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

Abstract

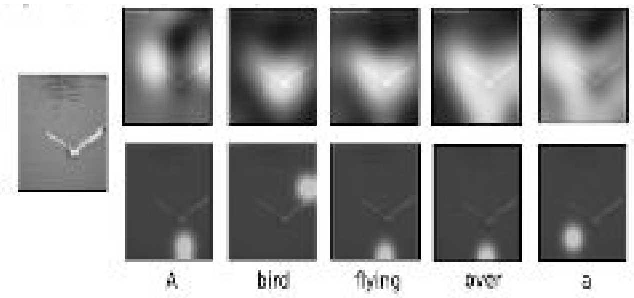
We introduce an attention-based model that automatically learns to describe the content of images. We describe how we can train this model in either a deterministic manner or stochastically together with validation of the use of attention through the state-of-the art performance on three benchmark datasets.

1. Introduction

Automatically generating captions of an image is one of the primary goals of computer vision. It is a task that is not only concerned with capturing the objects in an image but moreover capturing and expressing their relationships in a natural language in a way that resembles the human ability to compress visual information in a descriptive manner.

Provided a lot of research in the field of image caption generation, together with the resulting large classification datasets, the task has highly improved through using a combination of CNNs to obtain vectorial representation of images and recurrent neural networks to decode those representations into natural language sentences.

To preserve more descriptive captions this requires using more low-level representation and also requires a powerful mechanism to steer the model to information important to the task at hand.

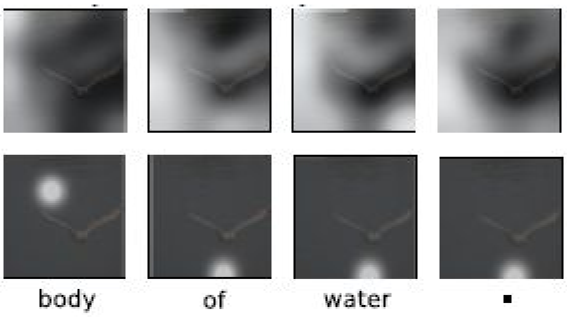


In the following points we will discuss:

- A two attention-based image caption generators under a common framework: A "soft" deterministic attention mechanism and a "hard" stochastic attention mechanism.
- How we can gain insight and interpret the results of this framework by visualizing "where" and "what" the attention focused on.
- Validation of the usefulness of attention in caption generation with state of the art performance on three benchmark datasets. Flickr8k, Flickr30k and MS COCO.

2. Related Work

In this section we provide relevant background on previous work on image caption generation and attention based on recurrent neural networks and inspired by the successful use of sequence to sequence training with neural networks for machine translation. One major reason image caption generation is well suited to the encoder-decoder framework of machine translation is because it is analogous to "translating" an image to a sentence.



"Attention over time. As the model generates each word, its attention changes to reflect the relevant parts of the image. "Soft" (top row) vs "hard" (bottom row) attention."

Attempts:

1. Kiros et al. (2014a): proposed a multimodal log-bilinear model that was biased by features from the image.
2. Kiros et al. (2014b): explicitly allow a natural way of doing both ranking and generation.
3. Mao et al.(2014): took a similar approach to generation but replaced a feed-forward neural language model with a recurrent one.
4. Both Vinyals et al. and Donahue et al.: use LSTM RNNs for their models. Unlike Kiros et al. (2014a) and Mao et al. *whose models see the image at each time step of the output word sequence*

All of the previous works represent images as a single feature vector from the top layer of a pre-trained convolutional network.

5. Karpathy & Li (2014): proposed to learn a joint embedding space for ranking and generation whose model learns to score sentence and image similarity as a function of R-CNN object detections with outputs of a bidirectional RNN.
6. Fang et al. (2014): proposed a three-step pipeline for generation by incorporating object detections.

Their model first learns detectors for several visual concepts based on a multi-instance learning framework. Unlike these models, our proposed attention framework learns latent alignments from scratch.

This allows our model to go beyond "objectness" and learn to attend to abstract concepts.

Prior to the use of neural networks:

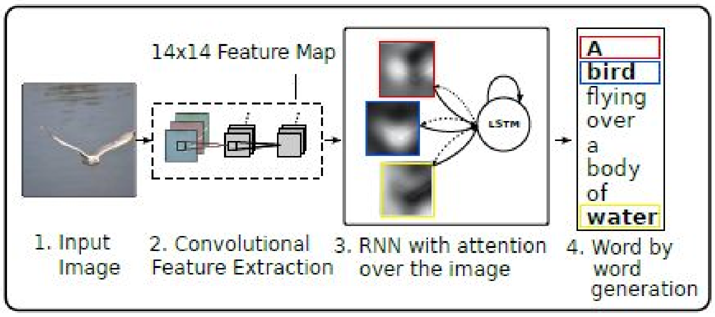
- *First approach*: generating caption templates which were filled in based on the results of object detections and attribute discovery.
- *Second approach:* based on first retrieving similar captioned images from a large database modified then to fit the query.

These approaches typically involved an intermediate "generalization" step to remove the specifics of a caption that are only relevant to the retrieved image.

Our work directly extends the work of Bahdanau et al. (2014); Mnih et al. (2014); Ba et al. (2014).

3. Image Caption Generation with Attention Mechanism

3.1. Model Details


14x14 Feature Map
LSTM
A
bird
flying
over
a
body
of
water
1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

3.1.1. ENCODER: CONVOLUTIONAL FEATURES

$$y = \{\mathbf{y}_1, \ldots, \mathbf{y}_C\}, \ \mathbf{y}_i \in \mathbb{R}^K$$

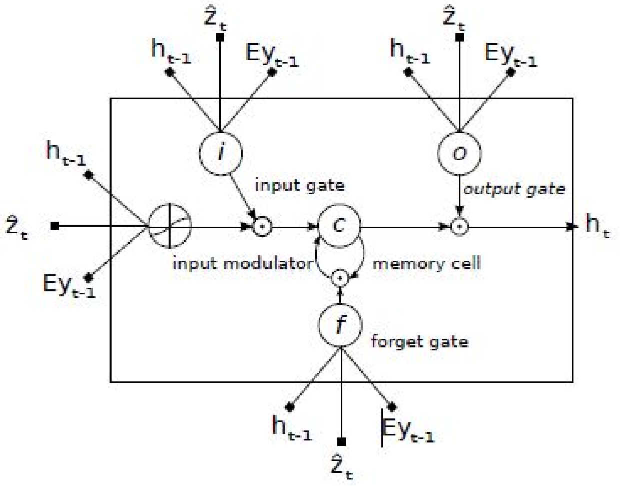
Our model takes a single raw image and generates a caption y encoded as a sequence of 1-of-K encoded words.

(K: size of the vocabulary, C: length of the caption.)

We extract features from a lower convolutional layer where the extractor produces L D-dimensional vectors corresponding to a part of the image. This allows the decoder to selectively focus on certain parts of an image by selecting a subset of all the feature vectors.

$$a = \{\mathbf{a}_1, \ldots, \mathbf{a}_L\}, \ \mathbf{a}_i \in \mathbb{R}^D$$

3.1.2. DECODER: LONG SHORT-TERM MEMORY NETWORK



LSTM cell:

- It mitigates the vanishing gradient problem, which is where the neural network stops learning because the updates to the various weights within a given neural network become smaller and smaller. It does this by using a series of ‘gates’.
- Each cell learns how to weigh its input components *(input gate)***,** while learning how to modulate that contribution to the memory *(input modulator)***.** It also learns weights which erase the memory cell *(forget gate)*, and weights which control how this memory should be emitted *(output gate)***.**

We use a LSTM that produces a caption by generating one word at every time step conditioned on a context vector $\hat{\mathbf{z}}_t$ provided by the attention mechanism, the previous hidden state and the previously generated word.

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t).$$

- $\mathbf{i}_t, \mathbf{f}_t, \mathbf{c}_t, \mathbf{o}_t, \mathbf{h}_t$ are the input, forget, memory, output and hidden state of the LSTM.
- z is the context vector
- E is an embedding matrix, m and n denote the embedding and LSTM dimensionality.
- σ and $\odot$ represent the logistic sigmoid activation and element-wise multiplication.

Steps to compute the context vector :

- Use the annotation vectors $\mathbf{a}_i, i = 1, \ldots, L$ corresponding to the features extracted at different image locations.
- For each location i, generate a positive weight α_i interpreted either as the probability that location i is the right place to focus for producing the next word (the “hard” but stochastic attention mechanism)
- The weight i of each annotation vector ai is computed by an attention model fatt for which we use a multilayer perceptron conditioned on the previous hiddenstate h_{t-1}.

$$e_{ti} = f_{\text{att}}(\mathbf{a}_i, \mathbf{h}_{t-1})$$
$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})}.$$

- Once the weights (which sum to one) are computed, the context vector z is computed by

$$\hat{\mathbf{z}}_t = \phi\left(\{\mathbf{a}_i\}, \{\alpha_i\}\right)$$

(where ϕ is a function that returns a single vector given the set of annotation vectors and their corresponding weights)

- The initial memory state and hidden state of the LSTM are predicted by an average of the annotation vectors fed through two separate MLPs (multiple layer perceptron) (init,c and init,h):

$$\mathbf{c}_0 = f_{\text{init,c}}\left(\frac{1}{L}\sum_{i}^{L} \mathbf{a}_i\right)$$

$$\mathbf{h}_0 = f_{\text{init,h}}\left(\frac{1}{L}\sum_{i}^{L} \mathbf{a}_i\right)$$

We use a deep output layer to compute the output word probability given the LSTM state, the context vector and the previous word.

$$p(\mathbf{y}_t|\mathbf{a}, \mathbf{y}_1^{t-1}) \propto \exp(\mathbf{L}_o(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbf{h}_t + \mathbf{L}_z\hat{\mathbf{z}}_t))$$

$\mathbf{L}_o \in \mathbb{R}^{K\times m}$, $\mathbf{L}_h \in \mathbb{R}^{m\times n}$, $\mathbf{L}_z \in \mathbb{R}^{m\times D}$, and $\mathbf{E}$ are learned parameters initialized randomly.

4. Learning Stochastic “Hard” vs Deterministic “Soft” Attention

4.1. Stochastic “Hard” Attention

$$p(s_{t,i} = 1 \mid s_{j<t}, \mathbf{a}) = \alpha_{t,i}$$
$$\hat{\mathbf{z}}_t = \sum_i s_{t,i}\mathbf{a}_i.$$

s_t **:** as where the model decides to focus attention when generating the t^{th} word.

$s_{t,i}$: is an indicator one-hot variable which is set to 1 if the i-th location (out of L) is the one used to extract visual features.

$\{\alpha_i\}$: Parameter of multinoulli distribution.

$\hat{\mathbf{z}}_t$: A random variable.

$$\frac{\partial L_s}{\partial W} = \sum_s p(s \mid \mathbf{a}) \left[\frac{\partial \log p(\mathbf{y} \mid s, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid s, \mathbf{a}) \frac{\partial \log p(s \mid \mathbf{a})}{\partial W} \right].$$

L_s*:* a variational lower bound on the marginal log-likelihood log p(y | a) of observing the sequence of words y given image features a.

- ➔ By sampling the location s_t from a multinouilli distribution.

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

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \lambda_r (\log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) - b) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right]$$

- ➔ A moving average baseline is used to reduce the variance in the Monte Carlo estimator of the gradient, it is estimated as an accumulated sum of the previous log likelihoods with exponential decay:

$$b_k = 0.9 \times b_{k-1} + 0.1 \times \log p(\mathbf{y} \mid \tilde{s}_k, \mathbf{a})$$

- ➔ To further reduce the estimator variance, an entropy term on the multinouilli distribution H[s] is added.

(Both techniques improve the robustness of the stochastic attention learning algorithm.)

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \lambda_r (\log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) - b) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right]$$

λ_r and λ_e:Two hyper-parameters set by cross validation.

This formulation is equivalent to the REINFORCE learning rule.

Stochastic "hard" attention means learning to maximize the context vector $\hat{z}$ from a combination of a one-hot encoded variable $s_{t,i}$ and the extracted features .

- **Hard**: $s_{t,i}$ a hard choice is made at each feature a_i.
- **Stochastic**: s_t is chosen from a multinoulli distribution.

4.2. Deterministic “Soft” Attention

- ➔ We take the expectation of the context vector directly $\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t] = \sum_{i=1}^{L} \alpha_{t,i} \mathbf{a}_i$
- ➔ Formulate a deterministic attention model by computing a soft attention weighted annotation vector $\phi(\{\mathbf{a}_i\}, \{\alpha_i\}) = \sum_i^L \alpha_i \mathbf{a}_i$

  This corresponds to feeding in a α soft weighted context into the system.
- ➔ We define the normalized weighted geometric mean for the softmax k^{th}word prediction can be approximated well by using the expected context vector:

$$NWGM[p(y_t = k \mid \mathbf{a})] \approx \mathbb{E}[p(y_t = k \mid \mathbf{a})]$$

The deterministic attention model is an approximation to the marginal likelihood over the attention locations.

Deterministic soft-attention means learning by maximizing the expectation of the context vector.

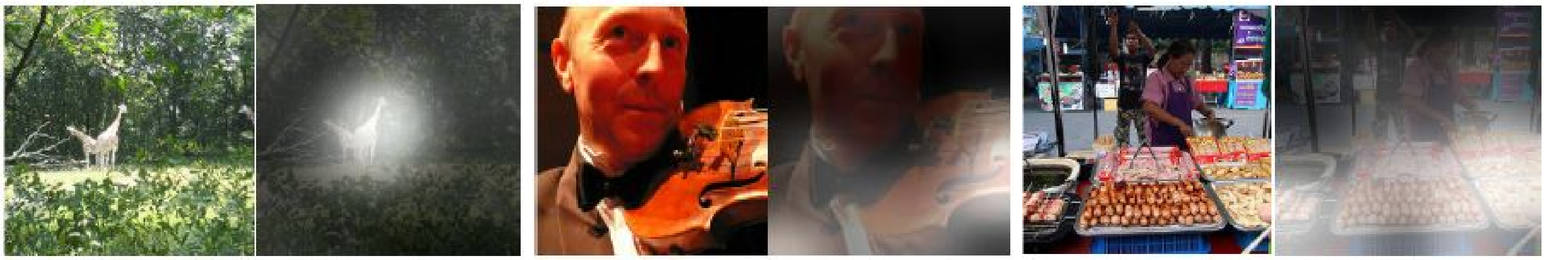
- **Deterministic**: $s_{t,i}$ is not picked from a distribution.
- **Soft**: the individual choices are not optimized, but the whole distribution.

4.2.1. Doubly Stochastic Attention

By construction, $\sum_i \alpha_{ti} = 1$ they are the output of a softmax.

The model is trained end-to-end by minimizing the following penalized negative log-likelihood:

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



A large white bird standing in a forest.

A man wearing a hat and a hat on a skateboard.

A woman is sitting at a table with a large pizza.

“Examples of mistakes where we can use attention to gain intuition into what the model saw.”

4.3. Training Procedure

Both variants of our attention model were trained with stochastic gradient descent using adaptive learning rate algorithms.
(Flickr8k dataset: RMSProp - Flickr30k/MS COCO dataset: Adam algorithm)

- ➔ To create the annotations AI used by our decoder, we used the Oxford VGGnet.
- ➔ In principle however, any encoding function could be used. In addition, with enough data, we could also train the encoder from scratch (or fine-tune) with the rest of the model.
- ➔ In our experiments we use the 14x14x512 feature map of the fourth convolutional layer before max pooling. This means our decoder operates on the flattened 196x512 (i.e L D) encoding.
- ➔ As our implementation requires time proportional to the length of the longest sentence per update, we found training on a random group of captions to be computationally wasteful. To mitigate this problem:
    - ◆ In preprocessing, we build a dictionary mapping the length of a sentence to the corresponding subset of captions. Then, during training we randomly sample a length and retrieve a mini-batch of size 64 of that length.
- ➔ With soft attention, Whetlab1 is used in our Flickr8k experiments.
- ➔ Some of the intuitions we gained from hyperparameter regions it explored were especially important in the Flickr30k and COCO experiments.

5. Experiments

5.1. Data

- ➔ Report results on the popular Flickr8k (8,000 images) and Flickr30k dataset (30,000 images) and Microsoft COCO dataset (82,783 images).
- ➔ The Flickr8k/Flickr30k dataset both come with 5 reference sentences per image, but for the MS COCO dataset, some of the images have references of more than 5 images which for consistency were truncated to 5. There was also some basic tokenization applied to the MS COCO dataset to be consistent with the tokenization in the Flickr datasets.
- ➔ Using a fixed vocabulary size of 10,000.
- ➔ Results are reported with the frequently used BLEU metric which is the standard in the caption generation literature. We report BLEU from 1 to 4.
- ➔ In addition we report another common metric METEOR.

5.2. Evaluation Procedures

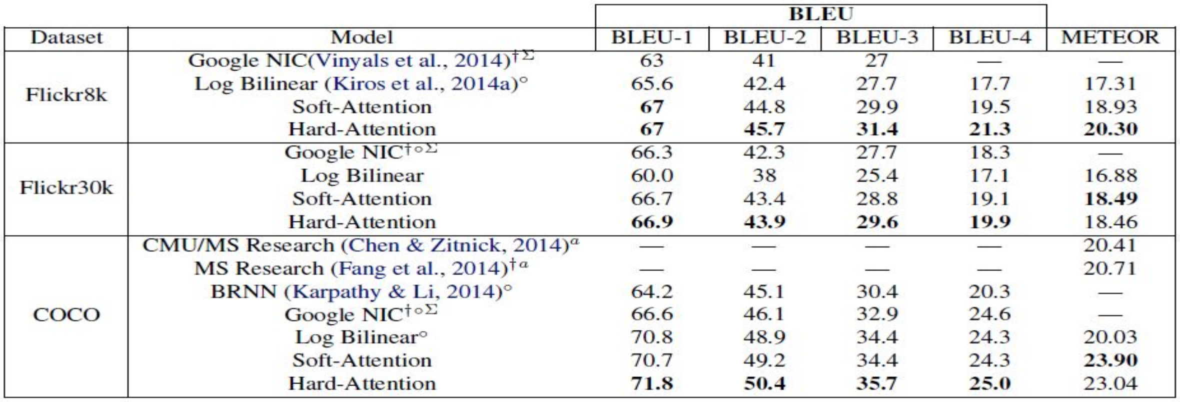
Challenges that exist for comparison:

1. The first is a difference in choice of convolutional feature extractor.

(use the comparable GoogLeNet/Oxford VGG features, but for METEOR comparison we note some results that use AlexNet.)

2. The second challenge is a single model versus ensemble comparison.
3. Finally, there is a challenge due to differences between dataset splits.

But differences in splits do not make a substantial difference in overall performance in this evaluation.



Dataset	Model	BLEU				
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
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

5.3. Quantitative Analysis

By experimenting and validating the quantitative effectiveness of attention:

1. We obtain state of the art performance on the Flickr8k, Flickr30k and MS COCO.
2. We note that we are able to improve the state of the art performance METEOR on MS COCO that we speculate is connected to some of the regularization techniques.
3. We also note that we are able to obtain this performance using a single model without an ensemble.

5.4. Qualitative Analysis: Learning to attend

- An extra layer of interpretability is added to the output of the model. Other systems that have done this rely on object detection systems to produce candidate alignment targets.

(This approach is much more flexible, since the model can attend to “non object” salient regions.)

- The model learns alignments that correspond very strongly with human intuition. Especially in the examples of mistakes, we see that it is possible to exploit such visualizations to get an intuition as to why mistakes were made.

6. Conclusion

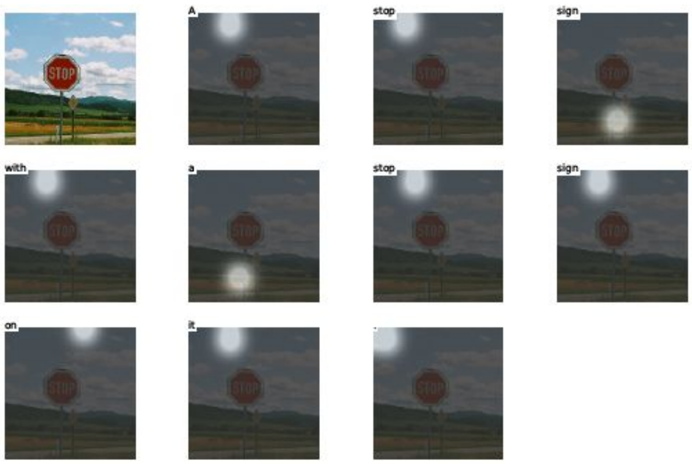
➢ We propose an attention based approach that gives state of the art performance on three benchmark datasets using the BLEU and METEOR metric.

➢ We also show how the learned attention can be exploited to give interpretability into the models generation process.

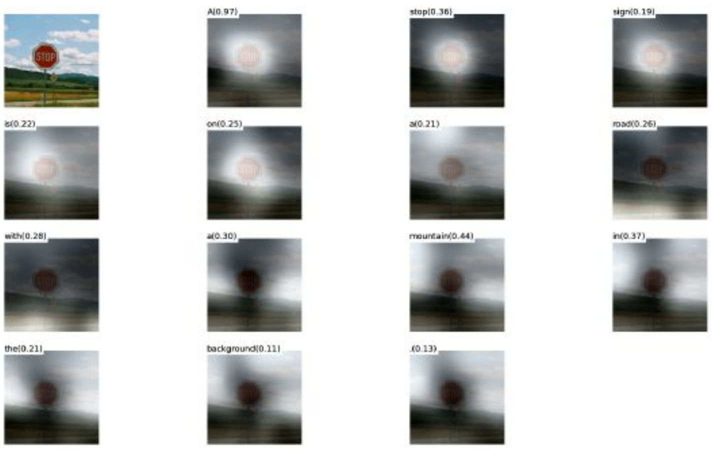
➢ And demonstrate that the learned alignments correspond very well to human intuition.

Appendix

Visualizations from our “hard” (a) and “soft” (b) attention model. White indicates the regions where the model roughly attends to.



(a) A stop sign with a stop sign on it.



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