Research Proposal: 3D Convolutions for deep learning and its applications to surveillance

Raesetje Bonjo Sefala, 844165

May 10, 2017

Contents

Abstract

We consider using Convolutional Neural Networks(CNNs) to solve the problem of human action recognition in videos. A CNN is form of deep learning neural network model used to automate the task of feature extraction to effectively detect and classify multimedia. A 2 Dimensional Convolutional Neural Network (2D CNN) is a CNN which has a 2D kernel (filter) which is used to capture spatial features from an image frame while a 3 Dimensional Convolutional Neural Networks (3D CNN) uses a 3D kernel to capture both spatial and temporal features from a group of image frames(video). Although both models can classify videos, One might argue that the temporal dimension in 3D kernels allows 3D models to better recognise actions in videos since they can capture the motion information found in between the image frames of the video.This paper proposes to compare a 2D CNN model with a 3D CNN model. The best model will be used to classify human actions from videos being fed to the model in real time.

1 Introduction

Human action recognition is an important branch of computer vision. It deals with using computing power, algorithms and sensors or data sets to try and enable computers to read and understand the environment, particularly what human beings are doing in a particular state.This knowledge can be applied in important fields such as surveillance, human computer interaction and gaming. Recognizing human actions in a videos is not yet an easy task to achieve due to low resolution videos, different points of view(as a result, a computer may view one subject in two different cases as two different subjects), cluttered backgrounds, etc. However, scientists have had a breakthrough in solving this problem with lower error rates by using machine learning techniques such as deep learning.

Deep learning is a method of machine learning which allows models which have multiple layers in between the input and output layers to learn data representations with multiple layers of abstraction [LeCun et al., 2015], [Fukushima and Miyake, 1982]. Convolutional Neural Network(CNN) is a form of deep learning neural network used to effectively detect and classify multimedia [LeCun et al., 2015]. A 2 Dimensional CNN (2D CNN) is a CNN which has a 2D kernel(filter) to create feature maps using only the spatial dimensions of the data while a 3D CNN uses both the spatial and temporal aspects of the data.

This paper aims to investigate whether the temporal dimension adds an advantage to a model trained to recognise actions. It essentially investigating whether a 3D CNN performs better than a 2D CNN in the task of recognizing actions in videos. Performance will be measured by of the amount of the recognition error each model will produce and the time taken by each model to produce results given the same data set and hardware. We will be using the KTH data set which includes 2391 videos taken over a homogeneous background. The videos have 6 categories of human actions(walking, jogging, running, boxing, hand waving and hand clapping).

2 Background and Related Work

2.1 Introduction

The previous chapter introduced the aim of the paper and some definitions and concepts relating to the main focus of the paper. This section will be investigating the background required to construct CNNs and some related work.

2.2 Background

2.2.1 2D CNNs

In 2D CNNs, a 2D kernel performs 2D convolutions on a 2D input(an image) at the convolutional layers to get features to form a new feature map. The value of unit at position (x, y) in the jth feature map in the ith layer, is given by:

$$v_{ij}^{xy} = \tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}\right), \tag{1}$$

where tanh() is the hyperbolic tangent function, bij is the bias for this feature map, m indexes over the set of feature maps in the (i 1)th layer connected to the current feature map, and the wth term is the value at the position (P, Q) of the kernel connected to the kth feature map, and Pi and Qi are the height and width of the kernel, respectively.
A CNN architecture can be built by putting together multiple layers of convolution and subsampling in alternating fashion.[Ji et al., 2013]

3 Methodology

3.1 Introduction

In order to test the two hypothesis stated in the previous chapter, a controlled experiment will be conducted. The KTH data set will be used to conduct this investigation. It consists of 2391 "simple" videos. The term "simple" in this case refers to the fact that the videos have a homogeneous background, are all in grey scale and with only one person appearing per video, performing a specific action. This section provides the details of the experiment.

3.2 Research Hypothesis

As discussed in the introduction, Convolutional Neural Networks(CNNs) can be used to classify actions in videos. Both 2D CNNs and 3D CNNs can be used to classify videos but the question is, which model performs better in terms of the amount of time taken to train and classify and the testing error value. We can fix the hyper parameters and use the same hardware and data set in order to evaluate the models. The proposed research hypotheses are:

3.2.1 Hypothesis 1:

The 3D CNN takes less time to train and classify as compared to the 2D CNN model.

3.2.2 Hypothesis 2:

The 3D CNN model produces a lower classification error value when tested with unseen data as compared to the 2D CNN model

For the above hypotheses to be tested, the following methodology will be carried out.

3.3 Methodology

3.3.1 Phase 1: Implementation

In order to find out which model between the 2D and 3D CNN performs better with reference to the hypotheses, the models should initially be built on the same hardware and with the same hyper parameters (i.e. the same number of features, size of features, the window size, window stride, number of nodes and the number of hidden layers) to prevent unfair advantage. The results of this phase will be two models, a 2D CNN and a 3D CNN ready for training.

3.3.2 Phase 2: Training

After building the two models, we now have to train the models using the KTH data set. The models' architecture will consist of methods such as Convolutional layers, Pooling, subsampling, and the Rectifying Linear Unit (ReLU) to prepare the features for the full connection layer. The fully connected layers will be using a cost function to improve the weights of the network. Before the training process begins a timer will be set up to take measurements for the first hypothesis (time taken aspect) and stopped at the end of training. The resulting product of this phase will be two trained CNN models ready for testing

3.3.3 Phase 3: Testing

The evolved models should be able to classify actions in the videos accordingly. The testing data will then be classified by the two models and the errors from both the models will be used to test the second hypothesis (which model yields the best classification accuracy). The results obtained can then be used to either reject or accept the research hypotheses.

3.3.4 Phase 4: User Interface design

In this last phase, a Web based interface will be built to allow real time monitoring of human action recognition from a small office space as part of the project. This system will be using the best model as determined by this investigation.

3.4 Conclusion

The 3D CNN can be labeled better than the 2D CNN if it takes less time to train and yields a lower error value. This section proposed a methodology to be used in the research project. It laid out a high level process to be followed in order to test the hypotheses for rejection or acceptance. The next chapter provides a more elaborate plan on how to implement the experiment.

4 Research Plan

4.1 Introduction

The previous section introduced the hypothesis and the research methodology. This section provides the details on the plan of how the methodology will be carried out. This section will be divided into 3 sub-sections. Sub-section 4.2 provides a detailed implementation of the methodology, Sub-section 4.3 looks into the potential issues which may arise and Sub-section 4.4 lays out the proposed time plan.

4.2 Deliverables

4.2.1 Phase 1: Implementation

In order to test the two hypotheses fairly, we need to control the hardware and all the hyper parameters for the models.

- **Hardware:** The NVIDIA titan Graphical Processing Unit (GPU) will be used to train the 2 models on a standard core i7 computer with 8 gigabytes of RAM.
- **Software:** The two models will be written in python 3.5. Theano and Kerras libraries will be used to implement both models
- **2D and 3D CNN models:**

  The 3D model will be built according to how it was built in ([Ji et al., 2013]). They followed the set up of the HMAX model where a 9-frame cube was used as input like in ([Jhuang et al., 2007]). To reduce the amount of memory used, the resolutions of the individual input frames where reduced to 80×60 pixels as opposed to the original spatial resolution of 160×120 pixels. We also propose to use the same 3D architecture used in ([Ji et al., 2013]). The architecture will be like that in figure 1. We will implement 3 convolutional layers, 2 sub-sampling layers and 1 full connection layer with a soft-max function and 6 output nodes representing the the 6 categories (walking, jogging, running, boxing, hand waving and hand clapping) from the KTH data set.

After completing this phase, the 2 resulting models should be ready for training.

4.2.2 Phase 2: Training

The KTH data set will be used to train and test the models. A 60/40 split between the training and testing data respectively will be used. The KTH data set consists of 2391 videos

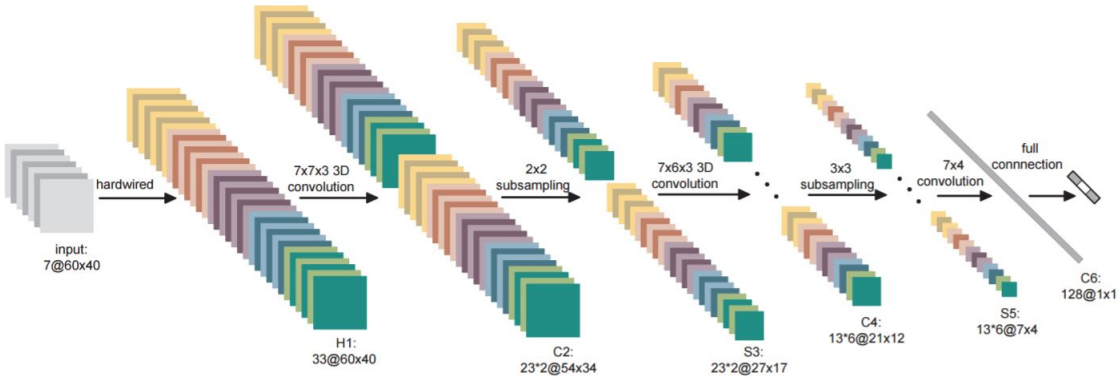

Figure 1: 3D CNN architecture [Ji et al., 2013]

in grey scale and with a homogeneous background. The videos have 25 subject but with only one subject appearing per video. Although we do not have to be concerned about multiple targets, we have other things such as human shadows which may appear in the video and so our model should be able to handle those.
A timer will be set up to record the amount of training time taken by each model as part of evaluating each model. Once the training phase ends, the models will be fully trained and ready for testing(i.e. the model will be able to classify the videos).

4.2.3 Phase 3: Testing

After testing the models with a 60/40 split on the data, the models should be ready for testing.Since the data set is labeled, we can use the labels to determine the classification accuracy. Before testing can begin, we need to set up a timer to record how long the testing takes for each model to classify stacks of image frames.
At the end of this phase, we should be able to compare the classification errors from the two models and then compare them with those of ([Ji et al., 2013]).

4.2.4 Phase 4: Build the user interface

Having tested the models and determining which model works best, we can now build a user interface using the best model to monitor a private room for the project. This will be a web based interface streaming over the room and then reporting on the activities happening in the room.

4.3 Potential issues

4.3.1 Training time

One potential issue could be in the training time of the models. If it happens that there is no GPU to help reduce the training time of the models, a CPU implementation will be used. Using the CPU implementation will result in longer training times and given that debugging and fine tuning the models is expected, this may use up a lot of the research time. To mitigate not meeting the deadline, the project implementation as a whole should start early to allow room for any disappointment.

4.3.2 No significant difference between the models

After the testing phase, it may be discovered that the two models perform equally as good in terms of their classification errors and the time taken to classify actions in videos. Although this situation is not ideal, it shows that it does not really matter which model you use for this data set. Depending on the time taken for model to classify the action enacted in a video, we can choose either model for the project.

4.4 Time plan

The following table illustrates a time journey of the implementation of the deliverables. Time for training and testing is not determined, this is to give room for any problems which may occur which will require training to be restarted. The entire project and preparation for presentations will end early so as to provide leeway should problems occur as well. No hours are allocated for times where there will be no human interaction with the system.

Week	**Activity**	**Time(Hours)**
May 10	Create a basic 2D CNN which classifies images	10
May 17	Create a basic 2D CNN which classifies images	10
May 24	*Examination Preparation*	–
May 31	*Examination Week*	–
June 07	*Examination Week*	–
June 14	*Winter Vacation*	–
June 21	*Winter Vacation*	–
June 28	Design and implement 2D CNN architecture	25
July 05	Design and implement 2D CNN architecture	25
July 12	*Attend RMB finance Winter school*	–
July 19	Design and implement 3D CNN architecture	25
July 19	Design and implement 3D CNN architecture	25
July 26	Training of the 2D and 3D CNN	–
August 2	Training of the 2D and 3D CNN	–
August 9	Training of the 2D and 3D CNN	–
August 16	Training of the 2D and 3D CNN	–
August 23	Training of the 2D and 3D CNN	–
August 30	Testing of the 2D and 3D CNN	–
September 6	Building the surveillance system using the best model	30
September 13	Building the surveillance system using the best model	30
September 20	Report write-up	30
September 27	Report write-up	30
October 4	Presentation preparation	20
October 11	Presentation preparation	20

4.5 Conclusion

The proposed research plan is driven by the deliverables. We do acknowledge that issues may arise during the project course which is why we aim to start early so that we do not miss the deadline. The activities to be carried out are ordered in the same way as the phases in the deliverables and each activity is given enough time so as to make room for failure and restarting. However, the working hours proposed in the time plan may change in time.

5 Conclusion

The question that if a 2D CNN can be used to recognise actions in videos, what are the advantages of using a 3D CNN which basically is a 2D CNN but with a temporal dimension aspect added to it? The aim of this experiment is to provide quantitative evidence on which model classifies quicker and more accurately given a specific data set.

The proposed experiment should be able to tell us that in the described controlled environment, which model between the 2D CNN and the 3D CNN performs better on recognising videos from the KTH data set. We are ultimately looking for a model which will be able to classify actions from videos in almost real time for an automated surveillance system.

We do not expect the trained model to be able to generalize over videos with more complex details per frame unless if we train the model with a different data set and then pre-process the input in order to generalise for things like multi-targets.

References

Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

Hueihan Jhuang, Thomas Serre, Lior Wolf, and Tomaso Poggio. A biologically inspired system for action recognition. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. Ieee, 2007.

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35 (1):221–231, 2013.

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