Oxford Brookes University

Video Classification using Recurrent Neural Networks

by

Misbah Munir

A thesis submitted in partial fulfillment for
MSc. Computer Vision

in the
Department of Computing and Communication Technologies

September 2017

Oxford Brookes University

Abstract

Department of Computing and Communication Technologies

MSc. Computer Vision

by Misbah Munir

Video Classification is an important and widely researched problem in the field of Computer Vision. Machine Learning models have proved to perform significantly well in order to generate good results. Existing methods focus on traditional Convolutional Neural Network (CNN) models to analyse visual data. Since the dimensionality of this data is very high making it complex in nature. Therefore the learning algorithms are restricted in the way they process the information, there is a high risk of losing the useful information. Usually the video clips are of variable length but they are required to be mapped onto a smaller and fixed dimensional vector representations of features in order to be processed by CNNs. This also creates an overhead of information preprocessing before the results can be achieved. In recent years, Recurrent Neural Networks (RNNs) have proven their strength to for analysing sequence-based data without losing much information. In this research, we aim to investigate the strengths of RNN models with a focus on Long-term Recurrent Neural Network (LRCN) model to classify the videos. Since visual data is sequential in nature and high complex, therefore, theoretically RNNs can be a great fit for its analysis. In these experiments we aim to analyse the results we achieve on video classification problem using LRCN.

Acknowledgements

There are many people I would like to acknowledge for providing me technical help and and otherwise support - Dr. Fabio Cuzzolin, Suman Saha, Raluca Vagner, Dr. Ruomei Yan, Dr. Faye Mitchell, Fareena Saleh and Sam Varney...

And above all, my family...

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The aim of Computer Vision is to induce human-like visual observation and decision making capabilities into the machines, if not better[1]. Some evidence shows that the field emerged as a summer project where a computer was expected to *explain what it saw* through the attached camera[2]. This approach only proved that the problem of vision is very complicated and an extensive research needs to be performed in order to achieve the set goal.

Since the inception, Computer Vision problem has been divided into smaller problems in an iterative manner, for example, image enhancement, object detection, noise removal, segmentation, video processing, and many more in order to understand the semantic information for the data. This makes Computer Vision an interdisciplinary field having close ties to machine learning, pattern recognition, image procession and computer graphics. One of the most important problems faced in the field of Computer Vision is that of Video Classification. In other terms, the process of Video Classification automatically distinguishes between different types of videos based on their semantic data. The classification problem can be addressed in many ways based on its potential applications in different industries. First example is the medical industry, which can benefit from these methods in order to detect and/or verify a person having a particular disease and its severity based on physical appearance and symptoms of the patient. Secondly, suspicious behaviour can be detected through the video stream captured via surveillance cameras. We can also classify and block the online visual content based on appropriate audiences, for example, horror scenes censored for the audience under the age of 12 years. It has been observed that the classification applications can assign a finite number of class labels to a set of videos but we can train the algorithm to detect an anomaly if a situation arises.

[1] http://www.bmva.org/visionoverview
[2] https://en.wikipedia.org/wiki/Computer_vision

Extensive research has been performed to classify the videos based on their semantic data[3–5] which can potentially lead to the development of applications of video classification in various fields including health care, security and surveillance, legal services, education and entertainment. Machine Learning algorithms have proven to be a good fit to solve the problem of Computer Vision in general and video classification in particular[4]. The research trends have shifted more towards deep learning methods for understanding the visual data. Neural Networks have provided a nonlinear method of mathematically modelling the data. The layered architecture in this set of learning algorithms is the key for generating good results. The increase in the number of layers, the concept known as depth, has provided fruitful results but they can be further optimized. Depth, as explained by Bengio[3], is a concept that was only introduced by analysing the flow of information in human brain. Intuitively, adding depth can improve the quality of results however, there is high risk of computational inefficiency at a higher level. 3D Convolutional Neural Networks (CNNs) have so far been the most successful model for video classification[4] despite the computational complexity and the limitations to the architecture. Therefore, these results need further optimisation due to computational costs due to various unresolved issues including dimensionality reduction, data manipulation and sequence based learning optimisation. More details about the current research trends is provided in Chapter 2 of this report.

With the availability of better processing machines like GPUs, the deep learning methods have provided promising direction in achieving even better results. However, with easy access to high resolution cameras and more interest in video capturing, there has been an exponential increase in visual data collection and sharing. This has, in turn, had a positive impact on the scientific research in two main ways. Firstly, the quality of data available has significantly increased providing more detailed information as compared to the data captured through lower resolution cameras. Consequently, the dimensionality of the data has increased exponentially. More data means more training examples for the learning algorithms, which can have positive impact on achieving generalised solutions. Secondly, the available data is more diverse which means there is a better chance of abstracting the concepts in a better way and understanding underlying semantics. The quality of data may vary for different devices, which is why we need more flexible and robust solutions to the problem of learning.

Recurrent Neural Networks [6] (RNNs) are relatively new idea for computer vision research but very efficient nevertheless. The RNNs are conceptualised based on the information flow in human brain where memory and recurrence are important in order to develop a better understanding of available information. Scientists have achieved great

results by using RNNs for applications like speech recognition [7] and text processing [8]. However, in case of computer vision, the RNNs are well known for their ability to generate a word sequence from images or videos. [9].

Despite being the algorithm of choice for many sequential data processing applications for speech and text processing, to the best of our knowledge, only a few have investigated the strength of RNNs for video classification problem. Visual data is also sequential in nature with much higher dimension as compared to other form of digital data. This makes the problems associated with visual data processing more complex in nature therefore needing better computational resources and more optimised data models. Since, RNNs have overall provided better results on sequential data modelling; it is intuitive to use them for visual data analysis too. In this research, the possibility of RNNs as potential visual data classification solution has been investigated using a particular style of RNN known as Long Short-Term Memory (LSTM) in combination with Convolutional Neural Networks (CNNs). Details about the approach used have been provided in Chapter 3 of this report followed by the Chapter 4 containing experimental details and achieved results.

In a nutshell, this report discusses the baseline aspects about the possibility of RNNs as a possible solution to the Video classification problem. The overview for Video Classification research has been provided, applications and effectiveness of RNNs and a review of the work carried out to classify videos through the selected model of RNN. Then, a discussion about the system architecture followed by the overview of experiments performed and discussion about the results achieved. This reports also provides a critical reflection about the research project management, success and failures have been discussed along with potential issues associated with the research for potential social impact and at professional level in Chapter 5. Some ideas about the further steps which can be taken in order to improve the results have also been discussed.

Chapter 2

Literature Review

This chapter provides an overview of the latest trends in the research of Video Classification in order to better understand the concept and goals of the research. Secondly, discussion about the deep learning methods for solving the stated problem has been provided. Finally, an overview of the Recurrent Neural Networks research and its applications in the Computer Vision Research have been discussed.

2.1 Video Classification

Video Classification is an important area of Computer Vision research. It is a special method of understanding the semantic content of videos with a specific goal of automatically categorising the videos in different classes[10]. There are many application classes for the classification problems which attempt to label the data based on objects, events, actions or simply genre of a particular video[11]. Historically, the approach of video classification was done by extracting features, processing the features into fixed sized vector representation of a video and finally label prediction via a classifier[4, 12, 13]. This approach was very difficult to follow since it require more in-depth understanding of feature analysis. Since human understanding of vision is limited[14] therefore, it is difficult to analyse the set of known features that might lead to better understanding of the data. Secondly, the limited power of computational resources was a significant factor limiting the success rate.

2.2 Deep Methods for Video Classification

In 2014, the trend for using deep methods for machine learning started getting attraction by the research community after first proposed by Karpathy[4]. It was proposed to replace the deep CNNs to perform all three tasks described in previous section after taking inspiration by the advancements in image-based classification, automatic segmentation and labelling[15–17]. There have been many approaches used to efficiently use the CNN models by exploiting the feature representation by deep networks as well as in the form of hand-crafted features.

2.3 Recurrent Neural Networks

Recurrent Neural Networks are special type of deep learning networks which provides promising direction in the research of sequence based learning. These models have been biologically inspired by human brain[18]. In technical terms, humans process information using temporal information as a major factor for developing understanding. Many machine learning methods, including most deep networks lack this ability which generate a semantic gap for proper data processing.

The introduction of recurrence in the research of sequential data processing have generated some *magical*[1] results specially in the field of signal processing for speech[7] and text[8]. Speech signals can be represented as a waveform in time domain[2], hence can be visualised as two dimensional data sequence. In contrast, video is also a sequence of data (images) with much higher dimension[2] as described in Chapter 4. There are many ways of generating feature hierarchies based on the

In case of modelling visual data using RNNs, image captioning have achieved some great results. It models a single input vector to a sequence of outputs generated to describe an image[18–20]. The main strength of RNN that needs to be exploited is sequence modelling which can be achieved by performing sequence learning on visual data. The results achieved by modelling video sequence to generate captions for videos as a sequence of outputs has also shown some promising results in the direcion of adopting RNNs[21–23]. So far, sequence generation seems like a major part of learning through RNN. RNNs can also map a sequence of input to a static output.

[1] http://karpathy.github.io/2015/05/21/rnn-effectiveness/
[2] https://en.wikipedia.org/wiki/Audio_signal

2.3.1 Long Short-Term Memory

Originally proposed in 1997[24], to overcome the long-term dependency problem in RNNs which was a major factor negatively affecting the adaptability of RNNs.

More recently there has been an increased interest in utilising the strength of LSTM-RNNs for sequence to static mapping has generated some interesting results. Long Recurrent Convolution Network (LRCN) is a simple end-to-end model that classify a video based on deep features in order to classify visual data. The evaluations based on the UCF-101 dataset have shown some great results[2]. In another research, fused Spatial and Motion features multi stream feature feed to LSTM network for categorise the videos in UCF-101 dataset have achieved similarly positive results[25]. Similarly, increasing the depth with added feature pooling achieved some interesting insights but did not prove that the stacked LSTM can have any significant performance gain for the system[26].

As per the literature review, there are some promising results which led to the belief that recurrence is a positive step in the direction of efficient classification. However, most of the data is restricted in evaluation techniques. Most of the researches have been performed on UCF-101 dataset which shows the viability but restricts the versatility of generalisation of the hypothesis. Therefore, our research is focused on generalisation of the problem by expanding the research to a smaller dataset with more diverse problem set. This would give a better idea as if the data convergence is as optimised as we expect it to be or do we need to find some alternate methods of optimisation.

Chapter 3

Methodology

In this section, a detailed discussion about the our methodology have been provided. Initially, the system architecture has been explained followed by details about each module in the architecture. Our approach analyse the problem using a only a fixed number of videos classes as stated in the dataset under consideration.

3.1 Long Recurrent Convolution Network

A Long-term Recurrent Convolutional Network[2] (LRCN) joins a deep visual feature extractor (CNN) with a model that can figure out how to model the temporal progression for a task which involves sequence based information.

At an abstract level, the system architecture is modelled based on the concept proposed by Jeff Donahue[2]. The system was developed keeping end-to-end trainable model as a concept which gets a sequence of input, process it by extracting features from key frames and finally generating an average of output sequence generated by the LSTM module. Figure 3.1 provides the visual representation of our system architecture.

3.1.1 Supervised Learning Model

This model is a supervised learning method where the system can be tested on unseen data after a significant number of examples have been provided to the system. The model generalise after it sees the data a certain number of time and have achieved a significantly low error rate on training data. The system can also be trained using unsupervised learning method but that would conflict with the set goals of the research.

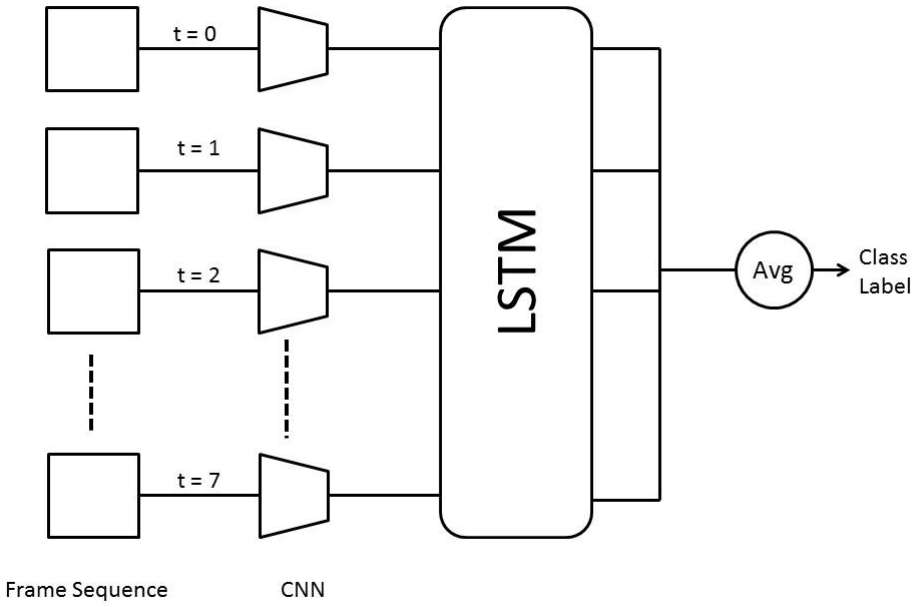

FIGURE 3.1: Long Recurrent Convolution Network - System Architecture

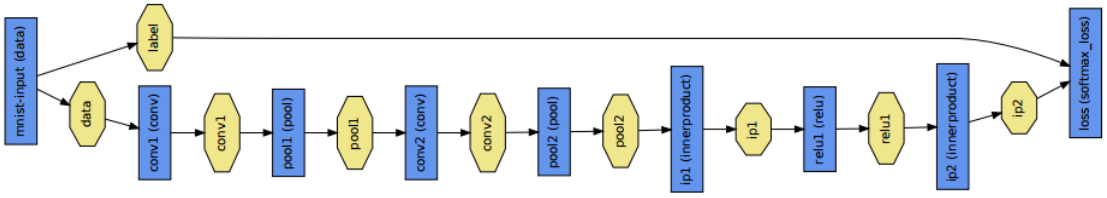

FIGURE 3.2: CaffeNet Model with Data Blobs and Convolution layers shown in yellow and blue boxes respectively[1]

3.2 Deep Feature Extractor

First module in the system extracts the local features from each image frame provided to the system. The frame sequence is sparsely selected in the time domain with no other information retained whatsoever. CaffeNet[1] which is a variation of AlexNet[15] has been used in order to extract features from the frame. CaffeNet was originally designed to replicate AlexNet for Caffe libaries, however a small error where two layers were interchanged resulting in a new model. CaffeNet architecture is shown in the Figure 3.2. There is no difference in model efficiency for both feature extractors.

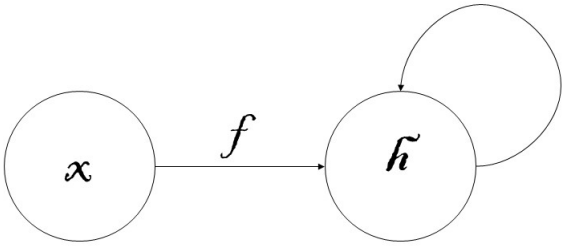

FIGURE 3.3: A Basic RNN Model

3.3 Recurrent Neural Network

RNNs is the main model learning module in this system. It is capable of learning long-term dependencies in the sequence. Since the models are biologically inspired, therefore they provide a close correlation with the information flow pattern in human brain. The concept of recurrence utilise the temporal information provided by the sequence. Figure 3.3 shows a basic RNN where the first input vector from a sequence is responsible for a prediction in one time step. This information is then used as an input at the next time step (usually) concatenated with the next input vector in a sequence[27]. This phenomenon is termed as state of the system. The system maintains its state at each time interval based on the incoming information. It can be changed if a new evidence is provided to the system.

If we unfold this model, we can visualise how the RNN models process information in time domain. Figure 3.4 shows how the model generate sequence with a step in time.

Correction: there is a random input value which is provided at the t0 for a RNN. In the Figure 3.4, it can not be seen.

3.3.1 Long Short-Term Memory

There are many forms of RNN models, however, LRCN model takes advantage of LSTM learning model. LSTM is a version of RNN where *h(x)* gets replaced by an LSTM unit in the system. Figure 3.5 shows a basic LSTM unit. The forget gate in this unit adds greatly to the efficiency of the system. Its main purpose is decision making - what information need to be stored and which information can be discarded. This is the key element to solve the long term dependency problem in the system. This reduces the complication in model generalisation and adds to the efficiency of the system.

Mathematically the model parameters are discussed in figure 3.6

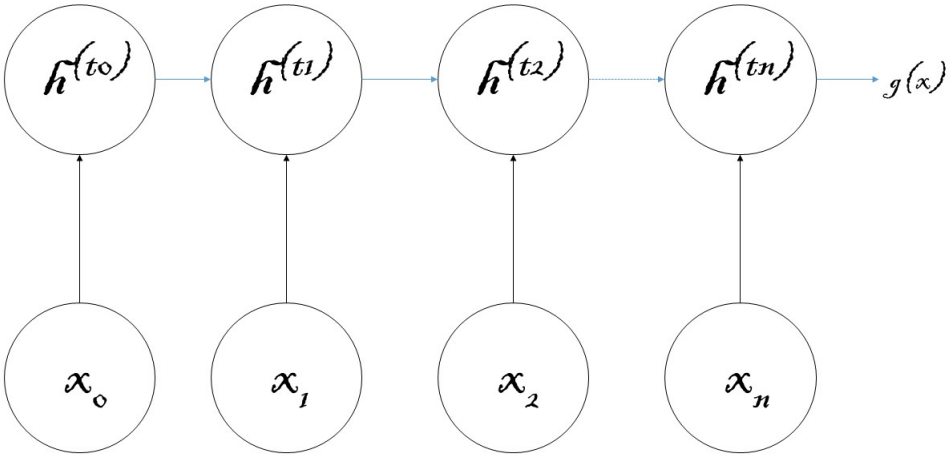

FIGURE 3.4: RNNs Unfolded

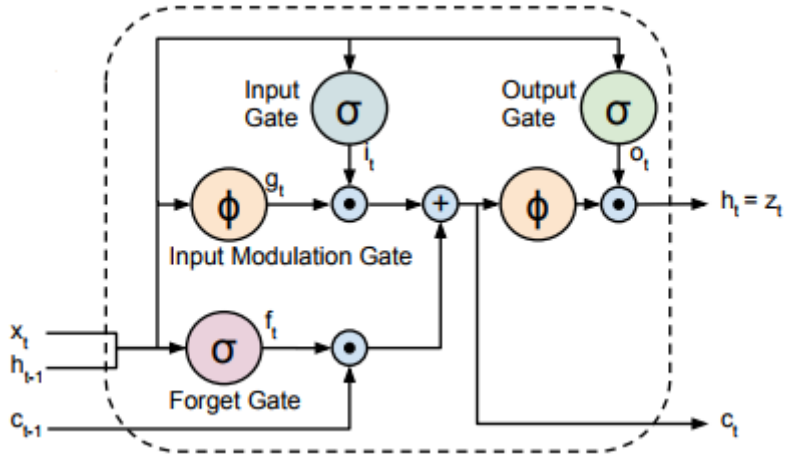

FIGURE 3.5: An LSTM Unit[2]

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
g_t &= \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

FIGURE 3.6: The Mathematical Model of LSTM[2]

3.4 Model Optimisation

There are two options available for optimisation of the learning model - Stochastic Gradient Descent[28] (SGD) and Adaptive moment estimation(Adam)[29]. SGD is the first choice for many researchers since it is a standard method achieving state-of-the-art results. However, Adam optimiser provides approximately same results with more efficiency. It improves the adaptive learning rates of each model parameter. The momentum - mean and variance of the gradients are used to optimise the system as a set of moments. The LRCN model provides flexibility to choose between both optimisation options, however, we have chosen to utilise the Adam method due to computation time constraints.

There are three main differences between the two methods is the moments variables used by Adam. It uses the two moments - mean and variance of the gradient to optimize the learning instead of using just the gradient descent. In case of programming, the Adam Optimiser was taken as off the shelf module and was plugged in the system as a black box.

3.5 Evaluation Method

The system is set to predict a class label as a static integer. In any supervised learning model, the class labels are associated with the test data which are hidden from the model in order to make the prediction. The predicted class label is than compared with the actual class label using the XOR operation in order to calculate the number of correctly classified labels. These class labels are than compared to the total number of classes in order to calculate the percentage of correct classifications. Our model uses the same method to evaluate the model.

The next chapter provides the evaluation details abbot the LRCN method explained in this chapter.

Chapter 4

Experiments and Results

This chapter provides the details about the experiments carried out in order to generate the results. First, the prerequisites have been discussed in order to the dataset used for experimentation and other variables involved in the data followed by the generated results with a discussion.

4.1 Dataset

The dataset used in the experimentation is HMDB 51[30]. It is a collection of short video clips about different actions being performed by the human actors in the video clips. There are 51 action classes for approximately 7,000 video clips which varies in length. Each action class contains at least a 100 examples. The video clips used in the dataset come from different sources including clips from movies with better resolution and static background as well as captured from a low resolution camera with a possibility of shaking camera or moving background. This adds to the complexity of information but make the information more general. There are two main reasons to use this dataset. Firstly, the size of dataset is adequately small and secondly, the dataset is commonly used by the research community for the action classification task. We also performed some experiments on a subset of UCF-101[31] dataset where only 4 classes were considered in order to observe the results to better understand the performance of the algorithm.

4.1.1 Dimentionality Reduction

A video is composed of a sequence of image - frames. A sequence of 30 frames makes up to form a clip for a 1 second video. Each frame is a set of integers with can be represented as a 3D matrix. Hence, the amount of data encapsulating the information about a single

video clip is very high. In the datasets, each video sequence is at least a few seconds long encapsulating the complete action. Processing that large data is computationally very expensive. Therefore, there is a need to sub-sample the data without losing important information.

In order to achieve this, the data has been heavily down sampled and some key frames were chosen based on length of the video and the provided batch size in order to make efficient use of the available resources. The frames were sampled at a regular interval which was calculated based on the length of the video. Therefore, it is speculated that some videos provide more accurate information about the action than others regardless of down sampling.

4.2 System Configurations

The experiments were performed in the systems available to Artificial Intelligence and Vision Group in the Cognitive Robotics Lab at the CCT department of Oxford Brookes University. There was some flexibility in management of resources however, the decisions about choosing a particular technology or platform depended upon multiple factors including cost, accessibility, flexibility and available support. A visual representations is provided through figure 4.1.

4.2.1 Software Configuration

The experiments were performed on a Linux-based system, using the latest version of Ubuntu. The development of the software has been done using the software platforms including MatLab, Lua, Torch and Python. In order to access the remote systems, the Big-IP Edge Client was used to create a VPN and PuTTY as a Telnet client.

4.2.2 Hardware Configuration

The systems include a set of GPUs on Linux based systems with variable memory options. Table 4.1 provides the details about the system configurations and the set of technologies (hardware and software) that were used to perform experimentation.

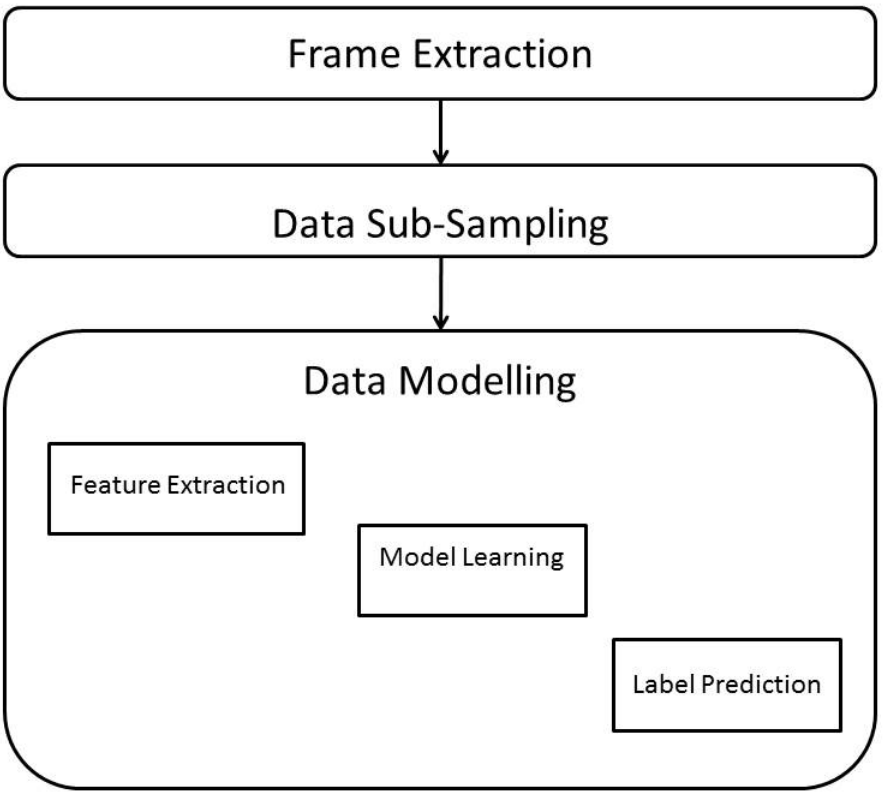

FIGURE 4.1: Our system in detail

4.3 Results

The system processed information by taking 8 frames per video and feed them sequentially to the LSTM generating an average of the generated number sequence. The frames were chosen in time domain by dividing the sequence into 8 approximately equal sets and choosing the last frame from each set. This shows that the system was heavily down-sampled for most videos. Two main factors caused this significant down-sampling: (1) to improve the efficiency of the system and (2) the selected set of frames were enough for a human to recognise the specified action action. There were three different people who were shown a set of actions to identify where they could identify the action correctly 95% of the time, given that they knew the set of choices beforehand.

4.3.1 Model Parameters

As mentioned above, the model was trained one video at a time with batch size being 8. The learning rate was set to 10e-6. The LSTM with the width of 256 was used.

Number of Classes	Maximum Accuracy
2	80%
4	71.9%
13	67.8%
51	8.9%

TABLE 4.1: Accuracy of Classification on HMDB-51 Dataset

4.3.2 Evaluation

In order to quantitatively analyse the results, the statistics based error rate generator was used. The number of test examples were provided to the system where the predicted class was matched with the actual class label and the error rate was generated based on the percentage of correctly classified examples.

4.3.3 Experimental Results

It was observed that the system performs significantly well on smaller number of classes. The highest classification accuracy was achieved on 2-class classification where the system achieved a maximum of 92% accuracy. Adding more classes significantly reduced the performance of the system. It was also observed that the accuracy is inversely proportional to the similarity of actions. But, in some cases the similarity of action was not a prime factor in dropping the accuracy. For example, brush hair and cart wheel are completely different actions as per human understanding, but for the system, they were fairly similar which led to only 67.2% accuracy. If we have a look at a completely different pair of classes, that is, chew and catch, the system correctly identified the right class exactly 80% of the time. In case of recognition among 13 different action classes which includes chew, catch, brush hair, cart wheel and more, the classification accuracy was 67.8%. This led to the belief that there is a possibility of the system to need more information in order to make better decisions and improve its learning ability. Lastly the system was tested to identify the labels for examples from 51 action classes. The performance of the system dropped significantly and achieved only 8.09% accuracy.

4.3.4 Subset of UCF-101

There was a significant improvement in the performance when the training was applied on the small subset chosen from UCF-101 dataset for action classification. There were some key differences in the data sampling for UCF and HMDB

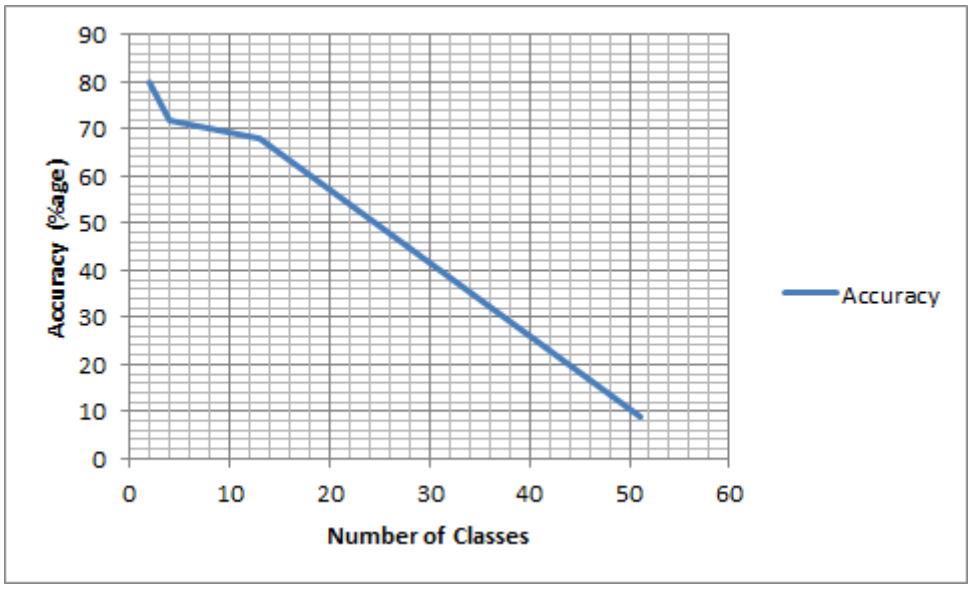

FIGURE 4.2: Comparative Classification Accuracy for HMDB dataset

4.3.5 Limitations

As discussed the varied results have been achieved based on the available data, however, there are a few limitations associated with the research as follows:

1. The time constraint is a major factor which limited the scope of experiments in a significant manner. The experiments were done with the set of parameters that would improve the computation time.

2. The resources for experimentation were limited and there was limited to no external support (by technical team at the respective company) available if external resources were utilised such as Amazon Cloud, etc.

3. Technically, the experimentation was limited to a certain set of programming environments that were completely new in perspective of learning new technologies - Torch, Lua, etc.

4.4 Further Steps

The main objective of the system was achieved by creating an algorithm in order to analyse the ability of RNNs to correctly classify the provided videos. So far, the achievement is a basic understanding about the algorithm and how it works. There is still a great need to optimise the results through various means. Here are some propositions about the future work:

1. Feature extraction method can be improved as a step forward in order to improve the results. At frame level, there can be multiple ways of feature processing including temporal features provided with frame level deep features. This approach will significantly increase the training time and hence negatively affect the computation efficiency of data. However, that can lead to potentially good results.

2. The strength of RNNs is mainly achieved by their ability to effectively process the sequential information. In this research, the input sequence was set to a fixed size. However, the videos for different classes may vary in length due to the nature of a particular class. For example, it takes more time to identify the activity of brushing hair than laughing. So the number of frames required to process the information in both cases can be different. There is a need to analyse the mapping results based on variable length input sequence which is a clearly defined limitation in this system.

3. Video Captioning is an important application of sequential data processing using RNNs which provide a sequence-to-sequence representation of visual data. A 2-layered RNN network model is used to represent the captioning data efficiently known as Encoder-Decoder Network. In most applications, increase in depth have shown improvements in information representation which can be exploited in order to improve the results of video classification. Hence, the impact of having an RNNs as a hidden layer of the model as a potential solution can be investigated.

4.5 Conclusion

The statistics clearly show that the system is capable of achieving some good results provided the data is distinctive and visually separable. There are some important details that needs to be considered in order to improve the system. The temporal information needs to be preserved and specially the sub-sampling of data must be done with caution. There have been some insights into the feasibility of the idea in terms of computational efficiency however, there is still a need for more detailed research before completely discarding RNNs as a potential model to improve classification efficiency - computational or otherwise. In order to re-generate the results, the code and supporting experimental data has been provided in the form of a DVD along with this document. Appendix A describes the data contents in detail.

Chapter 5

Critical Reflection

This chapter provides an overview of the work done in order to undertake this research. The work has been reviewed based on different aspects including technical aspects, future approach, challenges faced and the impact on personal development.

5.1 Project Development

In order for the project to be a success, there is a need of clearly identified goals and objectives. In case of this short-term project, the goals of the project were clearly identified. Initial literature review was done in order to have a detailed analysis about the project feasibility which led to actual development. Since, the research provides evidence on the possibility of using recurrence as a potential factor in optimising and improving the video classification task, a basic model was devised and implemented in order to get the results.

As the goal of this research was clearly identified at the beginning and the hypothesis was clearly stated, the classic waterfall model was followed for undertaking the project. Though the project ran smoothly for its course however, there were some changes done to the original plan of the project, where some tasks took longer than planned and some milestones were achieved in shorter time than anticipated.

5.1.1 Software Development Model

The model for Software Development followed to divide and achieve the goals and objectives is the classic waterfall model[32] which divides the activities of project development into a list of activities only take place in a sequential manner. This method of project

development provides a very clear and logical order of activities in order to achieve the final goal. This approach works greatly if the time allocated for a task is limited. However, in a longer run, the agile approach where the same set of activities can be performed in an iterative manner are thought to be a better strategy.

5.1.2 Risk Management

There were multiple risks associated with the project which had a potential to negatively impact the progress. Those risks were clearly identified based on the available information and previous experiences, the mitigating strategies were planned and implemented in order to minimise the effect.

Murphy's law states, *anything that can go wrong, will go wrong.* Although the risk mitigation strategies were carefully followed, some risks were unavoidable and hence the course of project execution was impacted. The updated risk matrix has been provided in the table below for more details. There are two important lessons learned through this exercise:

1. There is always a chance of an unknown risk which cant be avoided in any way, therefore it is important to plan for success and add some buffer time at the end of each activity.
2. Time management through discipline is the most important attribute in success of the project.

The initially identified risks were quantitatively analysed for two factors, that is, likelihood and impact. In both cases, the the lowest value being 0 and the highest value being 5. The risk was identified and categorised in two classes - technical and non-technical. Following sections provide more information about the risks that were identified, their updated impact and their occurrence during the project course.

5.1.2.1 Technical Risks

This category of risk involves the potential problems due to technical problems for project development or management problems that can in turn directly influence the research. With proper planning and management most of these risks can be avoided. Table 5.1 provides an overview of such risks.

Risk	Severity	Probability	Occurrence and Comments
Data Loss	5	1	Yes, only the data at local machines was lost due to unknown reasons
Insufficient Computational Resources	3	3	No, resources were in high demand therefore scheduling was an added factor
Code/Writing Block	5	2	No, having a block in order to perform at the best of abilities is humanly impossible, buffers in the development helped minimize the problems
Getting Late Results (Computation Inefficiency)	3	1	No, since the code was very optimized and the dataset used was not of very large scale
Insufficient (or too much) Documentation	3	1	No, the documentation was started ahead of the deadline

Table 5.1: Updated Risk Matrix for Technical Risks

Risk	Severity	Probability	Occurrence and Comments
Lack of Communication	5	2	Yes, as most of the group members were away for different projects
Unproductiveness (due to unavoidable circumstances)	4	1	Slightly, as there were few unavoidable circumstances beyond control
Health Issues	5	1	No

Table 5.2: Updated Risk Matrix for Non-technical Risks

5.1.2.2 Non-technical Risks

This category of risks is associated with the external factors that have a possible effect on the research. These risks are of high severity with uncertainty of the likelihood. There is no way of working around these risks but the impact can be minimised by better management. See Table 5.2 for an overview of the risk matrix.

5.2 Legal, Social, Ethical and Environmental Issues

BCS, The Chartered Institute for IT[1] provides comprehensive guidelines about the code of conduct in order to provide professional services for the information technology professionals. These serve as a set of guiding principles for the professionals in order to perform their duties to bring the technology for the good of society. These guidelines cover all the aspects of professional behaviour to avoid legal, social, ethical and environmental issues for IT professionals. Same guiding principles were followed as a reference to carry out this research. There are many issues that can be associated with the potential applications of this research, which have been discussed in this section.

5.2.1 Legal Issues

There are two main legal aspects which are crucial in this project. These issues are which are related to the data that is used in the conducting the research. First, the code that has been used in the project needs to be appropriately marked, that is, the work done by any individual needs to be properly accredited. Most of the code used to perform this research was available as open source under MIT[2] licence which provides the permission to obtain a copy of the software and associated documentation files without restriction, with a requirement to add the copyright statement for the licence. The condition has been met and the code has been submitted with included copyright statement in a separate text file as well as within the code.

Second main legal concern is the permissions associated with the dataset that has been used to train and analyze the efficiency of the algorithm. Since, only free publicly available dataset was used in the experiments, therefore, it was verified beforehand if it has any restriction associated with its use for the research study.

5.2.2 Social Issues

Socially, the research can impact the community or individuals in different ways depending upon the application it is used for. The main aim of the research is to provide a general opinion about a video based on its content. Although in this research, only analysis of a particular methodology has been provided, therefore there is no added social issue in that aspect is expected. However, video classification applications have a potential of facing social issues if the video is wrongly classified which ultimately can lead to the potential problem for using algorithm for social good. Since, using RNNs

[1] http://www.bcs.org/upload/pdf/conduct.pdf
[2] https://opensource.org/licenses/MIT

are a novice idea for performing video classification, and the results are not perfect at the moment, therefore, the risk of facing this issue is very high. One way of working around this problem is to consider the audio input, if available, to potentially improve the results.

5.2.3 Ethical Issues

Third important factor to consider is the set of ethical issues that might be involved in carrying out the research includes abidance by the law, present another scientists work as ours - plagiarism. Both of these issues were handled by clearly identifying the legal requirements of the data and providing proper references within the code and the documentation. The information needs to be clearly communicated with honesty. It was taken care of in a very careful manner. However, there is always a risk of communication due to potential misinterpretation or words.

5.2.4 Environmental Issues

Since, it was inevitable to print the report and helping materials for the research, there was significant amount of paper used in this process, which is the major environmental issue caused by this research. Secondly, carbon emissions caused by the use of computational resources and internet research are also important environmental factors associated with this research, however, there is no known way to avoid this problem at the moment.

5.3 Personal Development

The research was undertaken as a development module for the Masters degree in Computer Vision. The objective of the exercise was to gain in-depth knowledge about the research area, improve research and development skills including soft skills and technical skills, get hands on experience of software development and management. There were certain skills that were learned during this course. The table 4.3 provides a brief overview about the skills acquired and the level of improvement. Although there is a numeric value (0 - 5) associated with each skill, it has been assigned based on personal reflection.

In case of developing research skills some tools were identified and some methods were learned with the help of fellow students in the same research group.

Category	Skill	Improvement Factor	Comments
Technical	Coding	4	Languages learned: Lua (with Torch repository), Improved skills: Python, Matlab
Technical	Operating Systems	5	Gained experience using the Linux based systems, remote access via command line and Virtual Private Networks
Technical	Version Control	3	Cloud based version control system GitHub was used in order to keep backup for the code
Technical	Research Skills	3	Build up on the research skills acquired through taught modules, however, still find some room of improvement due to knowledge gap
Soft	Time Management	3	Management through task prioritisation was learned in due course after facing some failures as mentioned in the risk matrix
Soft	Confidence	5	The lack of confidence in technical skills due to lack of knowledge about the platforms was overcome

TABLE 5.3: Brief Overview of Personal Development

5.4 Final Remarks

This research was a great experience overall. There have been some set backs which can be overcome by taking better measures for risk identification, assessment and mitigation strategies. The knowledge and experience gained through taught modules was a very important factor in understanding the technical details. The practical exercises provided a baseline for performing initial analysis, specially based on Matlab, which helped greatly in system modules of video processing and data sampling.

Appendix A

Research Data

This requirements of the dissertation was met be providing the right set of deliverable to the department in partial fulfilment of the MSc degree. All of these files are available in digital format with the dissertation document in the form of a DVD. Following items are included:

A.1 Registration Document

This document contains the main aim of the research along with set deadlines. It also contains the main goals of the research, the supervisor of the research and some initial deliverable were identified.

A.2 Project Plan Document

Second deliverable of the project was the plan document for the project. This document explains in detail about the goals and objectives of the research, the proposed final outcomes. It also covers the feasibility report about the plausibility of the concept and finally a brief literature review of the topic in general with special emphasis on the algorithm under consideration.

A.3 Progress Presentation

The PDF version of the progress presentation slides have also been provided. The presentation was a reflection on the goals and realistic reflection on the achievable aspects

of the goals. It also reflected some changes which were made on the course due to some uncovered new evidence.

A.4 Code Repository

The folder named 'code' contains the programming data for the research. These files are placed in different folders which may or may not overlap each other. This data can be further divided into three categories:

A.4.1 LUA Scripts

The script files with the extension LUA are the main script for model training, validation and testing. Some files are utility files which are responsible for data loading and processing before feeding it to the training model.

A.4.2 Data pre-processing Scripts

These scripts were used to extract frames from the videos, renaming the video files in order to avoid any unwanted errors, data sampling and cleaning.

A.4.3 Dataset References

The text files contain references to the folders containing frames extracted and sampled from each video, the video label and pointing to the set of frame sequence.

A.4.4 Preprocessed Data Files

These are the frames extracted from the videos, sub-sampled and stored separately to be referred from the text files explained previously.

A.4.5 Research Papers

A folder including all the research papers which have been referred in this report are provided. The referred books have not been added.

Appendix B

Code Setup

The code is provided with this document in the form of a DVD. Here is a step-by-step guide to run the code on a Linux based system:

B.1 Pre-requisite Tools - Installation Instructions

1. **Install Lua 5.2**: First step in setting up the system is to install Lua 5.2. It is important to make sure that the correct version is installed, otherwise the model can't be trained due to memory management problem in Lua 5.1.

2. **Install Torch Libraries**: The Torch Libraries needs to be installed with the correct version of Lua. The helpful resource for installation instructions is available at `http://torch.ch/docs/getting-started.html`

3. **Other Libraries**: There are few Lua modules that are required to execute the script. These can be installed using Luarocks Package Manager. The packages which are required includes image, cutorch, cunn, cudnn. Apart from these, the ffmpeg module for the respective operating system must be installed.

4. **Matlab**: This tool is necessary for data preparation. The scripts have been provided in *Data Organisation* folder.

5. **Python 2.7**: A small script that extracts frames from video sequences has been written in Python. This script is also available in the *Data Organisation* folder.

B.2 Data Preparation - HMDB51

First you need to download the HMDB51[30] dataset from the respective repository. If your dataset is composed of of video files, make sure that the naming convention of each category and video file is correct. There must not be any character other than alphabet or digits (a-z, 0-9) or any space in the file names. Than run the data organisation methods in a sequential manner:

1. Run the *RenamingFiles.m* script to manage the file names of the dataset.
2. Run the *extractFrames.py* script to convert the video files in to sequence of image files.
3. Execute the *OrganizeData.m* script to clean up the data leaving only 8 frames per video.
4. Finally, use the *somefile.m* script to create text files for training, validation and test data that would provide the model a reference to the data location.

B.3 Model Training and Testing

1. *train.lua* is the first script that loads the data and initialise the model as per the parameters set in the options set. The options that must be changed includes number of classes, references to the files containing paths of train, validation and test set, model location, location to the folder containing the frame sequence. Some other options let you choose the parameters that can be changed in order to train the model including learning rate, number of epochs, optimisation method, etc.
2. You need to make sure that your system is training and saving the model properly. The last epoch will generate and save the model at your specified location as *cp final.t7*.
3. Test the trained model using the *test.lua* which will generate the model accuracy for classification results under the name of action recognition.

Bibliography

[1] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093, 2014. URL `http://arxiv.org/abs/1408.5093`.

[2] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[3] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[4] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[5] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 568–576. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5353-two-stream-convolutional-networks-for-action-recognition-in-videos.pdf`.

[6] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, page 3, 2010.

[7] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

[8] Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.

[9] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4534–4542, 2015.

[10] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. *CoRR*, abs/1503.08909, 2015. URL `http://arxiv.org/abs/1503.08909`.

[11] M. Roach, J. Mason, N. Evens, L. Xu, Nicholas Evans, Li qun Xu, F. Stentiford, and Martlesham Heath. Recent trends in video analysis: A taxonomy of video classification problems, 2002.

[12] Heng Wang, Muhammad Muneeb Ullah, Alexander Klaser, Ivan Laptev, and Cordelia Schmid. Evaluation of local spatio-temporal features for action recognition. In *Proceedings of the British Machine Vision Conference*, pages 124.1–124.11. BMVA Press, 2009. ISBN 1-901725-39-1. doi:10.5244/C.23.124.

[13] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477, 2003.

[14] Denis Schluppeck. Science still battles to understand the human eye. 2016. URL `https://www.ft.com/content/3f2eb926-654c-11e6-8310-ecf0bddad227`.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. URL `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf`.

[16] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

[17] Camille Couprie, Clément Farabet, Laurent Najman, and Yann LeCun. Indoor semantic segmentation using depth information. *arXiv preprint arXiv:1301.3572*, 2013.

[18] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

[19] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014.

[20] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[21] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pages 4507–4515, 2015.

[22] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593, 2016.

[23] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014.

[24] Sepp Hochreiter and Jrgen Schmidhuber. Long short-term memory. 9:1735–80, 12 1997.

[25] Zuxuan Wu, Yu-Gang Jiang, Xi Wang, Hao Ye, and Xiangyang Xue. Multi-stream multi-class fusion of deep networks for video classification. In *Proceedings of the 2016 ACM on Multimedia Conference*, MM '16, pages 791–800, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3603-1. doi: 10.1145/2964284.2964328. URL `http://doi.acm.org/10.1145/2964284.2964328`.

[26] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702, 2015.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[28] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[29] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL `http://arxiv.org/abs/1412.6980`.

[30] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[31] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[32] Roger S Pressman. *Software engineering: a practitioner's approach*. Palgrave Macmillan, 2005.