applied sciences

MDPI

Article

# Similarity Estimation for Large-Scale Human Action Video Data on Spark

**Weihua Xu, Md Azher Uddin, Batjargal Dolgorsuren, Mostafijur Rahman Akhond, Kifayat Ullah Khan, Md Ibrahim Hossain and Young-Koo Lee ***

Data and Knowledge Engineering Lab, Department of Computer Science and Engineering, Kyung Hee University, Suwon 446-701, Korea; weihuaxu18@163.com (W.X.); azher006@yahoo.com (M.A.U.); bdolgorsuren@khu.ac.kr (B.D.); mostafij@khu.ac.kr (M.R.A.); kualizai@khu.ac.kr (K.U.K.); ibrahim.bd@khu.ac.kr (M.I.H.)

* Correspondence: yklee@khu.ac.kr

Received: 6 April 2018; Accepted: 8 May 2018; Published: 14 May 2018


**Abstract:** The amount of human action video data is increasing rapidly due to the growth of multimedia data, which increases the problem of how to process the large number of human action videos efficiently. Therefore, we devise a novel approach for human action similarity estimation in the distributed environment. The efficiency of human action similarity estimation depends on feature descriptors. Existing feature descriptors such as Local Binary Pattern and Local Ternary Pattern can only extract texture information but cannot obtain the object shape information. To resolve this, we introduce a new feature descriptor, namely Edge based Local Pattern descriptor (ELP). ELP can extract object shape information besides texture information and ELP can also deal with intensity fluctuations. Moreover, we explore Apache Spark to perform feature extraction in the distributed environment. Finally, we present an empirical scalability evaluation of the task of extracting features from video datasets.

**Keywords:** human action similarity estimation; feature extraction; edge detection

## 1. Introduction

A large amount of human action video data has sharply increased due to the expansion of multimedia devices, the Internet and human activities. Users perform different actions that have similarities between different human action videos stored on the database. Similarity estimation of human action video data has been connected with various human activity applications such as similarity measure to recognize human activity [1] and similarity measure for matching correspondence [2,3]. Moreover, dancing games such as Let's Dance with Mel B [4] and Just Dance [5] are developed for entertaining purposes so that users can have fun performing basic dance steps according to the reference dance steps. Different techniques have been used to measure the similarity between different human actions. Zhu et al. [1] applied smart-phone sensors to study the unfixed direction and position problems of smart-phone sensors; then, a similarity model is trained and used to recognize human activity. Han et al. [6] used smart-phone sensors to train several activity models and lastly applied trained models to judge human activities. Zhu et al. [1] constructed a similarity model based on two different activity feature vectors using acceleration sensors of smart-phones to collect data, and compared feature similarities of extensive location to recognize human activities. However, wearable sensors on human bodies are still not convenient. Several research papers [6,7] claimed that wearable sensors are obtrusive for a person. In order to resolve this issue, we devise a novel scenario for measuring human action similarity by comparing the videos of human actions considering image processing techniques. Suppose we have one reference video on the database and one user performed

that video, so, in order to obtain similarities, we compute the similarity between each user frame with the frames of reference videos.

On the other hand, in order to process a large amount of human action video data, we need to adopt a distributed environment. Usually, researchers utilize Apache Spark [8] and Hadoop to process big data in parallel. Apache Spark processes a large amount of data in a distributed way, and it increases efficiency since it performs in memory operation. Spark provides a comprehensive framework to manage and process big data. Uddin et al. [9] performed action recognition using an ALMD (Adaptive Local Motion Descriptor) based feature descriptor and the Spark MLlib Random Forest [10]. Human action recognition using Distributed Trajectory based features and Distributed Gaussian Mixture Models is proposed in [11], while our work focuses on estimating the similarity between the user performed action video and reference action video that is stored on the database. Lv et al. [12] employed Spark to detect near duplicate videos. Similarly, Liu et al. [13] deployed a Hadoop based environment to work with J2EE (Java 2 Platform, Enterprise Edition) [14] and Flex for managing large scale videos in a distributed way.

Feature extraction from video frames plays an important role in similarity measures because here human body position extracted features are compared. Previously, several feature descriptors were proposed; Local Binary Pattern (LBP) is one of the most popular feature descriptors due to its high efficiency of texture extraction. Feng et al. [15] adopted a Local Binary Pattern to extract facial appearance features and then applied it to a coarse-to-fine classifier. However, in [15], the Local Binary Pattern just gets facial texture features and cannot extract facial shape feature information. One more challenging issue is that similar actions performed between user videos and reference videos could be located in different frames since movement speed in different human action videos is different. Hence, how to define which user frame is compared with which reference frame is challenging.

In this work, we devise a novel framework for measuring human action similarity on the top of Apache Spark. At first, we deploy a Apache Spark based cluster environment. Our proposed method is divided into a feature extraction phase and similarity measure of extracted features respectively. We store all human action videos into the Hadoop Distributed File System (HDFS) and videos stored in HDFS are loaded into a Spark cluster represented by resilient distributed datasets (RDDs) [16]. During the process of RDD partition and transformation, we adopt flatMap (e.g., which is similar to map function, but each input element can be mapped to 0 or more output elements) included in Spark Core to process human action similarity computation in parallel in every worker node including the master node. Moreover, we proposed a novel feature descriptor called Edge based Local Descriptor pattern that can extract shape features and texture features as well, which is inspired from Local Binary Pattern [17], and Local Ternary Pattern [18] that only extract texture information. We obtain the final feature vector using our proposed ELP. Next, feature vectors are saved into HDFS. After that, we compute the similarity between feature vectors using cosine similarity. Finally, in order to validate the performance of our proposed descriptor, we performed mean average precision on the UCF (University of Central Florida) Sports action dataset [19]. The experimental result shows that proposed ELP has better performance, compared with other feature descriptors. Moreover, the scalability of proposed approach is evaluated.

The rest of the article is organized as follows: in Section 2, related work is shown. In Section 3, we describe our proposed framework including preprocessing, feature extraction and similarity measure components. In Section 4, we implement the proposed method by conducting experiments and analyze experiment results. In Section 5, we discuss performance of the proposed approach and application scenarios using the proposed method. In Section 6, we get conclusions and future work.

## 2. Related Work

In this section, we are going to discuss some related research to measure human action similarity. However, most of the existing research largely depends on external sensor based approach. Han et al. [6] proposed a smartphone-based hierarchical activity modeling and real time activity

recognition that extends the Naïve Bayes approach for the processing of activity modeling and real-time activity recognition, and study the Coarse classification and the Fine classification for creating the modified template-based classification. Afterwards, Zhu et al. [1] applied smart-phone sensors to study the unfixed direction and position problems of smart-phone sensors, then a similarity model is trained and used to recognize human activity. However, wearable sensors are inconvenient and clumsy.

In order to solve this issue, we measure human action similarity by comparing the videos of human action considering image processing techniques. One of the most popular feature descriptors is the Local Binary Pattern operator. In previous research, the Local Binary Pattern operator has been employed to analyze the face features [15]. Ojala et al. [17] proposed a simple and efficient, multi-resolution approach to do gray-scale and rotate invariant texture classification based on applying Local Binary Patterns (LBP). However, Local Binary Patterns can only extract static texture information. Similarly, Hafiane et al. [20] proposed an approach with multi resolution to handle proportional variations and use a set of classification, which achieves the better capability of texture recognition and its computation theory outperforms classical algorithms. Afterwards, Bashar et al. [21] proposed a feature descriptor, which is used to extract facial appearance features and constructed with a new local texture pattern called the median ternary pattern (MTP) aiming for recognizing facial expression. However, MTP is also only for texture feature extraction. A novel human activity recognition method is proposed, utilizing independent components of activity to mold information from image sequences and Hidden Markov Model (HMM) [22]. Similarly, Uddin et al. [9] came up with a novel feature descriptor called Adaptive Local Motion Descriptor (ALMD) that considers motion and appearance and they utilize this feature descriptor to do batch processing in a Spark cluster environment. ALMD is a kind of combination of LBP combining two continuous frames for moving motion feature extraction. Tan and Triggs [18] proposed a feature extraction technique by integrating the advantages of normalizing robust illumination, selecting new local texture-based face representations, matching through transforming distance, kernel-based feature extraction and ways of fusing multiple features to prompt recognition more reliable under uncontrolled lighting. However, this integration in applications is time-consuming. Jabid et al. [23] proposed a robust facial image descriptor to analyze toner and skin information effectively for recognizing face expression called Local Transitional Pattern. Abdesselam [24] proposed a method for improving LBP-based methods by constructing two different LBP histograms one for edge pixels and the second for non-edge pixels. However, the idea in [24] relies on salient regions, where human attention focuses and human observation is not verified. An automated way for recognizing facial expression using gradient-based ternary texture patterns are presented in [25]. Discriminative robust local binary pattern and discriminative robust ternary pattern in [26] are proposed for tackling the issue of differentiating a bright object from a dark background. Boudissa et al. [27] proposed an algorithm and a framework for real-time pedestrian detection regarding to low-resolution images, and they introduced a novel threshold selection way using the mean-variance of the background samples. However, their proposed method can only tackle partially the problems of edge-based methods. Li and Zhang [28] proposed a novel approach to extract features by fusing gradient direction and LBP features. Rodriguez et al. [29] introduced a template-based method for recognizing human actions called Action MACH based on a Maximum Average Correlation Height (MACH) filter.

Zhang et al. [30] proposed a comprehensive cloud-based architecture and a platform for batch processing integrated with fast processing that can provide intelligent analysis and storage with a robust solution for a large amount of video data. Kim et al. [31] presented a Hadoop-based Distributed Video Transcoding System in a cloud computing environment that is used to transform various video codec formats into the MPEG-4 video format. Tan and Chen [32] came up with an approach for fast and parallel video processing on MapReduce-based clusters, which is able to cope with large-scale video data and processing time is reduced obviously. They implemented application scenarios such as motion detection, face detection and tracking. Based on the parallel processing and flexible storage capabilities of cloud computing, Liu et al. [13] presented a practical

massive video management platform using Hadoop, which processes rapidly by using MapReduce with good usability, performance as well as availability. Moreover, they analyzed I/O simultaneous property. Zaharia et al. [16] came up with Resilient Distributed Datasets (RDDs), a distributed memory abstraction that lets programmers perform in-memory computations with cache on large clusters in a fault-tolerant manner. They make full use of RDD and cache to improve computation speed in Spark clusters. Lastly, Wang et al. [11] explored Apache Spark to recognize large-scale human action and implemented several techniques on Spark for recognizing human action recognition.

## 3. Proposed Framework

In this section, we describe the process of similarities estimation of human action video data using the proposed method. Initially, all human action videos are stored into the HDFS. Then, all these human action videos are loaded into RDD. Feature extraction is implemented using proposed ELP. After that, extracted ELP feature vectors are saved into the HDFS. Lastly, Cosine Similarity is applied to compute similarities between extracted ELP feature vectors. Figure 1 demonstrates the proposed framework and processing mechanism for large-scale human activity videos and measuring human action similarity in the distributed environment.

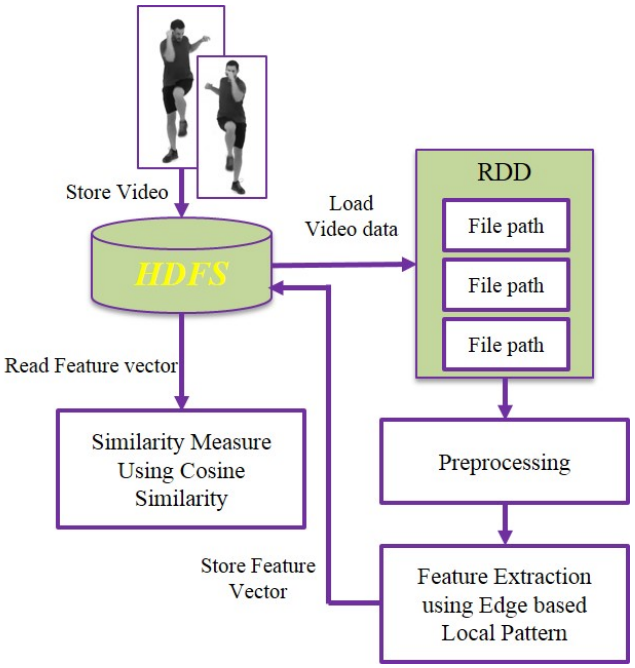


**Figure 1.** The proposed framework for human action similarity estimation.

### *3.1. Preprocessing*

In this phase, we performed the preprocessing on stored video data. Here, at first, we extract the frames from each videos, and then converted this frame into a gray level frame from RGB color frame. After that, we resize each frame so that user input frame and stored database video frame has the same size. Lastly, we perform background subtraction operation on each frame in order to separate the foreground human from the background. In our case, many users performing human activity videos following the reference human activity videos on the database. First of all, all of these videos are stored into the HDFS storage; then, the dataset stored in HDFS is loaded into the Spark cluster. For reference, for human activity videos that are stored in the database, the pre-processing step including the feature extraction step is performed in offline mode, while users performing human activity videos are processed on the fly. All nodes including the master node and worker nodes

perform this work in parallel. Partitions of this dataset are cached into worker nodes. All loaded data partitions and the created collection of objects are managed by RDD. An RDD is the representation of the information and a collection of partition record. RDD operations are accomplished in parallel using each worker node. In each partition, frames are extracted from the loaded videos. Next, frames are converted into gray scale from RGB mode. Then, these gray level frames are resized to $720 \times 404$. Lastly, we perform the background subtraction. Because our goal is to analyze human action similarity, we only need to analyze foreground objects (e.g., humans). In our work, humans are foreground objects. Here, we use Gaussian probability distribution [22] to process background subtraction. Here, we employed static background images to subtract the background from the frames of each activity video clip. The probability of being a pixel to be background is given as

$$P(R(x,y)) = \frac{1}{C}\sum_{i=1}^{C}\frac{1}{\sqrt{2\pi\sigma^2}}exp(\frac{-(R(x,y)-B_i(x,y))^2}{2\sigma^2}), \quad (1)$$

where $C$ is the number of background images. $P(R(x,y))$ is the probability of being a pixel of the background in $x$ and $y$ position of the recent frame. $B_i(x,y)$ and $R(x,y)$ are the intensity values of the pixel in $x$- and $y$-position of the $i$-th background image and recent frame, respectively.

### *3.2. Feature Extraction*

#### 3.2.1. Local Binary Pattern (LBP)

Local pixel information from each frame can be computed using basic Local Binary Pattern operator [17]. The Local Binary Pattern (LBP) is calculated by checking the pixel values of an image or frame with a threshold value in the neighborhood area. The $LBP_{P,R}$ generates $2^P$ different output codes. However, the basic LBP has several drawbacks. It is susceptible to noise, as slight variation in the intensities of the neighbors can completely change the resultant binary code [9].

#### 3.2.2. Sobel Operator for Edge Detection

In our work, we have utilized the Sobel Operator in order to obtain the shape information of the foreground human activities. The Sobel Operator is mainly used for detecting the edges in an image. Edges in an image represent areas where illumination changes significantly. Edges are the ending of one area and also the beginning of another area. Edge points refer to those points with coordinates $[x,y]$ and located in obvious illumination change regions. In mathematics, derivation is adopted to represent the speed of change. Mathematically, images can be regarded as two-dimensional discrete functions. Gradients of images are derivations of two-dimensional discrete functions. Suppose we have a function $f(x,y)$, the gradient of $f(x,y)$ in $(x,y)$ is a vector given by:

$$\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}]^T, \quad (2)$$

where $\nabla f$ stands for the gradient of $f(x,y)$ in $(x,y)$. Computation of this vector value is given by:

$$|\nabla f| = mag(\nabla f) = [(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2]^{\frac{1}{2}}, \quad (3)$$

where $|\nabla f|$ stands for the magnitude of the gradient of $f(x,y)$ in $(x,y)$. In order to improve the efficiency, it is normally given by:

$$\nabla f \approx |G_x| + |G_y|, \quad (4)$$

where $\nabla f$ stands for approximate absolute values of the gradient of $f(x,y)$ in $(x,y)$. The gradient magnitude is represented as:

$$|G| = \sqrt{Gx^2 + Gy^2}. \quad (5)$$

3.2.3. Edge Based Local Pattern Descriptor

In this section, we illustrate the proposed Edge based Local Pattern Descriptor that is applied for extracting human action features. We utilize the Sobel Operator to compute gradient magnitude, which represents the shape information of human activity. Firstly, we apply Sobel Operator to obtain the gradient magnitude. Then, on these gradient values, we apply our method to find the features. The following equations are used for the ELP. Figure 2 demonstrates the computation process of our proposed ELP:

$$ELP(x_c, y_c) = \sum_{n=1}^{n-1} s(g_p - g_c) \times 2^p, \tag{6}$$

$$s(x) = \begin{cases} 1, & \text{if } x \geq M_c \,, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

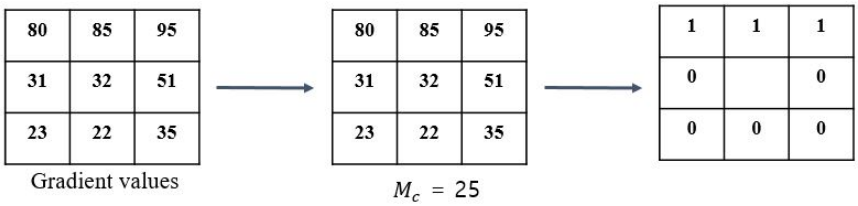


**Figure 2.** An example of Edge based Local Pattern Descriptor.

Here, $M_c$ denotes the average of $|g_p - g_c|$ values of the local neighbor pixels around the center pixel $(x_c, y_c)$ and $g_p$ corresponds to the gradient value of the local neighbor pixels. We compute the difference between the gradient value of each neighbor pixel and that of the center pixel sequentially. Then, we do the summation of these eight pairs of differences. After that, we can get the average of the difference value between neighbor gradient values and the center gradient value by computing the mean value of these eight pairs of differences. Thus, we get the $M_c$. If the difference between a neighbor gradient value and the center gradient value is greater to or equal to $M_c$, this neighbor value is transformed into 1. Otherwise, it is transformed into 0. We implement the same process to other neighbor gradient values in sequence. Therefore, we get a binary string. Next, we just need to compute the binary string to transform the binary string into decimal values and do summation. At the moment, it means that we obtain the ELP value of the center pixel. We repeat the implementation to compute all other pixels to get ELP value of all pixels of input frame. Thus, we get ELP feature vectors.

*3.3. Similarity Measure*

Cosine similarity is applied to estimate the similarity of human action from two human action videos (e.g., user performed video and reference video). We also need to consider the human movement speed when comparing two different videos. Hence, we apply the neighbor frame comparison scenario. Suppose we have one reference video and one user performed a video to be compared for obtaining similarities. We compare each user frame with three neighbor frames of reference videos. The reason is that if a user performs a motion late, then it is a wrong motion in consideration of the whole action performance similarity because we are comparing motion similarity between a user action video and reference action video. More importantly, all action performances are successive. The maximum comparison computation value of each comparison group is the final similarity value of the user frame. The comparison process of all frames between two videos are demonstrated in Figure 3. Table 1 shows similarity measures with different neighbor frames. Two feature vectors correspond to the user video $F_q$ and the reference video $F_d$, respectively:

$$cos\theta = \frac{F_q \cdot F_d}{\|F_q\| \cdot \|F_d\|}. \tag{8}$$

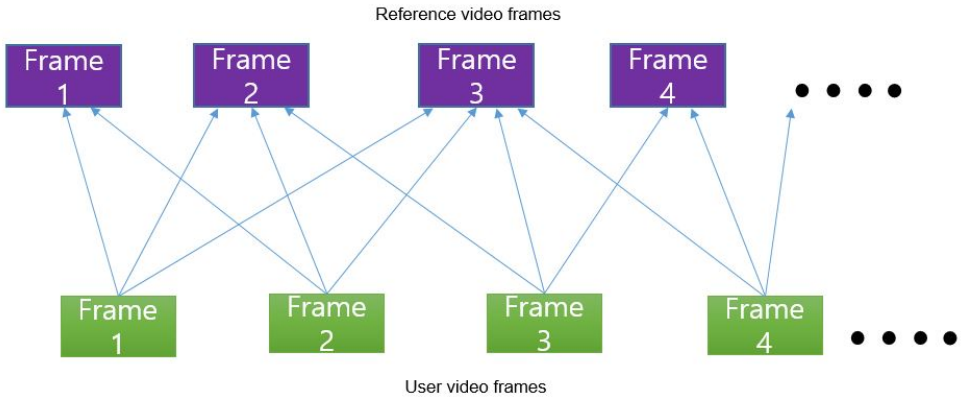


**Figure 3.** Similarity measure computation framework.

In Algorithm 1, for the process of the feature extraction algorithm, input is human action video data, and output is ELP feature vectors. We have *Video_Name N* and *Video_Data D* for input. We have *Video_Name N* and *Feature_Vector V* for output. Load human action video data stored in the HDFS. Then, $flatMap$ stage is implemented circularly. The background image is defined as $BI$ . The number of frames is defined as $i$. For all video data, background subtraction is applied. it is defined as $Ba$ after finishing background subtraction. Then, ELP is applied to all $Ba$, thus feature vectors are obtained. In this way, all feature vectors of relevant human action video are obtained, defined as $(N, V)$.

**Algorithm 1:** Feature Extraction

**Input:** RDD [VideoName $N$ , VideoData $D$]

**Output:** RDD [VideoName $N$ , FeatureVector $V$]

**for all** ($N$,$D$) $\in$ RDD[$N$,$D$] **parallel do**
**/*flatMap stage*/**
  BI $\leftarrow$ BackgroundImage
  **for** i=1 **to** NumberofFrame
    a $\leftarrow$ D[i]
    Ba $\leftarrow$ apply BackgroundSubtraction(a,BI)
    V[i] $\leftarrow$ apply ELP(Ba)
  **end for**
  add ($N$,$V$ to result RDD)
**end for**

**Table 1.** Similarity measure with different neighbor frames.

| # of Frames | Neighbor Frames | Computation Time (Seconds) |
|---|---|---|
| 136 | 3 | 94.3782 |
| 136 | 4 | 108.8361 |
| 136 | 5 | 129.1928 |
| 272 | 3 | 170.9647 |
| 272 | 4 | 194.7050 |
| 272 | 5 | 232.8930 |

## 4. Experiments and Evaluation

### *4.1. Experiment Setup*

In order to validate the performance of human action similarity on Spark, we measured mean Average Precision for ELP. We employ Hadoop and Spark cluster with configuration setup to

implement the proposed method. We use three nodes. One node is the master node. At the same time, all three nodes are worker nodes. Hence, three nodes work parallell in the distributed environment. Each node has four cores, 32 GB memory as well as 3.0 GHz. The Hadoop version that we adopt in this work is 2.7.1. The Spark version that we adopt in this work is 1.6.2.

### *4.2. Datasets*

UCF YouTube action dataset [19] is employed in our implementation. It is composed of eleven action categories such as basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, trampoline jumping, volleyball spiking, as well as walking with a dog. It consists of 1375 human action activity videos. Figure 4 shows several sample frames.


**Figure 4.** Sample frames of the UCF (University of Central Florida) Sports action dataset.

### *4.3. Experimental Results and Analysis*

In order to validate the performance of the proposed feature descriptor, we have computed the mean average precision (mAP) for the proposed method. Figure 5 demonstrates precision result.

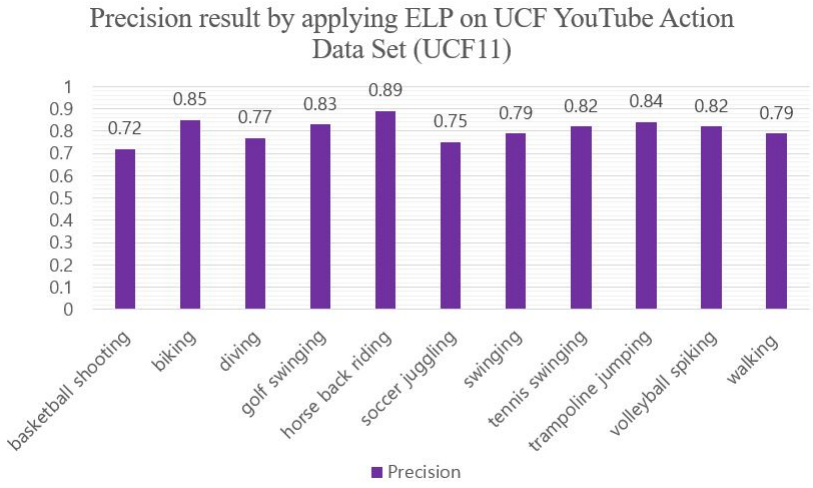


**Figure 5.** Mean average precision by applying our proposed method.

We observed that MAP (Mean average precision) of horse back riding is the highest (0.89). MAP of biking and tennis swinging are 0.85 and 0.84, respectively. MAP of basketball shooting is the lowest (0.72) due to dataset complexity. We also compared the proposed ELP with LBP [17], MBP [20], LTP [18] and MTP [21] feature descriptors. Experimental results show that the proposed ELP shows better performance. Figure 6 demonstrates the performance of the proposed ELP.

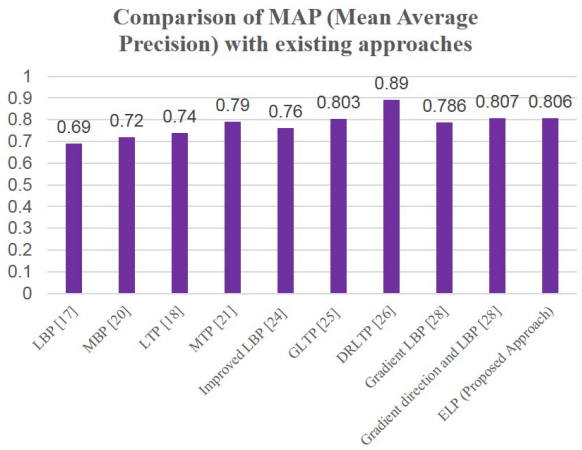


**Figure 6.** Comparison between proposed Edge based Local Pattern descriptor (ELP) and other feature extraction methods using mean average precision.

The MAP of our proposed ELP is 0.806. The proposed ELP shows competitive results compared with other feature descriptors. The MAP of ELP is a little bit higher than the MTP. MAP of LBP is 0.69, which is the lowest. However, DRLTP (Discriminative Robust Local Ternary Pattern) [26] shows the best results among all of them, since they contain both edge and texture information along with discriminative power.

Furthermore, in order to guarantee the scalability of our proposed approach, we test the scalability of our proposed approach by implementing on the stand-alone, two nodes and three nodes, respectively. The computation time is 82,562 s when we process the whole computation on the standalone. The computation time is 50,037 s when we implement on two nodes. The computation time is 33,024 s when we implement our proposed method on three nodes. Figure 7 illustrates the scalability of our proposed approach when using three different nodes.

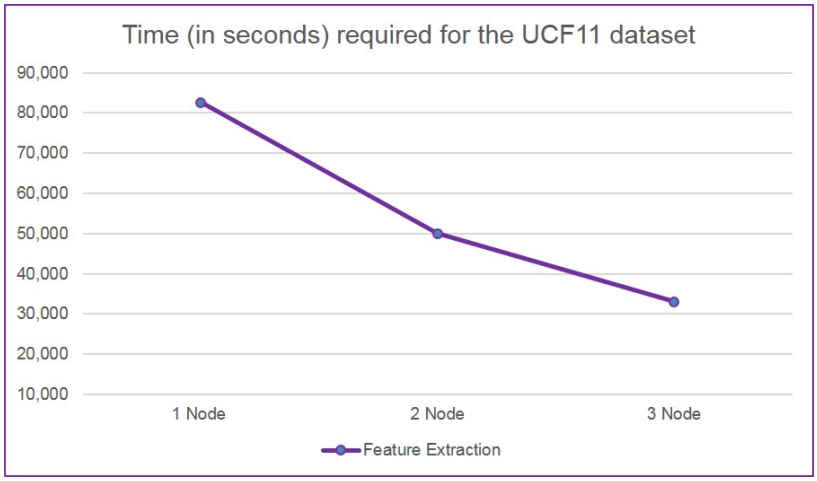


**Figure 7.** Scalability of human action similarity measure in Spark using three different nodes.

Therefore, we can notice that the computation time decreases when the number of nodes increase. The computation time reduces dramatically when we run it on two nodes. The time reduction of applying three nodes is less than applying two nodes. We observe that both the size of datasets and the number of nodes determine computation time of processing in a distributed environment.

## 5. Discussion

ELP shows competitive results on the UCF Sports action dataset. Our proposed approach for similarity measure of human action video data on top of Spark can be employed in different application scenarios such as estimating the similarity in exercise and dancing games [5]. For example, lots of users are performing exercise or dance activities following reference videos in the database, while cameras record all their actions. After they finish their performance, these recorded videos are delivered to the HDFS and compared with reference videos saved in the HDFS. In this case, the proposed method can measure similarities of users' actions and reference actions. Therefore, it can assist users to know how similar their actions are to reference actions when they do these kinds of human activities.

## 6. Conclusions

In this paper, we propose a novel approach for similarity estimation using Spark to process large-scale human action activity video data. Moreover, we introduce a new feature descriptor, namely an Edge based Local Pattern descriptor, which can obtain shape features and deal with intensity fluctuations besides texture features. We also illustrated an empirical scalability evaluation for feature extraction task from the video dataset. The experiment results demonstrate that the proposed method can resolve similarities estimation for large scale human action video data. In the future, we will try to adopt a Dynamic Time Warping approach in order to address the issue of estimating similar actions performed in two videos that are located in different frame lengths.

**Author Contributions:** Y.-K.L. provided guidance for improvement during the discussions; W.X. conceived the key idea and was in charge of writing the manuscript; M.A.U. contributed to the idea optimization, experiment implementation and updating of the manuscript; B.D. worked on analysis of experiment results; M.R.A. and M.I.H. assisted with the video processing phase; K.U.K. evaluated the proposed idea.

**Acknowledgments:** This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2016-0-00406, SIAT CCTV Cloud Platform).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Zhu, Y.; Wang, C.; Zhang, J.; Xu, J. Human activity recognition based on similarity. In Proceedings of the IEEE 17th International Conference on Computational Science and Engineering (CSE), Chengdu, China, 19–21 December 2014; pp. 1382–1387.
2. Lee, S.Y.; Sim, J.Y.; Kim, C.S.; Lee, S.U. Correspondence matching of multi-view video sequences using mutual information based similarity measure. *IEEE Trans. Multimedia* **2013**, *15*, 1719–1731. [CrossRef]
3. Lehmann, A.; Bouthemy, P.; Yao, J.F. Rank-test similarity measure between video segments for local descriptors. In *International Workshop on Adaptive Multimedia Retrieval*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 71–81.
4. Black Bean. Let's Dance with Mel B. Available online: https://www.playstation.com/en-gb/games/lets-dance-with-mel-b-ps3/ (accessed on 19 April 2018).
5. Ubisoft. Just Dance. 2011. Available online: http://just-dancethegame.ubi.com/jd-portal/en-gb/home/index.aspx (accessed on 19 April 2018).
6. Han, M.; Bang, J.H.; Nugent, C.; McClean, S.; Lee, S. A lightweight hierarchical activity recognition framework using smartphone sensors. *Sensors* **2014**, *14*, 16181–16195. [CrossRef] [PubMed]
7. Lu, H.; Yang, J.; Liu, Z.; Lane, N.D.; Choudhury, T.; Campbell, A.T. The jigsaw continuous sensing engine for mobile phone applications. In Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, Zurich, Switzerland, 3–5 November 2010; pp. 71–84.

8. Zaharia, M.; Chowdhury, M.; Franklin, M.J.; Shenker, S.; Stoica, I. Spark: Cluster computing with working sets. In Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud'10), Boston, MA, USA, 22–25 June 2010.
9. Uddin, M.A.; Joolee, J.B.; Alam, A.; Lee, Y.K. Human Action Recognition Using Adaptive Local Motion Descriptor in Spark. *IEEE Access* **2017**, *5*, 21157–21167. [CrossRef]
10. Meng, X.; Bradley, J.; Yavuz, B.; Sparks, E.; Venkataraman, S.; Liu, D.; Freeman, J.; Tsai, D.B.; Amde, M.; Owen, S.; et al. MLlib: Machine learning in apache spark. *J. Mach. Learn. Res.* **2016**, *17*, 1235–1241.
11. Wang, H.; Zheng, X.; Xiao, B. Large-scale human action recognition with Spark. In Proceedings of the IEEE 17th International Workshop on Multimedia Signal Processing (MMSP), Xiamen, China, 19–21 October 2015; pp. 1–6.
12. Lv, J.; Wu, B.; Yang, S.; Jia, B.; Qiu, P. Efficient large scale near-duplicate video detection base on spark. In Proceedings of the 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, USA, 5–8 December 2016; pp. 957–962.
13. Liu, X.; Zhao, D.; Xu, L.; Zhang, W.; Yin, J.; Chen, X. A distributed video management cloud platform using hadoop. *IEEE Access* **2015**, *3*, 2637–2643. [CrossRef]
14. J2EE. Available online: http://www.oracle.com/technetwork/java/javaee/appmodel-135059.html (accessed on 19 April 2018).
15. Feng, X. Facial expression recognition based on local binary patterns and coarse-to-fine classification. In Proceedings of the Fourth International Conference on Computer and Information Technology (CIT'04), Wuhan, China, 16 September 2004; pp. 178–183.
16. Zaharia, M.; Chowdhury, M.; Das, T.; Dave, A.; Ma, J.; McCauley, M.; Stoica, I. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation, San Jose, CA, USA, 25–27 April 2012.
17. Ojala, T.; Pietikainen, M.; Maenpaa, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 971–987. [CrossRef]
18. Tan, X.; Triggs, B. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Trans. Image Process.* **2010**, *19*, 1635–1650. [PubMed]
19. Soomro, K.; Zamir, A.R. Action Recognition in Realistic Sports Videos. In *Computer Vision in Sports*; Springer International Publishing: Cham, Switerland, 2014.
20. Hafiane, A.; Seetharaman, G.; Zavidovique, B. Median binary pattern for textures classification. In Proceedings of the International Conference Image Analysis and Recognition, Montreal, QC, Canada, 22–24 August 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 387–398.
21. Bashar, F.; Khan, A.; Ahmed, F.; Kabir, M.H. Robust facial expression recognition based on median ternary pattern (MTP). In Proceedings of the International Conference on Electrical Information and Communication, Khulna, Bangladesh, 13–15 February 2014; pp. 1–5.
22. Uddin, M.Z.; Lee, J.J.; Kim, T.S. Shape-based human activity recognition using independent component analysis and hidden markov model. In Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Wrocław, Poland, 18–20 June 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 245–254.
23. Jabid, T.; Chae, O.S. Local Transitional Pattern: A Robust Facial Image Descriptor for Automatic Facial Expression Recognition. In Proceedings of the International Conference on Computer Convergence Technology, Seoul, Korea, 29 November–1 December 2011; pp. 333–344.
24. Abdesselam, A. Improving local binary patterns techniques by using edge information. *Lect. Notes Softw. Eng.* **2013**, *1*, 360. [CrossRef]
25. Ahmed, F.; Hossain, E. Automated facial expression recognition using gradient-based ternary texture patterns. *Chin. J. Eng.* **2013**, *2013*, 831747. [CrossRef]
26. Satpathy, A.; Jiang, X.; Eng, H.L. LBP-based edge-texture features for object recognition. *IEEE Trans. Image Process.* **2014**, *23*, 1953–1964. [CrossRef] [PubMed]
27. Boudissa, A.; Tan, J.K.; Kim, H.; Shinomiya, T.; Ishikawa, S. A novel pedestrian detector on low-resolution images: Gradient LBP using patterns of oriented edges. *IEICE Trans. Inf. Syst.* **2013**, *96*, 2882–2887. [CrossRef]
28. Li, Y.; Zhang, L. Feature Fusion of Gradient Direction and LBP for Facial Expression Recognition. In Proceedings of the Chinese Conference on Biometric Recognition, Tianjin, China, 13–15 November 2015; Springer: Cham, Switzerland, 2015; pp. 423–430.

29. Rodriguez, M.D.; Ahmed, J.; Shah, M. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
30. Zhang, W.; Xu, L.; Duan, P.; Gong, W.; Lu, Q.; Yang, S. A video cloud platform combing online and offline cloud computing technologies. *Pers. Ubiquitous Comput.* **2015**, *19*, 1099–1110. [CrossRef]
31. Kim, M.; Cui, Y.; Han, S.; Lee, H. Towards efficient design and implementation of a hadoop-based distributed video transcoding system in cloud computing environment. *Int. J. Multimedia Ubiquitous Eng.* **2013**, *8*, 213–224.
32. Tan, H.; Chen, L. An approach for fast and parallel video processing on Apache Hadoop clusters. In Proceedings of the 2014 IEEE International Conference on Multimedia and Expo (ICME), Chengdu, China, 14–18 July 2014; pp. 1–6.