Webology (ISSN: 1735-188X)
Volume 17, Number 1, 2020

Transformer Network For The Recognition Of Human Activity Using Smartphone Data

Abhishek Jain[1], Parag Jain[2], Aman Kumar[3], Gaurav Chaturvedi[4], Gaurav Gupta[5], Deepak Arya[6]

[1,2,4,5,6]Dept. of CSE, Roorkee Institute of Technology, Roorkee, UK, India

[3]Department of Computer Science, FOE, TMU, UP, India

Abstract

Automatic behavior analysis for sports players, older people, and IoT applications requires sensor-based human activity identification. Conventional means rely on finger features to tend to fluctuate from raw data using preset mathematical concepts and therefore are incapable of domain adaptation. We proposed a Transformer network in this study to recognize six human actions from Smartphone data. Each participant did six activities while wearing a Smartphone (Samsung Galaxy S II) around their waist (walking, walking upstairs, and walking downstairs, sitting, standing, and laying). The tests were recorded and labeled manually and partitioned randomly into two groups. Furthermore, the performance of the proposed model is evaluated.

Keywords: Transformer Network, Smartphone data, Human activities.

1. Introduction

Wearable electronics are becoming more common in human populations due to the exponential rise of computer technology. Smartphone use, in particular, is compelling, regardless of one's financial situation. Due to miniaturization technology, several sensors are included in a low-cost Smartphone. The inbuilt sensors in smartphones, such as the accelerometer and gyroscope, create a vast quantity of usable data that may be utilized to forecast and categorize human behaviors automatically. Human activity recognition has the potential to be employed in senior homes, particularly in nations where the average age of the population is increasing[1]. Similarly, it may aid in the analysis of a sports player's movements and, as a result, aid in the improvement of player performance. Similarly, with smart homes and IoT-based systems, sensor-based human activity identification is crucial. Owing to the potential utility in a wide array of uses, it would be an active subject of research in object recognition, and considerable breakthroughs have been made in recent decades[2].

The following two major categories may be used to categorize human activity algorithms.

- Reconditioning of vision-based activities
- Activity identification based on sensors

The purpose of both types of algorithms is the same. The manner of data collecting and processing, on the other hand, is considerably different.

Vision-Based activity recondition

Tracking and understanding the behavior of agents using films captured by numerous cameras is a highly significant and difficult issue. Computer Vision is the key approach used. Vision-based action recognition has a variety of applications, including human contact, user experience design, robotics training, and monitoring. RGB or RGBD data is used to assess human activities in vision-based activity recognition [13]. Many solutions rely on advanced methods and portray anthropogenic classification as a problem of scene comprehension. The visual flow field of a video is used to define a complex system. The dynamic system's stability analysis is then utilized to characterize human behavior. Furthermore, a social interaction model is employed to describe human interaction in crowded environments.

Sensor-based activity recognition

The creation of diverse context-aware applications in new sectors has been spurred by human activity recognition technology that analyses data gathered from many kinds of sensing devices, including vision sensors and embedded sensors. Human motion detection based on sensors has been employed in a range of real-world applications, such as smart homes and hospitals. Furthermore, the rapid expansion of wireless sensor networks (WSN) has led to a massive amount of data being gathered from a range of sensors, such as sensing devices, object detectors, and monitoring devices[3].

Identification based on sensors Instead of extracting visual information (raw pixel, gradients, edges, orientation, etc.) and movement characteristics (optical flow) from the image, perception-action identification collects 1D sensed data.

In this paper, a Transformer Network method is proposed and its performance of it was evaluated.

The rest of the paper is organized as follows; in section-2 the literature review is given. In section-3 the used Transformer Network is given. Section-4 deals with the proposed methodology. Section-5 deals with the results and discussion and finally, the paper is concluded in section-6.

2. Literature Review

[4] had looked at the use of body-worn sensors to monitor the vibrations that influence the human body while alpine skiing, both in terms of general injury prevention and back overuse injury prevention. For each section examined, the spectrogram density was calculated. In addition, as a measure of the intensity of vibration exposure, the rhizome (RMS) acceleration impacting the neck and back was calculated based on the vestibular speed and along the sacrum's longitudinal axis. In

both Gc and SL skiing, the PSD ratings of both the vibrations acting at the neck were significantly elevated for harmonics below 30 Hz. The thigh and hip joints progressively lowered the vibrations as they traveled and through the body. For harmonics above 4 and 10 Hz, PSD values were particularly evident in the lower back, while a comparing of GS and SL revealed that GS had higher PSD values and larger RMS values. As a consequence, any solution that may reduce and/or eliminate ski-related vibrations should be sought out and implemented.

[5] had shown that extending the temporal extents of LTC-CNN models improves action recognition accuracy. The impacts of a variety of low-level expressions, such as actual video pixels and input images vector fields, were also explored, and it was shown that high light flow estimates are crucial for learning proper action models. In this article, video representations are learned using machine learning with long-term temporal convolutions. The results of two tough human action recognition criteria, UCF101, and HMDB51 were deemed state-of-the-art.

[6] had presented a method in which a matrix of particles is placed on a picture, which then initiates a dynamical system characterized by visual features, which offers high-level worldwide optical flow. The author also demonstrated a technique for automatically assessing videos in actual settings and determining the entrance (sources) and exit (sinks) points for the most relevant pedestrian flows. The approach employs optical flow to identify pedestrian movements and considers the reconstructed video as a collection of sequences. The others are processed separately to produce trackless, which get aggregated to provide globe trajectories that characterized supplies and sinks as well as pedestrian movement within the scene. Finally, local parts information is connected to produce a global aggregate of traces that identify sources and sinks while also measuring pedestrian activity across them.

[7] had proposed a customer deep learning-based approach for virtual human influence categorization. We recommend using Dcnns in combination with fundamental statistical features that preserve information about the general shape of data series for local extraction of features. The impact of time series length on recognition rate was also investigated, and it was set to one second, permitting for time-series activity categorization. Allows greater and UCI, which include labeled sensor readings from 36 and 30 people, accordingly, but also a bridge experiment, are utilized to assess the correctness of the recommended technique. The results show that the proposed model provides cutting-edge performance at low computational complexity and without the need for human feature engineering.

3. Transformer Network

A converter is a deeper training algorithm that rates the relevance of each piece of the raw data separately using the self-attention technique. Processing (NLP) and computer science are two of their main uses (CV). The Transformer network was first introduced for a machine translation issue in which both the input and output are sequence data, but because of its strong performance in processing sequential data, it was swiftly adopted into other fields such as music and seismic

waves[8]. Transformer network uses a method of attention The core principle of focus is to pay attention to something learn a scoring system that gives each piece of input a distinct weight[9].

In particular, input temporal data $xx_i \in RR^{L\times M}$ First, a group of variables of different Lengths and dimensions M is linearly mapped. $KK \in RR^{L\times d}$ Queries $QQ \in RR^{L\times d}$ and $VV \in RR^{L\times d}$, where d is the dimension of KK, QQ, and VV. An attention model computes weights for each key about the query using those key-value element pairs (k_i, v_i) and a query q, then aggregates the values using these weights to generate the value corresponding to the query as specified by:

$$Attention(QQ, KK, VV) = soft\max\left(\frac{QQKK^T}{\sqrt{d}}\right)VV$$

The Estimated model is used to transform an input into a single vector that follows a posterior distribution and has a total sum of one. When the keys KK and queries QQ are similar in the preceding equation, self-attention is obtained. The ability to capture extended relationships between input data while simultaneously training neural networks is one of the key advantages of this approach over RNN and CNN.

We just use the encoder element of the Transformer in this project since our purpose is to categorize Smartphone data rather than generate a new sequence[10]. An input layer, a positional encoding layer, and a stack of NN identical encoder layers make up the Transformer encoder[11]. The input layer uses a fully-connected (FC) network to transfer the input time-series data to a vector of another dimension, which is a necessary step for the model to use the attention mechanism. Because the model lacks recurrence and convolution, There is just no knowledge of the sequence's arrangement; therefore positional encoding is utilized to represent the incoming sequence's relative and absolute position data by adapting sine and cosine functions of various frequencies to each input element[12]. The resultant vector is input into N encoder layers, where N is a pre-defined number that specifies the neural network's depth. Each encoder layer has two sub-layers: self-attention and a fully-connected feed-forward layer with addition and normalization operations in between. The architecture as a whole changes the input sequence into a new output sequence that includes data from all other input elements.

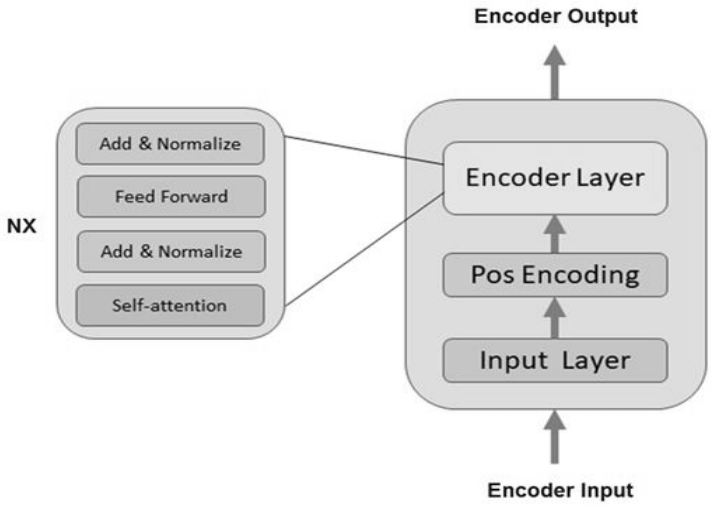

Figure 1. The architecture of the Transformer-based encoder model

4. Methodology

Data Set Information

The experiments were carried out on an organization of 30 people children aged 19 to 45-year - old. Each participant performed six tasks while having a Samsung Galaxy II smartphone across their belt (walking, walking upstairs, walking downstairs, sitting, standing, and laying). To use the smartphone's inbuilt multiple sensors, we measured the 3-axial sensor and 3-axial direction of rotation at a specified frequency of 50Hz. The tests were videotaped so that the results could be labeled manually. The obtained data were divided into two groups at random, with 70% of the participants providing tuples and 30% providing test data.

Sensor data was or before utilizing noise filtration before being collected in 2.56 sec remedied panes with 50% overlaps (128 readings per window). The gravitational and body movements components of the instrument acceleration data were separated into free image and gravity using a Butterworth low-pass filter. Because only low pass components of the gravitational force are predicted, a filter with just a stopband of 0.3 Hz was used. The time and spectrogram data were combined to provide a vector of attributes for each frame.

The model may accommodate many parallel sequences of raw data, including each axis of both the accelerometer sensor data. The model learns to extract from observational sequences and assign internal traits to different types of activities.

Robots for organizations offer the benefit of understanding features from the raw data sets, removing the requirement for domain expertise to generate input features manually. The model is expected to learn an internal structure of the time series analysis and, in principle, perform comparably to training images on an artificially enhanced version of the dataset.

Data Pre-processing step

The raw dataset must first be loaded into memory.

The three primary signal types in the original information are total acceleration, body speed, plus body gyroscope. There are three data axes on each one. This means how each time step contains nine characteristics in total. Further, each data set was separated into 128-time steps by spanning windows of 2.56 seconds. The design features (rows) views in the previous section are linked to these data windows. This means that each row of information has (128 * 9) elements, for a total of 1,152. This is almost half the size of both the 561 component vectors in the previous section, showing that some information is redundant. The signals are kept in the /Inertial 're interested directory's train and test subdirectories. For every one of the training and testing data to load, each vector from each signal is retained in one's file, producing in ninth input files but one output file. We can transfer these records in batches because of the consistent specific module and file naming conventions.

With spacing among columns, the input signal is in CSV format. Several files each contain an Array collection which may be imported. The loading file () tool loads data and returns it as a Data matrix if the document's fill path is specified.

The input for a homogeneous entity (train or tests) may now be imported into a singular muti NumPy array [samples, time steps].

Each term of the criteria data file has 128 intervals and nine features, with the sample data equaling the number of rows.

The load group() method implements this behavior. We can merge all of the importing 3D arrays into one 3D array with properties segregated on the 3d model using the NumPy method stack(features).

This technique will be used to download all of the data from an input signal for a specific group, like train or test.

The upload information group () procedure loads entire audio voltage data and produces data for a specific group using consistent naming standards across folders.

Finally, each of the train and test datasets may be loaded.

The resulting data is given as a decimal for the grading rubric. Before building a rnn cross-classification rules, these class codes must be hot input. The categorical() Keras function may help us do this.

The preload dataset() method implements this behavior by providing the train and tests X and y components, which are ready for training and evaluating the specified models.

Fit and Evaluate Model

Now because the input has been captured and is ready for modeling, we may create, fit, and assess a Converter model. We can make a method called evaluate model() that takes in the training and validation datasets, fits a model on the testing set, evaluates this one on the test data, and returns a performance estimate. We first should define the Titans model using the TensorFlow backend module. The model needs [samples, time steps, and characteristics] in a three-dimensional input. We exported the dataset in this manner, with each sampling being one pane of time series analysis, each window comprising 128 times, and each initial condition including nine independent factors. The model's output will be a six-element vector containing the probability of a certain window relating to each of the four activity types. Certain transmitter and receiver dimensions are required for fitting the model, and we may collect these from the testing set. A single convolutional Separator layer will be constructed for the model. The machine is then modified to the dataset, requiring the introduction of a dropout layer to reduce it. Finally, before generating judgments using a final feature layer, a thick fully connected layer analyses the features collected by the Inverter hidden layer. We'll use the Adam version of SVM to tune the network, and the categorized squared error loss function will be used since we're undertaking a multi-class classification problem. The template will be assisted for a specified iteration number, in this case, 15, with a total of 64 samples and 64 vents of data supplied to the model well before weights are updated.

After the hutment has always been fit, it is checked on the validation set, and the efficiency of the model upon this test data is returned.

Summarize Results

We can't assess the model's ability only on is there one examination.

This is because human brains are stochastic, meaning that developing the same model sitting on the very same data will result in a different model. This is a network trait that gives the model its adaptability, but it also makes model evaluation a bit more complex. We'll run the model assessment process many times and then total up the results of each run. We may, for example, use evaluate model() ten times. As a consequence, It will be necessary to construct a crowd of model assessment results. To characterize the sample of scores, the standard deviations of the output may be computed and given. The standard deviation reflects the dispersion of the model's accuracy from the mean, whereas the mean provides the model's accuracy percentage on the dataset. The summarize results() function sums together a run's output. As seen below, we may integrate the repetitive evaluation, data collecting, and statistics summary into one primary function for the research, run experiment().

Before such success is released, the model is evaluated 10 times by default.

5. Results and discussion

Table 1 Performance parameters of the proposed model

Metrics	parameters
Accuracy	98.41
sensitivity	97.43
specificity	100.00
F1 Score	98.70

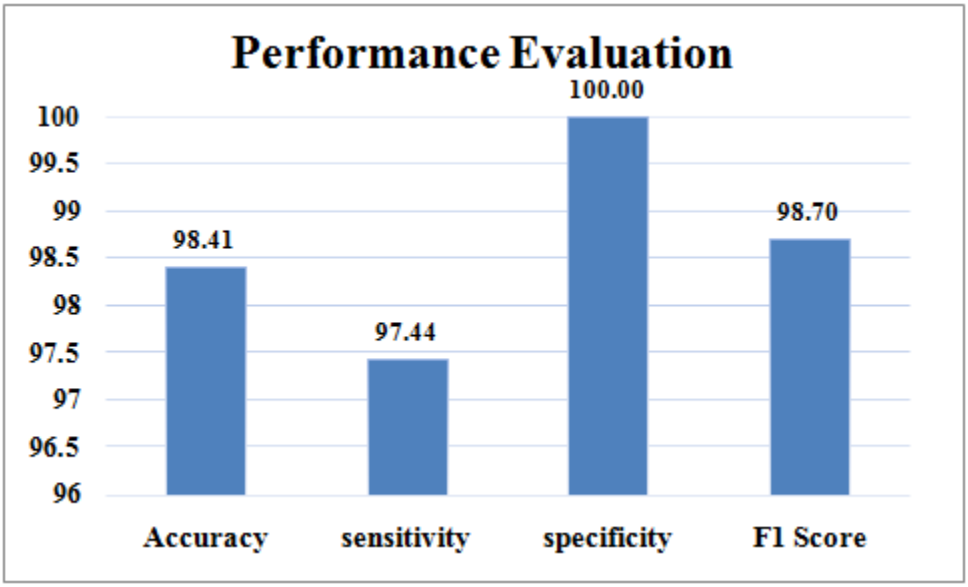

Figure 2. performance graph of the proposed model

From the above figure, it is observable that the proposed model accuracy is 98.41, sensitivity is 97.44, specificity is 100 and the F1 score is 98.70.

6. Conclusion

The proposed method is implemented with the help of python in a system with an Intel i7 core processer. In this research, we proposed a Transformer network to recognize six human behaviors using Smartphone data. Each participant completed six tasks while wearing a Samsung Galaxy S II Smartphone around their waist (walking, walking upstairs, walking downstairs, sitting, standing, and laying). The tests were manually recorded and categorized, then divided into two groups at random. Furthermore, the suggested model's performance is assessed. The performance of the proposed model is evaluated with performance metrics and the appropriate graph is visualized in the results section. Hence the accuracy of the model is 98.41 which indicates that the proposed model is accurate in classifying human activities using Smartphone data.

References

[1] S. K. Challa, A. Kumar, and V. B. Semwal, “A multibranch CNN-BiLSTM model for human activity recognition using wearable sensor data,” Vis. Comput., no. 0123456789, 2021, doi: 10.1007/s00371-021-02283-3.

[2] Jain, A., Dwivedi, R. K., Alshazly, H., Kumar, A., Bourouis, S., & Kaur, M. (2022). Design and Simulation of Ring Network-on-Chip for Different Configured Nodes. CMC-COMPUTERS MATERIALS & CONTINUA, 71(2), 4085-4100.

[3] S. Mahmud et al., “Human activity recognition fromwearable sensor data using self-attention,” Front. Artif. Intell. Appl., vol. 325, pp. 1332–1339, 2020, doi: 10.3233/FAIA200236.

[4] Kumar, A., & Jain, A. (2021). Image smog restoration using oblique gradient profile prior and energy minimization. Frontiers of Computer Science, 15(6), 1-7.

[5] G. Varol, I. Laptev, and C. Schmid, “Long-Term Temporal Convolutions for Action Recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 6, pp. 1510–1517, 2018, doi: 10.1109/TPAMI.2017.2712608.

[6] S. D. Khan, S. Bandini, S. Basalamah, and G. Vizzari, “Analyzing crowd behavior in naturalistic conditions: Identifying sources and sinks and characterizing main flows,” Neurocomputing, vol. 177, pp. 543–563, 2016, doi: 10.1016/j.neucom.2015.11.049.

[7] Gupta, N., Vaisla, K. S., Jain, A., Kumar, A., & Kumar, R. (2021). Performance Analysis of AODV Routing for Wireless Sensor Network in FPGA Hardware. Computer Systems Science and Engineering, 39(2), 1-12.

[8] Kumar, S., Jain, A., Kumar Agarwal, A., Rani, S., & Ghimire, A. (2021). Object-Based Image Retrieval Using the U-Net-Based Neural Network. Computational Intelligence and Neuroscience, 2021.

[9] S. M. Mousavi, W. L. Ellsworth, W. Zhu, L. Y. Chuang, and G. C. Beroza, “Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking,” Nat. Commun., vol. 11, no. 1, pp. 1–12, 2020, doi: 10.1038/s41467-020-17591-w.

[10] S. Ramachandra, A. Hoelzemann, and K. Van Laerhoven, Transformer Networks for Data Augmentation of Human Physical Activity Recognition, vol. 1, no. 1. Association for Computing Machinery, 2021.

[11] Kumar, S., Jain, A., Shukla, A. P., Singh, S., Raja, R., Rani, S.& Masud, M. (2021). A Comparative Analysis of Machine Learning Algorithms for Detection of Organic and Nonorganic Cotton Diseases. Mathematical Problems in Engineering, 2021.

[12] H. G. Madanat and A. S. Khasawneh, “Impact of total quality management implementation on effectiveness of human resource management in the Jordanian banking sector from employees’ perspective,” Acad. Strateg. Manag. J., vol. 16, no. 1, pp. 114–148, 2017.

[13] A. Kumar, H. Hashmi, S. A. Khan and S. Kazim Naqvi, "SSE: A Smart Framework for Live Video Streaming based Alerting System," 2021 10th International Conference on System Modeling & Advancement in Research Trends (SMART), 2021, pp. 193-197, doi: 10.1109/SMART52563.2021.9675306.