Human Activity Recognition from Accelerometer Data

Assignment #3

Image Based Biometrics 2019/20, Faculty of Computer and Information Science, University of Ljubljana

Jernej Vivod
vivod.jernej@gmail.com

***Abstract*—Activity recognition is the problem of predicting the current activity of a person. It is typically presented as a supervised classification problem where the goal is to create a successful predictive model using a training set of labeled data. In this study, we evaluate the use of convolutional neural networks, long short-term memory neural networks and classification based on engineered features using a random forest model in predicting activities from raw accelerometer data. We build simple models for each of the competing classification methodologies and evaluate them on three freely available datasets using 5 runs of 10-fold cross-validation. We compute the average precision, recall and F-score for each method. We plot confusion matrices resulting from evaluating the models on withheld test sets.**

I. Introduction

Human activity recognition from accelerometer data represents an important research area with many applications, especially in the fields of healthcare and smart environments. Data from these widely available and inexpensive sensors can be used to monitor the physical activity of patients, detect falls as well as provide the clinicians with a stream of data that can be used to infer valuable information about the patient's state. Accelerometers can also be used for fitness tracking, theft protection as well as in many other applications in the emerging Internet of things ecosystems.

In this study, we aim to compare three different approaches to building a model capable of recognizing different activities solely using data obtained from worn accelerometers. We formulate the activity recognition problem as a supervised classification problem. We evaluate the performance of using either a convolutional neural network, a long short-term memory neural network or a baseline random forest model to perform the classifications using normalized and segmented raw accelerometer data. We also test the performance of using feature engineering in tandem with a random forest model to classify the activities.

II. Methodology

A. Used Datasets

We used three freely accessible datasets [1][2] to evaluate the performance of our models. The class distributions for the datasets are illustrated in figures 1, 2 and 3.

We can see that the representation of different activities in the datasets is imbalanced and warrants careful examination of model evaluation results as well as the use of methods designed to alleviate the issues arising from training a classification model on such datasets.

B. Data Preprocessing

The datasets are comprised of raw sensor data in the form of a time series. We segmented the data into overlapping segments of samples corresponding to three seconds of measurements. We set the overlap of the segments to 50%. Choosing a large overlap runs the risk of having very similar data examples in both the training and test sets which can lead to unrealistically high performance estimates. By not introducing overlaps, however, we severely reduce the number of available training and test examples. This can be especially problematic in the case of imbalanced datasets.

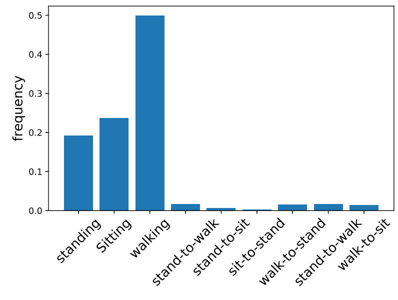

Figure 1. Class distribution for the first dataset.

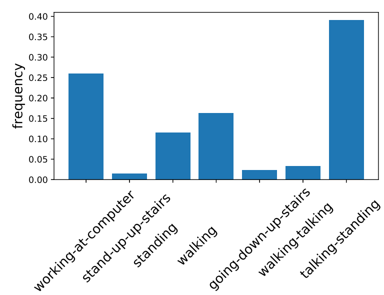

Figure 2. Class distribution for the second (Single chest mounted accelerometer) dataset.

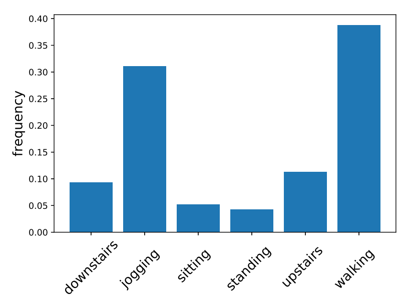

Figure 3. Class distribution for the third (Actitracker) dataset.

Figure 4 shows a visualization of time series data contained in a randomly selected three-second segment.

The activity corresponding to a particular data segment was determined as the mode of the labels of the time series data points spanned by the segment.

For use with the baseline random forest model, the segments were reshaped into a single row vector.

We used random oversampling to generate additional exam-

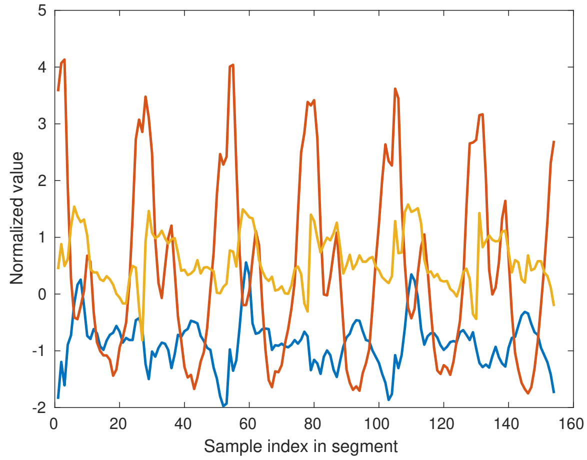

Figure 4. Visualization of data contained in a three-second segment corresponding to the activity of walking.

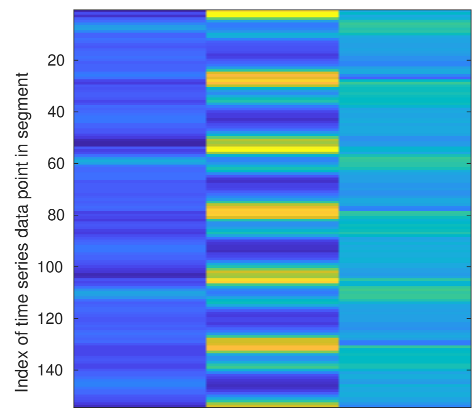

Figure 6. Visualization of a data segment corresponding to the activity of walking.

ples for minority classes for the training of classification models. Care was taken not to include synthetic examples in any of the test sets and only use them to balance the training set used to construct the models.

With random oversampling, we duplicate randomly chosen examples from the minority classes to increase their representation in the training set [3]. Here, the use of methods that generate unique synthetic training examples in the vicinity of existing ones for the minority classes can be problematic due to the high-dimensional nature of the examples. This can cause synthetic examples to be close to existing ones by the used distance metric but far by the relevant features.

C. The Convolutional Neural Network Model

Convolutional neural networks can learn to recognize patterns characteristic of certain activities without the need for explicitly crafting features describing the time series segment [4]. The architecture of a convolutional neural network tries to mimic the connectivity patterns and organization of the visual cortex.

We used the Keras deep learning framework [5] to implement a simple convolutional neural network. The architecture of the used model is shown in figure 5.

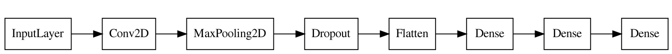

Figure 5. Architecture of the simple convolutional neural network model evaluated in our study.

The size of the input layer is set with respect to the number of samples and signals in the segment. Similarly, the size of the output layer is set with respect to the number of unique class values (number of different activities).

Figure 6 shows a visualization of an example of an image-like matrix input to the convolutional neural network corresponding to the activity of walking.

D. The Long Short-Test Memory Neural Network Model

Long short-term memory neural networks are especially suited for time series analysis as they are capable of learning long-term dependencies. The long short-term recurrent neural network tries to remember all the past knowledge that the network is seen so far and to forget irrelevant data. This is achieved by introducing different activation layers called gates for specific purposes [6].

We used the Keras deep learning framework [5] to implement a simple convolutional neural network. Figure 7 shows the simple LSTM neural network model used in this study.

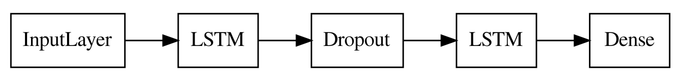

Figure 7. Architecture of the simple LSTM neural network model evaluated in our study.

The size of the input layer is again set with respect to the number of samples and signals in the segment, as is the output layer according to the number of unique class values (number of different activities).

E. Classification of Activities using Feature Engineering

A more traditional approach to activity recognition from raw acceleration sensor data is to first describe the segments of the time series with features describing the characteristics of that particular segment. This yields a vector of quantities characterizing each segment. These vectors can then be used as training data for a predictive model. Crafting relevant features from raw data is a non-trivial task that is subject to much research and often requires significant domain expertise [7].

We used various time and frequency domain features to characterize the segments and build a set of training and test examples to evaluate the performance of typical machine learning models on such a dataset.

Table I lists the features used to describe the sensor data segments.

Feature selection can be used to estimate the predictive power of features. We evaluated the use of several Relief-based algorithms to select a subset of most relevant features but this did not result in any noteworthy improvement in the performance of the random forest classifier [8][9].

Type of Feature	Feature
Time-domain	mean, standard deviation, variance, median, maximum, minimum, range, rms, argmin, argmax, energy, entropy, skewness, kurtosis, interquartile range, mean absolute deviation, occupied bandwidth, lower bound for occupied bandwidth, upper bound for occupied bandwidth, occupied band power.
Frequency-domain	band power, energy, mean, max, min

Table I
Features used to describe the data segments.

III. Results

In this section, we present the results of evaluating our methods. We tested each classification method described in section II using 5 runs of 10-fold cross-validation on each of the three datasets. Table II shows the cross-validation scores for each method (with random oversampling). The score for the baseline random forest model is 0.9361 on the first dataset, 0.8730 on the second (Single chest mounted accelerometer) dataset and 0.8441 on the third (Actitracker) dataset.

	Method		
Dataset	CNN	LSTM	Feature engineering + RF
1	0.9941	0.9897	0.9361
2	0.9752	0.9631	0.9177
3	0.9955	0.9811	0.9677

Table II
CV scores for methods evaluated on different datasets.

Figures 8, 9 and 10 show the normalized confusion matrices resulting from evaluating the methods using a train-test split on the first dataset.

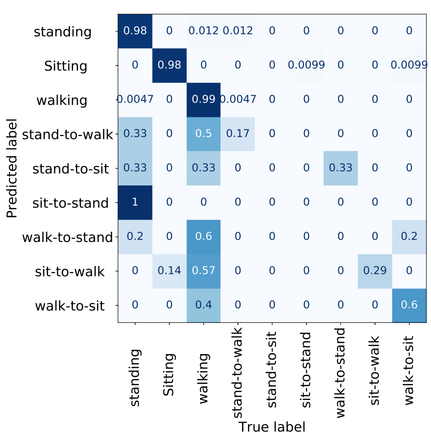

Figure 8. Normalized confusion matrix for the CNN model.

Tables III, IV and V show the mean precision, recall, F-score and support for each class value obtained by using each method on each dataset for all cross-validation folds.

IV. Conclusion

This study focuses on comparing three competing methods of classifying performed activities using only raw accelerometer data. All three studies methods outperform the baseline random forest classifier acting as a sanity check for the usefulness of the implemented methods and models. All methods exhibited

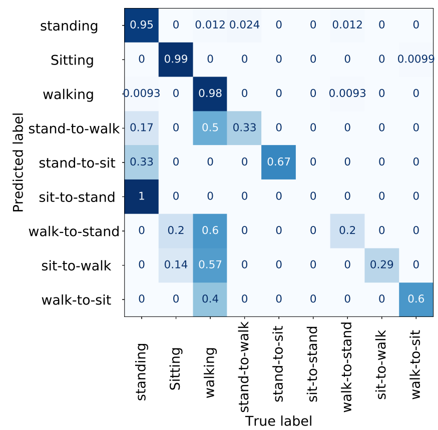

Figure 9. Normalized confusion matrix for the LSTM model.

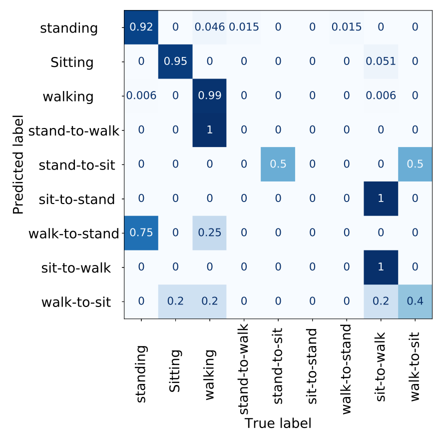

Figure 10. Normalized confusion matrix for the feature engineering method.

	Activity	
score	Standing	Sitting
Precision	0.9970, 0.9316, 0.9229	0.9877, 0.9916, 0.9750
Recall	0.9976, 0.7886, 0.9577	0.9990, 0.8395, 0.9817
F-score	0.9973, 0.8214, 0.9395	0.9933, 0.9040, 0.9781
	Walking	Stand-to-walk
Precision	0.9981, 0.9872, 0.9487	0.9680, 0.6400, 0.0400
Recall	0.9993, 0.9791, 0.9840	0.9720, 0.6400, 0.0200
F-score	0.9987, 0.9831, 0.9660	0.9693, 0.6400, 0.0260
	Walk-to-sit	Stand-to-sit
Precision	0.9533, 0.9333, 0.4100	0.5600, 0.4000, 0.0000
Recall	0.9400, 0.8333, 0.2500	0.5600, 0.4000, 0.0000
F-score	0.9427, 0.8667, 0.3033	0.5600, 0.4000, 0.0000
	Sit-to-stand	Walk-to-stand
Precision	0.9593, 0.3944, 0.4367	0.9810,0.7895, 0.6028
Recall	0.9100, 0.6500, 0.1917	0.9500,0.8800, 0.5083
F-score	0.9298, 0.4570, 0.2570	0.9627,0.8285, 0.5277
Sit-to-stand	Stand-to-walk	
Precision	0.9800, 0.5000, 0.6100	
Recall	0.9300, 0.8600, 0.5300	
F-score	0.9532, 0.5189, 0.5411	

Table III
Precision, recall, F-score and Support for the different methods evaluated on the first dataset. The leftmost value represents the result for the CNN model, the center value represents the result for the LSTM model and the rightmost value represents the result for the feature engineering method.

problems with the successful classification of underrepresented

	Activity	
score	Working-at-computer	stand-up-up-stairs
Precision	0.9967, 0.9907, 0.9806	0.8349, 0.8449, 0.8566
Recall	0.9741, 0.9854, 0.9867	0.9674, 0.9751, 0.4711
F-score	0.9851, 0.9812, 0.9836	0.8899, 0.8798, 0.5956
	standing	walking
Precision	0.9280, 0.9012, 0.8557	0.9886, 0.8886, 0.8972
Recall	0.9578, 0.9056, 0.7279	0.9824, 0.9153, 0.9663
F-score	0.9420, 0.9123, 0.7853	0.9854, 0.9019, 0.9301
	going-down-up-stairs	walking-talking
Precision	0.9340, 0.9040, 0.8873	0.9171, 0.9028, 0.7983
Recall	0.9836, 0.8812, 0.7612	0.9703, 0.9119, 0.5741
F-score	0.9565, 0.9011, 0.8157	0.9404, 0.8932, 0.6601
	talking-standing	
Precision	0.9900, 0.9800, 0.91	
Recall	0.9800, 0.9800, 0.96	
F-score	0.9854, 0.9612, 0.9376	

Table IV

Precision, recall, F-score and Support for the different methods evaluated on the second (Single chest mounted accelerometer) dataset. The leftmost value represents the result for the CNN model, the center value represents the result for the LSTM model and the rightmost value represents the result for the feature engineering method.

	Activity	
score	Going downstairs	Jogging
Precision	0.9866, 0.9746, 0.9224	0.9983, 0.9842, 0.9867
Recall	0.9890, 0.8830, 0.8858	0.9985, 0.9841, 0.9891
F-score	0.9877, 0.9772, 0.9035	0.9984, 0.9875, 0.9879
	Sitting	Standing
Precision	0.9987, 0.9965, 0.9933	0.9989, 0.9906, 0.9905
Recall	0.9995, 0.9841, 0.9833	0.9981, 0.9901, 0.9834
F-score	0.9991, 0.9872, 0.9883	0.9985, 0.9959, 0.9869
	Going upstairs	Walking
Precision	0.9877, 0.9356, 0.8966	1.0000, 0.9912, 0.9800
Recall	0.9858, 0.9438, 0.9043	1.0000, 0.9921, 0.9800
F-score	0.9865, 0.9461, 0.9003	0.9968, 0.9898, 0.9814

Table V

Precision, recall, F-score and Support for the different methods evaluated on the third (Actitracker) dataset. The leftmost value represents the result for the CNN model, the center value represents the result for the LSTM model and the rightmost value represents the result for the feature engineering method.

activities despite using random oversampling as a means of mitigating the imbalances in the datasets. Evaluating the methods using confusion matrices constructed from results on withheld test examples show that all three methods again outperformed the baseline random forest model. It would be worth estimating the methods on datasets with an equal number of training examples from all different classes and to thoroughly study the crafting of features that would best differentiate frequently misclassified activities. It should be noted that the features created to implement the feature engineering method did not include some of the features deemed by state of the art research to be the most informative. The convolutional neural network and long short-term memory neural network models did not include some state of the art extensions that were demonstrated to improve their performances such as the use of multi-headed models and other more advanced architectures. All code code used to perform this study can be found at https://github.com/Architecton/human-activity-recognition.

References

[1] A. Godfrey, A. Bourke, G. ÓLaighin, P. Ven, and J. Nelson, "Activity classification using a single chest mounted tri-axial accelerometer," *Medical engineering physics*, vol. 33, pp. 1127–35, 05 2011.

[2] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *SIGKDD Explor. Newsl.*, vol. 12, no. 2, p. 74–82, Mar. 2011. [Online]. Available: https://doi.org/10.1145/1964897.1964918

[3] P. Kaur and A. Gosain, *Comparing the Behavior of Oversampling and Undersampling Approach of Class Imbalance Learning by Combining Class Imbalance Problem with Noise*, 01 2018, pp. 23–30.

[4] A. Bevilacqua, K. MacDonald, A. Rangarej, V. Widjaya, B. Caulfield, and T. Kechadi, "Human activity recognition with convolutional neural networks," in *Machine Learning and Knowledge Discovery in Databases*, U. Brefeld, E. Curry, E. Daly, B. MacNamee, A. Marascu, F. Pinelli, M. Berlingerio, and N. Hurley, Eds. Cham: Springer International Publishing, 2019, pp. 541–552.

[5] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[6] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, June 2019.

[7] S. Preece, J. Goulermas, L. Kenney, and D. Howard, "A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data," *Biomedical Engineering, IEEE Transactions on*, vol. 56, pp. 871 – 879, 04 2009.

[8] J. Vivod, "Ocenjevanje atributov s posplošitvami algoritma relief," Undergraduate Thesis, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, Večna pot 113, Ljubljana, 2019.

[9] ——, "skrelief," https://github.com/Architecton/skrelief, 2020.