Stacked RNNs for Encoder-Decoder Networks: Accurate Machine Understanding of Images

John Lambert
Department of Computer Science, Stanford University
johnwl@stanford.edu

Abstract

We address the image captioning task by combining a convolutional neural network (CNN) with various recurrent neural network architectures. We train the models on over 400,000 training examples (roughly 80,000 images, with 5 captions per image) from the Microsoft 2014 COCO challenge. We demonstrate that stacking a 2-Layer RNN provides better results on image captioning tasks than both a Vanilla LSTM and a Vanilla RNN.

1 Introduction: this section introduces your problem, and the overall plan for approaching your problem

Th
Not only do machine-generated captions offer scene understanding of arbitrary photographs, they also provide a way to reduce the workload of radiologists and clinicians as they diagnose patients via medical image analysis. Previous efforts in this space have been hindered by a lack of a large enough, curated data set, mapping images to free-text.

2 Background/Related Work: This section discusses relevant literature for your project

Encoder-decoder models have achieved extraordinary results in recent neural machine translation work [10] [6]. I use a ConvNet to encode information in one language (pixels), and an LSTM to decode the information into another language (human natural language).[10]

3 Approach:

3.1 Encoder-Decoder Model

I use the flattened (4096 x 1) output of a VGG-16 Net's FC-7 layer to initialize the hidden state of a recurrent neural network.

3.2 Vanilla RNN Model

For $t = 1, ..., n-1$

$$e^{(t)} = x^{(t)}L$$
$$h^{(t)} = sigmoid(h^{(t-1)}H + e^{(t)}I + b_1)$$
$$\hat{y}^{(t)} = softmax(h^{(t)}U + b_2)$$

where $h^0 = h_0$ is the initialization vector for the hidden layer. I use the CNN fc7 output for the image to initialize h_0. $x^{(t)}L$ is the product of L with the one-hot row-vector $x^{(t)}$ representing the index of the current word.

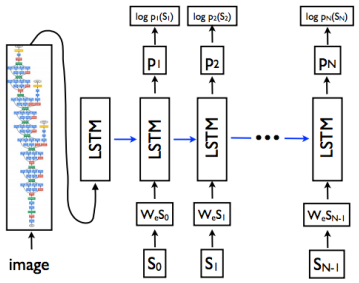

Figure 1: Encoder-Decoder Diagram, from [10]

The vectors y_t have size $|V|$ + 1, where V is the token vocabulary, and where the additional token is for a special <END> token. I use an average cross-entropy loss function on the vectors y_t, where the targets at times $t = 0, ..., T-1$ are the token indices for s_{t+1}, and the target at $t = T$ is the $<$ END $>$ token. I will ignore the vector y_{-1}. I plan to use a size of 512 for my tokens and hidden layers.

At test time, I will feed the visual information x_{-1} to the RNN. At every single time step, I will sample the most likely next token and feed it into the RNN in the next time step, repeating the process, until the special <END> token is sampled.

3.3 Vanilla LSTM Model

For the RNN, I use a vanilla RNN. The LSTM architecture is a memory cell which can maintain its state over time, with non-linear gating units which regulate the information flow into and out of the cell. [4] [5].

Similar to the vanilla RNN, at each timestep we receive an input $x_t \in R^D$ and the previous hidden state $h_{t-1} \in R^H$; the LSTM also maintains an H-dimensional *cell state*, so we also receive the previous cell state $c_{t-1} \in R^H$. The learnable parameters of the LSTM are an *input-to-hidden* matrix $W_x \in R^{4H \times D}$, a *hidden-to-hidden* matrix $W_h \in R^{4H \times H}$ and a *bias vector* $b \in R^{4H}$. At each timestep, we first compute an *activation vector* $a \in R^{4H}$ as $a = W_x x_t + W_h h_{t-1} + b$. We then divide this into four vectors $a_i, a_f, a_o, a_g \in R^H$ where a_i consists of the first H elements of a, a_f is the next H elements of a, etc. We then compute the *input gate* $g \in R^H$, *forget gate* $f \in R^H$, *output gate* $o \in R^H$ and *block input* $g \in R^H$ as

$$i = \sigma(a_i) \qquad f = \sigma(a_f) \qquad o = \sigma(a_o) \qquad g = \tanh(a_g)$$

where σ is the sigmoid function and $\tanh$ is the hyperbolic tangent, both applied elementwise.

Finally we compute the next cell state c_t and next hidden state h_t as

$$c_t = f \odot c_{t-1} + i \odot g \qquad\qquad h_t = o \odot \tanh(c_t)$$

where $\odot$ is the elementwise product of vectors.

$$p_{t+1} = Softmax(h_t)$$

3.4 Stacked RNN / LSTM Model

Pascanu et al. continue the work of Hihi and Bengio [3] in [7]. They define the Stacked-RNN as follows:

$$h_t{}^{(l)} = f_h{}^{(l)}(h_t{}^{(l-1)}, h_{t-1}{}^{(l)}) = \phi_h(W_l{}^T h_{t-1}{}^{(l)} + U_l{}^T h_t{}^{(l-1)})$$

where, $h_t{}^{(l)}$ is the hidden state of the l-th level at time t. When l = 1, the state is computed using x_t instead of $h_t{}^{(l-1)}$. The hidden states of all the levels are recursively computed from the bottom level l= 1.

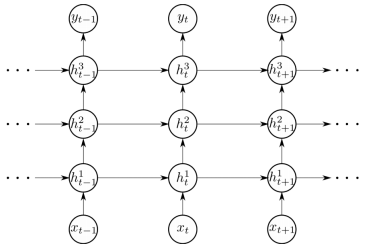

Figure 2: Stacked Recurrent Neural Network Architecture.

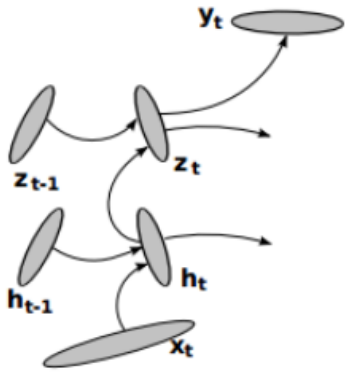

Figure 3: Another View of a Stacked RNN.

4 Experiment:

4.1 Data Set

The MS COCO challenge is a competition to achieve the highest quality sentence descriptions of images. Microsoft provides a publicly-available dataset of ground-truth (human-annotated) captions and images.

I use just over 400,000 training examples. Each ground-truth caption is 17 words in length, including a START and END token. I use a preliminary vocabulary of 1004 words (including one class to map all other, unknown words to an UNK token). I define one epoch as one pass over 400,000 training examples

4.2 Vanilla RNN Results

4.2.1 RNN Decoder on Full Data Set

12 epochs required 9.49 hours on a CPU. The model is not extremely robust, but it does learn very interesting, correct captions at times that were not realized in the ground-truth text.

I used a batch size of 25, a word embedding size of 256, a hidden size of 512 in my hidden state, 17 time-steps, 12 epochs over 400,000 training examples, dropout of 0.9, and a learning rate of 5e-3.

4.3 Vanilla LSTM Results

4.3.1 LSTM Decoder on Full Data Set

I trained for 4 epochs on a GeForce GTX TITAN GPU. I used a learning rate of 1e-3.

4.4 Stacked RNN / Stacked LSTM Results

4.4.1 2-Layer Stacked RNN, Trained for 4 Epochs

4.4.2 2-Layer Stacked RNN, Trained for 10 Epochs. 5 Times lower learning rate than Stacked-RNN trained for 4 epochs, with batch size 32 instead of 25.

I decay the learning rate to 96 percent every 20,000 epochs.

4.5 Perplexity, CIDEr, METEOR, BLEU

Qualitative evaluation is not sufficient. I use the coco-captions code produced by Microsoft [2] to compute a BLEU,METEOR, nad CIDEr score for generated

Figure 4: GT: many people are walking or on bikes near the trains
RNN: two red and white trucks sitting on display in a UNK

Figure 5: GT: a UNK out herd of horses UNK on a snow covered field
RNN: two horses are standing close to a body of water

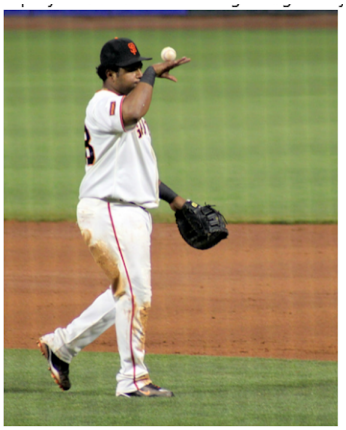

Figure 6: GT: a pitcher UNK a baseball on the back of his hand
RNN: two children play ball with a UNK getting ready to hit a ball

Table 1: Quantitative evaluation of full image predictions on 30 train images.

2***Model**	B-1	B-2	B-3	B-4	METEOR	CIDEr
Vanilla LSTM	TBD	TBD	TBD	TBD	TBD	TBD
Vanilla RNN	TBD	TBD	TBD	TBD	TBD	TBD
Stacked RNN,4 Epochs	0.250	0.113	0.051	0.000	TBD	TBD
Stacked RNN,10 Epochs	0.285	0.123	0.047	0.000	0.081	0.086
Karpathy et al.	62.5	45	32.1	23.0	19.5	66.0

captions. Papineni et al. provide a way that correlates closely to human judgment by analyzing the co-occurrences of n-grams between the candidate and reference translation sentences. Perplexity is not a sufficient metric [2] [1] [9].

$$BLEU_N(C,S) = b(C,S,)e^{\sum w_n * log(C * P_n(C,S))}$$

Note that Karpathy's numbers may or may not be scaled by 100, in comparison to mine. B-n is BLEU score that uses up to n-grams. High is good in all columns. I test on 30 examples from training set.

I also use the perplexity of the model to evaluate the performance of the LSTM for given generated text. The perplexity is the geometric mean of the inverse probability for each predicted word [**DBLP:conf/cvpr/VinyalsTBE15**].

5 Conclusion:

Image captioning is a very feasible task. The LSTM is slightly more effective than RNN, as I presumed, as its more complex architecture can capture more nuances than the Vanilla RNN. The 2-Layer Stacked RNN was most effective. Experimentation demonstrated that the choice of learning rate was critical to the success of our models.

Figure 7: GT: a hand holding a banana with a table
in background
LSTM: a banana and gray and white polar UNK holding a UNK of dishes and mouse rest beside a sit at outdoor kitchen corner

Figure 8: GT:a bunch of people are standing around
some buildings
LSTM: a UNK of a couple of vehicle bed many chairs and people are both hanging down ice leaning outside palm trees and a black of small boys on the water

Figure 9: GT: a close up of sliced pizza on a plate
LSTM: pizza and other at the end of the animal lays down end of elephants stall outfit over a huge vase of water pitcher throwing ball in parking lot where kites

Figure 10: GT: a gray and white cat perched atop a microwave oven
LSTM: a cat sit down UNK and covered lawn chairs facing water of small train

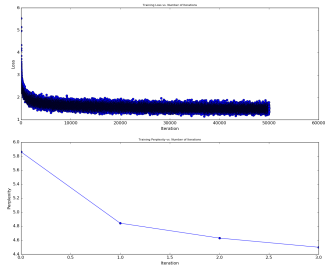

Figure 11: Loss and Perplexity of LSTM Trained for 4 Epochs.

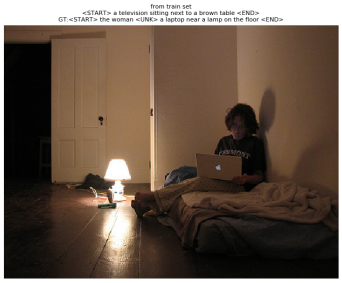

Figure 12: GT: the woman UNK a laptop near a lamp on the floor
2-Layer RNN: a television sitting next to a brown table

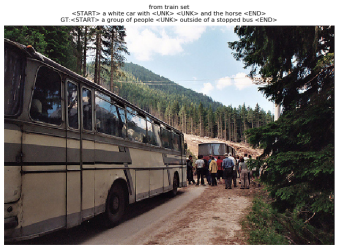

Figure 13: GT: a group of people UNK outside of a stopped bus
2-Layer RNN: a white car with UNK UNK and the horse

Figure 14: GT: a girl leaning on a table with a cake
2-Layer RNN: a home plate in a kitchen setting next to a pile of candles

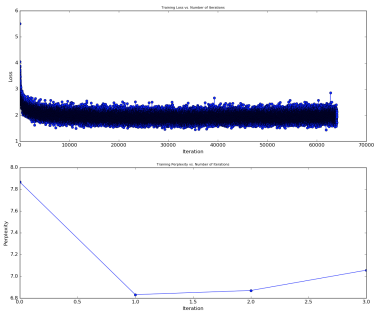

Figure 15: Training Set Loss and Perplexity of 2-Layer Stacked RNN over 4 epochs.

Figure 16

Figure 17

Figure 18

Figure 19

Figure 20

Figure 21

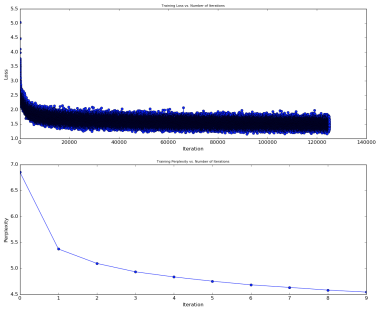

Figure 22: Training Set Loss and Perplexity of 2-Layer Stacked RNN over 10 epochs, with learning rate decay.

5.1 Future Work

5.1.1 Initialize Word Embeddings with GloVe

Pennington, Socher, and Manning introduced the State-of-the-Art word embeddings in [8]. They combine co-occurence statistics with prediction methods as follows:

$$J(\Theta) = \frac{1}{2}\sum_{i,j=1}^{W} f(P_{ij})({u_i}^T v_j - log(P_{ij}))^2$$

5.1.2 Clip Gradients

Karpathy et al. [6] state that clamping gradients elementwise was crucial to their success. In future work, we hope to evaluate the effect of clipping gradients in similar fashion in our recurrent networks.

5.1.3 Bidirectional RNN

5.2 Bidirectional RNN

We hope to extend our evaluations to include that of a bidirectional RNN for image captioning, as defined by Karpathy et al.:

$$x_t = W_w I_t$$

$$e_t = f(W_e x_t + b_e)$$

$${h_t}^f = f(e_t + W_f {h_{t-1}}^f + b_f)$$
$${h_t}^b = f(e_t + W_b {h_{t-1}}^b + b_b)$$

Figure 23: Mammogram Image of a Cancerous Lump, published by Stanford [2].

$$s_t = f(W_d({h_t}^f + {h_t}^b) + b_d)$$

[6]

5.2.1 Extension to New Image Domain: Mammography

Dr. Daniel Rubin, Associate Professor of Radiology and Medicine at Stanford University, and Francisco Gimenez, PhD, have provided with me a .csv file of 270,706 lines of natural language narratives, each mapped to an accession number of an imaging scan. These imaging scans are stored in a picture archiving and communication system (PACS) at Stanford Hospital.

Each report includes anatomical observations, along with a classification along various Breast Imaging Reporting and Data System (BI-RADS) categories (I-VI). We will apply our image captioning to the imaging data, and we hope to achieve better performance than a radiology clinician. I will place my generated narratives (withholding the ground-truth reports) and ask a group of radiologists, to evaluate the performance of the LSTM. Another gauge of my model's success will be its ability to generate text describing images with very malignant breast cancers.

Acknowledgments

Thank you to Justin Johnson for his helpful training and model selection advice.

References

[1] Xinlei Chen et al. "Microsoft COCO Captions: Data Collection and Evaluation Server". In: *CoRR* abs/1504.00325 (2015). URL: http://arxiv.org/abs/1504.00325.

[2] X. Chen et al. "Microsoft COCO Captions: Data Collection and Evaluation Server". In: *arXiv:1504.00325* (2015).

[3] Salah El Hihi and Yoshua Bengio. "Hierarchical Recurrent Neural Networks for Long-Term Dependencies". In: *Advances in Neural Information Processing Systems 8 (NIPS'95)*. Ed. by D. S. Touretzky, M. Mozer, and M.E. Hasselmo. MIT Press, 1996. URL: http://www.iro.umontreal.ca/~lisa/pointeurs/elhihi_bengio_96.pdf.

[4] Klaus Greff et al. "LSTM: A Search Space Odyssey". In: *CoRR* abs/1503.04069 (2015). URL: http://arxiv.org/abs/1503.04069.

[5] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[6] Andrej Karpathy and Fei-Fei Li. "Deep visual-semantic alignments for generating image descriptions." In: *CVPR*. IEEE Computer Society, 2015, pp. 3128–3137. ISBN: 978-1-4673-6964-0. URL: http://dblp.uni-trier.de/db/conf/cvpr/cvpr2015.html#KarpathyL15.

[7] Razvan Pascanu et al. “How to Construct Deep Recurrent Neural Networks”. In: *CoRR* abs/1312.6026 (2013). URL: http://arxiv.org/abs/1312.6026.

[8] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. “GloVe: Global Vectors for Word Representation”. In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[9] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. “CIDEr: Consensus-based Image Description Evaluation”. In: *CoRR* abs/1411.5726 (2014). URL: http://arxiv.org/abs/1411.5726.

[10] Oriol Vinyals et al. “Show and tell: A neural image caption generator”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 2015, pp. 3156–3164. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298935. URL: http://dx.doi.org/10.1109/CVPR.2015.7298935.

6 Experimental Evaluation and Findings

Computation of BLEU, CIDEr, and METEOR metrics are forthcoming. However, these quantitative Metrics cannot reason about new, accurate information that the network has learned about the images that was not present in the ground truth.