VISUAL IMAGE DESCRIPTION

Yash Aggarwal[1], Sidhant Arora[2] , Nidhi Sengar[3]
Department of Information Technology,
Maharaja Agrasen Institute of Technology, Delhi, India

Abstract : This study has been undertaken to build networks capable of putting words to images, that relate to both the scene and the real world, and to output clear image captions. All these tasks that we as people can do without much effort.
We thought that the reader may benefit from image captioning and its applications. Largely, image captioning helps in retrieval of images, by allowing us to sort and request pictorial or image-based content in new ways. This model can drastically help and aid the visually-impaired with subtitles or annotations in real-time. However, we think that image captioning is far more than what the current applications present and can find use in numerous fields like science, art or even economics.

1 INTRODUCTION

Image caption generation has emerged as a challenging and important research area following advances in statistical language modelling and image recognition. The generation of captions from images has various practical benefits, ranging from aiding the visually impaired, to enabling the automatic and cost-saving labelling of the millions of images uploaded to the Internet every day. The field also brings together state-of-the-art models in Natural Language Processing and Computer Vision, two of the major fields in Artificial Intelligence.

There are two main approaches to Image Captioning: bottom-up and top-down. Bottom-up approaches, such as those by [1] [2] [3], generate items observed in an image, and then attempt to combine the items identified into a caption. Top-down approaches, such as those by [4] [5] [6], attempt to generate a semantic representation of an image that is then decoded into a caption using various architectures, such as recurrent neural networks. The latter approach follows in the footsteps of recent advances in statistical machine translation, and the state-of-the-art models mostly adopt the top-down approach.

Our approach draws on the success of the top-down image generation models listed above. We use a deep convolutional neural network to generate a vectorized representation of an image that we then feed into a Long-Short-Term Memory (LSTM) network, which then generates captions. Figure 1 provides the broad framework for our approach.

One of the main challenges in the field of Image Captioning is overfitting the training data. This is because the largest datasets, such as the Microsoft Common Objects in Context (MSCOCO) dataset, only have 160000 labelled examples, from which any top-down architecture must learn (a) a robust image representation, (b) a robust hidden-state LSTM representation to capture image semantics and (c) language modelling for syntactically-sound caption generation. The problem of overfitting

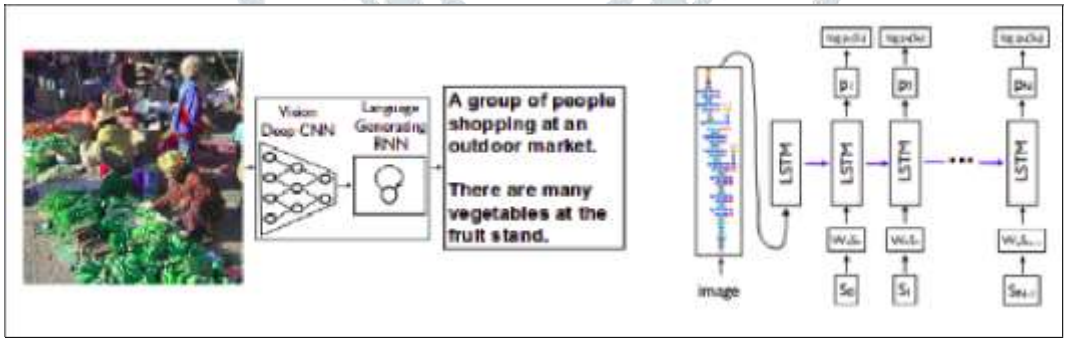


Figure 1: (Left) Our CNN-LSTM architecture, modelled after the NIC architecture described in [6]. We use a deep convolutional neural network to create a semantic representation of an image, which we then decode using a LSTM network. (Right) A unrolled LSTM network for our CNN-LSTM model. All LSTMs share the same parameters. The vectorized image representation is fed into the network, followed by a special start of sentence token. The hidden state produced is then used by the LSTM predict/generate the caption for the given image. Figures taken from [6]

manifests itself in the memorization of inputs and the use of similar sounding captions for images which differ in their specific details. For example, an image of a man on a skateboard on a ramp may receive the same caption has an image of a man on a skateboard on a table.

To cope with this, recent advances in the field of Image Captioning have innovated at the architecture-level, with the most successful model to date on the Microsoft Common Objects in Context competition using the basic architecture in

Figure 1 augmented with an attention mechanism [7]. This allows it to deal with the main challenge of top-down approaches, i.e. the inability to focus the caption on small and specific details in the image. In this paper, we approach the problem via thorough hyper-parameter experimentation on the basic architecture in Figure 1.

2 RELATED WORK

In this section, we describe relevant background on recurrent neural networks and image caption generation. Recently, several methods have been experimented with for automatic image caption generation. [1] first proposed learning a mapping between images, meanings and captions using a graphical model based on human-engineered features. The pioneering use of neural networks for image caption generation was suggested by the multi-model pipeline in [8], which demonstrated that neural networks could decode image representations from a CNN encoder and that also showed that the resulting hidden dimensions and word embeddings contained semantic meaning (i.e. ”image of a blue car” - ”blue” + ”red” produces vectors close to that produced by ”image of a red car”).

Top-down approaches: These initial efforts were followed by [6] and [9] which used more modern CNNs for encoding and replaced feedforward networks in [8] with recurrent neural networks, in particular LSTMS [10]. [9] also demonstrated the use of these models on video captioning tasks. One of the main contributions of [6] was that it showed that a LSTM that did not receive the image vector representation at each time step was still able to produce state-of-the-art results, unlike the earlier work by [8]. The common theme of these works is that they represented images as the top layer of a large CNN (hence the name ”top-down” as no individual objects are detected) and produced models that were end-to-end trainable.

Bottom-up approaches: [11] instead approach the problem by dividing it into two simpler prob- lems. Firstly, they train a CNN and bi-directional RNN that learns to map images and fragments of captions to the same multimodal embedding, demonstrating state-of-the-art results on informational retrieval tasks. Secondly, they train a RNN that learns to combine the inputs from various object fragments detected in the original image to form a caption. This improved on previous works by allowing the model to aggregate information on specific objects in the image rather than working from a singular image representation. A similar line of research was pursued in [12], which trained object detectors to identify fragments in images and proposed a three-step pipeline for combining these detected fragments into a caption. However, one disadvantage of these models is that they were not end-to-end trainable.

One way of bridging and compensating for the weaknesses of the two approaches above is atten- tion. There is an extensive line of research around incorporating attention-mechanisms in neural networks, such as in question-answering [13], handwriting generation [14] and image generation [15]. Attention allows models to focus on specific aspects of the input while ignoring others; the model has to learn what to focus on. This has been addressed via reinforcement learning techniques in [16] and with variational auto-encoders in [15]. A correlate of attention mechanisms is also the iterative generation of outputs rather than the single-pass approach adopted in most encoder-decoder frameworks, where outputs are iteratively constructed through a series of modifications emitted by the decoder, each of which is observed by the encoder [15]. This project directly extends the work of simpler architectures from [6] and [9].

3 APPROACH

The model framework adopted in this paper is analogous to recent successful approaches in statistical machine translation. Using an encoder recurrent neural network, these models learn an ex- pressive representation of the original sentence, and use another recurrent neural network to decode that representation in the target language. The advantages of using recurrent neural networks and the model architecture above are the ability to handle sequences of arbitrary length, and more importantly, the end-to-end maximization of the joint probability of the original and target sentence, which have produced state-of-the-art results in machine translation. Drawing inspiration from these approaches, we propose a natural extension by ”decoding” a caption given an image ”encoding”.

3.1 Model architecture overview

Figure 1 contains the architecture of the model trained in this paper. We represent an image using the 1024 1 final layer of GoogleNet, denoted as $g(I)$ for an image I. We train a linear transformation of $g(I)$ that maps it into the 512 1 input dimensions expected by our LSTM network. This entire pipeline of image representation generation is represented by:

$$CNN(I) = W^{(I)}g(I) + b^{(I)} \quad (1)$$

We initialize a recurrent neural network with initial state equal to zero. We then feed the image representation $CNN(I)$ in as the first input of a dynamic length LSTM, i.e. $x_{-1} = CNN(I)$. Subsequent inputs are the start of sentence token and all the words in the sentence, denoted by $x_t = W_e S_t$ for $t = 0...N$ 1 where S_i is a V 1 one hot vector representing word i, S_0 and S_N are one hot vectors representing special start of sentence and end of sentence tokens, and W_e is a $512x\ V$ word embedding matrix. Each hidden state of the LSTM emits a prediction for the next word in the sentence, denoted by $p_{t+1} = LSTM(x_t)$ for $t = 0...N$ 1. The model is fully described by the set of equations:

$$x_{-1} = CNN(I) \quad (2)$$

$$x_t = W_e S_t \text{ for } t = 0 \ldots N-1 \quad (3)$$

$$p_{t+1} = LSTM(x_t) \text{ for } t = 0 \ldots N-1 \quad (4)$$

(5)

Finally, we evaluate the parameters of the model at each iteration using the cross entropy loss of the predictions on each sentence. The loss function minimized is therefore:

$$J(S|I;\theta) = -\sum_{t=1}^{N} \log p_t(S_t|I;\theta) \quad (6)$$

where $p_t(S_t)$ is the probability of observing the correct word S_t at time t. This loss is minimized with regards to parameters in the set θ, which are all the parameters of the LSTM above, the param- eters of the CNN and the word embeddings.

3.2 LSTM caption generator

The LSTM function above can be described by the following equations where $LSTM(x_t)$ returns p_{t+1} and the tuple (m_t, c_t) is passed as the current hidden state to the next hidden state.

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1}) \quad (7)$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1}) \quad (8)$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1}) \quad (9)$$

$$c_t = f_t \circledS c_{t-1} + i_t \circledS tanh(W_{cx}x_t + W_{cm}m_{t-1}) \quad (10)$$

$$m_t = o_t \circledS c_t \quad (11)$$

$$p_{t+1} = \text{Softmax}(m_t) \quad (12)$$

The forget gates f_t allow the model to selectively ignore past memory cell states and the input gates i_t allow the model to selectively ignore parts of the current input. The output gate o_t then allows the model to filter the current memory cell for its final hidden state. The combination of these gates bestow (1) an ability to learn long-term dependencies and reset these dependencies conditioned on certain inputs and (2) the avoidance of vanishing and exploding gradients.

3.3 Input representation

In the literature of recurrent neural networks, the information contained by the sequence of past words $S_0, S_1, .., S_t$ at time $t + 1$ is represented by a fixed length hidden state h_t. The next hidden state is a non-linear function of the past hidden state and the current input, which produces an updated memory that can capture non-linear dependencies through time.

$$h_{t+1} = f(h_t, x_{t+1}) \quad (13)$$

Having fully defined our model in the overview, we now need to make concrete our choice of the function f and the way we represent inputs x. The function f that is implemented in this paper is the LSTM. The LSTM cell [10] has become increasingly popular in recent years due to its ability to capture long-term dependencies in sequence prediction problems and to cope with the vanishing

/ exploding gradient problems in recurrent neural networks.

Images: To represent images, we propose the use of a convolutional neural network to map images I to fixed length vector representations. Deep convolutional neural networks have achieved state- of-the-art performances in image classification in recent years. Specifically, we use the architecture of GoogleNet [6] which achieved the best performance in ILSVRC 2014 using an innovative batch normalization technique. Then, $x = CNN(I)$ is a D_i 1 vector, where D_i is the fixed dimension of any inputs to the LSTM.

Words: To represent words, we use a word embedding of size D_i V where V is the size of the vocabulary.

4 EXPERIMENTS

Dataset: We measure the performance of this architecture on the Microsoft Common Objects in Context (MSCOCO) dataset. The MSCOCO data comprises 82783 training images, 40504 validation images and 40775 test images. Each image is accompanied by at least five captions of varying length. In the training set, there are 414113 captions in total, for an average of 5.002 captions per image. We preprocess the caption dataset by replacing words that

appear less than five times in the training dataset with an ¡UNK¿ token, prepend each sentence with a SOS token, and append each sentence with a EOS token. The final vocabulary size is 8843. The mean and median length of the post-processed captions is 12.55 and 12 respectively. Figure 2 plots the histogram of caption lengths. There are long right tails in the empirical distribution, with the maximum caption length topping out at 57. We train all models on the entire training dataset and tune our hyperparameters on the validation set. We hold out 4000 images from the validation set as our test set as per theGoogle paper [6] to make our results comparable.

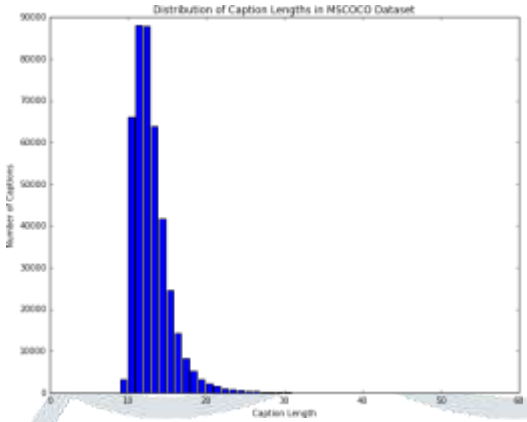
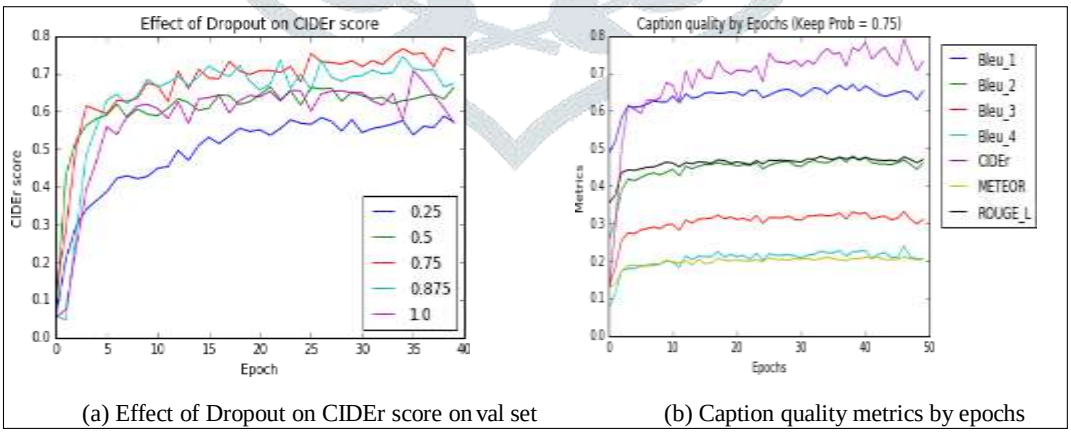


Figure 2: Distribution of Caption Lengths in the MSCOCO dataset. The mean caption length is 12.55. There is a substantial right tail in the empirical distribution.

Metrics: Given recent advances in statistical machine translation, state-of-the-art models have be- gun to progress beyond BLEU-1 scores to BLEU-4 scores. Hence, we report our models' performance on BLEU-4. In addition, we also report our score on recently devised metrics `METEOR` and `CIDEr` which test for alignment with ground truths and human consensus respectively, therefore capturing additional improvements to caption quality beyond BLEU metrics.

Benchmarks: We reference three benchmarks to gauge the difficulty of the problem and the improvement our model brings. The first benchmark is a random generation of words from the vocabulary until the end-of-sentence token is emitted. The second benchmark is a nearest neighbors approach which compares image vectors and returns the caption of the closest image. The last benchmark taken from the original Google paper [6] are human generated captions from Amazon Mechanical Turk. These are displayed in Table 1. There is a significant discrepancy of 11.8 BLEU- 4 points and 48.9 CIDEr points between Human-generated captions and the nearest-neighbor approach, indicating this is indeed a difficult problem to solve. We also report the results of two papers [6] and [17] that achieve state of the art results using a similar model and an attention model respectively.

4.1 Dropout



(a) Effect of Dropout on CIDEr score on val set (b) Caption quality metrics by epochs

Figure 3: Hyper parameter tuning on the validation set

In image caption generation problems, overfitting is a common problem due to the relatively small number of training examples for the complexity and ideal diversity of the generated captions. To combat this, we first conduct extensive hyper parameter tuning on dropout. Figure 3 contains a few

Table 1: Scores on the MSCOCO validation set

METRIC	*BLEU-4*	*METEOR*	*CIDER*
4 Layer CNN-LSTM	23.1	21.1	78.2
3 Layer CNN-LSTM	23.7	**21.7**	81.3
2 Layer CNN-LSTM	**24.4**	21.5	**81.7**
1 Layer CNN-LSTMs:			
$KP = 1.0$	21.3	19.9	70.8
$KP = 0.875$	23.3	20.8	74.5
$KP = 0.75$	23.9	21.1	79.1
$KP = 0.50$	21.3	19.8	67.2
$KP = 0.25$	19.7	18.5	59.6
Google NIC [6]	**27.7**	**23.7**	**85.5**
Hard-Attention [17]	25.0	23.04	-
Random	4.6	9.0	5.1
Nearest Neighbor	9.9	15.7	36.5
Human	21.7	25.2	85.4

interesting insights. Firstly, the plot of metrics by epochs on the right shows that most metrics plateau at around epoch 5 while the CIDEr score keeps on increasing. However, the qualitative analysis of generated captions by epoch conducted in Figure 4 (left) in the following section show that caption quality is indeed improving, just perhaps not in a way that is captured by the BLEU scores. Hence, we choose to use the CIDEr score as our main hyperparameter tuning metric. We then study the effect of dropout on CIDEr scores on the validation set on the left chart in Figure 3, which plots keep probabilities of 0.25, 0.50, 0.75, 0.875, 1.00 . Making the keep probability too low reduces the ability of the model to capture signal in the training set, leading to poorer max CIDEr scores and slower convergence as shown by the purple (keep prob = 0.25) line. However, making it too high leads to overfitting, as shown by the red line (keep prob = 0.75) being above the green line for almost all epochs on the validation set. Given the difference between 0.75 and 1.00, we ran an additional experiment investigating if 0.875 would perform better than 0.75 but results did not improve. Hence, for future model experiments, we fix the dropout keep probability = 0.75.

4.2 Depth of LSTM network

We also examined the effect of adding more layers to the decoder LSTM network. This was driven by the observation that some of the validation metrics (except CIDEr) were beginning to max out at around epoch 5-10 in Figure 3, which suggests we reach the limits of the single-layer LSTM network early in the training regime. Adding additional layers leads to improvements in all metrics: BLEU-4, METEOR and CIDEr. Table 1 displays the scores of all our tested models on the three caption quality metrics with varying LSTM depth and dropout probabilities. Adding additional layers beyond a second layer does not show improvements as the model begins to overfit, as shown by the slight dip in BLEU-4 and CIDEr scores between the 2 layer and 3 layer CNN-LSTMs.

We fix the model parameters at dropout keep-probability = 0.75 and number of layers = 2 and proceed with qualitative evaluations of the generated captions.

4.3 Qualitative analysis of generated captions

Figure 4 (left) displays a sample of generated captions from our CNN-LSTM architecture on a set of previously unseen images. These were generated using our best model, the 2-layer, 0.75 keep probability CNN-LSTM. Captions higher up in Figure 4 (left) are generated at earlier epochs. We note that the CNN-LSTM model is increasingly able to learn to distinguish important details in images. For example, in Image 2, it first identifies a man riding a skateboard, and then begins

to

Figure 4: (Left) Change in generated captions over time. Captions higher up in the figure were generated at earlier epochs. The CNN-LSTM model learns to identify pictures in increasing detail and corrects its earlier mistakes. These captions were generated from our validation set. (Right) A sample of generated captions from unseen validation images, categorized by error type.

identify snow around epoch 20, before realizing that this is a person on a snowboard in the air, but with snow in the background. We also see evidence of correction of mistakes. In Image 4, it first identifies the red train in the background as a red umbrella, but is able to remove that from its generated caption in a later epoch, as it improved its understanding of what an umbrella is. In addition, Figure 4 shows areas where the model needs to improve. It is unable to distinguish a doughnut from a hot dog in Image 3, although identifying 'a young boy' over 'a man' and 'a hot dog' instead of 'a cell phone' are good improvements. It leaves out the chopsticks in the background and the red train / train tracks in Images 1 and 4 respectively, which by qualitative evaluation are crucial details if these captions are to be useful. It also is unable to identify that Image 5 does not contain a suit, having been confused by the tie-shaped necklace being donned by the man.

Figure 4 (right) provides a sample of generated captions on unseen images. We categorized them into three groups: Correct, Partially Correct and Wrong. Captions in which there is a minor error or for which the caption is technically correct but misses the point of the image are labelled as 'Partially Correct'. For example, the street with an Apple store in China (as deduced by Chinese characters in the background signage) is described as 'a street with cars and cars on the street'. Ignoring the syntactic awkwardness of the generated caption, it is also thoroughly uninformative. There are also captions where it is relatively clear that the model has been confused by parts of the image. For example, the image for the 'herd of elephants walking through a field' contains significant greenery around the elephant enclosure, which undoubtedly contributed to the error. The sink in the kitchen image looks similar to a stove, causing the model to identify two stoves in the image. Lastly, the man in the suit is holding a camera, but the model may have seen various 'selfie' images in the same pose, causing the identification of a cell phone rather than a camera. In the 'Wrong' category, the teddy bear is erroneously identified as a cat and street signs are identified as stop signs.

4.4 Visualizing effect of emitted words on hidden states

This section explains our methodology of visualizing and interpreting hidden states in our LSTM and presents some of our main findings. The majority of prior research has treated the encoder- decoder framework as a black box, focusing instead on evaluating the generated captions. We show that identical emitted words move the hidden state in similar ways despite differences in the previous context (i.e. different images).

Methodology: We use our best model to generate captions from the validation set on unseen images. In the caption generation process, we feed an image into our CNN which produces an encoding that is fed into our LSTM for decoding. As words are emitted, we feed the current predicted sentence back into the LSTM for it to predict the next word. As we feed each new predicted word into the LSTM, we record the hidden state that results from the combination of the new word and the previous hidden state. We do this for the 4000 held-out images in our validation set.

These have a median sentence length of 12 leading to approximately 48000 hidden states of dimension 512 1. We use t-SNE [18] on the entire 48000 512 matrix to reduce the space of hidden states to two dimensions, and then generate plots of the hidden state for each sentence.

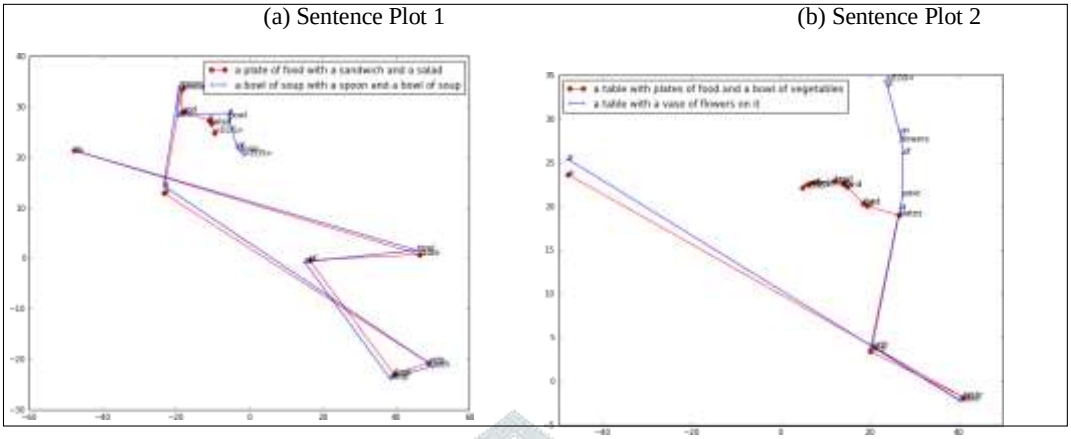


Figure 5: Two examples of sentence pairs that demonstrate that the hidden dimension reflects se- mantic meaning and that emitted words affect different contexts in similar ways. The legend of the plots contains our generated captions.

Results: The charts in the middle column of Figure 5 contain the projection of the hidden states of a sentence into two dimensions. The first sentence pair demonstrates that emitted words with close semantic meanings ('plate' and 'bowl', 'food' and 'soup') move the hidden state in the same direction, despite being conditioned on different image vectors. The second sentence pair shows that the emitted sequence 'a table with' moves the hidden dimension in similar ways and that the hidden state sentence representation only diverges once the sentence begins to describe differences in the images. For example, the two line graphs diverge substantially only once the unrelated concepts of 'food' and 'vase' are emitted.

5 Conclusion

The model can be used for smaller applications but not for commercial ones as it is limited by the dataset provided for training and testing of the model. Executing further epochs would result in over fitting of the model.

Given the resources and data available, the results are satisfactory and can be further worked upon to increase efficiency for more complex applications. We conducted an extensive hyperparameter search over the CNN-LSTM model architecture, pro- ducing a best model that achieves results that are 3.3 BLEU-4 points and 3.8 CIDEr points behind the state-of-the-art, using a keep probability of 75% for dropout and two layers for our decoder LSTM network. A thorough quantitative and qualitative analysis of the generated metrics suggests that the model is able to sensibly caption a wide variety of images from the MSCOCO dataset. Partial errors tend to occur due to lack of attention to specific details in images (e.g. labeling a picture of elephants walking in an enclosure as 'elephants in a field' due to being misdirected by trees in the background). This suggests that the attention-mechanisms explored in recent work may yield improvements to this task.

Our main novel contribution to the field is exploring the effect of emitted words on hidden states in the LSTM that were previously treated as black boxes. We demonstrated that semantically-close emitted words (e.g. 'plate' and 'bowl') result in similar movements in hidden state despite different previous context and that divergences in hidden state occur only upon emission of semantically-far words (e.g. 'vase' and 'food').

References

[1] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 15–29, Berlin, Heidelberg, 2010. Springer-Verlag.

[2] Polina Kuznetsova, Vicente Ordonez, Alexander C. Berg, Tamara L. Berg, and Yejin Choi. Collective generation of natural image descriptions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 359–368, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[3] Siming Li, Girish Kulkarni, Tamara L. Berg, Alexander C. Berg, and Yejin Choi. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Confer- ence on Computational Natural Language Learning*, CoNLL '11, pages 220–228, Strouds- burg, PA, USA, 2011. Association for Computational Linguistics.

[4] Xinlei Chen and C. Lawrence Zitnick. Learning a recurrent visual representation for image caption generation. *CoRR*, abs/1411.5654, 2014.

[5] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L. Yuille. Deep captioning with multi- modal recurrent

neural networks (m-rnn). *CoRR*, abs/1412.6632, 2014.

[6] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555,2014.

[7] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. *CoRR*, abs/1603.03925,2016.

[8] Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. Unifying visual-semantic embed- dings with multimodal neural language models. *CoRR*, abs/1411.2539, 2014.

[9] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venu- gopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, abs/1411.4389, 2014.

[10] Sepp Hochreiter and J ürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[11] Andrej Karpathy and Fei-Fei Li. Deep visual-semantic alignments for generating image de- scriptions. *CoRR*, abs/1412.2306,2014.

[12] Hao Fang, Saurabh Gupta, Forrest N. Iandola, Rupesh Kumar Srivastava, Li Deng, Piotr Doll ár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Ge- offrey Zweig. From captions to visual concepts and back. *CoRR*, abs/1411.4952, 2014.

[13] Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Pe- ter Ondruska, Ishaan Gulrajani, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. *CoRR*, abs/1506.07285, 2015.

[14] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[15] Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. DRAW: A recurrent neural network for image generation. *CoRR*, abs/1502.04623, 2015.

[16] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. *CoRR*, abs/1406.6247,2014.

[17] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

[18] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.