Recurrent Query Patch Generation for Visual Question Answering

Christopher Menart, Adam Stiff, Ashutosh Pandey, Debanjan Nandi
The Ohio State University
281 W. Lane Ave
menart.1@osu.edu, stiff.4@osu.edu, pandey.99@osu.edu, nandi.20@osu.edu

Abstract

Convolutional Neural Networks have made great strides in object detection and classification, but often at great computational cost. Meanwhile, attention mechanisms in image captioning and question-answering tasks, as well as LSTM networks in language, point to the value of sequential processing for the understanding of complex signals. We propose a network architecture inspired by (but ultimately quite divergent from) biology for the processing of large images in sophisticated tasks.

1. Introduction

Computer Vision has led the way for much of deep learning research with large and successful architectures such as VGG [8] and Microsoft ResNet [4]. Traditionally, these networks comprise tens of millions of parameters and are trained on the extremely large ImageNet dataset [2]. ResNet, among the latest of these networks, achieved an error rate of 4.49% on this challenging task.

However, these architectures are often very difficult to train on anything but very large datasets. Furthermore, they may not be entirely suitable for tasks more granular or difficult than simple identification. One of the anticipated challenges to the advancement of semantic segmentation is the loss of information resolution which is traded off by these traditional architectures [3]. Semantic segmentation is a task which requires labeling every pixel in an image, and thus requires detailed scene understanding. Many other tasks of interest also require detailed scene understanding.

This loss of information is implemented with pooling layers, which serve two purposes: they more quickly entangle the computation over each part of the image with information from the whole image, and they allow a convolutional network to compute a high-dimensional feature vector within a tractable amount of memory. Accomplishing these important objectives without a similar loss of information would be a major step forward with many vision tasks.

Besides computational concerns, the task of scene understanding also motivates this work. While the identification tasks that motivated the design of convolutional networks require shift-invariant features, many object relations are inherently spatial, and it is these relations that present the biggest challenge for scene understanding. A sequential visual processing model offers the same benefits for identification on a local scale, while providing a second channel of information for identifying relations among recognized objects in the scene, in the form of the distances and directions between processed regions of the image.

While these dual concerns of computational cost and relation recognition motivated the general idea of a sequential visual processing model, the Visual Saccade architecture that we propose actually fails to address either problem. This is because of challenges inherent in the non-differentiability of the operation of selecting a subregion on which to perform further operations. Our approach requires a finite partition of the input image, and performing a comparison to each subpart at every time step; this is computationally quite expensive. Furthermore, the network does not have direct access to information about the spatial relationships between selected subparts, eliminating the possible benefit of that information. Nonetheless, we introduce what we believe to be a fairly novel application of image generation, for the purpose of querying an input image for information relevant to a task.

The Visual Saccade architecture analyzes images in a recurrent fashion by repeatedly 'viewing' cropped subsections of a larger image. The hope of our design is that the hidden recurrent state entangles the computation over all viewed portions of the image without direct reliance on convolution or pooling beyond processing of each subsection individually. This potentially allows computation to occur without loss of resolution. Computation only occurs on one subsection of the image at a time thereby further reducing the large spatial complexity (memory burden) of traditional convolution.

The choice of how to represent questions is a major experimental parameter in this work. We explored several such representations of questions varying from complex representations such as using final hidden state of an LSTM network with non-pretrained word-embeddings to simpler ones as just the average of pre-trained GloVe vectors of the words in the sentence. Despite having slightly lower accuracy than our best result, we report our main result using the latter representation, due to significant benefits in terms of computational cost. We test our method on the popular VQA Challenge [1], which is built on Microsoft's COCO dataset [5].

2. Related Work

Much of the interest in attention mechanisms has been driven by [9]. This work is based on a "soft" attention mechanism, which produces a continuous-valued map used to create a weighted sum of information from across an entire image. A hard attention mechanism is also implemented, though it is less performant and given less attention.

Soft attention mechanisms are fully differentiable, which may be a contributing factor in their success; however, they have a downside in them being computationally costly. Soft attention networks must process the entire input over which they wish to compute attention, which saves no memory over not having an attention mechanism at all. Hard attention mechanisms have the potential to reduce the memory burden of processing large inputs in certain situations, such as tasks where a large image must be viewed repeatedly, but only some of the image is relevant at any given time.

The most fundamental difference between the hard attention mechanism of the captioning LSTM of [9] and the method we propose is the lack of immediate feedback at each time step. The captioning network utilizes recurrence in order to produce a sequence of output, while our method utilizes recurrence to accumulate information for later use in answering the question.

Training for deep learning models without immediate feedback was explored in depth by [6], who apply reinforcement learning to the problem of playing games on the Atari 2600. There is a subtle difference between this approach and the methods we propose here; the model from Mnih et al seeks only to learn the best decision at each time step, and the time-delayed rewards given are a function only of those decisions. The network is not recurrent at all. But in the Visual Saccades model, the hard decisions made by the model select input information, which is itself fed to part of the model, which produces answers, on which loss (equivalently, reward) is computed. Both portions of this model must be trained in tandem.

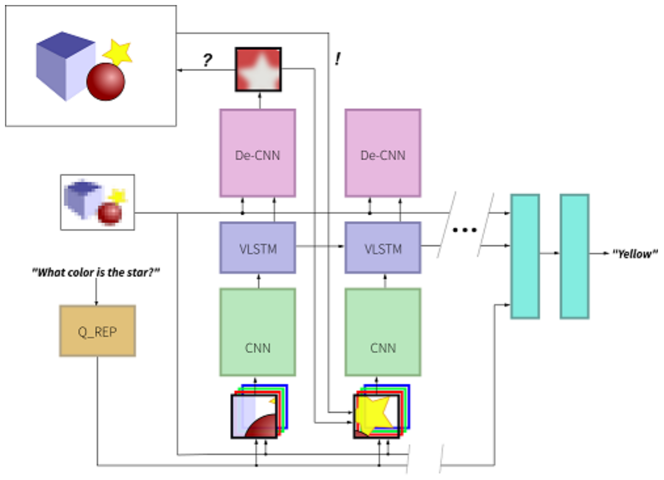

Figure 1. Overview diagram of the VLSTM architecture, depicting the inputs and outputs involved in the transition between two successive time steps.

3. Visual Saccade VLSTM

Here we describe the general structure of our model; see Figure 1 for an overview diagram. The VLSTM system is built around an LSTM core architecture with input coming from a relatively small (compared to state-of-the-art image recognition architecures) convolutional network. The input to that convolutional network, in turn, comes from 1) a low-resolution version of the input image, 2) a single full-resolution subregion (or "patch") of the input image, 3) the previously-generated query patch, and 4) the representation of the question; all concatenated into a 10-channel representation. Unlike other LSTM architectures for similar tasks, the number of time steps is not tied to the length of any input; rather, at each of an arbitrary number of timesteps, a different region of the input image is chosen and focused on. Theoretically, the number of timesteps could even be learned, though for these experiments we choose an arbitrary episode length of 10.

The output of this core LSTM at every time step is interpreted as high-level image features by a deconvolutional network that mirrors the structure of the input convolutional network. The deconvolutional network generates a "query patch," which is then compared with each of the non-overlapping full-resolution patches of the input image. The closest-matching patch (in terms of pixel-channel-wise absolute difference) is chosen as input to the CNN on the next time step, and during training, the $\mathcal{L}_1$ loss between the generated and chosen patch is backpropagated into the deconvolutional network.

The question itself is represented as either the final hidden state of an LSTM, or as the average of GloVe vectors of the question's constituent words. This fixed representation is passed into the CNN at every time step as a single channel of the input, as mentioned above.

Question-answering output is produced only at the final timestep, when the VLSTM hidden state is passed into fully-connected layers which produce softmax output and cross-entropy loss. Additional inputs to the fully-connected layers are the low-resolution image and the question representation.

We describe the component networks in more detail in the subsections that follow.

3.1. Patch Input Convolutional Network (CNN)

As alluded to above, the input to the CNN at every time step consists of three $32 \times 32 \times 3$ images and one 1024-dimensional question representation (described below), reshaped into $32 \times 32 \times 1$, all concatenated along the channel axis. The three images are the generated 3-channel patch from the previous timestep, the one 3-channel patch from the input image that most closely matches the generated patch in terms of $\mathcal{L}_1$ distance (the "selection"), and the 3-channel low-resolution version of the full input image, obtained via bicubic interpolation. The generated patch is initialized to zeros, and the selection is initialized to the central-most patch of the image. We pre-process images by resizing them to $512 \times 512 \times 3$ via bicubic interpolation. The patches available for selection are obtained by splitting the resized images into disjoint $32 \times 32 \times 3$ patches, i.e. with a stride of 32. A stride of 16 was considered, which would have resulted in overlapping patches. However, this was rejected due to the risk of poor interaction with the novelty loss we employed to increase variety in patch selection (see below).

The input is fed into a four-layer convolutional network without pooling layers. We use a stride of 2 at every layer to halve the resolution while doubling the number of channels, finally reshaping the $2 \times 2 \times 512$ representation into a flat 2048-dimensional vector to feed into the VLSTM. We use leaky ReLU activations, dropout of 0.2, and batch normalization at every layer.

3.2. Patch Generation and Selection (De-CNN)

Patches are generated by deconvolving the output of the VLSTM concatenated with the low-resolution image. The architecture of the deconvolutional network (De-CNN) mirrors that of the CNN, using four layers and doubling resolution at each layer. Again, we use dropout of 0.2, batch normalization, and leaky ReLU activations. The final layer is a simple affine layer.

The output of the deconvolutional network is taken as a patch, to be matched to one of the available patches in the image. The closest patch in terms of $\mathcal{L}_1$ distance is taken as the matching patch, and the $\mathcal{L}_1$ loss is backpropagated into the De-CNN. The De-CNN also receives error gradients via those channels of the input to the CNN which contain the generated patch. We term the $\mathcal{L}_1$ difference between a generated patch and the actual patch it most closely resembles the *patch loss*. We denote this loss for the patch generated at time T as

$$L_{\text{patch}_T} = |x_T - \arg\min_j (x - y_j)\,| \tag{1}$$

where x_T is the generated patch and y_j denotes a pre-segmented image patch.

Early experiments demonstrated that the network was quite content to sit in one place on the image and generate minor variations on a theme. To encourage the network to look at different parts of the image, we introduced an additional loss term at the output of the De-CNN, which we call the *diff loss*. Admittedly naively, we take the diff loss as the inverse of the sum of the $\mathcal{L}_1$ distances between a generated patch and all previous generated patches. That is,

$$L_{\text{diff}_T} = \frac{1}{\sum_{t=0}^{T} |x_T - x_t|} \tag{2}$$

where x_t is a patch generated at time step t.

3.3. Question Representation (Q_Rep)

Multiple representations for question sentences are included in our experiments.

The first representation, which we call 'QLSTM', is computed with a 2-layer LSTM network, initialized with uniform random values and trained end-to-end in the model with dropout at a 50% rate. This network is tied to the length of the question sentence, and operates before the main VLSTM begins processing the image.

The second representation is to use the mean of all GloVe vectors [7] that make up the sentence. This representation is pre-trained and may be computed ahead of time.

3.4. VLSTM and Fully-connected Layers

The Visual LSTM is a 2-layer LSTM with 1024 units per layer. Within the LSTM we use output dropout, state dropout, and variational recurrence, that is, the same dropout mask is repeated across timesteps within the same episode. For all of these dropouts we use a rate of 0.5. We fix the number of timesteps at 10, arbitrarily. The final hidden state of the LSTM is concatenated with the Q_Rep, and passed to the first layer of the fully-connected network. The fully connected network consists of two hidden layers of 1324 nodes with ReLU activations followed by a 1000-node softmax layer with cross-entropy loss. The total loss train-

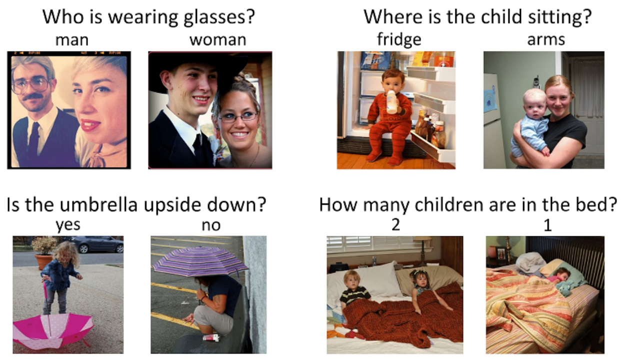
Who is wearing glasses?
man
woman
Where is the child sitting?
fridge
arms
Is the umbrella upside down?
yes
no
How many children are in the bed?
2
1

Figure 2. Image-Question pairs from the VQA dataset.

ing the network then is

$$L_{\text{total}} = \frac{1}{T}\left(\sum_{i=1}^{T}(\lambda_1 L_{patch_i} + \lambda_2 L_{diff_i})\right) - \sum_{n=1}^{N} z_n \log(p_n) \tag{3}$$

where T is the total number of time steps (fixed to 10), λ_1 and λ_2 are fixed to 1, N is the number of answer classes (i.e. 1000), and p_n is the output of the softmax layer (predicted probability) for class n, and z_n is 1 if class n is the true label, and 0 otherwise.

4. Experiments

We evaluate this architecture on the Visual Question Answering task [1], because it seems to be a task which requires deep scene understanding, and could benefit from a technique which successfully improves the integration of both small details and global context. Questions in the VQA Challenge dataset frequently reference multiple objects of either similar or different types.

Specifically, we evaluate on the 1000-class task, where the model can select between the 1000 most common answers to a question posed about any image. Answers are considered correct if they match with the responses given by three out of ten human annotators.

Results we show on this task make use of several different representations for questions, and/or several additional components for the loss function used in training. The final experiment takes advantage of all features discussed above.

We compare this to several 'baseline' experiments which take advantage only of question representations; these amount to 'lucky guesses', chosen from answers that the network determines are appropriate for a given question.

5. Discussion

In these experiments, results obtained from question features only are in many cases nearly as accurate or just as accurate as results obtained by combining them with visual features via the VLSTM architecture. These results suggest that the accuracy being achieved is due mainly or solely to question understanding, rather than visual understanding.

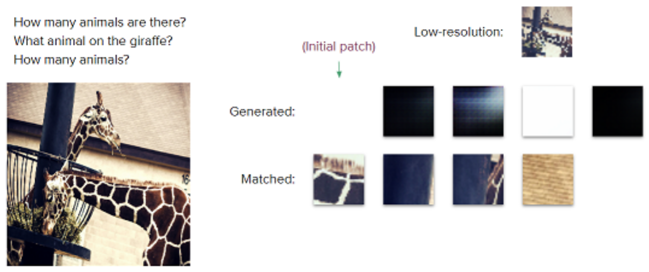

Figure 3. Example of patch generation.

In fact, VLSTM output was totally degenerate until the question representation was provided directly to the final fully-connected layers.

We attribute this in part to undesirable patch generation in the trained system. Figure 3 illustrates a typical operation of the patch generation system. While the GAN output does vary over time, due in part to the novelty loss term [may be discussed here or in architecture], it does not closely resemble any portion of the actual image. The closest-match patches selected by this system appear to be largely random in nature, and unlikely to contain sufficient information for answering any of the example questions.

The VLSTM hidden state, even at larger sizes, clearly failed to encode sufficient representational information for the deconvolution operation to reconstruct any portion of the image.

Multiple challenges impacted the project. The exploration results also suggest that the amount of visual information presented in patches was too small given a limited number of timesteps. At ten image patches of size 32×32, the network could view up to 10,240 pixels in the course of a single episode. Images, which were all resized to a standard 256×256, each have 65,536 pixels. It may have been more effective to use larger patch sizes, such as 64×64, or to operate for a much greater number of timesteps.

However, even training a network of ten saccades was computationally difficult. Propagation of gradients through ten timesteps of operation was slow when moderately-sized convolution and deconvolutional networks were both involved. While spreading out the processing over time-steps does reduce the difficulty of the forward pass, the backward pass of backpropagated gradient descent does remains unchanged. In that regard, the repeated application of a mere four convolutional layers is equivalent in training difficulty to a 40-layer network, likely with far less expressive capacity.

This is the main theoretical challenge to the effectiveness of Saccade VLSTMs.

Question Representation	Accuracy
QLSTM w/non-pretrained word embeddings	41.2%
Average of pretrained GloVe vectors	38.67%

Table 1. Baseline Experiments

Model	Accuracy
QLSTM + VLSTM	42.0%
QLSTM (not connected to FC Layer) + VLSTM	24.47%
Pretrained GloVe vectors + VLSTM	38.81%
Pretrained GloVe + Patch Loss + Diff Loss + VLSTM	39.86%
Pretrained GloVe + Patch Loss + VLSTM	**40.35%**
VQA Challenge baseline [1]	62.70%

Table 2. Primary Experiments

6. Conclusion

In this paper, we propose a novel recurrent architecture for the analysis of large images in complex tasks. This method is motivated by the structure of biological vision and implemented using an LSTM framework.We demonstrate the results of experiments in several configurations on the task of visual question answering.

These results do fail to show a benefit from the method, due to the computational costs imposed in attempting to train recurrent architectures for a large number of timesteps, and also due to some degeneracy in the methods of foveal patch selection.

This does contradict one of the suggested benefits of this approach, namely the efficiency of sequential processing. It may be that our ability to emulate biological neural structures is limited by the nature of backpropagation as we understand it today, which requires preserving all information across time throughout the operation of a model for the purposes of optimization.

However, many alternative configurations have also been proposed, and in the future, such sequential techniques may bear fruit for vision tasks.

References

[1] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Batra, and D. Parikh. Vqa: Visual question answering. 2015. 2, 4, 5

[2] J. Deng, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical database. 2009. 1

[3] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation. 1

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. 2015. 1

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick. Microsoft coco: Common objects in context. 2014. 2

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, and I. Antonoglou. Playing atari with deep reinforcement learning. 2013. 2

[7] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. 3

[8] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 2015. 1

[9] K. Xu, J. Lei, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. 37:2048–2057, 2015. 2