Visual Recognition: Mini project 2

Group-31

Sriram Munagala IMT2020030
Pratham Dandale IMT2020038
Dhanvi Medha IMT2020529

- Problem:

  Design a CNN-LSTM system (preferably in pytorch) that can perform image captioning on flickr8 dataset.Use the features that are extracted by the CNN as an input to the LSTM System.

- Flickr8 dataset:

  The Flickr8 dataset is a publicly available dataset of images collected from the popular photo-sharing website, Flickr. It consists of 8,000 images, each with associated textual tags describing the content of the image.
  The images in the Flickr8 dataset are diverse in terms of content, ranging from landscapes and animals to people and objects.

- Preprocessing

  The preprocessing part of the code focuses on preparing the image captions for further processing.

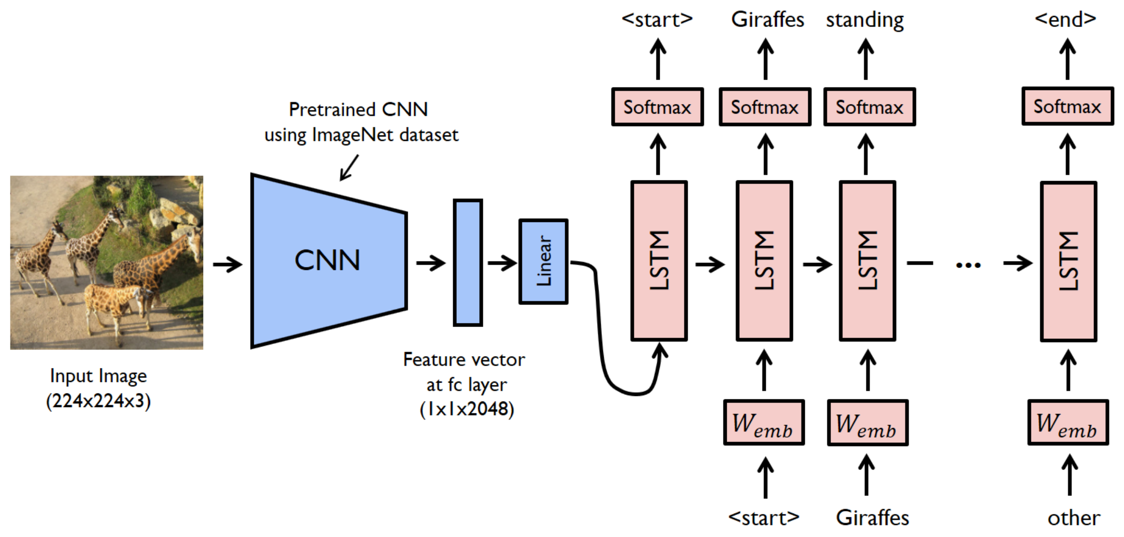
  Reading Captions: The code reads captions from a CSV file that contains image IDs and their corresponding descriptions. It extracts the image IDs and captions from the caption file.

  Descriptor Mapping: The code uses a custom dictionary class called `my_dictionary` to create a mapping between image IDs and their captions. For each image ID, multiple captions are stored as a list in the dictionary. The captions are processed by adding a start token ("<start>") at the beginning and an end token ("<end>") at the end.

The purpose of this preprocessing is to organize the captions for each image, making it easier to access and manipulate the data during the subsequent steps of the code.

- Architecture details: Baseline

LSTM:

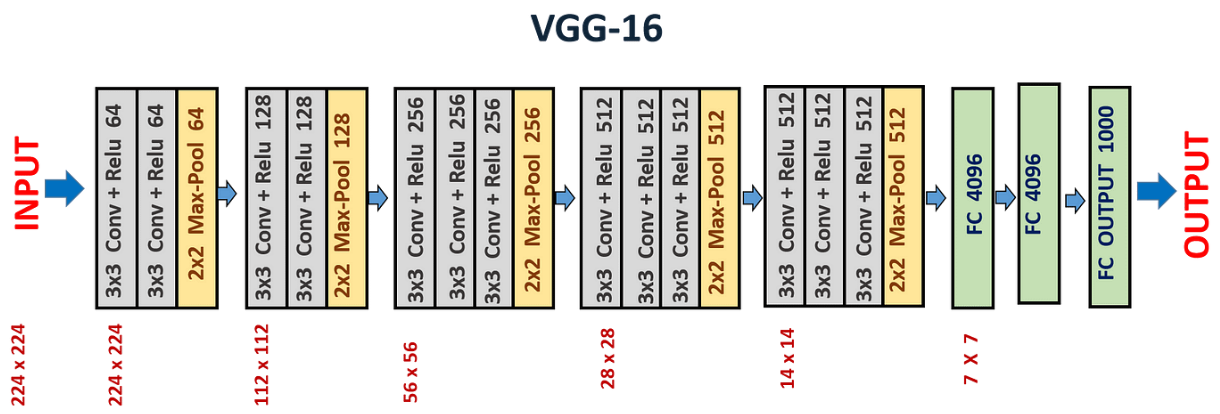


To build our sequential learning model using LSTMs, we combined the preprocessed image data and integer tokens representing captions. The image data was reduced to a 256-dimensional feature vector using a Linear layer with ReLU activation.

Our LSTM architecture takes in the image feature vector and a start-of-sentence token as input. This allows the model to consider both the image context and the initial clue for generating captions. The LSTM uses its hidden state to generate captions sequentially.

During training, we used Categorical Cross Entropy loss to update the model's parameters. We employed the Adam optimizer with a fixed learning rate.

Our LSTM-based architecture leverages image features and word embeddings to generate captions. By combining the image context and start-of-sentence token, the model learns to generate relevant captions based on the hidden state. VGG:



We used VGG-16 as pre trained CNN model to extract feature vector from input images.

Image Feature Layers:

- The input `inputs1` represents the image features extracted using a pre-trained convolutional neural network (CNN) such as VGG16 or ResNet.
- Dropout is applied with a rate of 0.4 to help prevent overfitting.
- The output of the dropout layer is passed through a fully connected layer (`Dense`) with 256 units and ReLU activation.

Sequence Feature Layers:

- The input `inputs2` represents the preprocessed caption sequences.
- An embedding layer is used to convert the input sequences into dense vector representations. It uses an embedding dimension of 256 and masks zero values.
- Dropout is applied with a rate of 0.4 to the embedded sequences.
- The embedded sequences are then passed through an LSTM layer with 256 units. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that can capture sequence dependencies.

Decoder:

- The output of the image feature layers (`fe2`) and the sequence feature layers (`se3`) are combined element-wise using the `add` function.
- The combined features are then passed through a fully connected layer with 256 units and ReLU activation.

Output:

- The output layer is a fully connected layer (`Dense`) with a number of units equal to the vocabulary size. It uses the softmax activation function to generate a probability distribution over the vocabulary.
- The model aims to predict the next word in the caption sequence.

Model Compilation:

- The model is compiled with the categorical cross-entropy loss function and the Adam optimizer.

Training:

- The model is trained for a specified number of epochs using a custom data generator (`data_generator`) to generate batches of training data.
- The generator takes the training data, image features, tokenizer, maximum caption length, vocabulary size, and batch size as input.
- Each epoch consists of steps_per_epoch number of steps to process the entire training data.
- The model is fitted using the generator for each epoch.

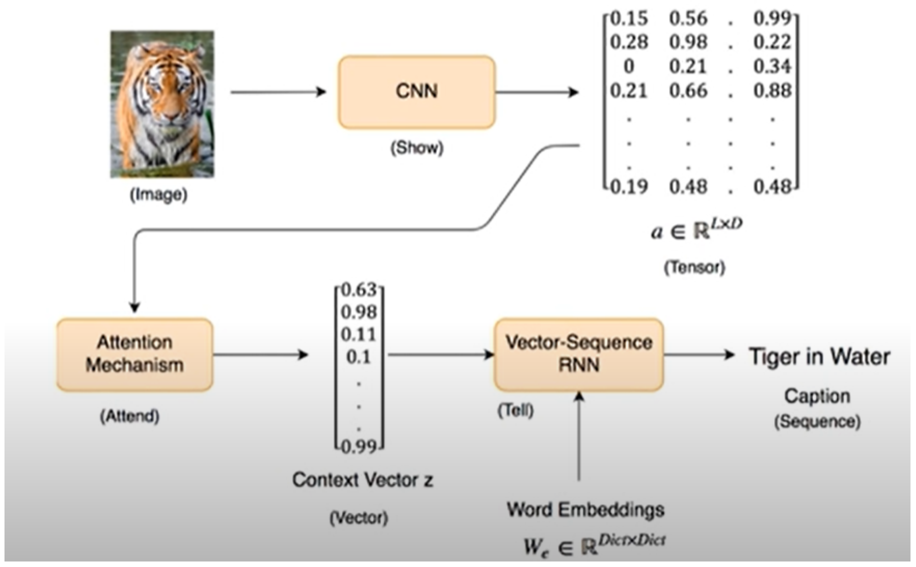
Model Saving:

- After training, the model is saved in the specified location (`WORKING_DIR+'/best_model.h5'`) for future use.

Overall, this architecture combines image features with sequence features and uses an LSTM-based decoder to generate captions for images.

Modified Baseline:

We have used an Attention model with 3 simple linear layers.We transform the image tensor into a specific attention score and use the alpha parameter that we get here to calculate the weights sensor.



Results Analysis:

We have used two scoring methods to evaluate the models:

BLEU(Bilingual Evaluation Understudy) Score: It measures the similarity between the machine-generated translation and one or more reference translations. It calculates the precision of n-grams (contiguous sequences of n words) in the machine translation by comparing them with the reference translations. The BLEU score ranges from 0 to 1, with 1 indicating a perfect match between the translation and the references.

Meteor Score: It measures the quality of a machine-generated translation by considering several factors such as unigram precision, recall, stemming, synonymy, and word order. It combines these factors into a single score using a harmonic mean. METEOR takes into account a wider range of linguistic phenomena and tries to capture fluency and adequacy better than BLEU.

The metric used for calculation loss was categorical cross entropy loss.

Results:

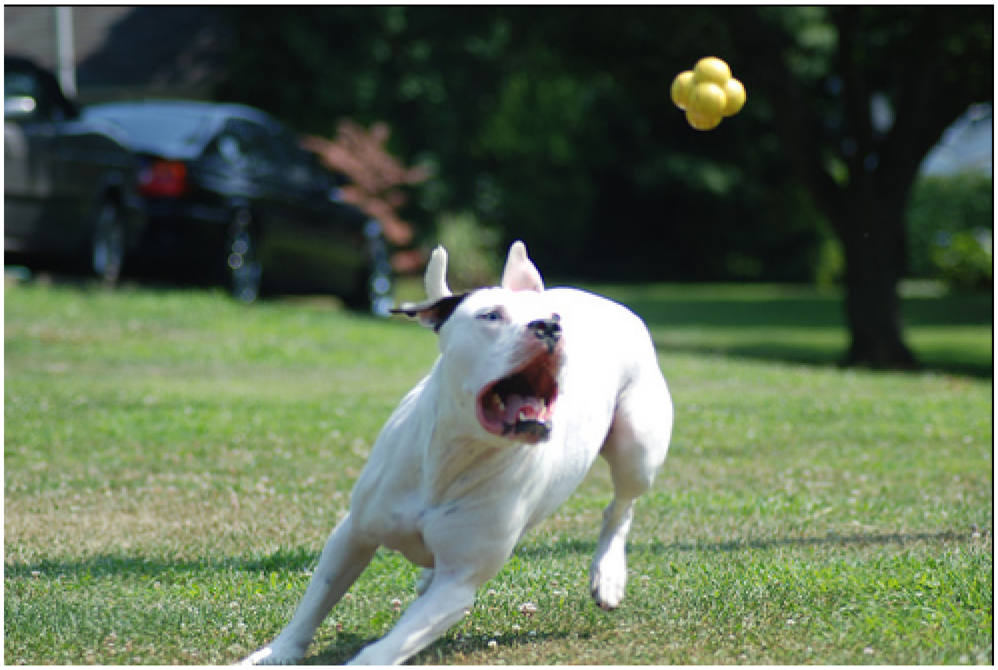
CNN Model	Model	Activation Function	Epochs	BLEU Score	Meteor Score
VGG16	Baseline	RelU	12	BLEU1:0.3158 BLEU2:0.1344 BLEU3:0.0687 BLEU4:0.0199	0.4528
VGG16	Baseline	RelU	20	BLEU1:0.3187 BLEU2:0.1412 BLEU3:0.0552 BLEU4:0.0159	0.4528

VGG16	Modified Baseline	Tanh	20	BLEU1:0.3144 BLEU2:0.1513 BLEU3:0.0711 BLEU4:0.0155	0.459

Increase Epochs results in slightly better results and both the models performed comparatively but modified baseline outperformed the baseline.

Increasing the LSTM Layers will give the model more parameters to work with and more features can be found.

Example-1:
Actual Description:
startseq dog prepares to catch thrown object in field with nearby cars endseq
startseq white dog is about to catch yellow ball in its mouth endseq
startseq white dog is about to catch yellow dog toy endseq
startseq white dog is ready to catch yellow ball flying through the air endseq
startseq white dog running after yellow ball endseq



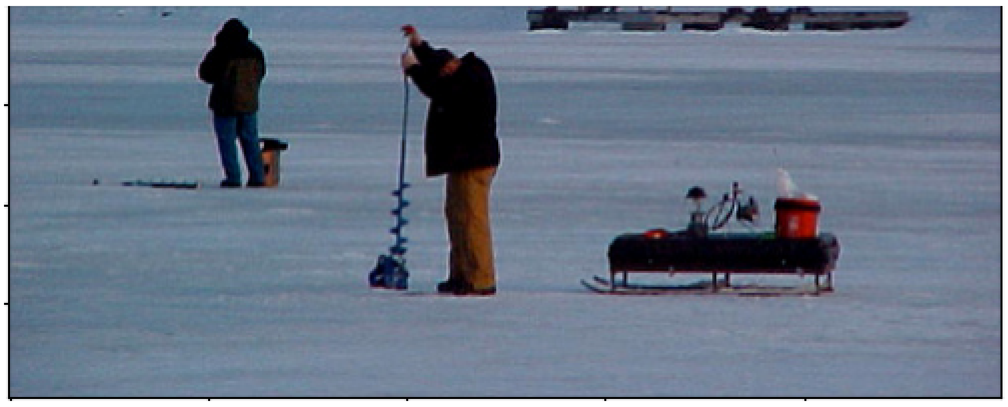
Predicted Sequence:

startseq white dog with orange collar is running in park endseq

Subjective comparision: In this example,the model successfully predicted that there is a dog running in the scene but failed to predict the yellow toy it is chasing after instead tried to fill it in as an orange collar even though that prediction is wrong.

Example-2:

Actual Description:
startseq man drilling hole in the ice endseq
startseq man is drilling through the frozen ice of pond endseq
startseq person in the snow drilling hole in the ice endseq
startseq person standing on frozen lake endseq
startseq two men are ice fishing endseq

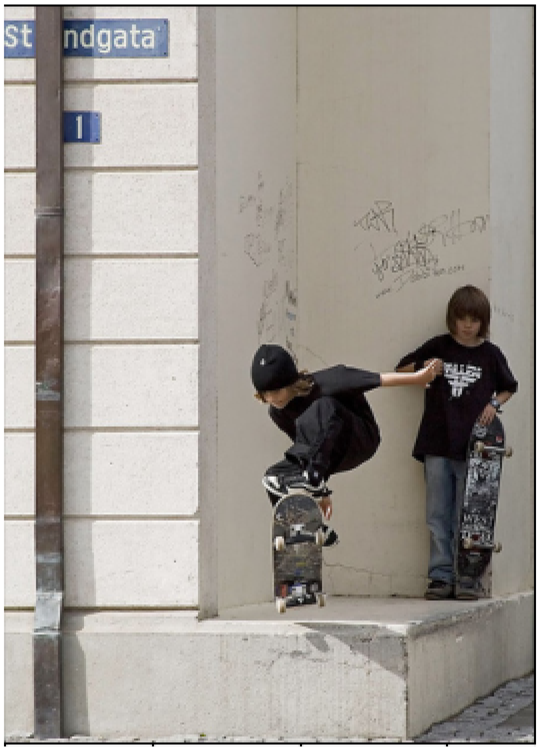


Predicted Sequence:

startseq man drilling hole in the ice endseq

Subjective comparision:The model has detected the man drilling in the ice successfully but failed to the detect the man standing in the background.

Example-3:
Actual Description:
startseq boy takes jump on his skateboard while another boy with skateboard watches endseq
startseq child is performing skateboard trick while another child with skateboard leans on wall endseq
startseq little boy skateboarder is doing trick on his board while another young skateboarder watches endseq
startseq young boy skateboarder jumping on platform on skateboard endseq
startseq two skateboarders endseq



Predicted Sequence:

startseq man in black jacket is doing skateboard trick on brick wall endseq

Subjective comparision: In this example,the model successfully detected that there is aboy standing on the brick wall but missed to detect the second boy standing behind him with another skateboard.

References:

- https://www.youtube.com/watch?v=y2BaTt1fxJU&t=730s&pp=ygUabHN0bSBjbm4gaW1hZ2UgY2FwdGlvbmluZyA%3D

- https://www.youtube.com/watch?v=y2BaTt1fxJU&t=730s

- https://paperswithcode.com/dataset/flickr-8k

- https://medium.com/analytics-vidhya/cnn-lstm-architecture-and-image-captioning-2351fc18e8d7