
Image Captioning with Recurrent Neural Networks

Kaiyuan Wang

k5wang@ucsd.edu

Merve Kilic

mkilic@ucsd.edu

Utkarsh Jain

utjain@ucsd.edu

Marialena Sfyraiki

msfyraiki@ucsd.edu

Abstract

Image Captioning is a prominent problem in computer vision and natural language processing. The main goal of this problem is to generate a caption that closely describes an image. In this regard, we explore an encoder-decoder architecture that encodes an image using a Convolutional Neural Network (CNN) and uses it to generate captions with a Recurrent Neural Network (RNN). In this project, we choose the pre-trained ResNet50 model as our encoder and explore the power of Long Short-Term Memory (LSTM) networks to generate captions. We experiment with Vanilla RNNs and also try passing the encoded image at every time step in our LSTM decoder to observe their effect on the quality of the generated captions. We use the COCO dataset and after hyperparameter tuning achieve a performance¹ of **(1.67, 66.66, 7.76)** for the baseline model, **(1.68, 64.53, 6.50)** for the Vanilla RNN model, and **(1.60, 67.37, 7.58)** for the model with the image encoding given at each timestep. From our experiments, we find that the model with image encoding passed at every time step gives the best performance, followed by the LSTM model and then the Vanilla RNN model. Furthermore, we notice that stochastic sampling is superior to deterministic sampling while generating captions. Lastly, we notice how different hyperparameters effect the overall performance and we discuss our observations later.

1 Introduction

Image Captioning task is an important problem in artificial intelligence which employs knowledge from computer vision and natural language processing to generate captions that closely describe an image. Obtaining such automatically generated descriptions for images has a myriad of applications such as helping visually impaired people to understand the world and their surroundings, early childhood education, image retrieval, etc.

In this project, we employ a neural network model that takes in an image as input and generates a caption for it. The whole model contains an encoder that encodes an image as a feature representation and a decoder that uses the feature representation to generate captions. The encoder uses a pre-trained ResNet50 model [3], for generating a feature representation for a given image. The decoder learns a dense feature embedding for the words in the vocabulary and uses it, along with the encoded image, to generate captions. The pre-trained encoder is frozen during training. To train the decoder, we use teacher-forcing that uses the teaching signal from the training dataset at the current time step, $target(t)$, as input in the next time step $x(t+1) = target(t)$, rather than the output $y(t)$ generated by the network. This is found to make the learning faster and lead to quicker convergence.

In our experiments, we use a subset of the Common Objects in Context (COCO) dataset [5] which is a large-scale object detection, segmentation, key-point detection, and captioning dataset with over

¹performance reported in form of (test loss, bleu1, bleu4)

328K images. However, we only use 9,000 training images and around 2,000 validation images respectively for training and validating the model. Figure 1 shows that our model learns to generate meaningful captions that closely describe the images. Our model can be further improved by using more powerful image encoders and decoders motivated by recent advancements.



Figure 1: Examples of images, actual captions, and generated captions

2 Related Work

The COCO dataset was introduced in [5] with the goal of placing the question of object recognition in the context of the broader question of scene understanding. Among other annotations, every image is associated with 5 captions. The COCO dataset now serves as a benchmark dataset in the deep learning research community. Other similar datasets include Flickr8K, Flickr30K, and IAPR TC-12.

More recently [8] used the COCO dataset as the benchmark for their novel and effective approach of using attention mechanism over image features to generate better captions. In their work, the authors used a similar encoder-decoder approach as ours, however, instead of encoding an image as a one-dimensional feature vector, they extract multiple features from a lower convolutional layer, each of which is a representation corresponding to a part of the image. Then, using an attention module, they generate a context vector, which is a dynamic representation of the relevant part of the image, at every timestep. Using the context vector, along with the LSTM state and previous word, they compute the output word probability for the next word. Based on their experiments, they achieved state-of-the-art performance on several metrics on various benchmarking datasets, including COCO dataset. Based on a similar idea of attention mechanism [1] proposed a novel combined bottom-up and top-down visual attention mechanism that enables attention to be calculated more naturally at the level of objects and other salient regions. The bottom-up mechanism in their model works by proposing several “regions of interest” in the image, each associated with a feature vector. The top-down mechanism then weighs each feature during caption generation, using the existing partial output sequence as context, and generates a combined feature that is used as input to the decoder.

ResNet was proposed in [3] where the authors presented a residual learning framework to ease the training of networks that are substantially deep. They experiment with residual networks of various depths and achieved state-of-the-art results on the ImageNet dataset and COCO object detection dataset. In our project, we use a pre-trained ResNet50 model (depth of 50 layers) to encode the images.

Vanilla RNN (or Elman RNN) was introduced in [2] where the author brought forth a novel neural network architecture that incorporated the concept of “time” in neural networks. This was achieved by using recurrent connections between hidden units and memory units, and trainable parameters from the memory units to the hidden units. LSTM network was proposed in [4] to address the issue of vanishing gradients in Vanilla RNNs. It achieves this by using several gates which control the gradient flow and effectively deal with the vanishing gradient problem. We will be using Vanilla RNNs and LSTMs in our project as a decoder to generate captions.

3 Methods

In this section, we explain the architecture of the encoder and decoder module of our model (please see Figure 2), the decoding strategies we use during evaluation, how our model learns the word embedding, and the training procedure.

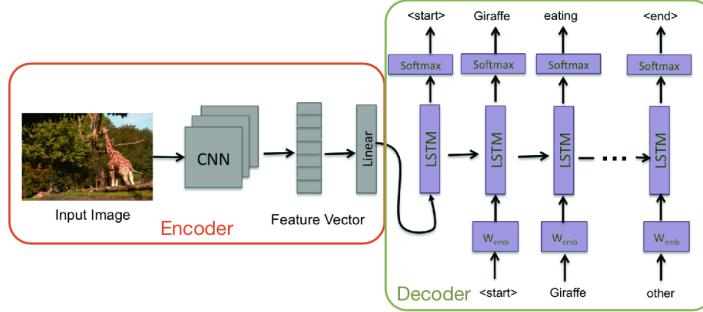


Figure 2: Encoder-Decoder Architecture for Image Captioning. [7]

3.1 Encoder

For our experiments, we use a pre-trained ResNet50 network [3]. However, we remove its last fully-connected layer and add a trainable linear layer to map from the penultimate hidden layer dimension to a desired image embedding dimension. This encoded representation is then passed to the decoder where it is used to generate the captions.

The dimension of the image embedding is set to the word embedding dimension in the LSTM and Vanilla RNN decoder because we pass the encoded image as an input in the first timestep. However, this constraint is relaxed in Architecture 2 because the image is concatenated with the words and then passed to the LSTM at each time step.

3.2 Vanilla RNN Decoder

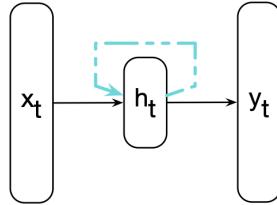


Figure 3: Vanilla Recurrent Neural Network

In this section, we consider the most basic class of RNNs called Vanilla Networks (or, Elman RNNs). Fig 3 illustrates the working behind a Vanilla RNN. The input vector at time t , x_t , is multiplied to a weight matrix and passed through a non-linear function to generate h_t , which is also called the hidden state at time t . This hidden state is then used to compute the corresponding output, y_t . As you can see in the figure, the recurrent link (represented with a dashed line) combines the hidden state from the preceding point in time, i.e h_{t-1} and input x_t to generate h_t . Using h_{t-1} this way serves as a memory, or context, that encodes previous computations and guides future decisions. Given x_t and h_{t-1} , h_t is computed as follows:

$$h_t = \tanh(Uh_{t-1} + Wx_t) \quad (1)$$

where, $U \in R^{d_h \times d_h}$, $W \in R^{d_h \times d_{in}}$, $h_t, h_{t-1} \in R^{d_h}$, $x_t \in R^{d_{in}}$, and $h_0 = \{0\}^{d_h}$ (d_{in} and d_h are the input and hidden sizes respectively).

In our decoder, we use a two-layered Vanilla RNN to form a stacked RNN, with the second RNN taking in the outputs of the first RNN and generating the final hidden states. These hidden states are then passed through a linear layer with a softmax activation function to generate a probability distribution over the set of words in the vocabulary. A word is then sampled from this distribution using the strategy discussed in section 3.5. The sampled word is embedded into a lower dimensional space and passed as input to the stacked RNN in the next time step. The encoded image from the encoder is used as the input in the first time step and the hidden states are initialized to zeros.

3.3 LSTM Decoder

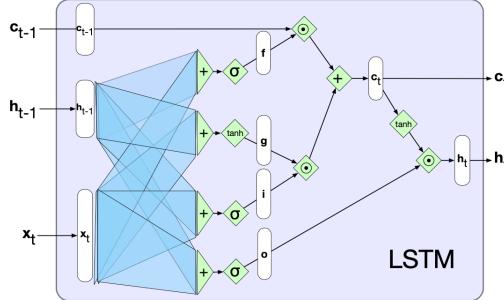


Figure 4: A single Long Short-Term Memory Network Unit

An LSTM processes information in the same way a Vanilla RNN does. However, to maintain relevant information for a longer period of time, it employs the concept of a ‘memory cell’. It also uses different gates, such as forget gate, update gate, and output gate, that allow more control over the gradient flow and effectively deals with the vanishing gradient problem. Figure 4 gives an accurate visualization of how all the computations occur.

In our decoder, we use a two-layered LSTM to form a stacked RNN, with the second LSTM taking in the outputs of the first LSTM and generating the final hidden states. These hidden states are then passed through a linear layer with a softmax activation function to generate a probability distribution over the set of words in the vocabulary. A word is then sampled from this distribution using the strategy discussed in section 3.5. The sampled word is embedded into a lower dimensional space and passed as input to the stacked LSTM in the next time step. The encoded image from the encoder is used as the input in the first time step. The hidden states and cell states are initialized to zeros.

3.4 Architecture 2

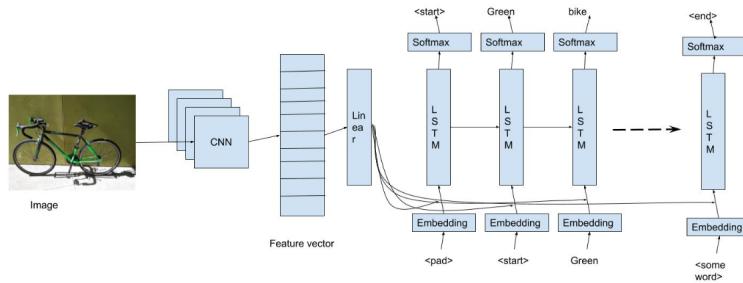


Figure 5: Architecture 2 for Image Captioning

In the previous models, we were using the encoded image as the input for the first time step. In architecture 2, however, the encoded image is concatenated with the word embedding for “*< pad >*” and passed to the stacked LSTM as input in the first timestep. The output is used to generate a word using the strategy discussed in section 3.5. The sampled word is embedded into a lower dimensional space, concatenated with the encoded image, and passed as input to the stacked LSTM in the next

time step. Since we are no longer using the encoded image as a sole input to the LSTM, and rather using it along other words, we no longer have to make sure that the image encoding size matches the word embedding size. As usual, the hidden states and cell states are initialized to zeros. Please refer to Figure 5 for easy visualization of this process.

3.5 Generating Strategy

The output from every time step from our decoder basically returns a probability distribution over the entire set of vocabulary, and we use this distribution to select a word for the next time step. There are two strategies for choosing the next word:

- **Deterministic Sampling:** In this strategy, we choose the most probable word and use it in the next time step. Although this sounds intuitive, in practice it has certain downsides. First, for a particular image, it will always generate the same caption which is not optimal. Second, it is seen this strategy is prone to grammatical errors and doesn't perform very well when generating long sequences.
- **Stochastic Sampling:** In this strategy we use the probability distribution over the vocabulary to sample a word. This results in a more novel and realistic caption generation. However, in order to gain more control over the randomness of the generated caption, we augment the softmax function with temperature scaling. The scaling is done by dividing the logit vector (the raw scores generated by the final linear layer) by a temperature term T , followed by the softmax function.

$$y_j = \frac{\exp(o_j/T)}{\sum_n \exp(o_n/T)} \quad (2)$$

where o_j is the output from last layer, n is the size of the vocabulary, and T is the temperature.

The temperature controls how stochastic the sampling is: as the temperature approaches 0, the distribution becomes nearly deterministic and collapses to a point mass, and as it goes to infinity, the distribution becomes completely uniform, thus increasing the uncertainty.

3.6 Training Procedure

To train our model we use the concept of teacher-forcing which uses the teaching signal from the training dataset at the current time step, $\text{target}(t)$, as input in the next time step $x(t+1) = \text{target}(t)$, rather than the output $y(t)$ generated by the network. In simpler terms, while training, instead of using the words predicted by the decoder in the next timestep, we use the ground-truth label as the input. However, during inference, since we do not have the ground-truth labels, we let the network run on its own by using the network's prediction to obtain the next input word. We also limit the size of the generated caption to 20 during inference.

We keep the pre-trained ResNet50 encoder frozen during training and only learn the parameters associated with the linear layer after the encoder, the embedding layer in the decoder, the RNN, and the final linear layer when backpropagating over the error. We use CrossEntropyLoss over the raw scores produced by the final linear layer to compute the error and gradients, and employ the Adam optimizer for gradient descent. We then conduct experiments to adjust the hyperparameters of our models.

Experiments

We run four kinds of experiments in this project. In the first one, we try tuning the learning rate and batch size of the baseline model while keeping the rest of the hyperparameters at default configuration (word embedding dimension: 300, two hidden layers of 512 LSTM units). We try the values 1e-3, 5e-4, and 1e-4 for the learning rate, and 32, 64, and 128 for the batch size. This corresponds to section 3.1 in the assignment handout.

Second, for the Vanilla RNN model, we keep the same set of hyperparameters as the baseline model, but tune over the learning rate and batch size. This corresponds to section 3.2.1 in the assignment

handout. For the learning rate, we try the values 1e-3, 5e-4, and 1e-4, while the values we use for the batch size are 32, 64, and 128.

For the third experiment, we try tuning the embedding and hidden sizes of the baseline model (using the optimal learning rate and batch size found from the first experiment) to improve its performance. This corresponds to section 3.2.2 in the assignment handout. We try values of 200, 300, and 400 for the embedding size and values of 256, 512, and 1024 for the hidden dimension.

For the fourth experiment, we tune Architecture 2 over the image embedding dimension, word embedding dimension, and learning rate. This corresponds to section 3.3 in the assignment handout. We try values of 200, 300, and 400 for the image embedding dimension, 300, 400, and 600 for the word embedding dimension, and 1e-4, and 5e-4 for the learning rate.

Please check section 4 for the optimal setting of hyperparameters and its evaluation on the test dataset for each experiment.

3.7 Learning Word Embeddings

We use PyTorch’s nn.Embedding layer to map every word in the vocabulary to a low-dimensional embedding space that is initialized randomly drawn from a uniform distribution. During training, the gradients from the loss function backpropagate to this layer, and the associated parameters are adjusted to optimize the model’s performance.

4 Results

Hyperparameter Settings and Evaluation

The best setting of hyperparameters for each model and the corresponding BLEU-1 scores, BLEU-4 scores, and cross entropy loss value computed on the test set are reported in Table 1. Please check appendix A for hyperparameter tuning experiments.

Model	Image Dim	Word Dim	Hidden Dim	lr	bs	Bleu1	Bleu4	Test Loss
Baseline	300	512	1e-4	32	66.65182	7.54300	1.69074	
Vanilla RNN	300	512	1e-4	64	64.53590	6.50821	1.68191	
Tuned Baseline	400	512	1e-4	32	66.66819	7.76160	1.67754	
Architecture 2	200	400	512	1e-4	64	67.37103	7.57806	1.60253

Table 1: Best hyperparameters for each model and test evaluation

Training and Validation Loss Curves

The plots of training and validation loss vs number of epochs for each model are illustrated in Figure 6.

Examples of Generated Captions

Please check appendix B for examples of captions generated by the tuned baseline model for different temperature settings, appendix C for the tuned RNN model, and appendix D for tuned Architecture 2.

5 Discussion

Here we discuss the results and explain some of the observations we made while running the experiments.

Model Comparison

Based on our experiments and performance on test dataset, we conclude that Architecture 2 works the best as compared to the other two in all evaluation metrics. We believe this is because by

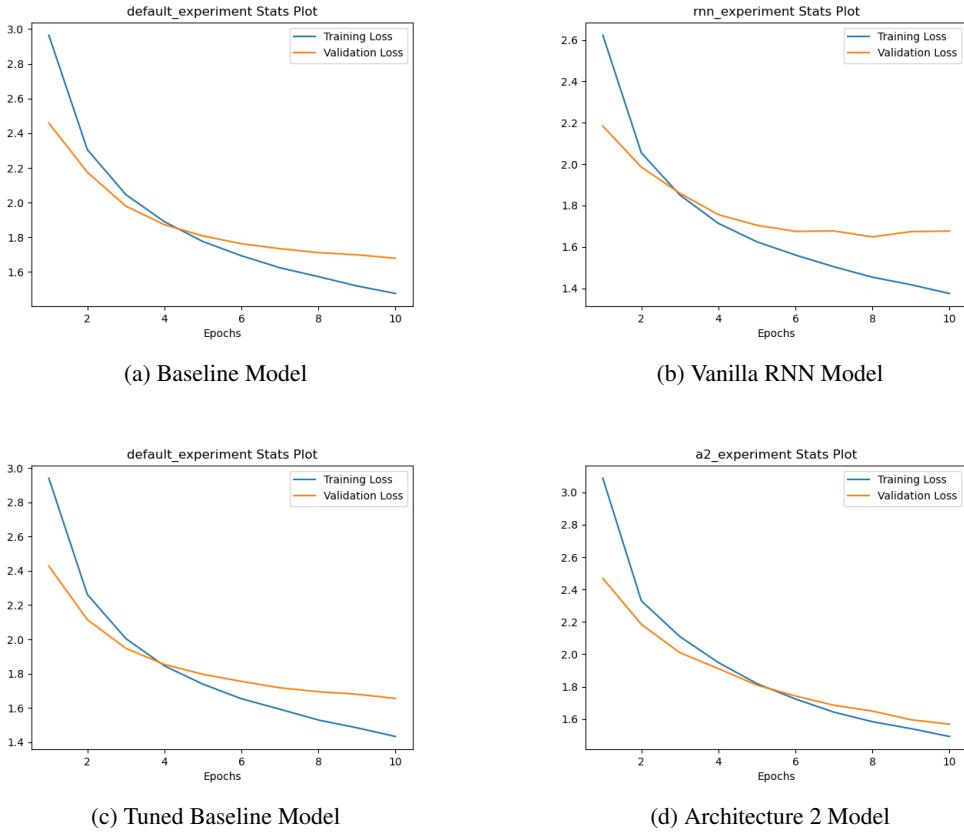


Figure 6: Train/Validation curve using the best hyperparameters for each model

feeding the image at every time step to the model, we are essentially letting the model focus less on “remembering” the image and focus more on learning representations that lead to better caption quality. On the other hand, since we are feeding the image only at the first time step, we make it harder for the other two models to retrieve the image embedding information at later time steps. This is because after every time step, the other two models lose some of the encoded image information. This is equivalent to the “information bottleneck” problem in the Seq2Seq Machine Translation networks [6]. From Figure 6, we also notice that the validation loss curve closely follows the training loss curve for Architecture 2 and shows no sign of overfitting until the 10th epoch. Because of the limited computational power, we could not train it for longer, but we expect Architecture 2 to perform even better with more training.

We also notice that the baseline model with the LSTM decoder performs better than the Vanilla RNN decoder model. This might be true because of the superiority of LSTM networks in dealing with the vanishing gradients problem and the overall flow of information and gradient in the network. Vanilla RNNs suffer from the vanishing gradients problem which prevents them from learning effective representations as compared to LSTMs. This can also be seen by the difference in the test loss and the BLEU scores of the two models. While experimenting with the two models, we also notice that LSTMs do a better job of generating longer captions than the Vanilla RNN model. This might be true because of the design of LSTMs that helps them remember meaningful information for longer periods of time which in turn helps them generate more “continuous” captions. Furthermore, based on the training/validation loss curve in Figure 6, we can see that the Vanilla RNN model starts overfitting after the 8th epoch, whereas the baseline model continues learning more. We notice that the baseline takes longer to overfit and produces better results.

Furthermore, while tuning the baseline model, we basically run two kinds of experiments. In the first kind of experiment, we tune over learning rate and batch size, and in the second one, we tune over word embedding dimension and hidden dimension. Based on our experiments, and Table 1,

we see that tuning word embedding dimension and hidden dimension produced better results and we notice a good drop in the test loss. Not just that, but we also noticed an improved BLEU1 score and a sharp increase in the BLEU4 score. Also, we observe that increasing the word embedding dimension generated better results, which might be because it increases the overall representational capacity of the model. Based on the training/validation loss curve in Figure 6, we also notice that the tuned baseline model achieves an overall lower training and validation loss by the 10th epoch and still shows no sign of overfitting.

Effect of Hyperparameters

While tuning the word embedding dimension and hidden dimension, we found that increasing the hidden dimension did not help as much as increasing the word embedding dimension. This is possible because to generate better captions, the model basically should learn better word representations. So, in a way the word embedding dimension kind of acts like a bottleneck here. Making it too small would prevent the model to learn effective word representations. However, making the word embedding dimension too high also resulted in lower performance. The same trend can be seen with the image embedding size as well.

We also observe that using a learning rate of 1e-4 produced much better results than a learning rate of 5e-4. This felt strange at first because both values are pretty low and in the previous assignments we usually found 1e-3 to produce optimal results. This might be because RNNs work very differently than the feed-forward networks we have seen in past assignments. Basically, in an RNN, since we use the same weights over all the time steps, the gradients keep on accumulating on the same sets of weights again and again. Using a lower learning rate helps the model take smaller steps over those “higher” gradient values.

Caption Generation Strategy

Through our experiments, we found that stochastic sampling is superior to deterministic sampling. Stochastic generation offers better diversity, i.e. the ability to choose from a wider range of potential words at each time step. Stochastic generation also offers better robustness against inferior outputs because there’s always a variety of word choices at any given time step. Whereas, with the deterministic strategy we always generate the same caption which is not optimal. Also, we observed that the best captions generated by the deterministic approach had lower BLEU scores as compared to those generated by the stochastic sampling approach.

While experimenting with different values of temperature in stochastic sampling, we found that lower values of temperature usually resulted in better BLEU1 and BLEU4 performance and also produced meaningful captions. However, reducing it to very low values resulted in almost deterministic caption generation. Meanwhile, increasing it to high values usually resulted in lower BLUE1 and BLEU4 performance. This makes sense if we see how stochastic sampling works. Using a really low-temperature value just makes the probability distribution concentrate on one word, which then leads to deterministic caption generation. Whereas, a high-temperature value makes the probability distribution more and more uniform and the model ends up picking words that might not describe the image or make sense in the sentence in general.

Scope for Improvement

We believe that using pre-trained word embeddings, like GloVe vectors, and fine-tuning them on our task would result in better performance. This is because GloVe vectors and other commonly available word embeddings are trained on huge datasets and do a better job of learning word representations.

Furthermore, during our experimentation, we saw that although our models did a good job of generating captions, they still sometimes missed out on a few details in the image. If you see appendix B you would see a lot of such examples. For instance, the caption generated for the left most image in Figure 7 completely misses out the bottle of wine in the image. We believe that using an encoder that can pay attention to the details and generate a more effective encoding could help our decoder to generate better captions.

6 Team Contribution

- **Kaiyuan Wang:** Worked on implementing the train() and val() functions in the experiment.py file. Also wrote Visualization.ipynb notebook which helps visualize generated captions and the associated images. Tuned the baseline model on word embedding dimension and hidden dimension. Wrote the first three sections of the report and collaborated on other sections.
- **Utkarsh Jain:** Worked on implementing and integrating early stopping mechanism during training and loading the best model during test time. Worked on implementing the resnet encoder and the baseline decoder in the model_factory.py file. Ran experiments on baseline model. Wrote the “Methods” section of the report and worked with others on finishing the report.
- **Merve Kılıç:** Worked on implementing the test() function in the experiment.py file. Implemented the Vanilla RNN decoder architecture and integrated it with the Decoder() class in model_factory.py file. Ran experiments on the Vanilla RNN model. Worked on the “Methods” section and “Results” section of the report.
- **Marialena Sfyraiki:** Worked on understanding how BLEU scores work and implemented the compute_bleu_score() helper function in the experiment.py file. Implemented Architecture 2 and integrated it with the Decoder() class in model_factory.py file, and wrote the generate_captions() function in the decoder. Ran experiments on Architecture 2. Worked on the “Methods” and “Discussion” sections of the report.

References

- [1] Peter Anderson et al. “Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.
- [2] Jeffrey L. Elman. “Finding structure in time”. In: *Cognitive Science* 14.2 (1990), pp. 179–211. ISSN: 0364-0213. DOI: [https://doi.org/10.1016/0364-0213\(90\)90002-E](https://doi.org/10.1016/0364-0213(90)90002-E). URL: <https://www.sciencedirect.com/science/article/pii/036402139090002E>.
- [3] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. DOI: 10.48550/ARXIV.1512.03385. URL: <https://arxiv.org/abs/1512.03385>.
- [4] 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: <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [5] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2014. DOI: 10.48550/ARXIV.1405.0312. URL: <https://arxiv.org/abs/1405.0312>.
- [6] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. “Sequence to Sequence Learning with Neural Networks”. In: *CoRR* abs/1409.3215 (2014). arXiv: 1409.3215. URL: <http://arxiv.org/abs/1409.3215>.
- [7] Anurag Tripathi, Siddharth Srivastava, and Ravi Kothari. “Deep Neural Network Based Image Captioning”. In: *Big Data Analytics*. Ed. by Anirban Mondal et al. Cham: Springer International Publishing, 2018, pp. 335–347. ISBN: 978-3-030-04780-1.
- [8] Kelvin Xu et al. *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*. 2015. DOI: 10.48550/ARXIV.1502.03044. URL: <https://arxiv.org/abs/1502.03044>.

Appendix A: Hyperparameter Tuning

Model	Image Dim	Word Dim	Hidden Dim	lr	bs	Bleu1	Bleu4
Baseline	300	512	1e-3	64	61.17416	6.32826	
			5e-4	64	62.53496	6.99643	
			1e-4	64	65.65333	6.85300	
			1e-4	32	66.65182	7.54300	
			1e-4	128	62.51799	5.25601	
Vanilla RNN	300	512	1e-3	64	53.11193	2.71601	
			5e-4	64	62.24884	6.39600	
			1e-4	64	64.53590	6.50821	
			5e-4	32	61.51282	5.90524	
			1e-4	32	63.55750	6.90025	
			5e-4	128	61.95502	6.20733	
			1e-4	128	62.22749	5.68200	
LSTM	200 200 300 300 400 400 400	512 1024 256 1024 256 512 1024	1e-4	64	65.07194	6.66888	
			5e-4	64	63.65530	7.35463	
			1e-4	32	65.12327	6.32260	
			1e-4	32	66.64962	8.03710	
			5e-4	64	64.68249	7.49722	
			1e-4	32	66.66819	7.76160	
			1e-4	32	66.38546	8.11547	
LSTM arch2	200	300	512	1e-4	64	66.43983	7.21069
	200	400		1e-4	64	67.37103	7.57806
	400	400		1e-4	64	66.42556	7.39015
	400	400		5e-5	64	65.00900	5.91048
	600	600		1e-4	64	54.98837	1.57368

Table 2: Hyperparameter Tuning

Appendix B: Captions with Baseline Decoder

Captions generated with hyperparameter setting of: word embedding dimension = 400; hidden dimension = 512; learning rate = 1e-4; batch size = 32; temperature = 0.1 unless specified differently.

Actual captions:
 1. a small table that is made into a computer .
 2. a computer , a keyboard and a mouse and a bottle of wine on ...
 3. a sony computer with a website about theory iether reason
 4. a phone with a keyboard , mouse , and a bottle of tea
 5. a tiny laptop hooked up to a mouse and keyboard

Generated caption (100.0 blue1):
 a laptop and a keyboard on a table .



Actual captions:
 1. a couple of teddy bears sitting on either side of a christma...
 2. two teddy bears are sitting on a table with a tree .
 3. two dressed up teddy bears are sitting by some lights
 4. two teddy bears are arranged around a tiny christmas tree .
 5. teddy bears sitting with a christmas tree with lights

Generated caption (100.0 blue1):
 two teddy bears are sitting on a table .



Actual captions:
 1. a red stop sign sitting under a green street sign .
 2. a pole with a stop sign on it , plus a street sign for w. vli...
 3. a light pole with a stop sign , a street sign and a pair of ...
 4. a pair of white shoes hanging from a street sign
 5. a pole with a stop sign , street sign , and a pair of shoes ...

Generated caption (100.0 blue1):
 a stop sign and a street sign on a pole



Figure 7: Good captions produced by baseline model

Actual captions:
 1. a computer animated girl brushes her teeth while two adults ...
 2. an animation picture of some people getting ready
 3. simulated photo of a cartoon girl with adults nearby in...
 4. a cgi girl brushing her teeth in front of two cgi adults .
 5. a picture of some some characters in a hallway

Generated caption (17.647 blue1):
 a woman holding a white and black tie and a woman standing ext to a person .



Actual captions:
 1. a tall surf boarder riding small breaking wave on open ocean .
 2. a man surfing across a wave in the ocean .
 3. a man surfing the waves on his surf board .
 4. man surf boarding on small waves on a clear day .
 5. a man riding waves at the ocean on his surf board

Generated caption (17.647 blue1):
 a large body of water with a large body of water



Actual captions:
 1. a high speed passenger train is bound for new york .
 2. the subway train is going fast up the tracks .
 3. a silver train on train station next to signage .
 4. a train station with a train moving down the tracks .
 5. an abstract photograph of a moving train on its way to new y...

Generated caption (17.647 blue1):
 a city street filled with cars and cars



Figure 8: Bad captions produced by baseline model

Actual captions:
 1. a large airliner jet flying through the blue sky .
 2. an airplane flying in a clear blue sky .
 3. a large white airplane flying through the sky .
 4. the bottom of an airplane flying in the sky .
 5. white jet plane flying in the sky with engines in the wings ...

Generated caption (88.889 blue1):
 a large airplane flying through a blue sky .



Actual captions:
 1. a baseball player holding a bat next to a dugout .
 2. a batter holding the bat up to hit the ball .
 3. a major league baseball player gets ready to hit a ball .
 4. batter warming up near dug out during major game .
 5. a baseball player practicing his swing at a game .

Generated caption (88.889 blue1):
 a baseball player is swinging a bat at a ball .



Actual captions:
 1. a young boy eating breakfast in bed .
 2. a little boy sitting on a bed with stuffed animals and a foo...
 3. a boy is sitting in bed eating some food
 4. a little boy eating off a tray in bed .
 5. a boy is sitting in bed and eating breakfast from a tray .

Generated caption (88.889 blue1):
 a woman sitting on a couch with a stuffed animal .



Figure 9: Deterministic captions produced by baseline model

Actual captions:
 1. a woman in yellow plays a game of tennis .
 2. a woman is playing tennis on a court
 3. a female tennis athlete in yellow dress returns a volley
 4. a woman hitting a tennis ball on the court in a tournament
 5. there is a woman playing a game of tennis .

Generated caption (88.889 blue1):
 a man is playing tennis on a court .



Actual captions:
 1. a man on a surf board rides a wave
 2. a man on a surfboard riding a wave in the ocean
 3. a young man in a colorful swim suit and yellow shirt is ridi...
 4. a person riding a surfboard on a wave in the ocean .
 5. a person in shorts and a tee surfing in the ocean

Generated caption (88.889 blue1):
 a man in a wetsuit riding a wave on a surfboard .



Actual captions:
 1. a train on train tracks at a train station
 2. a train sitting next to a railway platform .
 3. a train stopped and letting people on .
 4. multi colored locomotive at the front of train cars in depot...
 5. a locomotive is pulled up to a shiny boarding area .

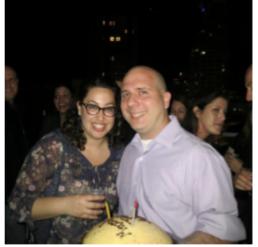
Generated caption (88.889 blue1):
 a train on a train track in a station .



Figure 10: Low-temperature captions (temperature = 0.001) produced by baseline model

Actual captions:
 1. man holding up a yellow cake with candles on the top of it .
 2. a young couple holding a small cake smiling most happily .
 3. a man and a woman in a bar holding drinks and smiling .
 4. a man and woman posing for a photo with a cake .
 5. a man and a woman stopped to smile and pose for a photo .

Generated caption (0 blue1):
 obstetrical deplaining precariously ornamental greeted assembly fours source contain beans be v-shaped



Actual captions:
 1. a plate with meat and vegetables next to a fork .
 2. a meal consists of beef , broccoli and cauliflower , and bla...
 3. a white plate topped with a salad and beans .
 4. white plate with beans and vegetables on a striped place ma...
 5. a close up of a plate of food with broccoli

Generated caption (0 blue1):
 posed eat impact feed kitty jog dives apart backs jog s
 squatting protecting legged huddle windup loved of exploring



Actual captions:
 1. a pan filled with an assortment of vegetables and a wooden s...
 2. a skillet full of broccoli and vegetables cooking .
 3. a pan filled with stir fry vegetables and a wooden spoon .
 4. some food is cooking in a small pan
 5. vegetables being prepared in a pot with a wooden spoon .

Generated caption (0 blue1):
 posters chimney competitive beads motorful female ornamental figure several ink shelving sides



Figure 11: High-temperature captions (temperature = 5) produced by baseline model

Appendix C: Captions with Vanilla RNN Decoder

Captions generated with hyperparameter setting of: word embedding dimension = 300; hidden dimension = 512; learning rate = 1e-4; batch size = 64; temperature = 0.1 unless specified differently.



Figure 12: Good captions produced by Vanilla RNN model

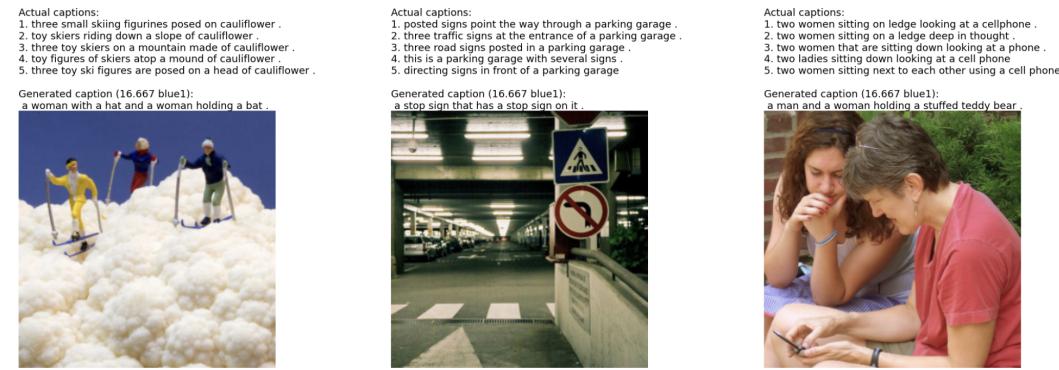


Figure 13: Bad captions produced by Vanilla RNN model

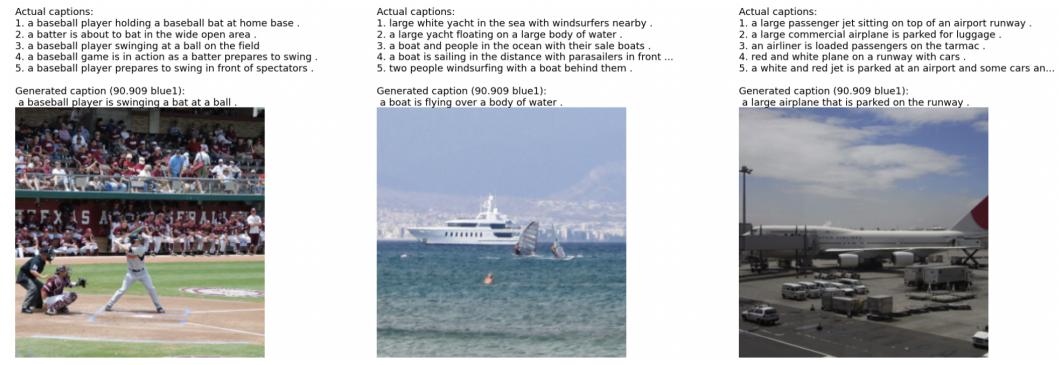


Figure 14: Deterministic captions produced by Vanilla RNN model

Actual captions:
 1. a man is playing tennis and is going to return the ball
 2. a man playing tennis on a tennis court.
 3. a man at the net makes a forehand volley in a tennis match .
 4. a young man playing tennis and preparing to hit the ball .
 5. a man is on a court with a tennis racket



Actual captions:
 1. a group of people are reading a menu at the table
 2. a group of people sit at a large table while talking
 3. people sitting on the long table with plates of food .
 4. a long table full of people on both sides .
 5. a long table accommodating many people while eating



Actual captions:
 1. close up of a plate with food on it .
 2. a piece of fish on a sandwich next to a lemon .
 3. this is a fish sandwich on a bun with a slice of lemon on th...
 4. a sandwich that is on a plate with the top piece of bread of...
 5. a plate that has bread and chicken on it .



Figure 15: Low-temperature captions (temperature = 0.001) produced by Vanilla RNN model

Actual captions:
 1. a room full of people sitting down on chairs .
 2. people in business attire are sitting at a conference .
 3. a man in a suit holding a soccer ball with other people arou...
 4. a group of people seated in chairs together and facing the s...
 5. a group of men and women are seated

Generated caption (6.25 blue1):
 automotive agitated sculture downwards champagne blinds amon...
 gst capable thermos between office unload cabbage sole mini...
 -fridge full



Actual captions:
 1. a large building with a clock on it 's face and a bird statu...
 2. this is a picture of an ornate building with a statue of an ...
 3. a colorful building with a clock tower has waving flags .
 4. a clock tower on an official building with an eagle on top
 5. a train station with an eagle statue at the top .

Generated caption (6.25 blue1):
 upcoming code papers relaxed pinata pencil apex recovers tom...
 ates hedgehog curve you wedge lorikeet tray darkly slipping...
 g statement



Actual captions:
 1. a banana peel on a piece of board .
 2. a banana hanging from the side of a wooden fence .
 3. a banana peel hanging over the top rung of a wooden fence .
 4. a banana peel draped over a fence board
 5. a peeled banana sitting on a wooden fence .

Generated caption (6.25 blue1):
 blue bushels polar fronted siberian nesting vert flowerpot I...
 ogos camouflage electronic peas presenting disturbed instru...
 ment



Figure 16: High-temperature captions (temperature = 5) produced by Vanilla RNN model

Appendix D: Captions with Architecture 2

Captions generated with hyperparameter setting of: image embedding size = 200, word embedding dimension = 300; hidden dimension = 512; learning rate = 1e-4; batch size = 64; temperature = 0.1 unless specified differently.

Actual captions:
 1. baseball players swing at a baseball with a bat during a gam...
 2. a baseball game with the batter hitting the ball.
 3. a batter swinging the bat at a baseball game .
 4. a batter is up during the baseball game .
 5. a professional baseball player swinging his bat at the baseb...

Generated caption (100.0 blue1):
 [a baseball player swinging a bat at a ball.]



Actual captions:
 1. an aeroplane soaring high in a clear sky .
 2. a prop airplane flying in a cloudy sky
 3. the under side of an airplane in flight .
 4. there is a plane flying in the sky
 5. a small airplane is flying in the gray sky .

Generated caption (100.0 blue1):
 [an airplane is flying in the sky.]



Actual captions:
 1. a man holding a tennis racquet on a tennis court .
 2. a man taking a swing at a tennis ball
 3. a tennis player is standing close to the net .
 4. a man in black shirt and blue shorts playing tennis .
 5. a man standing in the middle of a tennis court with a large ...

Generated caption (100.0 blue1):
 [a tennis player is playing tennis on a court.]



Figure 17: Good captions produced by Architecture 2

Actual captions:
 1. two women walking down the street holding umbrellas .
 2. view from behind of two women under umbrellas
 3. two people in dresses walking with umbrellas and bags .
 4. the back of two women holding up umbrellas and carrying bags .
 5. two people are walking away from the camera with umbrellas .

Generated caption (0 blue1):
 [a man is standing on a skateboard]



Actual captions:
 1. dog jumping in air to catch flying disc with adult sitting i...
 2. a dog jumping in the air is holding a frisbee .
 3. a dog jumping up in the air with a frisbee .
 4. an australian shepherd dog catches a pink frisbee in the par...
 5. a dog with a kerchief has jumped in the air catching a pink ...

Generated caption (0 blue1):
 [a woman holding a dog in a field]



Actual captions:
 1. black and white photo of cars parked on the street .
 2. large town of parked cars on lot and on street .
 3. an old , black and white street scene is pictured
 4. a bunch of old cars that are parked in a lot and on the stre...
 5. an old photo of a street view with many cars parked .

Generated caption (0 blue1):
 [a large street and a large building .]



Figure 18: Bad captions produced by Architecture 2

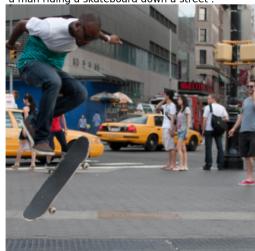
Actual captions:
 1. a collaboration of people in different pictures doing things .
 2. a series of images of young men painting and holding kites .
 3. various children and adults are making their own kites .
 4. a variety of colorful pictures with people doing various act...
 5. a few people working with colored fabrics in different ways ...

Generated caption (18.182 blue1):
 a woman sitting on a bench next to a woman .



Actual captions:
 1. a young man doing a flip on a skateboard in the middle of a ...
 2. a man is doing a skateboard trick in a city .
 3. a person riding a skate board on a city side walk .
 4. a man is doing skateboard tricks off the side of a busy stre...
 5. a man doing a leap on a skateboard on a city sidewalk .

Generated caption (18.182 blue1):
 a man riding a skateboard down a street .



Actual captions:
 1. a large united states air force plane sitting on a tarmac at...
 2. a series of planes and jet airliners on a sandy area .
 3. the large airplane has very large wings near many other plan...
 4. many air planes on a run way with a sky background
 5. air force plane on a runway with several other planes .

Generated caption (18.182 blue1):
 a large airplane is sitting on the runway .



Figure 19: Deterministic captions produced by Architecture 2

Actual captions:

1. two people standing next to each other on top of the snow .
2. two people standing in the snow with skies on the ground and...
3. two people standing in the snow with ski equipment .
4. two people with hats standing next to ski gear .
5. skiers with equipment looking at snow covered rise in distant...

Generated caption (11.111 blue1):

[a man is skiing down a snowy slope.]



Actual captions:

1. people at the baggage claim area of an airport .
2. three people standing before airport counters below airport ...
3. people standing at counters of booths being served .
4. the baggage delivery section of an air port .
5. patrons are going to the shops of an airport .

Generated caption (11.111 blue1):

[a group of people sitting on a table with a large clock]



Actual captions:

1. a collaboration of people in different pictures doing things .
2. a series of images of young men painting and holding kites .
3. various children and adults are making their own kites .
4. a variety of colorful pictures with people doing various act... .
5. a few people working with colored fabrics in different ways ...

Generated caption (11.111 blue1):

[a woman sitting on a bench next to a woman.]



Figure 20: Low-temperature captions (temperature = 0.001) produced by Architecture 2

Actual captions:

1. a photographer setting up his equipment for a photo shoot .
2. a person holding up a purple suit case .
3. a man wearing jeans holds up a purple suitcase .
4. a man holding a suit case above his head in front of a poste...
5. a man standing in a room holding a piece of purple luggage o...

Generated caption (0 blue1):

blooming nearest enjoying parasails hood unfolded
n 'n pancake together roast 2009 n n n recliner
bareback shower point marking vanity privacy linked condit



Actual captions:

1. a city bus is shown with a bike in front .
2. a street view , left , shows a curb with a yellow fire hydrant .
3. a bike is attached to the front of this bus .
4. a city bus has a bike on a front of it .
5. a bus that has a bike on the front of it .

Generated caption (0 blue1):

Cents breads container slave duke lookin n n n lande
d features chillin flowers n white highway edible



Actual captions:

1. one person holds the back of a surfboard while another perso...
2. a girl is standing on a surfboard while a man holds her Bal...
3. a girl standing on one leg on a surfboard while a man holds her Bal...
4. a person stands on one leg on one end of a surfboard while a...
5. a man holding a surfboard with a woman jumping off of it .

Generated caption (0 blue1):

clap sucking sorted avocado mingle



Figure 21: High-temperature captions (temperature = 5) produced by Architecture 2