Text Generation from image

Kruti Thukral (012586041)

Motivation

- Aid to the blind
  - Help visually impaired people better understand the content of images on the web
- Efficient google image search
- Automatic captioning of frames from CCTV footage
  - Timely notification of suspected behaviour will reduce crime rate
- Self driving cars
  - Can give a boost to the self driving system by inferring from the surroundings

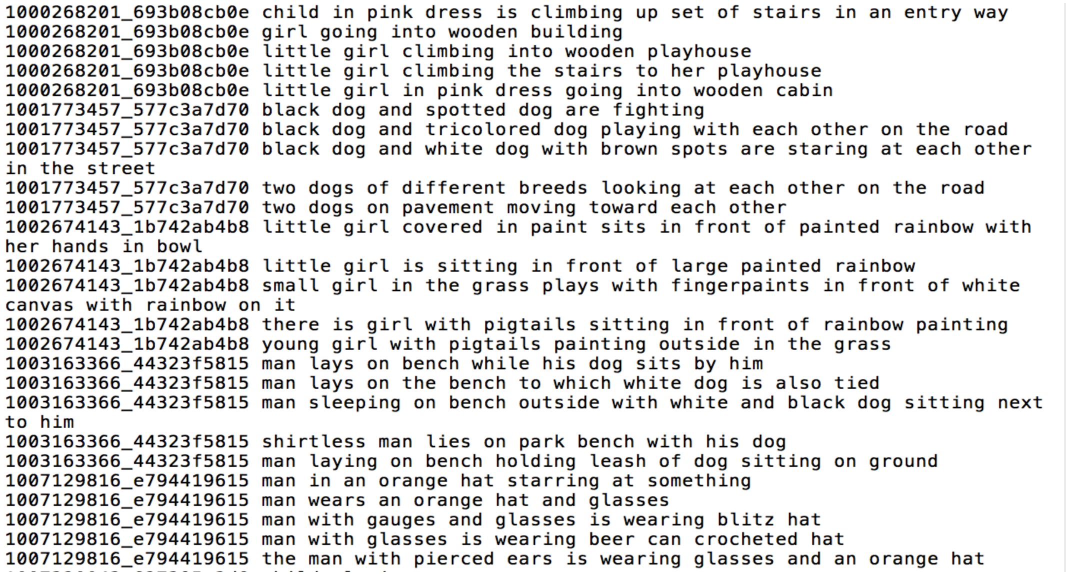
Data Collection

- Popular Datasets
  a. Flickr 8k (containing 8k images)
  b. Flickr 30k (containing 30k images)
  c. MS COCO (containing 180k images)

- Dataset used in project
  a. Flickr 8k - 5 captions for each image
  b. Training Set — 6000 images
  c. Dev Set — 1000 images
  d. Test Set — 1000 images

Data Understanding and cleaning - Captions

- Upper case to lower case
- Removing special tokens (like ‘%’, ‘$’, ‘#’, etc.)
- Eliminating words which contain numbers (like ‘hey199’, etc.)



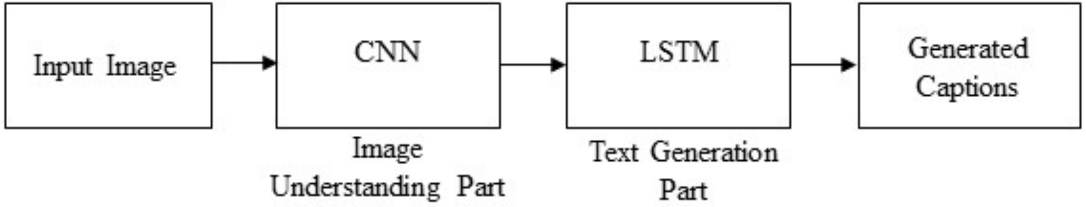
```
1000268201_693b08cb0e child in pink dress is climbing up set of stairs in an entry way
1000268201_693b08cb0e girl going into wooden building
1000268201_693b08cb0e little girl climbing into wooden playhouse
1000268201_693b08cb0e little girl climbing the stairs to her playhouse
1000268201_693b08cb0e little girl in pink dress going into wooden cabin
1001773457_577c3a7d70 black dog and spotted dog are fighting
1001773457_577c3a7d70 black dog and tricolored dog playing with each other on the road
1001773457_577c3a7d70 black dog and white dog with brown spots are staring at each other
in the street
1001773457_577c3a7d70 two dogs of different breeds looking at each other on the road
1001773457_577c3a7d70 two dogs on pavement moving toward each other
1002674143_1b742ab4b8 little girl covered in paint sits in front of painted rainbow with
her hands in bowl
1002674143_1b742ab4b8 little girl is sitting in front of large painted rainbow
1002674143_1b742ab4b8 small girl in the grass plays with fingerpaints in front of white
canvas with rainbow on it
1002674143_1b742ab4b8 there is girl with pigtails sitting in front of rainbow painting
1002674143_1b742ab4b8 young girl with pigtails painting outside in the grass
1003163366_44323f5815 man lays on bench while his dog sits by him
1003163366_44323f5815 man lays on the bench to which white dog is also tied
1003163366_44323f5815 man sleeping on bench outside with white and black dog sitting next
to him
1003163366_44323f5815 shirtless man lies on park bench with his dog
1003163366_44323f5815 man laying on bench holding leash of dog sitting on ground
1007129816_e794419615 man in an orange hat starring at something
1007129816_e794419615 man wears an orange hat and glasses
1007129816_e794419615 man with gauges and glasses is wearing blitz hat
1007129816_e794419615 man with glasses is wearing beer can crocheted hat
1007129816_e794419615 the man with pierced ears is wearing glasses and an orange hat
```

Data Preprocessing

- Images
  - Convert all the images to size 299x299 as expected by the inception v3 model
- Captions
  - Come up with unique words in the caption dataset and create a vocabulary
  - Represent every unique word in the vocabulary by an integer (index)
  - We have 1652 unique words in the corpus and thus each word will be represented by an integer index between 1 to 1652.
  - Calculate maximum length of any caption(In our case, it comes to 34)
  - Add “startseq” and “endseq” to every caption

Architecture

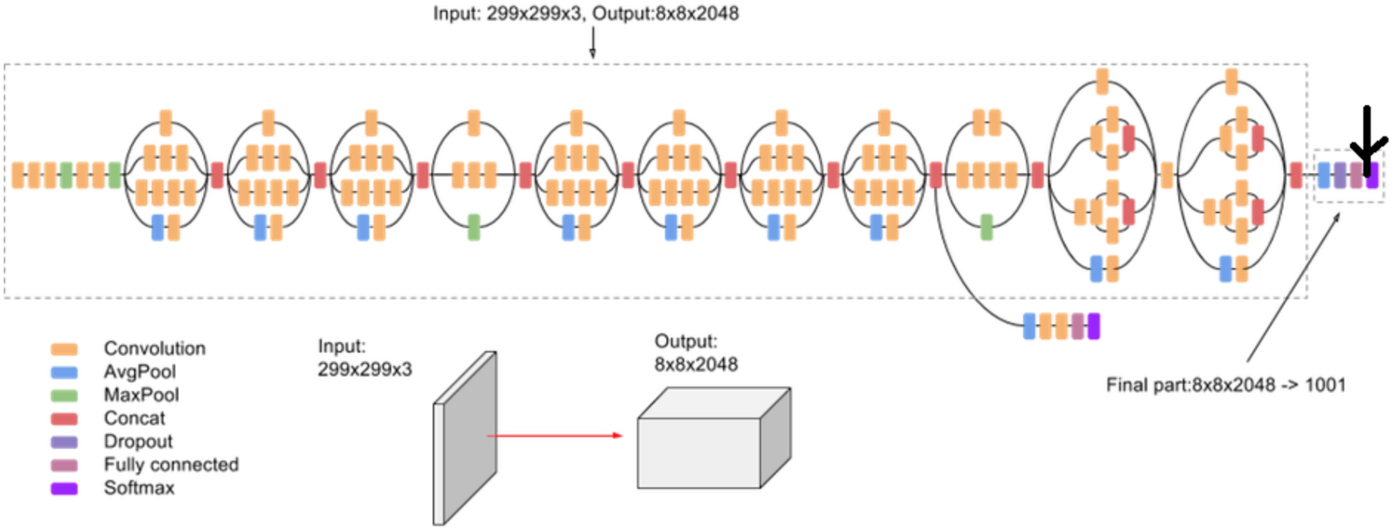
- A convolution neural network that encodes an image into a compact representation, followed by a recurrent neural network that generates a corresponding sentence.
- Global image features are extracted from the hidden activations of CNN
- Image features are fed into an LSTM to generate a sequence of words



Steps

- Convert image to feature vector using CNN
    - Leverage transfer learning by using InceptionV3 model (Convolutional Neural Network) created by Google Research. This model was trained on Imagenet dataset to perform image classification on 1000 different classes of images.
    - Our purpose here is to get fixed-length informative vector for each image.
    - Remove the last softmax layer from the model
    - Pass every training and test image to the CNN model (inception v3) to get the corresponding 2048 length feature vector
    - Stored the feature vector in a pickle file
- Generate text using LSTM
    - Compute data matrix for image and caption
    - Map the every word (index) to a 200-long vector using a pre-trained GLOVE Model
    - As we have two inputs, image vector and captions, create a merge model
    - For training the model, Use SGD with “adam” optimizer and compute loss using categorical_crossentropy

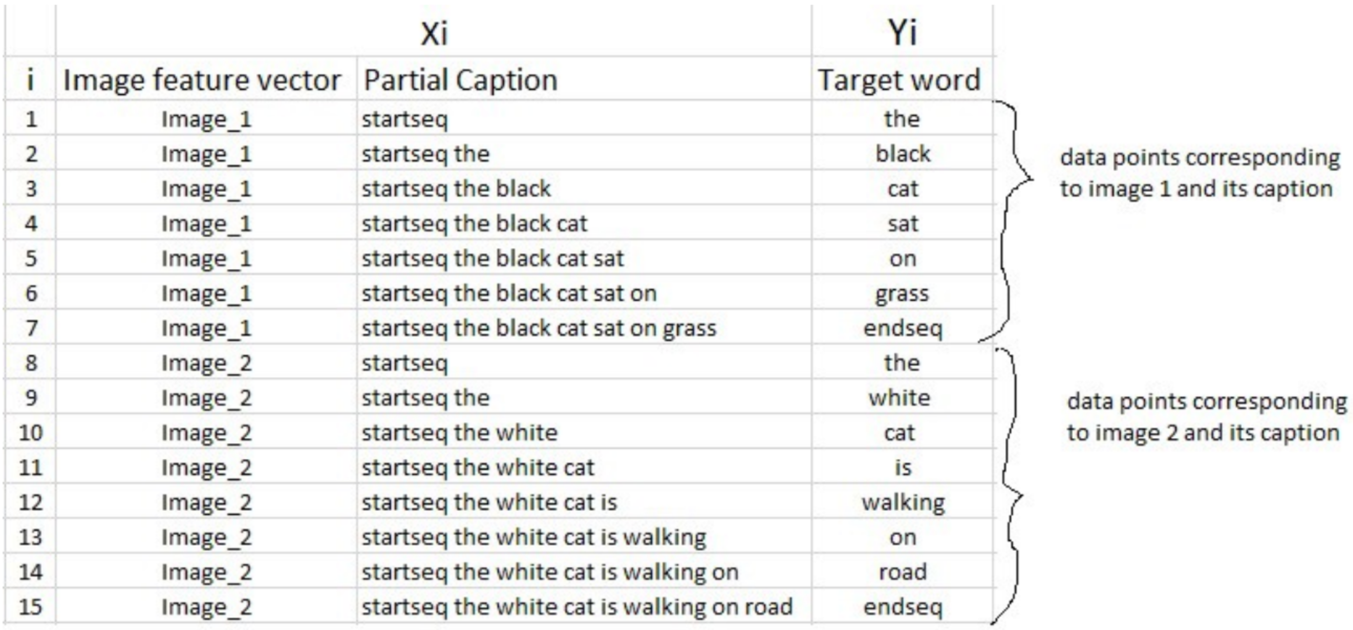
Feature extraction of images



At this point we extract the 2048 - length vector by removing the last softmax layer which performs 1000-class classification

Feature Vector Extraction (Feature Engineering) from InceptionV3

Data Matrix for image and captions



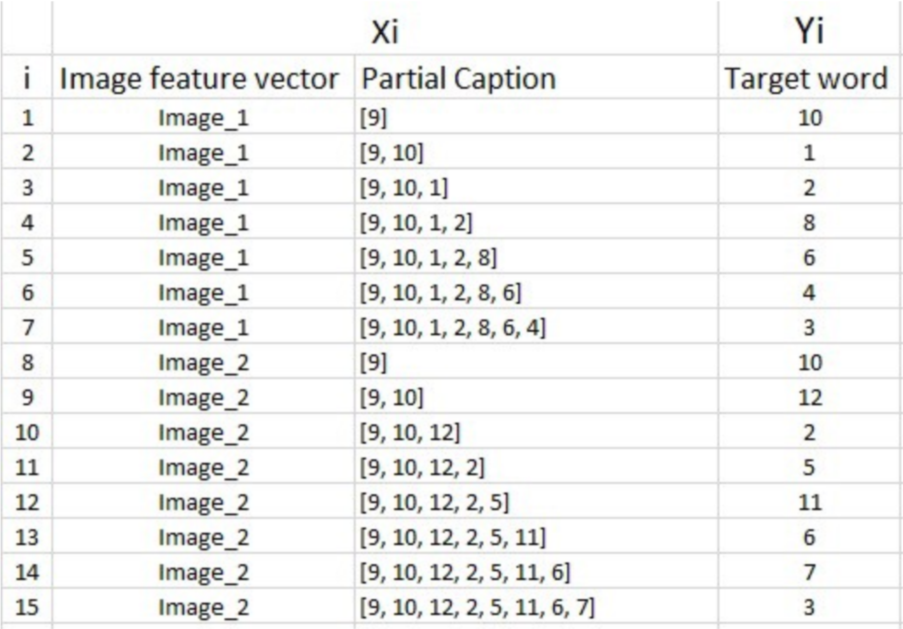
	Xi		Yi
i	Image feature vector	Partial Caption	Target word
1	Image_1	startseq	the
2	Image_1	startseq the	black
3	Image_1	startseq the black	cat
4	Image_1	startseq the black cat	sat
5	Image_1	startseq the black cat sat	on
6	Image_1	startseq the black cat sat on	grass
7	Image_1	startseq the black cat sat on grass	endseq
8	Image_2	startseq	the
9	Image_2	startseq the	white
10	Image_2	startseq the white	cat
11	Image_2	startseq the white cat	is
12	Image_2	startseq the white cat is	walking
13	Image_2	startseq the white cat is walking	on
14	Image_2	startseq the white cat is walking on	road
15	Image_2	startseq the white cat is walking on road	endseq

data points corresponding to image 1 and its caption

data points corresponding to image 2 and its caption

Data Matrix for both the images and captions

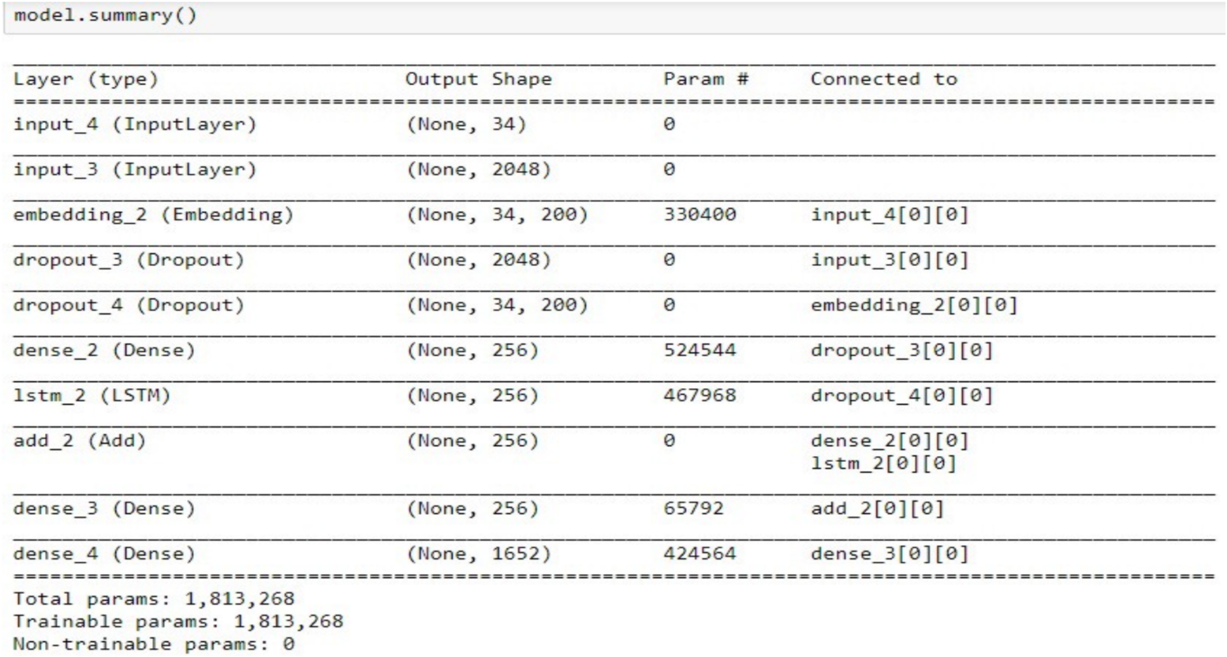
Data Matrix for image and captions



	Xi		Yi
i	Image feature vector	Partial Caption	Target word
1	Image_1	[9]	10
2	Image_1	[9, 10]	1
3	Image_1	[9, 10, 1]	2
4	Image_1	[9, 10, 1, 2]	8
5	Image_1	[9, 10, 1, 2, 8]	6
6	Image_1	[9, 10, 1, 2, 8, 6]	4
7	Image_1	[9, 10, 1, 2, 8, 6, 4]	3
8	Image_2	[9]	10
9	Image_2	[9, 10]	12
10	Image_2	[9, 10, 12]	2
11	Image_2	[9, 10, 12, 2]	5
12	Image_2	[9, 10, 12, 2, 5]	11
13	Image_2	[9, 10, 12, 2, 5, 11]	6
14	Image_2	[9, 10, 12, 2, 5, 11, 6]	7
15	Image_2	[9, 10, 12, 2, 5, 11, 6, 7]	3

Data matrix after replacing the words by their indices

Model Summary

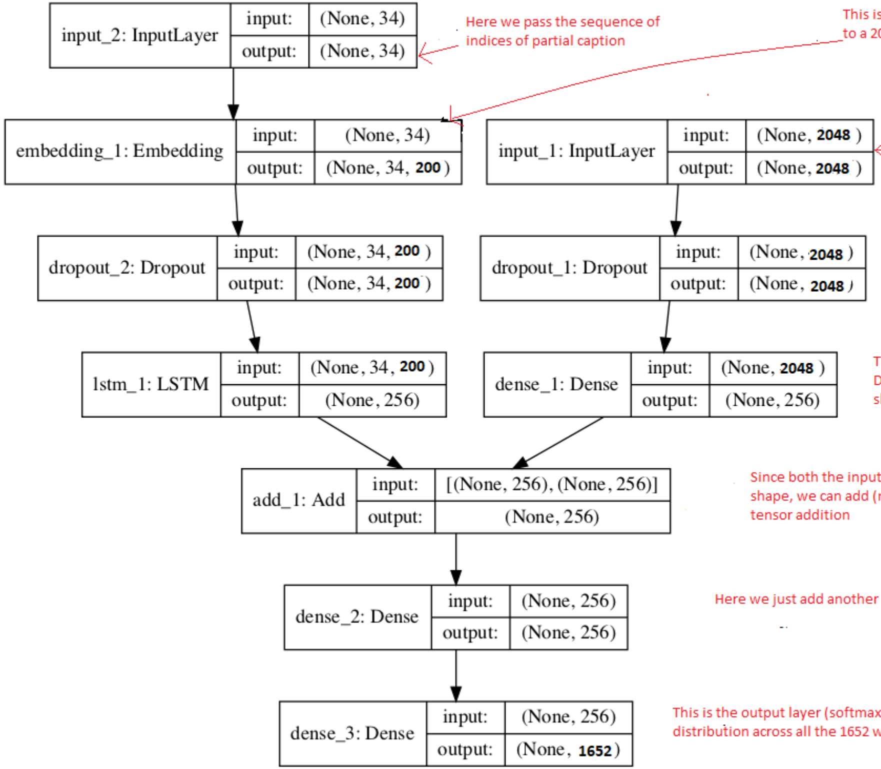


```
model.summary()
```

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_4 (InputLayer)            (None, 34)           0
__________________________________________________________________________________________________
input_3 (InputLayer)            (None, 2048)         0
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, 34, 200)      330400      input_4[0][0]
__________________________________________________________________________________________________
dropout_3 (Dropout)             (None, 2048)         0           input_3[0][0]
__________________________________________________________________________________________________
dropout_4 (Dropout)             (None, 34, 200)      0           embedding_2[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 256)          524544      dropout_3[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   (None, 256)          467968      dropout_4[0][0]
__________________________________________________________________________________________________
add_2 (Add)                     (None, 256)          0           dense_2[0][0]
                                                                 lstm_2[0][0]
__________________________________________________________________________________________________
dense_3 (Dense)                 (None, 256)          65792       add_2[0][0]
__________________________________________________________________________________________________
dense_4 (Dense)                 (None, 1652)         424564      dense_3[0][0]
==================================================================================================
Total params: 1,813,268
Trainable params: 1,813,268
Non-trainable params: 0
```

Summary of the parameters in the model

Model Structure



input_2: InputLayer — input: (None, 34) — output: (None, 34)

embedding_1: Embedding — input: (None, 34) — output: (None, 34, 200)

input_1: InputLayer — input: (None, 2048) — output: (None, 2048)

dropout_2: Dropout — input: (None, 34, 200) — output: (None, 34, 200)

dropout_1: Dropout — input: (None, 2048) — output: (None, 2048)

lstm_1: LSTM — input: (None, 34, 200) — output: (None, 256)

dense_1: Dense — input: (None, 2048) — output: (None, 256)

add_1: Add — input: [(None, 256), (None, 256)] — output: (None, 256)

dense_2: Dense — input: (None, 256) — output: (None, 256)

dense_3: Dense — input: (None, 256) — output: (None, 1652)

Here we pass the sequence of indices of partial caption

This is where every index gets mapped to a 200 dimensional vector

Here we pass the image feature vector of length 2048

The two dropout layers at this level are used to avoid overfitting in the model

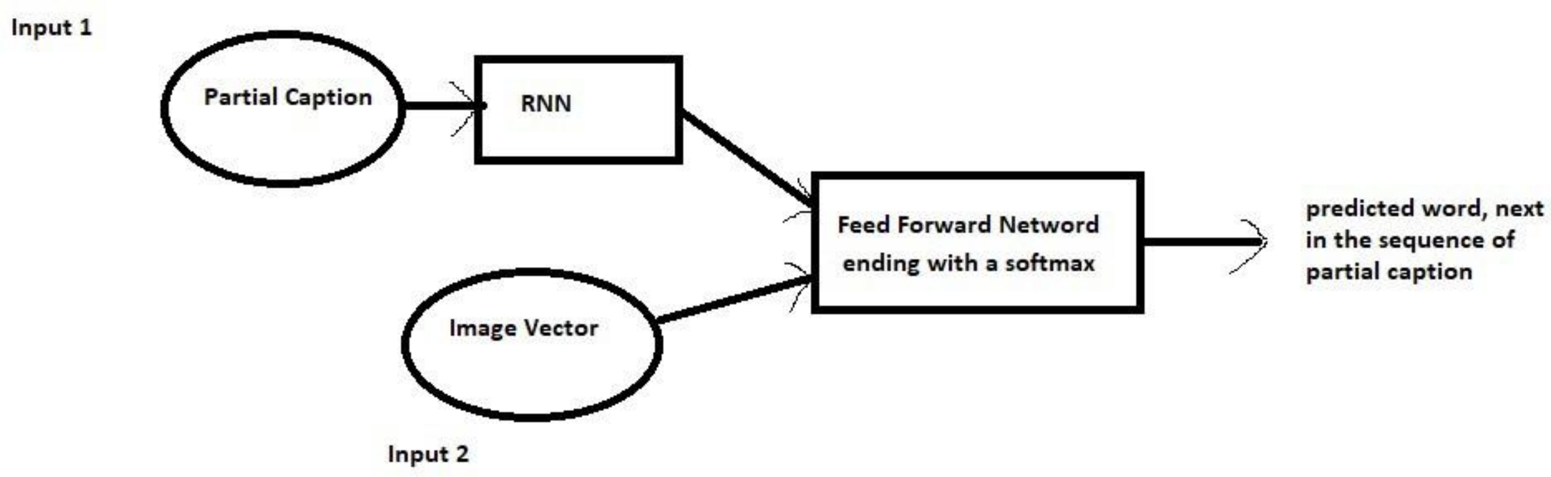
The output of both LSTM (in case of caption model) and Dense layer (in case of image model) at this level is of the shape (Batch_size, 256)

Since both the input tensors to this layer are of the same shape, we can add (merge) them into a single tensor using tensor addition

Here we just add another dense layer

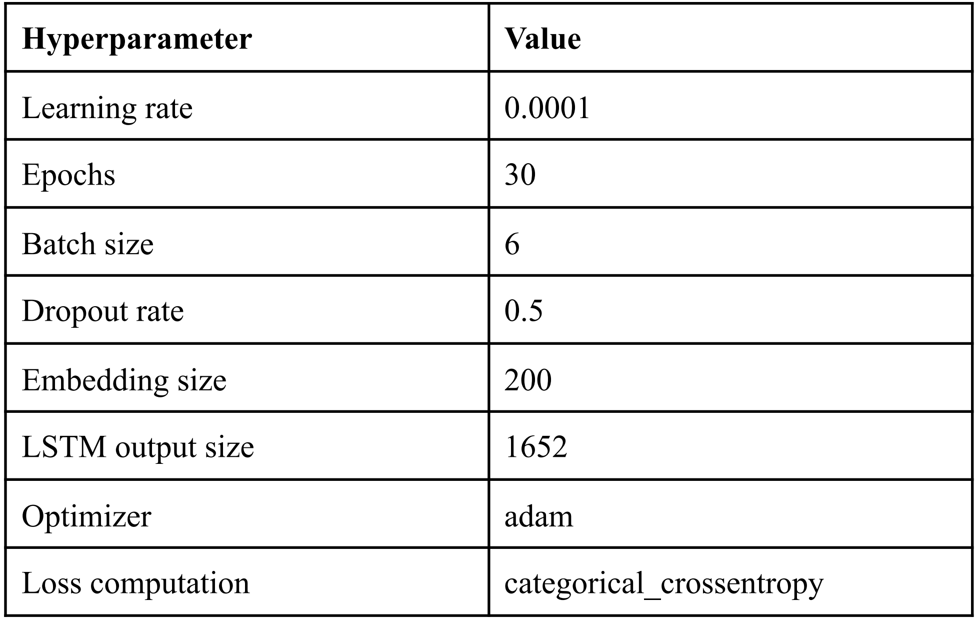
This is the output layer (softmax) which generates the probability distribution across all the 1652 words in the vocabulary

Merge model structure



Hyper-parameters tuning

- The model was then trained for 30 epochs with the initial learning rate of 0.001 and 3 pictures per batch (batch size).
- After 20 epochs, the learning rate was reduced to 0.0001 and the model was trained on 6 pictures per batch.



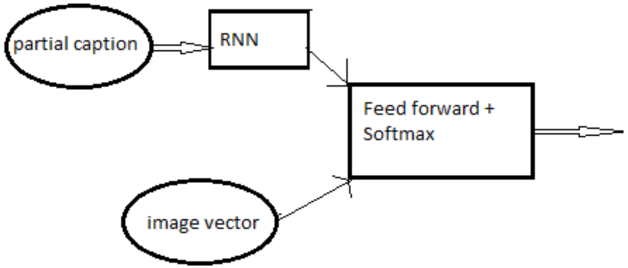
Hyperparameter	**Value**
Learning rate	0.0001
Epochs	30
Batch size	6
Dropout rate	0.5
Embedding size	200
LSTM output size	1652
Optimizer	adam
Loss computation	categorical_crossentropy

Inference Methods

- Sampling
    - Iteratively generate caption one word at a time
    - **Greedily** select the word with the maximum probability
    - Stop the iterations on receiving an ‘**endseq**’ token which means the model thinks that this is the end of the caption or when a maximum **threshold** of the number of words generated by the model is reached.
- Beam Search
    - Iteratively consider the set of the k best sentences up to time t as candidates to generate sentences of size t + 1
    - keep only the resulting best k of them
- Sampling inference method has been used in the project to predict the caption for the test image

Sample iteration using sampling

Input Caption: "startseq"



Probability Distribution generated by the softmax

Word	**Probability**
black	
cat	
endseq	
grass	
is	
on	
road	
sat	
startseq	
the	
walking	
white	

This probability must be maximum.

Predicted word **"the"**

Resulting caption after iteration **1:**

"startseq the"

Evaluation Metrics

- **BLEU**
  - Metric for evaluating a generated sentence to a reference sentence
  - Cumulative BLEU calculate individual n-gram scores at all orders from 1 to n and weight them by calculating the weighted geometric mean

Metrics	Score
Cumulative BLEU-1	**0.438373**
Cumulative BLEU-2	**0.251009**
Cumulative BLEU-3	**0.171954**
Cumulative BLEU-4	**0.079744**

Reference Paper

Show and Tell: A Neural Image Caption Generator

Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3156-3164

Links

A. Youtube link for individual project presentation and project details
https://www.youtube.com/watch?v=y9f2BcCA-Xo&feature=youtu.be

A. Github link for project implementation
https://github.com/kruti-thukral/image_captioning

A. Dataset can be downloaded from https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/
Datasets that need to be downloaded from the above link are as follows
Flickr8k_Dataset.zip
Flickr8k_text.zip

A. Pre-trained Glove embeddings can be downloaded from
https://nlp.stanford.edu/projects/glove/
glove.6B.zip from above link was used in the project