# IMAGE CAPTIONING AND NEURAL MACHINE TRANSLATION

**Taniya Rajani** [*1] **Tarun Kateja** [*1]

## Abstract

The task of Image Captioning and Neural Machine Translation has attracted many researchers due to their exciting applications in many fields such as virtual assistants, editing tools, image indexing and translation. Image Captioning integrates the two major artificial intelligence fields: computer vision and natural language processing. The models studied for both Image Captioning and Machine Translation belong to the same family of encoder-decoder models. There has been great research in this fascinating area and in this project, we have explored different deep neural network architectures to generate descriptive captions for an image and also for neural machine translation which is the task of converting source sentence in one language into another language. At last, we have integrated Image Captioning with Machine Translation to predict the description of an image in English and then translate the English caption to Hindi Language. Models have been evaluated using the state-of-the-art BLEU score and through human evaluations.

## 1. Introduction

Generating descriptive sentence for an image is a very challenging task. It requires expertise in both image processing to extract relevant features from images as well as natural language processing to incorporate textual information in a model and generate text for a given image. There has been good research and scientists have proposed various ways to generate descriptions for images using different architectures of neural networks. Image captioning model encodes the input using an encoder to generates sequences into a fixed length vector and a decoder to decode it, word by word, into a sequence. We are using Convolutional Neural Networks (CNN) to encode an image because there are good CNN models trained on images and some have been trained to classify images in thousand categories. Thus, these pre-trained models capture the essence of an image. Utilizing features from a pre-trained models into a new model is a technique called Transfer Learning where we leverage the features from a model like CNN which is trained for a specific task and utilize those while training a new model like to generate a sequence for an image. Thus, we need not to train the CNNs from scratch. We experimented with three different model architectures. Figure 1 depicts an end-to-end neural network consisting of a vision CNN followed by a language generating RNN. It generates complete sentences in natural language from an input image. In the first model, we used pre-trained VGG16 network model's weighted layers to extract features from images as the encoder and features from encoder are fed to LSTM network. LSTM is trained to predict each word of the sentence after it has seen the image (I) as well as all preceding words (s) as defined by $p(S_t|I, S_0, ...., S_{t\_1})$. In the second model, we replaced VGG16 network model with ResNet18 as the encoder and kept the rest architecture same as first model. We also wanted to explore attention mechanism for image captioning which is widespread in deep learning and has very promising results. Through attention mechanism the model can choose only those parts of the encoding that it thinks is relevant to the task at hand. In image captioning, we consider some pixels more important than others. This mechanism is employed in the third model architecture, here we used Encoder as pre-trained ResNet101 network model and extracted its last three layers to extract features from images and in decoder, weighted encoded images part (attention) and text are fed to the LSTM network. Figure 2 illustrates the visual attention mechanism for image captioning.

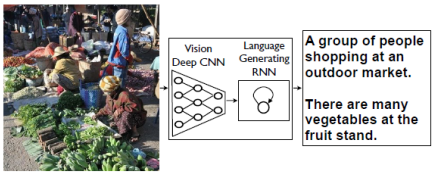


*Figure 1.* An example of Image Captioning

In the second part of this project, we have worked on Neural Machine Translation (NMT) which is relatively newer approach of statistical machine translation in the spirit of deep

[*]Equal contribution [1]University of Illinois at Chicago, Illinois, USA. Correspondence to: Tarun Kateja <tkatej2@uic.edu>.

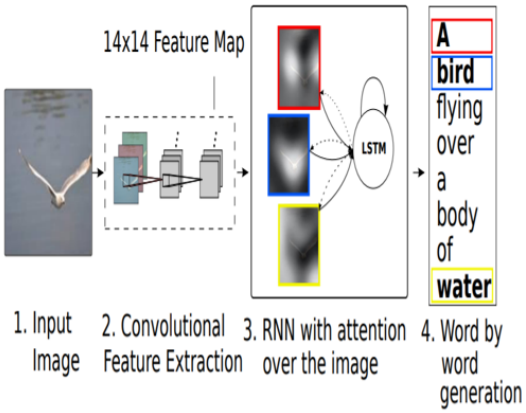


*Figure 2.* Attention in Image Captioning

representational learning. The task of machine translation is not always straightforward. It needs to encode all the information from a text in one language and convert it to a text with the contextual meaning and semantically correct text in another language. NMT consists of encoders and decoders. The encoder neural network captures the input information and convert the source sentence into a fixed-length vector and the decoder neural network provides a translation from the encoded vector. This is a sequence to sequence task and Recurrent Neural Networks (RNNs) are designed to take sequences of text as inputs or return sequences of text as outputs, or both. They're called recurrent because the network's hidden layers have a loop in which the output and cell state from each time step become inputs at the next time step. This recurrence serves as a form of memory. It allows contextual information to flow through the network so that relevant outputs from previous time steps can be applied to network operations at the current time step. Convolutional Neural Networks cannot perform such task. Thus, we have used RNNs as decoder and encoder both for machine translation. Below is an illustration of NMT with an RNN based encoder-decoder architecture. Figure 3 shows a NMT as a stacking recurrent architecture for translating a source sequence ABCD to a target sequence XYZ. Here, <eos> marks the end of a sentence

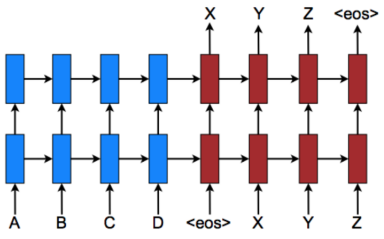


*Figure 3.* Neural Machine Translation

For NMT, we explored two different architectures to translate text from English language to a text in Hindi Language. In the first model, the encoder of a seq2seq network is GRU to extract the information from source sentence and the decoder is another GRU with Attention that generates the text in target language at each timestamp using encoder information and previous timestamp prediction. Through attention, model automatically learn to detect more important words in a sequence while generating the corresponding words in the output sequence. In the second model, we replaced GRU cell with a LSTM cell and increased the complexity by keeping model bidirectional and multilayer.

## 2. Related Work

The main inspiration of our work comes from recent advancement in deep learning and specifically in image classification, object detection and machine translation. In computer vision and natural language processing, these areas have been explored with sequence to sequence training with neural network and many use recurrent neural network architectures for generating and encoding a sequence. But when it comes to images, CNNs are the first choice for extracting information from images. It is like translating an image to a sequence of words.

For image captioning task, we are inspired from architectures presented in the paper(Vinyals et al., 2015) from Google, where they have used a pretrained model on ImageNet data to avoid overfitting and extract the features from images which they pass it to decoder for further processing. This interesting work was improved by (Xu et al., 2015) introducing an attention based model that automatically learns to describe the content in the image.

For machine translation, we referred to the work (Bahdanau et al., 2014), which concluded that using fixed length is a bottleneck in improving the performance of the basic encoder–decoder architecture, and propose to extend this by allowing a model to automatically (soft-) search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly.

## 3. Image Captioning Models

### 3.1. Model 1 - VGG16 as Encoder + LSTM as Decoder

To generate a descriptive caption we use a neural and probabilistic framework in our models. This model as shown in figure 4, encodes a image into fixed dimensional vector and uses this representation to decode a desired output. The encoder of our interest is CNN and the decoder is state-of-art RNN. We want to maximize the probability of the correct description given the image by using the following

formulation:

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{(I,S)} logp(S|I;\theta) \tag{1}$$

where $\theta$ are the parameters of our model, I is an image, and S is correct transcription. Since S represents any sentence, its length is unbounded. Thus, it is common to apply the chain rule to model the joint probability over $S_0, ..., S_N$, where N is the length of this particular example as

$$logp(S|I) = \sum_{t=0}^{N} logp(S_t|I, S_0, .....S_{t-1}) \tag{2}$$

where we dropped the dependency on $\theta$ for convenience. In training, (S, I) is a training example, and we optimize the sum of the log probabilities as described in (2). The model is trained using the stochastic gradient descent.

It is common to model $p(S_t|I, S_0, ..., S_{t-1})$ with a RNN, where the variable number of words we condition upon up to t$-$1 is expressed by a fixed length hidden state $h_t$. This memory is updated after seeing a new input xt by using a non-linear function f:

$$h_{t+1} = f(h_t, x_t) \tag{3}$$

For f, we use Long-Short Term Memory (LSTM) net, which has shown state-of-the art performance on sequence tasks such as translation. The choice is natural because of its performance on many similar tasks and ability to deal with vanishing and exploding gradient problems. The core of a LSTM is its memory cell and its behaviour is controlled by "gates"- layers which are applied multiplicative and thus can either keep a value from the gated layer either 1 or 0. Please refer to figure 5, for architecture of a LSTM cell. In particular, three gates are being used which control whether to forget the current cell value (forget gate f), if it should read its input (input gate i) and whether to output the new cell value (output gate o). The definition of the gates and cell update and output are as follows:

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1}) \tag{4}$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1}) \tag{5}$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1}) \tag{6}$$

$$c_t = f_t \odot C_{t-1} + i_t \odot h(W_{cx}x_t + W_{cm}m_{t-1}) \tag{7}$$

$$m_t = o_t \odot c_t \tag{8}$$

$$p_{t+1} = Softmax(m_t) \tag{9}$$

where $\odot$ represents the product with a gate value, and the various W matrices are trained parameters. The nonlinearities are sigmoid $\sigma\odot$ and hyperbolic tangent $h\odot$. The last equation $m_t$ is what is used to feed to a Softmax, which will produce a probability distribution $p_t$ over all words.

The model is trained to predict each word of the sentence at time t after it has seen the image and all previous words. In more detail, if we denote the input image by I and true sentence by $S = (S_0, ..., S_N)$, the unrolling procedure is as follows:

$$x_{-1} = CNN(I) \tag{10}$$

$$x_t = W_e S_t, \quad t \in (0....N-1) \tag{11}$$

$$p_{t+1} = LSTM(x_t), \quad t \in (0....N-1) \tag{12}$$

Here, we encode each word as a one-hot vector $s_t$ of dimension equal to the size of the dictionary. Please note that we represent a special start word with $s_0$ a special stop word with $s_N$ which designates the start and end of the sentence. Using the stop word the LSTM understands that a complete sentence has been generated. Both the image and the words are mapped to the same space, the image by using a vision CNN, the words by using word embedding. The image I is only input once, at t = 1, to inform the LSTM about the image contents.

Our loss is the sum of the negative log likelihood of the correct word at each step as follows:

$$L(I, S) = -\sum_{t=1}^{N} logp_t(S_t) \tag{13}$$

The above loss is minimized w.r.t. all the parameters of the LSTM, the top layer of the image embedder CNN and word embeddings. We used greedy search to generate a sentence given an image. Greedy search is a straight-forward way, we transform the output of the decoder using a linear layer and sample the word with highest score at each time until we generate the end token.

### 3.2. Model 2 - ResNet18 as Encoder + LSTM as Decoder

This model is similar to that of model 1 described above with a small change in encoder model. Instead of using VGG16 pretrained model for encoding the image, here we are using ResNet18 as our CNN. For this model as well, we used greedy search to generate a sentence given an input image.

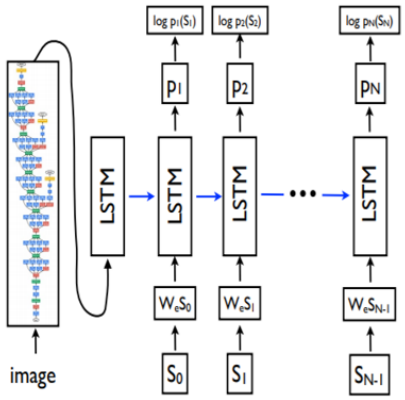


*Figure 4.* Model 1 Architecture

### 3.3. Model 3 - ResNet101 + LSTM with Visual Attention

In this model we supported our decoder module with visual attention over images to help our decoder look at different parts of images. In a typical setting with attention, model's gaze shift across the image. The same mechanism employed here can be used in any model where the Encoder's output has multiple points in space or time.

As mentioned in section 3.1, our model will generate sentence S of length t say from vocabulary of size K

$$s = (s_1 ....s_t), \quad s_i \in R^K \tag{14}$$

where t is the length of the caption. We use a CNN in order to extract a set of feature vectors referred as annotation vectors(Xu et al., 2015). The network produces L vectors, each with D dimensions corresponding to a part of the image.

$$a = (a_1 ....a_L), \quad a_i \in R^D \tag{15}$$

In order to obtain a correspondence between the feature vectors and portions of the 2-D image, we extract features from a lower convolutional layer unlike in model 1 which instead used a fully connected layer (we removed last 3 layers). For decoder, we used LSTM network as explained in section 3.1. The framework is also similar to that explained for model 1 in section 3.1. Using $T_{s,t} : R^s \rightarrow R^t$ to denote a simple affine transformation with parameters that are learned.

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix}$$

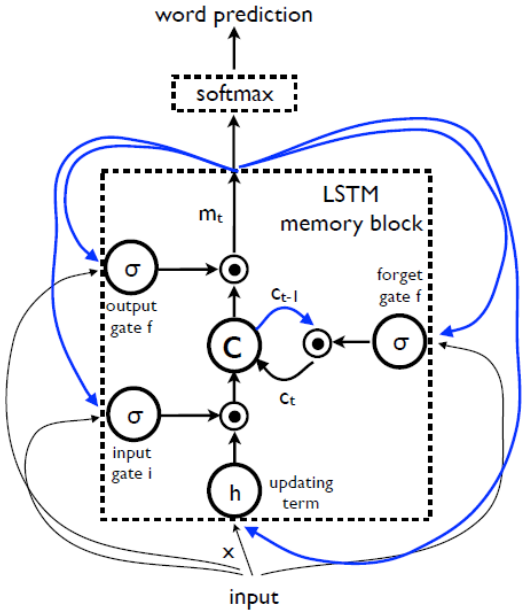


*Figure 5.* Long and Short-term Memory Networks

$$c_t = f_t \odot c_{t1} + i_t \odot g_t \tag{16}$$

$$h_t = o_t \odot tanh(c_t) \tag{17}$$

Here, $i_t, f_t, c_t, o_t, h_t$ are the input, forget, memory, output and hidden state of the LSTM, respectively. The vector $z \in R^D$ is the context vector, capturing the visual information associated with a particular input location, as explained below. $E \in R^{m*K}$ is an embedding matrix. Let m and n denote the embedding and LSTM dimensionality respectively and $\sigma$ and $\odot$ be the logistic sigmoid activation and element-wise multiplication respectively.

As explained in the paper (Xu et al., 2015), the context vector $\hat{z}_t$ is a dynamic representation of the relevant part of the image input at time t. We used mechanism $\phi$ that computes $\hat{z}_t$ from the annotation vectors $a_i, i = 1, ..., L$ corresponding to the features extracted at different image locations. For each location i, the process generates a positive weight i which can be interpreted either as the probability that location i is the right place to focus for producing the next word, or as the relative importance to give to location i in blending the $a_i$ together. The weight of each annotation vector $a_i$ is computed by an attention model $f_{att}$ for which we use a multilayer perceptron conditioned on the previous hidden state $h_{t1}$. For emphasis, we note that the hidden state varies as the output RNN advances in its output sequence: "where" the network looks next depends on the sequence of words that has already been generated.

$$e_{ti} = f_{att}(a_i, h_{t-1}) \tag{18}$$

$$\alpha_{ti} = \frac{exp(e_{ti})}{\sum_{k-1}^{L} exp(e_{tk})} \tag{19}$$

Once the weights (which sum to one) are computed, the context vector $\hat{z}_t$ is computed by

$$\hat{z}_t = \phi(a_i \alpha_i) \tag{20}$$

where $\phi$ is a function that returns a single vector given the set of annotation vectors and their corresponding weights.

The initial memory state and hidden state of the LSTM are predicted by an average of the annotation vectors fed through two separate MLPs (init,c and init,h):

$$c_o = f_{init,c}(\frac{1}{L}\sum_{i}^{L} a_i) \tag{21}$$

$$h_o = f_{init,h}(\frac{1}{L}\sum_{i}^{L} a_i) \tag{22}$$

The output word probability given the LSTM state, the context vector and the previous word is given by:

$$p(y_t|a, y_1^{t1}) \propto exp(L_o(Ey_{t1} + L_h h_t + L_z \hat{z}_t)) \tag{23}$$

We used beam search to generate a sentence given an image. This is where you don't let your decoder be lazy and simply choose the words with the best score at each decode-step. Beam Search is useful for any language modeling problem because it finds the most optimal sequence. We use a linear layer to transform the decoder's output into a score for each word in the vocabulary. It would be best if we could somehow not decide until we've finished decoding completely, and choose the sequence that has the highest overall score from a basket of candidate sequences.

## 4. Neural Machine Translation Models

### 4.1. Model 1 - GRU as Encoder and Decoder

The encoder of a this seq2seq network is gated recurrent unit (GRU) to extract the information from source sentence. The decoder is another GRU with Attention that generates the text. Max length of a sentence is kept 15. A gated recurrent unit (GRU) was proposed by (Cho et al., 2014) to make each recurrent unit to adaptively capture dependencies of different time scales. Similarly to the LSTM unit, the GRU has gating units that modulate the flow of information inside the unit, however, without having a separate memory cells. The activation $h_t^j$ of the GRU at time t is a linear interpolation between the previous activation h $h_{t-1}^j$ and the candidate activation ${h_t}^{\sim j}$ :

$$h_t^j = (1 - z_t^j){h_{t-1}}^j + z_t^j {h_t}^{\sim j} \tag{24}$$

where an update gate $z_t^j$ decides how much the unit updates its activation, or content. The update gate is computed by:

$$z_t^j = \sigma(W_z x_t + U_z h_{t-1})^j \tag{25}$$

This procedure of taking a linear sum between the existing state and the newly computed state is similar to the LSTM unit. The GRU, however, does not have any mechanism to control the degree to which its state is exposed, but exposes the whole state each time. The candidate activation $h_t^j$ is computed similarly to that of the traditional recurrent unit.

$${h_t}^{\sim j} = tanh(W x_t + U(r_t \odot h_{t-1}))^j \tag{26}$$

where $r_t$ is a set of reset gates and $\odot$ is an element-wise multiplication. The reset gate $r_t$ is computed similarly to the update gate:

$$r_t^j = \sigma(W_r x_t + U_r h_{t-1})^j \tag{27}$$

See Figure 6 for the graphical illustration of the GRU.

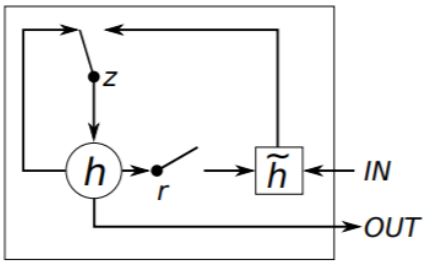


*Figure 6.* Gated Recurrent Unit (GRU)

### 4.2. Model 2 - LSTM as Encoder and Decoder

In this model architecture, we are using LSTM in place of GRU in both encoder and decoder. LSTM here is multi-layer and bidirectional in both encoder and decoder. The decoder is also with Attention that generates the text in target language. Max length of a sentence is kept 15.

## 5. Experimental Results

### 5.1. Image Captioning

We have used Flickr8k data-set (Hodosh et al., 2013) for Image Captioning model. It has images that were selected

*Table 1.* Model Comparison Using Corpus BLEU Score for Image Captioning.

| N-GRAMS | MODEL 1 | MODEL 2 | MODEL 3 |
|---|---|---|---|
| 1-GRAM | 0.3698 | 0.2855 | 0.6587 |
| 2-GRAM | 0.2321 | 0.1572 | 0.4525 |
| 3-GRAM | 0.1545 | 0.1096 | 0.3142 |
| 4-GRAM | 0.1132 | 0.0713 | 0.2197 |

manually to depict various scenes and situations. It consists of two types of data - Flickr8Dataset and Flickr8text. The Flickr8Dataset has 8,092 photos in JPEG format. The Flickr8text contains a number of files containing different sources of descriptions for the photographs. Each image has up to 5 captions, which are a total of 40460 captions. The images are split into 6000 train images, 1000 validation and 1000 test images.

We evaluated the models for Image Captioning using Bilingual Evaluation Understudy (BLEU) score shown in Table 1 and human evaluation.

The optimal parameters of Model 1 are Epochs = 24, Hidden dim = 512, Embedding dim = 300 and Learning rate = 1e-3. We used Google Colab freely available GPU to train models, the approximate training time for 1 epoch of Model 1 is 25 minutes and the final loss was around 1.9.

The optimal parameters of Model 3 are Epochs = 24, Hidden dim = 512, Embedding dim = 300, Attention dim = 512 and Learning rate = 1e-3. The approximate training time for 1 epoch of Model 3 is 15 minutes and the final loss was around 3.1.

**Model 1 Human Evaluation:**


*Figure 7.* Correctly Predicted Caption

Predicted Fig7: Two dogs are running together through the grass.

Actual: 1. A black and white dog running in front of a dog in the grass. 2. A brown dog and a white and black dog are running in grass. 3. A brown dog and black and white dog run along the green grass. 4. Two dogs, a golden and a white with black patches, are running. 5. Two dogs are running together through the grass.

Predicted Fig8: A man in a wetsuit is surfing on a big wave.


*Figure 8.* Correctly Predicted Caption

Actual: 1. A man in a black wetsuit is surfing on a big wave. 2. A man in a wetsuit surfing. 3. A surfer is riding a large wave. 4. Two surfers catch the same huge wave in the ocean. 5. Two surfers ride a wave.


*Figure 9.* Wrongly Predicted Caption

Predicted Fig9: A man in a helmet rides a bike.

Actual: 1. A climber stops to take a drink while climbing a snow covered mountain . 2. A man holding a cup on a snow mountain. 3. A man in yellow suit is holding a cup while standing in snow. 4. A mountain climber stops for a drink. 5. A mountaineer in a yellow jacket is drinking from a cup.


*Figure 10.* Wrongly Predicted Caption

Predicted Fig10: A man in a wetsuit on a surfboard.

Actual: 1. A bird eating. 2. A small , pale bird bends down to examine a crumb. 3. A small white bird looks at a small object. 4. A tan bird stands on a ledge about to eat something. 5. The bird leans over to a small piece of food.

Predicted Fig11: The dogs are in snow brown fence performing play.

Actual: 1. The dogs are in the snow in front of a fence. 2. The dogs play on the snow. 3. Two brown dogs playfully fight in the snow. 4. Two brown dogs wrestle in the snow. 5. Two dogs playing in the snow.

**Model 3 Human Evaluation:**

Predicted Fig12: Two brown dogs are playing in the snow.

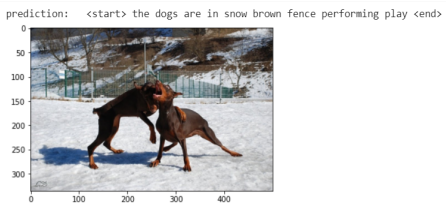


*Figure 11.* Wrongly Predicted Caption


*Figure 12.* Correctly Predicted Caption

Actual: 1. The dogs are in the snow in front of a fence. 2. The dogs play on the snow. 3. Two brown dogs playfully fight in the snow. 4. Two brown dogs wrestle in the snow. 5. Two dogs playing in the snow.


*Figure 13.* Correctly Predicted Caption

Predicted Fig13: A group of people are sitting on a bench.

Actual: 1. A group of people are gathered on a white pillared porch for a photo. 2. A group of people are sitting in front of a red brick and white trim building. 3. A group of people are sitting on the porch of a brick building. 4. A group of people gather in front of a red house. 5.A group of ten people posing outside of a classic-style building.

Predicted Fig14: A football player in a red uniform is running.

Actual: 1. A football player is in a red and white uniform Sooners 28. 2. A large football player in a red uniform. 3.A large football player in a red uniform looks to left. 4. The football player watches the game from the sidelines. 5. This football team wears red shirts and red helmets .

Predicted Fig15: a brown and white dog is playing in a pool.

Actual: 1. The small brown and white dog is in the pool. 2. Small dog is paddling through the water in a pool. 3. A dog

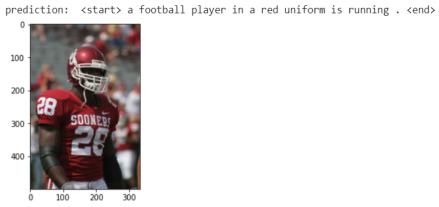


*Figure 14.* Correctly Predicted Caption

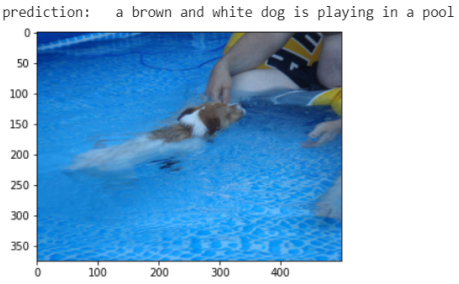


*Figure 15.* Correctly Predicted Caption

swims in a pool near a person. 4. A dog swims in a pool towards a person. 5. A brown and white dog swimming towards some in the pool.


*Figure 16.* Correctly Predicted Caption

Predicted Fig16: a man is riding a bicycle down a hill.

Actual: 1. A biker is racing downhill. 2. A biker prepares to go down a hill. 3. A person in a biking suit rides a bike down a hill. 4. A person in a blue biking uniform is ridding a dirt bike on a grassy track . 5. A person riding a bicycle travels down an overgrown hill.

Predicted Fig17: a women and a women are smiling and smiling

Actual: 1. A girl with dark brown hair and eyes in a blue scarf is standing next to a girl in a fur edged coat. 2. An Asian boy and an Asian girl are smiling in a crowd of people. 3. The girls were in the crowd. 4. Two dark haired girls are in a crowd. 5. Two girls are looking past each other in different directions while standing in a crowd


*Figure 17.* Wrongly Predicted Caption

### 5.2. Neural Machine Translation

For NMT model, we have used IIT Bombay English-Hindi corpus data-set. This data-set contains parallel corpus for English-Hindi from a variety of different sources like GNOME, HindEnCorp, TED talks, Different Indian Government websites, etc. We have used HindEnCorp dataset which consists of 273,885 sentences and a combination of HindEnCorp and GNOME dataset having with a total of 419,591 sentences. We performed sentence level BLEU score evaluation as shown in Figure 18 and 19, by randomly choosing a few sentences.

The optimal parameters of Model 1 are Epochs = 12, Hiddendim = 256, Embeddingdim = 256, dropout = 0.2, Attentiondim = 256 and Learningrate = 1e-2. The approximate training time for 1 epoch of Model 1 is 4 hours and the final loss was around 3.6.

The optimal parameters of Model 2 are Epochs = 5, Hiddendim = 512, Numlayers =2, Embeddingdim = 300, Attentiondim = 512 and Learningrate = 1e-2. The approximate training time for 1 epoch of Model 2 is 3 hours and the final loss was around 2.6.

**Model 1 (GRU) Human Evaluation:**

This model has captured some information of the source sentence nicely. For example, in the first example translation of figure 18, the model has captured invalid name. However, this did not work for all examples as it could not capture given files in the second example.

**Model 2 (LSTM Multi-Layer and Bi-directional) Human Evaluation:**

The model has accurately identified the two words for first and last example in figure 19. Hence, on comparing the sentence level BLEU scores from figure 18 and figure 19, we can say model 2 is better in translating English sentences to Hindi language.

```
English:  user name invalid
Actual:  उपयोगकर्ता नाम में अमान्य
Predicted:  अवैध नाम नाम नाम नाम नाम
Cumulative 1-gram: 0.166667
Cumulative 2-gram: 0.408248
Cumulative 3-gram: 0.553618
Cumulative 4-gram: 0.638943
0.6389431042462724

English:  delete the given files
Actual:  दी गई फ़ाइलें मिटाएँ
Predicted:  मिटाएँ
Cumulative 1-gram: 0.049787
Cumulative 2-gram: 0.049787
Cumulative 3-gram: 0.049787
Cumulative 4-gram: 0.049787
0.049787068367863944

English:  seek error on file s
Actual:  s फ़ाइल पर खोज त्रुटि
Predicted:  त्रुटि s
Cumulative 1-gram: 0.223130
Cumulative 2-gram: 0.223130
Cumulative 3-gram: 0.223130
Cumulative 4-gram: 0.223130
0.22313016014842982

English:  big brother tv series
Actual:  बिग ब्रदर
Predicted:  बिग बिग बिग बिग बिग बिग बिग बिग बिग
Cumulative 1-gram: 0.066667
Cumulative 2-gram: 0.258199
Cumulative 3-gram: 0.409157
Cumulative 4-gram: 0.508133
0.5081327481546147
```

*Figure 18.* NMT GRU Model Evaluation

## 6. Conclusion

We have successfully built models to generate descriptive captions given an image and translating a sentence from

```
English:  glorious revolution
Actual:  गौरवपूर्ण क्रान्ति
Predicted:  गौरवपूर्ण क्रान्ति
Cumulative 1-gram: 1.000000
Cumulative 2-gram: 1.000000
Cumulative 3-gram: 1.000000
Cumulative 4-gram: 1.000000
1.0

English:  home phone
Actual:  आवास फोन
Predicted:  फोन फोन
Cumulative 1-gram: 0.500000
Cumulative 2-gram: 0.707107
Cumulative 3-gram: 0.795536
Cumulative 4-gram: 0.840896
0.8408964152537145

English:  distance between circles
Actual:  वृत्ताकार के बिच की दूरी
Predicted:  वृत्ताकार के बीच के
Cumulative 1-gram: 0.389400
Cumulative 2-gram: 0.317944
Cumulative 3-gram: 0.431158
Cumulative 4-gram: 0.497609
0.49760938992507125

English:  always carbon  copy  to
Actual:  इन्हें हमेशा कार्बन प्रति  भेजें:
Predicted:  इन्हें हमेशा कार्बन प्रति  भेजें:
Cumulative 1-gram: 1.000000
Cumulative 2-gram: 1.000000
Cumulative 3-gram: 1.000000
Cumulative 4-gram: 1.000000
1.0
```

*Figure 19.* NMT LSTM Model Evaluation

English to Hindi by utilizing state of the art sequence to sequence architecture. We have leveraged transfer learning approach for extracting information from image and it is clear that successful implementation of visual attention has helped model significantly in extracting the correct context and information about the image and has generated better captions as shown in human evaluation section. The model is trained to maximize the likelihood of the sentence given the image. We compared our model with or without attention using corpus BLEU Score.

Intriguingly, we played around the two models by combining them. We used image captioning model 3 with visual attention to generate image captions and tried using both model 1 and model 2 described in machine translation section to generate captions for a few images in Hindi. The results are interesting. However, translation models are not very successful in translating the generated captions for the images. The reason for this can be the format of image captioning data is such that an object is performing an activity in a background (Example - A dog is playing in the snow). But, the machine translation data has different sources like Indian News Articles, Ted Talk Translations and Government Websites etc. Though, there are differences in the data sets, the translation model has been successful in correctly translating parts of the generated caption as showing in Figure 20 and Figure 21. This can be further improved by trying different parallel corpus which are closely aligned to image captions.


*Figure 20.* Image Captioning and Machine Translation Models


*Figure 21.* Image Captioning and Machine Translation Models

## References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Hodosh, M., Young, P., and Hockenmaier, J. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47: 853–899, 2013.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057, 2015.