Image Captioning with Recurrent Neural Networks

Henry Lister
Dept. of Computer Science
University of California, San Diego
La Jolla, CA 92093
hlister@ucsd.edu

Ryan Nishimoto
Dept. of Computer Science
University of California, San Diego
La Jolla, CA 92093
rnishimoto@ucsd.edu

Yash Shah
Dept. of Computer Science
University of California, San Diego
La Jolla, CA 92093
ynshah@ucsd.edu

Abstract

Recurrent Neural Networks' (RNNs') ability to keep track of past information makes them extremely efficient for tasks involving sequential inputs or outputs. Image captioning is one such task that requires generating outputs that are a sequence of words for which RNNs prove to be useful. In this work we present a deep RNN in action by captioning images taken from the Microsoft COCO 2014 dataset. We begin with a baseline RNN which consists of ResNet-50 as an encoder and an LSTM with 2 hidden layers as a decoder, optimized through Teacher forcing during training, experimenting with different temperatures for caption generation, hyperparameter tuning, and early stopping to prevent overfitting. This experiment achieves a BLEU-1 score of 63.159% and a BLEU-4 score of 7.054%. We then improve upon this model by changing the hidden size for our LSTM and the embedding size to achieve a BLEU-1 score of 65.66% and a BLEU-4 score of 6.976%. Thirdly, we replace our LSTM decoder with a Vanilla RNN trained using Backpropagation through time (BPTT) with the same hyperparameters as our baseline model with LSTM. This achieves a BLEU-1 score of 63.73% and a BLEU-4 score of 7.27%. Finally we tweak our LSTM decoder and feed in the image encodings at every time step along with the caption word embeddings. After experimenting with different hyperparameters, our model achieves a BLEU-1 score of 66.77% and a BLEU-4 score of 7.641%. In general, we find that including the image encoding for each time step, using Adam optimizer, and having close to equal or equal hidden and embedding sizes helped us better optimize our models and generate captions similar to what humans would provide for test data. [1]

1 Introduction

The task of image captioning involves automatically describing the content of an image. This problem situates itself at the intersection of computer vision—to be able to correctly identify main objects within an image without attending to background noisy objects—and natural language processing—generating captions like a human would using human language. We want to be able to keep track of past information (the image and words from a caption that we have already seen in previous time steps) to inform the decision we make for predicting the next word at the current time step. Recur-

[1] Code is available at https://github.com/cse151bwi22/cse151b-wi22-pa4-ynshah3

rent Neural Networks with their hidden states are specifically designed with the ability to process sequential information and is ideal for the task of image captioning [1].

A natural question to ask when using RNNs is what kind of model to use. Long Short-Term Memory (LSTM) models have almost entirely replaced Vanilla RNNs due to their ability to store information over extended periods of time while also creating a highway of efficient error backpropagation without being subjected to the vanishing gradients problem [2]. To deal with the problem of different types of input data—images and variable sized captions—we use an Encoder-Decoder architecture which uses a pretrained ResNet-50 to encode images and an Embedding Layer to create word embeddings from captions, both spanned onto the same space. A decoder RNN then takes this input and decodes it to generate captions [1].

In this work, we train three different types of RNN architectures to generate captions for images taken from the Microsoft COCO 2014 dataset. We begin with a baseline model with an LSTM as the decoder. We use 2 hidden layers, an embedding size of 300, and a hidden size of 512. We employ teacher forcing during training which helps the RNN to stay close to the ground-truth captions [3]. The model is optimized through hyperparameter tuning and using different temperatures to generate captions. Next, we experiment with the embedding size and the hidden size to improve the model. Thirdly, we replace our LSTM decoder with a Vanilla RNN trained using Back Propagation through Time (BPTT) and calculate BLEU scores using hyperparameters from the baseline model. Finally, we use an LSTM decoder but feeding in image encodings at every time step along with word embeddings from the captions.

Our baseline LSTM model achieves a BLEU-1 score of 63.159% and a BLEU-4 score of 7.054%. Experimenting with the embedding and hidden size achieves a BLEU-1 score of 65.66% and a BLEU-4 score of 6.976%. Our Vanilla RNN decoder achieves a BLEU-1 score of 63.73% and a BLEU-4 score of 7.27%. And our LSTM decoder with image encodings fed in at every time step achieves a BLEU-1 score of 66.77% and a BLEU-4 score of 7.64%. We find that including the image for every time step, using Adam optimizer, and having close to equal or equal hidden and embedding sizes helped us better optimize our models and generate captions similar to what humans would provide for test data. See results in Tables 1, 2, 3, and 4.

2 Related Work

For this work we referenced the Pytorch documentation [4] on how sequence to sequence learning, a task similar to image captioning, is carried out using RNNs and teacher forcing. Lecture slides [5, 6] were helpful in understanding how RNNs work.

Neural Image Caption (NIC) Generator. This paper [7] is a first in using the power of Concolutional Neural Networks (CNNs) as encoders to create rich representations of input images. Methods prior to this work made use of an encoder Recurrent Neural Network (RNN) to create word embeddings from captions as a way to perform sequence-to-sequence learning. This paper uses a Long Short-Term Memory (LSTM) model as a decoder since it is known to perform well on sequence tasks. In order to sample words from the output of the decoder, it uses Beam Search, which uses $k = 20$ best sentence candidates till time t to generate sentences of size $t + 1$. After tuning the model's hidden unit and embedding sizes and incorporating methods to prevent overfitting, it is able to score 27.2 BLEU-4.

Professor Forcing. We use Teacher forcing during training in order to force the model to output caption words that are close to the ground-truth captions. This paper [8] introduces the notion of Professor forcing to improve long-term sequence sampling. It is an adversarial method that is closely related to Generative Adversarial Networks (GANs). It tries to make the behavior of the model indistinguishable to whether the network is trained with or without teacher forcing by using an auxilliary model, a discriminator, to spot differences in behavior. This not only improved model generalization but also acted as a regularizer that sped up convergence.

3 Methods

Provided below is a description of how we sample outputs and create word embeddings, the architecture for our three models, and the range of values we sample for our hyperparameter search when training on the MS-COCO 2014 dataset.

3.1 Output Sampling and Word Embeddings

Each word in the caption is generated sequentially by feeding in an encoding of either the previous word generated (without teacher forcing), or the known caption to the LSTM decoder (with teacher forcing). To transform words into a form feedable to the LSTM, we use a trainable embedding layer which generates a one hot encoding for a word, then passes it through a trainable fully-connected linear layer with embedding size number of outputs. When training, we feed in the input encoding that we get from the encoder to the LSTM decoder. Since we use teacher forcing, the input to the LSTM for every time step after the first is the output of the word from the actual caption from the previous time step passed through the embedding layer to get a word embedding.

We generate captions using two different approaches. For the first approach, we use the word with the highest probability obtained after taking softmax over all word scores from the LSTM output. These word scores are obtained by first passing the LSTM output through a fully-connected linear layer with output size equal to the size of the vocabulary. This is the deterministic approach of caption generation since the word with the highest probability that we choose from the softmax is deterministic.

The other approach is to stochastically generate outputs which requires us to sample from a multinomial distribution obtained after dividing the word scores from the fully-connected layer with the temperature hyperparameter.

The temperature is some number greater than zero. When dividing the scores by a number between zero and one, the scores will increase distance between the small and large scores. The larger scores will have their values increased after all scores are put through a softmax, while the lower scores will have their values shrunk. In the limit, dividing by zero would lead to the highest score to dominate the softmax function and then sampling would always be equal to taking the index of the maximum of these scores. When the value is greater than one, the scores all become closer to each other and will lead to a distribution closer to a uniform distribution. Sampling with temperature less than one is like sampling from the outputs with greater confidence, while sampling with a larger temperature would be similar to having lower confidence in the outputs.

3.2 Baseline LSTM

Architecture. Our baseline LSTM Model is made up of two main components—an image encoder and a decoder. First, the images are fed into a frozen pretrained ResNet-50 model with the last fully connected layer modified to output the same number of features as the inputs to the LSTM (embedding size) and made to be trainable. The outputs from the ResNet-50 model are then fed into the LSTM network which is two layers deep. One final fully connected layer maps the output of the LSTM to a one hot encoding of the vocabulary.

The model itself has a ResNet-50 as the encoder with dropout introduced before the last fully-connected layer. The decoder has 2 layers with 512 hidden units each. The embedding size for the captions and the images is 512.

Hyperparameter Search. We explore a variety of different hyperparameters and their effects on the performance of the model measured by BLEU-1 and BLEU-4 scores. We experimented training with weight decay, learning rate, the number of hidden units, the embedding size, using different learning rate schedulers, and different optimizers. We also tested generation with a variety of temperatures for stochastic generation and also deterministic generation. The detailed results are summarized in the results section.

Our hyperparameter search spanned the set $\{0.01, 0.005, 0.0005, 0.00005\}$ for the learning rate, $\{0, 0.001, 0.0001\}$ for weight decay, {Deterministic, 0.1, 0.5, 0.01, 0.05, 0.001, 0.005, 0.0001, 0.0005, 0.00001, 0.00005} for the temperature, $\{256, 512, 1024, 2048, 4096\}$ for the hidden unit size, {150, 300, 512, 600, 1200} for the embedding size, {Adam, SGD} for the optimizer, {None, Step LR Scheduler, Annealing Cosine LR Scheduler} for the LR Scheduler, and dropout with 0.5 probability for the last fully connected layer of ResNet-50 for regularization. Here, 'deterministic' refers to deterministically sampling the best word from the model output without taking temperature into consideration.

3.3 Vanilla RNN Model

Architecture. The architecture for the vanilla RNN model is identical to the baseline LSTM model, except the LSTM cells in the baseline model are replaced with RNN cells. These are simpler than LSTM cells and do not have the extra gates that LSTM models have that allow them to be more resistant to the vanishing gradient problem. RNN cells are generally simpler than LSTM cells and are not as effective at holding important information for long periods of time compared to LSTM cells. The efficacy of the model is measured by BLEU-1 and BLEU-4 scores in the same way as the baseline LSTM model. Outputs are sampled from the decoder and input embeddings are obtained in the same way that they are for the baseline LSTM model.

Hyperparameter Search. The hyperparemeter search for the RNN model spans the following hyperparameters: learning rate: {0.0005,0.001,0.00005}, hidden units: {512,1024}, embedding size: {300,600}, LR scheduler: {none, Annealing Cosine LR Scheduler}, and temperature: {0.0001,0.001,0.1,0.5}. The optimizer was set to Adam and the weight decay was 0 for all experiments on the RNN model.

3.4 LSTM Model with Image Encodings at each time Step LR

Architecture. This architecture is identical to the Baseline LSTM architecture with a few minor adjustments to what inputs are passed in to the LSTM decoder. Instead of passing the output from the encoder at the first time step followed by the embedding of the previous word, the output from the encoder is concatenated with the embedding of the previous word at each time step and then passed into the LSTM decoder. For the first time step, the embedding of the token `<pad>` is concatenated with the image encoding.

Since we are concatenating image encodings with word embeddings, we can independently tune the embedding size for images and captions. For our best model, we use an embedding size of 150 for our captions and an embedding size of 300 for our image encodings. The LSTM decoder has 2 layers with 512 hidden units, the outputs from which pass through a fully-connected Linear layer to assign a score to each token in the vocabulary.

Hyperparameter Search. Our hyperparameter search encompasses the following sets—learning rate: {0.0005, 0.001}, weight decay: {0, 0.0004}, hidden units: {512, 1024}, embedding size for captions: {150, 300}, embedding size for images: {150, 300}, optimizer: {Adam, SGD}, LR Scheduler: {None, Step LR Scheduler, Annealing Cosine LR Scheduler}, and temperature: {Deterministic, 0.1, 0.5, 0.01, 0.05, 0.001, 0.005, 0.0001, 0.0005, 0.00001, 0.00005}. Here, 'deterministic' refers to deterministically sampling the best word from the model output without taking temperature into consideration.

We first test different embedding sizes and hidden units with the same temperature, optimizer, LR Scheduler, and weight decay. Once we have a model that performs best, we change the optimizer, LR Scheduler, and weight decay independently to see if there is a performance improvement. Finally, we test different temperatures to find the best model corresponding to the best temperature.

4 Results

In this section we go over experiments that we carry out for each model, plots that we generate, and model performance on training, validation, and test data.

4.1 Baseline LSTM Model

Best Hyperparameters The best performing LSTM Model was with a weight decay of 1e-4, a learning rate of 5e-4, with a temperature of 1e-3. The number of hidden units and embedding size was 512, and we used Adam as our optimizer with a cosine annealing learning rate scheduler. We added dropout regularization to the fully connected layer of ResNet-50.

Plot and Test Loss Figure 1 shows the plot for training and validation losses for our best model.

Analyzing the graph, both training and validation curves are decreasing. The validation curve almost completely overlaps the training curve, implying that the model does not overfit at all, and that the model performance might increase if we train the model for more epochs.

On the test data, we achieve a cross entropy loss of 1.615.

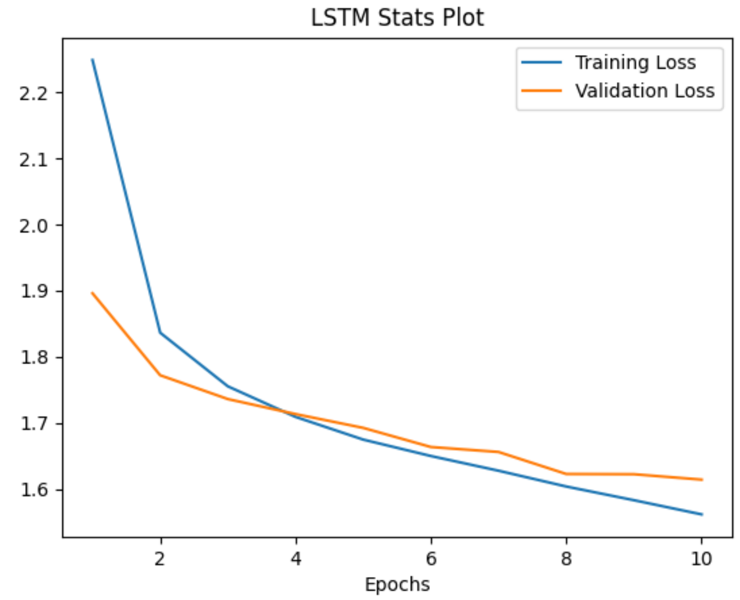


Figure 1: Training and validation losses for baseline lstm best model

BLEU Scores This model with our best hyperparameters was able to achieve a BLEU 1 score of 65.66 and a BLEU 4 score of 6.976. Table 1 shows the results for each experiment conducted.

From our experiments, we found that it took many more epochs when using SGD to get a BLEU scores close to some of the ones that we were seeing when using Adam. Changing the learning rate and adding decay did not seem to improve the performance of the model. Additionally, generating captions with temperature did not seem to have a positive impact on scores until we changed the embedding size. Once we had increased the embedding size from 300, we saw the model was actually early stopping due to overfitting. From there, we made adjustments to the learning scheduler and added a weight decay to help mitigate the overfitting and saw much better BLEU scores compared to the models with smaller embedding sizes. Based on our experiments, changing the embedding size was the most impactful on the BLEU score, and after doing so, we were able to adjust the other hyperparameters for further performance increase.

Generated Captions with Different Temperatures. Figures 2 and 3 provide images and their generated captions using the model with both a deterministic and stochastic approach with temperatures 5, 0.4, and 0.001.

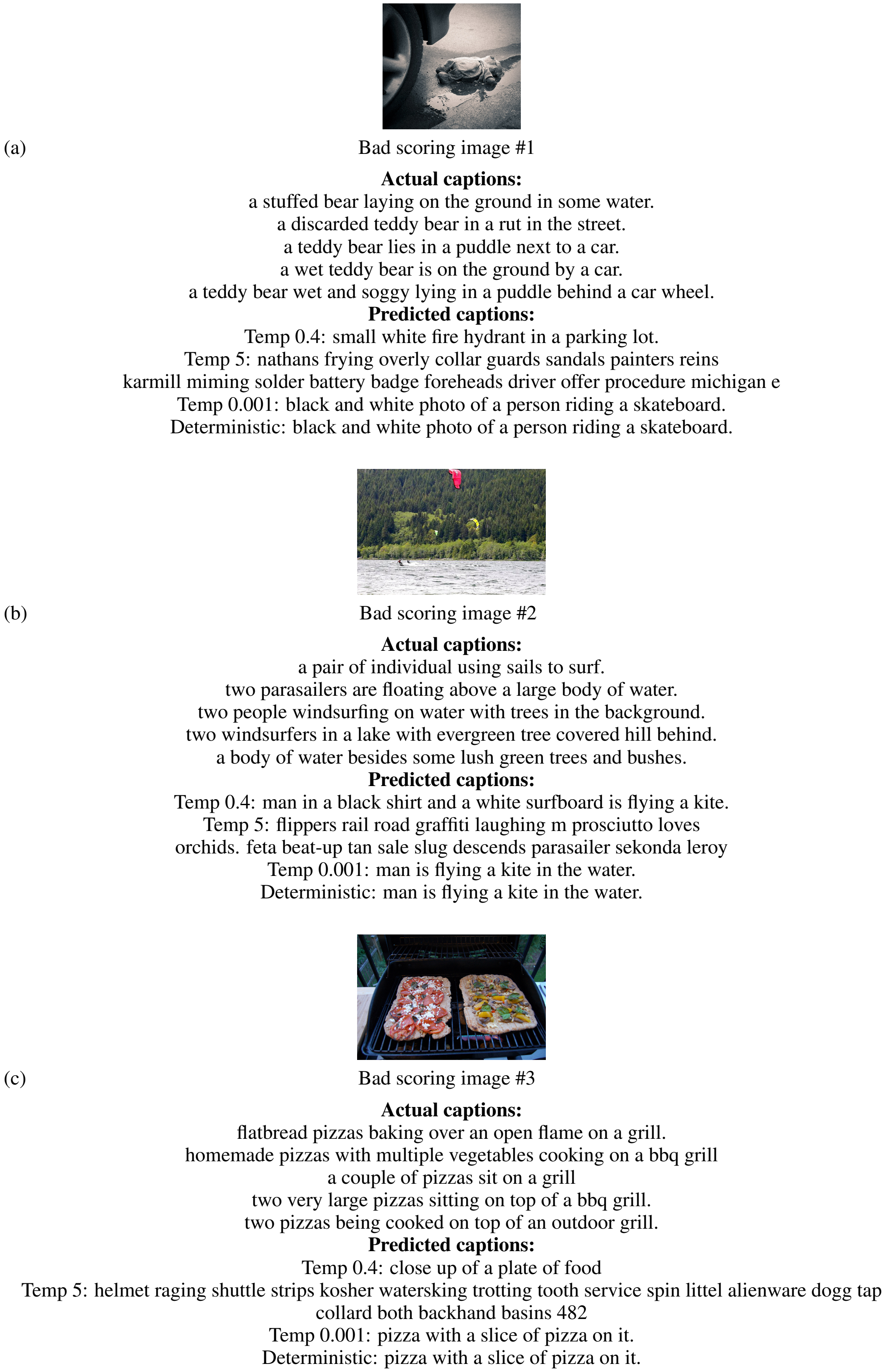
Figure 2 shows images that had bad scoring captions. Overall, the model was able to identify important objects when using temperatures of 0.4, 0.001, and 0. Captions generated with temperature of 5 did not make any sense and contained incomprehensible sequences of words. In two of the three examples, the network is able to identify key objects in the images, but is unable to associate the correct actions with them. For example in the second image, you can see that the model was able to pick up most of what was happening in the image with a few minor errors when the temperature was 0.4, 0.001, and deterministic. It identifies that there is surfing, and some sort of kite-like object. It fails to recognize the connection between the kite and the surfers that is the fact that they are para-

Table 1: Experiment results for LSTM model

LR	Weight decay	Hidden unit size	Embedding Size	Optimizer	LR Scheduler	Temp	BLEU-	
							1	4
5e-3	0	512	300	Adam	none	0.01	43.049	1.966
5e-5	1e-3	512	300	Adam	none	0.01	56.109	2.805
5e-4	1e-3	512	300	Adam	none	0.01	59.729	3.749
5e-4	0	512	300	Adam	none	0	63.159	7.054
5e-4	0	512	300	Adam	none	0.5	59.423	5.715
5e-4	0	512	300	Adam	none	0.1	62.976	6.848
5e-4	0	512	300	Adam	none	0.05	63.305	7.047
5e-4	0	512	300	Adam	none	0.01	63.155	7.068
5e-4	0	512	300	Adam	none	0.005	63.121	7.054
5e-4	0	512	300	Adam	none	0.05	63.172	7.091
5e-4	0	512	300	Adam	none	0.0005	63.134	7.054
5e-4	0	512	300	Adam	none	0.0001	63.159	7.054
5e-4	0	512	300	Adam	none	0.00005	63.163	7.068
5e-4	0	512	300	Adam	none	0.00001	63.159	7.054
5e-4	0	256	150	Adam	none	0.01	62.043	6.012
5e-4	0	256	300	Adam	none	0.01	62.475	6.367
5e-4	0	512	600	Adam	CosineLR	0.1	63.04	7.1883
5e-4	0	512	600	Adam	CosineLR	0.01	63.18	6.91
5e-4	0	512	600	Adam	CosineLR	0.05	62.885	**7.459**
Early stop after 6 epochs								
5e-4	0	1024	300	Adam	none	0.01	61.87	6.59
5e-4	0	512	600	Adam	none	0.01	61.177	6.075
5e-4	0	1024	600	Adam	none	0.01	61.107	5.809
5e-4	0	2048	600	Adam	none	0.01	61.263	6.153
5e-4	0	2048	1200	Adam	none	0.01	60.97	6.19
5e-4	0	4096	1200	Adam	none	0.01	61.92	6.472
Early stop after 8 epochs								
5e-4	0	512	150	Adam	none	0.01	62.791	6.842
Train for 25 epochs								
1e-2	1e-4	512	300	SGD	StepLR	0	55.47	1.96
5e-3	1e-4	512	300	SGD	none	0.1	58.27	3.64
5e-3	1e-4	512	300	SGD	none	0.3	56.64	3.2
5e-3	1e-4	512	300	SGD	none	0	58.95	3.98
Train for 25 epochs, early stop after 14 epochs								
5e-3	1e-4	512	300	SGD	StepLR	0.1	51.36	1.345
1e-2	1e-4	512	300	SGD	StepLR	0.1	54.75	1.91
Add dropout regularization to last fc layer of ResNet								
5e-4	1e-4	512	512	Adam	CosineLR	0.001	**65.66**	6.976

sailing. Oddly enough, the caption generated with a temperature of 5 which makes no grammatical or lexical sense happens to contain the word ”parasailer”. Similarly, the third image is found to have pizza in it, but it is not able to identify that it is being cooked. The model is completely unable to determine much about the first image with any temperature. It identifies a parking lot, fire hydrants, and person on a skateboard when using temperatures of 0.4 and 0.001. None of these are present in the image. The only thing that the model is able to correctly identify in this sample is in the caption generated deterministically which says that the image is black and white.

Figure 3 shows images that have good scoring captions. There was no difference between the deterministic and temperature 0.001 captions. The captions generated with high temperature again seemed to have random sequences of words with a few words matching actual items in the images. The 0.001 temperature and deterministic captions were able to capture most of what was contained



(a) Bad scoring image #1

Actual captions:
a stuffed bear laying on the ground in some water.
a discarded teddy bear in a rut in the street.
a teddy bear lies in a puddle next to a car.
a wet teddy bear is on the ground by a car.
a teddy bear wet and soggy lying in a puddle behind a car wheel.
Predicted captions:
Temp 0.4: small white fire hydrant in a parking lot.
Temp 5: nathans frying overly collar guards sandals painters reins karmill miming solder battery badge foreheads driver offer procedure michigan e
Temp 0.001: black and white photo of a person riding a skateboard.
Deterministic: black and white photo of a person riding a skateboard.

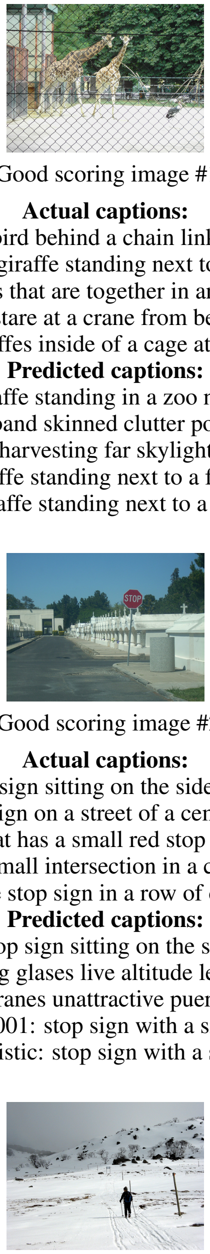
(b) Bad scoring image #2

Actual captions:
a pair of individual using sails to surf.
two parasailers are floating above a large body of water.
two people windsurfing on water with trees in the background.
two windsurfers in a lake with evergreen tree covered hill behind.
a body of water besides some lush green trees and bushes.
Predicted captions:
Temp 0.4: man in a black shirt and a white surfboard is flying a kite.
Temp 5: flippers rail road graffiti laughing m prosciutto loves orchids. feta beat-up tan sale slug descends parasailer sekonda leroy
Temp 0.001: man is flying a kite in the water.
Deterministic: man is flying a kite in the water.

(c) Bad scoring image #3

Actual captions:
flatbread pizzas baking over an open flame on a grill.
homemade pizzas with multiple vegetables cooking on a bbq grill
a couple of pizzas sit on a grill
two very large pizzas sitting on top of a bbq grill.
two pizzas being cooked on top of an outdoor grill.
Predicted captions:
Temp 0.4: close up of a plate of food
Temp 5: helmet raging shuttle strips kosher watersking trotting tooth service spin littel alienware dogg tap collard both backhand basins 482
Temp 0.001: pizza with a slice of pizza on it.
Deterministic: pizza with a slice of pizza on it.

Figure 2: Bad captions generated by baseline LSTM using temp 0.4, then tested on other temps



(a) Good scoring image #1

Actual captions:
giraffes and a bird behind a chain link fence at a zoo
a couple of giraffe standing next to each other.
two giraffes that are together in an enclosure.
two giraffes stare at a crane from behind a fence.
two giraffes inside of a cage at the zoo.
Predicted captions:
Temp 0.4: giraffe standing in a zoo next to a fence.
Temp 5: though headband skinned clutter poster vacation bunkbed
tie-dyed lo clinton agility federal harvesting far skylight shoes row see-through passersby
Temp 0.001: giraffe standing next to a fence and a fence.
Deterministic: giraffe standing next to a fence and a fence.

(b) Good scoring image #2

Actual captions:
a red stop sign sitting on the side of a road.
stop sign on a street of a cemetary
a road that has a small red stop sign by it
lone stop sign at small intersection in a cemetery of castes.
a lone stop sign in a row of crypts
Predicted captions:
Temp 0.4: stop sign sitting on the side of a road.
Temp 5: sop finding glases live altitude lettered heron postal
leaving conditions stage contents cranes unattractive puerto gear sightseers snow-capped say
Temp 0.001: stop sign with a sign on it.
Deterministic: stop sign with a sign on it.

(c) Good scoring image #3

Actual captions:
a person riding across a snow covered field holding ski poles.
person skiing cross country down a trail in the snow.
man standing on skis in a open field of snow.
cross country skier heading away from photographer towards hill.
a skier is alone on a snow mountain
Predicted captions:
Temp 0.4: man standing in a snow covered mountain.
Temp 5: unloading whiskey binders girls crack waxing civilians
drivers kneeing devilish cobbled breeze formed cheers prowesssimilar unappealing 13th kitch
Temp 0.001: person on skis in the snow with a mountain.
Deterministic: person on skis in the snow with a mountain.

Figure 3: Good captions generated by baseline LSTM using temp 0.4, then tested on other temps

in the image. For example, the first image accurately identifies the fence and giraffe in the image, the second one finds the stop sigh, and the third one finds the skis and snow. The grammar is a little bit strange for each of the ones generated deterministically. The giraffe is standing next to a fence ”and a fence” which is strange to mention that there are two fences in this way. The stop sign is also stated to have a sign on it. And in the third image, the person is ”with” a mountain instead of on it. When the captions are generated with 0.4 temperature they seem quite realistic and almost match the actual captions. For example the first image caption mentions that the giraffe is in a zoo and next to a fence. The second image now has the stop sign ”sitting” which is a figure of English speech that is a bit odd to comprehend since inanimate objects don't normally perform actions like sitting. It also knows now that it is sitting on the side of a road. This caption is one word away from one of the actual given captions. The third image is a little less successful compared to the other two and it's caption no longer mentions the skis. Instead the man ”is standing in a snow covered mountain” This sentence is similar to the ones generated deterministically in that it does not use the correct action words to describe the standing. People stand ”on” mountains typically instead of in them or with them.

4.2 Vanilla RNN Model

Best Hyperparameters. The best hyperparameters found for the vanilla RNN model were a learning rate of 0.0005, a temperature of 0.01, 512 hidden units, an embedding size of 600, Adam optimizer, no weight decay, and Cosine Annealing Learning Rate Scheduler.

Plot and Test Loss. The plot for the validation and training loss for the RNN model with the best hyperparameters is shown in figure 4. The RNN model with the best hyperparameters achieved a test loss of 1.455.

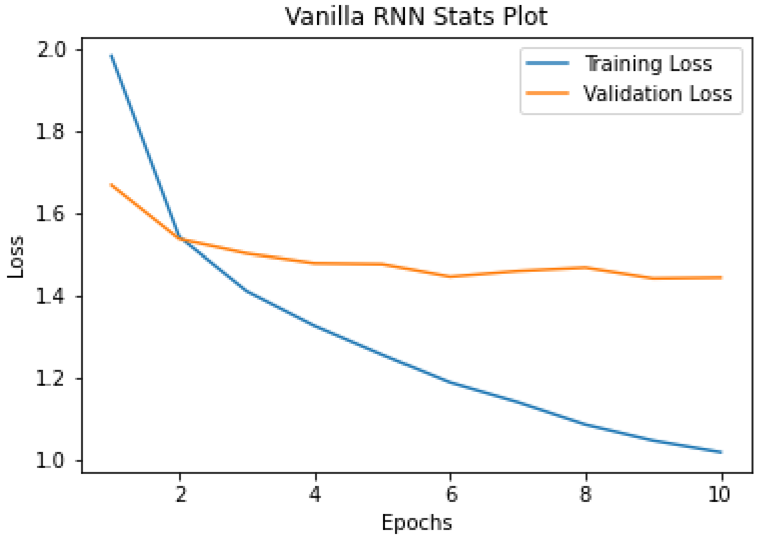
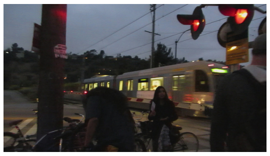


Figure 4: Training and validation losses for RNN best model

From the graph, it can be seen that the validation loss only decreases by a small fraction and then starts to level off. The training loss keeps decreasing, which means that the model has started overfitting. The gap between the curves increases but the model does not early stop since the loss does not rise.

BLEU Scores. The BLEU-1 score for the RNN model with the best hyperparameters on the test set was 63.57%, and the BLEU-4 score was 7.37%. Table 2 shows the results for each experiment conducted.

Using an LR Scheduler did not improve model performance by a large margin. A lower learning rate resulted in a model achieving the best BLEU scores. Changing the temperature had significant impact on model performance. We decided to report our best model as the one which resulted in



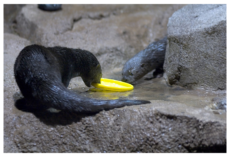
(a) Bad scoring image #1

Actual captions:

a number of people on bikes under a traffic light
a group of people on bicycles next to a passing train.
a train passing some people on bikes at night.
a large long train and a person on a bike.
a group of people sitting at a bus station waiting for a train.

Predicted captions:

Temp 0.4: man is walking down the street with an umbrella.
Temp 5: parasol classes apron communal diamond trains domicile south bayou family satisfied snowsuit shopped knife variations markers button walker they
Temp 0.001: man riding a bike down a street next to a building.
Deterministic: man riding a bike down a street next to a building.



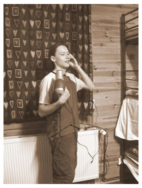
(b) Bad scoring image #2

Actual captions:

an animal biting a yellow frisbee next to another man.
otters investigating a frisbee thrown into their naturalistic enclosure.
two dark colored animals with a yellow plastic disc.
two otters that are playing with a frisbee.
two small otters playing with a yellow frisbee.

Predicted captions:

Temp 0.4: polar bear is walking on a beach
Temp 5: result lids jetway some participating tux towtruck sold huts tossing fourteen microwave barbed chew penguins blender seasonal ill rick
Temp 0.001: small bird is sitting on a rock.
Deterministic: small bird is sitting on a rock.



(c) Bad scoring image #3

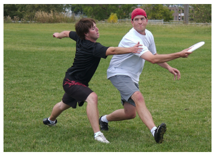
Actual captions:

a young person using a hair dryer near bunk beds.
a black-and-white photo of a woman using a hairdryer.
a person is blow drying their hair in a bedroom
a young man is holding a hair dryer.
person in black and white picture holding hair dryer

Predicted captions:

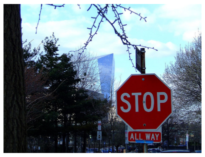
Temp 0.4: couple of people that are standing together
Temp 5: beg cosmetic painted mixed tether cock casting yelling defense tackle bump capture roped firetruck sponge cycles red-haired carrier off
Temp 0.001: man and a woman are standing outside in a room.
Deterministic: man and a woman are standing outside in a room.

Figure 5: Bad captions generated by rnn model using temp 0.4, then tested on other temps



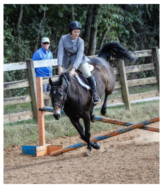
(a) Good scoring image #1

Actual captions:
two men are playing frisbee with one man on the offense and another guarding.
two men outside at a park playing frisbee golf
two men playing frisbee in a field of grass.
a man is grabbing at another man who is in the midst of throwing a frisbee
two young men playing a game of frisbee.
Predicted captions:
Temp 0.4: man is throwing a frisbee in a park.
Temp 5: bat for emissions devie girl receiving playground poised sitting on dancing teenager investigating weaving lined horses controller cemetery rowboats seasoning
Temp 0.001: man in a field throwing a frisbee.
Deterministic: man in a field throwing a frisbee.



(b) Good scoring image #2

Actual captions:
a stop sign stand next to the branches of a tree.
a stop sign is beside some trees with a building in the background.
traffic sign displayed at intersection in large city.
a stop sign with a tall building in the background.
a four way stop sign that is on the corner of an intersection
Predicted captions:
Temp 0.4: stop sign is next to a tall building.
Temp 5: ruins beautiful encounters padded remolded scene lanky ultra interaction venue exit sauteed dimly-lit inlet nuzzle rosters ropes gump could
Temp 0.001: stop sign with a street sign on it.
Deterministic: stop sign with a street sign on it.



(c) Good scoring image #3

Actual captions:
a woman that is sitting on the back of a horse.
a woman jumping a horse over two logs.
a women who is riding a horse that is performing a jump
a woman and her horse jump over an obstacle.
a woman racing a brown horse while a man in a beige hat watches.
Predicted captions:
Temp 0.4: man riding on the back of a brown horse.
Temp 5: turd energetic hay kickflip two-lane notes journal cursive spanish simpson , detour grown watch nostalgic mailing sure put animals
Temp 0.001: person riding a horse in a field.
Deterministic: person riding a horse in a field.

Figure 6: Good captions generated by rnn model using temp 0.4, then tested on other temps

Table 2: Experiment results for rnn model

LR	Hidden unit size	Embedding Size	Optimizer	LR Scheduler	Temp	BLEU-	
						1	4
5e-4	512	300	Adam	None	0.1	62.103	6.408
5e-5	512	300	Adam	None	0.1	61.92	5.209
5e-4	512	300	Adam	None	0.5	58.17	5.3882
5e-4	1024	300	Adam	None	0.0001	59.047	6.3388
5e-4	512	600	Adam	None	0.0001	61.592	6.433
1e-3	512	600	Adam	None	0.0001	58.232	5.274
5e-4	1024	600	SGD	None	0.0001	58.735	5.863
5e-4	512	600	Adam	CosineLR	0.0001	63.197	6.919
1e-3	512	600	Adam	CosineLR	0.001	60.91	6.4311
5e-4	512	600	Adam	CosineLR	0.01	63.565	**7.3361**
5e-4	512	600	Adam	CosineLR	0.5	57.604	4.916
5e-4	512	600	Adam	CosineLR	0.1	**63.73**	7.27

the highest BLEU-4 score since having a greater number of 4-gram sequences match the ground-truth captions means a more robust model that does not just produce words that accidentally are the words that the ground-truth captions have.

Generated Captions with Different Temperatures The captions generated by the vanilla RNN model were generated at various temperature values. A temperature of 5 produces random and incomprehensible results for every image as expected. Captions with a temperature value of 0.4 tend to be slightly more descriptive, using adjectives (such as ”brown horse” for good image #3 and the ”tall building” for good scoring image #2). It is notable that bad scoring image #1 also has a bad BLEU score for the architecture 2 model. This is even more notable because the bad scoring images are the bottom 3 BLEU scores for each model, so this is a shared trait of both models. The image most likely gives bad results for each model because it is hard to label, as the image is dark, slightly blurry, and the bikes take up a small portion of the image. Bad scoring image #2 possibly performed poorly due to its poor labeling, as some of the captions labeled the otters as an ”animal” and a ”man”. Bad scoring image #3 might have performed poorly due to the pose of the person in the picture, as standing slightly to the side might imply that there is another person in the photo, as both 0.4 temperature, 0.001 temperature, and deterministic generate captions stating that there are two people in the picture.

One possible explanation for the deterministic approach not describing the image well is that when the deterministic softmax picks the word at each time step that it is most confident about, it tends to focus on the most obvious aspect of the picture. Compared to temperature values of 0.4, the captions generated using deterministic often do not include more complex features of the image such as descriptive adjectives and parts of the background. A good example of this is the second good scoring image from figure 6. In this example, the caption generated with a temperature of 0.4 mentions the building in the background alongside the stop sign in the foreground, while the deterministic caption mentions the sign in the foreground twice. Very high temperature values like values greater than 5 lead to very poor results because high temperature values mean that words are picked from the set of all possible words from what is essentially a uniform distribution, which is why the words from captions with a temperature of 5 look like they were randomly generated.

4.3 LSTM Model with Image Encodings at each time step

Best Hyperparameters. For Architecture 2, which consists of an LSTM decoder and has image encodings passed in at each time step along with the word embeddings, the best set of hyperparameters found are a Learning rate of 0.0005, a weight decay of 0, 512 hidden units for the LSTM, an embedding size of 150 for the captions and an embedding size of 300 for the images, Adam a the optimizer, no LR Scheduler, and a temperature of 0.005 when stochastically generating captions.

Plot and Test Loss. Figure 7 shows the plot for training and validation losses for our best model.

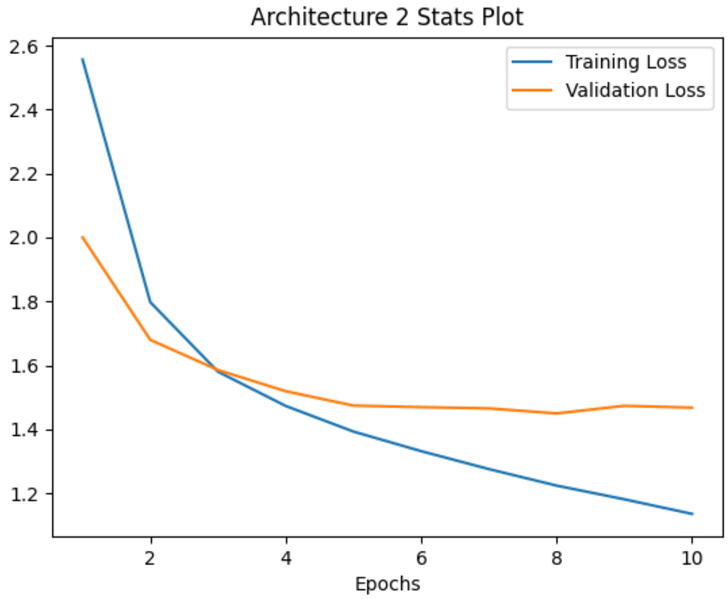


Figure 7: Training and validation losses for architecture 2 best model

Analyzing the graph, both training and validation curves are decreasing. The validation curve starts to flatten out at around 5 epochs, with the gap between the training and validation curves starting to increase. We implement early stopping with 5 patience epochs, and keep track of the best model across all epochs. This helps mitigate the problem of overfitting to some extent.

On the test data, we achieve a cross entropy loss of 1.614.

BLEU Scores. On the test data, we achieve a BLEU-1 score of 66.079 and a BLEU-4 score of 8.106. Table 3 reports BLEU scores for all experiments conducted.

Table 3: Experiment results for architecture 2

LR	Weight decay	Hidden unit size	Embedding size		Optimizer	LR Scheduler	Temp	BLEU-	
			Cap	Img				1	4
5e-4	0	512	300	300	Adam	None	0.001	65.263	7.9
5e-4	0	512	150	150	Adam	None	0.001	64.903	7.878
5e-4	0	1024	300	300	Adam	None	0.001	63.312	6.666
5e-4	0	512	150	300	Adam	None	0.001	66.041	**8.089**
5e-4	0	512	300	150	Adam	None	0.001	64.671	7.196
5e-4	1e-4	512	150	300	Adam	None	0.001	65.771	6.755
5e-4	1e-4	512	150	300	SGD	None	0.001	56.15	1.601
1e-3	0	512	150	300	Adam	StepLR	0.001	**66.744**	7.605
1e-3	0	512	150	300	Adam	CosineLR	0.001	65.874	8.057

From the above table, it is clear that the model without an LR Scheduler is able to achieve the highest BLEU-4 score across all experiments, whereas the model using a Step LR Scheduler with step size of 3 achieves the highest BLEU-1 score for a temperature of 0.001 when stochastically generating captions.

To analyze which model among the two performs better on test data, we experiment with different temperatures, with results recorded in Table 4.

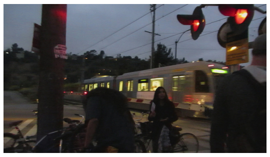
Table 4: Temperature changes when generating captions for architecture 2

Temperature	**BLEU-1 (%)**	**BLEU-4 (%)**
	With no LR Scheduler	
Deterministic	66.054	8.097
0.5	61.253	5.919
0.1	65.755	7.892
0.05	66.031	7.993
0.01	66.05	8.096
0.005	**66.079**	**8.106**
0.001	66.041	8.089
0.0005	66.073	8.097
0.0001	66.049	8.093
0.00005	66.051	8.097
0.00001	66.051	8.097
	With Step LR Scheduler	
Deterministic	66.722	7.615
0.5	61.375	5.584
0.1	66.51	7.687
0.05	66.71	7.592
0.01	**66.77**	**7.641**
0.005	66.705	7.576
0.001	66.744	7.605
0.0005	66.742	7.615
0.0001	66.729	7.615
0.00005	66.725	7.615
0.00001	66.722	7.615

A temperature 0.005 produces the highest BLEU-4 score for the model with no LR Scheduler, while a temperature of 0.01 generates the highest BLEU-1 score for the model with a Step LR Scheduler across all experiments. We choose to declare the model with the highest BLEU-4 score as the best model since having a higher score when 4-gram sequences match indicates that the model is more robust. Higher BLEU-1 scores can be scored even by a badly tuned model with a high temperature if it produces 1-gram sequences that exist in the actual captions.

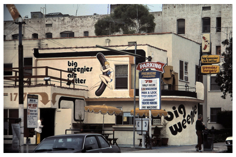
Generated Captions with Different Temperatures. Figures 8 and 9 provide images and their associated captions generated by our best model using the deterministic approach, and the stochastic approach with temperatures 5, 0.4, and 0.001.

For the bad captions in Figure 8, the model is either unable to identify key subjects within the image or fails to describe what the image actually contains. A very high temperature like 5 just outputs incomprehensible words that do not describe the image at all. A slightly lower temperature of 0.4 produces comprehensible sentences. However, it fails to identify the main events being portrayed in the images. For example, the first badly captioned image has several red hues and people waiting at a red traffic light, and the model assumes it is a *man in a red vest*. Similarly for the other two images, the model gets a general idea of what the image is about, but lacks in focusing on the details like what the sign reads in the second image. A low temperature of 0.001 is able to identify actions and events within images more correctly like *man riding a bike down the street* in the first image and *train traveling down the tracks* in the third image. It also identifies some background objects like the bicycle in the first image, the shop in the second image, and the forest in the third image. Deterministically generating captions produces the exact same captions as a temperature of 0.001 does in stochastic generation.



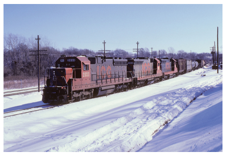
(a) Bad scoring image #1

Actual captions:
a number of people on bikes under a traffic light
a group of people on bicycles next to a passing train.
a train passing some people on bikes at night.
a large long train and a person on a bike.
a group of people sitting at a bus station waiting for a train.
Predicted captions:
Temp 0.4: man in a red vest is holding a red umbrella.
Temp 5: great banans mothers flight hitching puddle bank turkey safe sandwiches trolleys bomber text weeds traditional staffuniversity vendor couple
Temp 0.001: man riding a bike down a street.
Deterministic: man riding a bike down a street.



(b) Bad scoring image #2

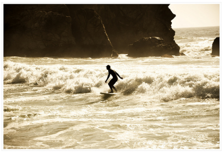
Actual captions:
in building has a picture of a woman in a bikini hugging a hot dog.
the building in the city has a lot of signs on it.
an image of a street sign with a corner restaurant
a restaurant with a number of signs for attention.
there is a restaurant that is called big weenies on the street
Predicted captions:
Temp 0.4: sign that reads “ reptile ” ”.
Temp 5: motorbike winner via upright kiwi ass head wedding gather take electricity crisscrossed also vessels riderless quickpetting plaster form
Temp 0.001: sign for a noodle shop with a sign on it.
Deterministic: sign for a noodle shop with a sign on it.



(c) Bad scoring image #3

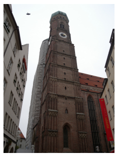
Actual captions:
a long red train traveling through snow covered country side.
three red and black train engines and its cars and snow
a train is coming down the tracks on a snowy day
a train drives down its tracks next to a snow bank
a train in a rural area covered with heavy snow
Predicted captions:
Temp 0.4: red train pulling into a train station
Temp 5: alert suite storefronts moves kitchens spring birdge slightly pitchers wade incredible produce mustangs clutching locomotive vacant stating paper plymouth
Temp 0.001: train traveling down tracks next to a forest.
Deterministic: train traveling down tracks next to a forest.

Figure 8: Bad captions generated by architecture 2 using temp 0.4, then tested on other temps



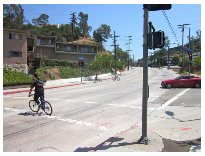
(a) Good scoring image #1

Actual captions:
a person on white surfboard riding a wave next to a cliff.
a man riding a wave on top of a surfboard.
a person is surfing in a on a wave
silhouette of a surfer catching a wave in the distance
the person is in the water having fun
Predicted captions:
Temp 0.4: man on a surfboard riding a wave.
Temp 5: focused mingle slop condition so server wet just sail ibm magazine call formed red announcer blocking sanitizers improve rolls
Temp 0.001: man riding a wave on top of a surfboard.
Deterministic: man riding a wave on top of a surfboard.



(b) Good scoring image #2

Actual captions:
a very tall building has a clock on the front of it.
a bird flying near a clock tower in a city.
a large church steeple with a clock on it
a very tall clock tower next to other tall buildings.
a large tall brick tower has a clock on top.
Predicted captions:
Temp 0.4: large tower with a clock on the top.
Temp 5: busy kites outdoors sleeved waiting enjoying bottles obstacle pennsylvania events towell array dripping restored elaborate kites force asian dixie
Temp 0.001: large clock tower with a clock on the top.
Deterministic: large clock tower with a clock on the top.



(c) Good scoring image #3

Actual captions:
car waits as person on bike crosses the road
a four lane street in the suburbs
a bicyclist crossing the street as a car waits to turn.
a man with a backpack riding a bicycle by a traffic light.
a full view of a suburban city with people.
Predicted captions:
Temp 0.4: man riding a bike on a street.
Temp 5: skater herds cross amenities wrap suffed muffs removal so brighton shoved toward wildebeast self-portrait crooks features police volkswagon nose
Temp 0.001: man riding a bike down a street.
Deterministic: man riding a bike down a street.

Figure 9: Good captions generated by architecture 2 using temp 0.4, then tested on other temps

For the good captions in Figure 9, temperatures 0.4, 0.001, and deterministic caption generation methods all produce captions that are very similar and correctly identify the contents of the image. A lower temperature of 0.001 does include some extra details like a *man being on "top" of the surfboard* in the first image, a *large "clock" tower* in the second image, and *man riding a bike "down" a street* instead of "on" a street. A very high temperature of 5 again produces incomprehensible captions with words completely unrelated to the content of the images.

5 Discussion

Below is a discussion of what worked and did not work with out models and why that might have happened.

5.1 Baseline LSTM Model

Optimizer. We found that using Adam as our optimizer provided the highest BLEU Score overall. In our experiments we found that when using stochastic gradient descent with momentum, the loss value after each epoch would be dropping at a constant rate until the last epoch. This indicated that we were not using a large enough learning rate and possibly needed to train for more epochs. After using a much larger learning rate and training for 25 epochs, we were able to achieve a higher performance, but still did not achieve a BLEU score close to models trained using Adam. Although it may be possible to achieve a much higher BLEU score, we decided to pursue using Adam when testing other hyperparameters due to it getting better performance in fewer epochs.

Learning Rate. After trying both a higher and lower learning rate of both 5e-4 and 5e-5 with the Adam optimizer, we found that the BLEU 1 score for the higher learning rate was over 10 points lower and the score for the lower learning rate was about 3 points lower. The higher learning rate performing much worse indicated that there was possible overfitting happening.

Embedding and Hidden Size. We tried lots of different embedding sizes for our baseline LSTM model but the results were, in general, similar to what we got with the default embedding size of 300. There was hardly any improvement without taking performing regularization to improve generalization. This was because as the embedding size increased, the model was unable to pay attention to the larger and more rich input that it was receiving. We tried increasing the number of hidden units to deal with this problem, but this resulted in the model overfitting to training data with the model early stopping after only 6-8 epochs of training. This meant that we had to use regularization methods like dropout and an LR Scheduler.

Dropout. Using dropout significantly improves model performance. Before introducing dropout, the model overfit to training data with a large gap between the training and validation curves. Dropout regularization ensured that the model did not start memorizing mappings between the training images and captions and this led to better generalization to test data.

5.2 Vanilla RNN Model

Performance vs Baseline LSTM Model. While the baseline LSTM model performed better than the RNN model, it was a surprisingly small difference, as the best LSTM model differs from the best RNN model by about 2%. This could be because the model generally did not have many vanishing or exploding gradients. Since LSTM models are able to overcome this issue, there would be a greater LSTM performance across the board if the RNN model had vanishing or exploding gradients. Perhaps if the model had more hidden layers or was trained to generate a longer sequence of words for the caption, or generally had more parameters, there would be a higher chance of vanishing or exploding gradients and the LSTM model would have a significantly higher performance than the vanilla RNN model. Overall, the hyperparameters that performed well for the vanilla RNN model most likely performed for the same reasons that they performed well for the other models.

Temperature. Sometimes, a higher temperature performed better than using a lower temperature during caption generation. This can be seen from the model with a temperature of 0.1 resulting in the maximum BLEU-1 score and a decent BLEU-4 score. One possible reason for this could be that as the temperature increases, the randomness with which the model samples from the multinomial distribution increases. Because we only performed a single trail for all experiments, it might have been the case that a higher temperature like 0.1 produced words that were semantically closer to the ground-truth caption words, being a random chance of event. This can also be seen from a temperature of 0.5 performing very poorly. In general, lower temperatures were more robust and produced consistently good BLEU scores.

5.3 LSTM Model with Image Encodings at each time Step LR

Optimizer. Analyzing the graphs using Adam and Stochastic Gradient Descent (SGD) as our optimizer, Adam starts with a much lower training and validation loss from the first epoch. SGD, however, starts with a very high training and validation loss and training is slow. Though Adam produces training and validation curves with gap between them increasing as we train for more epochs (indicating overfitting), SGD produces curves that almost overlap. We decide to go with Adam because it converges faster and the loss that it produces in the first epoch is the loss that SGD produces after training for 10 epochs.

Learning Rate and LR Scheduler. In order to overcome the overfitting that Adam produces, we use an LR Scheduler as a way to introduce regularization and improve generalization to unseen data. The gap between training and validation curves is smaller and the model performs better on test data, resulting in a higher BLEU score. We start with a higher learning rate when using an LR scheduler so as to prevent weights from not updating and the model stopping training without converging.

Embedding Size. Having the embedding size the same for both captions and images leads to poor performance on test data. This suggests that there is a wight difference between how visual and semantic information is used as input by the decoder. When using a smaller embedding size for the captions than for the images, the model performs much better. The BLEU score is also greater than what we see for the baseline LSTM model. This suggests that the decoder performs better when it sees the images at every time step in order to keep track of what is being attended to when generating captions. Also, a richer visual information seems more important than a richer semantic information.

5.4 Caption Generation

In general, smaller temperatures produced a more accurately performing model than larger temperatures. Surprisingly, the deterministic approach worked really well, with BLEU scores matching the BLEU scores of the best temperature from the stochastic caption generation approach. While the captions from the deterministic approach did not describe the image perfectly, it did tend to produce good BLEU scores when averaged over the entire test dataset. This could be because the stochastic approach when randomly sampling from the multinomial distribution of word probabilities picks a word that has the highest score, which is what the deterministic approach does when it chooses the maximum scoring word from out of the word probabilities. This results in the deterministic approach working comparable to the stochastic approach for small temperatures like 0.001. This is further reinforced by looking at the captions generated by temperature of 0.001 and the deterministic approach, both of which are identical in almost all of images we tested the, on.

Authors Contributions

Ryan worked on experimenting with different optimizers and temperatures for the baseline LSTM network. He worked on programming the baseline and architecture 2 models as a group and implemented early stopping as well as temperature sampling of the output words. For the report, focused mainly on the sections regarding the baseline LSTM model.

Henry worked on running experiments primarily with the vanilla RNN model, but also ran a few experiments for the baseline LSTM model. He worked on the baseline LSTM model and architecture 2 and implemented the vanilla RNN model. For the report, he mostly worked on the discussion and posting of results related to the vanilla RNN model and captions.

Yash worked on the experiments for Architecture 2 and the code for caption generation. He group programmed with Ryan and Henry on the implementation of the baseline and architecture 2 models, and also did debugging. For the report, he worked on the experiments and results for Architecture 2 under the Models, Results, and Discussion sections.

References

[1] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997.

[3] Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. 1(2):270–280, jun 1989.

[4] Pytorch documentation.

[5] Garrison W. Cottrell. *CSE 151B: Deep Learning*, chapter Lecture 7: Modeling Sequences (Recurrent Networks). 2022.

[6] Garrison W. Cottrell. *CSE 151B: Deep Learning*, chapter Lecture 8: Generative Modeling with RNNs. 2022.

[7] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator, 2015.

[8] Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4608–4616, Red Hook, NY, USA, 2016. Curran Associates Inc.