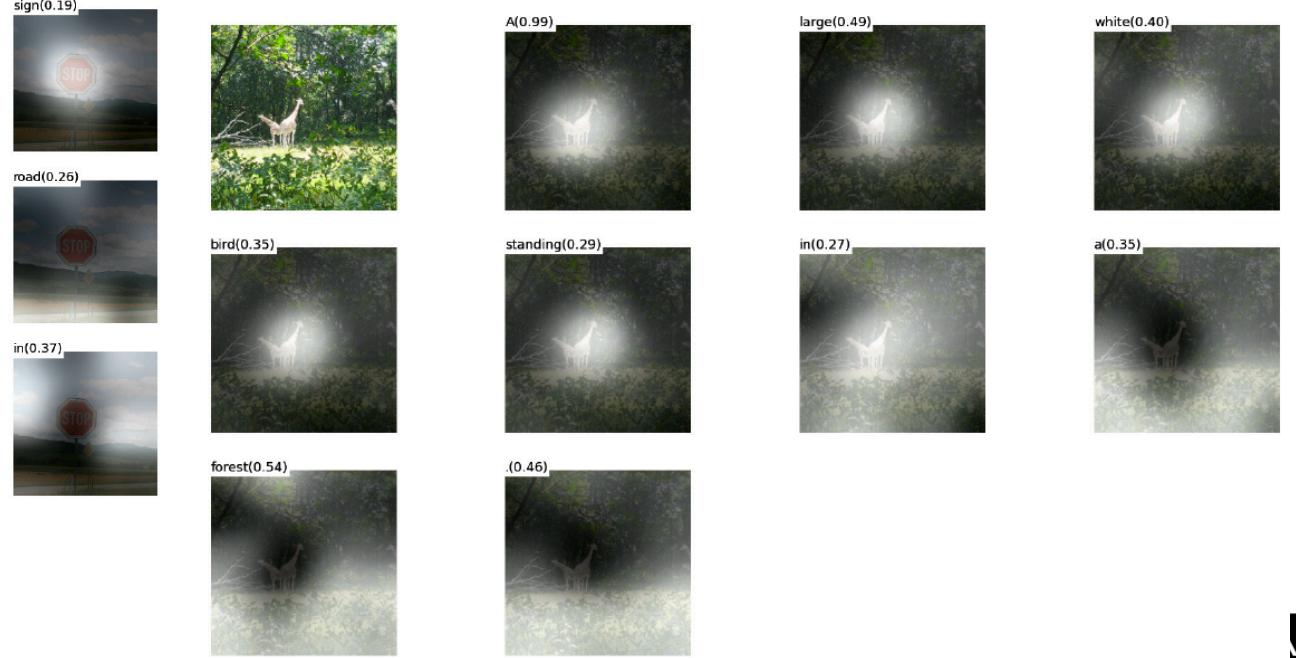


Soft (Deterministic) Attention

Correct Captioning



Incorrect Captioning



Soft (Deterministic) Attention

Objective function: L_s

- $\log p(y|a)$
 - Likelihood of observing words y given image feature a
- Model is differentiable and amenable to backpropagation

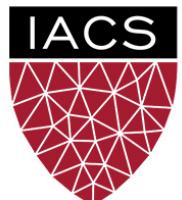
$$L_s = \log p(y|a)$$

Context Vector: \hat{z}_t

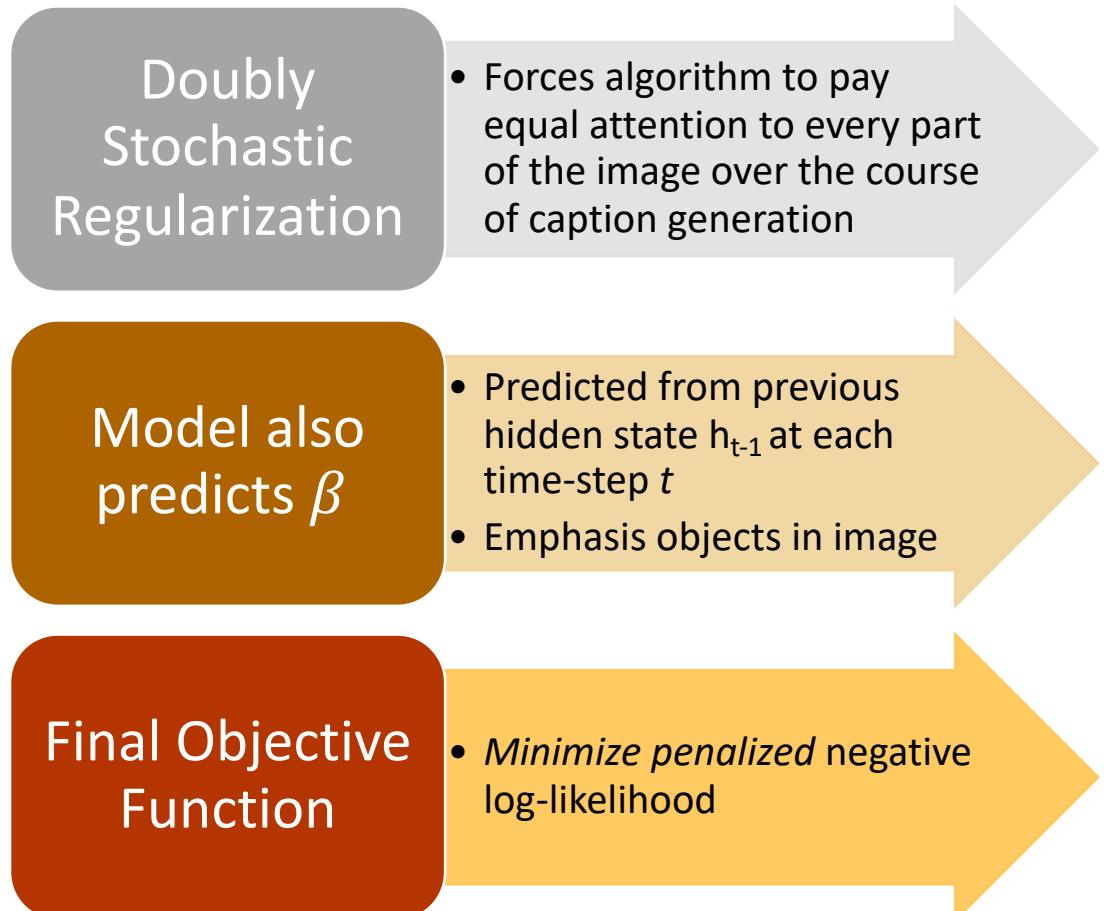
- Directly computes the expectation of the context vector

$$\hat{z}_t = \phi(\{a_i\}, \{\alpha_i\})$$

$$\mathbb{E}_{p(s_t|a)}[\hat{z}_t] = \beta_t \sum_{i=1}^L \alpha_{t,i} a_i$$



Soft (Deterministic) Attention



Encourage $\sum_t \alpha_{ti} \approx 1$ in addition to $\sum_i \alpha_{ti} = 1$

$$\begin{aligned}\widehat{z}_t &= \phi(\{a_i\}, \{\alpha_i\}) = \beta_t \sum_{i=1}^L \alpha_{t,i} a_i \\ \beta_t &= \sigma(f_\beta(h_{t-1}))\end{aligned}$$

$$L_d = -\log(P(y|x)) + \lambda \sum_i^L \left(1 - \sum_t^c \alpha_{ti}\right)^2$$

Datasets

Flickr8k

8,000 images

5 reference
sentences per
image

Flickr30k

30,000 images

5 reference
sentences per
image

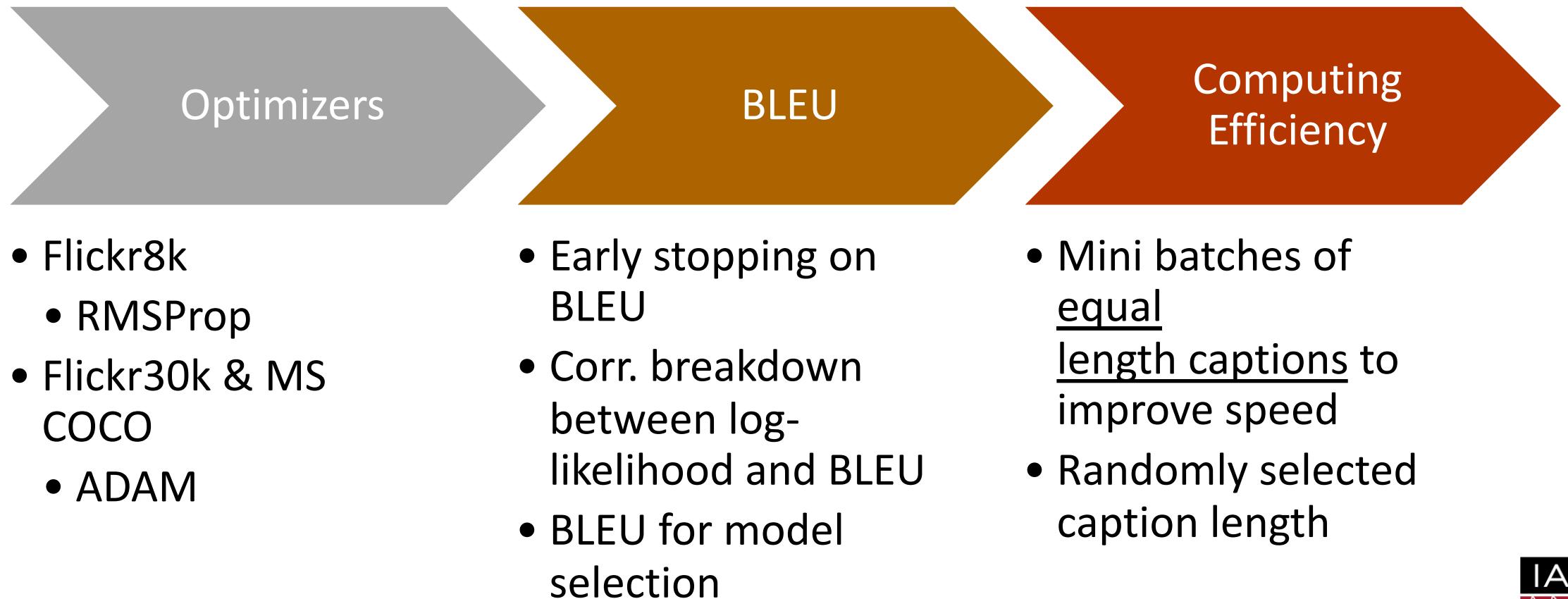
Microsoft COCO

82,783 images

Some images
have >5 reference
sentences



Training



Quantitative Evaluation

Table 1. BLEU-1,2,3,4/METEOR metrics compared to other methods, \dagger indicates a different split, (—) indicates an unknown metric, \circ indicates the authors kindly provided missing metrics by personal communication, Σ indicates an ensemble, a indicates using AlexNet

Dataset	Model	BLEU				METEOR
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	
Flickr8k	Google NIC(Vinyals et al., 2014) $^{\dagger\Sigma}$	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a) $^\circ$	65.6	42.4	27.7	17.7	17.31
	Soft-Attention	67	44.8	29.9	19.5	18.93
	Hard-Attention	67	45.7	31.4	21.3	20.30
Flickr30k	Google NIC $^{\dagger\circ\Sigma}$	66.3	42.3	27.7	18.3	—
	Log Bilinear	60.0	38	25.4	17.1	16.88
	Soft-Attention	66.7	43.4	28.8	19.1	18.49
	Hard-Attention	66.9	43.9	29.6	19.9	18.46
COCO	CMU/MS Research (Chen & Zitnick, 2014) a	—	—	—	—	20.41
	MS Research (Fang et al., 2014) $^{\dagger a}$	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014) $^\circ$	64.2	45.1	30.4	20.3	—
	Google NIC $^{\dagger\circ\Sigma}$	66.6	46.1	32.9	24.6	—
	Log Bilinear $^\circ$	70.8	48.9	34.4	24.3	20.03
	Soft-Attention	70.7	49.2	34.4	24.3	23.90
	Hard-Attention	71.8	50.4	35.7	25.0	23.04

Qualitative Analysis - Visualization



A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

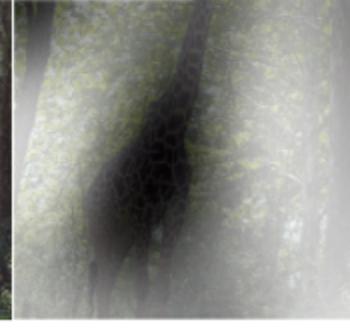
A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.



A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.

Conclusion & Future of Image Captioning

Conclusion

Alignments correspond well to human intuition

“Attention” can help with the model interpretation

Future

Transformers

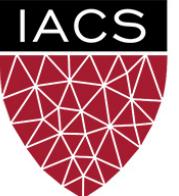
Can we replace CNNs and LSTMs with Transformers?



An Image is worth 16×16 words

Transformers for Image Recognition at Scale

Conference paper for the 2021 ICLR conference, published on ArXiv last week (an earlier version in which the author names were redacted for blind review was up a few weeks earlier).



State of the Art

Natural Language Processing

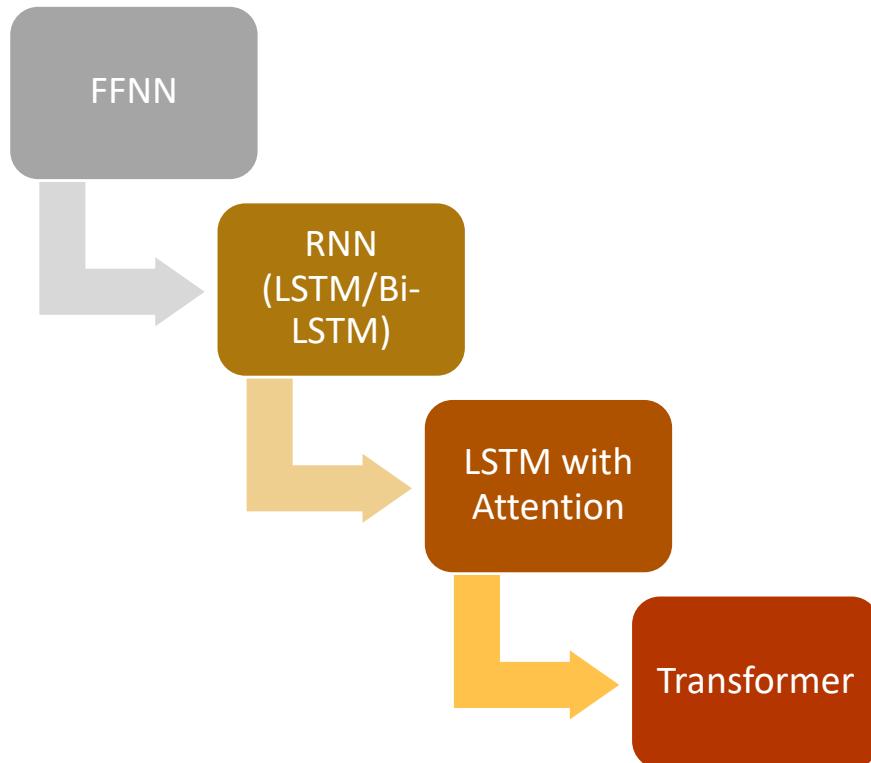
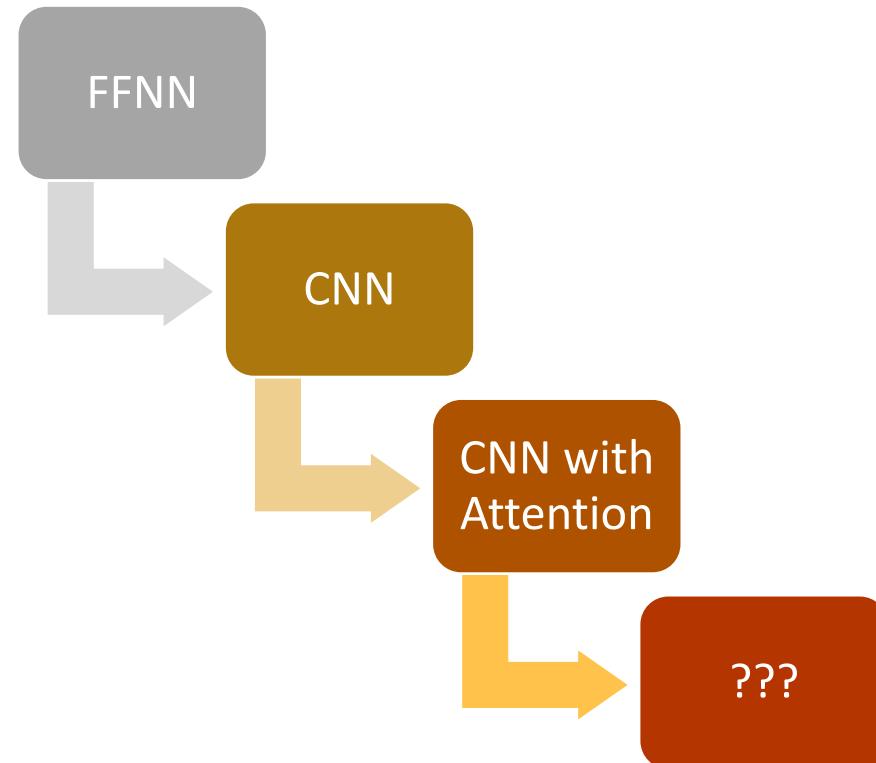
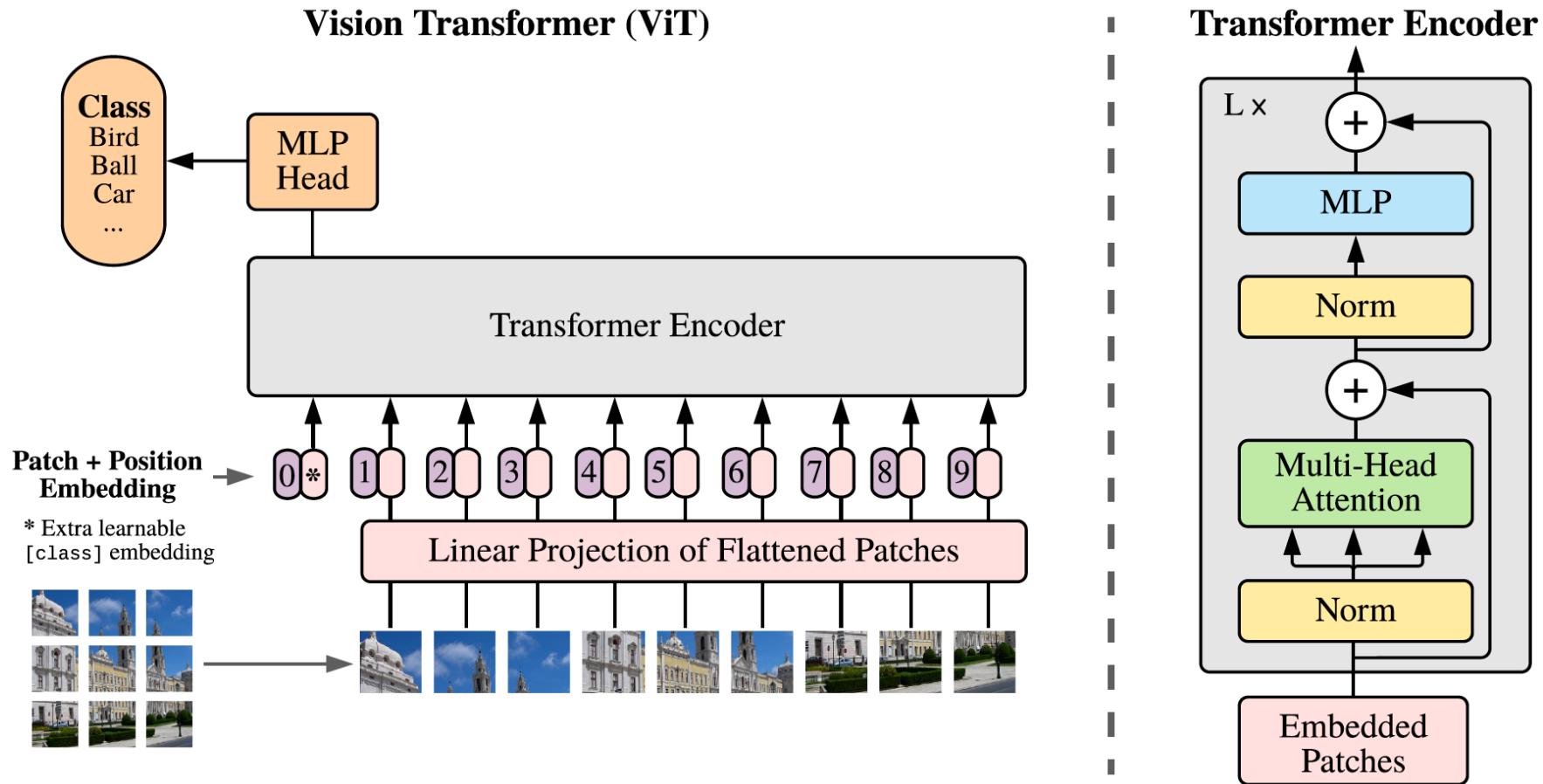


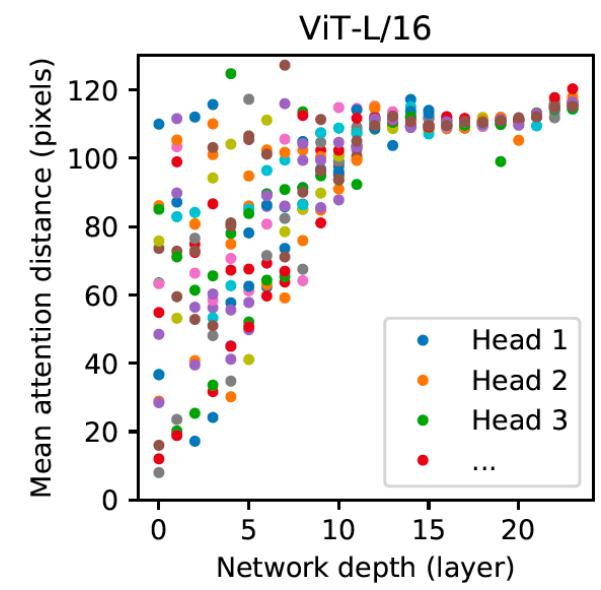
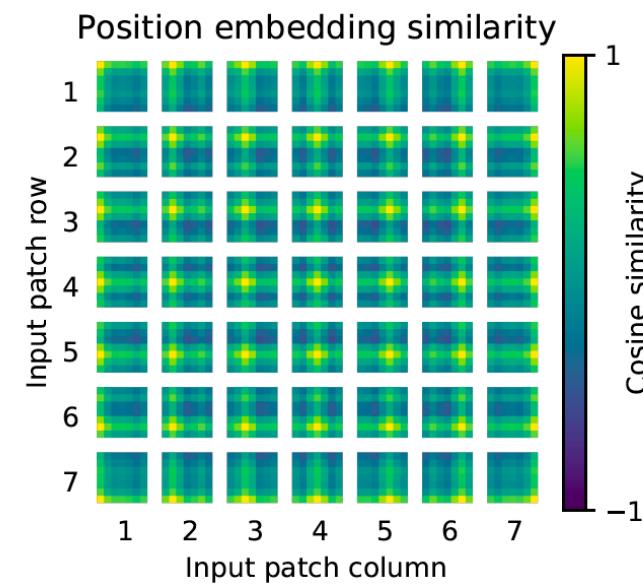
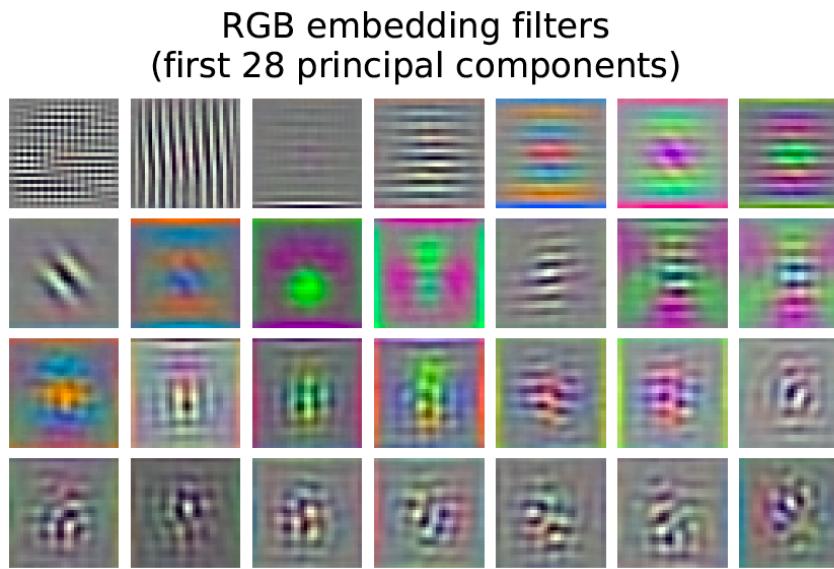
Image Recognition



Vision Transformer (ViT)



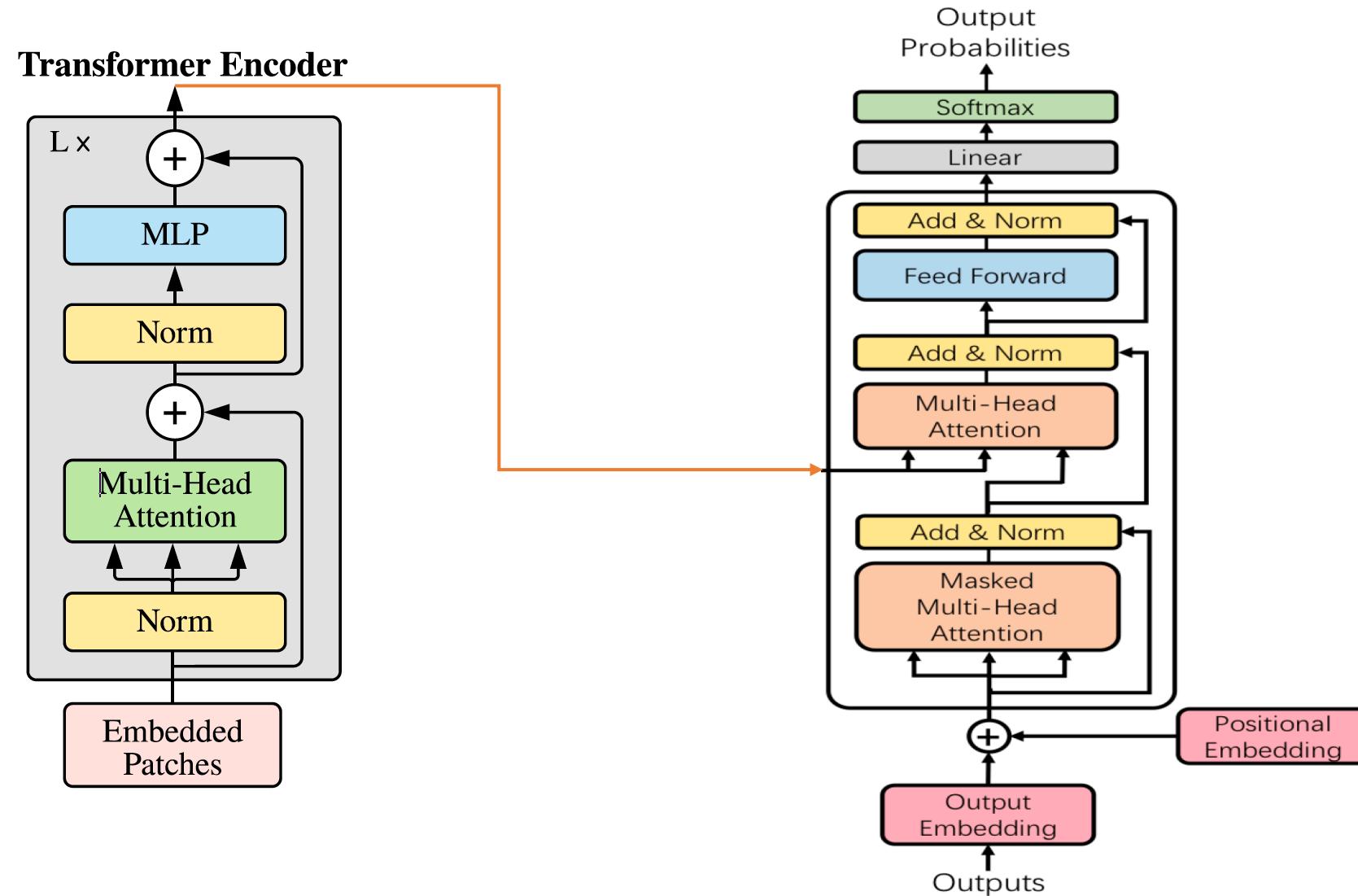
ViT: Learning similar features as CNN



ViT vs SOTA - Performance

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21K (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	88.55 ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	90.72 ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	99.50 ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	—
CIFAR-100	94.55 ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	—
Oxford-IIIT Pets	97.56 ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	—
Oxford Flowers-102	99.68 ± 0.02	99.74 ± 0.00	99.61 ± 0.02	99.63 ± 0.03	—
VTAB (19 tasks)	77.63 ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	—
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Image Captioning with Transformers?



Ed Questions

Cost Function

It's not very clear what kind of cost function is used to evaluate the performance, would be nice to elaborate that.

- There is a difference between cost function and the performance evaluation
- Cost Function is used for Training is $L_d = -\log(P(y|x)) + \lambda \sum_i^L (1 - \sum_t^C \alpha_{ti})^2$
- For Performance Evaluation, authors use 2-different measures:
 - BLEU: Bilingual Evaluation Understudy
 - Uses a modified form of precision to compare a translation against multiple reference translations
 - METEOR: Metric for Evaluation of Translation with Explicit Ordering
 - Based on the harmonic mean of unigram precision and recall, with recall weighted higher than precision

Cost Function

How would you handle/standardize the differences between dataset splits?

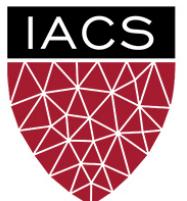
- Authors use the predefined splits of Flickr8k
- For Flickr30k and MS COCO, authors use publicly available splits used in previous work i.e. Karpathy & Li, 2014
- Other splits can be generated and are unlikely to impact the performance scores

Cost Function

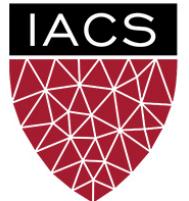
How to input and train with text data and image data at the same time? What are the preprocessing needed for constructing data pipeline?

```
train_captions = []
img_name_vector = []

for image_path in train_image_paths:
    caption_list = image_path_to_caption[image_path]
    train_captions.extend(caption_list)
    img_name_vector.extend([image_path] * len(caption_list))
```



Questions Please!



Thanks!

