Table of Contents

Abbreviation

CNN	Convolution Neural Network
GPU	Graphical Processing unit
MLP	Multi Layer Perception

Project Title

Improved Iterative Perceptual feature Extraction for Semantic Human Action Recognition using 3D CNN

Chapter 1 - Introduction

Abstract

Deep Learning is a subset collection of Machine Learning concerned where neural network algorithms inspired by human brain (what occurs spontaneously to human) learn from large amount of data through several layers for nonlinear transformation. The deep learning can process large number of features to increase the outcome accuracy.

The topic of Human activity recognition (HAR) is a prominent research area topic in the field of computer vision and image processing area. It has empowered state-of-art application in multiple sectors, surveillance, digital entertainment and medical healthcare. It is interesting to observe and intriguing to predict such kind of movements.

Human Activity Recognition has increased a great deal in research field particularly context-aware computing and multimedia - for the most part on the record of its ubiquity in human life and furthermore on our consistently expanding computational capacity. It is as a rule effectively sought after for a wide range of uses like keen homes, human conduct analysis, sports and even security frameworks.

The proposed application Human Activity Recognition is based on Deep Learning which is used to identify and verify the human activities from the videos. Deep Learning Algorithms leverage large datasets of human activities and learn from rich set of features and train the models and eventually outperform the human activities.

Scope of the Project

The main contributions of this project are:

- Load the Image Dataset
- Image Preprocessing
- Image Vectorization
- Training the Model
- Testing of Test Data

Domain Overview

Deep learning has had a tremendous impact on various fields of technology in the last few years. One of the hottest topics buzzing in this industry is computer vision, the ability for computers to understand images and videos on their own. Self-driving

cars, biometrics and facial recognition all rely on computer vision to work. At the core of computer vision is image processing.

An image is represented by its dimensions (height and width) based on the number of pixels. For example, if the dimensions of an image are 500 x 400 (width x height), the total number of pixels in the image is 200000.

This pixel is a point on the image that takes on a specific shade, opacity or color. It is usually represented in one of the following:

1. **Grayscale** - A pixel is an integer with a value between 0 to 255 (0 is completely black and 255 is completely white).
2. **RGB** - A pixel is made up of 3 integers between 0 to 255 (the integers represent the intensity of red, green, and blue).
3. **RGBA** - It is an extension of RGB with an added alpha field, which represents the opacity of the image.

Image processing requires fixed sequences of operations that are performed at each pixel of an image. The image processor performs the first sequence of operations on the image, pixel by pixel. Once this is fully done, it will begin to perform the second operation, and so on. The output value of these operations can be computed at any pixel of the image.

Image processing is the process of transforming an image into a digital form and performing certain operations to get some useful information from it. The image processing system usually treats all images as 2D signals when applying certain predetermined signal processing methods.

There are two types of methods used for image processing namely, analogue and digital image processing. Analogue image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. Digital image processing techniques help in manipulation of the digital images by using computers. The three general phases that all types of data have to undergo while using digital technique are pre-processing, enhancement, and display, information extraction.

Fundamental Image Processing Steps

Image Acquisition
Image acquisition is the first step in image processing. This step is also known as preprocessing in image processing. It involves retrieving the image from a source, usually a hardware-based source.

Image Enhancement
Image enhancement is the process of bringing out and highlighting certain features of interest in an image that has been obscured. This can involve changing the brightness, contrast, etc.

Image Restoration
Image restoration is the process of improving the appearance of an image. However, unlike image enhancement, image restoration is done using certain mathematical or probabilistic models.

Color Image Processing
Color image processing includes a number of color modeling techniques in a digital domain. This step has gained prominence due to the significant use of digital images over the internet.

Wavelets and Multiresolution Processing
Wavelets are used to represent images in various degrees of resolution. The images are subdivided into wavelets or smaller regions for data compression and for pyramidal representation.

Compression
Compression is a process used to reduce the storage required to save an image or the bandwidth required to transmit it. This is done particularly when the image is for use on the Internet.

Morphological Processing
Morphological processing is a set of processing operations for morphing images based on their shapes.

Segmentation
Segmentation is one of the most difficult steps of image processing. It involves partitioning an image into its constituent parts or objects.

Representation and Description

After an image is segmented into regions in the segmentation process, each region is represented and described in a form suitable for further computer processing. Representation deals with the imageâ€™s characteristics and regional properties. Description deals with extracting quantitative information that helps differentiate one class of objects from the other.

Recognition

Recognition assigns a label to an object based on its description.

Object identification in a scene is a nontrivial image recognition problem; it is difficult to arbitrarily draw a distinction between what is background and what is foreground within an image. Since object identification is not the primary focus of our work, we simplify some of the details surrounding image recognition in our domain by taking advantage of the background knowledge available in our domain. Namely, we know we are detecting blocks in a fairly static collaboration space. By static, we mean that the physical space does not move; rather, change comes in the form of new objects added to the space or the movement of objects in the static space, as opposed to changing the actual collaboration space itself.

Image pre-processing is the term for operations on images at the lowest level of abstraction. These operations do not increase image information content but they decrease it if entropy is an information measure. The aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task.

The aim of pre-processing is to improve the quality of the image so that we can analyse it in a better way. By preprocessing we can suppress undesired distortions and enhance some features which are necessary for the particular application we are working for. Those features might vary for different applications.

There are 4 different types of Image Pre-Processing techniques and they are listed below.

1. **Pixel brightness transformations(PBT)**
   Brightness transformations modify pixel brightness and the transformation depends on the properties of a pixel itself. In PBT, output pixelâ€™s value depends only on the corresponding input pixel value. Examples of such operators include brightness and contrast adjustments as well as colour correction and transformations.

   There are two types of Brightness transformations and they are below.

   1. Brightness corrections
   2. Gray scale transformation

   The most common Pixel brightness transforms operations are

   1. **Gamma Correction**
      Gamma correction is a non-linear adjustment to individual pixel values. While in image normalization we carried out linear operations on individual pixels, such as scalar multiplication and addition/subtraction, gamma correction carries out a non-linear operation on the source image pixels, and can cause saturation of the image being altered.

   2. **Sigmoid stretching**
      Sigmoid function is a continuous nonlinear activation function. The name, sigmoid, is obtained from the fact that the function is S shaped. Statisticians call this function the logistic function.

   3. **Histogram equalization**
      A histogram of an image is the representation of the intensity vs the number of pixels with that intensity. For example, a dark image will have many pixels which are black and few which are white. Representing that like a graph is what is called a histogram.

      Histogram equalization is a well-known contrast enhancement technique due to its performance on almost all types of image. Histogram equalization provides a sophisticated method for modifying the dynamic range and contrast of an image by altering that image such that its intensity histogram has the desired shape. Unlike contrast stretching, histogram modelling operators may employ non-linear and non-monotonic transfer functions to map between pixel intensity values in the input and output images.

2. **Geometric Transformations**
   Geometric transforms permit the elimination of geometric distortion that occurs when an image is captured. The normal

Geometric transformation operations are rotation, scaling and distortion (or undistortion!) of images.

Affine Transformation : Instead of defining the scale factors, the shearing factors and the rotation angle, it is common to merge these three transformation into one matrix. The combination of the four transformations is therefore defined as Affine Transformation

Perspective Transformation : change the perspective of a given image or video for getting better insights about the required information. Here the points needs to be provided on the image from which want to gather information by changing the perspective.

3. **Image Filtering and Segmentation**
   The goal of using filters is to modify or enhance image properties and/or to extract valuable information from the pictures such as edges, corners, and blobs. A filter is defined by a kernel, which is a small array applied to each pixel and its neighbors within an image

   Some of the basic filtering techniques are

   1. **Low Pass Filtering (Smoothing)**
      A low pass filter is the basis for most smoothing methods. An image is smoothed by decreasing the disparity between pixel values by averaging nearby pixels

   2. **High pass filters (Edge Detection, Sharpening)**
      High-pass filter can be used to make an image appear sharper. These filters emphasize fine details in the image â€“ the opposite of the low-pass filter. High-pass filtering works in the same way as low-pass filtering; it just uses a different convolution kernel.

   3. **Directional Filtering :**
      Directional filter is an edge detector that can be used to compute the first derivatives of an image. The first derivatives (or slopes) are most evident when a large change occurs between adjacent pixel values.Directional filters can be designed for any direction within a given space

   4. **Laplacian Filtering**
      Laplacian filter is an edge detector used to compute the second derivatives of an image, measuring the rate at which the first derivatives change. This determines if a change in adjacent pixel values is from an edge or continuous progression. Laplacian filter kernels usually contain negative values in a cross pattern, centered within the array. The corners are either zero or positive values. The center value can be either negative or positive.

Image segmentation is a commonly used technique in digital image processing and analysis to partition an image into multiple parts or regions, often based on the characteristics of the pixels in the image. Image segmentation could involve separating foreground from background, or clustering regions of pixels based on similarities in colour or shape.

There are two types of image segmentation techniques.

1. **Non-contextual thresholding**
   Thresholding is the simplest non-contextual segmentation technique. With a single threshold, it transforms a greyscale or colour image into a binary image considered as a binary region map. The binary map contains two possibly disjoint regions, one of them containing pixels with input data values smaller than a threshold and another relating to the input values that are at or above the threshold.

2. **Contextual segmentation**
   Non-contextual thresholding groups pixels with no account of their relative locations in the image plane. Contextual segmentation can be more successful in separating individual objects because it accounts for closeness of pixels that belong to an individual object. Two basic approaches to contextual segmentation are based on signal discontinuity or similarity. Discontinuity-based techniques attempt to find complete boundaries enclosing relatively uniform regions assuming abrupt signal changes across each boundary. Similarity-based techniques attempt to directly create these uniform regions by grouping together connected pixels that satisfy certain similarity criteria. Both the approaches mirror each other, in the sense that a complete boundary splits one region into two.

4. **Fourier transform**

   The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image.

   The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, image reconstruction and image compression.

   The DFT(Discrete Fourier Transform) is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image, i.e. the image in the spatial and Fourier domain are of the same size.

Literature Survey 1	
Title	Cross-Agent Action Recognition
Authors	Hongsong Wang and Liang Wang
Published Year	2018
Efficiency	It is a fast and easy procedure to perform Reduces the consumption of hardware resources Capable of further reducing the required level of human effort
Drawbacks	Additional configuration is required Solutions have been proved ineffective Difficulties to obtain better performance
Description	An action is something which is done by an agent. Most action recognition researchers merely focus on the actions to be recognized, and ignore the differences of agents. Philosophers and behaviorists discover that actions are common among many species, but are performed in different ways and with different levels of sophistication. In this paper, in order to bridge action recognition tasks between different agents, we introduce a new problem, cross-agent action recognition, i.e., recognizing action for one particular agent (target) while training from other agents (source). We model this problem under three different scenarios: single source and single target, multiple sources and single target, and multiple sources and multiple targets. To this end, corresponding methods based on transfer learning are proposed to address these problems. We further design three different strategies to model the situation when a partial labeled data is provided for the target. Experimental results show that the performances of the transfer method are generally better than those of the comparative method without transfer learning, especially when we have multiple sources. Particularly, the transfer method outperforms the others significantly when the source is a human adult. In addition, cross-agent method significantly improves the results when partially labeled data is provided for the target. These demonstrate that for action recognition, knowledge can be transferred across different agents. A straightforward application of this finding is to use human action (training data is abundant) data to enhance animal action recognition. In this paper, we introduce a new problem, cross-agent action recognition, for which the training data set is from one agent (source), and the test data set is from other agents (target). We formulate it under three different conditions: single source and single target, multiple sources and single target, multiple sources and multiple targets, and propose new models based on transfer learning to solve these problems. We apply our model to the case when the agent is a species and conduct different experiments according to the three different conditions. Our results are compared with those of the non- transfer method. We further design experiments to answer the question of whether our cross-agent method is necessary when we have a partially labeled data for the target agent.

Literature Survey 2	
Title	# Skeleton-Based Human Action Recognition With Global Context-Aware Attention LSTM Networks
Authors	Jun Liu , Gang Wang , Ling-Yu Duan , Kamila Abdiyeva and Alex C. Kot
Published Year	2018
Efficiency	Eliminating the huge workload of traditional methods Lowering the Complexity Threshold Reduces the consumption of hardware resources
Drawbacks	Difficulties to obtain better performance Heavyweight Solutions have been proved ineffective
Description	Human action recognition in 3D skeleton sequences has attracted a lot of research attention. Recently, long short-term memory (LSTM) networks have shown promising performance in this task due to their strengths in modeling the dependencies and dynamics in sequential data. As not all skeletal joints are informative for action recognition, and the irrelevant joints often bring noise which can degrade the performance, we need to pay more attention to the informative ones. However, the original LSTM network does not have explicit attention ability. In this paper, we propose a new class of LSTM network, global context-aware attention LSTM, for skeleton-based action recognition, which is capable of selectively focusing on the informative joints in each frame by using a global context memory cell. To further improve the attention capability, we also introduce a recurrent attention mechanism, with which the attention performance of our network can be enhanced progressively. Besides, a two-stream framework, which leverages coarse-grained attention and fine-grained attention, is also introduced. The proposed method achieves state-of-the-art performance on five challenging datasets for skeleton-based action recognition. In this paper, we have extended the original LSTM net- work to construct a Global Context-Aware Attention LSTM (GCA-LSTM) network for skeleton based action recognition, which has strong ability in selectively focusing on the infor- mative joints in each frame of the skeleton sequence with the assistance of global context information. Furthermore, we have proposed a recurrent attention mechanism for our GCA-LSTM network, in which the selectively focusing capa- bility is improved iteratively. In addition, a two-stream atten- tion framework is also introduced. The experimental results validate the contributions of our approach by achieving state- of-the-art performance on ve challenging datasets. ACKNOWLEDGEMENT This work was carried out at the Rapid-Rich Object Search (ROSE) Lab at Nanyang Technological Univer- sity (NTU), Singapore. The authors acknowledge the support of NVIDIA AI Technology Centre (NVAITC) for the donation of the Tesla K40 and K80 GPUs used for their research at the ROSE Lab. J. Liu would like to thank Qiuhong Ke from University of Western Australia for helpful discussions.

Literature Survey 3	
Title	# On Space-Time Filtering Framework for Matching Human Actions Across Different Viewpoints
Authors	Anwaar Ulhaq , Xiaoxia Yin , Jing He and Yanchun Zhang
Published Year	2018
Efficiency	Quick and Efficient to use Achieve sub-optimal performance. Minimizes the workload on infrastructures.
Drawbacks	Heavyweight High complexity of installing and maintaining Cannot meet current network business demands
Description	Space-time template matching is considered as a promising approach for human action recognition. However, a major drawback of template-based methods is computational overhead due to matching in spatial domain. Recently, space-time correlation-based action filters have been proposed for recognizing human actions in frequency domain. These action filters present reduction in time complexity as Fourier transform-based matching is faster than spatial template matching. However, the utility of such action filters is challenged due to a number of factors: 1) inability to deal with view variations due to implicit lack of support for view-invariance; 2) these filters can be trained only for one action class at a time, and separate filters are required for each action class with increased computational overhead; 3) these filters simply take average of similar action instances and behave no better than average filters; and 4) slightly misaligned action data sets create problems as these filters are not shift-invariant. In this paper, we try to address these shortcomings by proposing an advanced space-time filtering framework for recognizing human actions despite large viewpoint variations. Rather than using crude intensity values, we use 3D tensor structure at each pixel, which characterizes the most common local motion in action sequences. Discrete tensor Fourier transform is then applied to achieve frequency domain representations. Then, we form view clusters from multiple view action data and use space-time correlation filtering to achieve discriminative view representations. These representations are used in an innovative way to achieve action recognition despite viewpoint variations. Extensive experimentation is performed on well-known multiple view action data sets, including IXMAS, WVU, and N-UCLA action data set. A detailed performance comparison with the existing view-invariant action recognition techniques indicates that our approach works equally well for RGB and RGB-D video data with increased accuracy and efficiency. In this paper, we propose the concept of space-time cor- relation ltering for matching human actions captured from different viewpoints. It is based on maximizing spectral sep- aration that is very useful for separating multiple classes. The proposed space-time frequency domain lter overcomes the weaknesses of existing correlation lters by presenting improvements which include: (i) support for view invariance action recognition framework, (ii) single space-time lter for multiple action classes in single view cluster decreasing computational overhead, (iii) shift-invariant distance providing more generalization for test sequences and (iv) improved intra- class similarity measure contributing balanced treatment of low and high frequency information.

Literature Survey 4	
Title	# Bio-Inspired Human Action Recognition With a Micro-Doppler Sonar System
Authors	Thomas S. Murray , Daniel R. Mendat , Kayode A. Sanni , Philippe O. Pouliquen and Andreas G. Andreou
Published Year	2018
Efficiency	Achieve a well-balanced tradeoff among various parameters. May not meet the real-time requirement. Improve the operational efficiency.
Drawbacks	This system is Opportunistic and uncontrollable Complexity of its Real Time Implementation Cannot be implemented real time
Description	This paper explores computational methods to address the problem of doing inference from data in multiple modalities, where there exists a large amount of low dimensional data complementary to a much smaller set of high dimensional data. In this instance the low dimensional time-series data are active acoustics from a bio-inspired micro-Doppler sonar sensor system that include no or very limited spatial information, and the high dimensional data are RGB-depth data from a 3-D point cloud sensor. The task is human action recognition from the active acoustic data. To accomplish this, statistical models, trained simultaneously on both the micro-Doppler modulations induced by human actions and symbolic representations of skeletal poses, derived from the 3-D point cloud data, are developed. This simultaneous training enables the model to learn relations between the rich temporal structure of the micro-Doppler modulations and the high-dimensional pose sequences of human action. During runtime, the model relies purely on the active acoustic sonar data to infer the human action. Our approach is applicable to other sensing modalities, such as the millimeter wave electromagnetic radar devices. Using a multimodal dataset that incorporates both visual data, which facilitates the accurate tracking of human move- ment, and active acoustic data, which captures the micro- Doppler modulations induced by the motion, we have devel- oped algorithms for action recognition. The dataset consists of twenty-one actions and focuses on examples of orientational symmetry that a single active ultrasound sensor should have the most difculty discriminating. The combined results from three independent ultrasound sensors are encouraging and provide a foundation to explore the use of data from multiple viewpoints to resolve the orientational ambiguity in action recognition. Future lines of research are intended to explore the ap- plicability of the sensor to real-life scenarios. In this sense, experiments will be developed to evaluate aspects such as the distance limits of the system, especially in outdoor conditions, and the effects on accuracy of the angle of incidence between the ultrasonic module and the target object. One key aspect here is the potential active control of the micro-Doppler sonar for interrogating the scene, as, unlike audio, which comes from all directions and without control, the sonar device can be ac- tivated intermittently and directed towards the desired objects. The ability to disentangle the acoustic modulations of multiple people moving simultaneously and interacting with each other is another obstacle that will require future work to address.

Literature Survey 5	
Title	# Fisherposes for Human Action Recognition Using Kinect Sensor Data
Authors	Benyamin Ghojogh , Hoda Mohammadzade and Mozhgan Mokari
Published Year	2018
Efficiency	It is a fast and easy procedure to perform Simplify the implementation process. Lowering the Complexity Threshold
Drawbacks	Narrowly specialized knowledge Poor Application Performance Difficult and Less Commonly used
Description	This paper proposes a new method for view-invariant action recognition that utilizes the temporal position of skeletal joints obtained by Kinect sensor. In this method, the actions are represented as sequences of several pre-defined poses. After pre-processing, which includes skeleton alignment and scaling, the appropriate feature vectors are obtained for recognizing and discriminating the pose of every frame by the proposed Fisherposes method. The proposed regularized Mahalanobis distance metric is used in order to recognize both the involuntary and highly made-up actions at the same time. Hidden Markov model (HMM) is then used to classify the action related to an input sequence of poses. For taking into account the motion in the actions which are not separable by solely their temporal poses, histograms of trajectories are also proposed. The proposed action recognition method is capable of recognizing both the voluntary and involuntary actions, as well as pose-based and trajectory-based ones with a high accuracy rate. The effectiveness of the proposed method is experimented on three publicly available data sets, TST fall detection, UTKinect, and UCFKinect data sets. This article proposed a method for recognizing involuntary, normal and constrained actions. This method uses an effective framework for fusing the result of the proposed pose-based and trajectory-based approaches whenever it is required, and thus does not face any serious problem in various datasets. Experiments showed that this method performs well on the datasets including different types of actions. In this work, using the proposed Fisherpose method, a fea- ture vector is created for recognizing the pose of the body in each frame. Action is recognized by constructing sequence of poses and using HMM to model sequences of each action. Therefore, the proposed method is robust to different sequence length of actions and hence to different speed of performing actions. In addition, regularized Mahalanobis distance is pro- posed and utilized for considering both advantages of Euclid-ean and Mahalanobis distances for simultaneous recognition of involuntary and voluntary actions. The authors thank reviewers for their precious comments which improved quality of this work.

Literature Survey 6	
Title	# Human Action Recognition Algorithm Based on Multi-Feature Map Fusion
Authors	Haofei Wang and Junfeng Li
Published Year	2020
Efficiency	Keeping the control overhead at regular levels Tolerates Variations problems are solved on an end-to-end basis
Drawbacks	Signicantly increases capital and operating expenditures Solutions have been proved ineffective High complexity of installing and maintaining
Description	The emergence of the convolutional neural network greatly improves the accuracy of human action recognition. However, with the deepening of the network, fewer and fewer features are extracted, and in some datasets, due to the shooting angle, the size of the target to be recognized is different. To solve this problem, on the basis of resnext human action recognition method, we propose an improved resnext human action recognition method based on multi-feature map fusion. First, the video is uniformly sampled to generate training samples, and we generate samples with different frames as the input to the network. Second, we add n layers of up-sampling layers after layer 1 of resnext, to enlarge the feature maps and extract multiple feature maps, so that the extracted feature maps are clearer, and small targets can be better recognized. Finally, for the n results obtained, we use the weighted geometric means combination forecasting method based on L_1 norm to fuse and obtain the final result. In the process of experiment, using UCF-101 and HMDB-51 for verification, the accuracy of our model is 90.3% on UCF-101, which is higher than most of the state-of-art algorithms. At present, human action recognition is the focus and dif- culty of research and has a very wide application prospect, mainly used in monitoring, human-computer interaction, and other scenarios. Due to the complexity and diversity of human action, research on human action recognition has great challenges. This paper presented a solution to improve the performance of human action recognition. we proposed the architecture based on Multi-feature Map Fusion, which uses multiple up-sampling layers to enlarge feature maps, so that smaller targets can be better detected, at the same time, the information of the features extracted by the network is more and clearer. In our architecture, for the up-sampling method, the nearest neighbor interpolation method, the bilin- ear interpolation method, and the trilinear interpolation method have been studies. Experiments show that the effect of the feature map obtained by the trilinear interpolation method is better than the other two methods. Simultaneously, we used the clip with different sample-durations for training. The results indicate that with the number of sample-duration increases, the accuracies also improve. Finally, for the results obtained by the network, we did not use the method of averaging scores as mentioned in [14] to fuse the results. We proposed to use the weighted geometric means combi- nation forecasting method based on L_1 norm to fuse the obtained n results. The proposed architecture achieved 90.3% and 58.4% on UCF-101 and HMDB-51, which illustrates that the architecture is effective and comparable.

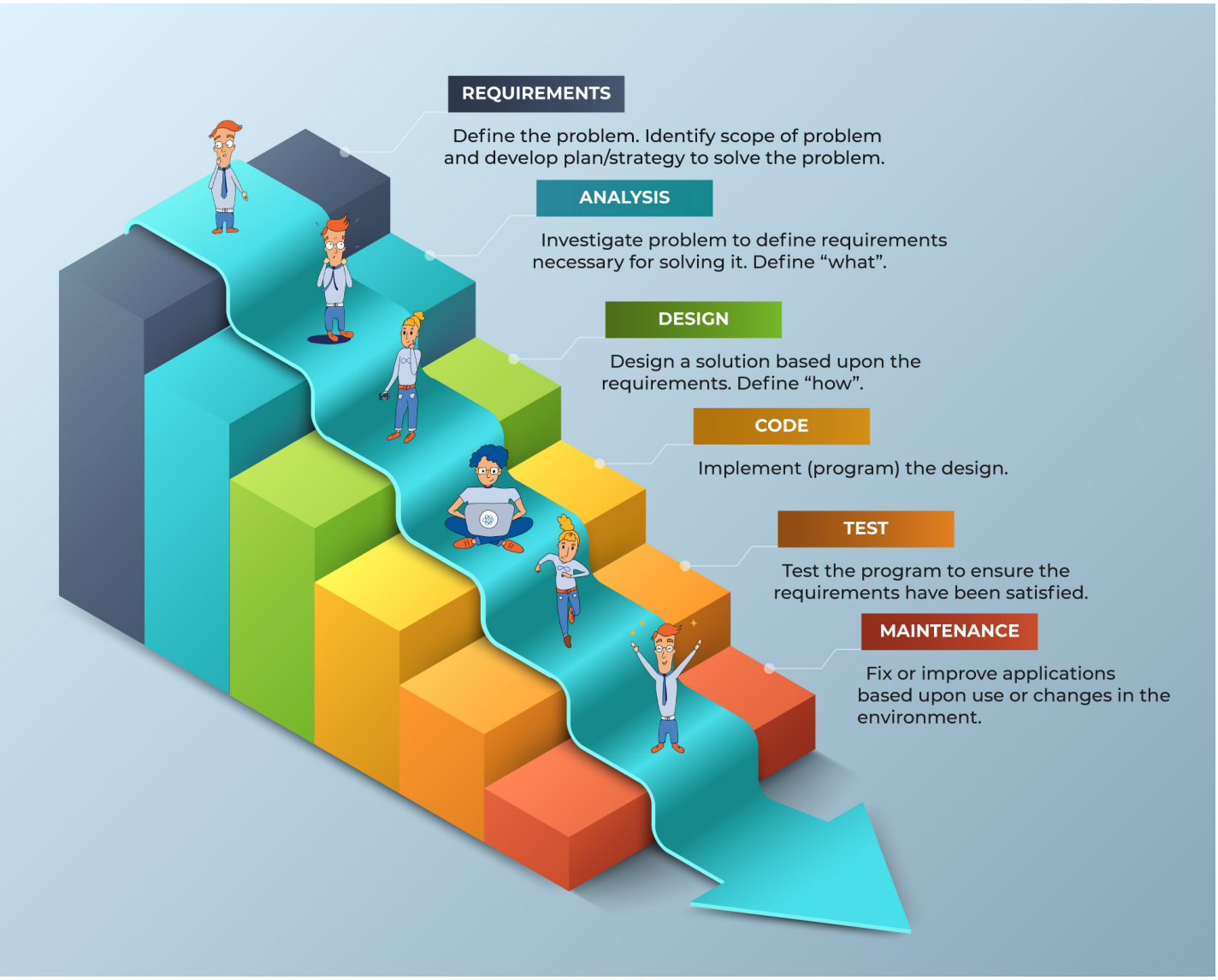
Literature Survey 7	
Title	# A Context Knowledge Map Guided Coarse-to-Fine Action Recognition
Authors	Yanli Ji , Yue Zhan , Yang Yang , Xing Xu , Fumin Shen and Heng Tao Shen
Published Year	2020
Efficiency	Offer increased flexibility Fast and efficient, but also as accurate as the state-of-the-art algorithms Achieve a well-balanced tradeoff among various parameters.
Drawbacks	Cannot meet current network business demands Difficult and Less Commonly used Difficult to be used in large-scale parallel computing.
Description	Human actions involve a wide variety and a large number of categories, which leads to a big challenge in action recognition. However, according to similarities on human body poses, scenes, interactive objects, human actions can be grouped into some semantic groups, i.e. sports, cooking, etc. Therefore, in this paper, we propose a novel approach which recognizes human actions from coarse to fine. Taking full advantage of contributions from high-level semantic contexts, a context knowledge map guided recognition method is designed to realize the coarse-to-fine procedure. In the approach, we define semantic contexts with interactive objects, scenes and body motions in action videos, and build a context knowledge map to automatically define coarse-grained groups. Then fine-grained classifiers are proposed to realize accurate action recognition. The coarse-to-fine procedure narrows action categories in target classifiers, so it is beneficial to improving recognition performance. We evaluate the proposed approach on the CCV, the HMDB-51, and the UCF101 database. Experiments verify its significant effectiveness, on average, improving more than 5% of recognition precisions than current approaches. Compared with the state-of-the-art, it also obtains outstanding performance. The proposed approach achieves higher accuracies of 93.1%, 95.4% and 74.5% in the CCV, the UCF-101 and the HMDB51 database, respectively. Since human actions involve complex semantic contexts which contribute to action recognition, we proposed a con- text knowledge map guided coarse-to-ne action recognition approach. In this paper, we extracted and fused multiple semantic contexts in action sequences, and used these semantic contexts to build a context knowledge map. Based on the map, the proposed approach adaptively separated actions to coarse granularity groups with high precisions, and a coarse-to-ne classication model was proposed for accurate recognition. We evaluated the proposed approach on the UCF101, HMDB- 51 and the CCV databases. Experiments veried the signicant effectiveness of the proposed approach. In addition, the pro- posed approach effectively improved recognition accuracies of ne-grained actions. Compared with the state-of-the-art approaches, our approach achieved outstanding performance in three databases. To achieve a better performance, we enhance our approach step by step in the experiment. Therefore, it is not available for end-to-end training. In our future research, we will try to integrate these procedures into an integrated network and try to reduce the complexity of the approach.

Chapter 3 - System Analysis

Waterfall Model

It is called as traditional approach. Waterfall is a linear method (sequential model) for any software application development. Here application development is segregated into a sequence of pre -defined phases.

Waterfall Model (Taken from Google.com)



1. Requirement gathering and documentation
   In this stage, you should gather comprehensive data approximately what this challenge requires. You may gather this data in a diffusion of ways, from interviews to questionnaires to interactive brainstorming. By means of the stop of this section, the task requirements have to be clean, and also you have to have a necessities file that has been allotted on your team.

2. System design
   Using the well-known requirements, your team designs the solutions. During this phase, no development will be happening. But the project team starts specification such as programming language or hardware requirements.

3. Implementation
   During this phase software development coding will be happening. Web application programmers take data from the previous stage and create a functional product. Web application programmers write source code in small pieces, which are integrated at the end of this phase or the beginning of the next.

4. Testing

Once all coding is done, testing of the product can begin. Testers methodically find and report any problems. If serious issues arise, your project may need to return to phase one for revaluation.

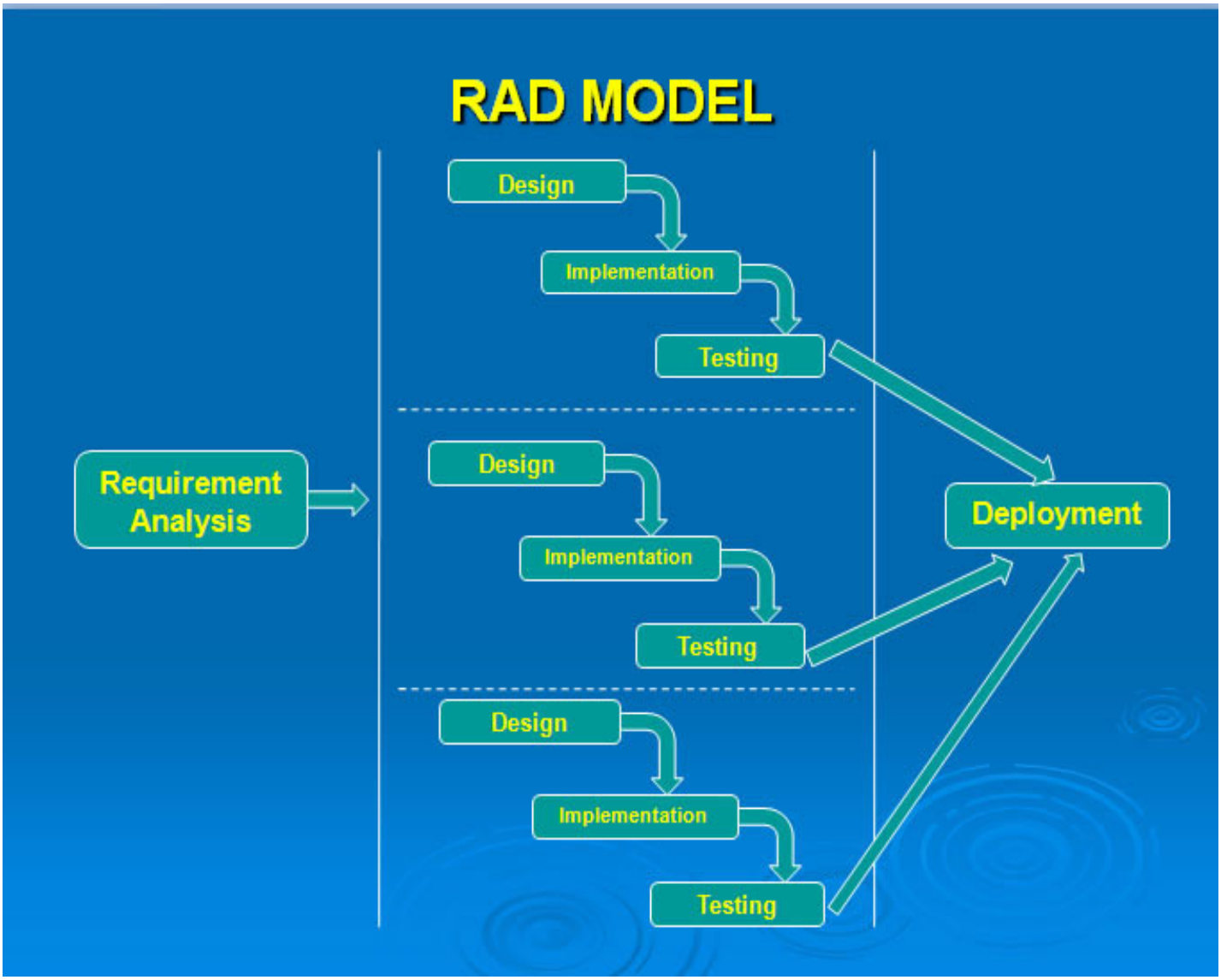
5. Delivery/deployment
   In this phase, the solution is complete, and your project team submits the deliverables to be deployed or released.

6. Maintenance
   The final solution has been implemented to the client and is being used. As troubles arises, the project team might also want to create patches and updates might also to deal with them. Again, huge troubles may also necessitate a return to segment one.

RAD Model

Rapid application development is an agile software development approach that focuses more on ongoing software projects and user feedback and less on following a strict plan. As such, it emphasizes rapid prototyping over costly planning.

RAD Model (Taken from Google.com)



1. Define Requirements
   Rather than making you spend months developing specifications with users, RAD begins by defining a loose set of requirements. â€œLooseâ€ because among the key principles of rapid application development is the permission to change requirements at any point in the cycle.

   Basically, developers gather the products gist. The client provides their vision for the product and comes to an agreement

with developers on the requirements that satisfy that vision.

This phase is equivalent to a project scoping meeting. Although the planning phase is condensed compared to other project management methodologies, this is a critical step for the ultimate success of the project.

During this stage, web application programmers, clients (software users), and team members communicate to determine the goals and expectations for the project as well as current and potential issues that would need to be addressed during the build.

Researching the current problem
Defining the requirements for the project
Finalizing the requirements with each stakeholders approval

A basic breakdown of this stage involves:

It is important that everyone has the opportunity to evaluate the goals and expectations for the project and weigh in. By getting approval from each key stakeholder and web application programmer, teams can avoid miscommunications and costly change orders down the road.

2. Prototype

In this rapid application development phase, the developerâ€™s goal is to build something that they can demonstrate to the client. This can be a prototype that satisfies all or only a portion of requirements (as in early-stage prototyping).

This prototype may cut corners to reach a working state, and thatâ€™s acceptable. Most RAD programming approaches have a finalization stage where developers pay down technical debt accrued by early prototypes.

All the bugs and kinks are worked out in an iterative process. The web application programmer designs a prototype, the client (user) tests it, and then they come together to communicate on what worked and what did not.

This method gives web application programmers the opportunity to tweak the model as they go until they reach a satisfactory design. Both the software web application programmers and the clients learn from the experience to make sure there is no potential for something to slip through the cracks.

3. Absorb Feedback

With a recent prototype prepared, RAD developers present their work to the client or end-users. They collect feedback on everything from interface to functionalityâ€”it is here where product requirements might come under scrutiny.

Clients may change their minds or discover that something that seemed right on paper makes no sense in practice. Clients are only human, after all. With feedback in hand, developers return to some form of step 2: they continue to prototype. If feedback is strictly positive, and the client is satisfied with the prototype, developers can move to step 4.

Because the majority of the problems and changes were addressed during the thorough iterative design phase, web application programmers can construct the final working model more quickly than they could by following a traditional project management approach.

The phase breaks down into several smaller steps:

Preparation for rapid construction
Program and application development
Coding
Unit, integration, and system testing

The software development team of programmers, coders, testers, and web application programmers work together during this stage to make sure everything is working smoothly and that the end result satisfies the clients expectations and objectives. This third phase is important because the client still gets to give input throughout the process. They can suggest alterations, changes, or even new ideas that can solve problems as they arise.

4. Finalize Product

During this stage, developers may optimize or even re-engineer their implementation to improve stability and maintainability. They may also spend this phase connecting the back-end to production data, writing thorough documentation, and doing any other maintenance tasks required before handing the product over with confidence.

Existing System

Skeleton-based human action recognition is becoming popular due to its computational efciency and robustness. Since not all skeleton joints are informative for action recognition, attention mechanisms are adopted to extract informative joints and suppress the inuence of irrelevant ones. However, existing attention frameworks usually ignore helpful scenario context information. In this paper, we propose a cross-attention module that consists of a self-attention branch and a cross-attention branch for skeleton-based action recognition. It helps to extract joints that are not only more informative but also highly correlated to the corresponding scenario context information. Moreover, the cross-attention module maintains input variables size and can be exibly incorporated into many existing frameworks without breaking their behaviors. To facilitate end-to-end training, we further develop a scenario context information extraction branch to extract context information from raw RGB video directly. We conduct comprehensive experiments on the NTU RGB+D and the Kinetics databases, and experimental results demonstrate the correctness and effectiveness of the proposed model.

Drawbacks of Existing System

- Narrowly specialized knowledge
- Difficult and Less Commonly used
- Poor Application Performance
- Complexity of its Real Time Implementation
- High complexity of installing and maintaining
- Heavyweight

Proposed System

The human recognition model is trained and learned based on neural network of the deep learning, which has strong robustness to illumination difference, human action change and human action occlusion.

The accuracy of the proposed system is largely depending on the many parameters. One of the main parameters is illumination condition. The best conventional utilized histogram normalization procedure is histogram equalization where one tries to adjust the image histogram into a histogram that is persistent for all brightness values.

The proposed model is a neural network with parameter W, which includes the weights matrix and bias parameters of all dense layers in the network. When one part of feature is not involved in the training, we remove the nodes and layers related to that part within the network.

The 3D neural network model used in this model is Transfer Learning. The videos were temporally cut down and last around tenth of a second. The trained model shows satisfactory performance in all stages of training, testing. Finally the results show promising activity recognition of over various human actions.

Advantages of Proposed System

Simplicity and Explainability.
Enhance correlation strength with finer and more compact information
Trustworthy and reliable, which refers to obtain explainability.
Quick and Efficient to use
Improve the operational efficiency.
Improve the quality and consistency of data

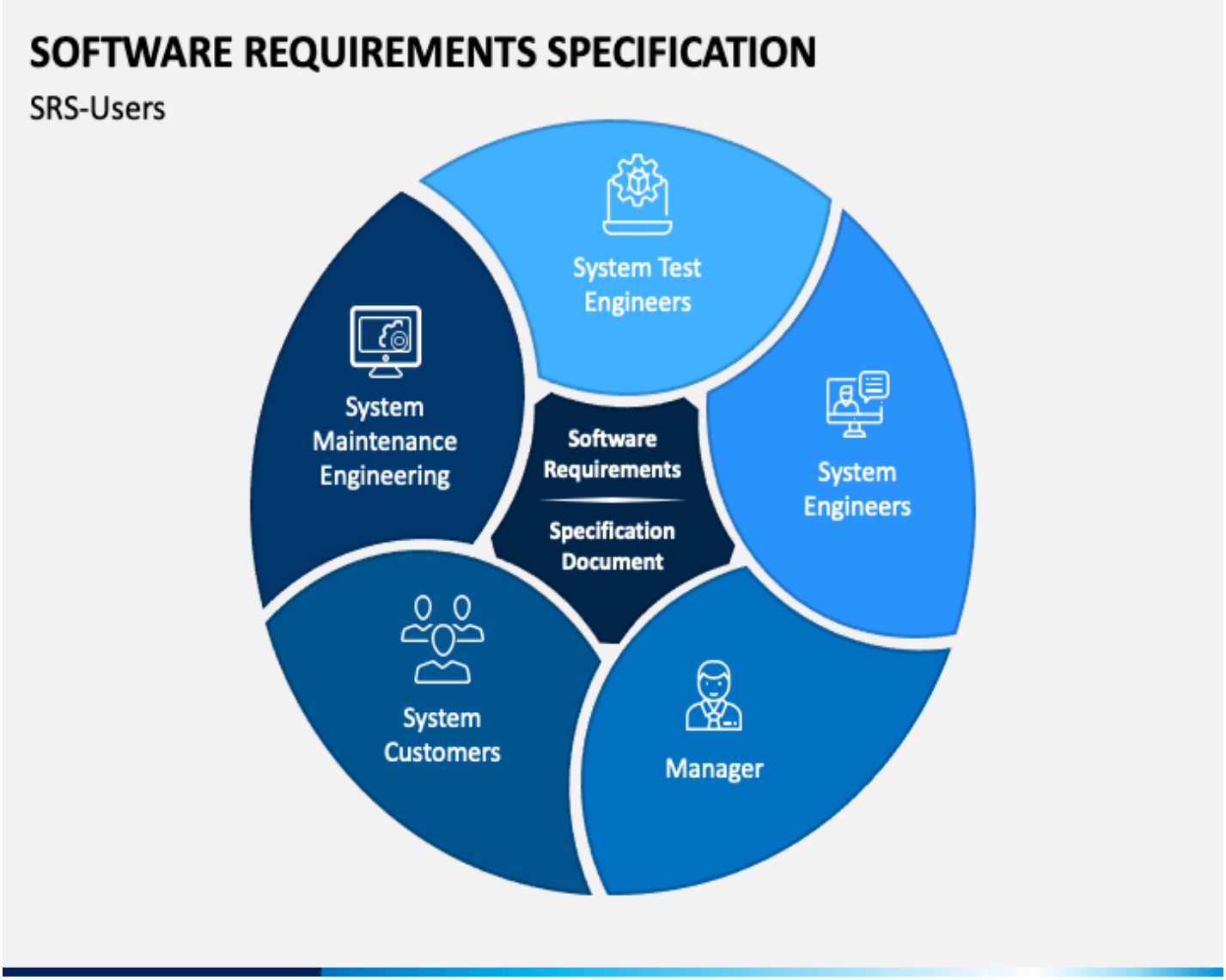
Improved traceability

Chapter 4 - Requirement Specification

Introduction

Clearly defined requirements are essential signs on the road that leads to a successful project. They establish a formal agreement between a client and a provider that they are both working to reach the same goal. High-quality, detailed requirements also help mitigate financial risks and keep the project on a schedule. According to the Business Analysis Body of Knowledge definition, requirements are a usable representation of a need.

Creating requirements is a complex task as it includes a set of processes such as elicitation, analysis, specification, validation, and management.



A System Requirements Specification (SRS) (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behavior of a system or software application.

Depending on the methodology employed (agile vs waterfall) the level of formality and detail in the SRS will vary, but in general an SRS should include a description of the functional requirements, system requirements, technical requirements, constraints, assumptions and acceptance criteria. Each of these is described in more detail below:

- Business Drivers
- Business Model
- Functional and System Requirements
- Business and System Use Cases

- Technical Requirements
- System Qualities
- Constraints and Assumptions
- Acceptance Criteria

Business Drivers

This section describes the reasons why the customer is looking to build the system. The rationale for the new system is important as it will guide the decisions made by the business analysts, system architects and developers. Another compelling reason for documenting the business rationale behind the system is that the customer may change personnel during the project. Documentation which clearly identifies the business reasons for the system will help sustain support for a project if the original sponsor moves on.

The drivers may include both problems (reasons why the current systems/processes are not sufficient) and opportunities (new business models that the system will make available). Usually a combination of problems and opportunities are needed to provide motivation for a new system.

Business Model

This section describes the underlying business model of the customer that the system will need to support. This includes such items as the organizational context, current-state and future-state diagrams, business context, key business functions and process flow diagrams. This section is usually created during the functional analysis phase.

Functional and System Requirements

This section usually consists of a hierarchical organization of requirements, with the business/functional requirements at the highest-level and the detailed system requirements listed as their child items.

Business and System Use Cases
This section usually consists of a UML use case diagram that illustrates the main external entities that will be interacting with the system together with the different use cases (objectives) that they will need to carry out. For each use-case there will be formal definition of the steps that need to be carried out to perform the business objective, together with any necessary pre-conditions and post-conditions.

The business use cases are usually derived from the functional requirements and the system use cases are usually derived from the system requirements.

Technical Requirements

This section is used to list any of the "non-functional" requirements that essentially embody the technical environment that the product needs to operate in, and include the technical constraints that it needs to operate under. These technical requirements are critical in determining how the higher-level functional requirements will get decomposed into the more specific system requirements.

System Qualities

This section is used to describe the "non-functional" requirements that define the "quality" of the system. These items are often known as the "-ilities" because most of them end in "ility". They included such items as: reliability, availability, serviceability, security, scalability, maintainability.

Constraints and Assumptions

This section will outline any design constraints that have been imposed on the design of the system by the customer, thereby removing certain options from being considered by the developers. Also, this section will contain any assumptions that have been made by the requirements engineering team when gathering and analyzing the requirements. If any of the assumptions are found to be false, the system requirements specification would need to be re-evaluated to make sure that the documented requirements are still valid.

Acceptance Criteria

This section will describe the criteria by which the customer will "sign-off" on the final system. Depending on the methodology, this may happen at the end of the testing and quality assurance phase, or in an agile methodology, at the end of each iteration.

The criteria will usually refer to the need to complete all user acceptance tests and the rectification of all defects/bugs that meet a pre-determined priority or severity threshold.

Hardware Requirements

Processor	1.4 GHz 64-bit processor
Disk Space	100GB Free Space
RAM	Minimum 8GB
Graphics Device	Super VGA (1024 x 768) or higher-resolution

Software Requirements

Python
Anaconda
Jupyter Notebook
TensorFlow Package
Plotly Package
Matplotlib Package

Python Language

Python is a free, open-source programming language. Therefore, all you have to do is install Python once, and you can start working with it. Not to mention that you can contribute your own code to the community. Python is also a cross-platform compatible language. So, what does this mean? Well, you can install and run Python on several operating systems. Whether you have a Windows, Mac or Linux, you can rest assure that Python will work on all these operating systems.

Python is also a great visualization tool. It provides libraries such as Matplotlib, seaborn and bokeh to create stunning visualizations.

Python coding style comprises physical lines as well as logical lines or statements. A physical line in a Python program is a sequence of characters, and the end of the line terminates the line sequence as opposed to some other languages, such as C and C++ where a semicolon is used to mark the end of the statement. A logical line, on the other hand, is composed of one or more physical lines. The use of a semi-colon is not prohibited in Python, although itÃ¢â‚¬â„¢s not mandatory. The NEWLINE token denotes the end of the logical line. A logical line that only contains spaces, comments, or tabs are called blank lines and they are ignored by the interpreter.

As we saw that in Python, a new line simply means that a new statement has started. Although, Python does provide a way to split a statement into a multiline statement or to join multiple statements into one logical line. This can be helpful to increase the readability of the statement. Following are the two ways to split a line into two or more lines:

Explicit Line Joining

In explicit line joining, we use a backward slash to split a statement into a multiline statement.

Implicit Line Joining

Statements that reside inside [], {}, or () parentheses can be broken down into two or more physical lines without using a back slash.

Multiple Statements on a Single Line

In Python, it is possible to club multiple statements in the same line using a semi-colon; however, most programmers do not consider this to be a good practice as it reduces the readability of the code.

Whitespaces and Indentation

Unlike most of the programming languages, Python uses indentation to mark a block of code. According to Python coding style guideline or PEP8, we should keep an indent size of four.

Most of the programming languages provide indentation for better code formatting and do not enforce to have it. But in Python it is mandatory. This is why indentation is so crucial in Python.

Comments in any programming language are used to increase the readability of the code. Similarly, in Python, when the program starts getting complicated, one of the best ways to maintain the readability of the code is to use Python comments. It is considered a good practice to include documentations and notes in the python syntax since it makes the code way more readable and understandable to other programmers as well, which comes in handy when multiple programmers are simultaneously working on the same project.

Following are different kinds of comments that can be included in our Python program:

Single Line Comments

Single line Python comments are marked with # character. These comments end at the end of the physical line, which means that all characters starting after the # character (and lasts till the end of the line) are part of the comment.

Docstring Comments

Python has the documentation strings (or docstrings) feature which is usually the first statement included in functions and modules.

Rather than being ignored by the Python Interpreter like regular comments, docstrings can actually be accessed at the run time using the dot operator.

It gives programmers an easy way of adding quick notes with every Python module, function, class, and method. To use this feature, we use triple quotes in the beginning of the documentation string or comment and the closing triple quotes at the end of the documentation comment. Docstrings can be one-liners as well as multi-liners.

Multiline Comments

Unlike some programming languages that support multiline comments, such as C, Java, and more, there is no specific feature for multiline comments in Python. But that does not mean that it is totally impossible to make multiline comments in Python. There are two ways we can include comments that can span across multiple lines in our Python code.

Python Block Comments: We can use several single line comments for a whole block. This type of comment is usually created to explain the block of code that follows the Block comment. Python Block comment is the only way of writing a real comment that can span across multiple lines. It is supported and preferred by Pythons PEP8 style guide since Block comments are ignored by Python interpreter or parser.

Data Types

One of the most crucial part of learning any programming language is to understand how data is stored and manipulated in that language. Users are often inclined toward Python because of its ease of use and the number of versatile features it provides. One of those features is dynamic typing.

In Python, unlike statically typed languages like C or Java, there is no need to specifically declare the data type of the variable. In dynamically typed languages such as Python, the interpreter itself predicts the data type of the Python Variable based on the type of value assigned to that variable.

Advantages of Python

Universal Language Construct

Support both High Level and Low Level Programming

Language Interoperability

Fastest Development life cycle therefore more productive coding environmentLess memory used because a single container hold

Multiple data types and each type doesnÃ¢â‚¬â„¢t require its own function

Learning Ease and open source development

Speed and user-friendly data structure

Extensive and extensible libraries.

Simple & support IoT

and many more

Anaconda Software

Anaconda is the data science platform for data scientists, IT professionals and business leaders of tomorrow. It is a distribution of Python, R, etc. With more than 300 packages for data science, it becomes one of the best platforms for any project.

Anaconda helps in simplified package management and deployment. Anaconda comes with a wide variety of tools to easily collect data from various sources using various machine learning and AI algorithms. It helps in getting an easily manageable environment setup which can deploy any project with the click of a single button.

Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms.

Anaconda Navigator is a desktop GUI that ships with Anaconda and lets you launch applications and manage conda packages, environments, and channels without having to use a command-line interface. It can search for packages in a local Anaconda repository or on Anaconda Cloud. With Navigator, you donâ€™t need to type commands in a terminal, it lets you work with packages and environments with just a click.

Jupyter Notebook

JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.

A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media. In other words: itâ€™s a single document where you can run code, display the output, and also add explanations, formulas, charts, and make your work more transparent, understandable, repeatable, and shareable.

Using Notebooks is now a major part of the data science workflow at companies across the globe. If your goal is to work with data, using a Notebook will speed up your workflow and make it easier to communicate and share your results.

As a server-client application, the Jupyter Notebook App allows you to edit and run your notebooks via a web browser. The application can be executed on a PC without Internet access, or it can be installed on a remote server, where you can access it through the Internet.

Its two main components are the kernels and a dashboard.

A kernel is a program that runs and introspects the users code. The Jupyter Notebook App has a kernel for Python code, but there are also kernels available for other programming languages.

The dashboard of the application not only shows you the notebook documents that you have made and can reopen but can also be used to manage the kernels: you can which ones are running and shut them down if necessary.

Jupyter Notebook Features

Pluggable authentication

Manage users and authentication with PAM, OAuth or integrate with your own directory service system.

Centralized deployment

Deploy the Jupyter Notebook to thousands of users in your organization on centralized infrastructure on- or off-site.

Container friendly

Use Docker and Kubernetes to scale your deployment, isolate user processes, and simplify software installation.

Live coding environments

Code can be changed and run in real-time with feedback provided directly in the browser

Code meets data

Deploy the Notebook next to your data to provide unified software management and data access within your organization.

TensorFlow

Deep learning is a subfield of machine learning that is a set of algorithms that is inspired by the structure and function of the brain. Deep learning is a subset of machine learning. There are certain specialties in which we perform machine learning, and that's why it is called deep learning. For example, deep learning uses neural networks, which are like a simulation of the human brain. Deep learning also involves analyzing large amounts of unstructured data, unlike traditional machine learning, which typically uses structured data. This unstructured data could be fed in the form of images, video, audio, text, etc.

TensorFlow is the second machine learning framework that Google created and used to design, build, and train deep learning models. You can use the TensorFlow library do to numerical computations, which in itself does not seem all too special, but these computations are done with data flow graphs. In these graphs, nodes represent mathematical operations, while the edges represent the data, which usually are multidimensional data arrays or tensors, that are communicated between these edges.

The name TensorFlow is derived from the operations which neural networks perform on multidimensional data arrays or tensors! Its literally a flow of tensors.

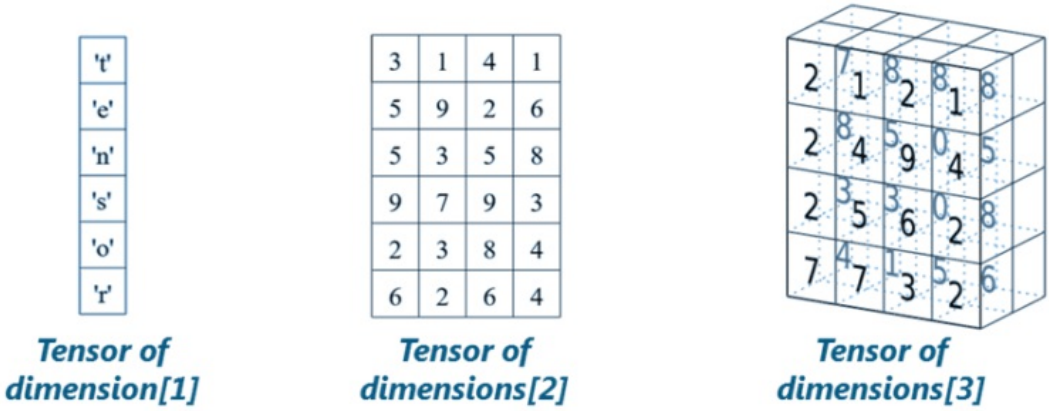
Using tf.keras allows you to design, fit, evaluate, and use deep learning models to make predictions in just a few lines of code. It makes common deep learning tasks, such as classification and regression predictive modeling

The other important aspect is TensorFlow is highly scalable. You can write your code and then make it run either on CPU, GPU, or across a cluster of these systems for the training purpose.

Generally, training the model is where a large part of the computation goes. Also, the process of training is repeated multiple times to solve any issues that may arise. This process leads to the consumption of more power, and therefore, you need a distributed computing. If you need to process large amounts of data, TensorFlow makes it easy by running the code in a distributed manner.

GPUs, or graphical processing units, have become very popular. Nvidia is one of the leaders in this space. It is good at performing mathematical computations, such as matrix multiplication, and plays a significant role in deep learning. TensorFlow also has integration with C++ and Python API, making development much faster.

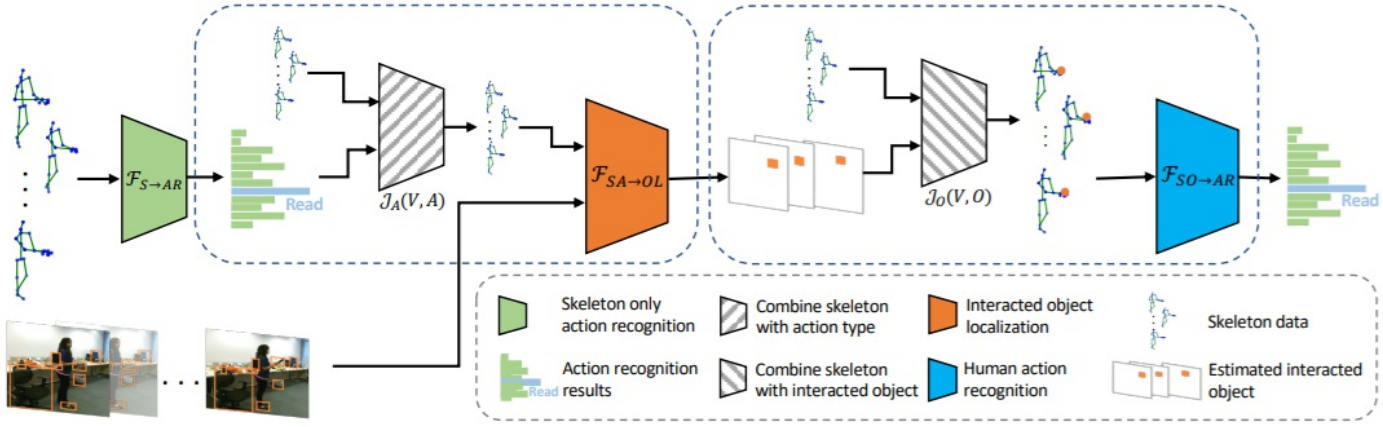
A tensor is a mathematical object represented as arrays of higher dimensions. These arrays of data with different sizes and ranks get fed as input to the neural network. These are the tensors.



You can have arrays or vectors, which are one-dimensional, or matrices, which are two-dimensional. But tensors can be more than three, four or five-dimensional. Therefore, it helps in keeping the data very tight in one place and then performing all the analysis around that.

Chapter 5 - System Design

Architecture



Algorithm Implemented

Convolution Neural Network

Advantages of Implemented Algorithm

Converts the independent activations into dependent activations
It learns every data over time
To extend the effective pixel neighborhood

Chapter 6 - System Implementation

Module 1 : Data Processing

Processing of the video signal prior to MPEG2 encoding can provide dramatic quality improvements in the subjective quality of the reconstructed video or bit-rate reductions in the generation of the compressed bit-stream. In a broadcast environment the video source is not guaranteed to be noise free. Noise in video signal will not only reduce video quality but also cost extra bits to code. Instead of using expensive external video pre-processors, state of the art broadcast quality real-time video encoders incorporate onboard video pre-processing to remove noise. An estimate of the noise level is essential for the control of subsequent filtering. Optimum performance is achieved when there exists a close coupling between the video preprocessor and the video encoder.

Module 2 : Global and Local Context Distillation

Knowledge distillation can transfer the learning ability of a large network to a small network. The condition for distillation is that the partial structure of the student network corresponds functionally to that of the teacher network. However, the manifold networks are structurally incompatible with the convolutional neural networks, leading to the difficulty of distillation between them. The output layers of the two networks are both fully connected layers, hence we choose to distill the knowledge on the top of networks.

In order to have an informative representation of human-object interaction HOIs, considering global and context relations between human and objects, features from both contextual views should be fully utilized. This may not be simply done by combining features from the two contexts, despite it is a standard way for gathering information from different sources or views. In contrast, we adapt a teacher-student framework to utilize global and local context of HOIs through knowledge distillation. To implement such a knowledge transfer, we incorporate soft labels from the teacher context graph network to guide the student context graph network during training, where these soft targets are probability distributions from the logits in the teacher network.

Module 3 : Training

We first train teacher network, which captures one view of context (e.g., global context) of HOIs along with hard labels, using cross-entropy loss. We then fix the teacher network and train the student network which is another view of HOIs (e.g., local context).

We adopt a simple yet effective model, to integrate the re-weighted object information with the skeleton data for action classification. We extend the node feature from only the coordinate information to the combined representation of human joint coordinate and its interacted object. From the skeleton flow to skeleton+object flow, it carries more semantic information and stronger expression ability.

By integrating interacted object with the skeleton data, we extend the pose flow to the pose+object flow. The semantics of human joints interacting with the objects could be better explored.

Chapter 7 - Software Testing

Testing documentation is the documentation of artifacts that are created during or before the testing of a software application. Documentation reflects the importance of processes for the customer, individual and organization. Projects which contain all documents have a high level of maturity. Careful documentation can save the time, efforts and wealth of the organization.

If the testing or development team gets software that is not working correctly and developed by someone else, so to find the error, the team will first need a document. Now, if the documents are available then the team will quickly find out the cause of the error by examining documentation. But, if the documents are not available then the tester need to do black box and white box testing again, which will waste the time and money of the organization. More than that, Lack of documentation becomes a problem for acceptance.

Benefits of using Documentation

Documentation clarifies the quality of methods and objectives.
It ensures internal coordination when a customer uses software application.
It ensures clarity about the stability of tasks and performance.
It provides feedback on preventive tasks.
It provides feedback for your planning cycle.
It creates objective evidence for the performance of the quality management system.

The test scenario is a detailed document of test cases that cover end to end functionality of a software application in liner statements. The liner statement is considered as a scenario. The test scenario is a high-level classification of testable requirements. These requirements are grouped on the basis of the functionality of a module and obtained from the use cases.

In the test scenario, there is a detailed testing process due to many associated test cases. Before performing the test scenario, the tester has to consider the test cases for each scenario.

In the test scenario, testers need to put themselves in the place of the user because they test the software application under the users point of view. Preparation of scenarios is the most critical part, and it is necessary to seek advice or help from customers, stakeholders or developers to prepare the scenario.

As per the IEEE Documentation describing plans for, or results of, the testing of a system or component, Types include test case specification, test incident report, test log, test plan, test procedure, test report. Hence the testing of all the above mentioned documents is known as documentation testing.

This is one of the most cost effective approaches to testing. If the documentation is not right: there will be major and costly problems. The documentation can be tested in a number of different ways to many different degrees of complexity. These range from running the documents through a spelling and grammar checking device, to manually reviewing the documentation to remove any ambiguity or inconsistency.

Documentation testing can start at the very beginning of the software process and hence save large amounts of money, since the earlier a defect is found the less it will cost to be fixed.

The most popular testing documentation files are test reports, plans, and checklists. These documents are used to outline the teams workload and keep track of the process. Lets take a look at the key requirements for these files and see how they contribute to the process.

Test strategy
An outline of the full approach to product testing. As the project moves along, developers, designers, product owners can come back to the document and see if the actual performance corresponds to the planned activities.

Test data
The data that testers enter into the software to verify certain features and their outputs. Examples of such data can be fake user profiles, statistics, media content, similar to files that would be uploaded by an end-user in a ready solution.

Test plans
A file that describes the strategy, resources, environment, limitations, and schedule of the testing process. Its the fullest testing document, essential for informed planning. Such a document is distributed between team members and shared with all stakeholders.

Test scenarios
In scenarios, testers break down the productÃ¢â‚¬â„¢s functionality and interface by modules and provide real-time status updates at all testing stages. A module can be described by a single statement, or require hundreds of statuses, depending on its size and scope.

Test cases
If the test scenario describes the object of testing (what), a scenario describes a procedure (how). These files cover step-by-step guidance, detailed conditions, and current inputs of a testing task. Test cases have their own kinds that depend on the type of testing, functional, UI, physical, logical cases, etc. Test cases compare available resources and current conditions with desired outcomes and determine if the functionality can be released or not.

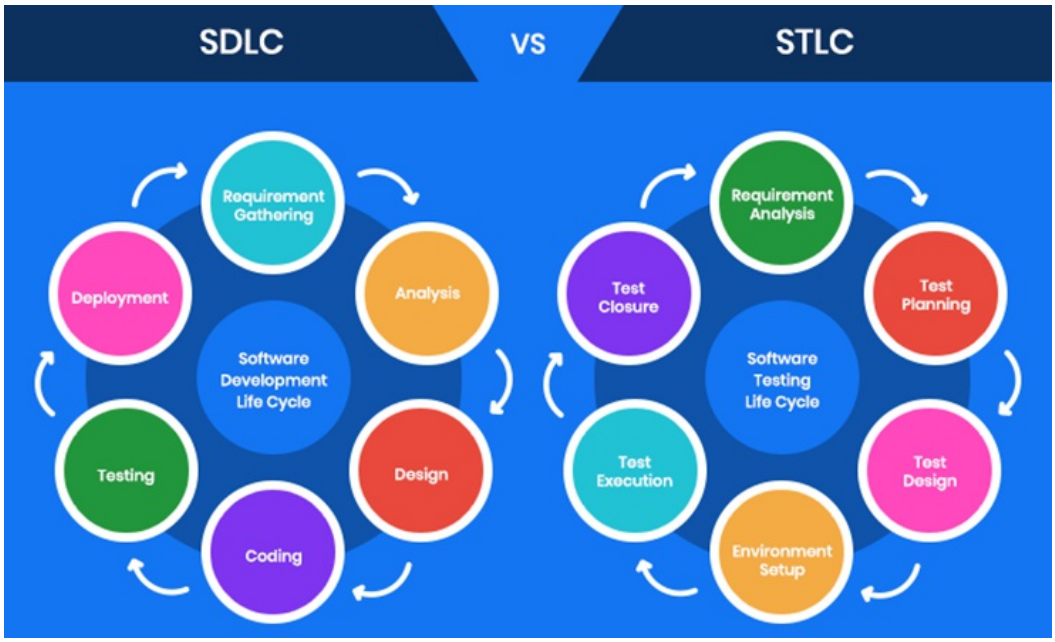
Traceability Matrix

This software testing documentation maps test cases and their requirements. All entries have their custom IDs Ã¢â‚¬â€ team members and stakeholders can track the progress of any tasks by simply entering its ID to the search.

The combination of internal and external documentation is the key to a deep understanding of all testing processes. Although stakeholders typically have access to the majority of documentation, they mostly work with external files, since they are more concise and tackle tangible issues and results. Internal files, on the other hand, are used by team members to optimize the testing process.

Unit Testing is not a new concept. It's been there since the early days of programming. Usually, developers and sometimes White box testers write Unit tests to improve code quality by verifying each and every unit of the code used to implement functional requirements (aka test drove development TDD or test-first development).

Software Testing Life Cycle (Taken from Google.com)



Introduction

Unit Testing frameworks are mostly used to help write unit tests quickly and easily. Most of the programming languages do not support unit testing with the inbuilt compiler. Third-party open source and commercial tools can be used to make unit testing even more fun.

List of popular Unit Testing tools for different programming languages:

- Java framework - JUnit
- PHP framework - PHPUnit
- C++ frameworks - UnitTest++ and Google C++
- .NET framework - NUnit
- Python framework - py.test

Software Testing Word Cloud (Taken from Google.com)

Functional Testing is a type of black box testing whereby each part of the system is tested against functional specification/requirements. For instance, seek answers to the following questions,

1. Are you able to login to a system after entering correct credentials?
2. Does your payment gateway prompt an error message when you enter incorrect card number?
3. Does your Add a customer screen adds a customer to your records successfully?

Test Driven Development

Test Driven Development, or TDD, is a code design technique where the programmer writes a test before any production code, and then writes the code that will make that test pass. The idea is that with a tiny bit of assurance from that initial test, the programmer can feel free to refactor and refactor some more to get the cleanest code they know how to write. The idea is simple, but like most simple things, the execution is hard. TDD requires a completely different mind set from what most people are used to and the tenacity to deal with a learning curve that may slow you down at first.

Functional Testing types include:

- Unit Testing
- Integration Testing
- System Testing
- Sanity Testing
- Smoke Testing
- Interface Testing
- Regression Testing
- Beta/Acceptance Testing

Non-functional Testing types include

- Load Testing
- Stress Testing
- Volume Testing
- Security Testing
- Compatibility Testing
- Install Testing
- Recovery Testing
- Reliability Testing
- Usability Testing
- Compliance Testing
- Localization Testing

Unit Testing

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

A unit can be almost anything you want it to be -- a line of code, a method, or a class. Generally though, smaller is better. Smaller tests give you a much more granular view of how your code is performing. There is also the practical aspect that when you test very small units, your tests can be run fast; like a thousand tests in a second fast.

Black Box testers don't care about Unit Testing. Their main goal is to validate the application against the requirements without going into the implementation details.

Unit Testing is not a new concept. It's been there since the early days of programming. Usually, developers and sometimes White box testers write Unit tests to improve code quality by verifying each and every unit of the code used to implement functional requirements (aka test drove development TDD or test-first development).

Most of us might know the classic definition of Unit Testing

Unit Testing is the method of verifying the smallest piece of testable code against its purpose

If the purpose or requirement failed then the unit test has failed. In simple words, Unit Testing means - writing a piece of code (unit test) to verify the code (unit) written for implementing requirements.

Blackbox Testing

During functional testing, testers verify the app features against the user specifications. This is completely different from testing done by developers which is unit testing. It checks whether the code works as expected. Because unit testing focuses on the internal structure of the code, it is called the white box testing. On the other hand, functional testing checks appÃ¢â‚¬â„¢s functionalities without looking at the internal structure of the code, hence it is called black box testing. Despite how flawless the various individual code components may be, it is essential to check that the app is functioning as expected, when all components are combined. Here you can find a detailed comparison between functional testing vs unit testing.

Integration Testing

INTEGRATION TESTING is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

Integration testing: Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. See also component integration testing, system integration testing.

Component integration testing Testing performed to expose defects in the interfaces and interaction between integrated components. System integration testing: Testing the integration of systems and packages; testing interfaces to external organizations (e.g. Electronic Data Interchange, Internet).

Integration tests determine if independently developed units of software work correctly when they are connected to each other. The term has become blurred even by the diffuse standards of the software industry, so I've been wary of using it in my writing. In particular, many people assume integration tests are necessarily broad in scope, while they can be more effectively done with a narrower scope.

As often with these things, it's best to start with a bit of history. When I first learned about integration testing, it was in the 1980's and the waterfall was the dominant influence of software development thinking. In a larger project, we would have a design phase that would specify the interface and behavior of the various modules in the system. Modules would then be assigned to developers to program. It was not unusual for one programmer to be responsible for a single module, but this

would be big enough that it could take months to build it. All this work was done in isolation, and when the programmer believed it was finished they would hand it over to QA for testing.

Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems. Integration testing is a key aspect of software testing.

System Testing

SYSTEM TESTING is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the systems compliance with the specified requirements. System Testing means testing the system as a whole. All the modules/components are integrated in order to verify if the system works as expected or not.

System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product. System testing is a method of monitoring and assessing the behaviour of the complete and fully-integrated software product or system, on the basis of pre-decided specifications and functional requirements. It is a solution to the question "whether the complete system functions in accordance to its pre-defined requirements?"

It's comes under black box testing i.e. only external working features of the software are evaluated during this testing. It does not requires any internal knowledge of the coding, programming, design, etc., and is completely based on users-perspective.

A black box testing type, system testing is the first testing technique that carries out the task of testing a software product as a whole. This System testing tests the integrated system and validates whether it meets the specified requirements of the client.

System testing is a process of testing the entire system that is fully functional, in order to ensure the system is bound to all the requirements provided by the client in the form of the functional specification or system specification documentation. In most cases, it is done next to the Integration testing, as this testing should be covering the end-to-end systems actual routine. This type of testing requires a dedicated Test Plan and other test documentation derived from the system specification document that should cover both software and hardware requirements. By this test, we uncover the errors. It ensures that all the system works as expected. We check System performance and functionality to get a quality product. System testing is nothing but testing the system as a whole. This testing checks complete end-to-end scenario as per the customerÃ¢â‚¬â„¢s point of view. Functional and Non-Functional tests also done by System testing. All things are done to maintain trust within the development that the system is defect-free and bug-free. System testing is also intended to test hardware/software requirements specifications. System testing is more of a limited type of testing;

Sanity Testing

Sanity Testing is done when as a QA we do not have sufficient time to run all the test cases, be it Functional Testing, UI, OS or Browser Testing. Sanity testing is a subset of regression testing. After receiving the software build, sanity testing is performed to ensure that the code changes introduced are working as expected. This testing is a checkpoint to determine if testing for the build can proceed or not. The main purpose of this testing is to determine that the changes or the proposed functionality are working as expected. If the sanity test fails, the build is rejected by the testing team to save time and money. It is performed only after the build has cleared the smoke test and been accepted by the Quality Assurance team for further testing. The focus of the team during this testing process is to validate the functionality of the application and not detailed testing.

Smoke Testing is done to make sure if the build we received from the development team is testable or not. It is also called as Ã¢â‚¬oeDay 0Ã¢â‚¬ check. It is done at the build level.

It helps not to waste the testing time to simply testing the whole application when the key features donÃ¢â‚¬â„¢t work or the key bugs have not been fixed yet. Here our focus will be on primary and core application work flow.

To conduct smoke testing, we do not write test cases. We just pick the necessary test cases from already written test cases. As mentioned earlier, here in Smoke Testing, our main focus will be on core application work flow. So we pick the test cases from our test suite which cover major functionality of the application. In general, we pick minimal number of test cases that wont take more than half an hour to execute.

The main aim of Sanity testing to check the planned functionality is working as expected. Instead of doing whole regression testing the Sanity testing is perform.

Sanity tests helps to avoid wasting time and cost involved in testing if the build is failed. Tester should reject the build upon build failure. After completion of regression testing the Sanity testing is started to check the defect fixes & changes done in the software application is not breaking the core functionality of the software. Typically this is done nearing end of SDLC i.e. while releasing the software. You can say that sanity testing is a subset of acceptance testing. We can also say Tester Acceptance Testing for Sanity testing.

Regression Testing

Regression Testing is a type of testing that is done to verify that a code change in the software does not impact the existing functionality of the product. This is to make sure the product works fine with new functionality, bug fixes or any change in the existing feature. Previously executed test cases are re-executed in order to verify the impact of change.

Regression Testing is a Software Testing type in which test cases are re-executed in order to check whether the previous functionality of the application is working fine and the new changes have not introduced any new bugs.

This test can be performed on a new build when there is a significant change in the original functionality that too even in a single bug fix. For regression testing to be effective, it needs to be seen as one part of a comprehensive testing methodology that is cost-effective and efficient while still incorporating enough varietyÃ¢â‚¬â€such as well-designed frontend UI automated tests alongside targeted unit testing, based on smart risk prioritizationÃ¢â‚¬â€to prevent any aspects of your software applications from going unchecked. These days, many Agile work environments employing workflow practices such as XP (Extreme Programming), RUP (Rational Unified Process), or Scrum appreciate regression testing as an essential aspect of a dynamic, iterative development and deployment schedule. But no matter what software development and quality-assurance process your organization uses, if you take the time to put in enough careful planning up front, crafting a clear and diverse testing strategy with automated regression testing at its core, you can help prevent projects from going over budget, keep your team on track, and, most importantly, prevent unexpected bugs from damaging your products and your companys bottom line.

Performance testing

Performance testing is the practice of evaluating how a system performs in terms of responsiveness and stability under a particular workload. Performance tests are typically executed to examine speed, robustness, reliability, and application size.

Performance Testing (Taken from Google.com)

Performance testing gathers all the tests that verify an applications speed, robustness, reliability, and correct sizing. It examines several indicators such as a browser, page and network response times, server query processing time, number of acceptable concurrent users architected, CPU memory consumption, and number/type of errors which may be encountered when using an application. Performance testing is the testing that is performed to ascertain how the components of a system are performing under a certain given situation. Resource usage, scalability, and reliability of the product are also validated under this testing. This testing is the subset of performance engineering, which is focused on addressing performance issues in the design and architecture of a software product.

Software Performance testing is type of testing perform to determine the performance of system to major the measure, validate or verify quality attributes of the system like responsiveness, Speed, Scalability, Stability under variety of load conditions. The system is tested under a mixture of load conditions and check the time required responding by the system under varying workloads. Software performance testing involves the testing of application under test to ensure that application is working as expected under variety of load conditions. The goal of performance testing is not only find the bugs in the system but also eliminate the performance bottlenecks from the system.

Load Testing is type of performance testing to check system with constantly increasing the load on the system until the time load is reaches to its threshold value. Here Increasing load means increasing number of concurrent users, transactions & check the behavior of application under test. It is normally carried out underneath controlled environment in order to distinguish between two different systems. It is also called as Ã¢â‚¬oeEndurance testingÃ¢â‚¬ and Ã¢â‚¬oeVolume testingÃ¢â‚¬. The main purpose of load testing is to monitor the response time and staying power of application when system is performing well under heavy load. Load testing comes under the Non Functional Testing & it is designed to test the non-functional requirements of a software application.

Load testing is perform to make sure that what amount of load can be withstand the application under test. The successfully executed load testing is only if the specified test cases are executed without any error in allocated time.v

Testing printer by sending large job. Editing a very large document for testing of word processor Continuously reading and writing data into hard disk. Running multiple applications simultaneously on server. Testing of mail server by accessing thousands of mailboxes In case of zero-volume testing & system fed with zero load.

Chapter 8 - Conclusion

In this paper, we proposed a human activity recognition system based on smart phone sensor recordings. First, the neighborhood component analysis-based feature selection is utilized to select suitable features from many available time and frequency domain hand crafted features. Next, these automatically selected features are fed into a dense neural network model to classify different human activities. When compared with existing methods, the proposed model achieved comparable accuracy with less features, thus, showing the efficacy of our approach. More importantly, less number of features are directly related to reduced computation time. We developed a 3D CNN model for action recognition in this paper. This model construct features from both spatial and temporal dimensions by performing 3D convolutions. The developed deep architecture generates multiple channels of information from adjacent input frames and perform convolution and subsampling separately in each channel. The final feature representation is computed by combining information from all channels

Chapter 9 - Future Work

Future works include exploring other feature selection methods as well as incorporating different databases.

Chapter 10 - Appendix - I Screenshots

J. Carreira and A. Zisserman, Quo vadis, action recognition? A new model and the kinetics dataset, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 47244733.

J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot, Global context-aware attention LSTM networks for 3D action recognition, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 16471656.

S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, An end-to-end spatio-temporal attention model for human action recognition from skele- ton data, in Proc. AAAI, 2017, pp. 42634270.

S. Sharma, R. Kiros, and R. Salakhutdinov, Action recognition using visual attention, Nov. 2015, arXiv:1511.04119. [Online]. Available: https://arxiv.org/abs/1511.04119

H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, HMDB: A large video database for human motion recognition, in Proc. Int. Conf. Comput. Vis., Nov. 2011, pp. 25562563.

F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, Residual attention network for image classication, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 31563164.

B. F. Skinner, The behavior of organisms: An experimental analysis.

C. Xu, S.-H. Hsieh, C. Xiong, and J. J. Corso, "Can humans y? action understanding with multiple classes of actors," in Computer Vision and Pattern Recognition. IEEE, 2015, pp. 22642273.

H. Wang and C. Schmid, "Action recognition with improved trajecto- IEEE, 2013, pp. 35513558.

H. Wang and L. Wang, "Modeling temporal dynamics and spatial congurations of actions using two-stream recurrent neural networks," in Computer Vision and Pattern Recognition. IEEE, 2017.

H. Wang, W. Wang, and L. Wang, "Hierarchical motion evolution for action recognition," in Asian Conference on Pattern Recognition. IEEE, 2015.

L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: towards good practices for deep action recognition," in European Conference on Computer Vision. Springer, 2016, pp. 2036.

S. Zhao, Y. Liu, Y. Han, R. Hong, Q. Hu, and Q. Tian, "Pooling the convolutional layers in deep convnets for video action recognition," Transactions on Circuits and Systems for Video Technology, 2017.

C. Ladha, N. Hammerla, E. Hughes, P. Olivier, and T. Ploetz, "Dogs life: wearable activity recognition for dogs," in International Joint Conference on Pervasive and Ubiquitous Computing. ACM, 2013, pp. 415418.

S. J. Pan and Q. Yang, "A survey on transfer learning," Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 13451359, 2010.

J. Blitzer, S. Kakade, and D. Foster, "Domain adaptation with coupled Statistics, 2011, pp. 173181.

L. Dong, N. Feng, P. Quan, G. Kong, X. Chen, and Q. Zhang, "Optimal kernel choice for domain adaption learning," Engineering Applications of Articial Intelligence, vol. 51, pp. 163170, 2016.

V. W. Zheng, D. H. Hu, and Q. Yang, "Cross-domain activity recogni- computing. ACM, 2009, pp. 6170. Computer Vision and Pattern Recognition. [30] L. Cao, Z. Liu, and T. S. Huang, "Cross-dataset action detection," in IEEE, 2010, pp. 19982005.

J. Liu, M. Shah, B. Kuipers, and S. Savarese, "Cross-view action recognition via view knowledge transfer," in Computer Vision and Pattern Recognition. IEEE, 2011, pp. 32093216.

M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu, "Transfer sparse coding for robust image representation," in Computer Vision and Pattern Recognition. IEEE, 2013, pp. 407414.

M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, "Graph regularized sparse coding for image representation," Transac- tions on Image Processing, vol. 20, no. 5, pp. 13271336, 2011.

J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple on Knowledge Discovery and Data Mining. ACM, 2008, pp. 283291.

J. Zheng, Z. Jiang, and R. Chellappa, "Cross-view action recognition via transferable dictionary learning," IEEE Trans. Image Process., vol. 25, no. 6, pp. 25422556, Jun. 2016.

Y.-G. Jiang, Q. Dai, W. Liu, X. Xue, and C.-W. Ngo, "Human action recognition in unconstrained videos by explicit motion modeling," IEEE Trans. Image Process., vol. 24, no. 11, pp. 37813795, Nov. 2015.

G. Zhang, J. Liu, H. Li, Y. Q. Chen, and L. S. Davis, "Joint human detection and head pose estimation via multistream networks for RGB-D videos," IEEE Signal Process. Lett., vol. 24, no. 11, pp. 16661670, Nov. 2017.

L. L. Presti and M. La Cascia, "3D skeleton-based human action classi- cation: A survey," Pattern Recognit., vol. 53, pp. 130147, May 2016.

Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural net- work for skeleton based action recognition," in Proc. CVPR, 2015, pp. 11101118.

J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in Proc. NIPS, 2015, pp. 577585.

D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proc. ICLR, 2015, pp. 115.

M. Sundermeyer, R. Schlter, and H. Ney, "LSTM neural networks for language modeling," in Proc. INTERSPEECH, 2012, pp. 194197.

J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classication," in Proc. CVPR, 2015, pp. 46944702.

J. Donahue et al., "Long-term recurrent convolutional networks recognition and description," in Proc. CVPR, 2015, for visual pp. 26252634.

S. Ma, L. Sigal, and S. Sclaroff, "Learning activity progression in LSTMs for activity detection and early detection," in Proc. CVPR, 2016, pp. 19421950.

J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3d human action recognition," in Proc. ECCV, 2016, pp. 816833. [30] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in Proc. ICLR, 2015, pp. 115.

J. Luo, W. Wang, and H. Qi, "Group sparsity and geometry constrained dictionary learning for action recognition from depth maps," in Proc. ICCV, 2013, pp. 18091816.

M. Meng, H. Drira, M. Daoudi, and J. Boonaert, "Human-object interaction recognition by learning the distances between the object and the skeleton joints," in Proc. FG, 2015, pp. 16.

J. Wang and Y. Wu, "Learning maximum margin temporal warping for action recognition," in Proc. ICCV, 2013, pp. 26882695.

H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian, "Real time action recognition using histograms of depth gradients and random decision forests," in Proc. WACV, 2014, pp. 626633.

A. Shahroudy, G. Wang, and T.-T. Ng, "Multi-modal feature fusion for action recognition in RGB-D sequences," in Proc. ISCCSP, 2014, pp. 14. 1598

F. Oi, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition," J. Vis. Commun. Image Represent., vol. 25, no. 1, pp. 2438, 2014.

R. Chaudhry, F. Oi, G. Kurillo, R. Bajcsy, and R. Vidal, "Bio- inspired dynamic 3D discriminative skeletal features for human action recognition," in Proc. CVPR, 2013, pp. 471478.

L. Xia, C.-C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3D joints," in Proc. CVPR, 2012, pp. 2027.

P. Wang, C. Yuan, W. Hu, B. Li, and Y. Zhang, "Graph based skeleton motion representation and similarity measurement for action recogni- tion," in Proc. ECCV, 2016, pp. 370385.

V. Veeriah, N. Zhuang, and G.-J. Qi, "Differential recurrent neural networks for action recognition," in Proc. ICCV, 2015, pp. 40414049.

J. Liu, A. Shahroudy, D. Xu, A. C. Kot, and G. Wang, "Skeleton- based action recognition using spatio-temporal LSTM network with trust gates," IEEE Trans. Pattern Anal. Mach. Intell., to be published, doi: 10.1109/TPAMI.2017.2771306.

Y. Li, C. Lan, J. Xing, W. Zeng, C. Yuan, and J. Liu, "Online human action detection using joint classication-regression recurrent neural networks," in Proc. ECCV, 2016, pp. 203220.

A. Kumar et al., "Ask me anything: Dynamic memory networks for natural language processing," in Proc. ICML, 2016, pp. 13781387.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in Proc. NIPS, 2015, pp.

24402448.

L. Yao et al., "Describing videos by exploiting temporal structure," in Proc. ICCV, 2015, pp. 45074515.

Y. Wang, S. Wang, J. Tang, N. OHare, Y. Chang, and B. Li. (2016). "Hierarchical attention network for action recognition in videos." [Online]. Available: https://arxiv.org/abs/1607.06416 .

K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras, "Two-person interaction detection using body-pose features and multiple instance learning," in Proc. CVPRW, 2012, pp. 2835.

G. Evangelidis, G. Singh, and R. Horaud, "Skeletal quads: Human ICPR, 2014, action recognition using joint quadruples," in Proc. pp. 45134518.

Chapter 11 - Appendix - II Sample Coding

J. Carreira and A. Zisserman, Quo vadis, action recognition? A new model and the kinetics dataset, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 47244733.

J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot, Global context-aware attention LSTM networks for 3D action recognition, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 16471656.

S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, An end-to-end spatio-temporal attention model for human action recognition from skele- ton data, in Proc. AAAI, 2017, pp. 42634270.

S. Sharma, R. Kiros, and R. Salakhutdinov, Action recognition using visual attention, Nov. 2015, arXiv:1511.04119. [Online]. Available: https://arxiv.org/abs/1511.04119

H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, HMDB: A large video database for human motion recognition, in Proc. Int. Conf. Comput. Vis., Nov. 2011, pp. 25562563.

F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, Residual attention network for image classication, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 31563164.

B. F. Skinner, The behavior of organisms: An experimental analysis.

C. Xu, S.-H. Hsieh, C. Xiong, and J. J. Corso, "Can humans y? action understanding with multiple classes of actors," in Computer Vision and Pattern Recognition. IEEE, 2015, pp. 22642273.

H. Wang and C. Schmid, "Action recognition with improved trajecto- IEEE, 2013, pp. 35513558.

H. Wang and L. Wang, "Modeling temporal dynamics and spatial congurations of actions using two-stream recurrent neural networks," in Computer Vision and Pattern Recognition. IEEE, 2017.

H. Wang, W. Wang, and L. Wang, "Hierarchical motion evolution for action recognition," in Asian Conference on Pattern Recognition. IEEE, 2015.

L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: towards good practices for deep action recognition," in European Conference on Computer Vision. Springer, 2016, pp. 2036.

S. Zhao, Y. Liu, Y. Han, R. Hong, Q. Hu, and Q. Tian, "Pooling the convolutional layers in deep convnets for video action recognition," Transactions on Circuits and Systems for Video Technology, 2017.

C. Ladha, N. Hammerla, E. Hughes, P. Olivier, and T. Ploetz, "Dogs life: wearable activity recognition for dogs," in International Joint Conference on Pervasive and Ubiquitous Computing. ACM, 2013, pp. 415418.

S. J. Pan and Q. Yang, "A survey on transfer learning," Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 13451359, 2010.

J. Blitzer, S. Kakade, and D. Foster, "Domain adaptation with coupled Statistics, 2011, pp. 173181.

L. Dong, N. Feng, P. Quan, G. Kong, X. Chen, and Q. Zhang, "Optimal kernel choice for domain adaption learning," Engineering Applications of Articial Intelligence, vol. 51, pp. 163170, 2016.

V. W. Zheng, D. H. Hu, and Q. Yang, "Cross-domain activity recogni- computing. ACM, 2009, pp. 6170. Computer Vision and Pattern Recognition. [30] L. Cao, Z. Liu, and T. S. Huang, "Cross-dataset action detection," in IEEE, 2010, pp. 19982005.

J. Liu, M. Shah, B. Kuipers, and S. Savarese, "Cross-view action recognition via view knowledge transfer," in Computer Vision and Pattern Recognition. IEEE, 2011, pp. 32093216.

M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu, "Transfer sparse coding for robust image representation," in Computer Vision and Pattern Recognition. IEEE, 2013, pp. 407414.

M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, "Graph regularized sparse coding for image representation," Transac- tions on Image Processing, vol. 20, no. 5, pp. 13271336, 2011.

J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple on Knowledge Discovery and Data Mining. ACM, 2008, pp. 283291.

J. Zheng, Z. Jiang, and R. Chellappa, "Cross-view action recognition via transferable dictionary learning," IEEE Trans. Image Process., vol. 25, no. 6, pp. 25422556, Jun. 2016.

Y.-G. Jiang, Q. Dai, W. Liu, X. Xue, and C.-W. Ngo, "Human action recognition in unconstrained videos by explicit motion modeling," IEEE Trans. Image Process., vol. 24, no. 11, pp. 37813795, Nov. 2015.

G. Zhang, J. Liu, H. Li, Y. Q. Chen, and L. S. Davis, "Joint human detection and head pose estimation via multistream networks for RGB-D videos," IEEE Signal Process. Lett., vol. 24, no. 11, pp. 16661670, Nov. 2017.

L. L. Presti and M. La Cascia, "3D skeleton-based human action classi- cation: A survey," Pattern Recognit., vol. 53, pp. 130147, May 2016.

Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural net- work for skeleton based action recognition," in Proc. CVPR, 2015, pp. 11101118.

J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in Proc. NIPS, 2015, pp. 577585.

D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proc. ICLR, 2015, pp. 115.

M. Sundermeyer, R. Schlter, and H. Ney, "LSTM neural networks for language modeling," in Proc. INTERSPEECH, 2012, pp. 194197.

J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classication," in Proc. CVPR, 2015, pp. 46944702.

J. Donahue et al., "Long-term recurrent convolutional networks recognition and description," in Proc. CVPR, 2015, for visual pp. 26252634.

S. Ma, L. Sigal, and S. Sclaroff, "Learning activity progression in LSTMs for activity detection and early detection," in Proc. CVPR, 2016, pp. 19421950.

J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3d human action recognition," in Proc. ECCV, 2016, pp. 816833. [30] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in Proc. ICLR, 2015, pp. 115.

J. Luo, W. Wang, and H. Qi, "Group sparsity and geometry constrained dictionary learning for action recognition from depth maps," in Proc. ICCV, 2013, pp. 18091816.

M. Meng, H. Drira, M. Daoudi, and J. Boonaert, "Human-object interaction recognition by learning the distances between the object and the skeleton joints," in Proc. FG, 2015, pp. 16.

J. Wang and Y. Wu, "Learning maximum margin temporal warping for action recognition," in Proc. ICCV, 2013, pp. 26882695.

H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian, "Real time action recognition using histograms of depth gradients and random decision forests," in Proc. WACV, 2014, pp. 626633.

A. Shahroudy, G. Wang, and T.-T. Ng, "Multi-modal feature fusion for action recognition in RGB-D sequences," in Proc. ISCCSP, 2014, pp. 14. 1598

F. Oi, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition," J. Vis. Commun. Image Represent., vol. 25, no. 1, pp. 2438, 2014.

R. Chaudhry, F. Oi, G. Kurillo, R. Bajcsy, and R. Vidal, "Bio- inspired dynamic 3D discriminative skeletal features for human action recognition," in Proc. CVPR, 2013, pp. 471478.

L. Xia, C.-C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3D joints," in

Proc. CVPR, 2012, pp. 2027.

P. Wang, C. Yuan, W. Hu, B. Li, and Y. Zhang, "Graph based skeleton motion representation and similarity measurement for action recogni- tion," in Proc. ECCV, 2016, pp. 370385.

V. Veeriah, N. Zhuang, and G.-J. Qi, "Differential recurrent neural networks for action recognition," in Proc. ICCV, 2015, pp. 40414049.

J. Liu, A. Shahroudy, D. Xu, A. C. Kot, and G. Wang, "Skeleton- based action recognition using spatio-temporal LSTM network with trust gates," IEEE Trans. Pattern Anal. Mach. Intell., to be published, doi: 10.1109/TPAMI.2017.2771306.

Y. Li, C. Lan, J. Xing, W. Zeng, C. Yuan, and J. Liu, "Online human action detection using joint classication-regression recurrent neural networks," in Proc. ECCV, 2016, pp. 203220.

A. Kumar et al., "Ask me anything: Dynamic memory networks for natural language processing," in Proc. ICML, 2016, pp. 13781387.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in Proc. NIPS, 2015, pp. 24402448.

L. Yao et al., "Describing videos by exploiting temporal structure," in Proc. ICCV, 2015, pp. 45074515.

Y. Wang, S. Wang, J. Tang, N. OHare, Y. Chang, and B. Li. (2016). "Hierarchical attention network for action recognition in videos." [Online]. Available: https://arxiv.org/abs/1607.06416 .

K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras, "Two-person interaction detection using body-pose features and multiple instance learning," in Proc. CVPRW, 2012, pp. 2835.

G. Evangelidis, G. Singh, and R. Horaud, "Skeletal quads: Human ICPR, 2014, action recognition using joint quadruples," in Proc. pp. 45134518.

Chapter 12 - References

J. Carreira and A. Zisserman, Quo vadis, action recognition? A new model and the kinetics dataset, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 47244733.

J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot, Global context-aware attention LSTM networks for 3D action recognition, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 16471656.

S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, An end-to-end spatio-temporal attention model for human action recognition from skele- ton data, in Proc. AAAI, 2017, pp. 42634270.

S. Sharma, R. Kiros, and R. Salakhutdinov, Action recognition using visual attention, Nov. 2015, arXiv:1511.04119. [Online]. Available: https://arxiv.org/abs/1511.04119

H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, HMDB: A large video database for human motion recognition, in Proc. Int. Conf. Comput. Vis., Nov. 2011, pp. 25562563.

F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, Residual attention network for image classication, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 31563164.

B. F. Skinner, The behavior of organisms: An experimental analysis.

C. Xu, S.-H. Hsieh, C. Xiong, and J. J. Corso, "Can humans y? action understanding with multiple classes of actors," in Computer Vision and Pattern Recognition. IEEE, 2015, pp. 22642273.

H. Wang and C. Schmid, "Action recognition with improved trajecto- IEEE, 2013, pp. 35513558.

H. Wang and L. Wang, "Modeling temporal dynamics and spatial congurations of actions using two-stream recurrent neural networks," in Computer Vision and Pattern Recognition. IEEE, 2017.

H. Wang, W. Wang, and L. Wang, "Hierarchical motion evolution for action recognition," in Asian Conference on Pattern Recognition. IEEE, 2015.

L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: towards good practices for deep action recognition," in European Conference on Computer Vision. Springer, 2016, pp. 2036.

S. Zhao, Y. Liu, Y. Han, R. Hong, Q. Hu, and Q. Tian, "Pooling the convolutional layers in deep convnets for video action recognition," Transactions on Circuits and Systems for Video Technology, 2017.

C. Ladha, N. Hammerla, E. Hughes, P. Olivier, and T. Ploetz, "Dogs life: wearable activity recognition for dogs," in International Joint Conference on Pervasive and Ubiquitous Computing. ACM, 2013, pp. 415418.

S. J. Pan and Q. Yang, "A survey on transfer learning," Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 13451359, 2010.

J. Blitzer, S. Kakade, and D. Foster, "Domain adaptation with coupled Statistics, 2011, pp. 173181.

L. Dong, N. Feng, P. Quan, G. Kong, X. Chen, and Q. Zhang, "Optimal kernel choice for domain adaption learning," Engineering Applications of Articial Intelligence, vol. 51, pp. 163170, 2016.

V. W. Zheng, D. H. Hu, and Q. Yang, "Cross-domain activity recogni- computing. ACM, 2009, pp. 6170. Computer Vision and Pattern Recognition. [30] L. Cao, Z. Liu, and T. S. Huang, "Cross-dataset action detection," in IEEE, 2010, pp. 19982005.

J. Liu, M. Shah, B. Kuipers, and S. Savarese, "Cross-view action recognition via view knowledge transfer," in Computer Vision and Pattern Recognition. IEEE, 2011, pp. 32093216.

M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu, "Transfer sparse coding for robust image representation," in Computer Vision and Pattern Recognition. IEEE, 2013, pp. 407414.

M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, "Graph regularized sparse coding for image representation," Transac- tions on Image Processing, vol. 20, no. 5, pp. 13271336, 2011.

J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple on Knowledge Discovery and Data Mining. ACM, 2008, pp. 283291.

J. Zheng, Z. Jiang, and R. Chellappa, "Cross-view action recognition via transferable dictionary learning," IEEE Trans. Image Process., vol. 25, no. 6, pp. 25422556, Jun. 2016.

Y.-G. Jiang, Q. Dai, W. Liu, X. Xue, and C.-W. Ngo, "Human action recognition in unconstrained videos by explicit motion modeling," IEEE Trans. Image Process., vol. 24, no. 11, pp. 37813795, Nov. 2015.

G. Zhang, J. Liu, H. Li, Y. Q. Chen, and L. S. Davis, "Joint human detection and head pose estimation via multistream networks for RGB-D videos," IEEE Signal Process. Lett., vol. 24, no. 11, pp. 16661670, Nov. 2017.

L. L. Presti and M. La Cascia, "3D skeleton-based human action classi- cation: A survey," Pattern Recognit., vol. 53, pp. 130147, May 2016.

Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural net- work for skeleton based action recognition," in Proc. CVPR, 2015, pp. 11101118.

J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in Proc. NIPS, 2015, pp. 577585.

D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proc. ICLR, 2015, pp. 115.

M. Sundermeyer, R. Schlter, and H. Ney, "LSTM neural networks for language modeling," in Proc. INTERSPEECH, 2012, pp. 194197.

J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classication," in Proc. CVPR, 2015, pp. 46944702.

J. Donahue et al., "Long-term recurrent convolutional networks recognition and description," in Proc. CVPR, 2015, for visual pp. 26252634.

S. Ma, L. Sigal, and S. Sclaroff, "Learning activity progression in LSTMs for activity detection and early detection," in Proc. CVPR, 2016, pp. 19421950.

J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3d human action recognition," in Proc. ECCV, 2016, pp. 816833. [30] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in Proc. ICLR, 2015, pp. 115.

J. Luo, W. Wang, and H. Qi, "Group sparsity and geometry constrained dictionary learning for action recognition from depth maps," in Proc. ICCV, 2013, pp. 18091816.

M. Meng, H. Drira, M. Daoudi, and J. Boonaert, "Human-object interaction recognition by learning the distances

between the object and the skeleton joints," in Proc. FG, 2015, pp. 16.

J. Wang and Y. Wu, "Learning maximum margin temporal warping for action recognition," in Proc. ICCV, 2013, pp. 26882695.

H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian, "Real time action recognition using histograms of depth gradients and random decision forests," in Proc. WACV, 2014, pp. 626633.

A. Shahroudy, G. Wang, and T.-T. Ng, "Multi-modal feature fusion for action recognition in RGB-D sequences," in Proc. ISCCSP, 2014, pp. 14. 1598

F. Oi, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition," J. Vis. Commun. Image Represent., vol. 25, no. 1, pp. 2438, 2014.

R. Chaudhry, F. Oi, G. Kurillo, R. Bajcsy, and R. Vidal, "Bio- inspired dynamic 3D discriminative skeletal features for human action recognition," in Proc. CVPR, 2013, pp. 471478.

L. Xia, C.-C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3D joints," in Proc. CVPR, 2012, pp. 2027.

P. Wang, C. Yuan, W. Hu, B. Li, and Y. Zhang, "Graph based skeleton motion representation and similarity measurement for action recogni- tion," in Proc. ECCV, 2016, pp. 370385.

V. Veeriah, N. Zhuang, and G.-J. Qi, "Differential recurrent neural networks for action recognition," in Proc. ICCV, 2015, pp. 40414049.

J. Liu, A. Shahroudy, D. Xu, A. C. Kot, and G. Wang, "Skeleton- based action recognition using spatio-temporal LSTM network with trust gates," IEEE Trans. Pattern Anal. Mach. Intell., to be published, doi: 10.1109/TPAMI.2017.2771306.

Y. Li, C. Lan, J. Xing, W. Zeng, C. Yuan, and J. Liu, "Online human action detection using joint classication-regression recurrent neural networks," in Proc. ECCV, 2016, pp. 203220.

A. Kumar et al., "Ask me anything: Dynamic memory networks for natural language processing," in Proc. ICML, 2016, pp. 13781387.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in Proc. NIPS, 2015, pp. 24402448.

L. Yao et al., "Describing videos by exploiting temporal structure," in Proc. ICCV, 2015, pp. 45074515.

Y. Wang, S. Wang, J. Tang, N. OHare, Y. Chang, and B. Li. (2016). "Hierarchical attention network for action recognition in videos." [Online]. Available: https://arxiv.org/abs/1607.06416 .

K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras, "Two-person interaction detection using body-pose features and multiple instance learning," in Proc. CVPRW, 2012, pp. 2835.

G. Evangelidis, G. Singh, and R. Horaud, "Skeletal quads: Human ICPR, 2014, action recognition using joint quadruples," in Proc. pp. 45134518.