The aim of the project is given a dataset of images in which English captions are embedded in the image itself. Unfortunately, these captions may or may not correctly correspond to the image in which it is embedded i.e. in some cases it has been randomly embedded on top of the background image and in the remaining examples it’s correctly embedded on top of correct image. The task is to predict the text embedded into the image (irrespective of the background image itself).

We will develop a model based on the Encoder-Decoder Architecture that can take an image and directly generate a sentence description of the image. The model that we have implemented is based on a combination of a Convolutional Neural Network (CNN - working as an image encoder) and a Recurrent Neural Network (RNN – which functions as an image decoder). The encoder would sample the salient features or likely descriptions from images by a simple feed-forward process to cull out of clutter in an image. Subsequently, the decoder uses this information and is conditioned to understand the important descriptions of the image and tries to learn how to describe what the model ”sees”.

• **Non-Competitive part**: The model has to be developed from scratch and trained on the given dataset - Images and a captions per image.

– Encoder: We will design a CNN based Encoder that handles variable sized images.

– Decoder: We will design a LSTM based Decoder for generating captions given for the image input.

– Implementation Details: We will use Cross Entropy Loss and we trained the decoder. Beam Search ,which uses a few top token with highest probabilities to construct a tree of possibilities and generates the most likely captions for the images.

The loss starts from 9 and declines to 4 at which gets stuck.

Our non competitive part encoder’s architecture is similar to VGG19 and decoder’s architecture is similar to LSTM.

Encoder’s Implementation-

Input to CNN is of the size 224\*224\*3 which is cropped randomly to downscale to 112\*112\*3 and the embedding size is 256 which is empirically determined to be fed to LSTM.

Decoder implementation -All the caption words are converted to tokens and fed to the embedding layer which converts to a fixed length vector to be fed into RNN (LSTM). We have used PyTorch library to pack the sequences in a batch to send it together to the LSTM layer

Competative part:

Encoder is VGG-16

Decoder is LSTM