# Social Media Image Caption Generation Using Deep Learning

Rishikesh Gawde
Vidyalankar Institute of Technology

# Social Media Image Caption Generation Using Deep Learning

1Omkar Nitin Shinde, 2Rishikesh Gawde, 3Anurag Paradkar
1Student, 2Student, 3Student
Vidyalankar Institute of Technology

*Abstract* - **Image captioning for Social Media—the task of providing caption of the content within an image—lies at the intersection of Computer Vision (CV) and Natural Language Processing (NLP). In this paper, we present a model based on a recurrent architecture that combines the recent advances in computer vision and machine translation and thus it can be used to generate captions for an image. It is an integral task which requires semantic understanding of images and the ability of generating Captions with proper relevant meaning and structure. The main aim of the paper is to train Convolutional Neural Network (CNN) and Long Short term memory (LSTM) model with many hyper parameters which extract features from the image and map these features to their appropriate descriptive keywords and apply it on a large dataset of pictures and combine the results with a Recurrent Neural Network (RNN) to generate dynamic and suitable captions and hash tags for social media for the classified image. Using the Flickr8K, Flickr30K datasets and also we had designed our own dataset of different categories of image in order to show more accuracy in generating captions.**

***keywords*** **- Image Captioning, Computer Vision, Convolutional Neural Network, Recurrent Neural Network, Long Short term memory**

## I. Introduction

All Humans can describe a scene in an image with no difficulty, but this task has been difficult for computers (Karpathy and Fei-Fei 2015). Scene Description can be instrumental, such as helping visually impaired people better understand images on the internet (Vinyals et al. 2015).

Scene Understanding has been an active area of research in computer vision since the task is broad and requires models that can perform various tasks as one. Mainly, scene understanding requires detecting and recognizing known objects, localizing the objects and learning the spatial relationships between the objects. The model should also be robust to changes in illumination changes, and be able to handle occlusions (Aarthi and Chitrakala 2017).

The task of image captioning is challenging since generating meaningful captions requires identifying essential objects and learning language dependencies and correctly using the recognized objects in a proper sentence based on the language.

Through this project, we aim to choose one approach and experiment with it and show the results that were obtained.

## II. Background

### Convolutional Neural Network

A convolutional neural network is an architecture of a neural network that has been used for image classification and object detection with great success. The most common architecture consists of three main operations that are repeated several times. Firstly, a convolution is applied to the image are not predetermined but are learned from the data. Further, this result is then compressed into smaller matrices with the help of pooling. Pooling is the process of aggregating results from regions in a manner similar to convolution. Pooling can be done in various ways such as max-pooling that retains the maximum element in the pooling region, min pooling or average pooling.

The size of the convolution filter, the pooling region, the number of filters to use and the number of convolution and dense layers are hyper parameters and need to be set by trying what works best for a problem. These are the parameters that need to be set properly for best results. (contributors 2018a)

There are various architectures of Convolutional Neural networks that have been used for different tasks, and there is active research in the area. Three models were used for the purpose of this project -VGG16, VGG19, and ResNet50.

The VGG architecture was first introduced in (Simonyan and Zisserman 2014) for image classification on the ImageNet data set. The architecture is known for its simplicity as it uses only a 3 ×3 convolutional layers stacked on top of each other. The VGG 19 model has more of these layers stacked compared to the VGG16. The volume is reduced in each step with the help of max pooling. These convolutional layers are followed by two fully connected layers that flatten the output and a softmax layer that predicts the probability of each of the objects the model was trained for. (Rosebrock 2017).

The Residual Networks (ResNet) introduced first in (He et al. 2016) consists of stacked Residual blocks. The residual block comprises a skip connection which makes it easier to learn the identity function and hence stacking theseresidual blocks helps us go more in-depth and avoid thediminishing gradient problem (Rosebrock 2017).

**Recurrent Neural Network**

One of the significant limitations of using traditional feed forward neural networks is they are trained with input and output vectors. The order of these vectors does not affect the predictions made after training, and they produce fixed size outputs. There is no way of learning sequences and learning context or dependencies in various vectors in the sequence when using traditional neural networks. This is where Recurrent Neural Networks come to help (Karpathy 2015).

The input to the hidden layer in an RNN (with a single hidden layer) is the input vector, along with the output of the hidden layer for the previous time step. The RNNs are trained to learn to predict the next word given the current example. This is done for multiple times in each iteration, which represents the length of the sequence that the RNN can learn and predict later. The Network is trained using back propagation through time, which adjusts the weights between the hidden layer for a given time step and the next time step. Once trained for var- ious iterations, the RNN can learn to model the sequence (contributors 2018c).

There are problems with RNNs, when learning long sequences, in situations such as the language translation of large documents, where it may be necessary to remember only the context over a small time period. For this purpose, Long Short Term Memory (LSTM) networks are used, where each cell has three gates - input, forget and output and can learn when to forget the previous context, along with other parameters. The LSTM is trained such that each LSTM cell updates its weights at each time step and the all the weights are updated after each iteration. This helps the network learn long sequences and decide which parts of the sequences are related with some context (Trask 2015) (contributors 2018b). Gated Recurrent Units (GRUs), another type of Recurrent Neural Network, have also become quite popular to solve the same problem. In a GRU unit, there is only an update gate and a reset gate; however, there exist variations of GRU such as a minimal gated unit in which there exists only one gate (forget gate). GRUs train faster compared to LSTMs while giving comparable results.

RNNs, LSTMs, and GRUs have been used to learn long sequences of text and music and to generate new documents given some starting words or phrases (Karpathy 2015).

## III. RELATED WORK

One of the influential papers by Andrej Karpathy et al. in image captioning divides the task into two steps: mapping sentence snippets to visual regions in the image and then using these correspondences to generate new descriptions (Karpathy and Fei-Fei 2015). The authors use a Region Convolutional Neural Network (RCNN) to represent images as a set of h dimensional vectors each representing an object in the image, detected based on 200 ImageNet classes. The authors represent sentences with the help of a Bidirectional Recurrent Neural Network (BRNN) in the same h dimensional space. Each sentence is a set of h dimensional vectors, representing snippets or words. The use of the BRNN enriches this representation as it learns knowledge about the context of each word in a sentence. The authors find that with such a representation, the final representation of words aligns strongly with the representation of visual regions related to the same concept. They define an alignment score on this representation of words and visual regions and align various words to the same region generating text snippets, with the help of a Markov Random Field. With the help of these correspondences between image regions and text snippets, the authors train another model that generates text descriptions for new unseen images (Karpathy and Fei-Fei 2015).

The authors train an RNN that takes text snippets and visual regions as inputs and tries to predict the next word in the text based on the words it has seen so far. The image region information is passed to the network as the initial hidden state at the initial time step, and the network learns to predict the log probability of the next most likely word using a softmax classifier. The authors use unique START and END tokens that represent the beginning and end of the sentence, which allows the network to make variable length predictions. The RNN has 512 nodes in the hidden layer (Karpathy and Fei-Fei 2015).

The network for learning correspondences between visual regions and text words was trained using stochastic gradient descent in batches of 100 image-sentence pairs. The authors used dropouts on every layer except the recurrent layers and clipped the element-wise gradients at 5 to prevent gradient explosion. The RNN to generate descriptions for unseen im- ages was trained using RMSprop which dynamically adjusts the learning rate (Karpathy and Fei-Fei 2015).

Kelvin Xu et.al (Xu et al. 2015) use the concept of attention to better describe images. The authors propose models that focus on which area of the image, and what objects in the image are being given attention and evaluate these models on different image captioning datasets. The idea be- hind the approach is that much like the human visual system, some parts of the image may be ignored for the task of im- age description, and only the salient foreground features are considered. The authors use a CNN to learn important features of the image and an LSTM (Long short-term memory network) to generate description text based on a context vector.

Jyoti Aneja et al. in (Aneja, Deshpande, and Schwing 2017) use a convolutional approach to generate description text instead of a simple RNN, and show that their model works at par with RNN and LSTM based approaches.

Andrew Shin et al. (Shin, Ushiku, and Harada 2016) use a second neural network, finely tuned on text-based sentiment analysis to generate image descriptions which capture the sentiments in the image. The authors use multi-label learning to learn sentiments associated with each of the im- ages, then use these sentiments, along with the input from the CNN itself as inputs to an LSTM to generate sentences which include the sentiment. The LSTM is restricted sothat each description contains at least one term from the senti ment vocabulary.

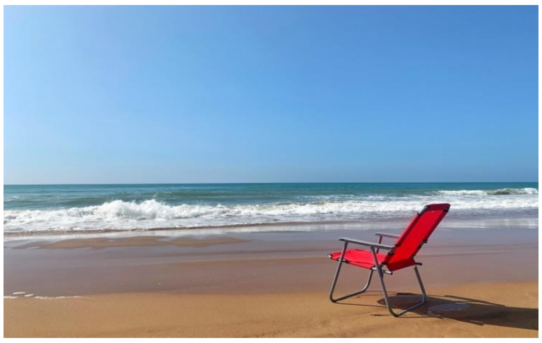


Figure 1: Sample image and captions

Alexander Mathews et al. (Mathews, Xie, and He 2016) emphasize how only a few image descriptions in most datasets contain words describing sentiments, and most descriptions are factual. The authors propose a model that consists of two CNN + RNN models each with a specific task. While one model learns to describe factual content in the image, the other learns to describe the sentimental ssociated, thus providing a framework that learns to generate sentiment based descriptions even with lesser image sentiment data.

Quanzeng You et.al in (You, Jin, and Luo 2018) propose approaches to inject sentiment into the descriptions generated by image captioning methods.

Tsung Yi Lin et.al in (Lin et al. 2014) describes the Microsoft Common Objects in Context dataset,that is widely used for benchmarking image captioning models.

.

## IV. APPROACH

### Dataset

The data set is a collection of 10000 images with five captions each, collected in one place, and available to be used for the benchmarking of image captioning and im- age querying approaches (Rashtchian et al. 2010).The authors show that better results can be achieved when multiple captions are used with each image, to train the model. A manual data set of 2000 images was created with relevant 50000 caption is was used to provide final results of model

Figure 1 is a sample image file in dataset. The image is paired with following five human-generated training captions:

- Sunsets and oceans. It's what I do.
- A mind-boggling, awe-inspiring, spine-tingling sunset.
- A sunset that good doesn't need a filter.
- Watch the sunset. Not Netflix.
- Here comes the sunset.
- Data Preprocessing

We divide the training data (10000) and the captions into three different data sets - the training set (8000), the validation set (1000) and the test set (1000). For each of the captions in the three data sets, we create a set of training input and target captions by shifting the training input caption by one word to get the training target caption.

### Image Preprocessing

To generate image features we use pretrained weights of CNNs trained on ImageNet image classification dataset (VGG16, VGG19, and ResNet50) and remove the final dense layers from the model. We preprocess images and generate image features using the by performing a forward pass on the image on using these weights and save these features to a file.

### Caption Preprocessing

To preprocess the image captions in the training data, we first identify all the words that are there in the data set. We then generate a histogram of the distribution of these words and drop the words that occur less than five times. We end up with the vocabulary of size 2531 words

Model takes an input image and then further generates a caption as shown in figures 2 and 3.

### Model

The model that was used for the project consists of two different input streams, one for the image features, and the other for the preprocessed input captions. The image features are passed through a fully connected (dense) layer to get a representation in a different dimension. The input captions are passed through an embedding layer. These two input streams are then merges and passed as inputs to an LSTM layer. The image is passed as the initial state to the LSTM while the caption embedding's are passed as the input to the LSTM. The architecture is shown in figure 2.

### Training

The model was trained first on Nvidia GeForce MX 250. We faced memory problems using different batch sizes and hence moved to a better GPU. We then used the Alienware R17 which includes intel i9 processor with 6 core, 32GB RAM and Nvidia GTX1080 8GB. Training the model takes about 6 hours

To train the model, for each image and each of the input captions that were generated during preprocessing, we pass the image features through the dense layer, and the pre- processed input captions to the embedding layer. We then use the image as the initial state to the LSTM, along with the caption which is passed as the input to the LSTM. The model outputs a predicted caption as shown in figure 3.

## Word Embedding

Word Embedding's provide a vector representation of words that can capture something about the context of the word. There are many pretrained word embedding available; however, our model learns the word embedding as part of the model itself.

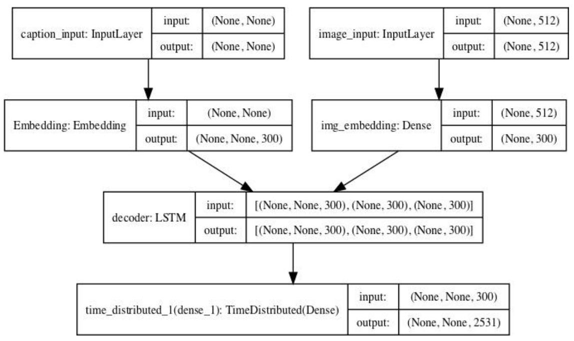

caption_input: InputLayer
input: (None, None)
output: (None, None)
image_input: InputLayer
input: (None, 512)
output: (None, 512)
Embedding: Embedding
input: (None, None)
output: (None, None, 300)
img_embedding: Dense
input: (None, 512)
output: (None, 300)
decoder: LSTM
input: [(None, None, 300), (None, 300), (None, 300)]
output: [(None, None, 300), (None, 300), (None, 300)]
time_distributed_1(dense_1): TimeDistributed(Dense)
input: (None, None, 300)
output: (None, None, 2531)

Figure 2: Model Architecture

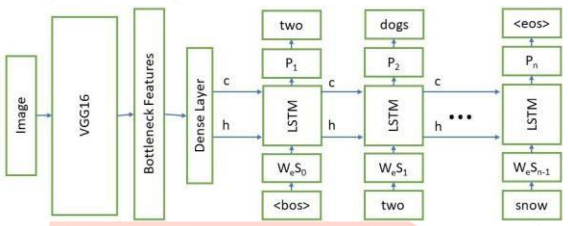


Figure 3: Training the model using VGG16 image features

## Evaluation Metric

The Bilingual Evaluation Understudy score (BLEU score) was used as the metric to evaluate the generated captions generated by the model. The BLEU score is a metric for evaluation of machine-generatedtranslations, but can also be used to evaluate machine generated sentences in various natural language processing tasks. It is a common metric for evaluating image captions (Jason Brownlee 2017).

The BLEU score is designed to give a score between 0 to 1 (often scaled to a range of 0-100) on a corpus level and often does not produce good results on individual sentences. The BLUE score can be calculated for various n-grams and represents for each sentence the relative number of matching n-grams in the reference sentences. The scores of all the sentences are combined using a geometric mean with a penalty applied to short sentences to prevent very short sentences that are not suitable translations, from having high scores (Wikipedia contributors 2018a).

For the purpose of this project we calculate BLEUscores for unigrams u to 4-grams (BLEU1 to BLEU4 respectively) to compare results of different models that were used. We use the images in the test set, and generate captions using our model.

| Metric | VGG16 | VGG19 | ResNet50 |
|---|---|---|---|
| BLEU1 | 51.21 | 52.59 | 51.56 |
| BLEU2 | 21.39 | 21.89 | 22.68 |
| BLEU3 | 8.28 | 8.20 | 8.95 |
| BLEU4 | 3.28 | 3.23 | 3.88 |

Table 1: Greedy (embedding size: 300, LSTM size: 300, learning rate: 0.0001, dropout: 0.2, batch size: 32, epochs:10)

| Metric | VGG16 | VGG19 | ResNet50 |
|---|---|---|---|
| BLEU1 | 54.26 | 54.98 | 56.74 |
| BLEU2 | 23.98 | 24.18 | 25.85 |
| BLEU3 | 10.01 | 10.02 | 10.89 |
| BLEU4 | 3.94 | 3.87 | 4.42 |

Table 2: Beam Search (embedding size: 300, LSTM size: 300, learning rate: 0.000051, dropout: 0.2, batch size: 32, epochs: 33)

## Optimization

The RMSprop optimization was used to minimize the loss and train the model. The problem with training deep networks for complicated tasks is that the gradient of these arbitrarily complicated functions can either explode or vanish as the errors are back propagated. RMSprop optimization uses a moving average of the squared gradients to normalize the gradient. This in effect adaptively changes the step size depending on the gradient value medium-RMSprop. RMSprop was developed for batch training of neural networks and it has been observed that RMSprop works well for LSTM networks.

**Inference**

To perform inference, we first obtain image embedding by passing the image through the CNN model and then the dense layer. Then to generate captions using the model, we first feed the LSTM cell as the first input and image embedding as its initial states. The LSTM produces a word and its hidden states, and we keep feeding this word and hidden states again to the LSTM cell reaches the max sentence length.

| Metric | VGG16 | VGG19 | ResNet50 |
|---|---|---|---|
| BLEU1 | 55.16 | 55.54 | 57.39 |
| BLEU2 | 24.73 | 24.66 | 26.51 |
| BLEU3 | 10.71 | 10.51 | 11.78 |
| BLEU4 | 4.28 | 4.46 | 4.97 |

Table 3: Beam Search (embedding size: 512, LSTM size: 512, learning rate: 0.000051, dropout: 0.2, batch size: 32,epochs: 33)

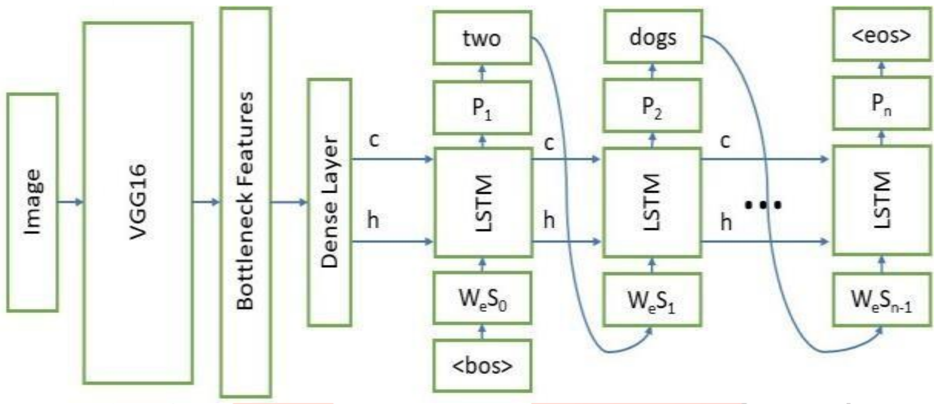


Figure 4: Inference: Generating captions once the LSTM is trained

**Beam Search**

Instead of sampling in a greedy approach described above, the better way is to do Beam Search where we keep k best sentences produced so far up to time t to generate sentences of size t + 1. In this way, we get k best sentences at the end. Beam Search significantly improved our BLEU scores as shown in table 2. We used various beam sizes for our experiments. Using Beam Search favors sentences with shorter lengths, and so we implemented length normalization to get the sentences with maximum average log probability.

## V. EXPERIMENTS

The model was trained 3 times for each of the CNNs models that we used. First, we trained the model using the learning rate as 0.0001 for 10 epochs and used greedy approach to generate captions. The BLEU scores for this set of hyper parameters are shown in table 1.

Next we decreased the learning rate to 0.000051, and trained the model to 33 epochs and used beam search to generate sentences. The BLEU scores for this set or hyper- parameters are given in table 2. We can see that there is a significant improvement in the BLEU scores

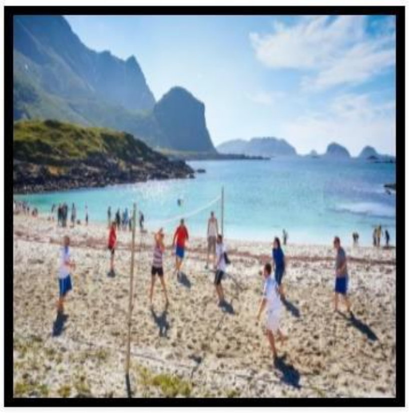
Finally we increased the size of the embedding layer and the dense layer from 300 to 512, increased the LSTM size from 300 to 512 and trained the model again for 33 epochs. The BLEU scores for these hyper parameters are given in table 3. We can see that increasing the LSTM size and the size of the embedding layer lead to even better results, even though it took significantly longer time to train the models with these hyperparameters.

We cleaned up our code and created two files that use argument parser to specify parameters for training and the model, and generating captions using trained weights. We also created a simple web application in python with the help of flask that allows users to upload an image and uses the the trained weights to generate image captions.

## VI. RESULTS

The table 3 shows the bleau score is highest with the model that uses ResNet to generate image features. The results shown here are for the ResNet50 model using a LSTM size of 512 and a embedding layer and a dense layer size of 512. The images in the results show captions, generated.

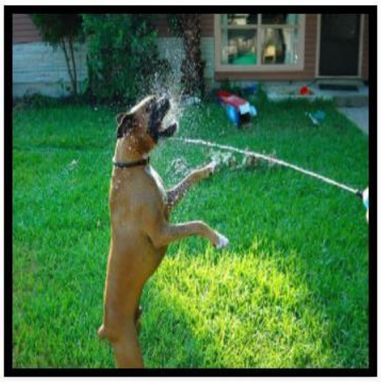
The sentence corresponds to the best caption according to the beam search.



Generated caption

*sandy toes sun kissed nose*

Figure 5: Beach: Results using ResNet50, LSTM size 512



Generated caption

*we are letting nature take its course*

Figure 6: Dog: Results using ResNet50, LSTM size 512

## VII. CONCLUSION

Through this project, we learned about the deep learning techniques used for image captioning problem. We experimented with three distinct CNN models and compared the results of different models using BLEU scores, By comparing we came to a conclusion that ResNet50 was the most suitable and efficient model for us. It captured more than enough information about the image to generate captions.

We learned that the result of generated captions is influenced by the training dataset. The Flickr8k and manual dataset contains many outdoor images of Nature, Beaches and sunsets and our model gives better results on outdoor images without people and is capable of differentiatitng between various natural objects.

We implemented beam search and found that the BLEU scores for sentences generated using beam search are significantly better Future Work.

## VIII. FUTURE WORK

In this project we implemented a model, and experimented with different parameters to see the results. Future exploration can be done to compare the current results to those obtained with using different CNN models. Further comparisons can be made to approaches that include visual and temporal attention. We also intend to create an Android app for the users.

## IX. ACKNOWLEDGMENT

We thank Professor Swati Sharma for giving us the opportunity to work on the project and for providing the necessary resources.

**References**

[1] Aarthi, S., and Chitrakala, S. 2017. Scene understandinga survey. In Computer, Com- munication and Signal Processing (ICCCSP), 2017 Interna- tional Conference on, 1–4. IEEE.
[2] Amazon Web Services. 2018. Amazon ec2 p2 instances.
[3] Aneja, J.; Desh- pande, A.; and Schwing, A. 2017. Convolutional image captioning. arXiv preprint arXiv:1711.09151.

[4] contributors, W. 2018a. Convolutional neural network — wikipedia, the free encyclopedia. [On- line; accessed 30-March-2018].
[5] contributors, W. 2018b. Long short- term memory — wikipedia, the free encyclopedia. [Online; accessed 30-March-2018].
[6] contributors, W. 2018c. Recurrent neu- ral network — wikipedia, the free encyclopedia. [Online; accessed 30-March-2018].
[7] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In Proceed- ings of the IEEE conference on computer vision and pattern recognition, 770–778.
[8] Jason Brownlee. 2017. A gentle in- troduction to calculating the bleu score for text in python.
[9] Karpathy, A., and Fei-Fei, L. 2015. Deep visual-semantic alignments for generating im- age descriptions. In Proceedings of the IEEE conference on computer vision and pattern recognition, 3128–3137.
[10] Karpathy, A. 2015. The unreasonable ef- fectiveness of recurrent neural networks. Andrej Karpathy's Blog.
[11] Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dolla´r, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In European conference on computer vision, 740–755. Springer.
[12] Mathews, A. P.; Xie, L.; and He, X. 2016. Senticap: Generating image descriptions with sentiments. In AAAI, 3574–3580.
[13] Rashtchian, C.; Young, P.; Hodosh, M.; and Hockenmaier, J. 2010. Collecting image annotations using amazon's mechanical turk. In Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, 139–147. Association for Computational Linguistics.
[14] Rosebrock, A. 2017. Imagenet: Vggnet, resnet, inception, and xception with keras. pyimagesearch website.
[15] Shin, A.; Ushiku, Y.; and Harada, T. 2016. Image captioning with sentiment terms via weakly-supervised sentiment dataset. In BMVC.
[16] Simonyan, K., and Zisser- man, A. 2014. Very deep convolutional networks for large- scale image recognition. arXiv preprint arXiv:1409.1556.
[17] Trask, A. 2015. Anyone can learn to code an lstm-rnn in python (part 1: Rnn). iamtrask github.io blog.
[18] Vinyals, O.; Toshev, A.; Bengio, S.;and Erhan, D. 2015. Show and tell: A neural image caption gen- erator. In Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, 3156–3164. IEEE.
[19] Wikipedia contributors. 2018a. Bleu — Wikipedia, the free encyclopedia. [Online; accessed 30-April-2018].
[20] Wikipedia contributors. 2018b. Cross entropy — Wikipedia, the free encyclopedia. [Online; accessed 30-April-2018].
[21] Arnav Arnav, Hankyu Jang, Pulkit Maloo Image Captioning Using Deep Learning
[22] Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In International Conference on Machine Learning, 2048–2057.