Image Captioning Using Traditional LSTM & RNN Methods

Joseph Warmus
University of California, San Diego
jwarmus@ucsd.edu

Eric He
University of California, San Diego
zuhe@ucsd.edu

Brydon Brancart
University of California, San Diego
bbrancar@ucsd.edu

Yigit Korkmaz
University of California, San Diego
ykorkmaz@ucsd.edu

Marcus Schaller
University of California, San Diego
mschalle@ucsd.edu

Abstract

Using Neural Networks to generate captions for an image is one of the most interesting applications of deep learning. This paper will discuss using RNN (recurrent neural networks) in conjunction with LSTM (Long short-term memory) to generate captions for COCO data-set. In order to quantify the effectiveness of the model the average Bleu score of the test set will be calculate. After training a number of models using different parameters, we found that we were able to achieve a Bleu score of 64.9 with our best trained models.

1 Introduction

Image Captioning has been an important and interesting problem in deep learning as it connects both natural language processing and computer vision. Some of its applications include monitoring social media posts and assistance for visually impaired. In this paper, we implemented encoder-decoder architecture to generate captions for image data. For encoder, we used a pretrained convolutional network ResNet50 and trained recurrent neural network as our decoder through ”Teacher Forcing”. The goal of this paper is to compare the performance between vanilla RNN and LSTM using different hyperparameter settings, as well as evaluating different strategies for text generation.

2 Related Work

There has been various approaches to solve image captioning problem. In the earlier attempts, Andrej Karpathy creates a method that resembles how humans describe the scenes. In his work, a CNN is first used to extract features about the objects and combined these with embedding of sentences to find the alignments between the segments of the sentences and corresponding areas in the image. After that, to generate captions, an RNN structure is used calculate hidden states, which will be used in generating captions [10]. On the other hand, a group of researchers in Google, tried to solve this problem by using a LSTM network rather than a plain RNN structure. Although both of these works approaches in extracting features from a given image, Google makes use of LSTM layers to generate captions. The feature vector representing image is inserted into LSTM sequence only once in the beginning, the inputs at the following states are the word vectors. Also, in their work, they used a beam search method to generate best captions [13].

Researchers in the Microsoft proposes an architecture inspired by machine-translation's encoder-decoder framework architecture. In that architecture, a deep Resnet based model is used for image feature extraction, but unlike other approaches, an entity recognition is used to detect if there is a celebrity or landmark in the given image. If detected, that is also fed into the language model along with the image vector. Their another improvement is that they implemented a logistic classifier to estimate confidence score which will be useful for better handling of the images that are difficult to describe [12].

Apart from these, there has been attempts to combine the attention in generating caption, and visualize this during this process to understand where and what the attention focused on [14]. Furthermore, when should the image be introduced to the model is a considerable question in this problem. Although most of the approaches uses a injecting architecture, where the image is fed into the RNN/LSTM that processes words, there has been some works using the merging architecture, merging the image with the output of RNN/LSTM which processes only words this time. It has been shown that both of these architectures have some superiority over the other one in different use cases [11].

3 Methods

3.1 Data Pre-processing

For the image captioning task, COCO (Common Objects in Context) dataset will be used. However, we will be using a small part of it during training and testing because the original dataset is quite large. In this dataset, there may exist more than one caption for a given image. Thus, a padding is applied to all of the reference captions to make them equal in size. This is a mandatory step for proper training. Also, a conversion from words to ids is applied before padding, which assigns a unique id to each of the words in the vocabulary of the all captions. The images in the dataset are normalized and resized to 256x256 before passing them to the model, and some data augmentations are tried for different experiments, which will be explained in their respective section.

3.2 Network Architectures

In this work, the network composes of 2 main parts with an embedding layer connecting them, namely, encoder and decoder. The encoder part is essentially a pretrained CNN with its last layer being replaced by a linear layer to match output dimension of the CNN with the embedding size. We preferred to use ResNet50, which is a 50 layer deep CNN pretrained with the images from the ImageNet dataset. After replacing the last layer with a linear layer, final output of the encoder is equal to the embedding size (300 by default). Finally, one dimensional batch normalization is applied to the output to make network more stable and fast, by reducing internal covariate shift. Before the decoder, captions are turned into embeddings via an embedding layer, which will be explained in detail below, and concatenated with the output of the encoder. This step is done to enable learning by teacher forcing, which will also be explained in a separate section below. For the decoder, 2 main versions are tested, LSTM and RNN with different hyper-parameters.

3.2.1 LSTM

Consisting of internal recurrently connected memory blocks, LSTM layers overcome the main problem RNNs have, short term memory. Recurrent Neural Networks cannot carry information from much earlier time steps, if the sequence is long. This is due to vanishing gradient problem where the gradient shrinks when back propagating through time, resulting in no contribution to learning. On the other hand, LSTMs have a gated structure which enables them to remove or add information to their cell state, where the main flow of information occurs. An example of this gated structure for the basic LSTM can be found in the figure below. The sigmoid function has a critical role in this gated structure. Since it outputs a value between 0 and 1, it acts like a deciding factor when the old state should be kept or removed, and a new information should be added or not. First of these gates, forget gate decides what amount of information from previous states should be thrown away by combining previous hidden state and current input, and feeding them in to a sigmoid. Second gate is called input gate. The main mission of this gate is to decide what new information needs to be stored in the cell state. This is done by using sigmoid and tanh functions together with combination of previous hidden state and the current input. Tanh here is used to squash values between -1 and 1 so that the network is regulated. Finally, the output gate is used to determine what will be produced as output. This output will be a filtered version of the cell state by feeding it to a tanh function and then multiplying with concatenated previous hidden state and current input. The result will be the next hidden state. Note that hidden states are the main elements that carry information about the past and they are also used in generating predictions. Basically, during the training procedure, the weights used in multiplying the concatenated hidden-input before feeding them into sigmoid and tanh functions are learned [2, 4, 6].

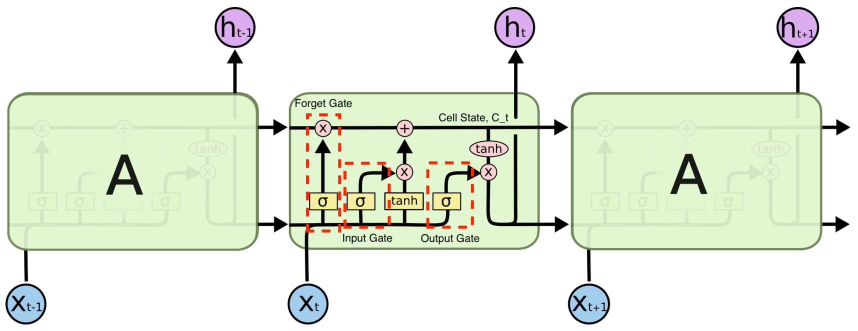


Figure 1: LSTM Unit Internal Structure [6]

Forget gate operate as follows;

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{1}$$

Input gate operate as follows;

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4}$$

Finally, output gate operate as follows;

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t * \tanh(C_t) \tag{6}$$

In our basic LSTM network, there are 2 LSTM layers, the second LSTM takes in the output of the first layer. The input size is initially 300, which is the same as embedding size and the hidden size is initially 512. Finally, after getting the output from the second LSTM, a linear layer is used to reduce the dimension to vocabulary size and to generate the output encoded words.

3.2.2 RNN

Recurrent neural networks process a sequence of vectors one by one, however, while processing the next vector, they also use the previous hidden state to utilize some information about the past. This can either be expressed as a loop between output and input or a chain of RNN units connected with their previous units' hidden states. Both of these expressions can be found in the figure below.

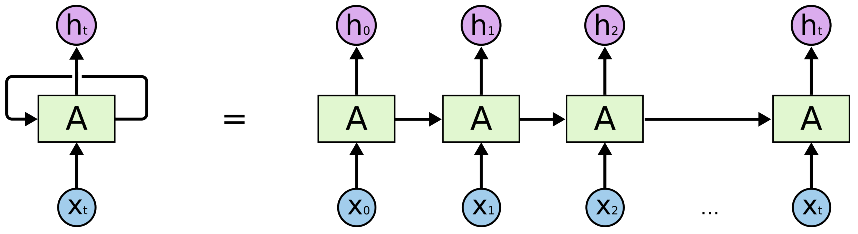


Figure 2: RNN Structure (Rolled / Unrolled) [6]

While calculating the hidden state for the current unit, the hidden state from previous unit is concatenated with the input and passed into tanh function. This step keeps the values flowing through network between -1 and 1, helping network regularization [2, 6, 5]. In our vanilla RNN, a hidden size of 512 and an input size of 300 are used initially. Also, we used 2 layers of RNN where the output of first one is given to the second one as input. Finally, a linear layer is used to reduce the dimension to the vocabulary size, generating encoded versions of the words.

3.3 Word Embedding

In order to apply teacher forcing during training, the captions must be fed into the network with the images as well. To this extend, the words should be represented as unique vectors. This is where embedding comes into play. One can think of embedding as a lookup table that stores dense vector representations of the words in the vocabulary. Note that, these dense representations can be learned as well. In other words, embedding layer takes the caption as input, which is basically a series of indices of words in the vocabulary, converts them to their one-hot encoded versions, and then feeds them into a linear layer with learnable weights to get outputs vectors with a shape of embedding-size. There are 2 main advantages of this method over using directly one-hot encoded versions of the words. The first advantage is the size. If you consider a large vocabulary, using a sparse representation, such as one-hot encoding can result in thousands or millions of dimensions, while using a dense vector representation can result in tens or hundreds of dimensions. The second advantage is that we can represent semantic similarity up to some point using dense representations. However, this cannot be done by using sparse representations, all words are treated as independent elements in the sparse representation [9, 8, 3].

3.4 Training and Teacher Forcing

In training process, the captions are fed along with images to generate word embeddings. Furthermore, since the captions vary in length, they are padded to a fixed length. In the forward pass, teacher forcing is used, that is using 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 done by converting images into a vector with the same size with embeddings by using CNN and linear layers, concatenated with word embeddings, and then they are fed into the LSTM/RNN. Initial hidden and cell states are initiated as zeros for the first LSTM/RNN layer. Finally, the hidden state of second LSTM layer is given as input to linear layer to generate one-hot encoded predictions.

The loss function is selected as cross entropy loss. During calculation, predictions are used with the targets. However, since the captions are converted into a packed sequence after getting their embeddings (to make inner calculations more optimized), the predictions are outputted as a packed sequence as well. Thus, targets are also converted into a packed sequence before calculating the loss. Also it is important to mention that teacher forcing method is used while calculating loss for validation and test sets as well.

3.5 Testing and Sample Generation

As mentioned above, teacher forcing is used while calculating the loss for test set. However, in order to generate samples, a different method is needed since this time we need the whole generated caption per image. To do this, the network's prediction should be the next input to network, rather than the next word in the training set. Two different methods are used to determine the prediction, deterministic and stochastic.

In both of these methods, the image is fed into the pre-trained CNN and features are extracted. These features are given as input to linear layer to get a vector of embedding dimension, and normalised using batch normalization. Then, unlike training, this vector is iteratively fed into the LSTM/RNN layers with maximum loop duration representing the possible maximum length caption. The initial hidden and cell states are None in this case, but they change every iteration. Furthermore, in each iteration, hidden state is given as input to last linear layer to get encoded output.

However, two generation methods differ in generating prediction(sample) from that output. In the first deterministic method, the sample is generated by finding the index corresponding to maximum element of the output vector. However, this method can get stuck in repetitive loops, since the result will be more or less the same every time. In the second method, we first take the softmax with temperature of the outputs.

$$y_i = \frac{exp(\frac{a_i}{T})}{\sum_j exp(\frac{a_j}{T})} \tag{7}$$

The temperature term, T in this equation decides how stochastic the process is. The lower the temperature, the more deterministic the prediction, with 0 temperature meaning completely deterministic process [1]. After taking the softmax of the output vector, the prediction is generated by creating a multinominal distribution using the softmax probabilities and drawing one sample from it. The sample represents the index of word predicted. This ensures that the generated caption will not be same for different trials. Note that both of these methods are repeated for each iteration, each hidden state, with every predicted word stored in a predicted caption. Also, considering the loop goes for the maximum possible caption length, there exists predictions for pads, start and end of the sentence in the predicted caption. Thus before calculating the BLEU scores, these predicted 'words' are removed from the generated caption.

BLEU-1 and BLEU-4 scores used to evaluate the similarity between generated and reference captions. BLEU score is a metric to evaluate machine-translated text [7]. It is generally between 0 and 1 (or 0 and 100). The higher the score, the more successful the generated text. The calculation of BLEU score is as follows:

$$BLEU = \min(1, exp(1 - \frac{reference\_length}{output\_length})) * (\prod_{i=1}^{4} precision_i)^{\frac{1}{4}} \tag{8}$$

Notice that this formula composes of two main parts, brevity penalty (min term), which penalizes the generated captions that are too short compared to the reference captions, and n-gram overlap (product term), which measures the similarity between unigrams, bigrams, trigrams, and four-grams of generated captions with their respective counterparts in reference captions. In the original version, the BLEU score is calculated using equal weights for up to 4-grams, in our project however we consider only the unigram (BLEU-1) and 4-gram (BLEU-4) by adjusting the weights.

4 Experiments and Results

Besides the following experiments, we also experimented with adding data augmentation to our input image. However, it was found that this reduced the overall Bleu score compared to the same model that was trained without data augmentation. Additionally, it should be noted that we weren't aware that we were meant to remove the punctuation from our test dataset. This is likely the reason for a lower Bleu score than the problem statement mentions.

4.1 Experimenting with Hidden Layer Dimensions

To determine the best hidden layer dimension for the two hidden LSTM node layers, we experimented with sizes of 128, 256, 512, 724, 1024, and 2048. The embedding dimension was held constant at 300.

Table 1: Hidden Layer Experimentation
Using AdamW, size 300 embedding, lr of 5e-4, 15 epochs, and no augmentation

Hidden Layer Size	Min Val Loss	Test Loss	Bleu1	Bleu4
128	2.5603	2.5448	64.09	8.401
256	2.4713	2.4718	64.17	8.736
512	2.4444	2.4317	64.58	8.711
724	2.4230	2.4149	64.52	8.896
1024	2.4159	2.4080	63.34	8.664
2048	2.3777	2.3747	63.24	7.745

The result of our experimentation seemed to be that 512 and 724 were the best performing hidden layer sizes. We ended up going with 512 because of its slightly higher bleu1 score, though 724 had higher bleu4 and slightly lower test loss.

4.2 Experimenting with Embedding Dimensions

In order to determine the best embedding dimension for caption generation, we experimented with embedding of sizes 200, 300, 400, 450, and 500. These experiments were performed by training networks with these embedding dimensions and two hidden layers of 512 LSTM nodes.

Table 2: Embedding Experimentation
Using AdamW, two 512-node Embedding layers, lr of 5e-4, 10 epochs, and no augmentation

Embedding Size	Min Val Loss	Test Loss	Bleu1	Bleu4
200	2.4487	2.4374	64.74	8.838
300	2.4444	2.4317	64.58	8.711
400	2.4306	2.4230	64.48	8.525
450	2.4343	2.4270	64.41	8.589
500	2.4346	2.4260	64.95	8.219

Through experimentation we determined that an embedding layer of 500. This resulted in the lowest test loss and highest Bleu1 score, although it also resulted in a lowered Bleu4 score.

4.3 LSTM vs RNN

We trained LSTM and RNN with different settings of hyper-parameters and reported the results of the best models below. We also tested both discrete and stochastic sampling techniques which will be explained in later sections. From our experiments, we found that generally LSTM converges faster and outperforms RNN with the same hyper-parameter settings.

As we can see from the plots, the training and validation loss for RNN oscillates at around 7th epoch and the model quickly overfits. On the other hand, LSTM has a much smoother trajectory and lower loss. In additon, we observed that loss increases as we increase the embedding size for RNN, whereas LSTM gives lower loss as embedding increases. We believe the LSTM mechanism helps to learn and generalize to larger word embedding, whereas RNN suffers from learning from longer sequences and is also more affected by the vanishing gradient problem.

We tested the two best models on the test sets and obtained 2.426 cross entropy loss for LSTM and 2.581 for the Vanilla RNN model.

Table 3: Best model for LSTM and RNN

Model type	Learn Rate	Optimizer	Hidden	Embedding	Generation	Test Loss
LSTM	5.00E-04	Adam (weight decay = 0.0001)	512	500	Deterministic	2.426
RNN	5.00E-04	Adam (weight decay = 0.0001)	512	300	Deterministic	2.581

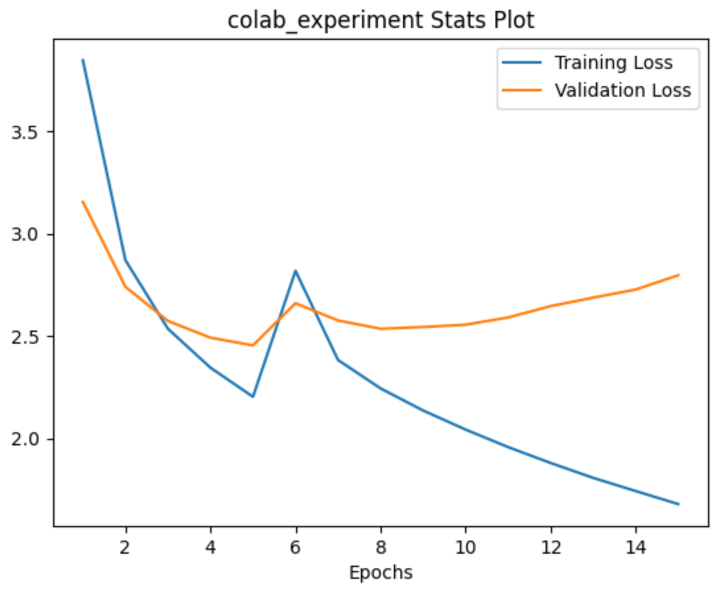


Figure 3: training and validation loss vs number of epochs for vanilla RNN

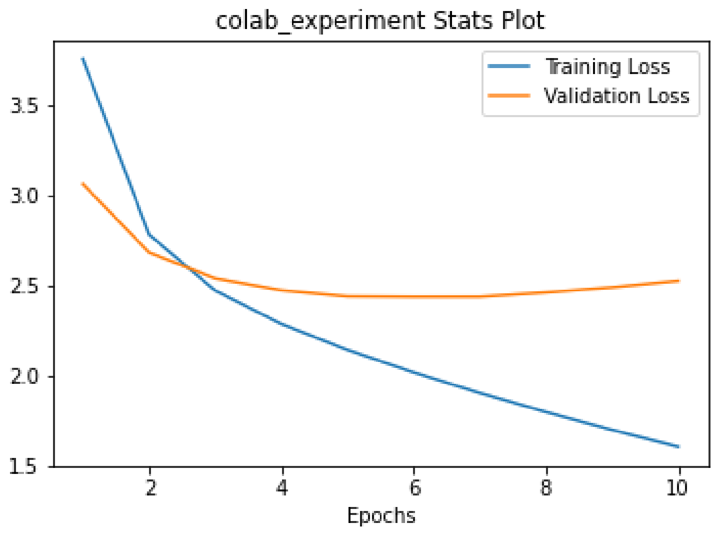


Figure 4: training and validation loss vs number of epochs for best LSTM

4.4 Experimenting with generation criteria

4.4.1 Stochastic vs Deterministic Generation

Two different criteria that can be used when generating captions are Stochastic generation and deterministic generation. Stochastic generation produces a probability distribution over each of the classes and allows the model to select words that are not necessarily the highest probability. Deterministic will always select the word that has the highest probability. We compared these two selection criteria on our best performing LSTM and RNN models. The table below will describe the hyper parameters used and the BLEU score that was generated for each of them.

Table 4: Best models test on Deterministic vs Stochastic

Model type	Hidden	Embedding	Bleu1	Bleu4	Generation
LSTM	512	500	64.978	8.447	Deterministic
LSTM	512	500	64.954	8.219	Stochastic (temp = .1)
RNN	512	300	62.616	8.175	Deterministic
RNN	512	300	62.6313	8.0975	Stochastic (temp = .1)

Interestingly we found the deterministic generation to perform better than stochastic. This is likely because our temp of 0.1 was too high resulting in the model selecting incorrect words.

4.4.2 Effects of Temperature on Generation

The temperature parameter changes the distribution of the model across all of the classes. The higher the temperature the more diversity and higher probability that the model will select a word with lower probability. We found that a lower temperature was indeed better with a temp of 0.1 working the best.

Table 5: Temperature tested on LSTM RNN model with an embedding size of 500 and hidden size of 512

Temp	**test loss**	**bleu1**	**bleu4**
0.1	2.426	64.954	8.219
0.2	2.426	64.489	8.191
0.7	2.425	56.574	4.899
1	2.425	44.897	2.734
1.5	2.425	20.052	1.204
2	2.426	7.539	0.612

5 Discussion

In this study, we implemented a image captioning network using ResNet50 as encoder and compared the performance of Vanilla RNN and LSTM for decoder. We found that LSTM outperformed RNN in almost all performance metrics when trained using the same hyper-parameters. In addition, we found for LSTM network, bigger embedding size and hidden size tend to give better cross entropy loss performance. However, low loss does not always translate to better text generation accuracy. In addition, for Vanilla RNN, we found that large hidden layer size led to poorer results with model over-fitting very quickly whereas the LSTM mechanism seems to overcome this problem. We also found interestingly that deterministic generation produced better results than the stochastic approach for both RNN and LSTM. One possible explanation could be inappropriate temperature setting.

After testing various hyper-parameters for training and text generation, the best performing model is a LSTM network with 512 hidden dimension, 500 embedding size, and deterministic generation mechanism. This model achieved test loss 2.426, Bleu1 64.978 and Bleu4 8.447. It outperformed the baseline model by about 0.01 in validation loss and 0.37 for Bleu1 score.

References

[1] How to sample from language models. https://towardsdatascience.com/how-to-sample-from-language-models-682bceb97277. Accessed: 2022-03-01.

[2] Illustrated guide to lstm's and gru's: A step by step explanation. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21. Accessed: 2022-03-01.

[3] Pytorch: Embedding. https://pytorch.org/docs/stable/generated/torch.nn.Embedding.html. Accessed: 2022-03-01.

[4] Pytorch: Lstm. https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html#torch.nn.LSTM. Accessed: 2022-03-01.

[5] Pytorch: Rnn. https://pytorch.org/docs/stable/generated/torch.nn.RNN.html. Accessed: 2022-03-01.

[6] Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2022-03-01.

[7] Understanding the bleu score. https://cloud.google.com/translate/automl/docs/evaluate. Accessed: 2022-03-01.

[8] What are word embeddings for text? https://machinelearningmastery.com/what-are-word-embeddings/. Accessed: 2022-03-01.

[9] Word embeddings: Encoding lexical semantics. https://pytorch.org/tutorials/beginner/nlp/word_embeddings_tutorial.html. Accessed: 2022-03-01.

[10] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions, 2015.

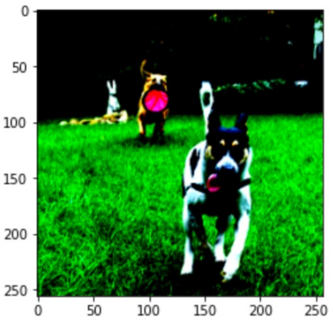
[11] M. TANTI, A. GATT, and K. P. CAMILLERI. Where to put the image in an image caption generator. *Natural Language Engineering*, 24(3):467–489, Apr 2018.

[12] K. Tran, X. He, L. Zhang, J. Sun, C. Carapcea, C. Thrasher, C. Buehler, and C. Sienkiewicz. Rich image captioning in the wild, 2016.

[13] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator, 2015.

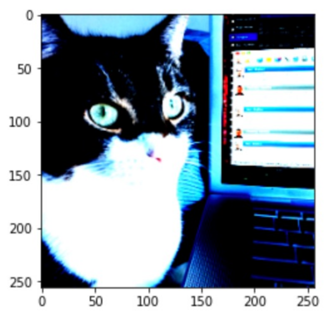
[14] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention, 2016.

APENDIX Caption Predictions: Good Predictions



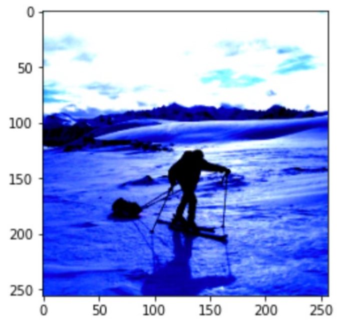
Generated
a dog is playing with a frisbee in the grass

Ground Truth
two dogs are playing with a frisbee in the grass
two dogs are running together and one of them has a frisbee
a brown dog with a toy chasing another dog
some dogs are being trained to play with other dogs
two dogs running through the grass in a spacious yard



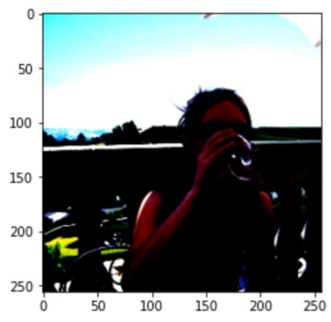
Generated
a cat is sitting on a desk with a laptop

Ground Truth
a cat sitting next to a laptop computer on a table
a grey and white cat standing besides a laptop
a cat sitting next to a laptop
the cat is sitting beside the computer looking at something
the cat is setting next to the laptop



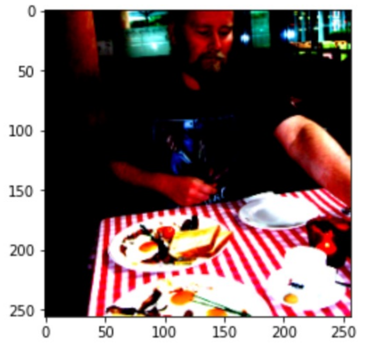
Generated
a man on skis is standing in the snow .

Ground Truth
a person riding skis on a flat snowy area
a person is walking on skies in a snow covered mountain
a man riding skis across snow covered ground while holding two ski poles
a person trekking across the snow dragging a bag
a person using cross country skis pulling a bundle behind them



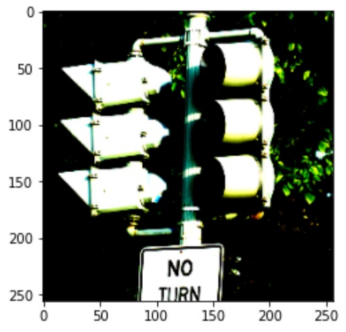
Generated
a woman is holding a glass of wine

Ground Truth
a lady is sitting on a terrace drinking a glass of wine
woman enjoying beverage from glass while sitting on terrace
a woman drinking a glass of wine in an outdoor setting
a girl in sunglasses and a black shirt holds a glass up to her mouth in an outdoor area
a woman drinking from a class on a balcony



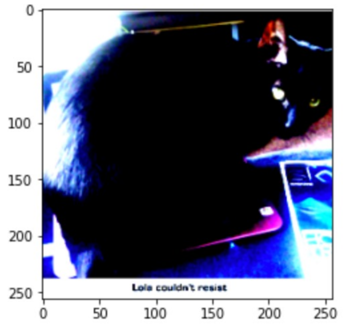
Generated
a man is sitting at a table with a plate of food .

Ground Truth
a man sitting at a table with two plates of breakfast food
a male with two plates of breakfast food on the table in front of him
a man in black shirt at a table with plate of food
a man sitting at a table with a bunch of breakfast food on a plate
a bearded man prepares to eat his breakfast



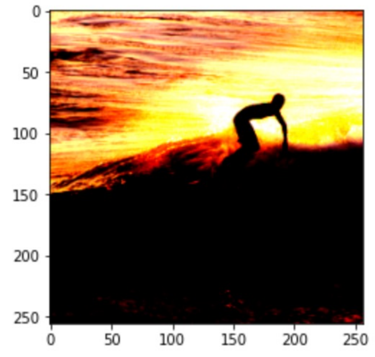
Generated
a traffic light is shown with a street sign .

Ground Truth
traffic signals and sign on pole at roadway intersection
a picture of a sign on a traffic light in front of trees
a traffic light with a no turn sign on it
a traffic light with a no turn on red sign under
a set of two street lights with a no turn on red sign



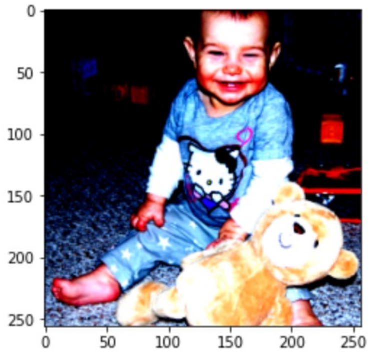
Generated
a cat is sitting on a laptop computer .

Ground Truth
a black cat sitting on top of a desk
a black cat sitting on a dining table
a black cat sits perched on top of a laptop
a large plate cat sitting on a little laptop
a black cat sits on a small pink laptop next to a dvd



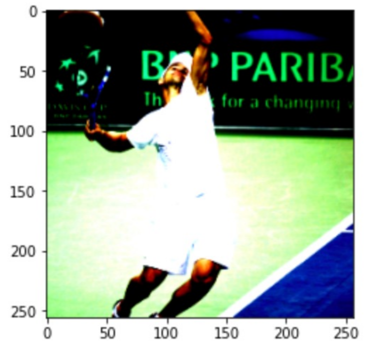
Generated
a man riding a surfboard on a wave

Ground Truth
a man on a surfboard in the water
two surfers are out in the ocean with surfboards
the silhouette of a man rides a wave on a white surf board
the man is riding the waves on his surfboard in the water
a surfer is riding a small wave in the ocean



Generated
a little boy that is holding a teddy bear

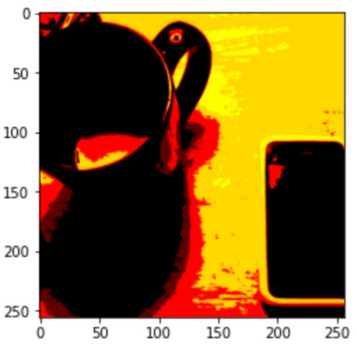
Ground Truth
a baby and a teddy bear in a playpen
small baby in a hello kitty sweater smiling for the camera
a baby is smiling while holding a teddy bear
a smiling baby sitting and holding a stuffed animal
a small child in a gray shirt and a teddy bear



Generated
a tennis player is getting ready to hit the ball

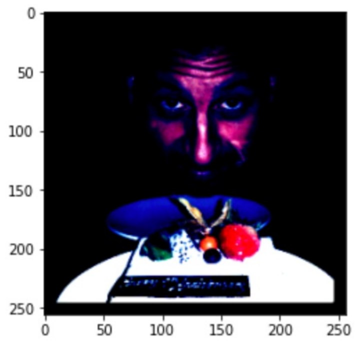
Ground Truth
a man on the tennis court is about to use his racket
tennis player looks as though he is beginning his serve
a tennis player with red shoes serves the ball
a man swinging his racket while playing tennis
a man serving while he plays a match of tennis

Caption Generation: Bad Predictions



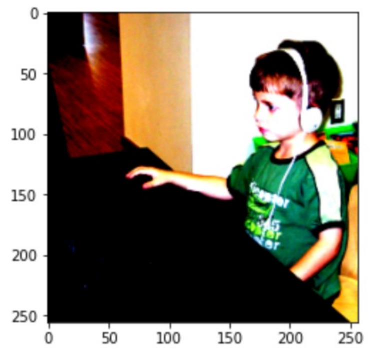
Generated
a pair of scissors are on a table

Ground Truth
a picture of a coffee cup and a smart phone on a table
a cup is sitting on a surface next to a cell phone
a filtered photo of a phone and coffee mug is shown
a cup sitting next to a smart phone
a very fuzzy image of a cell phone and a cup



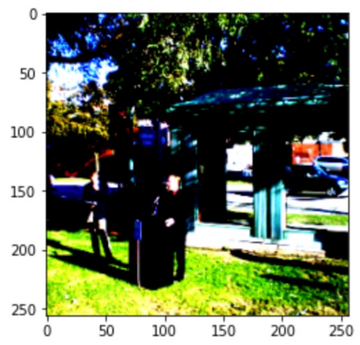
Generated
a woman holding a doughnut in her hand

Ground Truth
a man peers over a small plate behind a napkin lined with pieces of fruit
a man with beard leaning over a plate and a cake
a man posing behind a plate of food
a man poses for a happy anniversary photo
a man has his chin on a blue plate near some paper and utensils



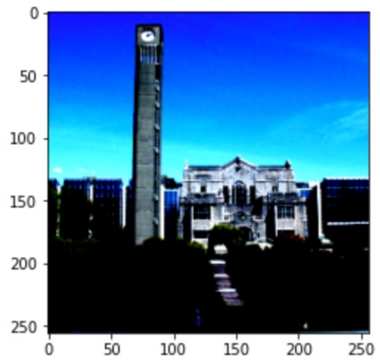
Generated
a woman sitting at a table with a laptop computer

Ground Truth
a kid sitting at a computer wearing headphones
a boy is wearing is white headphones at a black computer
a boy is sitting at a desk using a computer
a young boy wearing headphones using a desktop computer
a view of a child with headphones on interacting with a computer



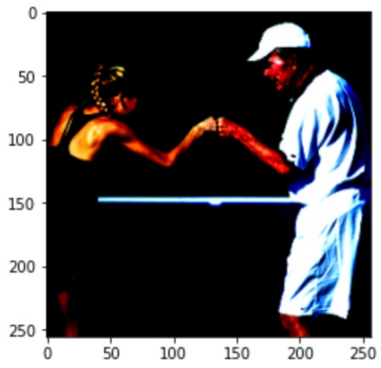
Generated
a group of people sitting on a bench next to a building

Ground Truth
man speaking from lectern at outdoor gathering in urban area
a man in a black suit talking at a podium outside
a man is speaking at a podium in a park
a man at a podium while a man a woman listen to him
a man is at a podium and giving a speech



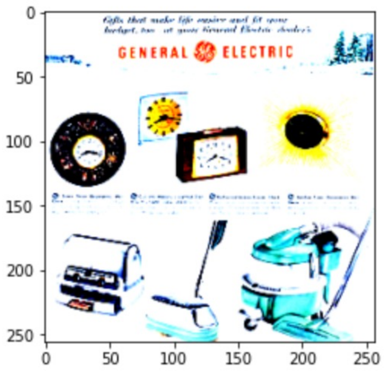
Generated
a large flock of birds flying through the sky

Ground Truth
a picture of buildings with a clock tower in front
a architecture picture of a clock tower and the surrounding buildings
a grey clock tower above grassy area and building
a municipal building with a tall clock tower framed against a blue sky
a tall clock tower in front of a group of buildings in a park like area



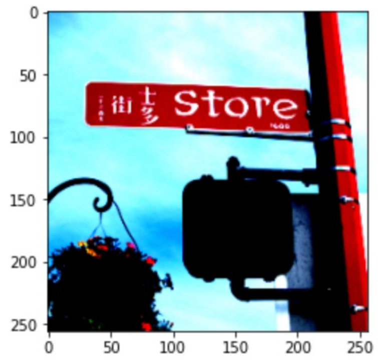
Generated
a tennis player is getting ready to hit the ball

Ground Truth
a woman hitting her coaches fist with her first on the court
a blond woman tennis player in black and a male in white professional tennis player greeting her coach with knuckles
a man and woman give each other a fist bump before a tennis game
an old man fist bumps a young woman



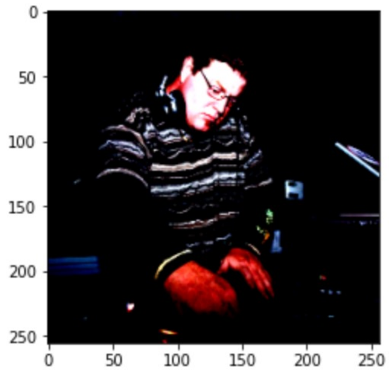
Generated
a display of different types of cakes on a table

Ground Truth
an advertisement from a christmas wonderland store with several ge products
a page in a general electrics catalog advertising christmas ideas
electronics advertised on a paper for christmas
an ad for general electric with clocks and appliances on it
a various items such clocks and kitchen appliances are advertized



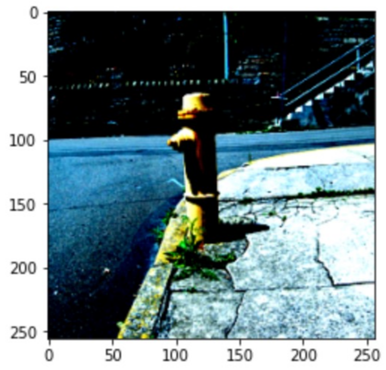
Generated
a stop sign that has been defaced

Ground Truth
a red and white street sign mounted on a red pole with a pedestrian traffic light
a street sign is printed in both english and foreign languages
a traffice sign is shown with asian writing
a red road sign appears to be shut down below a store sign
a stop sign that is very red in the street



Generated
a man in a suit and tie is sitting on a table

Ground Truth
a man is playing records in a dj booth
a dj works with his equipment for the party
a dj with headphones around his neck spinning a record
a disc jockey is working with his record player
a dj with earphones around his neck is using the equipment



Generated
a man is standing on a sidewalk with a bike

Ground Truth
a yellow fire hydrant on the corner of a street
a yellow fire hydrant is sitting on a broken side walk
a yellow fire hydrant next to a street corner
a yellow fire hydrant is sitting on the corner across from the park
a yellow fire hydrant is on the corner of an old sidewalk