A Major Project-II Report

On

EVOLUTION OF AUTOMATIC VISUAL DESCRIPTION TECHNIQUES - A SURVEY

Submitted in Partial fulfilment of the Requirement for the Degree of

Master of Technology

in

Computer Science and Engineering

Submitted By

Arka Bhowmik

2K17/CSE/04

Under the Guidance of

Mr. Sanjay Kumar

(Assistant Professor)

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahabad Daulatpur, Main Bawana Road,Delhi-110042

June 2019

CERTIFICATE

This is to certify that Project Report entitled ” **Evolution Of Automatic Visual Description Techniques- A Survey ”** submitted by **Arka Bhowmik** (2K17/CSE/04) in partial fulfilment of the requirement for the award of degree Master of Technology (Computer Science and Engineering) is a record of the original work carried out by him under my supervision.

Project Guide
Mr. Sanjay Kumar
Assistant Professor
Department of Computer Science & Engineering
Delhi Technological University

DECLARATION

I hereby declare that the Major Project-II work entitled **” Evolution Of Automatic Visual Description Techniques- A Survey ”** which is being submitted to Delhi Technological University, in partial fulfilment of requirements for the award of the degree of Master of Technology (Computer Science and Engineering) is a bona fide report of Major Project-II carried out by me. I have not submitted the matter embodied in this dissertation for the award of any other degree or diploma.

Arka Bhowmik
2K17/CSE/04

ACKNOWLEDGEMENT

First of all, I would like to express my deep sense of respect and gratitude to my project supervisor Mr. Sanjay Kumar for providing the opportunity of carrying out this project and being the guiding force behind this work. I am deeply indebted to him for the support, advice and encouragement he provided without which the project could not have been a success.

Secondly, I am grateful to Dr.Rajni Jindal, HOD, Computer Science & Engineering Department, DTU for her immense support. I would also like to acknowledge Delhi Technological University library and staff for providing the right academic resources and environment for this work to be carried out.

Last but not the least I would like to express sincere gratitude to my parents and friends for constantly encouraging me during the completion of work.

Arka Bhowmik
Roll No – 2K17/CSE/04
M. Tech(Computer Science & Engineering)
Delhi Technological University

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

1. RNN:	Recurrent Neural Networks
2. LSTM:	Long Short-Term Memory
3. Bi-LSTM:	Bidirectional LSTM
4. CNN:	Convolutional Neural Networks
5. BLEU:	Bilingual Evaluation Understudy Score
6. CIDEr:	Consensus-based Image Description Evaluation
7. ROGUE:	Recall-Oriented Understudy for Gisting Evaluation
8. TF-IDF:	Term Frequency Inverse Document Frequency
9. METEOR:	Metric for Evaluation of Translation with Explicit Ordering
10. SPICE:	Semantic Propositional Image Caption Evaluation
11. ResNet152:	Residual Network 152 with layers
12. VGG16:	Visual Geometry Group with 16 Layers
13. YOLO:	You Only Look Once
14. SAM:	Saliency Attentive Model
15. MSVD:	Microsoft research Video Description
16. MSCOCO:	Microsoft Common Objects in Context
17. SALICON:	Saliency in Context
18. OOC:	Out Of Context Image Dataset
19. MVAD:	The Montreal Video Annotation Dataset
20. LDA:	Latent Dirichlet Allocation
21. MLP:	Multi-Layer Perceptron
22. GAN:	General Adversarial Networks
23. GLOVE:	Global Vectors (for Word Representation)
24. GNIC:	GoogleNet Neural Image Captioner
25. HRNN:	Hierarchical RNN
26. HOG:	Histograms of Oriented Graphs
27. AVR:	Adaptive Visual Replacement
28. DAP:	Deep Action Proposals
29. JSON:	Java Script Object Notation
30. B1, 2, 3,4:	BLEU-1, BLUE-2, BLEU-3, BLUE-4
31. M:	METEOR
32. C:	CIDER

ABSTRACT

Describing the contents and activities in an image or video in semantically and syntactically correct sentences is known as captioning. Automated captioning has been one of the most competitive major trends in present day research with new sophisticated models being discovered every day. Captioning models require intense training and perform intense complex calculations before successfully generating a caption and hence, takes considerable amount of time even in machines with high specifications. In this survey, we go through the recent state-of-the-art advancements in automatic image and video description methodologies using deep neural networks and summarize the important concepts that can be inferred from the researches. The summarization has been done with detailed analysis of methodologies used along with explanation of referenced context. Along with detailed description of available datasets and methodologies. The focus of our research lies in techniques which are able to optimize existing concepts as well as incorporate new methods of visual attention to generate captions. This survey emphasizes on the importance of applicability and effectiveness of existing works in real life applications and highlights those computationally feasible and optimized techniques which can be supported in multiple devices ,including lightweight devices like smartphones.

Keywords: Image Captioning, Video Captioning, Activity Recognition, Deep Learning, Convolutional Neural Networks, Recurrent Neural Networks

CHAPTER 1: INTRODUCTION

1.1 Role of Deep Learning In Automated Caption Generation

Deep Learning has outperformed almost all other types of machine learning methods, training models with an enormous amount of data and getting state of the art results. One of the very popular usages of deep learning is image and video captioning. Captioning refers to describing the content of visual sources like image or video, in well framed sentences. Social networks like Facebook, Instagram and Twitter have gained global popularity as a platform for communication as well as sharing media . The shared images and videos are rich sources of information regarding the person's activities, location and appearance and this information can be leveraged for studying the interests of people for generating and making available content of interest to them. Surveying online media content by human workforces happens to be both computationally and economically infeasible. Such scenarios are great examples of usage of image[69] and video captioning[58,59] , as they can extract comprehensive information from such media , in human understandable languages. Apart from these, captioning models find use in accurately searching for image and video content online using image indexing or CBIR(Content Based Image Retrieval) and explaining activities inside an image and video without requiring to go through it. Captioners also play a significant aid for blind men and people with visual impairment , who are unable to perceive visual content by themselves. Visual description techniques[71] constitute a wide area of research, not only constrained to captioning, but also object detection and action recognition. The originality of captioning lies in action detection and recognition[57,75,76] where usually models are designed just to classify activities or actions in images and videos based on the poses and environment in consideration. Traditional machine learning techniques of detecting such features in image frames were to use computer vision techniques like HOG (Histogram of Gradients)[56],HOF[60] (Histogram of Optical Flow), Scale-Invariant Feature Transform (SIFT) which identify orientation and trajectories of pixels in motion and then apply classification algorithms like SVM to identify the gestures. These features have later been put to use in attention based captioning models. Plain vanilla Image Captioning models consist of two main steps to generate a caption. In the first step, the objects in the image are identified in different segments of the image and in the second part, a language model is used to frame the objects in a well-structured sentence that describes the image content. The first step is performed using a convoluted neural network that labels the image objects separately, with a particular confidence factor for each label. These convolutional networks are rather sophisticated and of great depth (depth referring to the number of hidden layers in the network) and need to be trained with a variety of images identify objects in an image accurately. The framing of sentences is performed by an RNN(Recurrent Neural Network) which takes in one word to be used as the starting of the sentence and then generates every next word based on the previous generations and the current object being focused in the image. The RNN has to be trained using human labelled captions of images (often 4 to 5 captions per image). RNN/LSTMs (Long Short Term Memory) for image captioning can be of two types: inject and merge. The inject model focuses both on the linguistic and perceptual features during training and hence requires both image features and generated word embeddings to be fed in every iteration into the LSTM . However in the merge architecture, in very iteration, the LSTM is trained only using word embeddings /labels and at a later stage, image features are merged into a dense layer along with the LSTM output and passed into a SoftMax function to provide the output caption. In the paper[55] “What is the role of an RNN In an Image Caption Generator”, the authors have performed an experimental study between the two architectures of image caption generator and found that , surprisingly the merge model has performed better, with the same basic training (without using hyperparameters). Figure 1 represents the two models. Dense layer is a fully connected layer with bias. ‘v’ implies the output dimension of the dense layer is same as vocabulary size. All other intermediate layers are of varying dimensions which depend on experimentation.

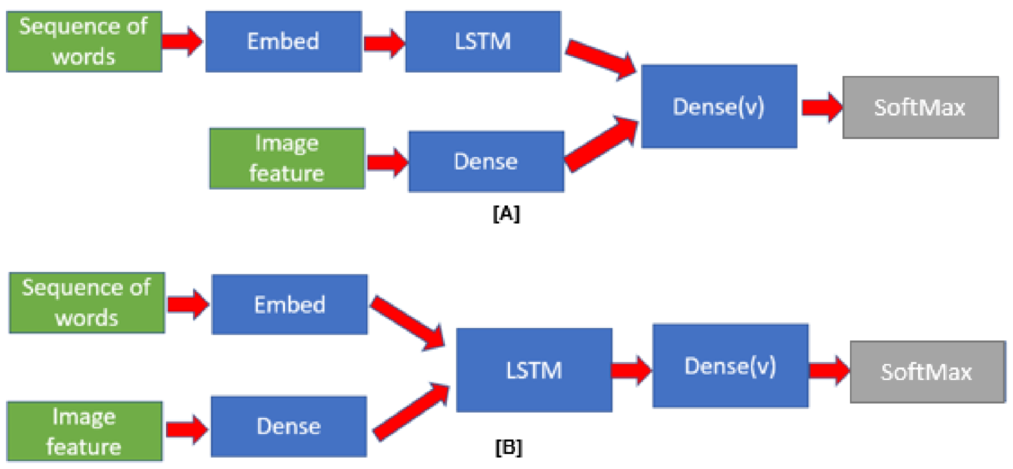


Fig 1: [A]The Merge Architecture [B]The inject architecture

However in rare cases inject models have outperformed merge models , with increase in number of states and training dataset. Due to better performance to size ratio, merge models have gained more popularity.

1.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) specialize in classifying images. Images are of 3 dimensions (height, width, number of channels). While first two dimensions correspond to the image resolution, the third dimension represents the number of channels (RGB) or pixel intensity values for each of the colors red, green and blue. Usually images of reduced dimensions are fed into the networks to prevent underfitting and also to avoid huge processing time. But even so, an image of size 224x224x3 thus if flattened into a 1-d vector would yield an input vector of length 150528 which is too huge to be fed into a normal neural network. CNN contain multiple layers of convolution, pooling and activation which in every instance simplifies our image into a more connected network without loss of information. At the end we have a fully connected network which is a normal neural network taking in a flattened 1-dimensional vector as input and classifying the image in its output.

1.2.1 Convolutional Layer:

The convolutional layer uses a filter/ kernel of small dimension like 3x3 to run over the whole image in fixed gaps of pixels called strides, computing the dot product sum of all pixels which lie in the kernel window. Then this sum is copied to a new matrix of reduced dimension. Thus, we get a comprehensive, yet reduced dimension feature of our image in a convolution layer. Each filter is associated with identifying a different type of feature. For a different level, a different filter may be used to highlight some other feature of the image, and this filter may have a different size. For example, one layer can be responsible for selecting image features based on shapes and edges. Another layer may have filters which select features based on the color intensities of the image. Strides define the number of pixels by which we can shift the filter to focus of a new set of pixels. Its value ranges from 1 to 3 usually. Having a better overlap among filters are better for preventing loss of image context. But this also increases training time and output volume. Depth refers to the number of filters that we want to use. Every filter adds to the depth of the generated output. So, if we use 3 filters, the output depth for that layer will become 3. The output volume from each layer depends on a set of hyperparameters namely: depth, stride and padding. One kernel is dedicated for each channel in that image. A 9x9 RGB image with 3, 3x3 kernels at stride 1 (per channel), will produce an RGB feature of size 7x7x3.

1.2.2 Pooling Layer:

The pooling layer is an untrainable layer. It operates a small kernel on the image at fixed stride, to pick the pixel with maximum intensity in the window and discard other pixels. In the end, the result will consist of a matrix of reduced dimension. The significance of such a layer is to reduce the dimension of the feature image. The image features are often sparse in nature and some cells might not contain features that are of use for classification. The pooling layer discards those cells and keeps important ones only.

1.2.3 RELU:

The RELU activation function is an activation function which sets all negative values to zero and keeps other values intact. Usually Convolution or Pooling layers are followed by a RELU activation.

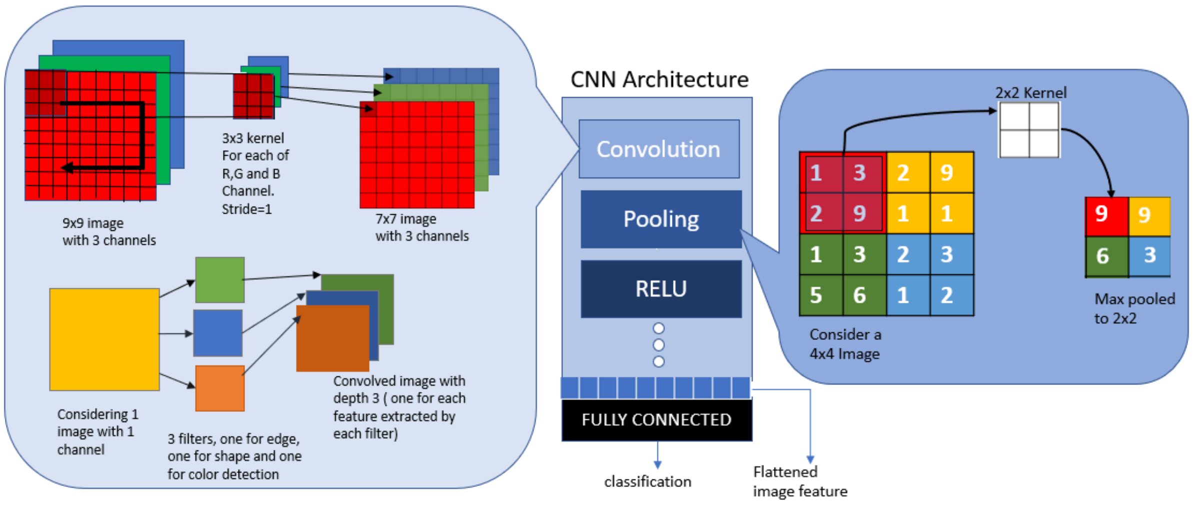

9x9 image with 3 channels
3x3 kernel For each of R,G and B Channel. Stride=1
7x7 image with 3 channels
Considering 1 image with 1 channel
3 filters, one for edge, one for shape and one for color detection
Convolved image with depth 3 (one for each feature extracted by each filter)
CNN Architecture
Convolution
Pooling
RELU
FULLY CONNECTED
classification
Flattened image feature
2x2 Kernel
1 3 2 9
2 9 1 1
1 3 2 3
5 6 1 2
Consider a 4x4 Image
9 9
6 3
Max pooled to 2x2

Fig 2: Components of a CNN

When images are not of acceptable dimensions, padding is necessary before sending it to further layers in the network, to achieve the specified dimension. In such cases padding with zeros is done. Usually sophisticated CNNs consist of various layers in their architecture and can predict image content of a huge number of classes.

Table 1 shows the performance of popular deep CNNs, which are used to classify everyday objects. The Top-1 and Top-5 Scores specify the correct prediction of objects from their images, by choosing the top 1 and top 5 prediction proposals, on testing with the ImageNet validation dataset.

Table 1: Comparative Analysis of Contemporary CNNs

Model Name	**Layers**	**Trainable Params**	**Size in MB**	**Year**	**Top-1 Score**	**Top-5 Score**
VGG16 [42]	23	138357544	528	2014	0.713	0.901
VGG19[42]	26	143667240	549	2014	0.713	0.900
Resnet101[43]	101	44707176	171	2016	0.764	0.928
Resnet152[43]	152	60419944	232	2016	0.766	0.931
InceptionV3[44]	159	23851784	92	2016	0.779	0.937
MobileNetV2[45,46]	88	3538984	16	2018	0.713	0.901

1.3 Recurrent Neural Networks:

Sequence driven data have a high amount of dependence of past and future output sequences. These outputs, hence can be seen as a function of time and previous outputs. To preserve the information of a previous sequence, we need some sort of memory and this is where Recurrent Neural Networks come to picture. Recurrent Neural Networks (RNNs) have "neural memory". They read inputs (like words) one at a time, and remember the context through the hidden layer activations that get passed from one time-step to the next. This allows a uni-directional RNN to take information from the past to process later inputs. Figure 3 shows a basic RNN structure. Here $x^{\langle t \rangle}$ is the input x at the tth time-step and $y^{\langle t \rangle}$ is the output y at the tth time-step which is predicted using pervious outputs i.e. $y^{\langle 1 \rangle}, y^{\langle 2 \rangle}, y^{\langle 3 \rangle} \dots y^{\langle t-1 \rangle}$. $a^{\langle t \rangle}$ is the activation for the tth timestep.

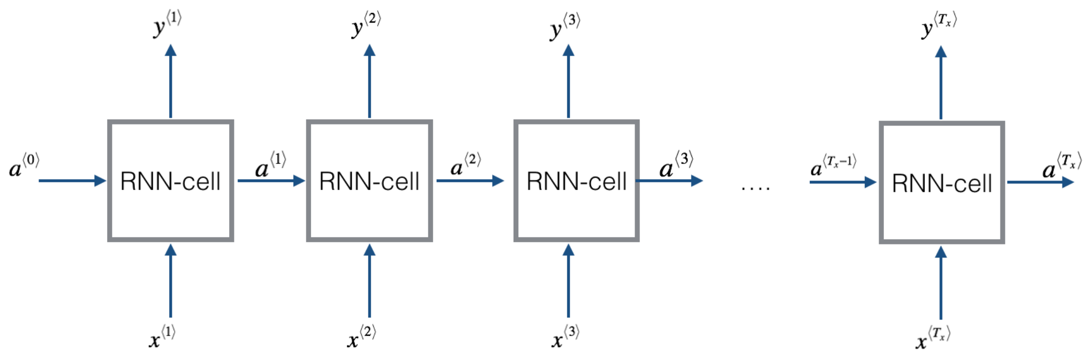
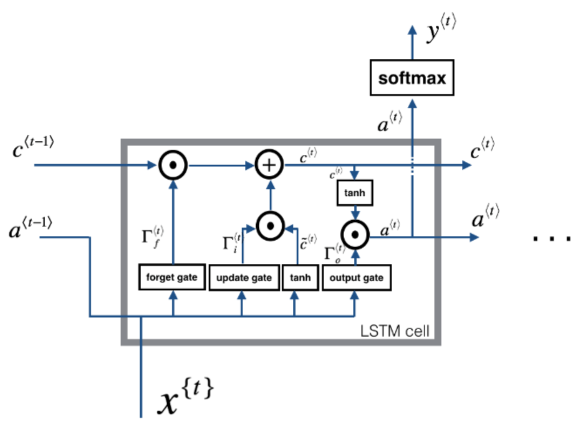


Fig 3: A generic RNN

One major drawback of ordinary RNNs is that in very later sequences, they tend to forget values of the initial sequences and often loose context. Also they are a victim of the vanishing gradient problem. These two things are easily tackled by modifying the RNN into an LSTM (Long Short-Term Memory). An LSTM [68] operates on gated memory, having 3 memory cells/ gates namely the update, output and forget gate. Figure 4 describes an LSTM cell. $c^{\langle t \rangle}$ is the memory cell value at tth timestep. W_i implies the weight matrix for the module i where i can belong to a hidden layer or any of the gates.



$$\Gamma_f^{\langle t\rangle} = \sigma(W_f[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_f)$$
$$\Gamma_u^{\langle t\rangle} = \sigma(W_u[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_u)$$
$$\tilde{c}^{\langle t\rangle} = \tanh(W_C[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_C)$$
$$c^{\langle t\rangle} = \Gamma_f^{\langle t\rangle} \circ c^{\langle t-1\rangle} + \Gamma_u^{\langle t\rangle} \circ \tilde{c}^{\langle t\rangle}$$
$$\Gamma_o^{\langle t\rangle} = \sigma(W_o[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_o)$$
$$a^{\langle t\rangle} = \Gamma_o^{\langle t\rangle} \circ \tanh(c^{\langle t\rangle})$$

Fig 4: Single LSTM cell

1.3.1 Forget gate:

Assuming we are reading words in a piece of text, and want use an LSTM to keep track of grammatical structures, such as whether the subject is singular or plural. If the subject changes from a singular word to a plural word, we need to find a way to get rid of our previously stored memory value of the singular/plural state. In an LSTM, the forget gate lets us do this:

$$\Gamma\langle t\rangle f=\sigma(Wf[a\langle t-1\rangle,x\langle t\rangle]+bf) \qquad (1)$$

Here, W_f are weights that govern the forget gate's behavior. We concatenate $[a_{\langle t-1\rangle},x_{\langle t\rangle}]$ and multiply by W_f. The equation above results in a vector $\Gamma_{\langle t\rangle}{}^{f}$ with values between 0 and 1. This forget gate vector will be multiplied element-wise by the previous cell state $c_{\langle t-1\rangle}$. So, if one of the values of $\Gamma_{\langle t\rangle}f$ is 0 (or close to 0) then it means that the LSTM should remove that piece of information (e.g. the singular subject) in the corresponding component of $c_{\langle t-1\rangle}$. If one of the values is 1, then it will keep the information.

1.3.2 Update gate:

Once we forget that the subject being discussed is singular, we need to find a way to update it to reflect that the new subject is now plural.

$$\Gamma_{\langle t\rangle}{}^{u}=\sigma(W_u[a_{\langle t-1\rangle},x_{\langle t\rangle}]+b_u) \qquad (2)$$

Similar to the forget gate, here $\Gamma\langle t\rangle^{u}$ is again a vector of values between 0 and 1. This will be multiplied element-wise with $\tilde{c}_{\langle t\rangle}$, in order to compute $c_{\langle t\rangle}$.

1.3.3 Updating the cell:

To update the new subject we need to create a new vector of numbers that we can add to our previous cell state. The equation is:

$$\tilde{c}_{\langle t\rangle}=\tanh(W_c[a_{\langle t-1\rangle},x_{\langle t\rangle}]+b_c) \qquad (3)$$

Finally, the new cell state is:

$$c_{\langle t\rangle}=\Gamma_{\langle t\rangle}f*c_{\langle t-1\rangle}+\Gamma_{\langle t\rangle}{}^{u}*\tilde{c}_{\langle t\rangle} \qquad (4)$$

1.3.4 Output gate:

Following two formulas determine which output is to be used :

$$\Gamma_{\langle t\rangle}{}^{o}=\sigma(W_o[a_{\langle t-1\rangle},x_{\langle t\rangle}]+b_o) \qquad (5)$$

$$a_{\langle t\rangle}=\Gamma\langle t\rangle^{o}*\tanh(c_{\langle t\rangle}) \qquad (6)$$

Where in equation 5 one decides what to output and in equation 6 we multiply that output by the activation of the previous state.

1.3.5 Types of RNN (By Input-Output Dimensions):

One-To-One: The simplest of the RNN architectures, this architecture takes in input of single dimension and outputs a single dimension output. This architecture is especially useful in areas where the input consists of a single entity, like an image and we have a single entity for its output (like a classifier). An image classification algorithm, which focuses on multiple parts of image in a sequence can be a great example of a one-to-one RNN.

One -To-Many: These RNN models, take in a single input but output sequences of data (a vector of entities). The best example for One-to-Many architecture is Image captioning using Inject architecture, where an image is fed into the RNN while the RNN outputs a sequence of words.

Many-To-One: An architecture where the input is a vector in sequence of objects and the output is a single entity. Text Classification and sentiment analysis is a good example of Many-to-One architecture, as in these scenarios, a vector of words serve as the input while a single classification (good/bad) might be the output.

Many-To-Many: In this case, both the input and output are sequences of data (vectors). They may or may not be of the same size. Models having different input and output sequence dimensions are challenging to make, especially where the output dimension is greater than the input dimension. Examples of many-to many architectures are Video Captioners with input as sequence of image frames and output as sequences of words, Text Translators where both input and output sequences are words.

Figure 5 describes various RNN architectures based on Input-Output dimensions

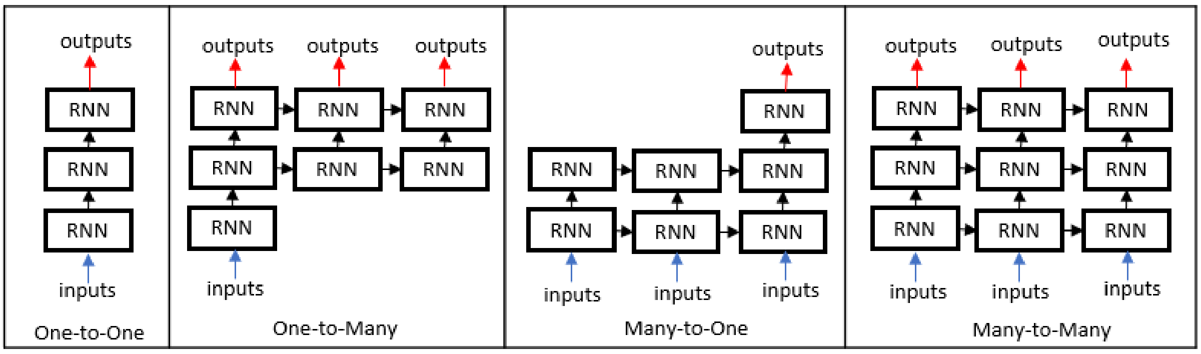


Fig 5: Various Input-Output-Based Architectures of RNN

1.3.6 Bi-Directional RNNs:

These RNN networks have a set of RNNs delivering training in forward direction and a separate set of RNNs delivering training in backward direction. They were introduced to increase the information amount in a network. In normal RNNs, one cannot reach the final future state from the current state. In Bi-RNNs future input information is reachable from the current state. Also, they retain the data length flexibility that normal RNNs have over basic MLPs. In a Bi-RNN, (Figure 6) both forward and backward RNN units produce outputs.

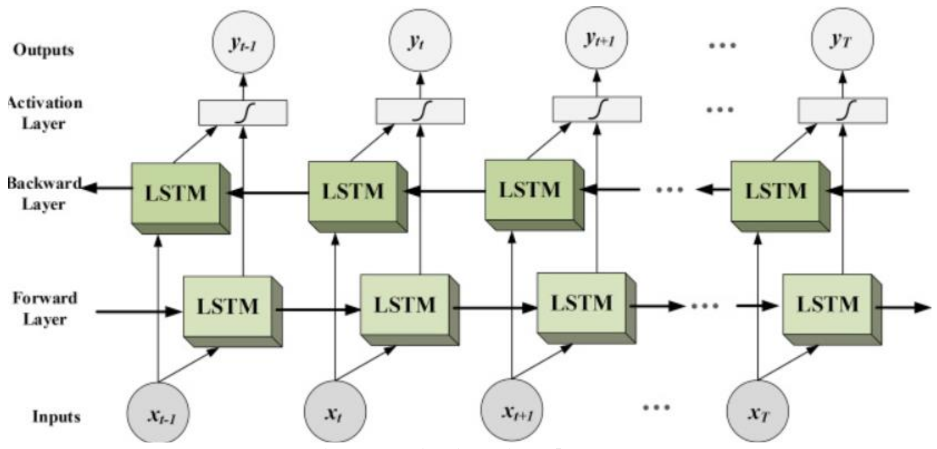


Fig 6: Bi-Directional RNNs

Bi-directional LSTMs have the ability to produce more context aware captions. This is applicable for both image and video captioning and has been put to use for the latter in many researches works (ABIVIRNet). If we have a bidirectional LSTM as the decoder, for classifying video, after feeding rich caption features to the encoder, the LSTM can modify generated captions based on both previously generated and future generations. This produces good results but increases the complexity, parameters and training time by a huge margin.

CHAPTER 2: ATTENTION IN CAPTIONING

2.1 Importance of Visual Attention

Modern image captioners often use attention-based mechanisms which require referring back to the current image region in focus, to generate more context aware captions. Such an approach requires an architecture similar to the inject architecture where the RNN requires to refer to the visual region as well as previously generated words to generate the next word and also decide the next region of focus [6]. Attention is mainly applied to a captioner in the sequence generation phase by the RNN. Most attention models use a multi-modal RNN which received input from the input image as well as the previous generated words. In every iteration, special external features from the image or sentence vectors serve as the attention feature and bias the RNN to produce words which are more related to the current scenario. The "Show and Tell" [12] model, also known as Google NIC is the most widely used baseline image captioning model. Its attention-based variant "Show, Attend and Tell" [64] was the first to introduce the concept of soft and hard attention in captioning, in the year 2015. Figure 7 gives us an overview of both baseline and attention-based image captioning models. The attention module has been highlighted in the RNN module of the attention based captioner in figure B.

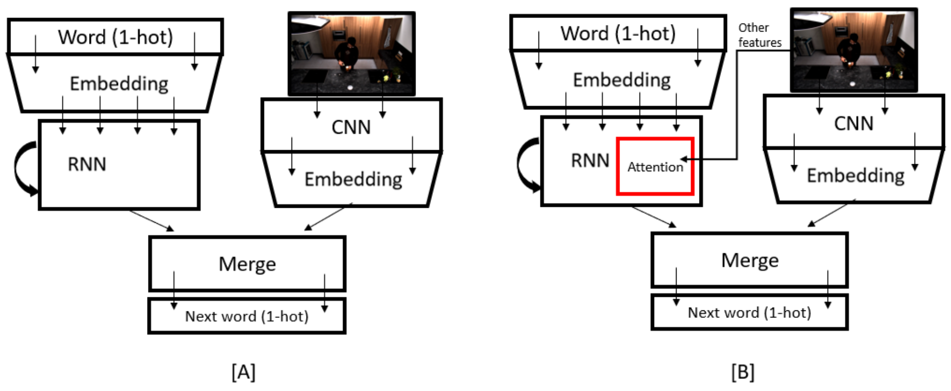


Fig 7: [A] Generic Baseline Image Captioner , [B] Generic Attention based Image Captioner

Figure 8 demonstrates the unsatisfactory performance of a captioning model if no attention is provided. The model used here is a VGG16 based model trained on Flickr8K dataset. The model identifies activities quite well but fails to accurately justify the quantities of objects or place of occurrence of the event.

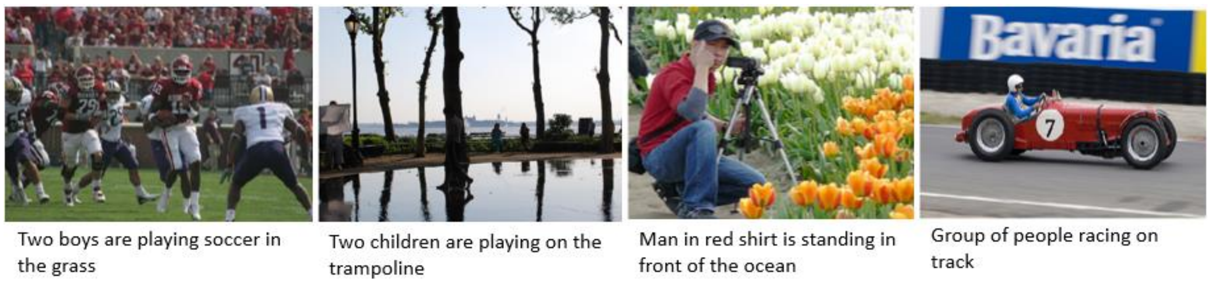

Two boys are playing soccer in the grass
Two children are playing on the trampoline
Man in red shirt is standing in front of the ocean
Bavaria
Group of people racing on track

Fig 8: Baseline model fails to provide attention to detail in captions

2.2 Types of Attention:

When we look at an image, we scan various parts of the image and identify its content, instead of focusing on the image as a whole. This implies that at a particular instance, we provide more attention to a specific region in the image. Modern captioners have tried to devise various methods of applying attention to their models. There can be two types of attention methods, neural networks are made to train on.

2.2.1 Soft Attention:

In soft attention [64,14] the model focuses on a specific part of the image, at a given instance, by distorting other parts of the image. Areas of more interest will be highlighted while other regions will have an attention of 0 (darkened). Figure 9 describes the concept.

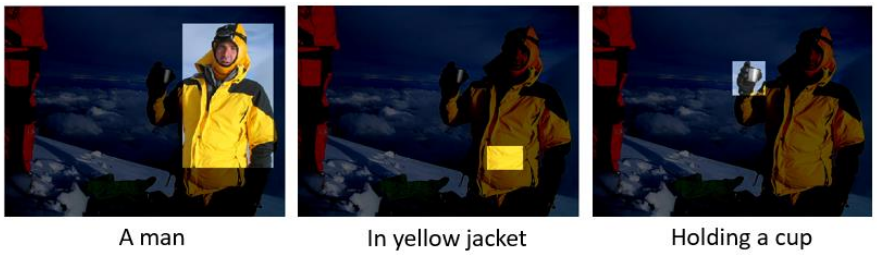


Fig 9: Soft attention highlighting important areas and distorting other regions

Soft attention computes in every step, how much weightage is to be given to a region of an image for generating a related word. Highlighted areas are obviously provided with higher weights (preserving original value) while darkened areas received lower weights.

Let x_1,x_2 and x_3 the marked sub-regions of an image. A score s_i is computed as a measure of attention to be provided to xi, (with the context $C=h_t-1$):

$$s_i = \tanh(W_cC+W_xX_i)=\tanh(W_ch_t-1+W_xx_i) \qquad (7)$$

si passed through a softmax function computes the normalized weight α_i.

$$\alpha_i=\text{softmax}(s_1,s_2,\ldots,s_i,\ldots) \qquad (8)$$

With softmax, α_i adds up to 1, and it can be used to compute a weighted average(Z) for x_1,x_2 and x_3. Finally, Z is used to replace x as the LSTM input.

$$Z=\sum\alpha_ix_i \qquad (9)$$

2.2.2 Hard Attention:

The concept of Hard Attention [64,15,16] realizes that there is often ,no right procedure in which an image is scanned by the human eye and hence is non-differentiable. Given such a scenario, in order to ensure the network is proceeding in the correct direction, in learning, reinforcement learning methods like Monte Carlo can be applied. Reinforcement learning techniques infer attributes from experiments where some samples which are able to achieve the target get a positive score while the ones which do not, get a negative score. Reinforcement Learning suffers from high variance problems when scaled to larger networks with more number of hidden layers. High variance means that one of the samples might be able to reach the reward state while another might fail. This causes problems in backpropagation. Monte

Carlo uses a stochastic method of updating weights by choosing a set of samples and updating weights by averaging out of all these samples, instead of focusing on only a few samples. Choosing a large number of samples can help mitigating the problem of variance. The following equation is used in Monte Carlo reinforcement learning. R_t is the reward signal and r is the reward received at time t. T is the total time span.

$$R_t(s,a) = \sum_t^T \gamma^t r_t \tag{10}$$

Comparing between soft and hard attention, soft attention is more popular because of its simplicity. Soft attention's main drawbacks are that it uses fixed grids to segregate regions in an image , whereas hard attention creates a new memory map of a region of the image, for use in the current iteration. But hard attention networks cannot be trained using back propagation and this makes it more complicated to train the attention module separately. Although hard attention is philosophically more appealing, it does not contribute much to enhancing the final caption output.

2.3 Various Experimented Methods of Applying Attention:

2.3.1 Scene Attention:

Scene specificity defines the global context of an image. Lets take for example, an image with a child in the park. A favorably generated caption could be "A boy is playing with a ball in the park". But, if the same photo was shot in the beach, a desirable caption would be A boy is playing with a ball in the beach" .The importance of the surrounding sceneries are immense, especially in video captioning where the topic of the video itself will be based on the background scene. The authors of [6] have described how to compute scene vectors which contain extracted scene-related global contexts and use them as one of the inputs in an LSTM as an attention module. The scene vectors are computed by initially clustering images using an unsupervised algorithm. Latent Dirichlet Allocation (LDA) [61] is a generative unsupervised clustering model that posits that each document is a mixture of a small number of topics and that each word's presence is attributable to one of the document's topics. LDA is an example of a topic model. Clustering based on applying LDA on ground truth captions of an image, can help us decide the topic attributes of an image. Apart from LDA, image clustering may be also be used to generate scene vectors by unsupervised K-Means clustering [62]. Image clustering will be generated scene vectors based on image specific features like color intensities and edges, unlike language based LDA, which extracts features which are more topic oriented. Fu et al. in their scene specific captioner have hence constructed an 80-dimensional scene vector which contains the probability of that particular image belonging to one of the 80 topics they clustered in the database using LDA. The labelled vectors thus generated by unsupervised clustering, can be used to train a small multi-layer-perceptron (MLP) based on the image CNN features and output the same scene vector. This MLP, which is trained to mimic a clustering algorithm, can then be used to generate topic vectors of images from the test dataset. Figure 10 describes the procedure for training the MLP and extracting topic vectors from an image.

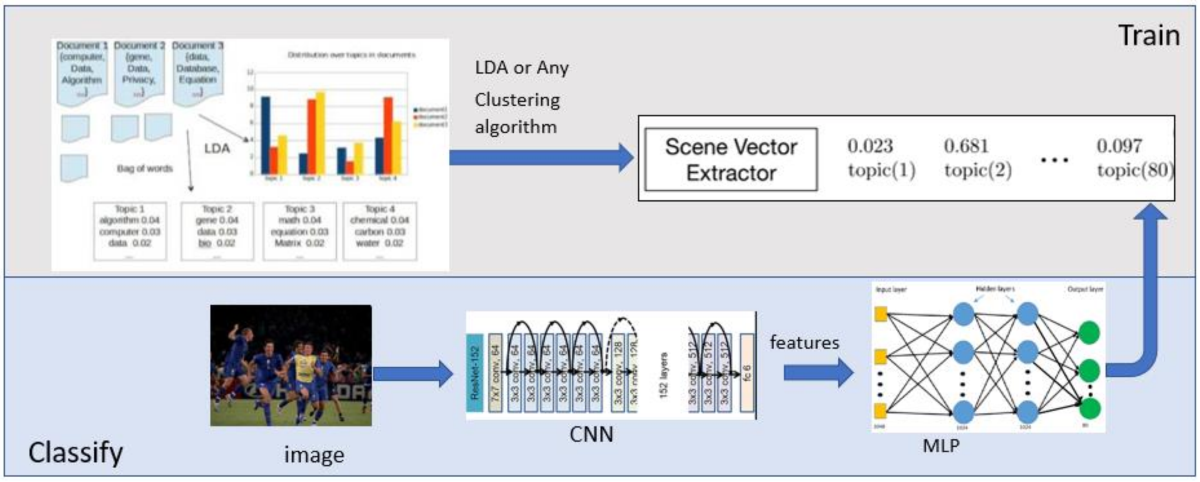


Fig 10: Scene Topic Extraction from an Image

This procedure can be an efficient solution towards a balanced attention measure for captioning as an MLP can be trained quickly, without requiring much computational resources However, the scene MLP from [6] is able to attain an accuracy of 70% only, while trying to imitate the LDA clustering based classification. The reason for this is purely due to the fact that the topic labels were generated based on ground truth image captions, yet the MLP was trained based on the image features. It revises the fact that labelled captions often lack description of complete image information. Better clustering simulations may be implemented based on both image features as well as ground-truth captions, using a multi-modal architecture.

2.3.2 Object or Region based Attention:

The most intuitive approach towards attention in an image lies in the way humans actually see an image. Our vision focuses on multiple objects in the image one at a time and sums them all up while describing them as a whole. This process of focusing on various objects in the image has been proposed in [6] as Region Based attention. This primarily requires detecting separate objects in the image and generating the CNN features of each of these objects. Object detection algorithms specifically build bounding boxes around objects in an image and sometimes associate with describing the objects as well. There are various object detection algorithms available like Selective search [7], R-CNN [8], YOLO [9]. Most of the object detection algorithms start from determining around 1000 to 2000 arbitrary image regions inside an image. R-CNN uses a deep CNN to classify get features of every region and uses an SVM to classify those regions. Selective search on the other hand provides a fast way of detecting objects by localizing similar regions into a single region set and segregating dissimilar ones. Selective search manages to segment an image in 1 second and takes 2 to 40 seconds to evaluate proposals. YOLO (you only look once) is one of the fastest object detection algorithms which use a more unified, yet simpler and faster method to deal with objects in an image. YOLO divides an image into a 7x7 grid where each grid predicts the object contained in that grid, by building a bounding box around it. It uses a 24-layer CNN trained with ImageNet data that predicts every grid content and later stitches neighboring boxes with overlapping content. YOLO suffers the drawback of detecting multiple small objects in a single grid. Despite its drawbacks YOLO is the fastest and well performing object detection algorithm managing to detect objects at 45 frames per second in a TITAN X GPU.

In paper [6] Selective search is used to extract the regions (or bounding boxes) from an image which are then processed for feature extraction. Selective search is best for generating hierarchical boxes where we have region overlaps. However, YOLO may be used for non-overlapping object-based attention. After regions have been generated, the top 29 regions are selected based on a linear classifier trained to distinguish between good (with more overlaps with bounding boxes) and bad regions (with lesser overlaps) and features of these regions are extracted.

In every iteration a single layer neural network is responsible for selecting the region of visual attention from the set of attention regions. The network takes input the previous predicted word, hidden state, set of regions and previous visual attention region to output the current region of interest. Then the regions can be added to an attention module of an LSTM to generate next word.

2.3.3 Saliency:

Saliency [3,4], in perspective of visual attention, refers to where a person focuses on a scene while studying its contents. Studying the various regions of gaze in an image can help us decide where to see, in an image, while captioning them. Recent studies regarding visual saliency have used datasets of images marked with fixation points captured using eye tracing glasses. CNNs can be trained to identify these fixation points automatically, but their performance often degrades due to the rescaling of features in the deeper layers. Saliency Attentive Model (SAM) [5] uses a dilated CNN which preserves the quality of the feature-set, instead of rescaling it and then uses a Convolutional LSTM to refine initial predictions of saliency. Their specialized LSTM uses space varying features instead of time and finally generates a single channel convolutional map containing the saliency features highlighted. People tend to focus their attention to the center of the image, by default, since the main content of the image is photographed usually in the center. However, this may change with multiple points of interest in the image and create confusion regarding where to actually focus in the image. Hence SAM uses a gaussian function to learn the center biases and adjust the saliency map accordingly. A pre-computed (using covariance and mean matrices from image data) Gaussian kernel adjusts the LSTM output according to its bias and the final map is rescaled to the size of the original image. The overview of the whole procedure is described in Figure 11.

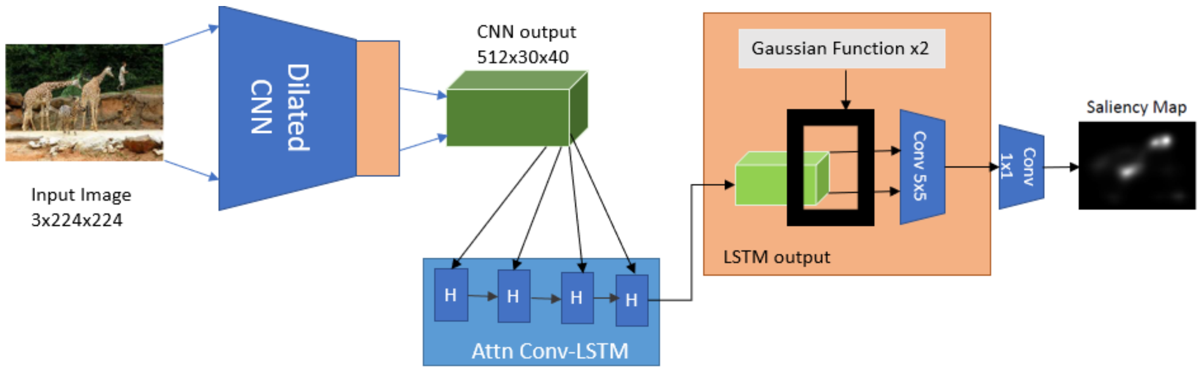


Fig 11: Saliency map generator using attention convolutional LSTM in SAM [3]

Saliency has been incorporated as an attention module in [13] to achieve state of the art results in captioning. The focus of their attention is dynamic as their implementation creates regions from images and focuses on saliency map of a particular selected region at every timestep to generate captions. Initially a Fully Convoluted Network extracts features of multiple segmented regions from the image and also computes their saliency map. An LSTM with attention module is responsible for caption generation. The attention module takes inference from both saliency map and contextual map (region-based feature map) of the current selected region to bias the generation of the next word in the sentence, by the LSTM. The next region is selected, similar to [6], based on the current hidden layer input of the LSTM. This type of sophisticated implementation hence uses attention in attention (saliency in region-based attention) concept to excellence in captioning. However, this also makes the overall procedure rather computationally expensive, having to generate region based and saliency map features for every segment in the image. The generic method of applying Saliency based attention is portrayed in Figure 12.

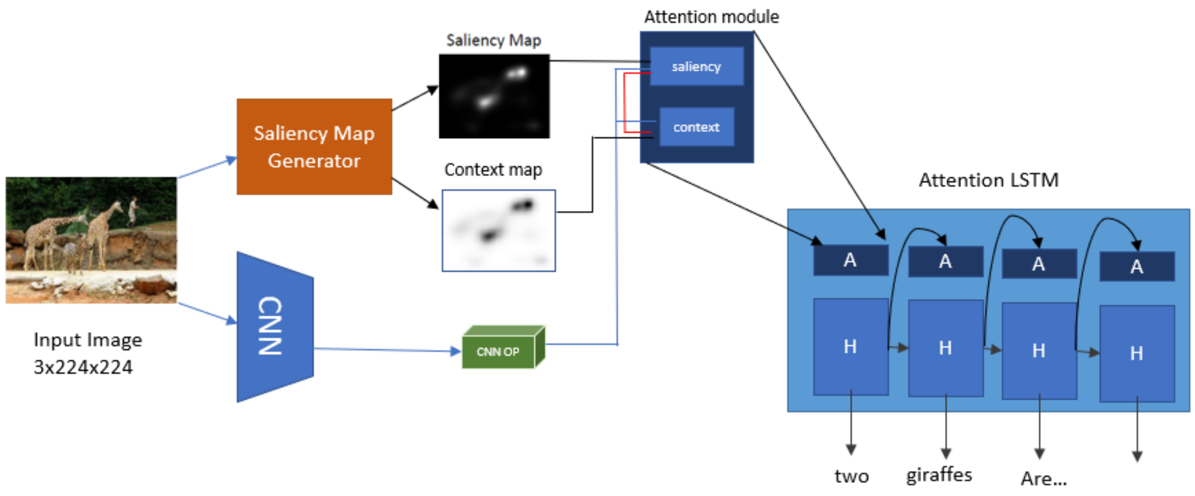


Fig 12: Saliency attention based Captioning

CHAPTER 3: MODEL ARCHITECTURE OPTIMIZATION METHODS

3.1 Glove:

Every captioner contains a word embedding layer which maps the one-hot encoded words, to a smaller sized representation. This layer has to be trained in the process of caption generation to learn the mapping of words existing in our vocab. The Glove [2] model is a pre-trained model which can represent a word in n-dimensional format. This representation of a word, preserves its semantic meaning. Ad-hoc models rely on Euclidean distance between vector representation of words to determine their semantic similarities, but this approach is highly limited to specific usage scenarios. Glove uses words occurring together to infer the similarity between them and this co-occurrence is represented by a log-bilinear model. The logarithmic ratio associate meanings which. are encoded as the n sized vectors. Glove vectors are suitable especially in word analogy tasks like captioning where one predicted word may have various other similar predictions without loss of overall meaning of the entire sentence. We are more likely to generate meaningful and syntactically correct sentences with glove embedding rather than self-embedding layers. Moreover, using pre-trained embedding will enable us to skip training this layer thus enhancing the overall training process of the captioning model.

Co-occurring words, with similar meanings are shown in Figure 13 The dot-product of two co-occurring words gives us the log of the probability of them co-occurring. The entities are arranged in their scenario of occurrence, signifying that the vector difference between the pairs are similar , when viewed from the given perspective.

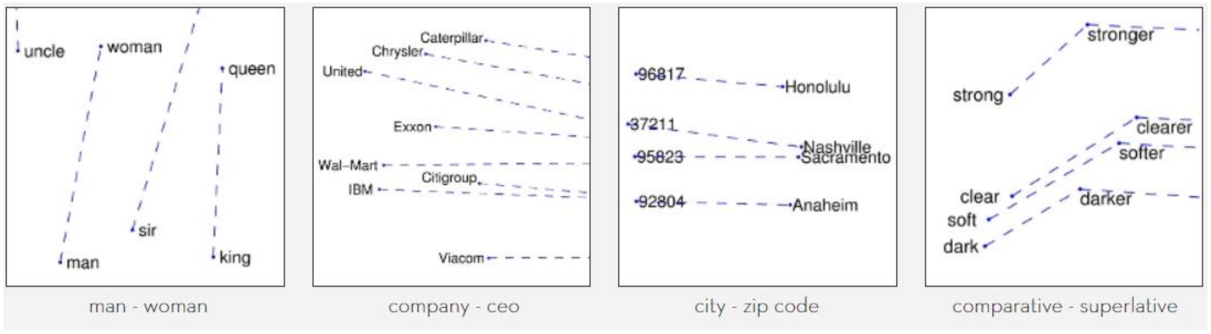


Fig 13: Similar or co-occuring words mapped to eachother (Image Source: GloVe)

3.2 Choice of words:

A good captioner has to be careful about its choice of words. In every timestep of sentence generation, a new word is appended to the current sentence which has to be chosen from a list of predicted words. Generating the best sentence will require us to try out all possible combinations of words, suitable for framing the sentence. But that will take exponential time. However, there are specialized algorithms to search for the best words from the list of words, at every iteration. The most popular searches used to achieve this are greedy search and beam search. These algorithms are used explicitly in the testing phase of the captioner.

3.2.1 Greedy Search:

In every iteration, words are generated with probabilities of their occurrence given the current sentence. Greedy search simply chooses the words with highest probability in every iteration. This operation is extremely fast, but often fails to generate the most optimal sentences.

3.2.2 Beam Search:

Beam Search [1] is a technique where we keep track of a few best-by-now sentences (their number determined by the beam size, B. It is used in the sentence generation phase to generate the next best possible word from a group of words and keeps the best word to be used for further generation of sentences. The log likelihood is measured for the words, with the given sentence, rather than multiplying with their probabilities repetitively, since multiplying probabilities often lead to underflow in floating point arithmetic, as we progress with the decimal digits. Time complexity of beam search in worst case time = O(B*m) where B is the beam size and m are the maximum depth of any path in the search tree. Usually best sentences are generated with values of B ranging between 3 to 5. For B=1, Beam search behaves like a greedy search. Modern captioners only use Beam Search for caption generation. Performance of beam search is 2 to 6 percent better than greedy search, on an average, based on the performance metrics discussed in section 2.

3.3 Enhancing Description Quality:

3.3.1 Multiple Instance Learning:

Even though annotated datasets have multiple sentences supporting an image, only one of them is used for validating the image caption. However, it may so happen that the other descriptions have other inferred concepts about the image which might have been missed by the current one. As a solution to this the OPR-MCM [11] (Online Positive Recall -Multiple Concept Mining) was proposed. It uses MIL (Multiple Instance Learning) by detecting the missing concepts from other captions, once the image caption has been generated based on the current trained model. The missing concepts are treated as negative concepts and selectively they are chosen to train the model again with these concepts so that we have a better and more complete caption generated in the following iterations.

3.3.2 GANS:

Even if generated captions are able to describe the content of an image, the sentence construction quality can be further refined to replicate that of a human constructed sentence by using Generative Adversarial Networks (GANS) [17]. The architecture GANS consist of a generator which generates an output and a discriminator (the adversary) which judges if the generated output is real or fake. Inspired by the Turing Test, GANs can be an economical and effective way of judging if generated captions from a captioner are generated by human. Good captioners will aim at generating captions which are almost indistinguishable from those proposed by humans. Over the years, multiple attempts have been made [18,19,20] to optimize caption generation using GANs. Discriminators for a captioning algorithm store information regarding the similarities between a caption and its image. The loss from the discriminator can contribute to the overall loss generated in every iteration of the training step to refine the model predictions. As the loss from GANs mainly focus on how far the syntax of the generated sentence is from replicating a human generated sentence, GANs can enhance the sentence construction and attention in generated captions, to appear more human like. Figure 14 describes how a discriminator can be trained on image and caption pairs.

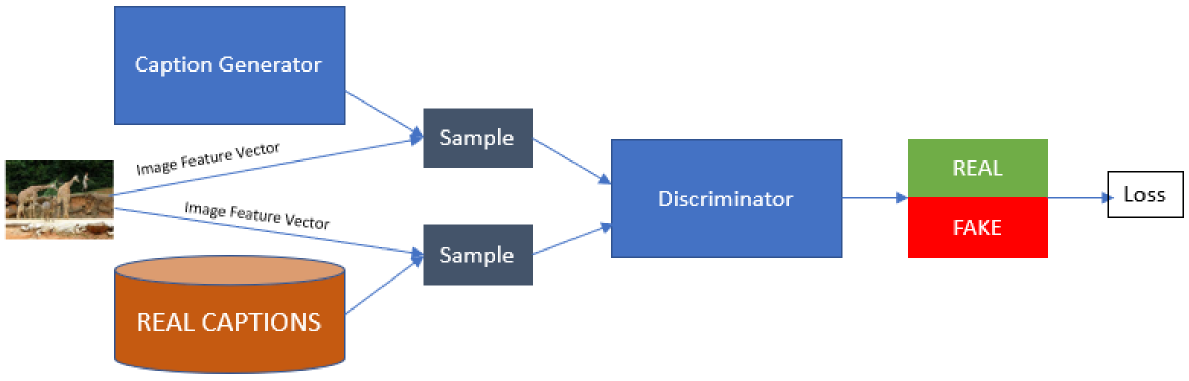
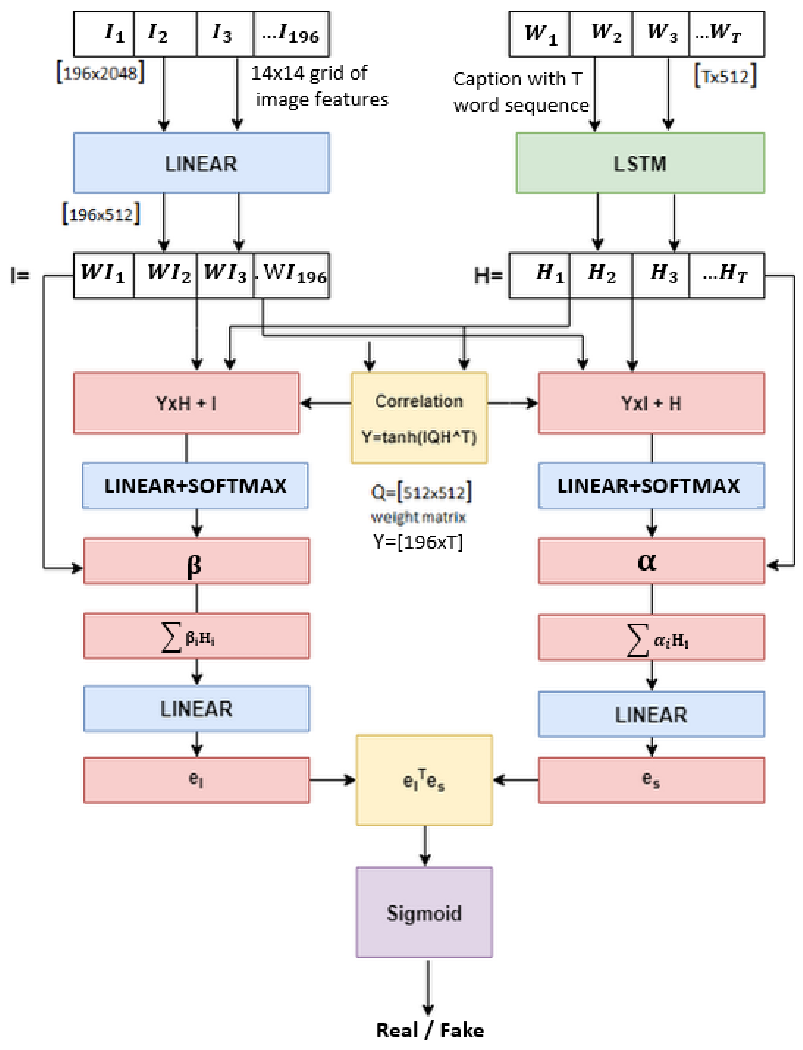


Fig 14: Role of GAN in optimizing Caption Generation

These images, caption pairs are formed in an embedding layer whose output is fed into the discriminator as input. Although most generators [19,20] feed into the discriminator after just before computation stage in discriminator, in [18] the authors chose to differ. They compute the correlation of the image, caption pair using bilinear transformations at an earlier stage to compute similarity between the image and its caption. Then they further process the image and caption features separately, with attention to their similarity matrix, produced in the previous step. Once processed separately, both the outputs are fed into the final stage of discriminator for the decision output. The loss generated from the discriminator will contribute to backpropagation in the network to enhance successive generations. The authors have used soft attention model based on Show and Tell [64] with 14x14 grids (196 regions) to generate 196 image feature vectors. In every iteration word are formed from a 512-state size output LSTM. Each word vector has an embedded length of 512. A caption is stated to be formed of T words. To generate the correlation between the current region of focus and the current word, a bilinear transformation is done between an embedded image vector and the word vector to get Y, the correlation map. This Y is used in further linear units in each module (word and image features) whose outputs are summed up together (using a weighted sum) and then fed into the final module of the discriminator. The loss from the output of the discriminator is backpropagated. Co Attention based discriminator proves to be able to classify images with unrelated objects as well, and hence it has also been tested on the OOC dataset to get good results. Figure 15 describes training the GAN in [18].



I_1 I_2 I_3 ...I_{196}
[196x2048]
14x14 grid of image features
LINEAR
[196x512]
I=
WI_1 WI_2 WI_3 .WI_{196}
W_1 W_2 W_3 ...W_T
Caption with T word sequence
[Tx512]
LSTM
H=
H_1 H_2 H_3 ...H_T
YxH + I
Correlation Y=tanh(IQH^T)
YxI + H
LINEAR+SOFTMAX
Q=[512x512] weight matrix
Y=[196xT]
LINEAR+SOFTMAX
β
α
$\sum \beta_i H_i$
$\sum \alpha_i H_i$
LINEAR
LINEAR
e_I
$e_I^T e_s$
e_s
Sigmoid
Real / Fake

Fig 15: Co-Attention Based Discriminator Model

Reinforcement learning [63,79] has also been applied in image captioning by modeling the caption generation stage as the rewarding scheme. On basis of the rewards, an inference can be generated which helps in making more contextual predictions for the next word. The main goal of such a technique revolved around minimizing the negative reward produced at the end of caption generation.

3.4 Ensembles:

An ensemble model is a model prepared by aggregating outputs of multiple models with same aim, to learn better. The concept of assembling suggests that poor performing models may perform better, if their learnings are put together. Lots of high variance results can be combined to make a low variance outcome and thus a desirable output with low bias and low variance may be achieved. The averaging of outputs is done by weighted voting mechanism where weight of contribution of every model may be proportional to the accuracy of the model. Assembling helps in regularizing the output without requiring early stopping. Ensembled models can be applied to captioning models as well, inference from multiple models can contribute to generation of a caption with more rich information and semantics. Various authors [6,79,18] have demonstrated enhanced performance by usage of ensemble models.

3.5 Variations in Captioning:

Apart from basic image captioning, several variants of the same, have also been put to practice in recent researches. Captioning RNNs have been trained to produce captions with sentiments and better semantic styles. Authors of [65] created SentiCap by modifying the GNIC [12] captioner to include another RNN for generation of captions with sentiments. In this case, a switching RNN was used, which is a two-step model, trained with a large dataset of factual captions and a small dataset with sentimental words (positive or negatively classified). For generating and validating such sentimental captions, a new caption dataset had to be prepared containing a positive and a negative sentence caption for every image. The model was successful in describing images with proper adjectives having positive or negative meaning and also determining the sentiment (positive or negative) of the input image. SemStyle [66] was introduced for generating stylized captions, which are more expressive and narrative in nature. There are two main challenges in developing a storyteller captioner like SemStyle :1) Preparing dataset captions with language style inspired from story books and 2) maintaining relevance of the caption words, to the corresponding image, while generating story-like sentences. The authors tackled the first challenge by replacing the verbs in MSCOCO captions with ones collected from novels. Then, they proposed an encoder-decoder model where a term generator encodes image features to semantic terms which are framed into sentences by a language generator. SemStyle achieved a SPICE score of 0.134 and 87% generated captions marked to be related to the image, by human judgement.

CHAPTER 4: DATASETS FOR IMAGE CAPTIONING

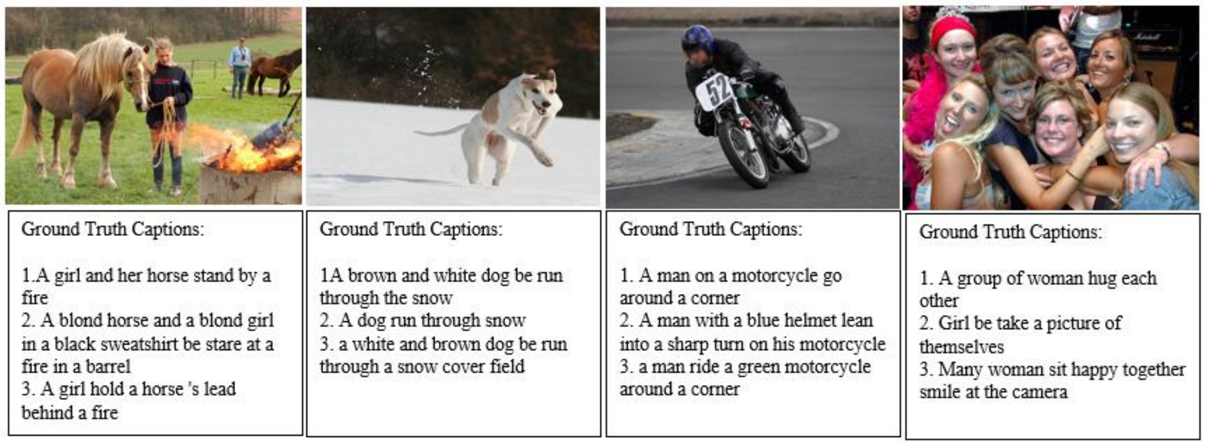
Having a labelled dataset [24] for captioning is the most important part of this task as gathering or preparing such datasets can be a time-consuming task. Some publicly available labelled datasets are as follows:

4.1 Labelled Image Datasets

ImageNet: The largest image dataset [23] in the world containing 14,197,122 images, organized according to the WordNet hierarchy. Most of them are labelled with nouns. Most data sources have ImageNet as their mother source.

MSCOCO: Microsoft Common Objects in Context (MSCOCO) [22] contains 120K images with 5 captions for each image. There are 80k images for Training and 40k images for Validation. The contents of the image cover a variety of commonly seen objects, animals and activities.

FickR: FickR [25] Contains 2 datasets. A large dataset with 30000 images with 5 captions each splits: 28000 images for Training and 2000 images for validation and another one with 8000 images with train and test split as 6000 and 2000. Figure 16 contains examples from FlickR dataset.



Ground Truth Captions:

1.A girl and her horse stand by a fire
2. A blond horse and a blond girl in a black sweatshirt be stare at a fire in a barrel
3. A girl hold a horse 's lead behind a fire

Ground Truth Captions:

1A brown and white dog be run through the snow
2. A dog run through snow
3. a white and brown dog be run through a snow cover field

Ground Truth Captions:

1. A man on a motorcycle go around a corner
2. A man with a blue helmet lean into a sharp turn on his motorcycle
3. a man ride a green motorcycle around a corner

Ground Truth Captions:

1. A group of woman hug each other
2. Girl be take a picture of themselves
3. Many woman sit happy together smile at the camera

Fig 16: Samples from FlickR Dataset with the top 3 Groud Truth Annotations

Both the FlickR and MSCOCO datasets are highly similar to eachother as most of the images in the datasets have the common mother-source , which is imagenet. Most images are originally taken from the FlickR website itself. However, the MSCOCO annotations are better in quality and describe images better, compared to FlickR.

SALICON: SALICON(Saliency in Context) [26] the largest available dataset for saliency prediction. It contains 20,000 images taken from the MSCOCO dataset. Every image in the SALICON dataset has small highlighted regions, denoting them to be fixation points for the human eye, or in other words, regions of focus and attention. Eye fixations have been simulated with mouse movements while preparing the dataset. SALICON can be used to study attention models in Image Captioning. There are other notable datasets focusing in saliency like MIT1003[27], MIT300[28], CAT2000[29] which may be used for the same. In Figure 17 are a few marked images from SALICON dataset.

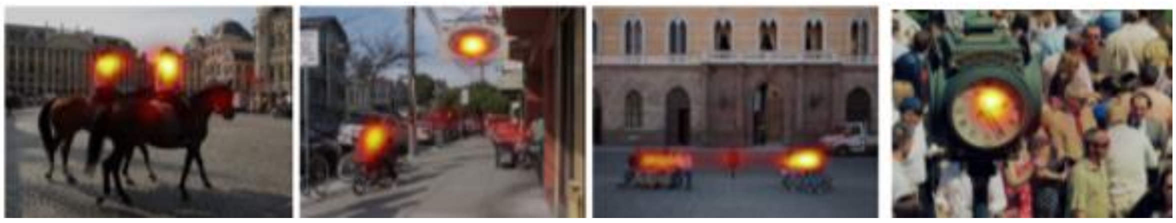


Fig 17: Sample Images from SALICON dataset. Highlighted areas are regions of Eye Fixation

OOC: The "Out of Context" (OOC) dataset [21] with contains 218 images containing objects that are highly unrelated to the context of the image. Such a dataset can prove to be highly challenging for a caption generator to be tested as a generator is usually trained on normal datasets where sentence formation is learnt based on relatable context only. It is highly recommended for future captioners to be tested on OOC for achieving state of the art results. Following Figure 18 are a few examples of the content of OOC dataset.

Caption: black car has gotten lodged atop a silver metal barrier on a roadway	Caption: a person looks up at a street light that is designed like a pot of coffee	Caption: a person hangs from a street sign using the sign to cover their face	Caption: A lady is flying a chair as if its a kite while walking along the water edge	Caption: a bird on the ground with keys to the laptop are scattered on the floor

Fig 18: Sample Images from the Out Of Context (OOC) dataset with Ground Truth Captions

4.2 Automatic Annotation Technique:

Annotating image datasets is a time consuming and costly procedure . Moreover it requires a specialized group of people having good perception and grammatical capabilities to properly annotate an image . As a novel solution for annotations, KunFu et al. [10] devised an algorithm that automatically annotates images using pseudo-pair generation. The concept of their method lies in segregating different facts from every image and sentence available to us and then generate new pseudo-pairs of images and sentences by replacing parts of image and sentence content with known values from the corpus generated. They initially generate a corpus of sentences and generate a knowledge base from them. Then they equate similarities between different entries in the knowledge base. Then pseudo sentences are generated by replacing concepts in the knowledge base items. Similarly, using various feature vector of images , pseudo image vectors are generated. Finally a captioner is trained to learn how to caption, by using adaptive visual replacement(AVR). The overview of the process is described in Figure 19.

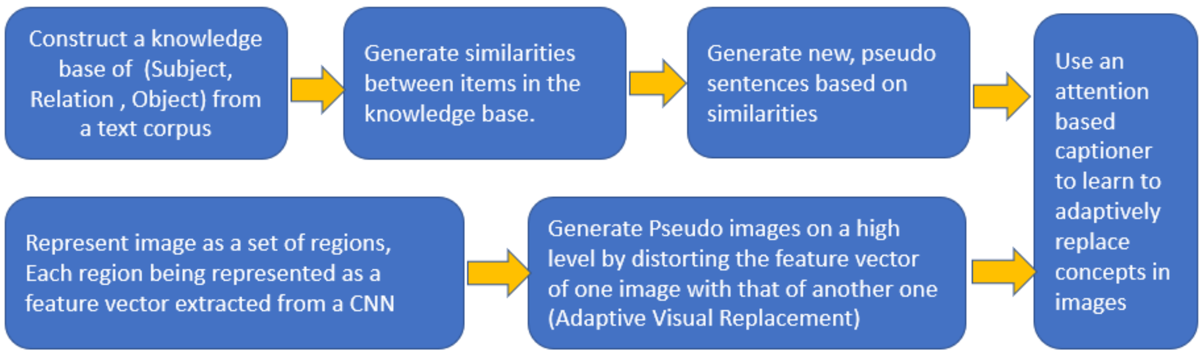

Construct a knowledge base of (Subject, Relation , Object) from a text corpus
Generate similarities between items in the knowledge base.
Generate new, pseudo sentences based on similarities
Use an attention based captioner to learn to adaptively replace concepts in images
Represent image as a set of regions, Each region being represented as a feature vector extracted from a CNN
Generate Pseudo images on a high level by distorting the feature vector of one image with that of another one (Adaptive Visual Replacement)

Fig 19: Generating pseudo images and sentences and teaching a captioner adaptive visual replacement

The annotation generation as proposed by [10] requires the corpus to have similar items existing in the knowledge base to successfully caption an image without human inference. This may be effective for datasets containing images of similar subject and objects . However for a very diverse dataset which may contain lots of unrelated objects or in areas that require dense captioning, their technique may face a lot of difficulties.

CHAPTER 5: EVALUATION METRICS

BLEU: Bilingual Evaluation Understudy Score (BLEU)[80] is a quick and inexpensive way to evaluate a generated sentence with reference to the original sentence. A perfect match gives a score of 1 and a perfect mismatch yields a 0 score. The approach works by counting matching word based n-grams (without considering word order) in the generated sentence to n-grams in the reference sentence. BLEU-1, BLEU-2, BLEU-3, BLEU-4 scores respectively signify scoring by considering 1,2,3 and 4 gram wordings respectively. BLEU is available in the NLTK library in python.

METEOR: Metric for Evaluation of Translation with Explicit Ordering[81] judges the generated statements by creating one to one mappings with reference statements . Here mapping refers to getting the same word in both reference and generated statements. Then using precision and recall of ratio mapped words to total words, the score is calculated. It tackles some shortcomings of BLEU like recall and word matching count.

ROGUE: Similar to BLEU, ROGUE(Recall-Oriented Understudy for Gisting Evaluation)[82] works with N grams. It calculates the N gram overlaps. ROGUE-L computes longest common subsequence between two statements. The major difference between ROGUE and BLEU is that BLEU is precision based while ROGUE is recall based. ROGUE cannot determine if the result is coherent or the sentences flow together in a sensible manner.

CIDEr: Consensus-based Image Description Evaluation(CIDEr)[83] takes into account the possibilities of having varying descriptions for the statement with same meaning. CIDEr score accounts for both precision and recall as the average cosine similarity between the generated

sentence and the reference sentences are calculated for n-grams of length . These n grams are chosen by giving higher weightage to more frequent keywords in the reference sentence and lower weightage to commonly occurring words (like articles) , after calculating the TF-IDF scores.

SPICE: Semantic n-gram overlap methods for caption evaluation are limited to matching of words instead of their latent meanings . The sentences might be judged as dissimilar even if they contain the same semantic meaning for example : “some vegetables are being cooked on a stove” and “ A carrot and a garlic is sitting in top of a black container ” will not be evaluated by BLEU scores to have similar meanings. Semantic Propositional Image Caption Evaluation (SPICE)[84] was proposed to tackle this limitation , so that fair judgements could be made for caption comparison. SPICE creates a dependency tree (scene graph) from objects occurring in related scenes to capture their semantic relations. The reference and ground truth captions are encoded into candidate representations and then To evaluate the similarity of candidate and reference scene graphs, we view the candidates from the scene graph to capture the similarity between two sentences.

BLEU, CIDER and METEOR scoring metrics have gained popularity due to their conventional usage for prolonged time. Apart from these metrics, captions have been evaluated based on crowd-sourced judgements, where a group of people vote generated captions on an online server based on relevance and appropriability .Over the years , judgement of generated captions have been made on basis of both ,to what extent the description is able to describe the image and , how close is it to a sentence framed by a human. Hodosh et al.[85] concluded in his research that evaluation metrics like BLEU and ROGUE do not do complete justice in caption evaluation and further stated that rank based metrics can be better at evaluating comparative quality of captions. S@k scores which retrieve the percentage of results relevant to the image, from the top k generated captions (analogous to recall) has been recommended by the authors to judge a caption's quality.

CHAPTER 6: IMAGE CAPTIONING RESULTS COMPARISON

Image Captioning has been quite a competitive topic in the research industry and with recent advances in deep learning, the results of the state of the art captioners are very impressive . The performance comparisons between various captioning architectures has been described in Table 6 and their architectures have been compared in Table 6. While baseline image captioners find it difficult to distinguish between objects having similar properties of colour or shape (like the sky and ocean, or a bed of flowers) , adding attention to the captioners , enhance the caption quality to a huge extent as the model acquires the capabilities to differentiate objects based on certain scenarios and hence bias the word generation towards the context of the image only. The performance variations are clearly visible in the basic GNIC[12] model and its attention based variant[64] , which performs better by a huge extent. Hard attention based captioners , although slower and harder to train, perform better by a great margin , compared to soft attention based model. The captions produced by the hard attention based GNIC still tops the list even after multiple models were introduced in later years. Region based attention[6], which is similar to hard attention hence, performs on par with hard attention . Scene attention based model[6] can be trained quickly as it requires minimal computation and achieves a perfect balance between computation complexity and performance. It can be ideal for caption generation in devices with limited computational abilities (for e.g. Smartphones). Even though it is computationally heavier and involves more sophisticated techniques of accurate eye tracking, saliency[13] based attention performs on par with scene attention , thus making scene attention to be the inevitable choice when it comes to performance to cost ratio. Reinforcement based learning methods like [18,79,63] which aim to generate inference from the errors in generated captions, have enhanced quality of generated captions in multitude of ways. GAN based captioners[18] aiming to generate more human-like sentences succeeds in generating well framed sentences with highly descriptive verbs compared to other captioning architectures. The feedback provided from actual human response regarding caption quality further boosted the network’s performance in[18]. Multiple Instance Learning boosted the caption scores of OPRMCM[11] to beat state of the art methods as the concept directly incorporates training of the model with concepts it has failed to identify in previous caption proposal. Multiple Instance Learning enables a model to generate captions with maximal detail coverage and often leads captions to be over-descriptive of objects in scenes. However, such a method can be universally applicable to most captioning models to boost their performance.

Table 2: Comparison between architectures of various image-captioners

Model	**Year**	**CNN**	**RNN**	**Attention**
GNIC[12]	2015	GoogleNet	LSTM	-
GNIC-Attn[64]	2015	VGG16	LSTM	Soft and Hard
RA+SS[6]	2017	ResNet152	LSTM	Region + Scene
Saliency[13]	2018	ResNet50	LSTM	Soft + Saliency

GAN (Co-attn) [18]	2018	ResNet 101	LSTM + GAN	Soft +SCST[79]
OPRMCM[11]	2019	VGG16	LSTM	MIL
SentiCap[65]	2018	VGG16	Switching LSTM	-
SemStyle[66]	2018	Inception V3	GRU	Soft
DeepReinf[63] Policy Value	2019	VGG16	LSTM	-

Table 3: Comparison between performances of various image-captioners

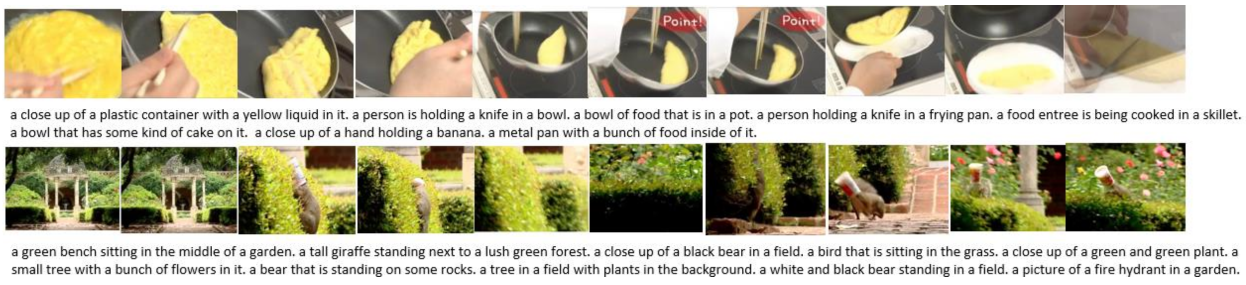
Dataset	**Model Name**	**Attention**	**B1**	**B2**	**B3**	**B4**	**M**	**C**
MSCOCO								
	GNIC[12]	-	-	-	-	27.7	23.7	85.5
	GNIC Soft[64]	Soft	70.7	49.2	34.4	24.3	23.9	-
	GNIC Hard[64]	Hard	71.8	50.4	35.7	25	23.04	-
	RA[6]	Region Based	71.7	54.8	40.9	30.2	24.2	92.6
	SS[6]	Scene	71.2	53.6	39.4	28.9	24.1	89
	RA+SS[6]	Region+Scene	71.7	54.9	41.1	30.6	24.5	93.3
	Saliency[13]	Saliency	70.8	53.6	39.1	28.4	24.8	89.8
	OPRMCM[11]	Missing Concepts	**75.8**	**59.6**	**46**	**35.6**	**27.3**	110.5
	DeepReinf[63] Policy Value	-	-	-	39.5	28.2	24.3	90.7
	GAN(Co-Attn) [18]	Soft+ Co-Attention GAN	-	-	-	33	27.1	**111.1**
Flickr8k	Baseline	-	54	34	25	15	-	-
	GNIC[12]	-	63	41	27	-		
	GNIC Soft[64]	Soft	67	44.8	29.9	19.5	18.93	-
	GNIC Hard[64]	Hard	**67**	**45.7**	31.4	**21.3**	20.3	-
	RA[6]	Region Based	59.5	40.4	26.2	16.6	17.8	39.9
	SS[6]	Scene	62.2	44	30.1	20.2	20	51.2
	RA+SS[6]	Region+Scene	61.2	43	29.6	19.8	19.5	48.9
	Saliency[13]	Saliency	63.5	45.6	**31.5**	21.2	**21.1**	**54.1**
Flickr30k	GNIC[12]	-	67	**45**	30	-	-	-
	GNIC Soft[64]	Soft	66.7	43.4	28.8	19.1	18.49	-
	GNIC Hard[64]	Hard	**66.9**	43.9	29.6	19.9	18.46	-
	RA[6]	Region Based	62.9	44.1	**30.6**	21	18.7	43.2
	SS[6]	Scene	63.2	44	30.1	**29.9**	18.3	38.9
	RA+SS[6]	Region+Scene	61.2	43	29.6	19.8	19.5	**48.9**
	Saliency[13]	Soft+Saliency	61.3	43.3	30.1	20.9	**20.2**	44.5
MSCOCO (Sentiment)	Senticap[65]	Positive Sentiments	49.1	29.1	17.5	10.8	16.8	54.4
		Negative Sentiments	50	31.2	20.3	13.1	16.8	61.8
MSCOCO	Semstyle[66]	-	65.3	-	-	23.8	21.9	76.9

(verbs replaced with novel wordings)								
OOC	GAN(Co-Attn) [18]	Soft+ Co-Attention GAN	-	-	-	17.9	17.3	45.8
SALICON	Saliency[13]	Soft+Saliency	69.2	51.4	37.2	26.9	22.9	73.3

CHAPTER 7: BASIC VIDEO CAPTIONING AND ITS PROBLEMS

7.1 Video Captioning

Excellent Image captioning has led way to research aspects in video captioning and classification. In the article “Deep Learning for Video Classification and Captioning” [67], the authors have discussed various approaches towards recent state of the art Video Captioning as well as providing sources of datasets and insight to the used evaluation metrics. Video Captioning inherently means to caption several frames of the video such that we get the contextual meaning of the series of frames in the video. It uses the exact science behind Image captioning. The most basic and intuitive approach towards video captioning would be to segment the video into a series of frames of distinct images, taken at intervals of 5 to 10 frames and captioning each frame. But this approach has its own problems and we demonstrate the issues of such an approach in Figure 20. We used a image captioning model similar to the one used in [6] with scene specific attention to caption every frame of the video. Some video captions generated have been compiled below with their corresponding frames. Repetitive captions and frames are not shown . At the very outset, it can be inferred that the model can generate satisfactory results as far as captioning per frame is concerned. The caption generation takes less than 0.5 sec per frame and hence could be suggested to be applicable for real-time video captioning , provided there is a setup for that.



a close up of a plastic container with a yellow liquid in it. a person is holding a knife in a bowl. a bowl of food that is in a pot. a person holding a knife in a frying pan. a food entree is being cooked in a skillet. a bowl that has some kind of cake on it. a close up of a hand holding a banana. a metal pan with a bunch of food inside of it.

a green bench sitting in the middle of a garden. a tall giraffe standing next to a lush green forest. a close up of a black bear in a field. a bird that is sitting in the grass. a close up of a green and green plant. a small tree with a bunch of flowers in it. a bear that is standing on some rocks. a tree in a field with plants in the background. a white and black bear standing in a field. a picture of a fire hydrant in a garden.

Fig 20: Naive Per-Frame video captioning

However one drawback of the model is context awareness of the entire video, due to processing of every frame disjointly. Also , if a model is trained based on image captioning , it generates frame descriptions in the same format as one would describe a separate image. These are some major problems that arise when using an 2D captioning model (image captioner) for spatio-temporal 3D data like videos. The temporal dependencies of a video are lost in such approaches . To enable context awareness and generate more synchronized sentences, which are linked to one another, we need both temporal and spatial features to be recognized.

7.2 Handling Spatio-Temporal Data:

3D CNNs: Normal CNNs are mainly 2D CNNs which take input a 2 dimensional RGB image and output its feature vector, which again may be a 2 Dimensional vector or a flattened single dimensional vector. 2D CNNs are great in preserving image features but for videos , they are lossy. This is due to the fact that they cannot preserve temporal dependencies. 2D convolution of a video (represented in 3D) still gives us an image. To preserve temporal dependencies, which are of huge significance inside a video, we need to add another dimension to our input and process all the sequence of video frames together as a 3 dimensional input. This is possible by using a 3D CNN ,for example, Conv3D[53]. In 3D CNNs, theoretically, we have the input as a fixed 3Dimensional vector where the first 2 dimensions are the frames height and width and the third dimension represents the time feature , representing the sequence in which that particular frame arrives. Note that it is mandatory for a video to be represented as a sequence of fixed number of frames to preserve the consistency for input dimensions in the 3D CNN. In 3D CNNs, we have 3D convolutional and pooling layers, and 3D kernels to adapt to them. The 3D kernels are usually of size 3x3x3 and operate on 2 strides (one across depth and one across height and width). In the convolutional. The rest of the process is similar to that of a 2D CNN. The output generated by the convolutional layer, after applying a single filter kernel, is again a 3D vector. Figure 21 describes the representation of a convolutional layer in the network for 3D vector. Apart from captioning and action recognition, 3D convolutions have also been implemented for use in Medical scene analysis of 3D images and other forms of 3D image classification.

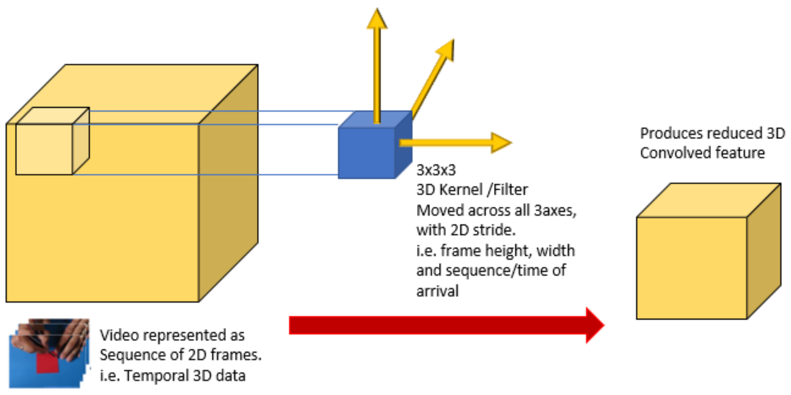

Produces reduced 3D Convolved feature
3x3x3
3D Kernel /Filter
Moved across all 3axes, with 2D stride.
i.e. frame height, width and sequence/time of arrival
Video represented as Sequence of 2D frames. i.e. Temporal 3D data

Fig 21: 3D convolution layer in a 3D CNN for Video data

3D CNNs are more complex and take more time to train than their 2D versions. One major drawback of using 3D CNNs is that it is mandatory to have all frames of the video together , as input. Hence it cannot be used for online processing of video (or streaming video), where the whole video is not available in one go. For such cases, it is better to have a 2D CNN model combined with an LSTM model for encoding temporal features.

7.3 Types of video captioning :

Video captioning can be of two different types 1) Summarized and 2) Paragraph . Summarized video captions are single sentence descriptions which summarize the entire video. Summarized captioning can be applied to small video clips, with duration of a few seconds, where number of subjects and activities may be limited to one . Complicated scenes with multiple objects require multiple sentences to be described properly and paragraph captioning is used for such scenarios. Paragraph captions are more complicated to implement and generate multiple sentences describing the activities in a video . Another technique of generating paragraph captions is to segment a long video into small clips containing single events, and generate summarized captions for each of the clips. However that requires identifying the important clips inside the video and require a trained model to identify such segments to extract them from the video automatically. Various methods have come to existence , with rigorous experimentation , over the years, to describe videos with captions. Some of the important ones have been described in the following sections.

7.4 Attention in Video:

Similar to attention in images, video attention plays a significant role in identifying the relations between several entities participating in the events in a video. Video attention primarily comprises of temporal attention which focuses on changes occurring in the video frames , with time. The temporal attention is best represented as a convolutional 3D feature, formed by concatenation of several features generated by computer vision description techniques like HOG, HOF, MBF. Some of them are described in this survey. Convolutional 3D features hence generated , can be fused with the 2D convolutional features to get the final fusion score vector which is then passed onto the encoder.

7.4.1 HOG:

Histograms of Oriented Graphs(HOG)[56] identifies changes in the video per frame , on a pixel level. HOG was primarily used in image processing and object detection techniques, especially people detection. (Figure 22)It involves calculation in the direction of alteration of pixel intensities of different regions and denoting their changes (like change in the degree of rotation of a pixel) for the regions in a histogram (cells indicating the range of degrees by which a pixel has been rotated).

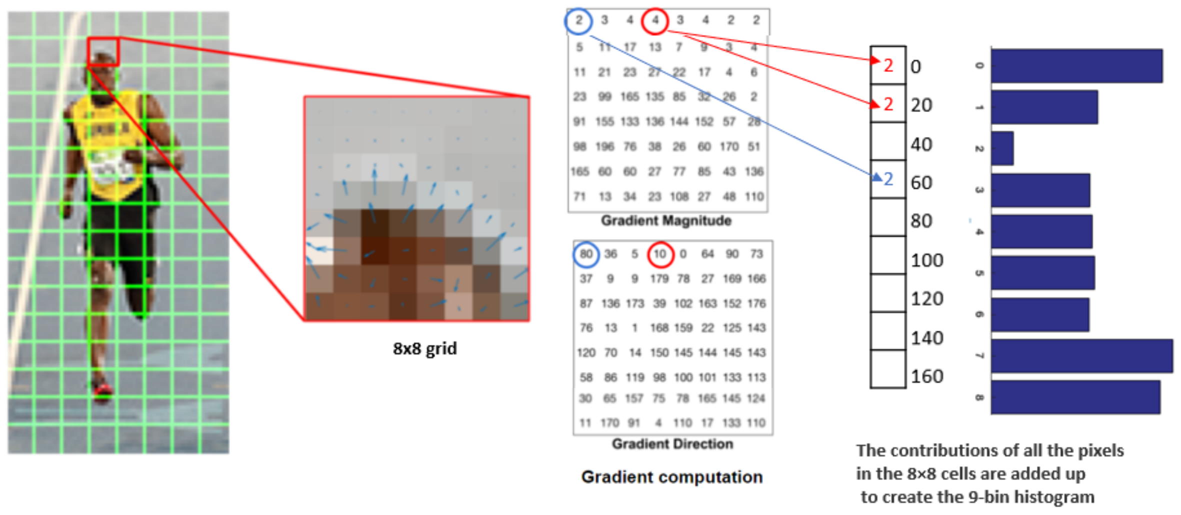

8x8 grid
Gradient Magnitude
Gradient Direction
Gradient computation
0
20
40
60
80
100
120
140
160
The contributions of all the pixels in the 8x8 cells are added up to create the 9-bin histogram

Fig 22: Calculation of HOG of an image

7.4.2 Flow of Action:

The flow of action or optical flow [35,72] highlights the direction of movement of objects of interest in the image frame. Visual techniques to highlight this are to warp a smaller scaled version of the image based on changing pixel values, focusing on the center of the image. Flow of action can be represented in a variety of ways. An example of warped image is shown in Figure 23, referenced from [35]: The flow of action concept has been put to use in the S2VT[32] model, and added to the CNN generated features, before the encoding process of video frames.

One can notice the change in the warped images of the scenery as the clouds arise in the second picture.

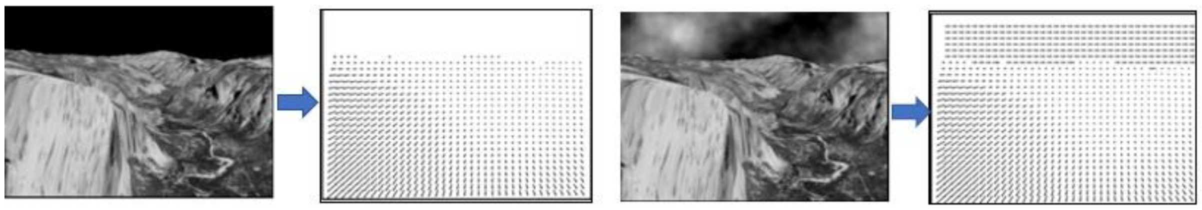


Fig 23: Flow of action highlighted by image warping

Soft Temporal Attention in Video: While soft attention in images corresponded to dividing the image into subregions and weighing the importance of each region in caption generation, the concept is a little different when it comes to videos. In videos[73], instead of regions, we calculate the weighted importance of the temporal features generated by the encoder, at the end of the encoding process. The model learns on which temporal regions to focus on, while decoding the next word in the video caption. The primary requirement for soft attention is having the spatial-temporal features of the video pre-generated. Figure 24 demonstrates a soft temporal attention module for video captioning.

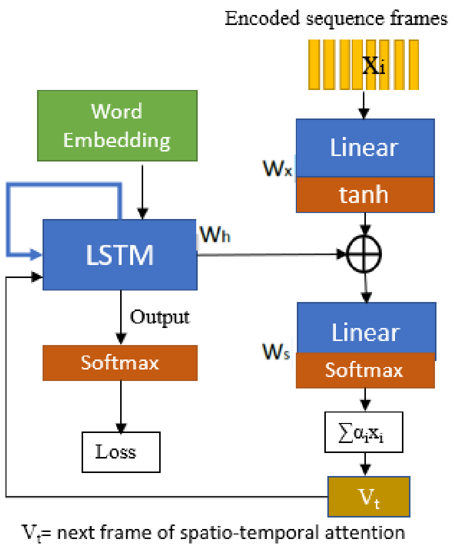


Fig 24: Generic Soft attention mechanism in Video

7.4.3 Scene/Topic Based Attention:

Scene attention inherently means focusing on the region of occurrence or neighborhood of the activity in a video. The neighborhood surroundings of a person or an activity play a significant role in deciding the context and subject of the image frame. Scene based attention in video is similar to the concept of scene based attention in image, but may be computed in several methods. A Dual CNN[37] approach may be used , where one CNN is dedicated to extract the frame background features while the other one extracts the complete video features . Figure 25 shows how to use dual CNNs to generate scene and original image features in a captioning model.

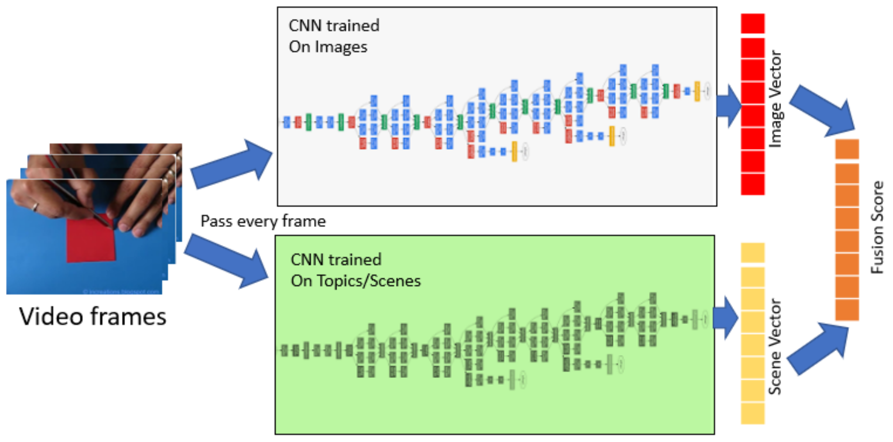


Fig 25: Dual CNN for scene and feature extraction from image frames

Siamese CNN or dual CNN approach has been very popular in researches regarding video descriptions, where two dedicated CNNs have been used in the model architecture, one devoted to spatial image features and the other detecting temporal features like optical flow. For distinguishing two adjacent frames, the displacement vector fields can be calculated and then the sequence of predictions can be averaged out together. This type of architecture may be especially useful when dealing with sports videos.

CHAPTER 8: Video Captioning using Summarization:

8.1 Initial studies:

Long before the advent of proper metrics to evaluate generated sentences, Barbu et al.[33] in 2012, devised a proper method to summarize videos using well framed sentences, and evaluated the results by crowdsourced judgement. Their method incorporated extraction of optical flow of events in the video, clustering of images related to various human poses and training a Hidden Markov Model (HMM) to generate sentences from a video. The model was trained to detect the direction of movements of objects in the video and describe the action using sentences. The performance was evaluated to achieve 49% accuracy in describing events accurately , given best out of the three sentences generated by the model.

For labelling long term temporal dynamics, we can use an LSTM to generate more context aware captions to summarize a video. An example of this may be accumulating sentences like "person making dough", "person baking dough" and "person adding topping", into something like "person is cooking". Although theoretically hard to explain, experimentally it has been proved that deeper RNNs are better , especially at predicting hierarchical sequences of time varying data. Video sequences are a great example of usage of such type of architecture where every scene carries forward the latent meaning of the previous scenes. Video captioning, which requires sequences of videos leading to a sequence of representational words finds a significant usefulness of this architecture. Hence the concept of stacked LSTMs in video captioning were introduced by Venugopalan et al. in their S2VT video captioning model[32].

8.2 Methods:

8.2.1 Stacked LSTM:

Stacked LSTMs [30,31] are multiple LSTMs stacked on top of each other, thus creating a deeper LSTM network. Often each layer of LSTM is responsible for identifying different set of temporal sequences .As we go deeper, an LSTM layer below received a sequence output rather than a single value output from the LSTM above it. This inherently requires the above layers of the LSTM to be of many-to-many architecture. Specifically, one output per input ,in every time step, rather than one output time step for all input time steps.

In a previous work[33] Venugopalan et al. created a video captioner where the most important feature from a feature set of video frames was max-pooled using a CNN and then an LSTM was used to decode the image for caption generation. The model totally ignored the sequence of frames in its approach as the pooling[36] was performed and hence the model was a failure in preserving the entire context of the video. To tackle this, Venugopalan et al. introduced a method where the sequence was preserved, namely the S2VT model which used stacked LSTMs.

8.2.2 Encoder-Decoder Module:

In S2VT the generation of sequences occur in two stages. The first stage is the encoding stage where sequence of video frames are read by the model and the second stage is called the decoding stage, where sequences of words are generated based on the sequence of video frames, encoded by the stacked LSTM. The encoder prepares a compact representation of the entire video while the decoder is for language generation ,preserving the temporal connection between video frames. The reason for keeping the two modules (Encoder and Decoder) separate is that the decoder's output may not be of any use to the encoder in the current process. As the entire information of the video , which is the only current information available, is already available to the encoder, encoder totally independent of the decoder. The architecture of S2VT is described in Figure 26. It uses the same set of LSTM for both encoder and decoder stages. In the encoding stage, only video frame features are fed into the LSTM and as no word vectors are added , the decoder is fed padded blank sequences. In the decoding phase, when the encoder has finished processing sequences, a "start" token word is passed into the decoder , as well as the output from the encoder , to start sentence generation. During this phase the upper LSTM is fed padded sequences, since we do not have requirement of further input from video frames.

There can be two methods of implementing an encoder-decoder approach in Keras library. The first method involves summarizing the entire video using the encoder LSTM's last output unit and using a repeatVector to distribute it among the decoder units, through the timesteps. The second approach involves a many-to-many encoder LSTM where sequences are returned at every timestep of the LSTM and fed into the decoder. Figure 26 describes the second approach.

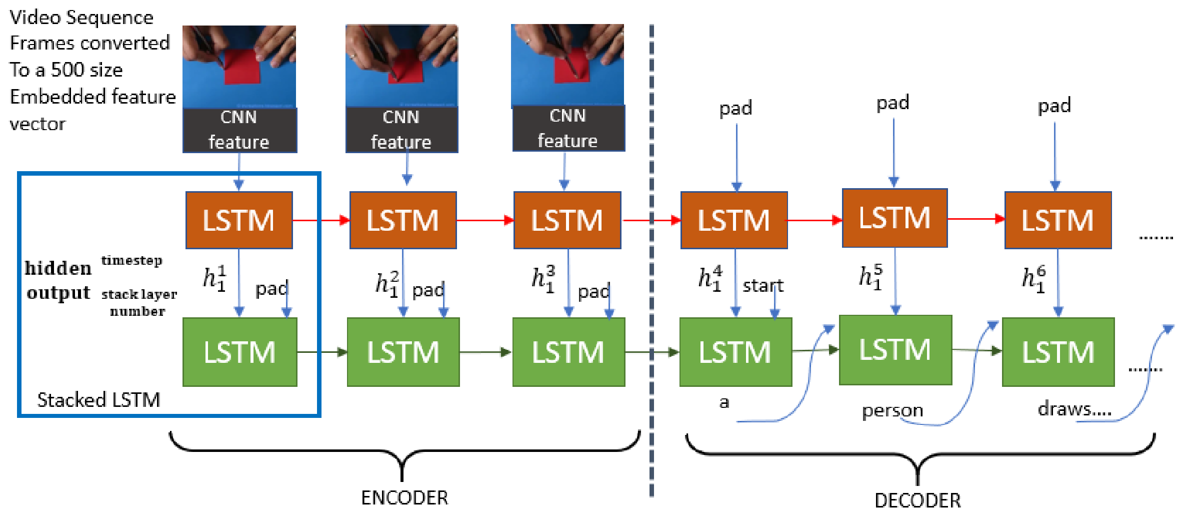


Fig 26: Single Encoder-Decoder Stacked LSTM used in S2VT

Following this, the encoder-decoder approach gained popularity in context of video captioning, for future researches. Variations of the encoder-decoder approach were put forward by many researchers, one such being [37].

Backward temporal dependencies can help in re-framing of those event descriptions that were based on only forward dependencies. To achieve both forward and temporal dependencies , the use of Bidirectional LSTM has been implemented by Peris st al.[37] in his attention based video captioner, ABIVIRNET (Attention Bidirectional Video Recurrent Net). The encoder model ,instead of creating compact sequences of the video, generates a new representation for the video frame features by combining the CNN feature of frames, and the output from each of the two LSTMs in the Bi-LSTM (LSTM output preservers the temporal features with frame interdependency). The decoder gets the combined feature vector and trains a soft attention model which decides the most important feature from the sequence, to be determining the next word prediction. The chosen frame is then passed onto a decoder LSTM to generate the next word. Figure 27 describes the generic diagram of ABIVIRNET. ABIVIRNET also uses a scene attention model . Although the concept of scene or topic based attention is similar to the one discussed in section 3, the method of extraction the scene vector by [37] requires another CNN which is specialized in identifying the topic of image scenes. This feature vector along with the original CNN output , is then fed into the Bi-Directional LSTM encoder for further processing.

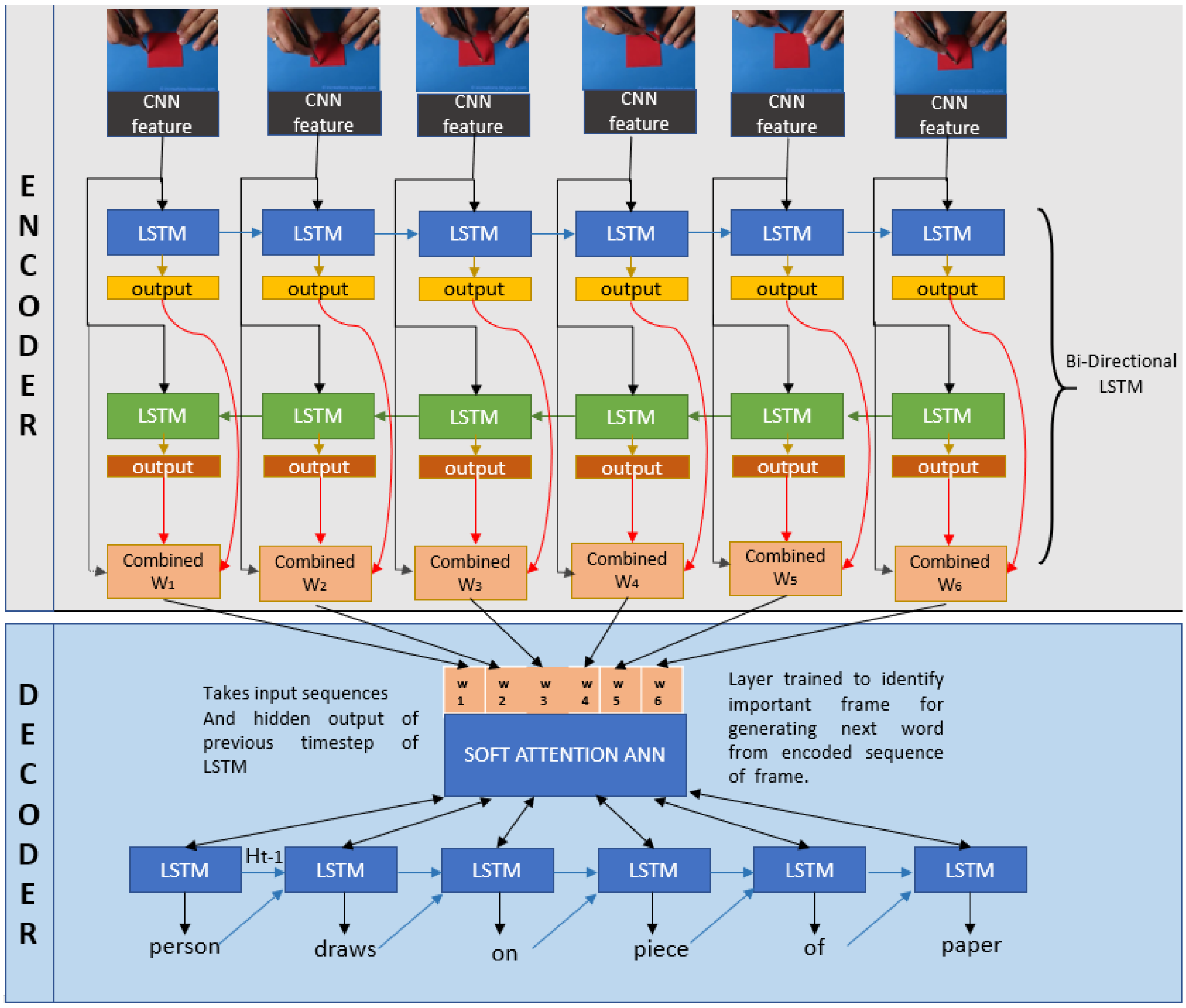
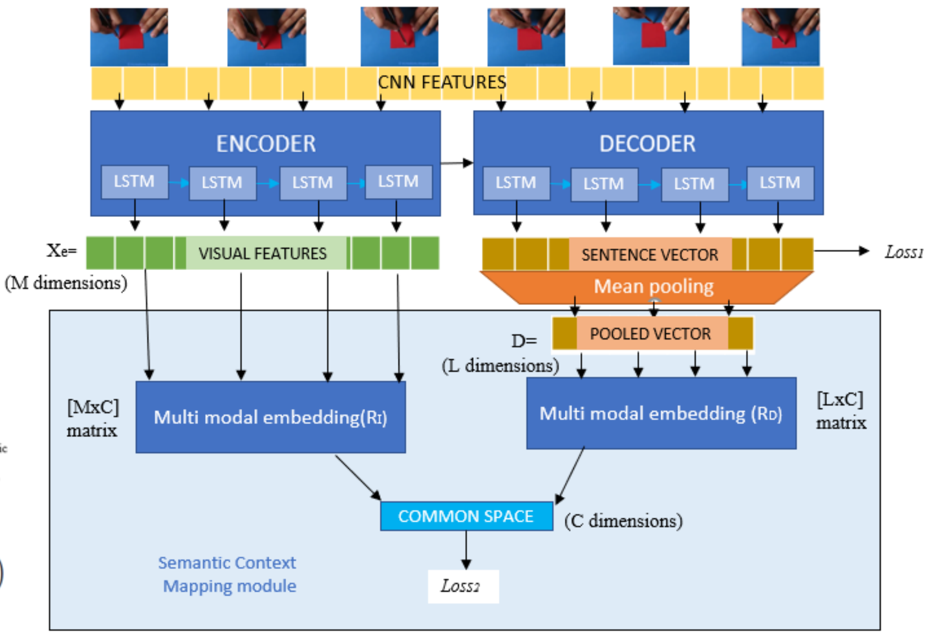


Fig 27: ABIVIRNET-Using Bi-Directional LSTM encoder with Soft attention decoder

8.2.3 Correlation between words and scenes:

Grasping the correlation between encoded sequences and generated words can semantically enhance generated video captions. Achieving this would be possible by having a common-space mapping mechanism for both the sentence vector and the encoded video sequence. Inferring from the differences between the actual and achieved values from the mapped reference, we can estimate a loss function which will contribute to the model’s learning process. Such a training procedure is expected to have better semantic consistency between the video frames and generated words.[35] describes a multi-modal mapping scheme of the generated L dimensional caption sentence(D) to a C dimensional vector space, after the decoding process is over. The encoded M dimensional video sequence (X) , similarly is also mapped to the same C dimension space using a function isomorphic to the previous one. In their implementation , these functions are essentially embedding matrices. The expected visual feature can be calculated as an inverse function based on the two embedding matrices and generated sentence. Difference between the expected and actual value can provide us with another loss measure(L_2) that can contribute to the model's training along with the initial loss(L_1). Meanwhile, L_1 can be calculated as the negative log likelihood of generating words of the sentence, given the previous words. The authors proved this approach to outperform the current state of the art captioners by generating captions with accurate context to the video. The model is summarized in Figure 28. The authors used a generic encoder -decoder model with the semantic consistency module.



$$\left.\begin{aligned} R_I &: R^M \rightarrow A^C \\ R_D &: R^L \rightarrow A^C \end{aligned}\right\} \text{Isomorphic functions } R_I \text{ and } R_D$$

$$R_I\left(\mathbf{X}_e\right) = R_D\left(\mathbf{D}'_{mean}\right)$$

$$\begin{aligned} X_e &= R_I{}^{-1} R_D\left(\mathbf{D}'_{mean}\right) \\ &= R\mathbf{D}'_{mean} \end{aligned}$$

$$Loss_1 = -\sum_{t=1}^{N_d} log(P(d_t|\mathbf{v}_t; d_1, d_2, \cdots, d_{t-1}; \mathbf{E}))$$

$$Loss_2 = \left\|X_e - R\mathbf{D}'_{mean}\right\|_F^2$$

Fig 28: Attention Based captioner with Semantic Consistency

8.2.4 Other Methods:

Fusing multiple attention and learning modules together can benefit a captioning model's performance to a great extent . This has been clearly demonstrated by Shi et al.[78] where a captioner model has been trained to identify boundary segments in a video (segments which are independent from each-other's contexts) with hierarchical language modelling and video prediction along with soft attention mechanisms. They used an encoder-decoder architecture similar to [37] using Bi Directional GRUs to encode the video and then trained two GRUs ,one for video prediction , to determine global context of a video and the other for caption generation using hierarchical modelling (using hidden states of another RNN in current input of current RNN). Such an approach incorporating multiple attention and learning modules , which is similar to generation of an ensemble model , have outperformed many state-of-the-art approaches as we can see from the results discussed in section 4.

CHAPTER 9: VIDEO PARAGRAPH CAPTIONING

Summarizing a complete video often misses out on important scenes in-between the video, that might not be directly related to the summarized caption. Moreover , summarization is performed with attention to only a handful of objects in the video and hence is often biased towards them. A complete video sequence usually consists of several types of activities , which would require description using separate sentences , instead of summarizing the whole video. The methods as described above, targeted describing clips containing single activities and hence could justify their techniques by using summarization. But in reality, completed video description techniques require both :

1. identification of the different activity sequences through time and
2. proper summarization of each sequence, with context to the next one.

This is a challenging task, as ideally it requires linking the current sentence with previously generated sentences , with context the objects involved in the previous activity . Such type of video description with para-phrases is called paragraph captioning. Although successful research, in this topic has been very limited , there are some commendable works which deserve mention and have been described in the following section.

9.1 Dense Video Phrase Captioning:

HRNN: Yu et al.[54] proposed a Hierarchical RNN(HRNN) model for generating paragraph sentences which are in context to the video. The model is divided into two modules: Sentence generator and Paragraph generator. Their approach is very similar to image captioning based approaches as their model mostly generates one sentence per video frame . The model uses two stages of RNNs (GRUs , more specifically). One RNN (RNN 1) is used for predicting the next word of the current sentence being generated for the current frame. The other unit (RNN 2)is used to determine the current state of the complete paragraph , which has been generated based on mean pooled sentence embeddings .Video pool features are generated using an encoder-decoder approach , however, the model also uses spatial soft attention (the same as soft attention applied for an image as described in section 3) which requires incorporating the actual frame of that segment and deciding which segment of the frame to focus on specifically. The soft attention module finds its significance in scenarios where one has to identify very small objects like cups, bowls which appear in cooking videos TACOS dataset. These features are often overlooked when only temporal soft attention is applied over the whole video. To identify action segments like hand movements, the authors also incorporated an optical flow attention module . Hence there are two attention feature channels in the model. During generation of sentence , RNN 1 is biased by the current paragraph state and the previous generated word and its output is passed onto a multimodal embedding layer. The multimodal embedding merges input from RNN1 and the features generated by the attention modules into a 1024 dimensional vector which is used to generate the next word of the sentence . The paragraph generator module is invoked every-time after a complete sentence is generated. It takes input a mean pooling of the word embeddings and the last state of RNN1 to compute the current sentence embedding. RNN2 processes this to generate the overall paragraph state. The paragraph state is only used to bias RNN1 to maintain consistency regarding the context in which the sentences are being generated. The model has been described in Figure 29.

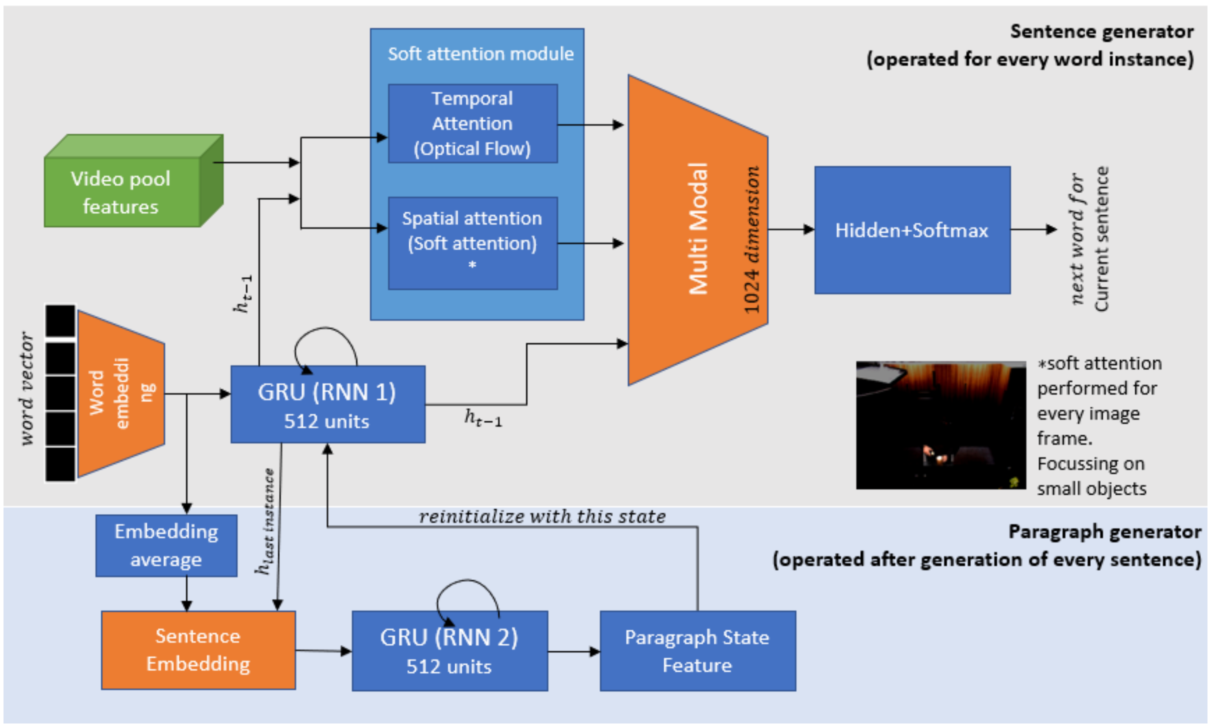

Sentence generator
(operated for every word instance)
Soft attention module
Temporal Attention (Optical Flow)
Spatial attention (Soft attention) *
Video pool features
Multi Modal
1024 dimension
Hidden+Softmax
next word for Current sentence
word vector
Word embedding
h_{t-1}
GRU (RNN 1) 512 units
h_{t-1}
*soft attention performed for every image frame. Focussing on small objects
$h_{last\ instance}$
reinitialize with this state
Paragraph generator
(operated after generation of every sentence)
Embedding average
Sentence Embedding
GRU (RNN 2) 512 units
Paragraph State Feature

Fig 29: HRNN for video paragraph captioning per frame

The HRNN approach purely relies on previously generated sentences to maintain consistency and context awareness. This approach performs well when tested on the TACOS Multilevel dataset , where the context lies only in cooking scenarios. However, this is prone to various challenges if used in a more diverse environment like MSR or MSVD dataset. Applying soft attention for every frame , increases the complexity of the procedure and makes it inconvenient to be applied for devices with limited computation power and memory. Also, event based approaches using DAPs have been found to be more logical than frame based paragraph generators. Figure 30 shows some sample outputs of HRNN in paragraph captions.

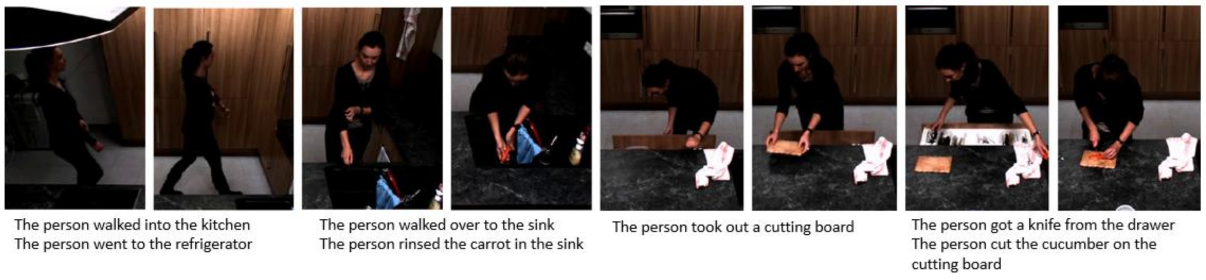

The person walked into the kitchen
The person went to the refrigerator
The person walked over to the sink
The person rinsed the carrot in the sink
The person took out a cutting board
The person got a knife from the drawer
The person cut the cucumber on the cutting board

Fig 30: Paragraph captions generated by HRNN (Source: [54])

9.2 Event Proposals Using DAP:

The primary concern for dense paragraph captioning is getting proper temporal action proposals from the video sequences. These proposals essentially containing the start and end time of a particular action event occurring in the video. As an optimum solution to this situation, Escorcia et al. proposed DAPs(Deep Action Proposals) model[51]. DAPs visualize a window frame of size T which is being run along a whole length video to generate P sequences of total proposals, all generated in a single pass. DAPs can be interpreted as a function which takes input a T-frame video sequence and outputs K-localization proposals denoting the time window in which that event occurs. Determining the window size , ‘T’ and the stride ‘δ’ at which the window is moved through the video is still an optimization problem which requires experimentation. The process of generation of the K action proposals has been summarized below and described in Figure 31. Such networks can be trained on unsegmented video clips , having ground truth time segment proposals for the various events contained in them (eg: THUMOS14[52], MSVD, ActivityNet (137videos))

1. A visual encoder module uses a Conv3D layer to visually encode the sequence frames in the window.
2. A sequence module consisting of LSTMs then take input from the Conv3D output to generate an encoded sequence from the last unit of LSTM at the end of the window time.
3. The encoded sequence is then passes into an event localizer. The event localizer takes input the encoded sequence and generates an output of size K , which correspond to K temporal action proposals in the stream.
4. The prediction module determines the confidence of the chosen proposals as generated from the localizer.

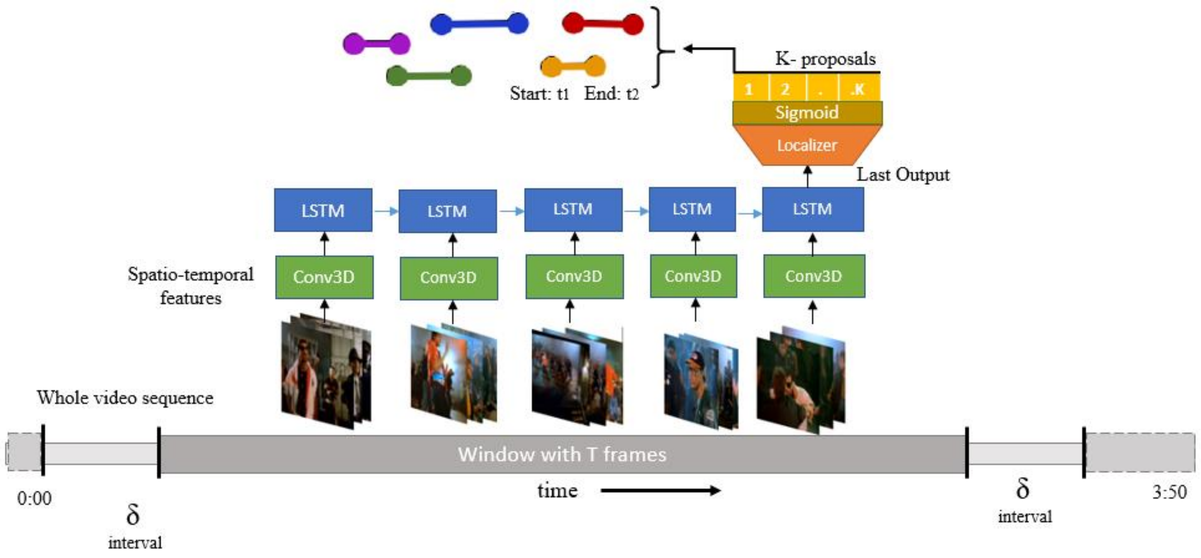


Fig 31: DAP’s Proposal module for finding activity sequences

DAPs have been found to be quite effective in proposing good temporal events and also have efficient runtime performance , with processing rates of 134FPS .on a Titan X GPU. Hence they have found their effective usage in dense video captioning. The critical factor for the performance of DAPs was found to be the window size T. However, the number of proposals , K did not have much impact on the overall performance.

Krishna et al. [50] devised a way to use DAPs for dense video captioning. Their version of DAP operated on multiple strides (1,2 4 and 8) with feature segments taken in as sequences of 16 frames. T=N*16 where N is the number of features and T is the window size for input to DAP. The stride length signifies a measure of the event duration for the current event. Longer strides aim to capture events of longer duration. Their adaptation of DAP is also specialized to produce the visual representation of the event, upon event detection, from the hidden state of the DAP’s LSTM module, for that timestep. The visual features are then fed into a LSTM captioning module to generate the final caption for the event. The authors have also incorporated a context attention module which is fed into the LSTM while captioning. Context attention pays weighted attention to the events prior to and events succeeding the current event to generate phrases with more context to related events. The context attention has found to be more accurate in the experimentation process. Figure 32 describes their model, highlighting the different important modules of the model. The training time took 2 days to converge on a Titan X GPU.

The authors also were responsible for producing the paragraph-caption-annotated version of Activity Net , also known as ActivityNet-Captions, which they have used to evalutate their model.The authors have reported each sentence to describe 36 seconds of a videos (each video being of 120sec duration on average) and all generated sentences to describe 94.6% of the content of the entire video.

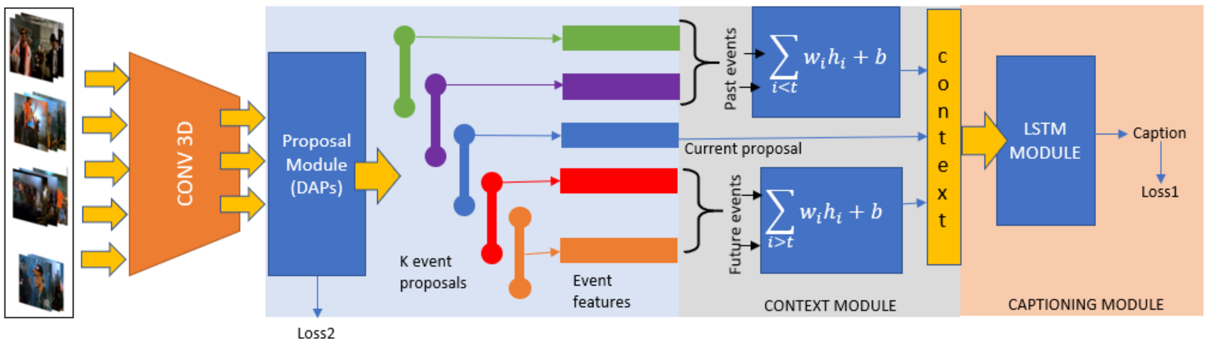


Fig 32: Dense Captioning with DAPs and temporal context attention

Although both are targeted towards generating paraphrase captions, the DAPs based Dense captioner differs from HRNN in both architecture and motive. The HRNN based approach is object centric as it focuses on objects occurring in the video. Also the word generation attention is based on previously generated sentences in the paragraph in HRNN where-as in the DAPs based dense captioner, attention is directly provided in context to the encoded video features of the past and previous events, instead of the generated sentences. The event based attention is more logical since the attention is based purely on the input data instead of generated outputs which may be erroneous.

CHAPTER 10: DATASETS USED FOR VIDEO CAPTIONING

Video description datasets are of two major types. Action Recognition datasets have short length video clips (2 to 15sec) which are classified into one of the many classes defining some activity or human posture. These datasets have a single word describing each video . Video captioning datasets have every video annotated with phrases or sentences that have a detailed summarized description of the event occurring in the video with its objects and surroundings. Datasets targeted towards captioning may even contain clips of longer duration. A major number of these datasets are annotated using the Amazon Mechanical Turk(AMT) services , which is a distributed workforce consisting of people who participate in work outsourced from companies , one of which happens to be annotation of videos. In this section we describe some popular video captioning datasets used widely. Table 4 contains a comparative analysis of the datasets .

MVAD: The Montreal Video Annotation Dataset (M-VAD) is a movie description corpus [40] containing 49,000 short video clips from 92 different movies. The annotations are provided from automatic alignments of Audio Descriptions of the scenes, which are mainly targeted towards blind people for describing a scene to them. In 2019 , Pini et al. extended the MVAD dataset by creating the MVAD-Name dataset, which focuses on identifying the entities in scenes by their names. A face-detection model had to be trained to identify faces from the MVAD dataset, to generate attributes for the MVAD-Name dataset.

MSVD: One of the most popular datasets for benchmarking video captioners, the Microsoft Video Description corpus(MSVD) [74], is a compilation of clips from segments of Youtube videos describing people doing a single activity in that clip. It consists of 1970 such clips, with an overall size of 1.72GB. Each video has varying number of annotations, in multiple languages. There is an average of 41 total annotations per video. In most works, only English captions have been filtered out for usage.

MPII: Similar to MVAD, the MPII-MD dataset[41] consists of 68375 video clips from Hollywood movies, with a single sentence description for each , based on the audio data of the script. The dataset consists of a diverse amount of topics due to the great variety in movie scenes. However, this adds to the fact of making the training process on such a dataset, quite challenging.

MSR Video to Text (MSR-VTT-10K): MSR-VTT(Microsoft -Video to Text) [39] is a huge corpus of 10000 video clips (total duration of 41.2 hrs) corresponding to various diverse topics with 20 annotations per clip, made by multiple AMT workers. The annotations have a 62.7% overlap , when it comes to describing the same video. It is a competent dataset when it comes to benchmarking video captioner models , due to its diverse content .The topics of the dataset vary across 20 categories. The clips from the dataset were prepared by segmenting videos into snapshots and segregating unlinked sequences into a different clip. The train , validation and test split of the dataset is by default set as 6,513 2,990 and 497 respectively.

ActivityNet: ActivityNet[49] consists of 137 untrimmed videos comprising of over 203 types of human activities , averaging at 1.5 different types of activities associated in each video. The total number of clips from the videos is 20k. Every distinguishable activity of the video is marked by their starting and ending time in the video to make the clip. The dasatset mainly consists of classification data and not sentences. In 2017, Krishna et al.[50] annotated the videos of ActivityNet to produce the ActivityNet-Captions dataset, where every significant activity in the video is described in well framed sentence using dense captions, for that specific timeframe. Every clip of the ActivityNet Captions dataset consists of around 1 to 3 sentences with each sentence ranging from 6 to 14 words.

Tacos Multilevel: Due to the limited availability of video-dataset which are annoted using phrase captions, the TACOS multilevel corpus [47] was prepared. The total size of the dataset in high definition quality is 29GB. The dataset content is very specific as it consists of only cooking videos, where most of the activities occur in the same room, but have involvement of different people , performing different activities with kitchen items.This dataset contains 185 long videos ,each with average duration of 6 minutes. Each video has multiple intervals in-between and each interval is described by few sentences. This is called phrased annotation, which describes the activities in the video in detail. There are 524788 sentences across 16145 distinct intervals in total, averaging at 87 intervals per video. The average length of a sentence is 8 words and the total vocabulary size is 2864. The vocabulary size happens to be relatively less , due to the limited context (cooking) in which objects appear. Because of the division of intervals , there are strong temporal dependencies among the interval frames, which needs to be given attention to while designing a captioner model for this dataset. The judgement regarding the similarity in description of the various sentences corresponding to the same video segment have been rated to be 3.27 out of a maximum of 5, for the entire dataset.

Table 4: Comparison between various video datasets used in video captioning

Dataset Name	**Annota-tion type**	**Number of Videos**	**Average Length per video**	**Sentences per video (Average)**	**Topics**	**Total no. of sentences**	**Vocab Size**
MSVD	Summarized	1970	10.2s	41(multilanguages)	Multiple	80827	12594
MVAD	Summarized	46009	6.2s	1-2	Movie Clips	56634	18092
MPII	Summarized	68337	3.9s	1	Movie Clips	68375	21700
MSR-VTT	Summarized	10000	14.8s	20	20 different topics	200000	29316
ActivityNet (Captions)	Phrase Captions	20000	180s	3	Multiple Human Activities	100000	-
Tacos-Multilevel	Phrase-captions	185	6min, 87 intervals per video	284	Cooking	52478	2864

CHAPTER 11: VIDEO CAPTIONING RESULTS COMPARISON

11.1 Summarization Based Results:

Venugopalan et al's LSTM architecture with mean pooling[34] focused on mainly frame based captioning, which lacked context of entire video. It's evaluation was made only on basis of individual image based training with image dataset and hence serves less justice to video-captioning. An enhancement to its architecture, the S2VT[32] performed better with its encoder-decoder stacked LSTM approach , retaining the spatio-temporal features of video data. AbiVirNet's inclusion of Bi-LSTMs in their S2VT proved that Bi-LSTMs are truly capable of summarizing temporal concepts better than normal LSTMs. However most of these approaches have limited support of fixed number of frames while processing a video.. Thus the models may work well for only short length videos.

Multi-modal approaches incorporating language and scene correlations, which are specifically targeted towards optimization of semantics of generated captions, have proven to be highly effective in maximizing a captioner's performance as far as caption scoring metrics like BLEU and METEOR are concerned.

Table 5: Comparison between architecture of various models used in video captioning by summarization

Model	Year	CNN	RNN	Video Encoding	Attention
LSTM-MeanPool[34]	2015	AlexNet	2-Layer LSTM	Mean Pooling	-
Enc-Dec +3D +Soft[73]	2015	GoogleNet+ 3DCNN	LSTM	Encoded Sequences	Soft Attention
S2VT[32]	2015	VGG+ Alexnet(flow)	Stacked LSTM	Encoded Sequences	(Dual CNN) Optical Flow
HRNE +Soft[77]	2015	GoogleNet +3D CNN	Hierarchical RNN	Encoded Sequences	Soft Attention
AbiVirNet[37]	2016	GoogleNet [objects]+ GoogleNet[scenes]	Bi-LSTM	Encoded Sequences	Scene Attention
aLSTM[35]	2017	InceptionV3	Encoder-Decoder LSTM	Encoded Sequences	Soft Attention+ Semantic Crossview Correlation
Boundary Aware+Video Prediction[78]	2018	ResNet 101	2 RNNs(Bi-GRU encoder + GRU decoder)	Encoded Sequences	Soft+ Hierarchical Language Model+ Video Prediction

Table 6: Comparison between performances of various video-captioners (summarization)

Model	Attention	Dataset	B1	B2	B3	B4	M	C
LSTM-MeanPool[34]	-	MSVD	-	-	-	30.7	27.66	-

Enc-Dec +3D +Soft[73]	Soft Attention	MSVD[74]	41.92	-	-	-	29.6	51.67
S2VT[32]	Spatio-Temporal	MSVD	-	-	-	-	29.8	-
AbiViRNet[37]	Scene + Objects	MSVD	52.8	-	-	-	31.3	67.2
HRNE +Soft[77]	Soft Attention	MSVD	79.2	66.3	55.1	43.8	33.1	-
		MVAD	-	-	-	-	6.8	-
aLSTM[35]	Semantic Similarity Mapping	MSVD	**81.8**	**70.8**	**61.1**	**50.8**	**33.3**	-
Boundary Aware+Video Prediction[78]	Soft +Video Prediction+ Hierarchcal Language Model	MSVD	-	-	-	50.3	32.9	**74.3**
		MSR-VTT	-	-	-	39.8	26.4	43.3

11.2 Dense Captioning Based Results

Table 7: Comparison between architectures of various video-captioners (dense)

Model	Year	CNN	RNN	Video Encoding	Attention
HRNN[54]	2016	C3D	Hierarchical GRU	Mean Pooling	Temporal Attention
DAPs[50]	2017	C3D	DAPs+ LSTM	Event Proposals	Past and Future Context (Soft)

Table 8: Comparison between performances of various video-captioners (dense)

Model	Attention	Dataset	B1	B2	B3	B4	M	C
DAPs(Dense) [50]	context	ActivityNet -Captions	26.33	13.98	8.45	5.52	10.03	29.92
HRNN [54]	Sentence embeddings	MSVD	81.5	70.4	60.4	49.9	32.6	65.8
	Sentence embeddings	TACOS Multilevel	60.8	49.6	38.5	30.5	28.7	160.2

Table 7 and 8 describe the performance of dense captioning networks. Existing Dense captioners operate on different datasets and have varying implementations. Hence it is hard to compare them with complete justice, if the test cases are not similar. The DAPs(Dense) captioner is action centric as it is trained on human activity datasets, which makes it more globally applicable than HRNN model which is trained only on cooking videos and hence lacks generality. When trained on the ActivityNet dataset by the authors of [50] , the HRNN was reported to have a CIDER score of 22.4, which is less than the DAPs based approach.

CHAPTER 12: IMPLEMENTATION

12.1 Baseline Image Captioner With Glove Embeddings

12.1.1 Architecture

The baseline captioning(Figure 33) model as designed, takes input the input sentence and the image features to output the next word of the sentence. We have created the dictionary of words by selecting words occurring at least 10 times in the entire set of ground truth caption sentences. The words have been one hot encoded and passed into a Glove embedding module. We have used the Glove 600B embedding which produces an output vector of 200 dimensions. The LSTM function can be framed as, predicting the next word of the sentence given the sequence of previous words of the sentence. Hence, the LSTM takes input of dimension (length of sentence, word embedding) from the Glove embedding layer. Finally the inputs from the LSTM and image features are fused together in a Add layer followed by a Dense layer to produce the next word of the sentence. The words are generated till the predicted word is the end token of the sentence.

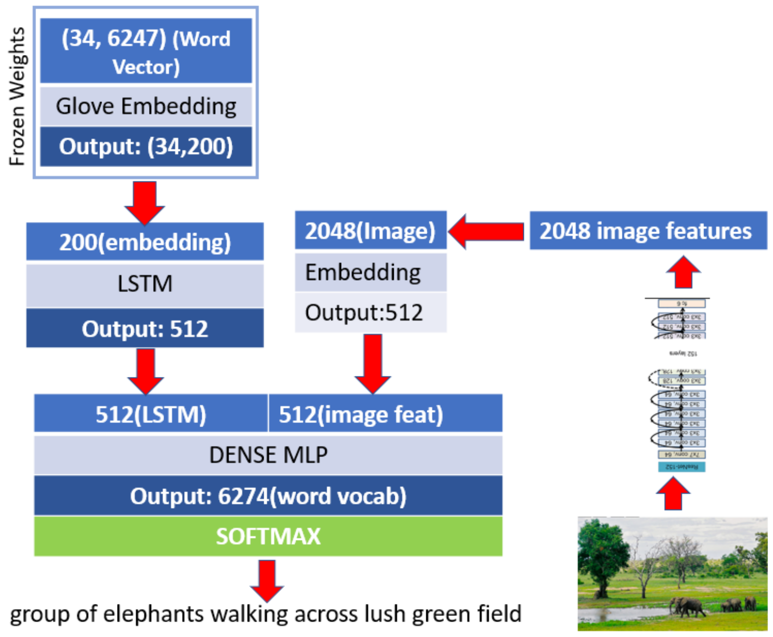
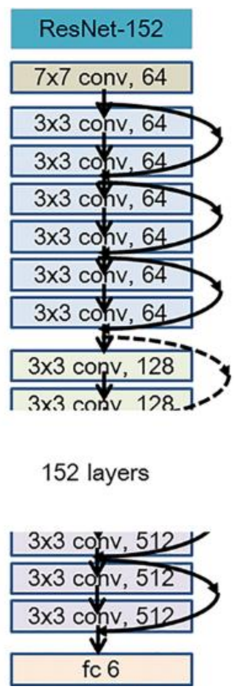


Fig 33: Baseline Glove Captioner

12.2 Image Captioning with Scene Attention

This model has been inspired from Kun Fu's [6] Scene specific image captioner as the model manages to produce great captions using optimal computations , which is very important if we want to caption videos in real time. We pre-process the images to generate feature vectors which are also used to generate scene vectors from a perceptron model, trained based on the image topics. Then the image features and scene vectors together are fed into an embedding input layer of our RNN-LSTM based captioner working in a merge architecture. In every iteration , the LSTM produces the next word of the caption based on the previous word, scene vector and image features.

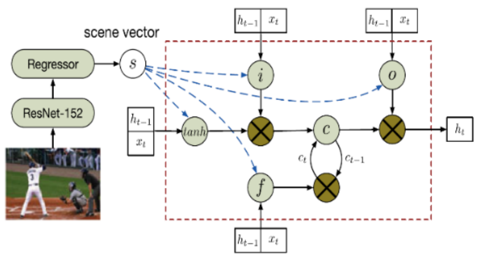
12.2.1 Resnet 152



Our model uses ResNet152 for initial feature extraction from image frames. ResNet152[43] is deep CNN, specialized in semantic segmentation and object detection experiments and hence it is one of the best models for identifying different objects in an image precisely. Leveraging the power of its 152 layers, it even won the COCO 2015 object detection competition. Caffe version of ResNet is publicly available from this link. Caffe is a python library specializing in deep learning models and models from contributers all around the world can be found in its “Model Zoo” which is a great source of research for downloading pre-trained models. We use Caffe’s Model zoo version of ResNet152 and construct the model from its weights and prototext using Caffe. We extract output from the pool5 layer of ResNet152 which generates features of dimension 1x2048. We have used MSCOCO dataset consisting of 82000 images from various categories to train our model. Every image has been fed into the ResNet152 to generate 2048 features per image and all the features have been saved in a pickle file in python for quick and easy access later on. The vocabulary has been generated by tokenizing captions available from MSCOCO image captions and contains 8843 words. The vocabulary consists of a JSON file with the mappings and inverse mappings of words and their respective numerical values.

12.2.2 Biasing with Scene Attention

The scene vectors are computed by initially clustering images using an unsupervised algorithm. We have used Latent Dirichlet Allocation(LDA)[61] like to extract a 80 dimensional scene vector which contains the probability of that particular image belonging to one of the 80 topics we have in our database. The scene vectors computed here are based on the ground truth mage captions and these are used for training a multilayer perceptron(MLP) to produce scene vectors from other testing data during testing. The training samples for the MLP are the feature of images from the same training dataset of MSCOCO with the target outputs being the LDA-inferred scene vectors. The process is similar to the one described in section 2.3.1. For imitating the clustering algorithm to generate the scene vector from testing images, we use a multilayer perceptron with two hidden layers with the sizes of 1,024 and 1024. We use SoftMax in the last layer and tanh function for others. The predicted scene-context vector is a continuous-valued vector, representing soft assignments of scenes or topics.



The LSTM unit which is our caption generator, takes in input both the image feature and its scene vector. The scene vector , in every iteration will bias the LSTM prediction, explicitly reminding it of the image context, while predicting the next word in the current iteration. An embedding layer compresses the word vector of 8843 to a 512 dimension before feeding it to the LSTM . The LSTM produces a 512 dimensional vector as output. This along with the image feature is merged together to be the input for the final dense MLP layer(output dim 8843) that produces the next word . The final model looks like what is described in Figure 34

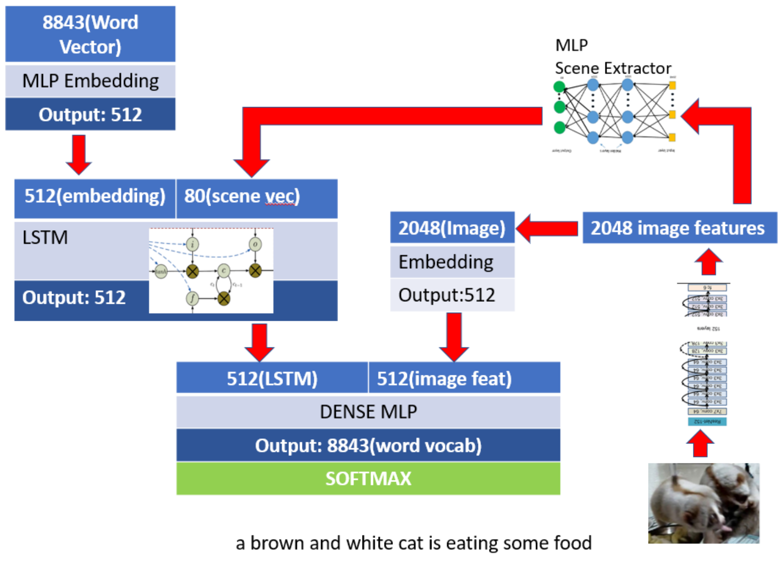


Fig 34: Scene Attention Based Model

12.3 Training and Results

12.3.1 Pre-processing

The training of both captioners has been done using MSCOCO dataset. The captions from the dataset have been filtered to not have any punctuations, special characters and numerical digits . The vocabulary has been prepared by tokenizing words from sentences. Less frequent words have been removed as they do not serve ay purpose in the training set due to their low frequency. Two vocabularies are prepared by selecting words occurring at least 10 time and 5 times in the dataset, having sizes 6247 and 8843 respectively. We notice that the smaller vocabulary does not hamper the quality of captions in any way, during experimentation.

For the RGB images , they have been resized to 224x224 pixels and passed into a ResNet152 to extract their 2048 dimensional representational feature vector and stored in a pickle file , with the image ids. For the scene attention model, we have similarly extracted the 80 dimensional scene vector using the trained MLP .

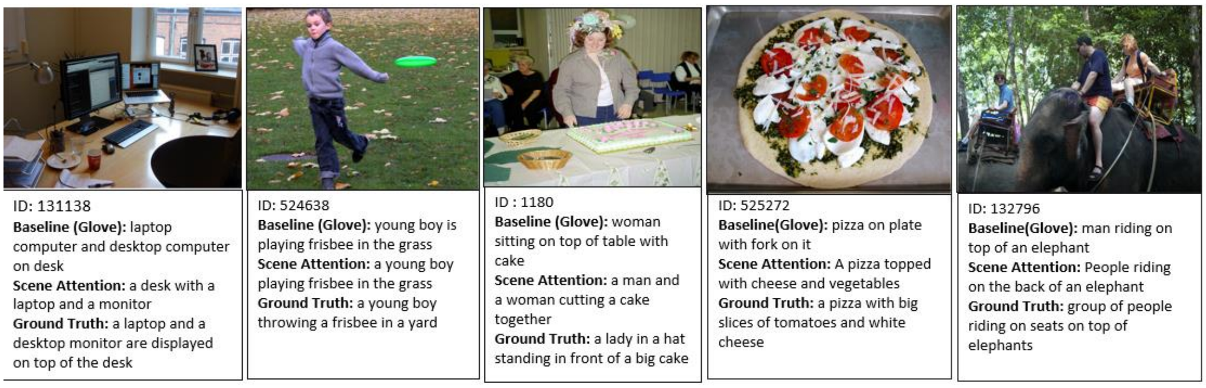
12.3.2 Results

Training time took 3 hours for the glove-baseline model on a device operating an NVIDIA 1060GTX graphics card and Intel core i7 (8th gen) processor. The Scene attention based model took 5 hours on a Nvidia K10 GPU to be trained. The training times are approximated based on the time taken by them to produce satisfactory captions on the training dataset. Following are the results from the experiments. We find significant improvement in the scene model from the baseline model. The Glove model performs on par with scene attention although scene attention scores slightly better than glove model.

Table 9: Experimental Results

Model	Attention	B1	B2	B3	B4	M	C
Baseline	-	67.2	49	33.8	25	22.3	80.5
Glove	-	68.9	51	37.6	27.5	23.2	85.2
SS[6]	Scene	71.2	53.6	39.4	28.9	24.1	89
Glove+Scene	Scene	72	54.1	40.2	29.1	24	89.6

The ensemble model by replacing the embedding layer of the scene attention model, with Glove has effective improvement on the scores as expected.Figure 35 shows the performance of two models tested by us, namely a Glove Embedding based baseline and a scene attention[6] based captioner, on the MSCOCO dataset. The IDs of the image are also provided for verification. It can be noted that due to the glove embeddings, even a baseline model can perform on par with an attention based model. The Glove based model in some instances, generates better captions than the scene attention model.



ID: 131138
Baseline (Glove): laptop computer and desktop computer on desk
Scene Attention: a desk with a laptop and a monitor
Ground Truth: a laptop and a desktop monitor are displayed on top of the desk

ID: 524638
Baseline (Glove): young boy is playing frisbee in the grass
Scene Attention: a young boy playing frisbee in the grass
Ground Truth: a young boy throwing a frisbee in a yard

ID : 1180
Baseline (Glove): woman sitting on top of table with cake
Scene Attention: a man and a woman cutting a cake together
Ground Truth: a lady in a hat standing in front of a big cake

ID: 525272
Baseline(Glove): pizza on plate with fork on it
Scene Attention: A pizza topped with cheese and vegetables
Ground Truth: a pizza with big slices of tomatoes and white cheese

ID: 132796
Baseline(Glove): man riding on top of an elephant
Scene Attention: People riding on the back of an elephant
Ground Truth: group of people riding on seats on top of elephants

Fig 35: Generated captions by Glove Embedded Baseline model and a Scene Attention based model on some MSCOCO testing images

12.4 Basic Paragraph-Video Captioning with Scene Attention

We have used our image captioning model for video captioning too. Our trained MSCOCO model was tested on all 1970 videos of MSVD data corpus. Since MSVD videos do not have labelling of every frame and just have a summary of the whole video, we have not been able to compare each caption generated in every frame . But our model has been able to clearly describe the whole videos whatsoever, with a few minor exceptions. To make videos suitable for analysis in our network, we first process every video into frames taken at 1 second interval using Python's open cv library and keep them in a folder containing their video id. Then the frames are each passed into the Resnet152 to generate 2048 sized features. All the video feature frames are kept in a pickle file.

In Figure 36, Step 1 describes the above process in the following diagram. For step 2, we then predict the scene vectors of the processed frames by passing every feature vector of every frame through our MLP for scene vector. We save the scene vectors separately in another pickle file. Finally in step 3 , we clean the ground truth captions provided in the MSVD csv file for every video. Every video has at least 4 labels and leach label is made in various languages. We filter the English labels only, keeping only lowercase alphabetical content, and discard the later. It is to be noted that MSVD also contains a lot of jargon captions which need to be discarded if comparisons are to be made with generated captions and ground truth. Here Pickle datafiles have been used because they preserve the data-structure of the content inside it and can easily be loaded directly into similar structure using the load function. Other filetypes like h5py, which are good for storing ML models , have various limitations when it comes to data structures as h5py doesn't support any structure other that size-1 Numpy arrays. Pickle however can easy store a list of dictionaries which are being produced in our scenario.

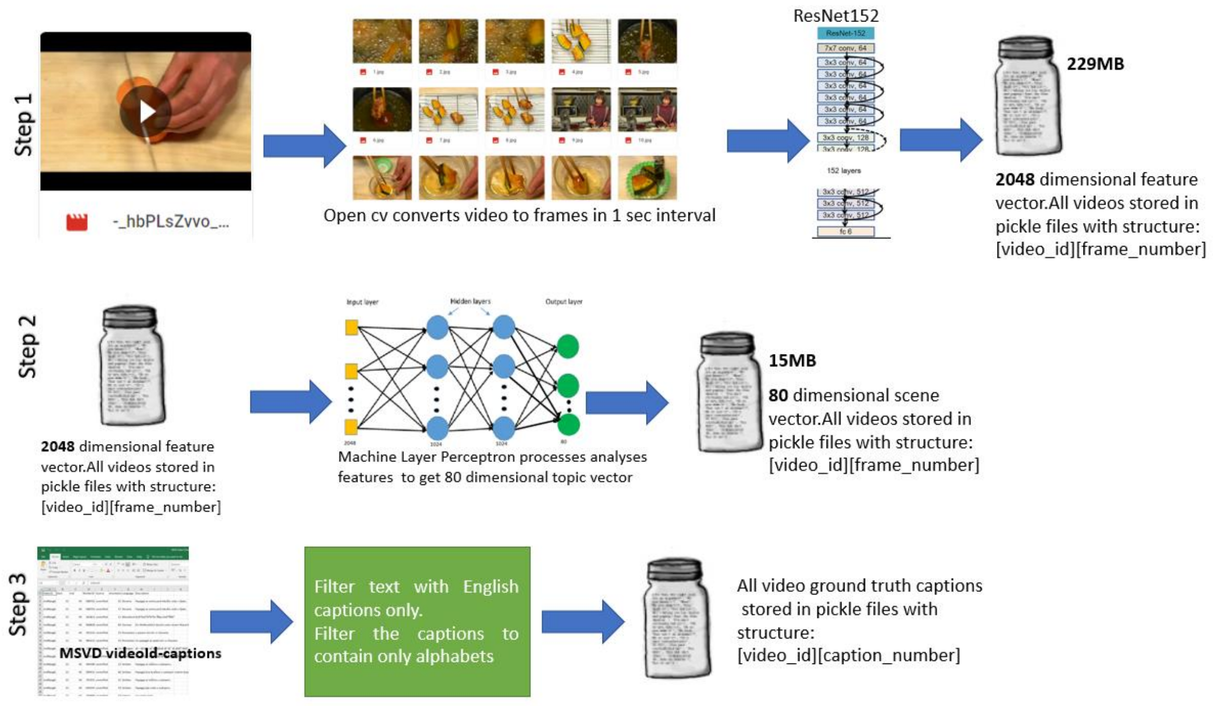
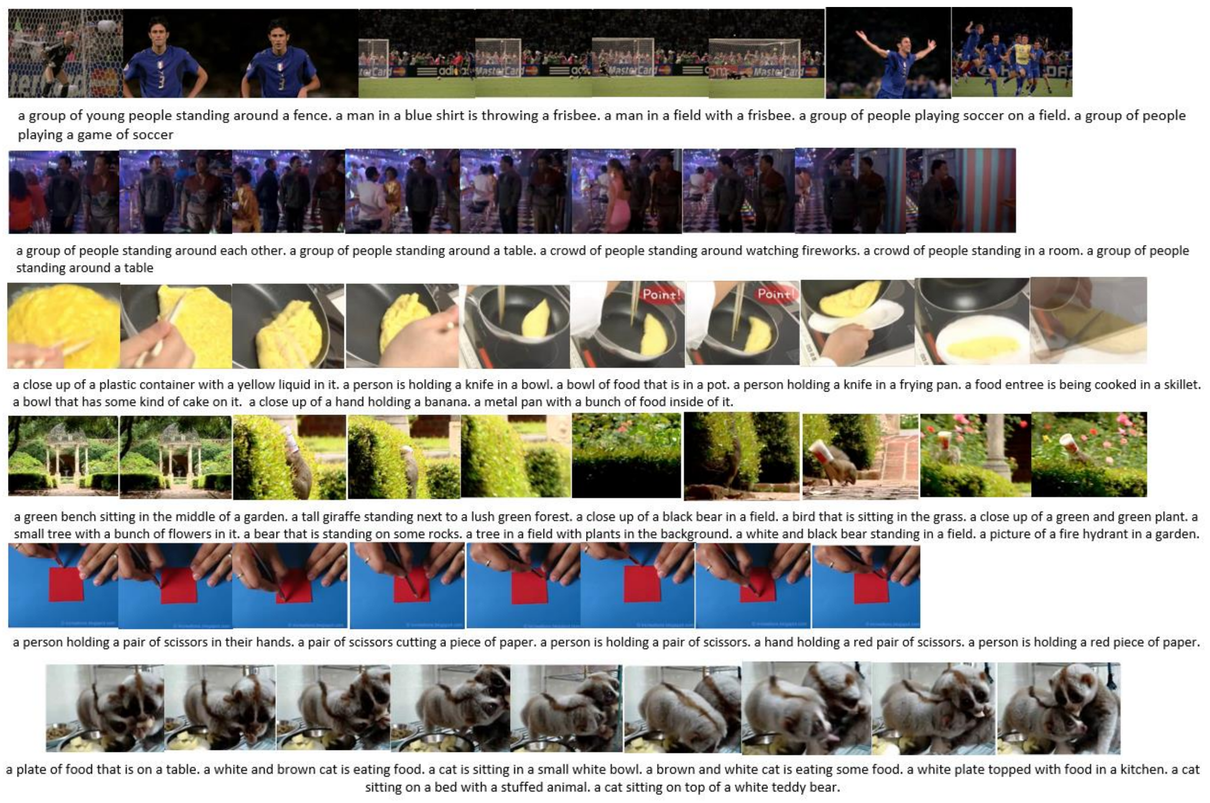

Step 1
-_hbPLsZvvo_...
Open cv converts video to frames in 1 sec interval
ResNet152
229MB
2048 dimensional feature vector.All videos stored in pickle files with structure: [video_id][frame_number]
Step 2
2048 dimensional feature vector.All videos stored in pickle files with structure: [video_id][frame_number]
Input layer
Hidden layers
Output layer
Machine Layer Perceptron processes analyses features to get 80 dimensional topic vector
15MB
80 dimensional scene vector.All videos stored in pickle files with structure: [video_id][frame_number]
Step 3
MSVD videoId-captions
Filter text with English captions only.
Filter the captions to contain only alphabets
All video ground truth captions stored in pickle files with structure: [video_id][caption_number]

Fig 36: Preprocessing MSVD videos for paragraph captioning

Figure 37 summarizes the results of the video captioner. The drawbacks of such an implementation has been discussed in section 7.1.



a group of young people standing around a fence. a man in a blue shirt is throwing a frisbee. a man in a field with a frisbee. a group of people playing soccer on a field. a group of people playing a game of soccer

a group of people standing around each other. a group of people standing around a table. a crowd of people standing around watching fireworks. a crowd of people standing in a room. a group of people standing around a table

a close up of a plastic container with a yellow liquid in it. a person is holding a knife in a bowl. a bowl of food that is in a pot. a person holding a knife in a frying pan. a food entree is being cooked in a skillet. a bowl that has some kind of cake on it. a close up of a hand holding a banana. a metal pan with a bunch of food inside of it.

a green bench sitting in the middle of a garden. a tall giraffe standing next to a lush green forest. a close up of a black bear in a field. a bird that is sitting in the grass. a close up of a green and green plant. a small tree with a bunch of flowers in it. a bear that is standing on some rocks. a tree in a field with plants in the background. a white and black bear standing in a field. a picture of a fire hydrant in a garden.

a person holding a pair of scissors in their hands. a pair of scissors cutting a piece of paper. a person is holding a pair of scissors. a hand holding a red pair of scissors. a person is holding a red piece of paper.

a plate of food that is on a table. a white and brown cat is eating food. a cat is sitting in a small white bowl. a brown and white cat is eating some food. a white plate topped with food in a kitchen. a cat sitting on a bed with a stuffed animal. a cat sitting on top of a white teddy bear.

Fig 37: Video paragraph captions generated by the model

12.5 Event Change trigger From Optical Flow (Online DAP)

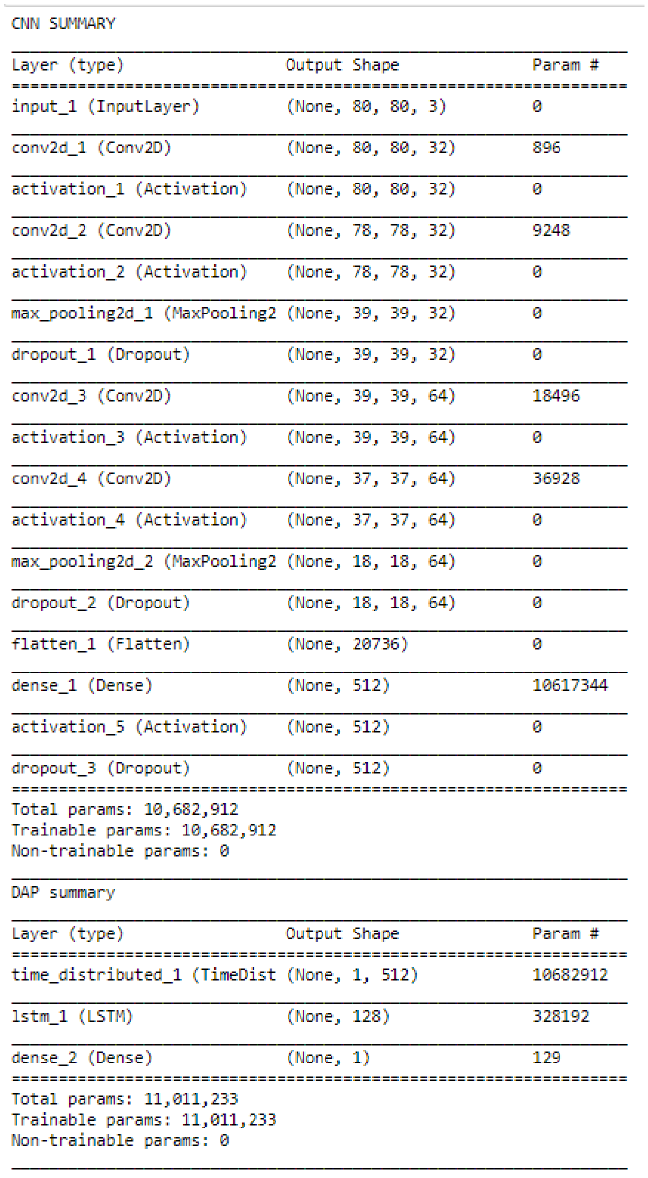
DAPs require a lot of memory and computation power and also suffer from their drawback of not being able to support event proposals for online video. To cope with this limitation of DAP, we need to process the video online, frame by frame. As such, the entire 3D spatio-temporal data is not available to us and hence using a Conv3D to parse the video is not rational. To cope with online videos, we propose an event trigger mechanism. Suppose the first 10 frames of the video correspond to a person cooking and from 11th frame onwards, we find that the person is walking away through the door. Captions from the first event have little in common with the captions to be generated for the second event and hence these two events can be treated disjoint. So, we need to assign a trigger to frame 11 to detect the change in event. The significance of detecting event change can help in the generation of online video captions. The change in event will trigger that the current event might not have anything to do with the past events and hence , during captioning, the caption will be free from any bias generated from the past events.

12.5.1 A possible architecture for Event Trigger

We propose that a small 2D convolutional network be used to extract the per-frame video flow of action features. For every timestep, the last k (value of k being 3 to 5 favorably) frame features can be encoded by an LSTM and outputted for detection of any event change , for the last k frames. This boils down to the usage of a CNN -LSTM model for detection of event changes in the video, where the CNN encodes the last k frames of the video and the LSTM determines whether there is any event change in these frames (binary output of 0 or 1).

Event triggering can also help us to efficiently break down the video into segments and extract encoded sequences of each segment disjointly. This will greatly enhance the processing performance of the video , for captioning.

Following model in Figure 38 summarizes a possible implementation of an event trigger. The CNN used for the purpose is a small CNN which can be used to identify CIFAR images. The FCN layer is removed and replaced with Embedding layer outputting 512-dimensional vector to be inputted into the online LSTM. Normal LSTMs require a sequence of inputs, to be operated on and usually output a summarization on the entire sequence. However, our model concerns processing video frames online, as they arrive one frame at a time. To support this challenging architecture, a stateful LSTM is required, which realizes that the previous states are not to be forgotten while processing the current frame of the video. A stateful LSTM has to know the batch size of the input, so that it may preserver the states for that batch only and after processing it can reset the states for the next batch. In our case, for online learning, the batch size is set to be 1 as, frames arrive one at a time. We reset the states of the LSTM after all sequences of a particular video have been processed. The LSTM outputs are transmitted to a Dense layer with a single output (0 or 1). The LSTM model used here is a one to one model. However, for better summarization, one may input a sequence of last 3 feature-frames of the video to the LSTM, thus designing a Many-to-One model. However, implementing such an architecture is challenging, if the CNN and LSTM are to be trained in the same model. Separately extracting features from a trained CNN like FlowNet [86,87] and inputting them into an LSTM may solve the issue.



```
CNN SUMMARY
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 80, 80, 3)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 80, 80, 32)        896
_________________________________________________________________
activation_1 (Activation)    (None, 80, 80, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 78, 78, 32)        9248
_________________________________________________________________
activation_2 (Activation)    (None, 78, 78, 32)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 39, 39, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 39, 39, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 39, 39, 64)        18496
_________________________________________________________________
activation_3 (Activation)    (None, 39, 39, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 37, 37, 64)        36928
_________________________________________________________________
activation_4 (Activation)    (None, 37, 37, 64)        0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 18, 18, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 18, 18, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 20736)             0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               10617344
_________________________________________________________________
activation_5 (Activation)    (None, 512)               0
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
=================================================================
Total params: 10,682,912
Trainable params: 10,682,912
Non-trainable params: 0
_________________________________________________________________
DAP summary
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
time_distributed_1 (TimeDist (None, 1, 512)            10682912
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               328192
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 11,011,233
Trainable params: 11,011,233
Non-trainable params: 0
_________________________________________________________________
```

Fig 38: Proposed architecture for online Event Trigger

12.5.2 Preparing Data and Training an event trigger model:

Frames can be extracted from a video at 1 FPS and from for each of the frame taken at a timestep t, if ‘t’ has been marked by an annotator to be a change of event, for the frame extracted at time t, we append a 1 to the sequence of triggers ,while for all other frames , a 0 is appended to the trigger sequence. The ActivityNet captions dataset is annotated with event begin an ending timing for every main event in the video. We use this dataset to create the event triggers. For every sequence start or end timestamp, we append a 1 to the trigger list at the frame taken at that timestamp and keep all other trigger values as 0 (for every other frame) in the list. A sample from the processed training dataset would look like the one in Figure 39.

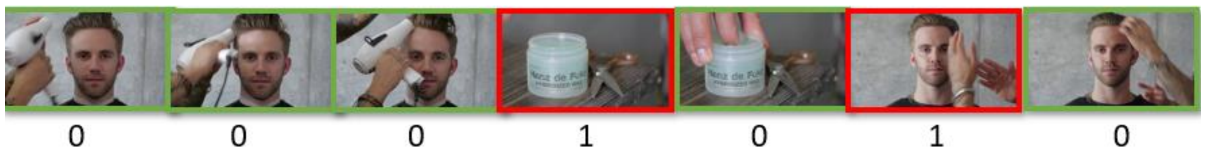


Fig 39: Event triggers in the dataset

CHAPTER 13: CONCLUSION

Image captioning has a lot of potential in the field of image and media search, given the perfection it has achieved over the years. With the increase in number of handheld devices like smartphones, it is important that such technology is made available to mobile devices by optimizing for support on hardware with lesser on-board computation power. Current implementations of captioning models, both image and video, although accurate, require high resource compensation to generate accurate and well framed captions and most models have been tested using sophisticated GPUs like Titan X and K80. However, from surveying recent model architectures, we conclude that caption generation can indeed be carried out in a more optimized way and to achieve that we propose the following:

13.1 Possible improvements in generic captioning models, applicable to both image and video:

- Using a faster and more lightweight CNN like MobileNet V2 to generate image features. MobileNet v2 is 40 times smaller than vGG16 and 20 times smaller than ResNet 152 and hence , object detection is likely to occur faster using MobileNets.
- Shrinking the vocabulary to only contain words which have significant occurrence in the caption training data, as word having lesser frequencies are more likely to have inadequate amount of training samples for optimum identification. This will also help in shrinking the size of embedding and multimodal word generation layers by a huge fraction. Glove embeddings have gained popularity for serving as a pre-trained embedding layer for word vectors, also preserving the word similarities and latent meanings. Pre-trained models save a lot of time in the training procedure as the embedding layer is one of the largest training modules of a captioning model. Moreover, Glove embeddings with a reduced dictionary can shrink the size of the embedding module by as much as 75%, without affecting the model's performance. For example, a normal 8843 size vocab may be embedded in a 512-dimension vector under normal circumstances. However, since Glove is a pretrained network with 200 and 300-dimensional word vectors, we can afford to further shrink the embedding layer to generate words having 200 dimensional features. A shrined vocabulary of size 6000 with each word having an embedding of 200 dimensions (6000*200) hence reduces the previous network size by 73.4%.
- Both Soft attention and Hard attention use a lot of computational power to generate proper attention features and this can be hard on the processor or GPU for small devices. Hence instead of using soft or hard attention, Scene attention from [6] can be a good alternative to provide attention of background context, in caption generation. Scene attention can be computed in various ways like image clustering or topics clustering or by training another CNN to identify scenes in the background of an image and concatenating image features with the topic features generated by the scene CNN.
- Models can be optimized using Multiple Instance Learning which focuses on retraining the captioner on sentences whose attributes have been missed by the generated caption. This will ensure that the model can extract all meaningful concepts from an image.
- Traditional training methods intuitively are teaching networks to identify relationships between objects occurring in the scenes portrayed in the images. Thus, some models often generate outputs which may be not in context to the image under test, but may have

occurred in previous training images, which has led the model to think that such objects co-occur in most scenarios. Re-training with datasets like OOC can help the model get rid of mis-inferred references of non-occurring objects in an image and teach it to be more object centric.

13.2 Possible improvements in video-captioners:

Video Captioning systems being more challenging and computationally expensive, are a lesser explored research area when compared to image captioning. One of the most challenging tasks in video captioning is processing the video data such that the spatio-temporal features are preserved in the sequences. Most models use a 3D convolutional network for this, but if computational requirements are to be reduced, 3D CNNs are not a feasible option. However, 2D convolutions are not able to preserve temporal dependencies. Moreover, in video paragraph captioning, the video needs to be processed in segments, where each segment has their own context and contributes to that particular region of the paragraph. Determining the strategically correct segments from a complete video is a challenging task also. Taking all these matters into consideration we propose the following to optimize video captioning systems for less computationally capable systems:

- For segregating video segments from a complete video, small CNNs may be used to extract temporal trajectories from small video frames of 80x80 dimension and then passed into a joint RNN, similar to the one used DAPs, to output the detection of a frame with different context than the current one. Optical flow features of two different frames are to contain very different features from one another and so it should be easy to train a small classifier network to distinguish such features from adjacent video frames. Instead of comparing two simultaneous frames side by side, having an RNN keep track of the flow sequences would be a more conventionally feasible way for detecting change in context. The dimension of trajectory features produced by the small CNN should be small enough to not burden the system, yet be capable of making proper segment proposals. Thus, the overall system, can be an optimized improvement over DAPs and achieve similar results, when trained with datasets similar to the unsegmented ActivityNet video-dataset, with event proposals marked in the training set.
- Instead of using 3D CNN, a 2D CNN can be used to extract spatial information from the video frames. In order to make up for the temporal dependencies, we can have a multi-modal RNN which takes attention from optical flow features and current frames to generate ideal captions. This will also enable the network to operate on variable length input, by not constraining the network to contain fixed number of frames as data representation need not be 3D anymore. This can enable the network to caption online videos also which contemporary video captioning networks still fail to achieve, as they require all concerned frames to be present in the input from the start.
- To support captioning of online videos, we need to encode the video data frames on the go. This requires a facility where encoding of a complete sequence of frames, which are in context to each other is performed by an RNN, till a frame with a different context is encountered. Here RNNs have the upper-hand as they are capable of dealing with data of varying dimensions by using padding. Stacked RNNs can achieve this, where the upper layer of RNNs will be responsible for encoding sequences and passing the output of the sequences (from the hidden layer) to the lower RNNs. The lower RNNs may be responsible

for decoding the sequences into the corresponding captions, wherever necessary. The technique is similar to the one proposed by Yu et al. [54], but depends on the input encoded-scene and optical flow instead of generated words, to decide the captions.

Captioning models can be optimized for better framed sentences in a plethora of ways, with emerging technologies like Reinforcement learning and GANs. However, optimizing the performance of captioners should be an important aspect of research as well. Automatic captioning is a very broad field of research which can benefit the media industry and search engines to a huge extent and while existing techniques of captioning have reached a bottleneck, they are yet to reach the pinnacle.

CHAPTER 14: REFERENCES

1. Freitag, M., & Al-Onaizan, Y. (2017). Beam search strategies for neural machine translation. *arXiv preprint arXiv:1702.01806*.
2. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).
3. L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 11, pp. 1254–1259, 1998.
4. C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry" in Matters of Intelligence, 1987, pp.115–141
5. Cornia, M., Baraldi, L., Serra, G., & Cucchiara, R. (2018). Predicting human eye fixations via an lstm-based saliency attentive model. *IEEE Transactions on Image Processing*, *27*(10), 5142-5154.
6. Fu, K., Jin, J., Cui, R., Sha, F., & Zhang, C. (2017). Aligning where to see and what to tell: Image captioning with region-based attention and scene-specific contexts. IEEE transactions on pattern analysis and machine intelligence, 39(12), 2321-2334.
7. Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, *104*(2), 154-171.
8. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).
9. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).
10. Fu, K., Li, J., Jin, J., & Zhang, C. (2018). Image-text surgery: Efficient concept learning in image captioning by generating pseudopairs. IEEE transactions on neural networks and learning systems, (99), 1-12.
11. Zhang, M., Yang, Y., Zhang, H., Ji, Y., Shen, H. T., & Chua, T. S. (2019). More is better: Precise and detailed image captioning using online positive recall and missing concepts mining. IEEE Transactions on Image Processing, 28(1), 32-44.
12. Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3156-3164).
13. Cornia, M., Baraldi, L., Serra, G., & Cucchiara, R. (2018). Paying more attention to saliency: Image captioning with saliency and context attention. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 14(2), 48.
14. Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching machines to read and comprehend. In Advances in Neural Information Processing Systems (pp. 1693-1701).
15. Mnih, V., Heess, N., & Graves, A. (2014). Recurrent models of visual attention. In Advances in neural information processing systems (pp. 2204-2212)
16. Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., & Wierstra, D. (2015). Draw: A recurrent neural network for image generation. arXiv preprint arXiv:1502.04623.

17. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).
18. Melnyk, I., Sercu, T., Dognin, P. L., Ross, J., & Mroueh, Y. (2018). Improved image captioning with adversarial semantic alignment. arXiv preprint arXiv:1805.00063
19. Bo Dai, Dahua Lin, Raquel Urtasun, and Sanja Fidler : Towards diverse and natural image descriptions via a conditional GAN. ICCV, 2017.
20. Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele : Speaking the same language: Matching machine to human captions by adversarial training ICCV, 2017.
21. Myung Jinchoi, Antonio Torralba, and Alan S. Willsky. Context models and out-of-context objects, 2012.
22. Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.
23. L. Fei-Fei, ImageNet: crowdsourcing, benchmarking & other cool things, CMU VASC Seminar, March, 2010
24. Ferraro, F., Mostafazadeh, N., Vanderwende, L., Devlin, J., Galley, M., & Mitchell, M. (2015). A survey of current datasets for vision and language research. arXiv preprint arXiv:1506.06833.
25. Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. Transactions of the Association for Computational Linguistics, 2:67–78
26. M. Jiang, S. Huang, J. Duan, and Q. Zhao, "SALICON: Saliency in context," in IEEE International Conference on Computer Vision and Pattern Recognition, 2015.
27. T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in IEEE International Conference on Computer Vision, 2009.
28. T. Judd, F. Durand, and A. Torralba, "A benchmark of computational models of saliency to predict human fixations," in MIT Technical Report, 2012.
29. A. Borji and L. Itti, "CAT2000: A Large Scale Fixation Dataset for Boosting Saliency Research," in IEEE International Conference on Computer Vision and Pattern Recognition Workshops, 2015
30. Pascanu, R., Gulcehre, C., Cho, K., & Bengio, Y. (2013). How to construct deep recurrent neural networks. arXiv preprint arXiv:1312.6026
31. Graves, A. (2013). Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850.
32. Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., & Saenko, K. (2015). Sequence to sequence-video to text. In Proceedings of the IEEE international conference on computer vision (pp. 4534-4542)
33. Barbu, A., Bridge, A., Burchill, Z., Coroian, D., Dickinson, S., Fidler, S., ... & Schmidt, L. (2012). Video in sentences out. arXiv preprint arXiv:1204.2742.
34. Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., & Saenko, K. (2014). Translating videos to natural language using deep recurrent neural networks. arXiv preprint arXiv:1412.4729.

35. Brox, T., Bruhn, A., Papenberg, N., & Weickert, J. (2004, May). High accuracy optical flow estimation based on a theory for warping. In European conference on computer vision (pp. 25-36). Springer, Berlin, Heidelberg.
36. Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., & Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4694-4702).
37. Peris, Á., Bolaños, M., Radeva, P., & Casacuberta, F. (2016, September). Video description using bidirectional recurrent neural networks. In International Conference on Artificial Neural Networks (pp. 3-11). Springer, Cham
38. Gao, L., Guo, Z., Zhang, H., Xu, X., & Shen, H. T. (2017). Video captioning with attention-based LSTM and semantic consistency. IEEE Transactions on Multimedia, 19(9), 2045-2055
39. Xu, J., Mei, T., Yao, T., & Rui, Y. (2016). Msr-vtt: A large video description dataset for bridging video and language. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5288-5296)
40. Torabi, A., Pal, C., Larochelle, H., & Courville, A. (2015). Using descriptive video services to create a large data source for video annotation research. arXiv preprint arXiv:1503.01070
41. Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In Proceedings of the IEEE Conference on computer Vision and Pattern Recognition (pp. 3686-3693)
42. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
43. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778)
44. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826)
45. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
46. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4510-4520)
47. A. Rohrbach, M. Rohrbach,W. Qiu, A. Friedrich, M. Pinkal, and B. Schiele: Coherent multi-sentence video description with variable level of detail. In German Conference on Pattern Recognition (GCPR), September 2014
48. Pini, S., Cornia, M., Bolelli, F., Baraldi, L., & Cucchiara, R. (2018). M-VAD names: a dataset for video captioning with naming. Multimedia Tools and Applications, 1-21
49. F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. C. Niebles: Activitynet: A large-scale video benchmark for human activity understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 961–970, 2015.

50. Krishna, R., Hata, K., Ren, F., Fei-Fei, L., & Carlos Niebles, J. (2017). Dense-captioning events in videos. In Proceedings of the IEEE International Conference on Computer Vision (pp. 706-715)
51. Escorcia, V., Heilbron, F. C., Niebles, J. C., & Ghanem, B. (2016, October). Daps: Deep action proposals for action understanding. In European Conference on Computer Vision (pp. 768-784). Springer, Cham
52. Xu, Z., Yang, Y., Hauptmann, A.G.: A discriminative cnn video representation for event detection. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2015)
53. Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision (pp. 4489-4497)
54. Yu, H., Wang, J., Huang, Z., Yang, Y., & Xu, W. (2016). Video paragraph captioning using hierarchical recurrent neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4584-4593)
55. Marc Tanti, Albert Gatt, Kenneth P. Camilleri: What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator?, ,2017
56. Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 1. IEEE, 886–893
57. Wang, W., Wu, Y., Liu, H., Wang, S., & Cheng, J. (2018, August). Temporal Action Detection by Joint Identification-Verification. In 2018 24th International Conference on Pattern Recognition (ICPR) (pp. 2026-2031). IEEE
58. Hossain, M. D., Sohel, F., Shiratuddin, M. F., & Laga, H. (2019). A Comprehensive Survey of Deep Learning for Image Captioning. ACM Computing Surveys (CSUR), 51(6), 118
59. Abbas, Q., Ibrahim, M. E., & Jaffar, M. A. (2018). Video scene analysis: an overview and challenges on deep learning algorithms. Multimedia Tools and Applications, 1-39
60. Wang, H., Kläser, A., Schmid, C., & Liu, C. L. (2013). Dense trajectories and motion boundary descriptors for action recognition. International journal of computer vision, 103(1), 60-79
61. Blei, David M.; Ng, Andrew Y.; Jordan, Michael I (January 2003). Lafferty, John, ed. "Latent Dirichlet Allocation". Journal of Machine Learning Research. 3 (4–5): pp. 993–1022.
62. Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image segmentation using K-means clustering algorithm and subtractive clustering algorithm. Procedia Computer Science, 54, 764-771
63. Shi, H., Li, P., Wang, B., & Wang, Z. (2018, August). Image captioning based on deep reinforcement learning. In Proceedings of the 10th International Conference on Internet Multimedia Computing and Service (p. 45). ACM
64. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning (pp. 2048-2057)
65. SentiCap: Generating Image Descriptions with Sentiments - Alexander Mathews, Lexing Xie, Xuming He

66. SemStyle: Learning to Generate Stylised Image Captions using Unaligned Text-Alexander Mathews_, Lexing Xie_, Xuming Hez
67. Wu, Z., Yao, T., Fu, Y., & Jiang, Y. G. (2016). Deep learning for video classification and captioning. arXiv preprint arXiv:1609.06782
68. Sepp Hochreiter and J¨urgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997
69. Tanti, M., Gatt, A., & Camilleri, K. P. (2018). Where to put the image in an image caption generator. Natural Language Engineering, 24(3), 467-489
70. Hodosh, M., Young, P., & Hockenmaier, J: Framing image description as a ranking task: Data, models and evaluation metrics. Journal of Artificial Intelligence Research, 47, 853-899 (20130
71. Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3128-3137)
72. L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. 2016b. Temporal segment networks: Towards good practices for deep action recognition. In ECCV pp. 20–36. DOI: 10.1007/978-3-319-46484-8_2. 11, 24
73. L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4507–4515
74. D. L. Chen and W. B. Dolan. : Collecting highly parallel data for paraphrase evaluation, In ACL, 2011
75. Bhatnagar, B. L., Singh, S., Arora, C., Jawahar, C. V., & CVIT, K. (2017, August). Unsupervised Learning of Deep Feature Representation for Clustering Egocentric Actions. In IJCAI (pp. 1447-1453)
76. Xu D, Yan Y, Ricci E, Sebe N : Detecting anomalous events in videos by learning deep representations of appearance and motion. Elsevier J Comput Vis Image Underst 156:117–127. https://doi.org/10.1016/j.cviu.2016.10.010 (2017)
77. Pan, P., Xu, Z., Yang, Y., Wu, F., & Zhuang, Y. (2016). Hierarchical recurrent neural encoder for video representation with application to captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1029-1038)
78. Shi, X., Cai, J., Gu, J., & Joty, S. (2018). Video Captioning with Boundary-aware Hierarchical Language Decoding and Joint Video Prediction. arXiv preprint arXiv:1807.03658
79. Rennie, S. J., Marcheret, E., Mroueh, Y., Ross, J., & Goel, V. (2017). Self-critical sequence training for image captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 7008-7024)
80. Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu "BLEU: a Method for Automatic Evaluation of Machine Translation", (ACL), Philadelphia, July 2002 , 311-318
81. S. Banerjee and A. Lavie: Meteor: An automatic metric for MT evaluation with improved correlation with human judgments, in Proc. ACL Workshop Intrinsic Extrinsic Eval. Measures Mach. Transl. Summarization, 2005, pp. 65–72.
82. C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in Proc. ACL Workshop Text Summarization Branches Out, 2004, pp. 26–26.

83. R. Vedantam, C. L. Zitnick, and D. Parikh, “Cider: Consensus based image description evaluation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 4566–4575
84. Anderson, P., Fernando, B., Johnson, M., & Gould, S. : Spice: Semantic propositional image caption evaluation. In European Conference on Computer Vision (pp. 382-398). Springer, Cham (2016, October).
85. Hodosh, M., Young, P., & Hockenmaier, J. :Framing image description as a ranking task: Data, models and evaluation metrics. Journal of Artificial Intelligence Research, 47, 853-899 (2013)
86. Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440)
87. Hui, T. W., Tang, X., & Change Loy, C. (2018). Liteflownet: A lightweight convolutional neural network for optical flow estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8981-8989).