VIDEO CAPTIONING USING CONVOLUTIONAL RECURRENT NEURAL NETWORKS

DHARANIDHARAN RAMASAMY KARUPPANASAMY
Department of Electrical Engineering & Computer Science
Syracuse University
Syracuse, NY
dramasam@syr.edu

***Abstract*—The aim of this project is to reproduce a model that generates natural language descriptions when an image/video is passed as an input. This idea of generating natural language description is based on an existing model that produces texts related to the given image/video with the use of Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) on Flickr 8k dataset. A much deeper and dense network VGG 19 is used in this experiment than using VGG 16 that has been already used to measure the accuracy of the experiment. The accuracy of the output generated is calculated using the BLEU score.**

Keywords—CNN, RNN, VGG19, BLEU score

I. INTRODUCTION

In the past, a remarkable amount of research work has been held in image classification and video classification tasks. Humans can easily understand an image or video given and can describe the same in their own way. Attempts to describe the photo or a video has been recent years of research interest. Unfortunately, human vocabulary model can generate words even when they haven't seen it before but for computers, it all depends upon a fixed range of inputs and it all depends on those inputs to generate the output. Without a word being shown to the system, the system can't generate the best possible description for the image or video. However various models have been generated to address this problem. The limitation is that the system can't produce a variety when a new image is being inputted. It will be able to produce descriptions only based on the descriptions that have been inputted for training so far. In this project we experiment the accuracy of the existing model using a different network for extracting the features of the image/video for generating a description for it. We provide a detailed review of existing models.

II. RELATED WORKS

Deep Visual-Semantic Alignments for Generating Image Descriptions: In this project, a deep neural network has been implemented to learn and understand the latent alignment between the sentences and the particular area of the image that they are describing. This idea has been developed form the work of associating words and images based on their sematic embeddings and is also based on internal modal alignment of the text and the image after being depend decomposed. This model doesn't depend on a fixed length and produces more meaningful words.

Deep learning for video classification and captioning: In this project, there are two areas of research focused on i.Video classification and the other is Video captioning. A deep learning model is being built to do both purposes. Video captioning being a vibrant area of research now a days, more research works are evolving around this topic.

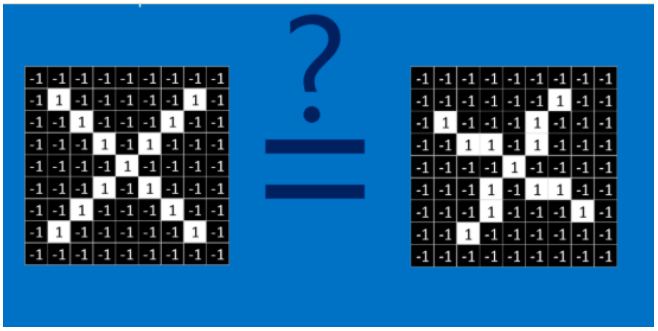
III. BASIC DEEP LEARNING MODULES

In this section we will analyze some of the basic deep learning modules implements so for particularly to tackle with video analysis problems.

Convolutional Neural Networks: It was Fukushima who first proposed the first model of connectivity between the neurons called "neoconitron" in 1980's. His work was on basis of McCulloch-Pitts model and visual perception mechanisms of animals. He applied the neurons randomly on the previous layer of network with similar parameters which became the model or predecessor for Convolutional Neural Networks. With this model as a base LeCun developed the modern framework of CNN which is nothing but LeNet -5 which was used for hand writing recognition. Most of the times when a new idea has been implemented in Neural networks the work will be related to CNN's. That's how CNN as a deep neural network has evolved in the past days and have become an integral part of deep

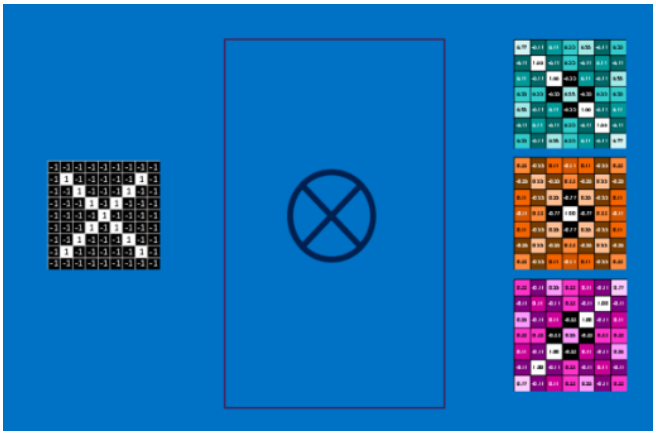
neural network field. They have started outperforming humans in some image classification tasks. Now that we saw how the CNN's evolved, we will have a look into how CNN's work. To illustrate how CNN'S work let's consider a simple example of predicting whether an image is X or O.

Convolution: Computers can't see the way how humans interpret images. For them its all a matrix value. If suppose the image of X is little different and if that image is passed to the model, it finds it difficult to classify the image because the matrix value of the slightly modified X would be different from that the model had learned already.

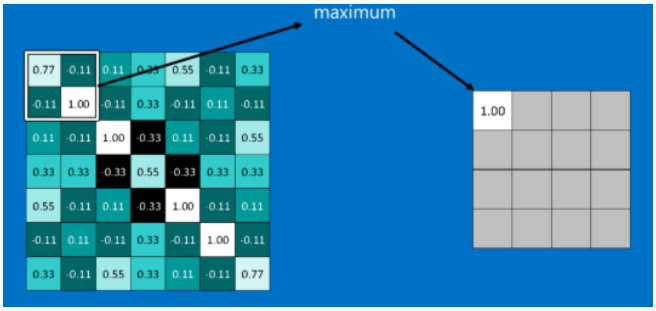


But humans can easily interpret that the given image is X. This would be the naïve approach if we are trying to solve the problem using a computer along with this problem. This is where CNN's outperform any basic model even if the image is rotated, shrunken or deformed. CNN takes small parts of the original images like piece by piece and compares these small images with the original image to find patterns with the original image and the smaller one. These smaller pieces of images are called filters. CNN works better while comparing these filters with the original image than comparing entire image. When given a new image CNN compares these filters with the entire image because CNN doesn't know where these images are present in the original image. The math behind this convolution working is to multiply all the values of the values of the pixel with the values of the original image where it matches. Sum up all these values and divide them by the count which obviously returns 1 and -1. It returns 1 if the pattern is matching with the original image and return -1 if the pattern doesn't match. This kind of matching is done for all our filters and this process is called convolution. It produces a 2D matrix where the 1,-1 and 0's would be filled. This turns out to be the convolution layer in practice with then CNN's.

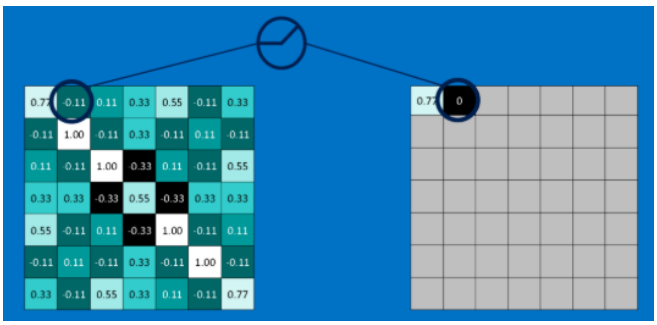
Pooling: Another useful or technique that the CNN uses is the pooling. Pooling is the technique of reducing the large image into smaller ones without losing any information or pertaining only the important feature of the image. Pooling is done by sliding a window generally of size 2 or 3 with a step size of 2.



The maximum value at each window is takes so that the important feature of the image is preserved. Pooling layer performs this operation and the result of pooling layer doesn't reduce the number of images but reduces the size of each image which reduces a lot of computational load to be performed.

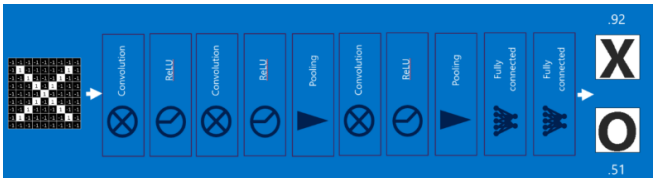


Rectified Linear Units: This is simple but an important layer. It focuses on making the negative values to 0. Basically, it does normalization of the input matrix that it has and thus changing the negative values to 0.



So basically, a single layer CNN has a convolution layer, pooling layer and ReLU unit. Building more layers reduces the size of an image to a large

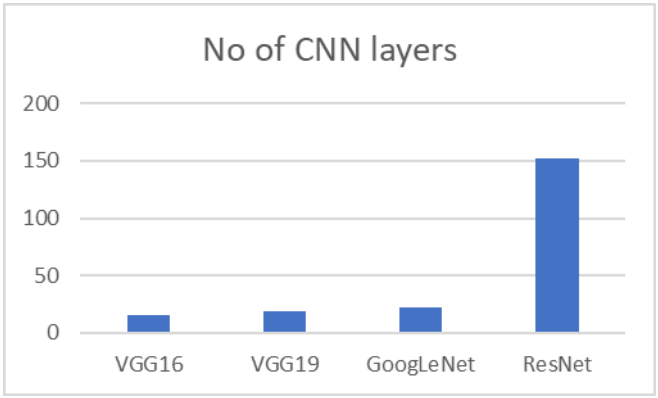
extent. This input matrix is converted to a single column and is given as input to a fully connected network which actually gives votes for the input thus predicting if the input given is X or O.



This is a multi-layered CNN.

Types of Deep Networks:

As we discussed earlier, LeNet-5 is a multi-layered network with 5 layers. LeNet-5 did not perform well when the number of output labels increased along with the computation load. To overcome this problem, Hinton developed a neural network called Deep Belief Neural network where each layer of the neural network is trained for an unsupervised work environment. This was ground for many of the complex neural networks built now. AlexNet has 5 convolutional neural networks in it with 3 fully connected layers after that. The difference between the two networks is that ReLu had been added and dropouts had been added in the ALexNet to reduce overfitting while training the network. This model AlexNet led to the development of VGG networks, GoogLeNet and ResNet.



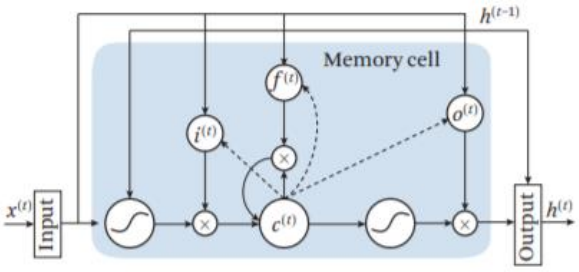
Recurrent Neural Networks: Sequence labelling requires cycles in the neural network and all the CNN's ae feed forward neural network. To address this issue recurrent neural networks have been introduced that has cycles to their own network. Since RNN's allow a cyclic connection there is a memory unit involved along with this which can store the previous input. But these memory units aren't big enough to hold a greater number of previous units.

$$\mathbf{h}^{(t)} = \sigma\left(W_x\mathbf{x}^{(t)} + W_h\mathbf{h}^{(t-1)} + \mathbf{b}_h\right)$$

$$\mathbf{z}^{(t)} = \text{softmax}\left(W_z\mathbf{h}^{(t)} + \mathbf{b}_z\right)$$

Here b_h and b_z are biases. The problems in RNN is vanishing and the other is that the gradient will explode. The weights will explode during backpropagation. To solve this issue Long-Short Term Memory (LSTM) networks are used.

Long short-term memory (LSTM): These types of networks are same as RNN, but the difference is that these can store long sequences than RNN's. The information flow into the network and out of the network is governed by the input, output and forget gates. The block diagram for LSTM is shown below:



With the help of memory units and gates, LSTM can produce long term memory of the given inputs and outputs. This type of network is now being used in most of the video analysis researches.

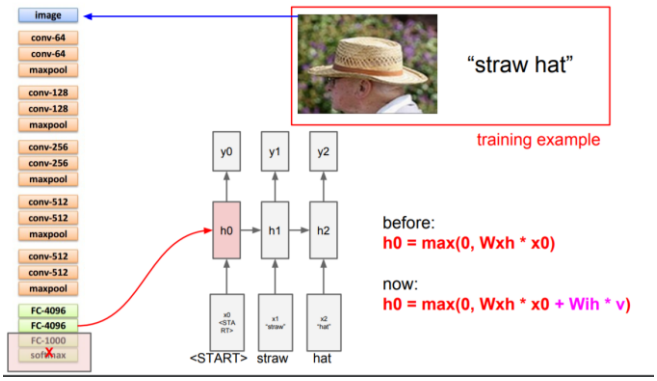
IV. Deep Learning Model for Image Captioning

In this section we will discuss about how we are going to develop a model for image captioning using VGG network. For the task of captioning the image, we use the pre-trained VGG 19 network as a convolutional neural network part. This convolutional neural network is used to extract features from the images. Keras provides an inbuilt function to import the VGG19 network. The model can be used for image captioning purposes, but the problem is each image must be sent to the network which involves a large amount of computational load to be performed. To reduce this computational load, we will first convert all images available into photo features and then pass this feature along with train descriptions to our model. These photo features are one dimensional vector of size 4096.This is the preprocessing step

done with image data. The preprocessing step that is involved with text data are

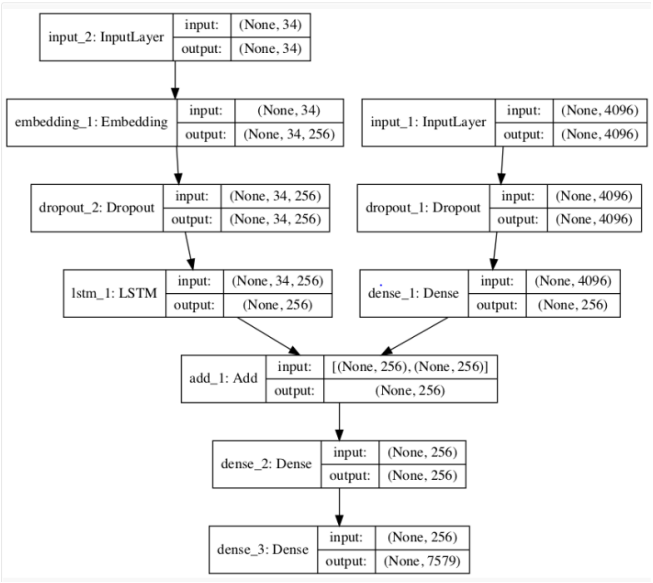
i. Remove punctuations
ii. Remove single word characters
iii. Remove special characters and numbers.

These are done to train the data faster because smaller the dictionary size, smaller is the computational load and so the computation will run faster. Once these steps are done, the data for text description is cleaned. The VGG19 network is used for image classification but here we are considering about the phot features. These photo features are the output of fully connected network before going to the classification. So in this network we actually remove the last layer from VGG19 because we need to compute the photo features to generate the photo description.



As shown in the diagram, the input images are pre-processed and sent as input to the recurrent neural networks. This recurrent neural network doesn't know how to start with describing the sentence. This can easily be done by adding additional weight to the h0 function and the weights get updated automatically. The dropout layer is added to remove variations from the input and produce the output. Each word to image sequence is matched. Now the model is fit and is trained on the image description and the image data through image tokenizer.

To measure the accuracy of the model we use a concept called BLEU value. If the value of BLEU score is closing to 1, it means that the model created so far is performing better. If the BLEU score is near to 0 then it means that the model is not performing good. We can combine the result of the individual frames to produce a video caption for the entire video. Here we got a BLEU score of 0.45



V. REFERENCES

- [1]All coding part is done with the help of this following tutorial on Image classification https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/
- [2]where_to_put_the_image_in_an_image_caption_generator.pdf
- [3]https://www.youtube.com/watch?v=yk6XDFm3J2c&t=572s
- [4]https://www.youtube.com/watch?v=FmpDIaiMIeA
- [5]http://brohrer.github.io/how_convolutional_neural_networks_work.html
- [6]https://cs.stanford.edu/people/karpathy/sfmltalk.pdf
- [7]https://machinelearningmastery.com/prepare-photo-caption-dataset-training-deep-learning-model/
- [8]https://en.wikipedia.org/wiki/BLEU
- [9]https://arxiv.org/abs/1411.4555
- [10]https://arxiv.org/abs/1502.03044
- [11]https://arxiv.org/abs/1703.09137
- [12]https://arxiv.org/abs/1708.02043
- [13]https://arxiv.org/abs/1601.03896
- [14]https://keras.io/models/model/
- [15]https://berkeley-deep-learning.github.io/cs294-131-s17/slides/saenko-talk.pdf
- [16]https://www.researchgate.net/publication/283335531_Video_Paragraph_Captioni

ng_Using_Hierarchical_Recurrent_Neural_Networks