Comparison of Data Preprocessing Methods and the Impact on Auto-encoder's Performance in Activity Recognition Domain

Adam Harasimowicz
Webinterpret

Comparison of Data Preprocessing Methods and the Impact on Auto-encoder's Performance in Activity Recognition Domain

Adam Harasimowicz
Gdansk University of Technology
Faculty of Electronics, Telecommunications and Informatics
Department of Computer Architecture
11/12 Gabriela Narutowicza Street, 80-233 Gdansk, Poland
Email: haras.adam@gmail.com

***Abstract*—Raw data preprocessing and feature extraction have strong influence on a performance of Machine Learning classifiers and usually these steps are performed by experts who are able to decide which information are important. However there are algorithms, commonly used in Deep Learning, which allow automatically extract meaningful descriptions of samples. It motivated us to investigate the impact of different data preprocessing methods on auto-encoder (AE). We also involved one of most recent algorithm's version which outperforms standard AE – Sparse Auto-Encoder. The experiments base on accelerometer data gathered during previous researches and they contain measurements of eight activities. Presented comparison of methods allows to choose a subset of the most promising algorithms for Human Activity Recognition problems.**

***Keywords*—activity recognition; accelerometer; autoencoder; deep learning; pattern recognition; data preprocessing; ubiquitous computing; context awareness; sensors**

I. Introduction

Activity recognition domain is connected with such research areas like pattern recognition, ubiquitous computing and context awareness. The works in this problem domain started in the late '90s and one of the first experiments were [1] and [2] according to [3, 4]. Since that time researchers have improved results, found most appropriate sensors [5, 6], size of samples [7] and developed such Humans Activity Recognition (HAR) systems like DiaTrace [8], iLearn [9], UbiFit [5, 10] and others [11-15]. All of those systems based on Machine Learning approach and so far the most widely applied method for feature extraction in HAR problems is feature engineering which requires an expert knowledge to choose the most accurate features set. Mostly there are used features from two groups - time domain and frequency domain [4] which are computed independently for every sample containing measurements from sensors. One of such features can be mean, standard deviation, variance, interquartile range, mean absolute deviation, minimum or maximum, energy, entropy, kurtosis and correlation coefficients.

However, there are some methods commonly used in Deep Learning (DL) which allow automatically select meaningful representation of data [16, 17]. It is achieved during features learning process [18, 19]. Such process can be performed in unsupervised or semi-supervised way what significantly increases number of possible application domains. There also have been shown that this approach connected with Deep Learning methods allows achieve state-of-art or outperform it in such domains like Natural Language Processing, Computer Vision and Speech Recognition [18]. But the performance of feature learning algorithms (like auto-encoder, Restricted Boltzmann Machine, Sparse Coding and Deep Belief Network) depends on initial preprocessing and normalization raw data.

II. MOTIVATIONS

Deep machine learning is dynamically developed area by researchers and, as it was shown, it outperforms previously used methods in many problem domains [18, 20]. There are also works about DL in HAR but they base on video processing [21]. Nevertheless, according to our knowledge, this new approach was not adapted in HAR, which base on sensors, except [22]. Moreover, because data preprocessing has significant impact on classification performance we wanted provide guidance for future researches. Also since [22] Deep Learning algorithms have been improved so our work consider this aspect too.

Additionally we can observe rapid development in areas of wearable technology and Internet of Things what will raise huge amount of data and some of these information could be used to activity recognition. Also due to high data availability unsupervised or semi-supervised learning methods will have potentially advantage over pure supervised algorithms.

III. AUTOMATIC FEATURE LEARNING

Features learning process and automatic extraction can be achieved using different methods. But most of them base on the same concept – finding lower-dimensional representation of data without loss of important information. For the experiments we decided to choose Sparse Auto-Encoder, which belongs to the wider family of similar methods called autoencoders, because it allowed to achieve results comparable with state-of-the-art in other areas [18].

Basic auto-encoder, presented in *Fig. 1*, is a neural network with one hidden layer. The network has equal number of input and output nodes as well as noticeable lower number of hidden neurons (this requirement is not absolutely necessary as it was mentioned in [18]). Using such structure we train auto-encoder to give the same output as an input. Training is performed with standard methods – it means that we act feed-forwarding, then we compute an error (e.g. mean squared error) and finally we perform backpropagation algorithm. Such steps are made many times. At the end auto-encoder should be able to reconstruct input based on encoded values in hidden layer. The decoding step is possible if auto-encoder is able to find statistically the most meaningful features. After finished the training we can drop part related with decoding due to encode part is enough to transform data to lower-dimension representation. In sparse variant of auto-encoder we additionally trying keep number of activated neurons small. To achieve this we use extra penalty term during optimization so that to penalize deviating from sparsity parameter [23].

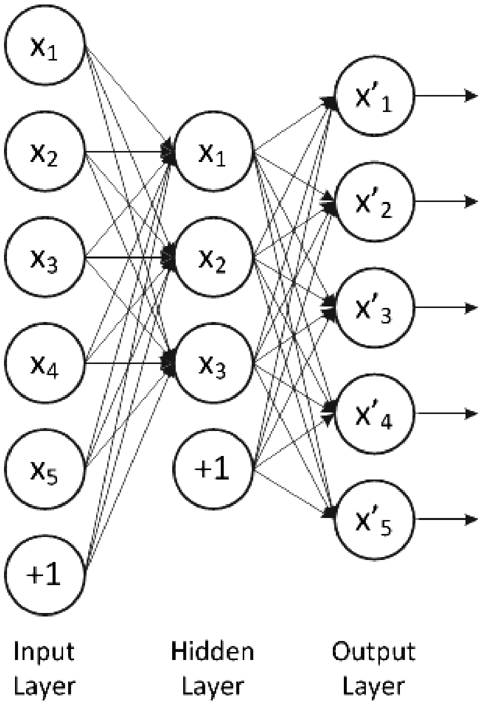

Figure 1 Simple Auto-encoder

IV. EXPERIMENTS

A. Data Set

To perform experiments we used data set gathered during previous researches [7, 24] which contains measurements from 3-axis accelerometer embedded in mobile phone. The data were collected during performing eight different activities: walking, running, sitting, lying, rotating, walking up stairs and down stairs and standing. In that time phone was stored in a front pocket of tester's trousers. Such device localization was chosen because of previous researches [25] which showed that this place is the most popular for carrying mobile phones. Also in gathering data were involved eight people – males and females and our testers had different age – from 12 to 53 years old.

The collected data were split into samples which contained measurements from two seconds periods - we chose such size because of previous work [7] which showed that it is enough length to achieved good classification accuracy. It means that one sample had 60 measurements – data were collected with frequency equal 10Hz and measurements were separated for each axis.

From the samples we formed two sets named as *light* and *full* due to long time of training for second case. The first set contains 23936 samples and the second one is around thirteen times bigger – 317951 samples.

B. Data Preprocessing

To experiments seven preprocessing methods were chosen. Also results for not preprocessed data were checked. Below we describe briefly each of those methods. Note that some of them, like PCA, also perform dimensionality reduction like auto-encoder, but the results could be different.

1) Standarization: Features are processed separately. We change data in such way that mean is equal zero and we have unit variance.

2) Scaling to a range: We scale all features that all of them lie between zero and one. Also for this method features are processed separately.

3) Normalization: We scale each sample separately in such way that individual samples have unit norm.

4) Principal Component Analysis (PCA): Based on Singular Value Decomposition linear dimensionality reduction of the data. In a result we get data projected to a lower dimensional space by the most significant singular vectors. Number of dimensions after reduction was chosen based on Minka's Maximum Likelihood Estimation (MLE).

5) PCA with whitening: The result is similar like for normal PCA. But because of whitening the features are less correlated with each other as well as all of them have the same variance. We also used Minka's MLE to choose number of dimensions.

6) Randomized PCA: Like PCA but it used approximated Singular Value Decomposition. Also instead choosing number of dimensions like for two previous methods we decided to reduce features space to twenty dimensions.

7) Randomized PCA with whitening: Like Randomized PCA with additional processing descirbed for PCA with whitening. Also in this case we reduced space to twenty dimensions.

C. Performing the Experiments

We compared all methods using *light* data set. For each of them we performed 5-fold cross validation to ensure that the results would be more stable. During each step of validation the data were proportionally divided into training and test sets in such way that proportions between samples belong to each class were kept. For each pair with algorithm and iteration of the cross validation the training and test sets were the same. For method which achieved the best result we made tests on the *full* data set with 6-fold cross validation.

The first step in our processing pipeline for each iteration was the data preprocessing in training set by one of the algorithms. Then modified samples were used to train Sparse Auto-Encoder. After that trained auto-encoder was used to transform training set and extract new feature vectors as an effect of unsupervised training. Then based on auto-encoder's output we trained k-Nearest Neighbors with *k* equal five. After fitting phase we performed tests – the test set was preprocessed with prepared in earlier phase algorithm. After that auto-encoder encoded data to obtain new feature vectors which were used as an input for already trained k-NN classifier.

In our experiments we used methods of preprocessing as well as k-Nearest Neighbors algorithm provided by Python's library – scikit-learn [26]. Conversely, we used our own implementation of Sparse Auto-Encoder with forty hidden neurons for all preprocessing methods except family of PCA algorithms. In that case we chose maximum from fifteen, half of an input units and number of the input units minus twenty.

V. RESULTS

In this section we present obtained results. For each of the methods we show two metrics – accuracy and precision. They are average values from all iterations of the cross validation. Also for the best one, which was checked, we present a confusion matrix which show what kind of activities were the most difficult to classify. The matrix contains cumulated values from all iterations of cross validation. As well as it contains results from tests performed on *full* data set.

According to the *Table I* the best results we achieved with normalization and for this method both metrics are above 99%. Also standardization, Principal Component Analysis and PCA with whitening provided accuracy and precision over 92%. In contrast, the worst results gave scaling to the range between zero and one which provided accuracy around 73.5% and precision close to 75%. What is interesting, even recognition based on raw data gave better results – both around 81%. Probably that situation was caused by dropping some significant information during preprocessing step with scaling method. Except scaling to the specific range all other methods allowed to improve performance of classification.

TABLE I. ACCURACY AND PRECISION FOR PREPROCESSING METHODS

Method	**Results**	
	Accuracy	***Precision***
Raw data	80.97%	80.93%
Standarization	96.09%	96.23%
Scaling to the range	73.43%	75.06%
Normalization	99.03%	99,04%
PCA	92.93%	92.83%
PCA with whitening	87.21%	87.54%
Randomized PCA	89.92%	89.72%
Randomized PCA with whitening	94.36%	94.4%

It can be surprise that such method like normalization provided the highest score, because in many domains preprocessing based on samples is worse than this one which bases on features. However, because values in feature vector can be completely different if we catch the measurements a little bit earlier or later, we should not strongly bind them with positions in vector and perform learning which bases on that. We should rather, during training, find a model which will be able to learn connections between features in vector. So it could explain why such preprocessing method like normalization provided the best results.

In the *Table II* we showed results – also precision and accuracy – for normalization. During six iterations of cross validation with using *full* data set the accuracy and the precision achieved the lowest result around 93% and the highest over 98%. An average scores for all steps are close to 96.6% and they are around 2.5 percentage points less than for *light* data set. Also the *Table III* contains the confusion matrix which shows that the most difficult to classify was walking down stairs with 72.32% of samples predicted correctly. The second activity, which was often misclassified, was walking up stairs – 87.77%, but that result was still 15 percentage points higher than the worst. For the rest of activities positive classification achieved level over 96.75% and four of them were over 99%.

TABLE II. ACCURACY AND PRECISION FOR NORMALIZATION WITH FULL DATA SET

Step	**Results**	
	Accuracy	***Precision***
Iteration 1	98,32%	98,33%
Iteration 2	94,82%	94,65%
Iteration 3	97,96%	97,97%
Iteration 4	97,95%	97,99%
Iteration 5	93,07%	92,87%
Iteration 6	97,74%	97,75%
Average	96,64%	96,59%

TABLE III. CUMULATIVE CONFUSION MATRIX FOR NORMALIZATION WITH FULL DATA SET

Activity	**Predictions**								
	Lying	***Rotating***	***Running***	***Sitting***	***Walking down stairs***	***Walking up stairs***	***Standing***	***Walking***	***True Positive***
Lying	134015	0	0	25	0	0	0	0	99,98%
Rotating	0	59292	0	16	1	1	1033	2	98,26%
Running	0	0	101170	0	28	19	0	756	99,21%
Sitting	167	658	3	231161	230	352	264	598	99,03%
Walking down stairs	0	1339	84	370	35173	6146	205	5316	72,32%
Walking up stairs	3	452	35	248	3159	55967	35	3863	87,77%
Standing	0	184	0	19	4	0	115478	19	99,80%
Walking	0	558	697	409	1865	2737	107	189590	96,75%

VI. CONCLUSION

As we showed auto-encoder algorithm can be adapted for Human Activity Recognition domain with his unsupervised learning process. Also according to obtained results preprocessing has strong influence on classification as well as on the AE. We showed that for our data set the best method was normalization and that result was a surprise for us because the algorithm bases on preprocessing of samples.

We also believe that using Stacked Auto-Encoder or Deep Believe Network instead auto-encoder connected with k-NN could improve presented results. This hypothesis bases on results obtained in other domains [18]. So as a future work these approaches can be checked.

REFERENCES

[1] O. X. Schlmilch, B. Witzschel, M. Cantor, E. Kahl, R. Mehmke, and C. Runge, “Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring,” Computers in Human Behavior, vol. 15, no. 5, pp. 571–583, 1999.

[2] F. Foerster, M. Smeja, and J. Fahrenberg, “Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring,” Computers in Human Behavior, vol. 15, no. 5, pp. 571–583, 1999.

[3] M.F.A. Abdullah, A.F.P. Negara, M.S. Sayeed, D. Choi and K.S. Muthu, “Classification Algorithms in Human Activity Recognition using Smartphones,” World Academy of Science, Engineering and Technology, vol. 68, pp. 422-430, 2012.

[4] O.D. Lara and M.A. Labrador, “A Survey on Human Activity Recognition using Wearable Sensors,” IEEE Communications Surveys & Tutorials,vol. 15, pp. 1192-1209, 2013.

[5] T. Choudhury, G. Borriello, S. Consolvo, D. Haehnel, B. Harrison, B. Hemingway, J. Hightower, P. Klasnja, K. Koscher, A. LaMarca, J.A. Landay, L. LeGrand, J. Lester, A. Rahimi, A. Rea and D. Wyatt, “The mobile sensing platform: An embedded activity recognition system,” IEEE Pervasive Computing, vol.7, pp. 32-41, 2008.

[6] Y. Wang, J. Lin, M. Annavaram, Q.A. Jacobson, J. Hong, B. Krishnamachari and N. Sadeh, “A framework of energy efficient mobile sensing for automatic user state recognition,” in Proceedings of the 7th international conference on Mobile systems, applications, and services, pp. 179-192, ACM, 2009.

[7] A. Harasimowicz, T. Dziubich and A. Brzeski, “Accelerometer-based Human Activity Recognition and the Impact of the Sample Size,” Advances in Neural Networks, Fuzzy Systems and Artificial Intelligence, pp. 130-135, WSEAS, 2014.

[8] G. Bieber, J. Voskamp and B. Urban, “Activity recognition for everyday life on mobile phones,” Proceedings of the 5th International Conference on Universal Access in Human-Computer Interaction. Part II: Intelligent and Ubiquitous Interaction Environments, pp. 289-296, Springer-Verlag, Berlin, Heidelberg, 2009.

[9] T. Saponas, J. Lester, J. Froehlich, J. Fogarty, J. Landay, “iLearn on the iPhone: Real-time human activity classification on commodity mobile phones,” in University of Washington CSE Tech Report UW-CSE-08-04-02, 2008.

[10] S. Consolvo, D.W. McDonald, T. Toscos, M.Y. Chen, J. Froehlich, B. Harrison, P. Klasnja, A. LaMarca, L. LeGrand, R. Libby, I. Smith, and J.A. Landay, “Activity sensing in the wild: a field trial of Ubifit Garden,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1797–1806, ACM, New York, NY, USA, 2008.

[11] J. Baek, S. Kim, H. Kim, J. Cho and B. Yun, “Recognition of User Activity for User Interface on a Mobile Device,” in Proceedings of the 24th South East Asia Regional Computer Conference, Thailand, 2007.

[12] G. Bieber, A. Luthardt, C. Peter and B. Urban, “The hearing trousers pocket: activity recognition by alternative sensors,” in Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments, pp. 44, ACM, New York, NY, USA, 2011.

[13] N. Ravi, N. Dandekar, P. Mysore and M.L. Littman, “Activity recognition from accelerometer data,” in Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence (IAAI), pp. 1541-1546, AAAI Press, 2005.

[14] J.R. Kwapisz, G.M. Weiss and S.A. Moore, “Activity recognition using cell phone accelerometers,” in Proceedings of the Fourth International Workshop on Knowledge Discovery from Sensor Data, pp. 10-18, 2010.

[15] S. Dernbach, B. Das, N.C. Krishnan, B.L. Thomas and D.J. Cook, Simple and complex activity recognition through smart phones,” in Proceedings of the 8th International Conference on Intelligent Environments, pp. 214-221, IEEE, 2012.

[16] G.E. Hinton, R.R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” Science, vol. 313, pp. 504-507, American Association for the Advancement of Science, 2006.

[17] Y. Bengio and Y. LeCun, “Scaling learning algorithms towards AI,” in Large-Scale Kernel Machines, vol. 34, 1-41, 2007.

[18] Y. Bengio, A. Courville, P. Vincent, “Representation learning: A review and new perspectives,” IEEE, 2013.

[19] Y. Bengio, “Learning deep architectures for AI,” Foundations and trends in Machine Learning, vol.2, pp. 1-127, Now Publishers Inc., 2009.

[20] I. Arel, D.C.Rose, T.P. Karnowski, “Deep machine learning-a new frontier in artificial intelligence research,” in Computational Intelligence Magazine, vol. 5, pp. 13-18, IEEE, 2010.

[21] G.W. Taylor, R. Fergus, Y. LeCun and C. Bregler, “Convolutional learning of spatio-temporal features,” Computer Vision-ECCV 2010, pp. 140-153, Springer, 2010.

[22] T. Plötz, N.Y. Hammerla and P. Olivier, “Feature learning for activity recognition in ubiquitous computing,” in Proceedings of the Twenty-Second international joint conference on Artificial Intelligence, vol. 2, pp. 1729-1734, AAAI Press, 2011.

[23] R.A. Ranzato, Y. Boureau and Y. LeCun, “Sparse feature learning for deep belief networks,” in Advances in neural information processing systems, pp. 1185-1192, 2008.

[24] A. Harasimowicz, “Recognition of user behaviors based on sensors measurements from the mobile device,” master thesis, ETI, Gdansk, 2013, unpublished.

[25] Y. Cui, J. Chipchase and F. Ichikawa, “A cross culture study on phone carrying and physical personalization,” in Usability and Internationalization. HCI and Culture, pp. 483-492, Springer, 2007.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A.Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, “Scikit-learn: Machine Learning in Python,” in Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.